Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 17, no. 3, pp. 245–264 (2013)
DOI: 10.7155/jgaa.00293

Maximal Neighborhood Search and Rigid Interval Graphs

Peng Li[1] *Yaokun Wu*[1,2]

[1]Department of Mathematics, and MOE-LSC
Shanghai Jiao Tong University, Shanghai 200240, China

Abstract

A rigid interval graph is an interval graph which has only one clique tree. In 2009, Panda and Das show that all connected unit interval graphs are rigid interval graphs. Generalizing the two classic graph search algorithms, Lexicographic Breadth-First Search (LBFS) and Maximum Cardinality Search (MCS), Corneil and Krueger propose in 2008 the so-called Maximal Neighborhood Search (MNS) and show that one sweep of MNS is enough to recognize chordal graphs. We develop the MNS properties of rigid interval graphs and characterize this graph class in several different ways. This allows us obtain several linear time multi-sweep MNS algorithms for recognizing rigid interval graphs and unit interval graphs, generalizing a corresponding 3-sweep LBFS algorithm for unit interval graph recognition designed by Corneil in 2004. For unit interval graphs, we even present a new linear time 2-sweep MNS certifying recognition algorithm.

Submitted:	Reviewed:	Revised:	Accepted:	Final:
September 2012	April 2013	May 2013	May 2013	May 2013
		Published: May 2013		
	Article type: Regular paper		Communicated by: G. Liotta	

Research supported by the National Natural Science Foundation of China (No. 11271255). We thank Derek Corneil, Wen-Lian Hsu, Jing Huang and Ziqing Xiang for various help during the preparation of this paper. The authors also wish to thank the anonymous referees for constructive comments.

[2] *Corresponding author*
E-mail addresses: lidetiansjtu@sjtu.edu.cn (Peng Li) ykwu@sjtu.edu.cn (Yaokun Wu)

1 Introduction

For any nonnegative integers n and m we use the notation $[n]$ for the set of first n positive integers and write $\langle m+1, n\rangle$ for $[n] \setminus [m]$. For any set S of cardinality n, an *ordering* of S is just a bijection from $[n]$ to S and two orderings σ and σ' of S are said to be the *reversal* of each other if $\sigma_i = \sigma'_{n+1-i}$ for every $i \in [n]$. Let G be a graph on n vertices. A *selection rule* $\mathfrak{S}$ is a map from $2^{V(G)} \setminus \{V(G)\}$ to $2^{V(G)} \setminus \{\emptyset\}$ such that $T \cap \mathfrak{S}(T) = \emptyset$ for any $T \subsetneq V(G)$. A *graph search algorithm* with a selection rule $\mathfrak{S}$ determines an ordering τ of $V(G)$ by selecting $\tau_1 \in \mathfrak{S}(\emptyset)$ and then going on inductively to pick τ_{i+1} from $\mathfrak{S}(\{\tau_1, \ldots, \tau_i\})$ for every $i \in [n-1]$. We call $\mathfrak{S}(\{\tau_1, \ldots, \tau_i\})$ a *slice at time* $i+1$ in the process of generating the ordering τ with the selection rule $\mathfrak{S}$, or merely a slice of τ, and denote it by the simplified notation $S_\tau(i+1)$, hoping that the selection rule is clear from context. Many different orderings may be generated by the same selection rule as there are many ways of breaking ties by choosing one element from a given slice. If $\mathfrak{S}'$ is a new selection rule satisfying $\mathfrak{S}'(T) \subseteq \mathfrak{S}(T)$ for any $T \in 2^{V(G)} \setminus \{V(G)\}$, we may think that we are introducing additional rule of breaking ties into the original search algorithm corresponding to $\mathfrak{S}$ and then restrict the output vertex orderings to a smaller range. For any graph search algorithm $\mathcal{A}$ with a selection rule $\mathfrak{S}$, any graph G and $u \in V(G)$ satisfying $u \in \mathfrak{S}(\emptyset)$, we write $\mathcal{A}(G, u)$ for the search algorithm applied to G with a selection rule $\mathfrak{S}'$ such that $\mathfrak{S}'(\emptyset) = \{u\}$ and $\mathfrak{S}' = \mathfrak{S}$ elsewhere. Each vertex ordering produced by $\mathcal{A}$ on a graph G will be called an $\mathcal{A}$ *ordering* of G and the last vertex of an $\mathcal{A}$ ordering is referred to as an $\mathcal{A}$ *end-vertex* of G.

A *multi-sweep graph search algorithm* generates several orderings $\tau_1, \tau_2, \ldots$ of the vertex set of a graph in turn with selection rules $\mathfrak{S}_1, \mathfrak{S}_2, \ldots$ in each sweep respectively. Usually, the rule $\mathfrak{S}_i$ will rely on the previous orderings $\tau_1, \ldots, \tau_{i-1}$. It is expected that these orderings will be better and better in some sense and the final ordering will provide us what we want, say a good certificate for either the membership or the nonmembership of the graph in certain graph class.

The *Maximal Neighborhood Search* (`MNS`) [7] is the graph search algorithm that at each step always picks a vertex whose set of neighbors already explored is maximal with respect to set inclusion. A graph search algorithm is an `MNS` *type algorithm* if all possible output vertex orderings of the algorithm are `MNS` orderings. Both `LBFS` [10, 23] and `MCS` [28, 29] are `MNS` type algorithms and have simple linear time implementations. Another quite useful and easily implementable `MNS` type algorithm is the so-called `LDFS` algorithm [7], which can run with a log factor off linear [14] (a more complicated version has a log log factor [5, 26]). A special implementation of `MCS`, called `LMCS`, is also an `MNS` type algorithm and is proposed to study chordal powers of graphs [3]. It worths mentioning that `LBFS` is a special breadth-first search while `MNS` may not be any breadth-first search.

Generalizing the idea of `LBFS+` [17, 24], for any graph search algorithm $\mathcal{A}$ with selection rule $\mathfrak{S}$, any graph G and any ordering τ of $V(G)$, let us propose here the graph search algorithm $\mathcal{A}+(G, \tau)$, which produces an ordering σ of $V(G)$ such that for each $i \in [|V(G)|]$, after $\sigma_1, \ldots, \sigma_{i-1}$ are determined, σ_i is chosen to be the vertex in $\mathfrak{S}(\{\sigma_1, \ldots, \sigma_{i-1}\})$ that appears last in τ. We mention that LDFS+ can be implemented in the same time as LDFS [5]. Additionally, LBFS+ can be executed in linear time as with LBFS itself [4, 10].

For any graph search algorithm $\mathcal{A}$ with selection rule $\mathfrak{S}$, given any graph G on n vertices and any $u \in V(G)$, the algorithm $\mathcal{A}^{\triangle}(G, u)$ visits the graph according to an ordering τ such that $\tau_1 = u$ and for any $i \in [n-1]$, after $\tau_1, \ldots, \tau_i$ are determined, τ_{i+1} is a vertex in $\mathfrak{S}(\{\tau_1, \ldots, \tau_i\})$ whose closed neighborhood is minimal with respect to set inclusion. Let us explain briefly that $\texttt{LBFS}^{\triangle}(G, u)$ has a simple linear time implementation. We first construct in linear time an ordering ρ of $V(G)$ such that $\rho_n = u$ and $\deg(\rho_1) \geq \cdots \geq \deg(\rho_{n-1})$. It is easy to see that we can then use any efficient execution of LBFS+(G, ρ) as a way of implementing $\texttt{LBFS}^{\triangle}(G, u)$.

An *interval representation* $\mathcal{I}$ of a graph G maps each $v \in V(G)$ to an interval of reals $\mathcal{I}(v) = [\ell_{\mathcal{I}}(v), r_{\mathcal{I}}(v)] \neq \emptyset$ such that $vw \in E(G)$ if and only if $v \neq w$ and $\mathcal{I}(v) \cap \mathcal{I}(w) \neq \emptyset$. If $r_{\mathcal{I}} - \ell_{\mathcal{I}}$ takes the constant value 1, the interval representation $\mathcal{I}$ is named a *unit interval representation.* A graph admitting an interval representation and a unit interval representation, respectively, is called an *interval graph* and a *unit interval graph* (UIG). *Chordal graphs* are the intersection graphs of sets of subtrees of a common host tree, which are also those graphs without chordless cycles of length at least four. It is apparent that $\{$unit interval graphs$\} \subseteq \{$interval graphs$\} \subseteq \{$chordal graphs$\}$.

Let G be a graph on n vertices and let τ be an ordering of $V(G)$. We use $N_G[v]$ and $N_G(v)$ for the closed neighborhood and the open neighborhood of v in G, respectively. A vertex v of G is *simplicial* if $N_G[v]$ is a clique. A vertex v of G is *admissible* if there are no vertices x and y such that there is an x, v-path avoiding $N_G[y]$ and there is an y, v-path avoiding $N_G[x]$. The set of vertices of G which are both simplicial and admissible is denoted by $AS(G)$. For any $j, k \in [n]$, we define $N_{G,\tau}[j] = \{i \in [n] : \tau_i \in N_G[\tau_j]\}$, $N^{\geq k}_{G,\tau}[j] = N_{G,\tau}[j] \setminus [k-1]$ and $N^{\leq k}_{G,\tau}[j] = N_{G,\tau}[j] \cap [k]$. For each $j \in [n]$, set $\ell_{G,\tau}(j) = \min\{i : i \in N_{G,\tau}[j]\}$ and $r_{G,\tau}(j) = \max\{i : i \in N_{G,\tau}[j]\}$. If $\tau_j\tau_{j+1} \in E(G)$ for every $j \in [n-1]$, then τ is a *consecutive* ordering of G. We call τ an *I-ordering* of G if $\langle j, r_{G,\tau}(j)\rangle \subseteq N_{G,\tau}[j]$ for every $j \in [n]$ [8] and we call τ a *UI-ordering* of G if $\langle \ell_{G,\tau}(j), r_{G,\tau}(j)\rangle = N_{G,\tau}[j]$ for every $j \in [n]$ [4]. The ordering τ is a *perfect elimination ordering* (PEO) provided τ_i is simplicial in $G[\tau_1, \ldots, \tau_i]$ for each $i \in [n]$. If τ is an MNS ordering and each of its MNS slices other than $S_\tau(1)$ is a clique, we call τ a *perfect slice ordering* (PSO) of G. We refer to τ as an *RI-ordering* of G provided it is both a consecutive I-ordering and a PSO. The set of all maximal cliques of G is denoted by $\mathscr{C}(G)$. A *clique tree* T of G is a tree with $\mathscr{C}(G)$ as its vertex set such that it holds for every $v \in V(G)$ that the nodes of T containing v induce a subtree of T. A clique tree which is a path is called a *clique path.* A *rigid chordal graph* is a graph with a unique clique tree and a rigid chordal graph is a *rigid interval graph* (RIG) if it has a clique path.

Theorem 1 *[7, §2.6] The* MNS *end-vertices of a chordal graph are simplicial.*

Theorem 2 *For any rigid interval graph G, $AS(G)$ coincides with its set of* `MNS` *end-vertices.*

Theorem 3 *Let G be a graph having a clique tree T. Then G is a rigid chordal graph if and only if we cannot find three different maximal cliques C_i, C_j and C_k such that*

$$\{C_iC_j,\ C_jC_k\} \subseteq E(T) \text{ and } (C_i \cap C_j) \setminus C_k = \emptyset. \tag{1}$$

Let G be a graph and let μ be an ordering of $\mathscr{C}(G)$. For each $v \in V(G)$, define $\ell_\mu(v) = \min_{v \in \mu_i} i$ and $r_\mu(v) = \max_{v \in \mu_i} i$. For any $v, w \in V(G)$, we write $v \prec_\mu w$ provided $\ell_\mu(v) < \ell_\mu(w)$ or $\ell_\mu(v) = \ell_\mu(w),\ r_\mu(v) < r_\mu(w)$. An ordering τ of $V(G)$ is *compatible* with an ordering μ of $\mathscr{C}(G)$ if τ is a linear extension of the partial order $\prec_\mu$. We say that an ordering τ of $V(G)$ is *left-compatible* with the ordering μ of $\mathscr{C}(G)$ provided $\ell_\mu(\tau_1) \leq \cdots \leq \ell_\mu(\tau_{|V(G)|})$.

Theorem 4 *Let G be a graph with m maximal cliques. (i) If a path $[\mu_1, \ldots, \mu_m]$ is the unique clique tree of G, then any ordering τ of $V(G)$ which is compatible with μ must be an RI-ordering. (ii) If G has an RI-ordering τ, then G is a rigid interval graph and has a clique path $[\mu_1, \ldots, \mu_m]$ such that $\tau_1 \in \mu_1 \setminus \mu_2$.*

Theorem 5 *[22] A graph is a unit interval graph if and only if it has a UI-ordering.*

As can be seen from Theorems 3, 4 and 5, rigid interval graphs include connected unit interval graphs [20, Lemma 2.9] and those chordal graphs having at most two maximal cliques that contain simplicial vertices [1, Theorem 4.3]. For later reference, we list below the result on the unit interval graphs. It is noteworthy that connected unit interval graphs are also characterized by the unique existence of so-called straight enumerations [9, Corollary 2.5].

Theorem 6 *[20, Lemma 2.9] A connected unit interval graph must be a rigid interval graph.*

A graph G is *prime* provided that any subset S of $V(G)$ satisfying $1 < |S| < |V(G)|$ contains two vertices u, v such that $N_G(u) \setminus S \neq N_G(v) \setminus S$. Hsu [12, Theorem 4.1] observes that every prime interval graph has a unique maximal clique arrangement in certain sense.

Example 7 *Fig. 1(a) is a rigid chordal graph but not any rigid interval graph. Fig. 1(b) is not any rigid interval graph; its vertex ordering* $1, 2, 3, 4, 5$ *is both a PSO and an I-ordering but not consecutive. Fig. 1(c) is a rigid interval graph but not a unit interval graph. Fig. 1(d) is a connected unit interval graph with a UI-ordering* $1, 2, 3, 4, 5$*; Its vertex ordering* $1, 3, 2, 4, 5$ *is an RI-ordering but not any UI-ordering. Fig. 1(e) is a prime interval graph, has a unique clique path, but has totally three clique trees.*

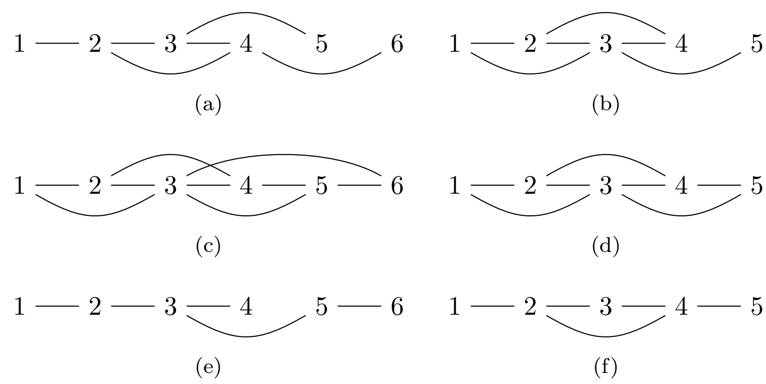

Figure 1: Six graphs.

In 2004, Corneil [4] presents a 3-sweep `LBFS` algorithm for the recognition of unit interval graphs and argues that it is the most easily implementable unit interval graph recognition algorithm thus known. We refer to [27, p. 194] for some interesting comments on what is a simple algorithm. Following the idea of this 3-sweep `LBFS` algorithm, we propose here two simple multi-sweep `MNS` algorithms, `2MNS-UI/RI`$^{\mathcal{A}}$ and `3MNS-UI/RI`$^{\mathcal{A}}$, for recognizing unit/rigid interval graphs based on an `MNS` type algorithm $\mathcal{A}$.

2MNS-UI/RI$^{\mathcal{A}}(G)$
▷ **Input** a connected graph G on n vertices
▷ **Output** a statement declaring whether or not G is a unit/rigid
▷ interval graph
Generate an MNS ordering σ of $V(G)$;
Do $\mathcal{A}^{\triangle}(G, \sigma_n)$ to yield a sweep τ;
if τ is a UI-ordering/RI-ordering of G
then conclude that G is a unit/rigid interval graph;
else conclude that G is not any unit/rigid interval graph.

3MNS-UI/RI$^{\mathcal{A}}(G)$
▷ **Input** a connected graph G on n vertices
▷ **Output** a statement declaring whether or not G is a unit/rigid
▷ interval graph
Generate an MNS ordering δ of $V(G)$;
Do $\mathcal{A}(G, \delta_n)$ yielding a sweep σ;
Do $\mathcal{A}+(G, \sigma)$ yielding a sweep τ;
if τ is a UI-ordering/RI-ordering of G
then conclude that G is a unit/rigid interval graph;
else conclude that G is not any unit/rigid interval graph.

Theorem 8 *The algorithm* 2MNS-UI/RI$^{\mathcal{A}}$ *outputs a UI-ordering when the input is a connected unit interval graph and outputs an RI-ordering when the input is a rigid interval graph.*

Theorem 9 *The algorithm* 3MNS-UI/RI$^{\mathcal{A}}$ *outputs a UI-ordering when the input is a connected unit interval graph and outputs an RI-ordering when the input is a rigid interval graph.*

A connection between 2MNS-UI/RI$^{\mathcal{A}}$ and 3MNS-UI/RI$^{\mathcal{A}}$ is illustrated in the next result.

Theorem 10 *Let G be a rigid interval graph on n vertices and let δ, σ, and τ be the three orderings of G generated by* 3MNS-UI/RI$^{\mathcal{A}}(G)$ *based on an* MNS *type algorithm $\mathcal{A}$. Then τ is an output of the algorithm $\mathcal{A}^{\triangle}(G, \sigma_n)$.*

Here is one more characterization of rigid interval graphs in terms of MNS properties, the proof of which indeed suggests a simple 4-sweep MNS algorithm for recognizing rigid interval graphs.

Theorem 11 *Let G be a graph on n vertices. Then G is a rigid interval graph if and only if it has two consecutive orderings τ and ρ such that τ is an I-ordering, ρ is an* MNS *ordering and $\rho_1 = \tau_n$.*

A *certifying algorithm* [18] is one whose output contains not only an answer of acceptance/rejection but also a certificate for the user to easily convince himself that the particular output is correct or something buggy happens in the implementation or design of the algorithm. Meister proposes a special breadth-first search, called min-LexBFS, and designs a certifying recognition algorithm for unit interval graph which basically consists of three sweeps of min-LexBFS [19, Theorem 16]. Note that min-LexBFS is incomparable with LBFS and even incomparable with MNS. Hell and Huang [11] modify Corneil's 3-sweep LBFS algorithm for recognizing unit interval graphs into a certifying algorithm. With an additional sweep to search for a certificate of membership/nonmenbership following the output ordering of the second MNS sweep, we can even turn 2MNS-UI$^{\mathcal{A}}$ into a certifying algorithm, called T-MNS$^{\mathcal{A}}$, for recognizing unit interval graphs. It may worth pointing out that the nonmembership certificate of our certifying algorithm is the same with that of the algorithm of Meister [19] and is different with that used in the algorithm of Hell and Huang [11].

To prepare for our discussion of T-MNS$^{\mathcal{A}}$ in Section 2, we need some more concepts. An *asteroidal triple* (AT) is a set of three vertices such that each pair of vertices is joined by a path that avoids the closed neighborhood of the third. Let G be a graph and let a, b, c, d be four vertices in G. We say that $\{a, b, c, d\}$ is a *claw centered at* a in G provided b, c and d are three independent vertices from the neighbors of a in G.

Theorem 12 *[15, 21] A graph is an interval graph if and only if it is AT-free and chordal; An interval graph is a unit interval graph if and only if it is claw-free.*

Theorem 13 *Let G be a connected graph without any chordless 4-cycle. Let u be both a simplicial vertex and an* MNS *end-vertex of G. Let $\mathcal{A}$ be an* MNS *type algorithm and let τ be an ordering produced by $\mathcal{A}^{\triangle}(G, u)$. Then either τ is a UI-ordering of G or G contains either a claw or an AT.*

In Section 2, we prove Theorem 13, introduce $\texttt{T-MNS}^{\mathcal{A}}$ and demonstrate its correctedness and implementation. In Section 3, we prove Theorems 2, 3, 4, 8, 9, 10 and 11 and then demonstrate the correctedness and implementation of $\texttt{2MNS-UI/RI}^{\mathcal{A}}$ and $\texttt{3MNS-UI/RI}^{\mathcal{A}}$. Finally, we conclude the paper with some remarks in Section 4.

2 Certifying algorithm for recognizing UIG

Lemma 1 *Let G be a graph on n vertices and let $i \in [n]$. Let τ be an* MNS *ordering of G such that it holds for every $j \in [i-1]$ that*

$$N^{\geq j+1}_{G,\tau}[j] \subseteq N_{G,\tau}[j+1]. \tag{2}$$

Then the following statements are true: (i) $N^{\leq \min\{r,t\}}_{G,\tau}[t] = \langle \ell_{G,\tau}(t), \min\{r, t\}\rangle$ for every $r \in [i]$ and $t \in [n]$; (ii) $\ell_{G,\tau}(j) \leq \ell_{G,\tau}(k)$ for every $j, k \in [n]$ such that $j \leq i+1$ and $j < k$; (iii) If $i = n$, then τ is a UI-ordering of G.

Proof: To prove (i), it suffices to verify it for $r = i$. As a consequence of Eq. (2), for $1 \leq j \leq j+s \leq i$ we have

$$N^{\geq j+s}_{G,\tau}[j] \subseteq N^{\geq j+s}_{G,\tau}[j+s] \tag{3}$$

If $\ell_{G,\tau}(t) \leq i$, claim (i) follows from Eq. (3) by putting $j = \ell_{G,\tau}(t)$ and $j+s = \min\{i, t\}$. If $\ell_{G,\tau}(t) > i$, we have $N^{\leq \min\{i,t\}}_{G,\tau}[t] = \emptyset = \langle \ell_{G,\tau}(t), \min\{i, t\}\rangle$ and so claim (i) still follows.

If $j = 1$, then $\ell_{G,\tau}(j) = 1 \leq \ell_{G,\tau}(k)$ and thus claim (ii) is trivial. Assume then $j > 1$. Since $j-1 \leq i$, an application of (i) gives $N^{\leq \min\{j-1,j\}}_{G,\tau}[j] = \langle \ell_{G,\tau}(j), \min\{j-1, j\}\rangle = \langle \ell_{G,\tau}(j), j-1\rangle$ and $N^{\leq \min\{j-1,k\}}_{G,\tau}[k] = \langle \ell_{G,\tau}(k), \min\{j-1, k\}\rangle = \langle \ell_{G,\tau}(k), j-1\rangle$. By the rule of MNS, $N^{\leq \min\{j-1,j\}}_{G,\tau}[j] \subsetneq N^{\leq \min\{j-1,k\}}_{G,\tau}[k]$ cannot happen and so $\ell_{G,\tau}(j) \leq \ell_{G,\tau}(k)$ follows, finishing the proof of (ii).

To deduce (iii), we pick arbitrarily $k \in [n]$ and intend to show $N_{G,\tau}[k] = \langle \ell_{G,\tau}(k), r_{G,\tau}(k)\rangle$. It follows from (i) that $N^{\leq k}_{G,\tau}[k] = \langle \ell_{G,\tau}(k), k\rangle$. It thus remains to verify $\tau(j')\tau(k) \in E(G)$ for each $j' \in \langle k+1,\ r_{G,\tau}(k)\rangle$. But (ii) implies $\ell_{G,\tau}(j') \leq \ell_{G,\tau}(r_{G,\tau}(k)) \leq k$. Accordingly, by (i) again, $k \in \langle \ell_{G,\tau}(j'), k\rangle = N^{\leq \min\{k,j'\}}_{G,\tau}[j']$, completing the proof. □

Corollary 14 *A graph G on n vertices is a unit interval graph if and only if it has an* MNS *ordering τ such that Eq. (2) is fulfilled for all $j \in [n-1]$.*

Proof: The backward direction is a consequence of Theorem 5 and Lemma 1. We now address the other direction. Let the graph G have a unit interval representation $\mathcal{I}$ and let τ be an ordering of $V(G)$ such that $\ell_{\mathcal{I}}(\tau_1) \leq \cdots \leq \ell_{\mathcal{I}}(\tau_n)$. It is easy to see that τ is what we want, finishing the proof. □

Lemma 2 *Let G be a connected graph on n vertices and let σ be an* `MNS` *ordering of G. If $G[V(G) \setminus N_G[\sigma_n]]$ has at least two different connected components, say A and B, then there exist $a \in A$, $b \in B$ and $c \in N_G(\sigma_n)$ such that $\{a, b, c, \sigma_n\}$ is a claw in G centered at c.*

Proof: Set $p = \min\{i : \sigma_i \in A\}$ and $q = \min\{i : \sigma_i \in B\}$. Without loss of generality, assume that $p < q$. Observe that $N_G[\sigma_q] \subseteq N_G(\sigma_n) \cup B$ and hence, as σ_q is the earliest vertex in B visited by σ, we conclude that $N^{\leq q-1}_{G,\sigma}[n] \supseteq N^{\leq q-1}_{G,\sigma}[q]$. Considering the fact that $n > q$, the rule of `MNS` then enables us assert that

$$N^{\leq q-1}_{G,\sigma}[n] = N^{\leq q-1}_{G,\sigma}[q]. \tag{4}$$

Furthermore, the connectedness of G and the rule of `MNS` ensure that there is a path in $G[\sigma_1, \ldots, \sigma_q]$ connecting $\sigma_p \in A$ and $\sigma_q \in B$, which shows that there are $r \in [q-1]$ and $s \in N^{\leq q-1}_{G,\sigma}[n]$ such that $\sigma_r \in N_G(\sigma_s) \cap A$. Subsequently, according to Eq. (4), we can check that the four vertices $\sigma_n, a = \sigma_r$, $b = \sigma_q$, and $c = \sigma_s$ form a desired claw in G, finishing the proof. □

Proof of Theorem 13: Let $n = |V(G)|$.

Case 1. It holds $N^{\geq i+1}_{G,\tau}[i] \subseteq N_{G,\tau}[i+1]$ for all $i \in [n-1]$. We conclude from Lemma 1 (iii) that τ is a UI-ordering of G.

Case 2. We can find a minimum $i \in [n-1]$ such that $N^{\geq i+1}_{G,\tau}[i] \nsubseteq N_{G,\tau}[i+1]$. By Lemma 1 (ii), it holds $p \leq q$ where $p = \ell_{G,\tau}(i)$ and $q = \ell_{G,\tau}(i+1)$. Take $s \in N^{\geq i+1}_{G,\tau}[i] \setminus N_{G,\tau}[i+1]$. Lemma 1 (i) says $N^{\leq i}_{G,\tau}[i+1] = \langle q, i \rangle$ while the connectedness of G gives $N^{\leq i}_{G,\tau}[i+1] \neq \emptyset$. This then leads to $\tau_i\tau_{i+1} \in E(G)$. Since $u = \tau_1$ is simplicial, the `MNS` rule forces $i > 1$ and so we have $p < i$; it is obvious that $q < i+1$. Let us record what we know about $E(G)$ now here:

$$\{\tau_p\tau_i, \tau_q\tau_{i+1}, \tau_i\tau_s, \tau_i\tau_{i+1}\} \subseteq E(G),\ \tau_{i+1}\tau_s \notin E(G). \tag{5}$$

Case 2.1. $p < q$. It follows from Lemma 1 (ii) that $p < q \leq \ell_{G,\tau}(s)$ and hence $p \notin N_{G,\tau}[i+1] \cup N_{G,\tau}[s]$. Noting Eq. (5) additionally, we find that the set $\{\tau_i, \tau_{i+1}, \tau_s, \tau_p\}$ is a claw in G with τ_i being its center.

Case 2.2. $p = q$. By Lemma 1 (i), it holds $N^{\leq i-1}_{G,\tau}[i] = N^{\leq i-1}_{G,\tau}[i+1] = \langle p, i-1 \rangle$. Henceforth, recalling $N^{\geq i+1}_{G,\tau}[i] \nsubseteq N_{G,\tau}[i+1]$, the rule of $\texttt{MNS}^{\triangle}$ implies the existence of an element $t \in N^{\geq i+2}_{G,\tau}[i+1] \setminus N_{G,\tau}[i] = N_{G,\tau}[i+1] \setminus N_{G,\tau}[i]$.

In light of Eq. (5) now, as $G[\tau_s, \tau_i, \tau_{i+1}, \tau_t]$ cannot be a 4-cycle, we obtain $\tau_t\tau_s \notin E(G)$. Here are what we know on $E(G)$ in this case:

$$\{\tau_p\tau_i, \tau_p\tau_{i+1}, \tau_i\tau_s, \tau_i\tau_{i+1}, \tau_t\tau_{i+1}\} \subseteq E(G),\ \{\tau_{i+1}\tau_s, \tau_t\tau_i, \tau_t\tau_s\} \cap E(G) = \emptyset. \quad (6)$$

Case 2.2.1. $p = q > 1$. As G is connected, we can find an element $k \in N_{G,\tau}^{\leq p-1}[p]$. By Lemma 1 (ii), we deduce from $\min\{s, t\} > i+1$ that $k < p = q = \min\{\ell_{G,\tau}(i), \ell_{G,\tau}(i+1), \ell_{G,\tau}(s), \ell_{G,\tau}(t)\}$ and hence get to

$$\{i, i+1, s, t\} \cap N_{G,\tau}[k] = \emptyset. \quad (7)$$

Case 2.2.1.1. $s \in N_{G,\tau}(p)$. Making use of Eqs. (6) and (7), we see that $\{\tau_p, \tau_{i+1}, \tau_s, \tau_k\}$ is a claw in G centered at τ_p.

Case 2.2.1.2. $t \in N_{G,\tau}(p)$. By Eqs. (6) and (7), $\{\tau_p, \tau_i, \tau_t, \tau_k\}$ is a claw with τ_p at the center.

Case 2.2.1.3. Neither s nor t lies in $N_{G,\tau}(p)$. By Eqs. (6) and (7), we can check that $[\tau_t, \tau_{i+1}, \tau_i, \tau_s]$ is a path missing $N_G[\tau_k]$, $[\tau_s, \tau_i, \tau_p, \tau_k]$ is a path missing $N_G[\tau_t]$ and $[\tau_k, \tau_p, \tau_{i+1}, \tau_t]$ is a path missing $N_G[\tau_s]$. In all, we find that $\{\tau_s, \tau_t, \tau_k\}$ is an AT of G.

Case 2.2.2. $p = q = 1$. According to Eq. (6), $\{i, i+1\} \subseteq N_{G,\tau}[1]$ and $\tau_s\tau_{i+1}, \tau_t\tau_i \notin E(G)$. Because $\tau(1) = u$ is a simplicial vertex of G, this leads to $\{s, t\} \cap N_{G,\tau}[1] = \emptyset$. Let A and B be the connected components of $G[V(G) \setminus N_G[\tau_1]]$ such that $\tau_s \in A$ and $\tau_t \in B$.

Case 2.2.2.1. $A = B$. This means that τ_s and τ_t can be connected by a path P which misses $N_G[\tau_1]$. In view of Eq. (6) and the fact that $\tau_s, \tau_t \notin N_G[\tau_1]$, we see that $\{\tau_s, \tau_t, \tau_1\}$ is an AT with three certificate paths $[\tau_s, \tau_i, \tau_1]$, $[\tau_1, \tau_{i+1}, \tau_t]$ and the path P.

Case 2.2.2.2. Now assume that $A \neq B$. Recall that $\tau_1 = u$ is an `MNS` end-vertex. By Lemma 2, there exist $a \in A$, $b \in B$ and $c \in N_G(\tau_1)$ such that $\{\tau_1, c, a, b\}$ is a claw centered at c, completing the proof. □

Given any `MNS` type algorithm $\mathcal{A}$, we now present an algorithm which we call T-MNS$^{\mathcal{A}}$, or the Two-Sweep `MNS` UIG Certifying Algorithm based on $\mathcal{A}$. Note that Theorems 1, 5, 12 and 13 validate the correctness of this algorithm. In the following pseudocodes of T-MNS$^{\mathcal{A}}$, we use annotations to indicate the correspondence between the parts of the algorithm with Theorem 1 and the different cases distinguished in the proof of Theorem 13.

T-MNS$^{\mathcal{A}}(G)$
▷ **Input** a connected graph G with $|V(G)| = n$
▷ **Output** an answer that G is a UIG as well as one of its UI-orderings

3 ▷ or a certificate showing that G is not a UIG
4 Generate an MNS ordering σ of G
5 **if** σ is not a PEO
6   **then return** an induced cycle of length at least four
7   ▷ Theorem 1
8 Do $\mathcal{A}^{\triangle}(G, \sigma_n)$ to yield an ordering τ
9 $N_{G,\tau}^{\geq i+1}[i] \leftarrow \{k : k \geq i+1 \text{ and } \tau_k \in N_G[\tau_i]\}$
10 **for** $i \leftarrow 2$ **to** $n-1$
11   **do if** $N_{G,\tau}^{\geq i+1}[i] \nsubseteq N_{G,\tau}^{\geq i+1}[i+1]$
12   **then** $p \leftarrow \min\{j : \tau_j \in N_G[\tau_i]\}$
13       $q \leftarrow \min\{j : \tau_j \in N_G[\tau_{i+1}]\}$
14   ▷ Case 2: we must have $p \leq q$ by Lemma 1
15   Pick an $s \in N_{G,\tau}^{\geq i+1}[i] \setminus N_{G,\tau}^{\geq i+1}[i+1]$
16   **if** $p < q$
17   **then return** $\{\tau_i, \tau_{i+1}, \tau_s, \tau_p\}$ is a claw
18   ▷ Case 2.1
19   Choose a t so that $\tau_t \in N_G[\tau_{i+1}] \setminus N_G[\tau_i]$
20   ▷ Case 2.2
21   **if** $p = q > 1$
22   **then** pick a k such that $k < p$ and $\tau_k \in N_G(\tau_p)$
23   **if** $\tau_s \in N_G(\tau_p)$
24   **then return** $\{\tau_p, \tau_{i+1}, \tau_s, \tau_k\}$ is a claw
25   ▷ Case 2.2.1.1
26   **if** $\tau_t \in N_G(\tau_p)$
27   **then return** $\{\tau_p, \tau_i, \tau_t, \tau_k\}$ is a claw
28   ▷ Case 2.2.1.2
29   **return** $\{\tau_s, \tau_t, \tau_k\}$ is an AT with certificate paths
30           $[\tau_s, \tau_i, \tau_p, \tau_k], [\tau_k, \tau_p, \tau_{i+1}, \tau_t]$ and $[\tau_t, \tau_{i+1}, \tau_i, \tau_s]$
31   ▷ Case 2.2.1.3
32   ▷ We now have $p = q = 1, \tau_s, \tau_t \in V(G) \setminus N_G[\tau_1]$
33       $A \leftarrow$ the connected component of $G - N_G[\tau_1]$ containing τ_s
34       $B \leftarrow$ the connected component of $G - N_G[\tau_1]$ containing τ_t
35   **if** $A = B$
36   **then** $P \leftarrow$ a path in A connecting τ_s and τ_t
37   **return** $\{\tau_s, \tau_t, \tau_1\}$ is an AT with three certificate
38   paths $P, [\tau_s, \tau_i, \tau_1]$ and $[\tau_1, \tau_{i+1}, \tau_t]$
39   ▷ Case 2.2.2.1
40   **else** choose $c \in N_G(A) \cap N_G(B) \cap N_G(\tau_1)$
41   choose $a \in A \cap N_G(c), b \in B \cap N_G(c)$
42   **return** $\{\tau_1, c, a, b\}$ is a claw
43   ▷ Case 2.2.2.2
44 **return** τ is a UI-ordering of the UIG G
45 ▷ Case 1

Step 4 can be done with any easily implementable `MNS` type algorithm, say

LBFS or MCS. Step 6 can be done in linear time using the algorithm of [30]. As mentioned in Section 1, one way to implement Step 8 is to adopt $\mathcal{A}$ as LBFS and then follow the implementation technique of LBFS+. Finally, a UI-ordering is very easy to test [4, p. 377] and so our algorithm really provides a good membership certificate.

Theorem 15 *[6, Theorem 3.1] Let G be an arbitrary graph. Then every vertex from $AS(G)$ must be an* LBFS *end-vertex.*

Corollary 16 *A connected graph G is a unit interval graph if and only if $AS(G) \neq \emptyset$, G is AT-free and contains neither induced 4-cycle nor claw.*

Proof: The "only if" part is trivial and so we just need to prove the "if" direction. By Theorem 15, any vertex from $AS(G)$ must be both simplicial and an MNS end-vertex. Accordingly, Theorem 13 implies the result. □

3 MNS properties of RIG/UIG

Lemma 3 *Let G be a graph with m maximal cliques and μ an ordering of $\mathscr{C}(G)$ such that $[\mu_1, \ldots, \mu_m]$ is a clique path of G. (i) If an ordering τ of $V(G)$ is both consecutive and left-compatible with μ, then $(\mu_{i+1} \cap \mu_{i+2}) \setminus \mu_i \neq \emptyset$ for every $i \in [m-2]$. (ii) If G has a PSO τ which is left-compatible with μ, then $(\mu_i \cap \mu_{i+1}) \setminus \mu_{i+2} \neq \emptyset$ for every $i \in [m-2]$.*

Proof: (i) Pick $i \in [m-2]$ and let $s = \max\{r : \tau_r \in \mu_{i+1}\}$. We aim to show that $\tau_s \in (\mu_{i+1} \cap \mu_{i+2}) \setminus \mu_i$, or equivalently, as $[\mu_1, \ldots, \mu_m]$ is a clique path of G, that $\ell_\mu(\tau_s) = i+1$ and $r_\mu(\tau_s) \geq i+2$. Because τ is left-compatible with μ and none of $\mu_{i+1} \setminus \mu_i$ and $\mu_{i+2} \setminus \mu_{i+1}$ can be empty, the maximality of s gives $\ell_\mu(\tau_s) = i+1$ and $\ell_\mu(\tau_{s+1}) = i+2$. On the other hand, since τ is a consecutive ordering, we have $\tau_s\tau_{s+1} \in E(G)$ and so $r_\mu(\tau_s) \geq \ell_\mu(\tau_{s+1}) \geq i+2$, as wanted.

(ii) Take $i \in [m-2]$ and choose $s = \min\{r : \tau_r \in \mu_{i+1} \setminus \mu_{i+2}\}$. We want to show that $\tau_s \in (\mu_i \cap \mu_{i+1}) \setminus \mu_{i+2}$. For this, we need only check that $\ell_\mu(\tau_s) \leq i$. If $\ell_\mu(\tau_s) \geq i+1$, then $s > 1$ and $\ell_\mu(\tau_s) = r_\mu(\tau_s) = i+1$. Pick $\tau_t \in \mu_{i+2} \setminus \mu_{i+1}$ and so $\tau_t\tau_s \notin E(G)$. Owing to the fact that τ is left-compatible with μ, it holds $s < t$. For any $p \in N_{G,\tau}^{\leq s-1}[s]$, the minimality assumption on s together with $N_G[\tau_s] \subseteq \mu_{i+1}$ forces $\tau_p \in \mu_{i+1} \cap \mu_{i+2}$ and hence $p \in N_{G,\tau}^{\leq s-1}[t]$. This means that $\{\tau_s, \tau_t\} \subseteq S_\tau[s]$. Recall that τ is a PSO and $s > 1$ and so $S_\tau[s]$ should be a clique. This implies that $\tau_t\tau_s \in E(G)$, reaching a desired contradiction. □

Lemma 4 *Let G be a graph with m maximal cliques and n vertices. (i) If G has a clique path $[\mu_1, \ldots, \mu_m]$, then any ordering τ of G which is left-compatible with μ must be an I-ordering of G. (ii) If G has an I-ordering τ, then G has a clique path $[\mu_1, \ldots, \mu_m]$ such that τ is left-compatible with μ.*

Proof: (i) Take $1 \leq i < j < k \leq n$. Since τ is left-compatible with μ, we see that $\ell_\mu(\tau_i) \leq \ell_\mu(\tau_j) \leq \ell_\mu(\tau_k)$. If $\tau_i\tau_k \in E(G)$, we will further have $\ell_\mu(\tau_k) \leq r_\mu(\tau_i)$ and hence $\ell_\mu(\tau_i) \leq \ell_\mu(\tau_j) \leq r_\mu(\tau_i)$ and so $\tau_i\tau_j \in E(G)$ follows.

(ii) As τ is an I-ordering, we can get an interval representation $\mathcal{I}$ of G by putting $\ell_{\mathcal{I}}(\tau_i) = i$ and $r_{\mathcal{I}}(\tau_i) = r_{G,\tau}(i)$ for each $i \in [n]$. Let μ be the natural clique ordering of $\mathscr{C}(G)$ such that $\cap_{v\in\mu_i}\mathcal{I}(v)$ is to the left of $\cap_{v\in\mu_j}\mathcal{I}(v)$ whenever $i < j$. It is not difficult to see that τ is left-compatible with μ. □

Proof of Theorem 3: We first prove the backward direction. Let T' be any clique tree of G and we want to show that $T' = T$. Take any two maximal cliques C and $\widehat{C}$ which are adjacent in T'. Let the path in T connecting C and $\widehat{C}$ be $[C = C_{i_1}, \dots, C_{i_k} = \widehat{C}]$ and then our task is to derive $k = 2$. By way of contradiction, suppose that $k > 2$. Pick $x \in C \setminus \widehat{C}$ and $y \in \widehat{C} \setminus C$. From Eq. (1) and $k > 2$ we find that there exists $v_j \in (C_{i_j} \cap C_{i_{j+1}}) \setminus C_{i_{j+2}} \subseteq (C_{i_j} \cap C_{i_{j+1}}) \setminus \widehat{C}$ for each $j \in [k-2]$ and there exists $v_{k-1} \in (C_{i_{k-1}} \cap \widehat{C}) \setminus C_{i_{k-2}} \subseteq (C_{i_{k-1}} \cap \widehat{C}) \setminus C$. This means that $[x, v_1, \dots, v_{k-1}, y]$ is a path in $G[V(G) \setminus (C \cap \widehat{C})]$. Since $C\widehat{C} \in E(T')$ and T' is a clique tree of G, we know that $C \cap \widehat{C}$ should be an x, y-separator in G, arriving at a desired contradiction.

We next suppose that T is the unique clique tree of G and try to show that no three different maximal cliques C_i, C_j and C_k can satisfy Eq. (1). Otherwise, replacing the edge C_iC_j by the new edge C_iC_k will yield from T a new clique tree of G, contradicting with the uniqueness of T. □

Proof of Theorem 4: (i) Let $n = |V(G)|$. By Lemma 4, τ is an I-ordering of G. We further check that τ is consecutive. Pick $i \in [n-1]$ and we want to show that $\tau_i\tau_{i+1} \in E(G)$. As τ is an I-ordering, it suffices to find a $k > i$ such that $\tau_i\tau_k \in E(G)$. If $r_\mu(\tau_i) = m$, because τ is left-compatible with μ, we see that $N_G(\tau_i) \supseteq \{\tau_{i+1}, \dots, \tau_n\}$ and so we are done. Otherwise, as G is a rigid interval graph, Theorem 3 implies that we can find $\tau_k \in (\mu_{r_\mu(\tau_i)} \cap \mu_{r_\mu(\tau_i)+1}) \setminus \mu_{r_\mu(\tau_i)-1}$ where we regard μ_0 as the empty set. It is clear that $\tau_i\tau_k \in E(G)$. It is also not hard to see that $\tau_i \prec_\mu \tau_k$ and so, as τ is compatible with μ, we get $i < k$, showing that this τ_k is what we are searching for. Finally, we need to show that τ is a PSO. Take $s \in \langle 2, n\rangle$ and make the convention that $\mu_0 = \mu_1$ and $\mu_{m+1} = \emptyset$. Since τ is left-compatible with μ, it holds $N^{\leq s-1}_{G,\tau}[i] \subseteq N^{\leq s-1}_{G,\tau}[s]$ for every $i \in \langle s, n\rangle$. Because G is a rigid interval graph and τ is compatible with μ, Theorem 3 asserts the existence of a $t \in [s-1]$ such that $\tau_t \in (\mu_{\ell_\mu(\tau_s)} \cap \mu_{\ell_\mu(\tau_s)-1}) \setminus \mu_{\ell_\mu(\tau_s)+1}$. We can now conclude that $S_\tau[s]$ is a subset of $\mu_{\ell_\mu(\tau_s)}$ and hence a clique.

(ii) By Lemma 4, G has a clique path $[\mu_1, \dots, \mu_m]$ such that τ is left-compatible with μ. It is clear that $\tau_1 \in \mu_1$. But $\tau_1 \in \mu_2$ is impossible as that will force $S_\tau[2] \supseteq (\mu_1 \cup \mu_2) \setminus \{\tau_1\} \supseteq \mu_1 \triangle \mu_2$ while $\mu_1 \triangle \mu_2$ cannot be a clique. Now, an application of Lemma 3 and Theorem 3 then completes the proof. □

The next simple theorem is the crux in our work to understand the `MNS` structure of rigid interval graphs. If we view `MNS` as traversing maximal cliques rather than vertices (a clique is traversed when its last vertex has been traversed), Theorem 3 together with Theorem 17 says that by applying an `MNS` type algorithm on a rigid interval graph we simply explore outward from one maximal clique in two directions and "flood" the unique clique path with an expanding wave that grows and finally stops at an end-clique.

Theorem 17 *Let G be a graph with a clique path $P = [\mu_1, \ldots, \mu_m]$ and let τ be an* MNS *ordering of G. Then, there exists $s \in [m]$ such that the following hold: (i) $\tau_1 \in \mu_s$; (ii) If $(\mu_i \cap \mu_{i+1}) \setminus \mu_{i+2} \neq \emptyset$ holds for each $i \in \langle s, m-2 \rangle$ and $s \leq \ell_\mu(\tau_r) < \ell_\mu(\tau_p)$, then $r < p$; (iii) If $(\mu_{i+2} \cap \mu_{i+1}) \setminus \mu_i \neq \emptyset$ holds for each $i \in [s-2]$ and $s \geq r_\mu(\tau_r) > r_\mu(\tau_p)$, then $r < p$.*

Proof: Let q be the maximum number such that $\{\tau_1, \ldots, \tau_q\}$ is a clique in G. By the MNS rule, we see that $\{\tau_1, \ldots, \tau_q\}$ must indeed form a maximal clique C of G. Take s so that $C = \mu_s$ and then claim (i) is satisfied.

By symmetry, it remains to verify (ii) in the sequel. Since μ corresponds to a clique path of G and vertices in μ_s appear earlier than those outside of μ_s in the ordering τ, we see that for any $v \in \cup_{t=s}^m \mu_t$, those neighbors of v appeared earlier than v in the ordering τ must all lie in $\cup_{t=s}^m \mu_t$. It is also clear that $[\mu_s, \ldots, \mu_m]$ is a clique tree of $G[\cup_{t=s}^m \mu_t]$. Henceforth, there is no loss of generality in assuming that $s = 1$. The remaining proof is by induction on $\ell_\mu(\tau_r)$. When $\ell_\mu(\tau_r) = 1$, we have $r \leq q < p$ and hence the claim is trivially true. Consider now $\ell_\mu(\tau_r) = i + 1 > 1$. Without loss of generality, we assume that $p = \min\{t : \ell_\mu(\tau_t) > i + 1\}$. This assumption as well as the fact that P is a clique path tell us that $N_{G,\tau}^{\leq p-1}[p] \subseteq N_{G,\tau}^{\leq p-1}[r]$. Because τ is an MNS ordering, to get $r < p$ it is sufficient to show $N_{G,\tau}^{\leq p-1}[r] \setminus N_{G,\tau}^{\leq p-1}[p] \neq \emptyset$. Pick t so that $\tau_t \in (\mu_i \cap \mu_{i+1}) \setminus \mu_{i+2} \subseteq (\mu_i \cap \mu_{i+1}) \setminus \mu_{\ell_\mu(\tau_p)}$. Since $\ell_\mu(\tau_t) \leq i$, the induction assumption gives $t < p$ and hence $t \in N_{G,\tau}^{\leq p-1}[r] \setminus N_{G,\tau}^{\leq p-1}[p]$, as desired. □

Example 18 *Let G be the graph depicted in Fig. 1(f). The* MNS *orderings starting at* 1 *can be listed as* 1, 2, 3, 4, 5 *and* 1, 2, 4, 3, 5*. Both orderings are I-orderings and this is also predicted by Lemma 4 (i) and Theorem 17. Observe that the former is even a UI-ordering but the latter is not any UI-ordering.*

Lemma 5 *Let G be a unit interval graph with a clique path $[\mu_1, \ldots, \mu_m]$. Every ordering τ of $V(G)$ which is compatible with μ is a UI-ordering of G.*

Proof: Suppose that $1 \leq p < q < r \leq |V(G)|$ and that $\tau_p\tau_r \in E(G)$. Our task is to show $\tau_p\tau_q, \tau_q\tau_r \in E(G)$. Since τ is compatible with μ, $\tau_p\tau_r \in E(G)$ and $p < q < r$, we can obtain $\tau_p\tau_q \in E(G)$. To finish the proof, by Theorem 12, we need only show the existence of a claw in G under the assumption of $\tau_q\tau_r \notin E(G)$. Since τ is compatible with μ, $\tau_p\tau_r \in E(G)$, $\tau_q\tau_r \notin E(G)$ and $p < q$, we see that $\ell_\mu(\tau_p) < \ell_\mu(\tau_q)$. Take $\tau_t \in \mu_{\ell_\mu(\tau_p)} \setminus \mu_{\ell_\mu(\tau_p)+1}$. It is no hard to check that $\{\tau_p, \tau_t, \tau_q, \tau_r\}$ constitutes a claw centered at τ_p, as desired. □

The part of Theorem 8 on unit interval graphs follows directly from Theorems 1, 12 and 13. Note that our ensuing proof of Theorem 8 is valid for both unit interval graphs and rigid interval graphs.

Proof of Theorem 8: Recall from Theorem 6 that all connected unit interval graphs are rigid interval graphs. Let $[\mu_1, \ldots, \mu_m]$ be the unique clique path of the input rigid interval graph G. By Theorem 4 and Lemma 5, it suffices to show that τ is compatible with μ or its reversal. We need only consider the case of

$m > 1$. Applying Theorems 3 and 17 on σ yields $\tau_1 = \sigma_n \in (\mu_1 \setminus \mu_2) \cup (\mu_m \setminus \mu_{m-1})$. Without loss of generality, assume $\tau_1 \in \mu_1 \setminus \mu_2$. Applying Theorems 3 and 17 on τ yields that τ is left-compatible with μ. Take $1 \leq i < j \leq n$ and suppose that $\ell_\mu(\tau_i) = \ell_\mu(\tau_j)$. Since τ is left-compatible with μ, τ_i and τ_j both lie in $S_\tau[i]$. Observe that $N_G(v) = \cup_{t=\ell_\mu(v)}^{r_\mu(v)} \mu_t \setminus \{v\}$ for each $v \in V(G)$. Therefore, the rule of $\mathcal{A}^\triangle(G, \sigma_n)$ gives $r_\mu(\tau_i) \leq r_\mu(\tau_j)$ and this completes the proof. □

Theorem 9 is immediate from Theorems 6, 8 and 10. But our proof of Theorem 10 here will rely on the following proof of Theorem 9.

Proof of Theorem 9: Let $[\mu_1, \ldots, \mu_m]$ be the clique path of the input rigid interval graph G. As in the proof of Theorem 8, our task is to show that τ is compatible with μ or its reversal when $m > 1$. Theorems 3 and 17 applied on δ says that $\sigma_1 = \delta_n \in (\mu_1 \setminus \mu_2) \cup (\mu_m \setminus \mu_{m-1})$. Without loss of generality, suppose that $\sigma_1 \in \mu_m \setminus \mu_{m-1}$. An application of Theorems 3 and 17 on σ gives that $\sigma_n \in \mu_1 \setminus \mu_2$ and that σ is left-compatible with μ', the reversal of μ. Utilizing Theorems 3 and 17 again yields that τ is left-compatible with μ. It is now sufficient to show that for any $1 \leq i < j \leq n$ fulfilling $\ell_\mu(\tau_i) = \ell_\mu(\tau_j)$ we have $r_\mu(\tau_i) \leq r_\mu(\tau_j)$. Let $\tau_i = \sigma_{i'}$ and $\tau_j = \sigma_{j'}$. The assumption $\ell_\mu(\tau_i) = \ell_\mu(\tau_j)$ means that τ_j lies in the `MNS` slice of τ when τ_i is to be chosen. According to the rule of $\mathcal{A}+$, the reason that the $\mathcal{A}+$ slice of τ at time i consists of τ_i itself can only be $i' > j'$. Because σ is left-compatible with μ', we get $r_\mu(\tau_i) = r_\mu(\sigma_{i'}) = m + 1 - \ell_{\mu'}(\sigma_{i'}) \leq m + 1 - \ell_{\mu'}(\sigma_{j'}) = r_\mu(\sigma_{j'}) = r_\mu(\tau_j)$, as desired. □

Proof of Theorem 10: According to the proof of Theorem 9, we can assume that τ is compatible with an ordering μ of $\mathscr{C}(G)$ that gives the unique clique path of G. Take $i \in \langle 2, n \rangle$ and let $S_\tau(i)$ be the `MNS` slice of τ at time i. Let $\ell_\mu(\tau_i) = p$ and $r_\mu(\tau_i) = q$. Then, we have $\ell_\mu(x) = p$ and $r_\mu(x) \geq q$ for any $x \in S_\tau(i)$. Consequently, we now get to $N_G[\tau_i] \subseteq \cap_{x \in S_\tau(i)} N_G[x]$. Since this holds for all $i \geq 2$, it is clear that τ is an output of the algorithm $\mathcal{A}^\triangle(G, \sigma_n)$. □

Proof of Theorem 11: For the forward direction, we use `3MNS-UI/RI`$^{\mathcal{A}}$ to generate in turn the orderings δ, σ and τ of G. We further do $\mathcal{A}+(G, \tau)$ to get an ordering ρ. Applying Theorem 9 on the first three sweeps δ, σ and τ shows that τ is an RI-ordering. Utilizing Theorem 9 again on the last three sweeps σ, τ and ρ, we find that ρ is an RI-ordering with $\rho_1 = \tau_n$.

We now turn to the backward direction. The graph G is surely a rigid interval graph when $|\mathscr{C}(G)| = m \leq 2$. We thus assume $m > 2$. As observed in Lemma 4, G has a clique path $[\mu_1, \ldots, \mu_m]$ such that the I-ordering τ is left-compatible with μ. Noting that τ is even consecutive, we conclude from Lemma 3 that

$$(\mu_{i+1} \cap \mu_{i+2}) \setminus \mu_i \neq \emptyset \tag{8}$$

for every $i \in [m-2]$. Let μ' be the reversal of μ. As τ is left-compatible with μ, it holds $\rho_1 = \tau_n \in \mu_m \setminus \mu_{m-1} = \mu'_1 \setminus \mu'_2$. By Theorem 17 and Eq. (8), the `MNS` ordering ρ must be left-compatible with μ'. Because ρ is assumed to be consecutive, Lemma 3 now asserts for every $i \in [m-2]$ that

$$(\mu_{m-i} \cap \mu_{m-i-1}) \setminus \mu_{m-i+1} = (\mu'_{i+1} \cap \mu'_{i+2}) \setminus \mu'_i \neq \emptyset. \qquad (9)$$

In view of Theorem 3, Eqs. (8) and (9) verify the validity of the backward direction, as desired. □

Theorems 8 and 9 as well as the proof of Theorem 11 provide three multi-sweep `MNS` algorithms for recognizing unit/rigid interval graphs. The algorithm corresponding to Theorem 11 does not involve recognizing RI-orderings but those corresponding to Theorems 8 and 9 do require us recognize an RI-ordering. Let us illustrate here how to do this in linear time. Let G be a graph with n vertices and m edges. It is known that checking a UI-ordering or an I-ordering only takes linear time [4, 10]. Henceforth, we focus on checking whether or not a given I-ordering τ of G is an RI-ordering. Observe that $N^{\leq i}_{G,\tau}[k] \subseteq N^{\leq i}_{G,\tau}[j]$ for any $1 \leq i < j < k \leq n$, which implies that τ is an `LBFS/LDFS/LMCS/MCS/MNS` ordering and the `LBFS/LDFS/LMCS/MCS/MNS` slice at time $i+1$ is $S_\tau(i+1) = \{\tau_s : s > i, N^{\leq i}_{G,\tau}[s] = N^{\leq i}_{G,\tau}[i+1]\}$. Consequently, we only need to check for all $i \in [n-1]$ whether or not $S_\tau(i+1)$ are cliques. A key observation is that if $S_\tau(i+1) = \{\tau_{i+1}, \ldots, \tau_s\}$ is a clique, then so is $S_\tau(i') = \{\tau_{i'}, \ldots, \tau_s\}$ for every $i' \in \langle i+1, s\rangle$. Since both `LBFS` [10, 23] and `MCS` [28, 29] are `MNS` type algorithms and have simple linear time implementations, we can efficiently determine a sequence of numbers $1 = t_1 < \cdots < t_q = n$ such that $S_\tau(t_j+1) = \{\tau_{t_j+1}, \ldots, \tau_{t_{j+1}}\}$ for $j \in [q-1]$. To tell whether or not τ is a PSO now reduces to checking whether or not $S_\tau(t_j+1)$ is a clique for every $j \in [q-1]$. This then confirms that we can recognize an RI-ordering in $O(n+m)$ time and so all the `MNS` algorithms reported in this paper have simple linear time implementations.

Theorem 19 *Let G be a graph with a clique path $P = [\mu_1, \ldots, \mu_m]$. (i) Each vertex $v \in \mu_1 \cup \mu_m$ is admissible. (ii) If P is even the unique clique tree of G, then every admissible vertex comes from $\mu_1 \cup \mu_m$ and $AS(G) = \mu_1$ when $m = 1$ and $AS(G) = (\mu_1 \setminus \mu_2) \cup (\mu_m \setminus \mu_{m-1})$ when $m \geq 2$.*

Proof: To prove the first statement, without loss of generality, we assume that $v \in \mu_1$. Let u and w be any vertices of G with $\ell_\mu(u) \leq \ell_\mu(w)$. Then, for any v,w-path Q in G, it follows from $\cup_{x\in Q}\langle \ell_\mu(x), r_\mu(x)\rangle \supseteq \langle \ell_\mu(v), r_\mu(w)\rangle \supseteq [\ell_\mu(w)] \ni \ell_\mu(u)$ that some vertex of Q appears in $N_G[u]$, hence showing that v is admissible.

We now direct our attention to (ii). As the case of $m = 1$ is trivial, we may assume $m \geq 2$. Note that a vertex from $\mu_1 \cup \mu_m$ is simplicial if and only if it lies in $(\mu_1 \setminus \mu_2) \cup (\mu_m \setminus \mu_{m-1})$. This combined with (i) reduces our task to showing that v is not admissible provided $1 < i = \ell_\mu(v) \leq r_\mu(v) = j < m$. Take $x \in \mu_1 \setminus \mu_2$ and $y \in \mu_m \setminus \mu_{m-1}$. By Theorem 3, there exist $x_1 \in (\mu_1 \cap \mu_2) \setminus \mu_m, \ldots, x_{i-1} \in (\mu_{i-1} \cap \mu_i) \setminus \mu_m$ and this ensures that a subsequence of $x, x_1, \ldots, x_{i-1}, v$ gives rise to a path connecting x and v in G that misses $N_G[y]$. By symmetry, there is a path connecting y and v that misses $N_G[x]$. We thus conclude that v is not admissible, finishing the proof. □

Proof of Theorem 2: This follows from Theorems 3, 15, 17 and 19. □

4 Concluding remarks

As with the multi-sweep graph search recognition of (unit) interval graphs, it is largely believed that LBFS is the correct graph traversal algorithm to be used. When discussing their interval graph recognition algorithm, Korte and Möhring [13, p. 74] make the comments that "BFS together with lexicographical tie breaking are essential for these results". After establishing some structural theorem, Simon [25] asserts that "Probably this theorem explains why lexBFS has to be preferred to maximum cardinality search in some applications like recognizing interval graphs". In [4, p. 373] Corneil gives the following description of his 3-sweep LBFS algorithm for recognizing unit interval graphs: "The first sweep is an arbitrary LBFS to find a "left-anchor". The following two LBFS sweeps, with a specific tie-breaking rule, provide an ordering of V that satisfies the "neighbourhood condition" if and only if G is a unit interval graph." Our work here develops the idea of Corneil in several aspects. Firstly, we find that to get a UI-ordering of a unit interval graph, the first LBFS sweep in Corneil's algorithm can be replaced by an MNS sweep, the second LBFS+ sweep by another MNS sweep and the third LBFS+ sweep by an MNS+ sweep, which may be a bit surprising due to the above comments of Korte-Möhring and Simon. Secondly, we find that our 3-sweep algorithm as well as its 2-sweep and 4-sweep variants (Theorems 8, 9 and 11) apply essentially to rigid interval graphs, a superclass of unit interval graphs, where an RI-ordering should be in place of a UI ordering as a membership certificate (Theorem 4). Thirdly, we suggest an algorithm $\mathcal{A}^{\triangle}$ for any MNS algorithm $\mathcal{A}$ and use $\mathcal{A}^{\triangle}$ to derive a certifying algorithm $\texttt{T-MNS}^{\mathcal{A}}$ for recognizing unit interval graphs. Lastly, we have a better understanding of the role of each MNS sweep in $\texttt{3MNS-UI/RI}^{\mathcal{A}}$: Roughly speaking, as seen from the proof of Theorem 9, the first sweep of MNS can capture an end-interval/ "left-anchor", a second sweep of MNS then sorts the left end-points of the intervals and gives us an I-ordering of those intervals and finally a third sweep of MNS sorts the right end-points for each group of intervals whose left end-points are nearby in certain sense and then reconstructs a required UI-ordering/RI-ordering. Note that we also earn more understanding of what is the "left-anchor" in a unit/rigid interval graph (Theorems 2 and 19); also see [16, Lemma 4.25].

The key idea of this paper is the formulation of the concept of rigid interval graphs and a realization of the fact that many earlier results on unit interval graphs may be better understood as results for rigid interval graphs. We develop here structural and algorithmic properties of rigid interval graphs, especially its properties related to MNS type algorithms. It seems interesting to see which kind of insight can be obtained on the rigid interval graphs and related graph classes from other viewpoints, say with the help of the PQ-tree techniques [2].

Building on some deep results from [8], we design a simple linear time 4-sweep LBFS algorithm for recognizing interval graphs in [16]. Compared with the work in [8, 16] on multi-sweep LBFS algorithms for recognizing interval graphs, our algorithms for recognizing rigid/unit interval graphs in this paper can work with much wider MNS type algorithms, the algorithm analysis is much easier and each step of the algorithm can be much more transparent to the users – all these are mainly due to the intrinsic simplicity of the rigid interval graphs from the viewpoint of clique tree representation.

References

[1] B. Aspvall and P. Heggernes. Finding minimum height elimination trees for interval graphs in polynomial time. *BIT*, 34:484–509, 1994. doi:10.1007/BF01934264.

[2] K. S. Booth and G. S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976. doi:10.1016/S0022-0000(76)80045-1.

[3] A. Brandstädt, V. D. Chepoi, and F. F. Dragan. Perfect elimination orderings of chordal powers of graphs. *Discrete Mathematics*, 158(1–3):273–278, 1996. doi:10.1016/0012-365X(95)00081-7.

[4] D. G. Corneil. A simple 3-sweep LBFS algorithm for the recognition of unit interval graphs. *Discrete Applied Mathematics*, 138(3):371–379, 2004. doi:10.1016/j.dam.2003.07.001.

[5] D. G. Corneil, August 9, 2011. Private communication to Yaokun Wu.

[6] D. G. Corneil, E. Köhler, and J.-M. Lanlignel. On end-vertices of lexicographic breadth first searches. *Discrete Applied Mathematics*, 158(5):434–443, 2010. doi:10.1016/j.dam.2009.10.001.

[7] D. G. Corneil and R. Krueger. A unified view of graph searching. *SIAM Journal on Discrete Mathematics*, 22(4):1259–1276, 2008. doi:10.1137/050623498.

[8] D. G. Corneil, S. Olariu, and L. Stewart. The LBFS structure and recognition of interval graphs. *SIAM Journal on Discrete Mathematics*, 23(4):1905–1953, 2009. doi:10.1137/S0895480100373455.

[9] X. Deng, P. Hell, and J. Huang. Linear-time representation algorithms for proper circular-arc graphs and proper interval graphs. *SIAM Journal on Computing*, 25(2):390–403, 1996. doi:10.1137/S0097539792269095.

[10] M. Habib, R. McConnell, C. Paul, and L. Viennot. Lex-BFS and partition refinement, with applications to transitive orientation, interval graph recognition and consecutive ones testing. *Theoretical Computer Science*, 234(1-2):59–84, 2000. doi:10.1016/S0304-3975(97)00241-7.

[11] P. Hell and J. Huang. Certifying LexBFS recognition algorithms for proper interval graphs and proper interval bigraphs. *SIAM Journal on Discrete Mathematics*, 18(3):554–570, 2005. doi:10.1137/S0895480103430259.

[12] W. Hsu. $O(m \cdot n)$ algorithms for the recognition and isomorphism problems on circular-arc graphs. *SIAM Journal on Computing*, 24(3):411–439, 1995. doi:10.1137/S0097539793260726.

[13] N. Korte and R. Möhring. An incremental linear-time algorithm for recognizing interval graphs. *SIAM Journal on Computing*, 18(1):68–81, 1989. doi:10.1137/0218005.

[14] R. Krueger. Graph searching. 2005. Ph.D. thesis, University of Toronto.

[15] C. G. Lekkerkerker and J. C. Boland. Representation of a finite graph by a set of intervals on the real line. *Fundamenta Mathematicae*, 51:45–64, 1962.

[16] P. Li and Y. Wu. A four-sweep LBFS recognition algorithm for interval graphs. Submitted. URL: http://math.sjtu.edu.cn/faculty/ykwu/4-sweep-LW.pdf.

[17] T. Ma. Unpublished manuscript.

[18] R. McConnell, K. Mehlhorn, S. Näher, and P. Schweitzer. Certifying algorithms. *Computer Science Review*, 5(2):119–161, 2011. doi:10.1016/j.cosrev.2010.09.009.

[19] D. Meister. Recognition and computation of minimal triangulations for AT-free claw-free and co-comparability graphs. *Discrete Applied Mathematics*, 146(3):193–218, 2005. doi:10.1016/j.dam.2004.10.001.

[20] B. Panda and S. K. Das. A parallel algorithm for generating bicompatible elimination orderings of proper interval graphs. *Information Processing Letters*, 109(18):1041–1046, 2009. doi:10.1016/j.ipl.2009.06.011.

[21] F. S. Roberts. Indifference graphs. In F. Harary, editor, *Proof Techniques in Graph Theory*, pages 139–146. Academic Press, New York, 1969.

[22] F. S. Roberts. On the compatibility between a graph and a simple order. *Journal of Combinatorial Theory, Series B*, 11(1):28–38, 1971. doi:10.1016/0095-8956(71)90010-4.

[23] D. Rose, R. E. Tarjan, and G. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing*, 5(2):266–283, 1976. doi:10.1137/0205021.

[24] K. Simon. A new simple linear algorithm to recognize interval graphs. In H. Bieri and H. Noltemeier, editors, *Computational Geometry - Methods, Algorithms and Applications*, volume 553 of *Lecture Notes in Computer Science*, pages 289–308. Springer, 1991. doi:10.1007/3-540-54891-2_22.

[25] K. Simon. A note on lexicographic breadth first search for chordal graphs. *Information Processing Letters*, 54(5):249–251, 1995. doi:10.1016/0020-0190(95)00024-7.

[26] J. P. Spinrad. Efficient implementation of lexicographic depth first search. In preparation.

[27] J. P. Spinrad. *Efficient Graph Representations*, volume 19 of *Fields Institute Monographs*. American Mathematical Society, 2003.

[28] R. E. Tarjan. Maximum cardinality search and chordal graphs. 1976. Unpublished lecture notes for CS 259, Stanford University.

[29] R. E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 13(3):566–579, 1984. doi:10.1137/0213035.

[30] R. E. Tarjan and M. Yannakakis. Addendum: Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 14(1):254–255, 1985. doi:10.1137/0214020.