Computing role assignments of proper interval graphs in polynomial time*

Pinar Heggernes† Pim van 't Hof† Daniël Paulusma‡

Abstract

An R-role assignment of a graph G is a locally surjective homomorphism from G to graph R. For a fixed graph R, the R-Role Assignment problem is to decide, for an input graph G, whether G has an R-role assignment. When both graphs G and R are given as input, the problem is called Role Assignment. In this paper, we study the latter problem. It is known that R-Role Assignment is NP-complete already when R is a path on three vertices. In order to obtain polynomial time algorithms for Role Assignment, it is therefore necessary to put restrictions on G. So far, the only known non-trivial case for which this problem is solvable in polynomial time is when G is a tree. We present an algorithm that solves Role Assignment in polynomial time when G is a proper interval graph. Thus we identify the first graph class other than trees on which the problem is tractable. As a complementary result, we show that Role Assignment is Graph Isomorphism-hard on chordal graphs, a superclass of proper interval graphs and trees.

1 Introduction

Graph homomorphisms form a natural generalization of graph colorings: there is a homomorphism from a graph G to the complete graph on k vertices if and only if G is k-colorable. A *homomorphism* from a graph $G = (V_G, E_G)$ to a graph $R = (V_R, E_R)$ is a mapping $r : V_G \to V_R$ that maps adjacent vertices of G to adjacent vertices of R, i.e., $r(u)r(v) \in E_R$ whenever $uv \in E_G$. A homomorphism r from G to R is *locally surjective* if the following is true for every vertex u of G: for every neighbor y of $r(u)$ in R, there is at least one neighbor v of u in G with $r(v) = y$. We also call such an r an *R-role assignment*. See Figure 1 for an example.

Role assignments originate in the theory of social behavior [9, 25]. A role graph R models roles and their relationships, and for a given society we can ask if its individuals can be assigned roles such that relationships are preserved: each person playing a particular role has among its neighbors exactly the roles prescribed by the model. In this way, a large network of individuals can be compressed into a smaller network that still gives some description of the large network. Role assignments are also useful in the area of distributed computing, in which one of the fundamental problems is to arrive at a final configuration where all processors have been assigned

*This work has been supported by EPSRC (EP/D053633/1 and EP/G043434/1) and by the Research Council of Norway. A preliminary version has been presented at IWOCA 2010 [17].

†Department of Informatics, University of Bergen, P.O. Box 7803, N-5020 Bergen, Norway. Emails: {pinar.heggernes, pim.vanthof}@ii.uib.no

‡School of Engineering and Computing Sciences, Durham University, Science Laboratories, South Road, Durham DH1 3LE, England. Email: daniel.paulusma@durham.ac.uk

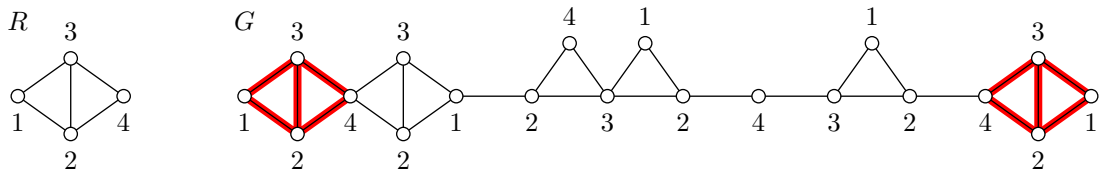

Figure 1: A graph R and a proper interval graph G with an R-role assignment.

unique identities. Chalopin et al. [4] show that, under a particular communication model, this problem can be solved on a graph G representing the distributed system if and only if G has no R-role assignment for any graph R with fewer vertices than G. Role assignments are useful in topological graph theory as well, where a main question is which graphs G allow role assignments to planar graphs R [27].

For a fixed graph R, the R-Role Assignment problem has as input a graph G, and asks whether G has an R-role assignment. The Role Assignment problem has as input an ordered pair of graphs (G, R) and asks whether G has an R-role assignment. Both problems are NP-complete; in fact R-Role Assignment is NP-complete on arbitrary graphs G, even when R is any connected bipartite graph on at least three vertices [12]. Hence, for polynomial time solvability of any of the problems, our only hope is to put restrictions on G. For Role Assignment, so far, the only known non-trivial graph class that gives tractability is the class of trees: Role Assignment is polynomial time solvable on input pairs (G, R) where G is a tree and R is arbitrary [13]. Are there other graph classes on which Role Assignment can be solved in polynomial time?

We show that Role Assignment can be solved in polynomial time on input pairs (G, R) where G is a proper interval graph and R is arbitrary. Our algorithm runs in time $\mathcal{O}((n+m)\cdot c_R)$, where n and m are the numbers of vertices and edges of G, respectively, and c_R is the number of connected components of R. Our work is motivated by the above question and continues the research direction of Sheng [29], who characterizes proper interval graphs that have an R-role assignment for some fixed role graph R with a small number of vertices. Proper interval graphs, also known as unit interval graphs or indifference graphs, are widely known due to their many theoretical and practical applications [3, 16, 28]. By our result, they form the first graph class other than trees on which Role Assignment is shown to be polynomial time solvable. In order to obtain our algorithm, we prove structural properties of clique paths of proper interval graphs related to role assignments. This enables us to give an additional result, namely a polynomial time algorithm for the problem of deciding whether there exists a graph R with fewer vertices than a given proper interval graph G such that G has an R-role assignment. As we mentioned earlier, this problem stems from the area of distributed computing [4]. It is co-NP-complete in general [5]. Finally, to indicate that Role Assignment might remain hard on larger graph classes, we show that it is Graph Isomorphism-hard for input pairs (G, R) where G belongs to the class of chordal graphs, a superclass of both proper interval graphs and trees.

2 Preliminaries

All graphs considered in this paper are undirected, finite and simple, i.e., without loops or multiple edges. A graph is denoted $G = (V_G, E_G)$, where V_G is the set of vertices and E_G is

the set of edges. We use n to denote the number of vertices and m to denote the number of edges of G. For a vertex u of G, $N_G(u) = \{v \mid uv \in E_G\}$ denotes the set of *neighbors* of u in G, also called the *neighborhood* of u. The *degree* of a vertex u in G is $\deg_G(u) = |N_G(u)|$. A graph $H = (V_H, E_H)$ is a *subgraph* of G if $V_H \subseteq V_G$ and $E_H \subseteq E_G$. For $U \subseteq V_G$, the graph $G[U] = (U,\ \{uv \in E_G \mid u, v \in U\})$ is called the subgraph of G *induced* by U. A graph is *complete* if it has an edge between every pair of vertices. A set of vertices $A \subseteq V_G$ is a *clique* if $G[A]$ is complete. A clique is *maximal* if it is not a proper subset of any other clique.

An *isomorphism* from a graph G to a graph H is a bijective mapping $f : V_G \to V_H$ such that for any two vertices $u, v \in E_G$, we have $uv \in E_G$ if and only if $f(u)f(v) \in E_H$. If there exists an isomorphism from G to H, then we say that G is *isomorphic* to H, and we write $G \simeq H$.

Let u and v be two vertices of a graph G. A *path* between u and v is a sequence of distinct vertices $P = u_1u_2 \cdots u_p$ starting at $u_1 = u$ and ending at $u_p = v$, where u_iu_{i+1} is an edge of G for every $i = 1, \ldots, p-1$. If uv is an edge as well we obtain a *cycle*. If G contains no edges between non-consecutive vertices of P then we say that P is an *induced path* or *induced cycle* in G. Sometimes we fix an orientation of P. In that case we write $u_i\overrightarrow{P}u_j = u_iu_{i+1}\cdots u_j$ and $u_j\overleftarrow{P}u_i = u_ju_{j-1}\cdots u_i$ to denote the subpath from u_i to u_j, or from u_j to u_i, respectively. The *length* of a path is the number of its edges; the *length* of a cycle is defined in the same way. A graph is *connected* if there is a path between every pair of vertices, and a graph is *disconnected* if it is not connected. A *connected component* of G is a maximal connected subgraph of G.

Let $A_1, \ldots, A_p$ be a sequence of sets. For $i = 1, \ldots, p$, we use shorthand notation $A_{\leq i} = A_1 \cup \cdots \cup A_i$ and $A_{\geq i} = A_i \cup \cdots \cup A_p$.

2.1 Chordal, Interval, and Proper Interval Graphs

A graph isomorphic to the graph $K_{1,3} = (\{a, b_1, b_2, b_3\},\ \{ab_1, ab_2, ab_3\})$ is called a *claw* with *center* a and *leaves* b_1, b_2, b_3. A graph is called *claw-free* if it does not contain an induced subgraph isomorphic to a claw. An *asteroidal triple (AT)* in a graph G is a set of three mutually non-adjacent vertices u_1, u_2, u_3 such that G contains a path P_{ij} from u_i to u_j with $P_{ij} \cap N_G(u_k) = \emptyset$ for all distinct $i, j, k \in \{1, 2, 3\}$. A graph is called *AT-free* if it does not have an AT.

A graph is *chordal* if it contains no induced cycle of length at least 4. A chordal graph has at most n maximal cliques [14]. A graph is an *interval graph* if intervals of the real line can be associated with its vertices in such a way that two vertices are adjacent if and only if their corresponding intervals overlap. Interval graphs are a subclass of chordal graphs: a chordal graph is an interval graph if and only if it is AT-free [20]. In addition, the following characterization of interval graphs in terms of forbidden induced subgraphs is due to Lekkerkerker and Boland [20].

Theorem 2.1 ([20])**.** *A chordal graph is an interval graph if and only if it does not contain any of the graphs depicted in Figure 2 as an induced subgraph.*

Since we use the forbidden subgraph characterization of Theorem 2.1 in the proof of our first result, we give an exact description of the forbidden induced subgraphs. The graph F_1 is obtained from a claw by subdividing each of its edges exactly once, i.e., F_1 is the graph with vertices $x, y_1, y_2, y_3, z_1, z_2, z_3$, and edges $xy_1, y_1z_1, xy_2, y_2z_2, xy_3, y_3z_3$. The graph F_2 is obtained from a cycle $x_1 \cdots x_6x_1$ by adding the edges x_1x_3, x_3x_5, x_3x_6, as well as a new vertex y and the edge x_6y. For every $k \geq 4$, F_3^k is the graph obtained from a path $x_1 \cdots x_k$ by adding two new vertices y, z and the edge yz, as well as the edges x_iy for $i = 2, \ldots, k-1$. Finally, for every

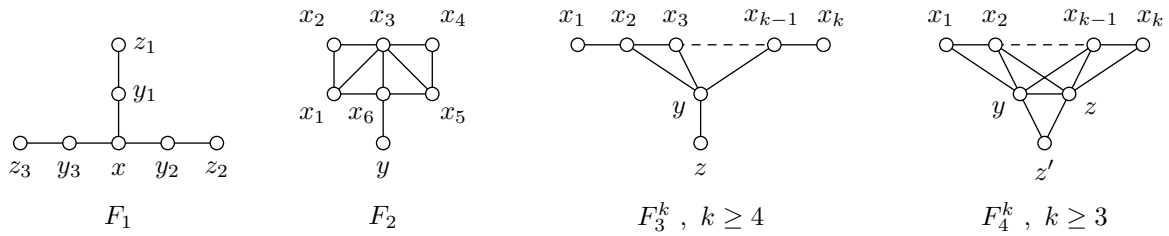

Figure 2: A chordal graph that is not interval contains one of these graphs as an induced subgraph.

$k \geq 3$, F_4^k is the graph obtained from a cycle $x_1 \cdots x_k z y x_1$ by adding the edges $x_i y$ and $x_i z$ for $i = 2, \ldots k-1$, as well as adding a new vertex z' and the edges yz' and zz'.

The following characterization of interval graphs is also well known and plays a central role in our results. A connected graph G with p maximal cliques is an interval graph if and only if there is an ordering $K_1, \ldots, K_p$ of the maximal cliques of G, such that for each vertex v of G, the maximal cliques containing v appear consecutively in the ordering. A path $P = K_1 \cdots K_p$ following such an ordering is called a *clique path* of G. We note that such a clique path P does not have to be unique, but every clique path of G has the same number p of bags by definition. By definition, for every vertex v of G, the maximal cliques containing v form a connected subpath in P. In this context, the maximal cliques of G are also called the *bags*[1] of P. A clique path of G has at most n bags and can be constructed in $\mathcal{O}(n+m)$ time (see e.g. [14]).

Given a clique path $P = K_1 \cdots K_p$ of an interval graph G, we say that K_i is the *first* bag in which a vertex u of G appears if $u \in K_i$ for $i = 1$ or $u \in K_i \setminus K_{i-1}$ for some $i \geq 2$. In the latter case, by the definition of a clique path, u is not in a bag K_h with $h \leq i-1$. If $u \in K_i$ for $i = p$ or $u \in K_i \setminus K_{i+1}$ for some $i \leq p-1$, then we say that K_i is the *last* bag in which u appears. In the latter case, u is not in a bag K_h with $h \geq i+1$, again by the definition of a clique path. We denote the index of the first bag of P in which u appears by $f_P(u)$ and the index of the last bag in which u appears by $l_P(u)$. Because bags of P correspond to maximal cliques, every bag K_i has the property that $i = f_P(u)$ for at least one vertex u, and $i = l_P(v)$ for at least one vertex v. Because G is connected, we also observe that each bag K_i contains at least one vertex from K_{i-1} for $i = 2, \ldots, p$. Note that, by the definition of a clique path, $K_{\leq i} \cap K_{i+1} = K_i \cap K_{i+1}$, and $K_{i+1} \setminus K_{\leq i} = K_{i+1} \setminus K_i$ for $i = 1, \ldots, p-1$.

An interval graph is *proper interval* if it has an interval representation in which no interval is properly contained in any other interval. An interval graph is a proper interval graph if and only if it is claw-free [28]. Equivalently, a chordal graph is a proper interval graph if and only if it is AT-free and claw-free. Chordal graphs, interval graphs, and proper interval graphs can all be recognized in $\mathcal{O}(n+m)$ time (see e.g. [3, 16]). Given a clique path P of an interval graph G and two vertices u and v of G, we say that u *transcends* v in P if $f_P(u) \leq f_P(v)$ and $l_P(u) > l_P(v)$. The following theorem [18], already implicit from several earlier works on proper interval graphs [6, 8, 23], shows that proper interval graphs do have a unique clique path. It will be used heavily in our proofs.

Theorem 2.2 ([6, 8, 18, 23])**.** *A connected chordal graph is a proper interval graph if and only*

[1]The term *bag* comes from tree and path decompositions. A clique path is a path decomposition where each bag is a maximal clique.

if it has a unique clique path P, and no vertex transcends any other vertex in P.

Two adjacent vertices u and v of a graph G are *twins* if $N_G(u) \cup \{u\} = N_G(v) \cup \{v\}$. For example, for every $k \geq 3$, the vertices y and z in the graph F_4^k in Figure 2 are twins. Let G be a connected proper interval graph with clique path $P = K_1 \cdots K_p$. Note that two vertices u and v of G are twins if and only if $f_P(u) = f_P(v)$ and $l_P(u) = l_P(v)$. We partition V_G into sets of twins. A vertex that has no twin appears in its twin set alone. We order the twin sets with respect to P, and label them $T_1, \ldots, T_s$, in such a way that $i < j$ if and only if for all $u \in T_i, v \in T_j$, it either holds that $f_P(u) < f_P(v)$, or else that $f_P(u) = f_P(v)$ and $l_P(u) < l_P(v)$. We call $T_1, \ldots, T_s$ the *ordered twin sets* of G. The ordered twin sets of G can be computed in $\mathcal{O}(n + m)$ time during the computation of the clique path. The following observation immediately follows from the definition of ordered twin sets and the definition of a clique path. Hence, this observation is even valid for interval graphs that are not proper.

Observation 2.3. *Let G be a connected proper interval graph with clique path $P = K_1 \cdots K_p$ and ordered twin sets $T_1, \ldots, T_s$. Then, for $h = 1, \ldots, s-1$, there exists a bag that contains twin sets T_h and T_{h+1}. Furthermore, if a bag contains twin sets T_b and T_c with $b < c$, then it contains twin sets $T_{b+1}, \ldots, T_{c-1}$ as well.*

2.2 Role Assignments

If r is a homomorphism from G to R and $U \subseteq V_G$, then we write $r(U) = \{r(u) \mid u \in U\}$. Recall that r is an R-role assignment of G if $r(N_G(u)) = N_R(r(u))$ for every vertex u of G. Graph R is called a *role graph* and its vertices are called *roles*. For a subset X of V_R, we write $r^{-1}(X) = \{u \in V_G \mid r(u) \in X\}$. If $X = \{x\}$, we simply write $r^{-1}(x)$ instead of $r^{-1}(\{x\})$. Let R' be a subgraph of R. Then $G[r^{-1}(V_{R'})]$ is the *preimage* of R' in G. We frequently make use of the following two known results.

Observation 2.4 ([12])**.** *Let G be a graph and let R be a connected graph such that G has an R-role assignment. Then each vertex $x \in V_R$ appears as a role of some vertex $u \in V_G$, i.e., for each vertex $x \in V_R$ there exists a vertex $u \in V_G$ such that $r(u) = x$. Furthermore, if $|V_G| = |V_R|$ then $G \simeq R$.*

Lemma 2.5 ([12])**.** *Let G and R be two graphs such that G has an R-role assignment r, and let $x_1 \cdots x_\ell$ be a path in R. Then for each $u \in V_G$ with $r(u) = x_1$ there exists a path $u_1 \cdots u_\ell$ in G, such that $u = u_1$ and $r(u_i) = x_i$ for $i = 1, \ldots, \ell$.*

Our first result, given in Theorem 2.6, shows that chordal graphs, interval graphs, and proper interval graphs are closed under role assignments. We need this result in Section 3.

Theorem 2.6. *Let G be a graph and let R be a connected graph such that G has an R-role assignment.*

(i) If G is a chordal graph, then R is a chordal graph.

(ii) If G is an interval graph, then R is an interval graph.

(iii) If G is a proper interval graph, then R is a proper interval graph.

Proof. Let r be an R-role assignment of G. If G is disconnected then clearly the restriction of r to each connected component G' of G is an R-role assignment of G'. Hence it suffices to show the result on a connected graph G.

Before we proceed with the proof of the theorem, we make the following useful observation. Due to Lemma 2.5 and since adjacent vertices of G cannot have non-adjacent roles in R, the preimage of an (induced) tree T contains an (induced) tree isomorphic to T. Due to the same reasons and because we only consider finite graphs, the preimage of an induced cycle C in R contains an induced cycle of length at least $|V_C|$.

Proof of (i). Suppose that G is chordal. If R is not chordal then R contains an induced cycle C of length at least four. By the observation above, the preimage of C in G contains an induced cycle of length at least four. This contradicts the assumption that G is chordal, so R must be chordal.

Proof of (ii). Suppose that G is an interval graph. Then G is chordal and from (i) we know that R is chordal. Assume for contradiction that R is not an interval graph. Then by Theorem 2.1, R contains one of the graphs F_1, F_2, F_3^k, or F_4^k, shown in Figure 2, as an induced subgraph.

Case 1: R contains F_1 as an induced subgraph.
Then, by the observation in the beginning of the proof, we find that G contains an induced F_1. Since G is an interval graph, this is not possible. Hence R does not contain F_1 as an induced subgraph.

Case 2: R contains F_2 as an induced subgraph.
Let u be a vertex of G with role x_6. Then, by repeatedly applying Lemma 2.5 with respect to the paths $x_6x_5\cdots x_1$ and x_1x_6 and the fact that G is finite, we find that u is on a cycle D in G of length $6d$ for some $d \geq 1$ such that the vertices of D have roles in repeated order $x_6, x_5, \ldots, x_1$. Note that D does not have to be an induced cycle of G. Also, u has a neighbor v in G with role y. Let s, s', t, t' be four vertices on D such that $ss'ut't$ forms a subpath of D and $r(s) = x_2$, $r(s') = x_1$, $r(t) = x_4$ and $r(t') = x_5$. Since D is a cycle, we can take a shortest path P_{st} in $G[V_D]$ from s to t not passing through a vertex from $\{s', t', u\}$. Suppose that P_{st} contains a neighbor u' of v. If v has more than one neighbor on P_{st}, then we choose u' to be the one closest to s on P_{st}. Vertex u' must have role x_6. Now we have a cycle $D' = u'vus's\overrightarrow{P_{st}}u'$, which we display in Figure 3 together with D. We observe that D' may not be induced. However, since we took P_{st} to be shortest and we took u' to be closest to s, the only chords possible on D' are incident to u or s'.

Suppose that u has a neighbor w on D' such that $w \notin \{s', v\}$. Assume that w is chosen closest to u'. Because u and u' have the same role, namely role x_6, we find that u and u' are not adjacent. This means that $w \neq u'$. Then the cycle $u'vuw\overrightarrow{P_{st}}u'$ is an induced cycle on at least four vertices. This is not possible, because G is chordal. Hence, the only neighbors of u on D' are s' and v.

Suppose that s' has a neighbor w' on D' such that $w' \notin \{s, u\}$. Assume that w' is chosen closest to u' (with possibly $w' = u'$). Then the cycle $u'vus'w'\overrightarrow{P_{st}}u'$ is an induced cycle on at least four vertices. This is not possible, because G is chordal. Hence, the only neighbors of s' on D' are s and u.

From the above we find that D' is induced. Since D' contains at least five vertices and G is chordal, this is not possible. We conclude that P_{st} does not contain any neighbor of v. Then, due to the paths P_{st}, $ss'uv$ and $tt'uv$, we find that $\{s, t, v\}$ forms an AT. This is not possible, because G is AT-free. Hence R does not contain an induced F_2.

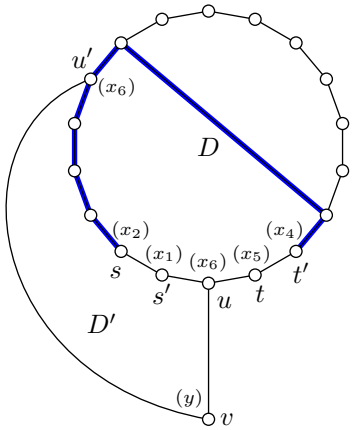

Figure 3: The cycles D and D', mentioned in Case 2. The roles of some vertices are shown between parentheses. The heavy edges indicate the path P_{st}.

Case 3: R contains an induced F_3^k for some $k \geq 4$.
Let t_2 be a vertex of G with role x_2. By repeatedly applying Lemma 2.5 with respect to the paths x_2x_3y and yx_2 and the fact that G is finite and chordal, we may assume without loss of generality that t_2 belongs to a triangle together with two vertices t_3 and u with roles x_3 and y, respectively. Let v be a neighbor of u with role z. Let t_1 be a neighbor of t_2 with role x_1. We apply Lemma 2.5 to find a path $t_3 \cdots t_k$ in G such that $r(t_i) = x_i$ for $i = 3, \ldots, k$. We consider the paths $t_1 \cdots t_k$, t_1t_2uv and $t_k \cdots t_3uv$ in order to find that $\{t_1, t_k, v\}$ is an AT in G; recall that the roles $x_1, \ldots, x_k, y, z$ form an induced F_3^k in R. Because G is AT-free, this is not possible. Hence R does not contain an induced F_3^k.

Case 4. R contains an induced F_4^k for some $k \geq 3$.
Consider the cycle $yzz'y$ in F_4^k. By using the same arguments as in Case 3, we find that G has three vertices u, v, w with roles y, z, z', respectively, that form a triangle. By using the same arguments as in Case 2, we also deduce that uv is an edge of a cycle D of G of length $d(k+2)$ for some $d \geq 1$ such that the vertices of D have roles in repeated order $z, y, x_1, \ldots, x_k$. Suppose that we chose u, v, w and D such that d is minimal over all triangles $uvwu$ with $r(u) = y$, $r(v) = z$ and $r(w) = z'$. Note that D does not have to be an induced cycle of G.

Let s and t be the two vertices such that $suvt$ forms a subpath of D; note that $r(s) = x_1$ and $r(t) = x_k$. Since D is a cycle, we can take a shortest path P_{st} in $G[V_D]$ from s to t not using the edge uv. Note that P_{st} is an induced path in G. Suppose that P_{st} contains a neighbor u' of w; a possible situation is depicted in Figure 4. Note that $u' \notin \{u, v\}$ because u and v are not on P_{st} by definition. Also, w is not adjacent to s because $r(s) = x_1$ and $r(w) = z'$ are not adjacent in R. Hence, u' can be chosen in such a way that u' is the only vertex on $s\overrightarrow{P_{st}}u'$ that is adjacent to w. Because u' is adjacent to w and w has role z', we find that u' must have role y or z. We consider each case.

Suppose that $r(u') = y$. Then u and u' have the same role. Consequently, u and u' are not adjacent. Let v' be the neighbor of u on $s\overrightarrow{P_{st}}u'$ that is closest to u'. Because u' and u are not adjacent, we find that $v' \neq u'$. Recall that u' is the only vertex on $s\overrightarrow{P_{st}}u'$ that is adjacent to w, and that P_{st} is an induced path in G. We then find that the cycle $uv'\overrightarrow{P_{st}}u'wu$ is a induced cycle on at least four vertices. This is not possible, because G is chordal. Hence $r(u') \neq y$.

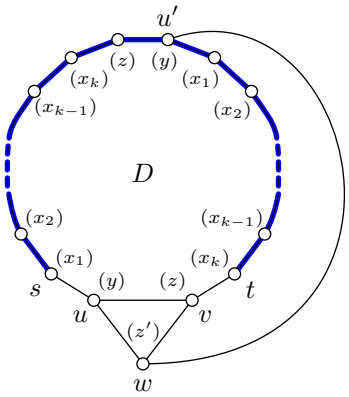

Figure 4: The cycle D mentioned in Case 4. The roles of the vertices are shown between parentheses. The heavy edges indicate the path P_{st}.

Suppose that $r(u') = z$. If uu' is an edge then we could take a shorter cycle D' of length $d'(k+2)$ with $d' < d$, and d would not be minimal. Hence uu' is not an edge. This means we can apply the same arguments as in the previous case, which leads us to conclude that $r(u') \neq z$. Because $r(u') \in \{y, z\}$, we get a contradiction. We conclude that P_{st} does not contain any neighbor of w.

If we consider the paths P_{st}, suw, and tvw, then we find that $\{s, t, w\}$ forms an AT in G. This is not possible, because G is an interval graph. Consequently, R does not contain an induced F_4^k.

We have shown that R, which is chordal, does not contain any of the graphs F_1, F_2, F_3^k, or F_4^k as an induced subgraph. By Theorem 2.1, R is an interval graph.

Proof of (iii). Suppose that G is a proper interval graph. Then G is interval, and from (ii) we have that R is an interval graph. If R has a claw as an induced subgraph then, by the observation in the beginning of the proof, its preimage in G contains a claw as an induced subgraph as well. Since G is proper interval and does therefore not contain a claw, R does not contain a claw either. Consequently, R is interval and claw-free, which means that R is a proper interval graph. □

Note that, for each of the three statements in Theorem 2.6, the reverse implication does not hold. In order to see this, let G be an induced cycle of length 6, and let R be a cycle of length 3.

3 Role Assignments on Proper Interval Graphs

We start with the following key result. Note that this result is easy to verify for paths.

Theorem 3.1. *Let G and R be two connected proper interval graphs such that G has an R-role assignment r. Let $P = K_1 \cdots K_p$ and $P' = L_1 \cdots L_q$ be the clique paths of G and R, respectively. Then the following holds: $r(K_i) = L_i$ for $i = 1, \ldots, q$, or else $r(K_i) = L_{q-i+1}$ for $i = 1, \ldots, q$.*

Proof. Let G and R be two connected proper interval graphs such that G has an R-role assignment r. Let P and P' be the clique paths of G and R, respectively. Let $P = K_1 \cdots K_p$, and

let $P' = L_1 \cdots L_q$. We will prove that $r(K_i) = L_i$ for $i = 1, \ldots, q$, or that $r(K_i) = L_{q-i+1}$ for $i = 1, \ldots, q$. We use induction on q.

Let $q = 1$. We apply the definition of a role assignment on the vertices in $K_1 \setminus K_2$ and find that $r(K_1) = L_1$ or $r(K_1) = L_q$.

Let $q \geq 2$. First suppose that $r(K_1) = L_1$. Let R' be the graph obtained from R after removing every vertex in $L_q \setminus L_{q-1}$. Let G' be the graph obtained from G after removing every vertex with role in $L_q \setminus L_{q-1}$. Because $q \geq 2$ and $r(K_1) = L_1$, we have not removed any vertex from K_1. Let F be the connected component of G' that contains K_1. Then the restriction r' of r to V_F is an R'-role assignment of F.

Let $P^* = J_1 \cdots J_s$ be the clique path of F. Because K_1 is an end bag of P, we find that $J_1 = K_1$ or $J_s = K_1$, say $J_1 = K_1$. Note that $L_1 \cdots L_{q-1}$ is the clique path of R'. Then, by the induction hypothesis, $r'(J_i) = L_i$ for $i = 1, \ldots, q-1$, or $r'(J_i) = L_{q-i}$ for $i = 1, \ldots, q-1$. Because $r(K_1) = L_1$ and $J_1 = K_1$, we find that $r'(J_i) = L_i$ for $i = 1, \ldots, q-1$.

Consider a bag L_i for an arbitrary $i \leq q-1$. By definition, L_i contains a vertex x that is not in L_{i+1}. Hence, x is not adjacent to any vertex in $L_q \setminus L_{q-1}$. Consequently, the vertex in J_i with role x is not adjacent to a vertex in G with role in $L_q \setminus L_{q-1}$. This means that J_i is a maximal clique in G.

Because F has a unique clique path, the maximal cliques of F appear in order $J_1, \ldots, J_s$ in P. No bag K_h of P is positioned between two bags J_i and J_{i+1} for some $1 \leq i \leq s-1$, because then $K_h \subseteq J_i$ or the vertices in $J_i \cap J_{i+1}$ transcend some vertex in K_h. The first case is not possible because K_h is a maximal clique of G. The second case is not possible due to Theorem 2.2. Recall that $K_1 = J_1$. Then, for $i = 1, \ldots, q-1$, we find that $K_i = J_i$, and consequently, $r(K_i) = r(J_i) = L_i$. As a result, all vertices in K_{q-1} with role in $L_{q-1} \cap L_q$ are in K_q, because they must have their required neighbors with roles in $L_q \setminus L_{q-1}$ in $K_{\geq q}$.

We will now show that K_q contains no vertex with role in $L_{\leq q-1} \setminus L_q$. In order to derive a contradiction, suppose that K_q contains a vertex u with role in $L_{\leq q-1} \setminus L_q$. Let $v \in K_{q-1}$ have a role in $L_{q-1} \cap L_q$. We deduced above that $v \in K_q$.

First suppose that $f_P(u) = q$. Because u has a role in $L_{\leq q-1} \setminus L_q$, we find that u cannot be in a bag with a vertex that has a role in $L_q \setminus L_{q-1}$. Hence $l_P(u) < l_P(v)$. Because $f_P(v) \leq q-1 < q = f_P(u)$, we find that v transcends u. This is not possible by Theorem 2.2. Now suppose that $f_P(u) \leq q-1$. Let u' be a vertex in $K_q \setminus K_{q-1}$. Then u' cannot have a role in $L_q \setminus L_{q-1}$ because u' is adjacent to u. Hence, the role of u' is in $L_{\leq q-1} \setminus L_q$, and we apply the arguments of the previous case with respect to u' instead of u to derive the same contradiction. We conclude that K_q contains no vertex with role in $L_{\leq q-1} \setminus L_q$.

Because K_q contains no vertex with role in $L_{\leq q-1} \setminus L_q$, we find that $r(K_q) \subseteq L_q$. If $r(K_q) \subset L_q$, then all the vertices of K_q must belong to K_{q+1}, because they all still need the vertices with roles in $L_q \setminus r(K_q)$ in their neighborhood. Then K_q would not be maximal. This is not possible. Hence, $r(K_q) = L_q$, as desired.

If $r(K_1) = L_q$ then we find that $r(K_i) = L_{q-i+1}$ for $i = 1, \ldots, q$ by exactly the same arguments. This finishes the proof of Theorem 3.1. □

Note that Theorem 3.1 is not valid for interval graphs, which can be seen with the following example. Let G be the path $u_1u_2u_3u_4$ to which we add a vertex u_5 with edge u_2u_5 and a vertex u_6 with edge u_3u_6. Let $P = K_1 \cdots K_5$ be a clique path of G with $K_1 = \{u_1, u_2\}$, $K_2 = \{u_2, u_5\}$, $K_3 = \{u_2, u_3\}$, $K_4 = \{u_3, u_6\}$ and $K_5 = \{u_3, u_4\}$. Let R be the 4-vertex path 1234. The unique clique path of R is $P' = L_1L_2L_3$ with $L_1 = \{1, 2\}$, $L_2 = \{2, 3\}$ and $L_3 = \{3, 4\}$. However,

we find that G has an R-role assignment r with $r(u_1) = r(u_5) = 1$, $r(u_2) = 2$, $r(u_3) = 3$, and $r(u_4) = r(u_6) = 4$.

Also note that we can apply Theorem 3.1 twice depending on the way the bags in the clique path of the proper interval graph G are ordered. Recall that the clique path of a proper interval graph is unique up to reversal. This leads to a rather surprising corollary that might be of independent interest.

Corollary 3.2. *Let G be a connected proper interval graph with clique path $P = K_1 \cdots K_p$, and let R be a connected graph. If G has an R-role assignment, then $R \simeq G[K_{\leq i}]$ and $R \simeq G[K_{\geq p-i+1}]$ for some $1 \leq i \leq p$.*

As an illustration of Corollary 3.2 we have indicated the two copies of R in G with bold edges in Figure 1. Due to Theorem 2.6 we do not need to restrict R to be a proper interval graph in the statement of the above corollary. Hence for any two connected graphs G and R, where G is proper interval with $|V_G| > |V_R|$, if G has an R-role assignment then G contains two (not necessarily vertex-disjoint) induced subgraphs isomorphic to R.

Theorem 3.1 only shows what an R-role assignment r of a proper interval graph G looks like at the beginning and end of the clique path of G. To derive our algorithm, we need to know the behavior of r in the middle bags as well should these bags exist. The main idea of our algorithm for proper interval graphs with "long" clique paths is to map the first q bags to R (assuming that R has q bags in its clique path), then identify a number of bags of G that can be "skipped" and then continue with finding an isomorphism between R and the subgraph of G induced by the next q bags of G, and so on. In order to be able to do this, we therefore give the following result, which is valid when R has at least three maximal cliques and the number of maximal cliques in G is not too small. The special cases when R has just one or two maximal cliques or G has only a few maximal cliques will be dealt with separately in the proof of Theorem 3.5.

Lemma 3.3. *Let G be a connected proper interval graph with clique path $P = K_1 \cdots K_p$. Let R be a connected proper interval graph with clique path $P' = L_1 \cdots L_q$ and ordered twin sets $X_1, \ldots, X_t$. Let r be an R-role assignment of G with $r(K_q) = L_q$. Let T be the subset of K_q that consists of all vertices with roles in X_t. Then the following holds if $q \geq 3$ and $p \geq 2q + 1$.*

(i) If there is a vertex in T not in K_{q+1}, then there exists an index $i \geq q + 1$ such that $K_{\geq q+1} \setminus K_q \subseteq K_{\geq i}$ and the restriction of r to $K_{\geq i}$ is an R-role assignment of $G[K_{\geq i}]$ with $r(K_i) = L_q$. Furthermore, if $i > q + 1$ then $r(K_h) \subseteq X_t$ for $h = q + 1, \ldots, i - 1$.

(ii) If all vertices in T are in K_{q+1}, then there exists an index $i \geq q+1$ such that $T = K_{i-1} \cap K_i$ and $T \cap K_{i+1} = \emptyset$, and the restriction of r to $K_{\geq i}$ is an R-role assignment of $G[K_{\geq i}]$ with $r(K_i) = L_q$.

Proof. Note that $t \geq 6$ because $q \geq 3$. We also observe that the restriction of r to $K_{\leq q}$ is an R-role assignment by Theorem 3.1.

Proof of (i). Suppose that $u \in T$ is not in K_{q+1}. Choose $i \geq q + 1$ to be the smallest index such that $|K_i| \geq |L_q|$. Hence, if $i > q + 1$ then $|K_h| < |L_q|$ for $q + 1 \leq h \leq i - 1$. Note that such an index i exists, because $p - q + 1 \geq 2q + 1 - q + 1 \geq q + 2$ and either $r(K_{p-q+1}) = L_q$ or $r(K_p) = L_q$, due to Theorem 3.1.

We first show that if $i > q + 1$ then $r(K_h) \subseteq X_t$ for $q + 1 \leq h \leq i - 1$. In order to derive a contradiction, suppose that there exists an index h' with $q+1 \leq h' \leq i-1$ such that $K_{h'}$ contains a vertex with role in $X_{\leq t-1}$. Choose h' in such a way that $r(K_h) \subseteq X_t$ for $q + 1 \leq h \leq h' - 1$.

Claim 1. $f_P(v) = h'$ for all $v \in K_{h'}$ with $r(v) \in X_{\leq t-1}$.

We prove Claim 1 as follows. Let $v \in K_{h'}$ with $r(v) \in X_{\leq t-1}$. First suppose that $v \in K_{\leq q}$. Since $r(v) \in X_{\leq t-1}$, we find that $r(v)$ is adjacent to some vertex in $X_{\leq t-1}$ to which $r(u)$ is not adjacent, i.e., $f_{P'}(r(v)) < f_{P'}(r(u)) = q$. By Theorem 3.1 and our assumption that $r(K_q) = L_q$, we have $r(K_a) = L_a$ for $a = 1, \ldots, q$. Hence, $f_P(v) < f_P(u)$. Since $l_P(u) = q$ and $l_P(v) \geq q + 1$, this means that v transcends u. This is not possible due to Theorem 2.2. We conclude that v does not appear in $K_{\leq q}$. By our choice of h', we then find that v is in $K_{h'} \setminus K_{\leq h'-1}$, so $f_P(v) = h'$ indeed. This completes the proof of Claim 1.

We claim that $K_{h'}$ contains a vertex with role in X_t. If $h' \geq q + 2$ then this claim follows from the definition of a clique path, which implies that $K_{h'-1} \cap K_{h'} \neq \emptyset$, and our choice of h', which implies $r(K_{h'-1}) \subseteq X_t$. Suppose that $h' = q + 1$. If all vertices of T are not in K_{q+1}, then there must be a vertex $v^* \in K_{q+1} \cap K_q$ with role in $X_{\leq t-1}$. Then $f_P(v^*) \leq q \leq h' - 1$ and this is not possible due to Claim 1. Hence, indeed, $K_{h'}$ contains a vertex with role in X_t. Consequently, we find that $r(K_{h'}) \subset L_q$.

By the definition of a clique path, $K_{h'} \setminus K_{h'+1} \neq \emptyset$. Let $u' \in K_{h'} \setminus K_{h'+1}$, so $l_P(u') = h'$. We claim that $r(u') \in X_t$. If not, then $r(u') \in X_{\leq t-1}$, and consequently, $f_P(u') = h'$ by Claim 1. However, we have $r(K_{h'}) \subset L_q$ and $|K_{h'}| < |L_q|$. This, together with $f_P(u') = l_P(u') = h'$, implies that u' misses at least one role of L_q in its neighborhood. This is not possible. Hence, $r(u') \in X_t$ indeed. We need this vertex u' in the proof of the following claim, and also in the rest of the proof.

Claim 2. There exists a vertex in $K_{h'}$ with role in X_{t-1}.

We prove Claim 2 as follows. In order to derive a contradiction, suppose that there is no vertex in $K_{h'}$ with role in X_{t-1}. Let v^* be a vertex in $K_{h'}$ with $r(v^*) \in X_{\leq t-1}$. Then we find that $r(v^*) \in X_{\leq t-2}$. Let s be a neighbor of v^* with role in X_{t-1}. Because $f_P(v^*) = h'$ by Claim 1 and $s \notin K_{h'}$, we find that $l_P(v^*) \geq h' + 1$ and $f_P(s) \geq h' + 1$. Since $r(v^*)$ belongs to $X_{\leq t-2}$, and $r(s)$ and $r(u')$ are both in $X_{\geq t-1}$, there exists a neighbor v' of v^* with $r(v')$ adjacent to neither $r(u')$ nor $r(s)$ in R. Hence v' is adjacent to neither u' nor s in G. Since $l_P(u') = h'$ and $f_P(s) \geq h' + 1$, we find that u' and s are not adjacent. However, then G has an induced claw with center v^* and leaves s, u', v', which contradicts the assumption that G is a proper interval graph. Hence we have proven Claim 2.

By Claim 2, there exists a vertex $v \in K_{h'}$ with $r(v) \in X_{t-1}$. Because $|K_{h'}| < |L_q|$, there exists a role $x \in L_q$ that is not in $r(K_{h'})$. This means that v is in $K_{h'+1}$; otherwise v will not get its required neighbor with role x. Let w be this neighbor, so $r(w) = x$.

Claim 3. There is no neighbor s of v that has $f_P(s) \geq h' + 1$ and $r(s) \in X_t$.

We prove Claim 3 as follows. Suppose that v is adjacent to such a vertex s. Then, since $r(v)$ belongs to $X_{\leq t-1}$, and $r(s)$ and $r(u')$ are both in X_t, there exists a neighbor v' of v with $r(v')$ adjacent to neither $r(u')$ nor $r(s)$. Hence v' is adjacent to neither u' nor s. Since $l_P(u') = h'$ and $f_P(s) \geq h' + 1$, we find that u' and s are not adjacent. However, then G has an induced claw with center v and leaves s, u', v', which contradicts the assumption that G is a proper interval graph. Hence we have proven Claim 3.

Claim 3 implies that $x \in X_{\leq t-1}$, because v is adjacent to w with $f_P(w) \geq h' + 1$ and $r(w) = x$. Let $z_\ell \in X_1$ and let $Q' = z_1 \cdots z_\ell$, with $x = z_1$, be a shortest path in R from x to a role $z_\ell \in X_1$. By Lemma 2.5 we find that G contains a path $Q = t_1 \cdots t_\ell$ with $t_1 = w$ such that $r(t_i) = z_i$

for $i = 1, \ldots, \ell$. Because Q' is shortest, we find that $t_2\overrightarrow{Q}t_\ell$ contains no vertex with role in L_q. Because $K_{h'}$ only contains vertices with roles in L_q and $w \notin K_{h'}$, this implies that $f_P(t_i) \geq h'+1$ for $i = 1, \ldots, \ell$.

Because w has role x and $x \in L_q$, we find that all roles of X_t appear as roles of neighbors of w. Because $r(u') \in X_t$, this means that w has a neighbor w' with $r(w') = r(u')$. Note that $f_P(w') \geq h' + 1$, because $f_P(w) \geq h' + 1$. Because $l_P(u') = h'$, this implies that u' and w' are two different vertices that are not adjacent.

We claim that $\ell \geq 3$. In order to see this, we first observe that v is not adjacent to a vertex with role in X_1. This is because $r(v) \in X_{t-1}$ is already adjacent to a role in X_t, namely $r(u')$, and then $q = 2$, whereas we assumed that $q \geq 3$. Suppose that $\ell = 1$. Then $x = z_1 \in X_1$, and v is adjacent to a vertex, namely w, with role $r(w) = x \in X_1$. This is not possible, as we just observed. Suppose that $\ell = 2$. Then $r(t_2) = z_2 \in X_1$, and we find that v is not adjacent to t_2, again due to the above observation. Since $r(w') \in X_t$, we also find that t_2 and w' are not adjacent. By Claim 3, v and w' are not adjacent. Then G has an induced claw with center w and leaves v, w', t_2, which contradicts the assumption that G is a proper interval graph. So, $\ell \geq 3$ indeed.

We claim that t_ℓ, u', w' form an AT in order to get a contradiction (recall that a proper interval graph is AT-free). To show this we first prove that t_ℓ, u', w' are three different vertices that are pairwise non-adjacent. We already deduced that u' and w' are two different non-adjacent vertices. Because $l_P(u') = h'$ and $f_P(t_\ell) \geq h' + 1$, we also find that u' and t_ℓ are two different non-adjacent vertices. As $t > 1$, vertices t_ℓ and w' with roles in X_1 and X_t, respectively, are different and non-adjacent.

Since Q' is a shortest path in R from x to a vertex in X_1, we find that Q neither contains v nor w', because these vertices have a role in L_q. Recall that w' has a role in X_t and that w, the first vertex of Q, has role in $X_{\leq t-1}$. Then we can also use the fact that Q' is a shortest path to deduce that w' has no neighbor on $t_2\overrightarrow{Q}t_\ell$.

In order to have a path from u' to t_ℓ, we claim that v is adjacent to t_2. Suppose not. By Claim 3, we find that v and w' are not adjacent. Since w' has no neighbor on $t_2\overrightarrow{Q}t_\ell$, vertices t_2 and w' are not adjacent. However, then G has an induced claw with center w and leaves t_2, v, w', which contradicts the assumption that G is a proper interval graph. Hence, v and t_2 are adjacent. This implies that G indeed contains such a path, namely the path $u'vt_2\overrightarrow{Q}t_\ell$.

We consider the three paths $u'vww'$, $u'vt_2\overrightarrow{Q}t_\ell$ and $w'w\overrightarrow{Q}t_\ell$. In order to finish our claim that $\{t_\ell, u', w'\}$ is an AT, we show that t_ℓ has no neighbor on vw, that u' has no neighbor on $w\overrightarrow{Q}t_{\ell-1}$, and that w' has no neighbor on $vt_2\overrightarrow{Q}t_{\ell-1}$.

Consider t_ℓ. Recall that v is not adjacent to a vertex with role in X_1. This is because $r(v) \in X_{t-1}$ is already adjacent to a role in X_t, namely $r(u')$, and then $q = 2$, whereas we assumed that $q \geq 3$. Hence t_ℓ with role $z_\ell \in X_1$ is not adjacent to v. Since Q is a shortest path in R, we find that Q' is an induced path in G. We already showed that $\ell \geq 3$, i.e., Q' contains at least three vertices. Hence we find that t_ℓ is not adjacent to $w = t_1$. Consider u'. Because $l_P(u') = h$ and each vertex in $w\overrightarrow{Q}t_{\ell-1}$ appears for the first time in bag $K_{h'+1}$ or later, we find that u' has no neighbor on $w\overrightarrow{Q}t_{\ell-1}$. Consider w'. We already deduced that w' and v are not adjacent, and that w' has no neighbor on $t_2\overrightarrow{Q}t_{\ell-1}$.

The above indeed implies that the vertices t_ℓ, u', w' form an AT, contradicting the assumption that G is proper interval. We conclude that $r(K_h) \subseteq X_t$ for $q + 1 \leq h \leq i - 1$.

Because $r(K_h) \subseteq X_t$ for $q + 1 \leq h \leq i - 1$, every vertex v with $q + 1 \leq f_P(v) \leq i - 1$ has a

role in X_t and still needs a neighbor with role in $X_{\leq t-1}$. Hence $K_{\geq q+1} \setminus K_q \subseteq K_{\geq i}$.

We claim that K_i contains a vertex with role in X_t. If $i \geq q+2$, then any vertex in $K_{i-1} \cap K_i$ has role in X_t. Hence, for $i \geq q+2$, this claim is true. Suppose that $i = q+1$. Let $s^* \in K_q \cap K_{q+1}$. If $s^* \notin T$, then every vertex of T is in K_{q+1}, as otherwise s^* will transcend such a vertex (we can prove this using the same arguments as in the proof of Claim 1). So, also for $i = q+1$, the claim is true.

Because $r(K_i) \cap X_t \neq \emptyset$ and $|K_i| \geq |L_q|$, we find that $r(K_i) = L_q$, and hence $|K_i| = |L_q|$. Since all vertices in K_i with role in $X_{\leq t-1}$ are not in $K_{\leq i-1}$, we find that the restriction of r to $K_{\geq i}$ is an R-role assignment. This proves (i).

Proof of (ii). Suppose that all vertices in T are in K_{q+1}. Because $t > 1$, we find that $L_1 \cap T = \emptyset$. Let $i \geq q+1$ be such that K_i contains a vertex $u \in T$, whereas K_{i+1} does not contain u, so $l_P(u) = i$. Since $L_1 \cap T = \emptyset$ and either $r(K_{p-q+1}) = L_1$ or $r(K_p) = L_1$ due to Theorem 3.1, such an index i exists. We choose $i \geq q+1$ to be the smallest index with this property, i.e., $T \subseteq K_h$ for $q+1 \leq h \leq i$. We observe that $r(K_i) \subseteq L_q$, because $T \subseteq K_i$.

Let C be the set of vertices in $K_q \cap K_i$ with role in $X_{\leq t-1}$. We claim that C is empty. In order to prove this, suppose that there exists a vertex $u^* \in C$. Using the same arguments as in Claim 1 of the proof of (i), we obtain $l_P(u^*) = l_P(u) = i$ and without loss of generality that $r(u^*) \in X_{t-1}$.

Let $L_q = X_b \cup \cdots \cup X_t$ for some $b \leq t-1$. Let $v^* \in K_i \setminus K_{i-1}$, so $f_P(v^*) = i$. Because $T \subset K_i$, we find that $r(v^*) \in X_{\geq b} \cap X_{\leq t-1}$. Then $v^* \in K_{i+1}$ is adjacent to a vertex s with $r(s) \in X_{b-1}$. Since $f_P(v) = i$ and a vertex with role in X_{b-1} is not adjacent to a vertex in T (which has a role in X_t), we find that $l_P(v) \geq i+1$ and $l_P(s) \geq i+1$. Since $r(u^*) \in X_{\leq t-1}$, roles $r(u^*)$ and $r(s)$ are adjacent. Hence, s is adjacent to a vertex s' with $r(s') = r(u^*)$. Because $l_P(u^*) = i$ and $f_P(s) \geq i+1$, we find that u^* and s' are two different and non-adjacent vertices. Let s'' be a neighbor of s' with $r(s'') \in X_t$. Then $f_P(s'') \geq i+1$, and we find that s'' and u^* are also non-adjacent.

Let $z_\ell \in X_1$ and let $Q' = z_1 \cdots z_\ell$ with $z_1 = x$ be a shortest path from x to z_ℓ in R. By Lemma 2.5 we find that G contains a path $Q = t_1 \cdots t_\ell$ with $t_1 = s$ such that $r(t_i) = z_i$ for $i = 1, \ldots, \ell$.

We claim that $\ell \geq 2$. Suppose that $\ell = 1$. Then vertex s' with $r(s') = r(u^*) \in X_{t-1}$ is adjacent to a vertex with role in X_1, namely s. This means that $q = 2$, whereas we assumed that $q \geq 3$. Hence, $\ell \geq 2$ indeed.

We claim that s' and v^* are adjacent. Suppose not. Recall that $r(s) \in X_{b-1}$, and that s' and v^* both have a role in $X_{\geq b}$. Then, because Q' is a shortest path, we find that t_2 is neither adjacent to s' nor to v^*. This means that G contains an induced claw with center s and leaves s', t_2, v^*, contradicting the assumption that G is proper interval. We conclude that indeed $s'v^*$ is an edge.

We claim that s'' and v^* are not adjacent. Suppose that they are. Recall that $r(v^*) \in X_{\leq t-1}$, and that u^* and s'' both have a role in X_t. This means that v^* is adjacent to a vertex v' with role in $X_{\leq b-1}$, whereas u^*, s'' are both not adjacent to such a vertex v'. Since u^* and s'' are not adjacent, we find that G contains an induced claw with center v^* and leaves s'', u^*, v', contradicting the assumption that G is proper interval. Hence s'' and v^* are not adjacent.

We claim that s and s'' are not adjacent. Suppose that they are. Then G contains an induced claw with center s and leaves s'', t_2, v^*. This is not possible, as we saw before. Hence, indeed s and s'' are not adjacent.

We now consider the three paths $u^*v^*s's''$, $u^*v^*st_2$, and $t_2ss's''$ and deduce from the above claims that t_2, u^*, s'' form an AT, contradicting the assumption that G is proper interval. We conclude that $C = \emptyset$.

Because $C = \emptyset$, we have $K_q \cap K_i = T$. This means that every $v \in K_i \setminus T$ has $f_P(v) \geq q+1$. Because $T \subset K_h$ for $q+1 \leq h \leq i$, every v with $q+1 \leq f_P(v) \leq i-1$ belongs to K_i and still needs all its neighbors with roles in $X_{\leq b-1}$. Hence the restriction of r to $K_{\leq i}$ is an R-role assignment. Because $T \subset K_i$, we deduce that $r(K_i) = L_q$ and $r(K_{i+1}) = L_{q-1}$ after applying Theorem 3.1. Hence $T \cap K_{i+1} = \emptyset$. This completes the proof of (ii). □

Let G and R be two connected proper interval graphs with clique paths $P = K_1 \cdots K_p$ and $P' = L_1 \cdots L_q$, respectively. A mapping $r : K_{\leq i} \to V_R$ for some $1 \leq i \leq p$ is a *starting R-role assignment* of $G[K_{\leq i}]$ if for all $u \in K_{\leq i} \setminus K_{i+1}$ we have that $r(N_G(u)) = N_R(r(u))$, and for all $u \in K_i \cap K_{i+1}$ we have that $r(N_G(u)) \subseteq N_R(r(u))$. Note that a starting R-role assignment of $G[K_{\leq i}]$ is an R-role assignment of G if and only if $i = p$. The idea of our algorithm will be to extend a given starting role assignment of $G[K_{\leq i}]$ to a starting role assignment of $G[K_{\leq i+1}]$ at each step i.

Let $1 \leq i \leq p$, and let r be a starting R-role assignment of $G[K_{\leq i}]$. We say that a vertex $u \in K_i \cap K_{i+1}$ *misses* role $x \in V_R$ if x is a neighbor of $r(u)$ in R, and x is not a role of a neighbor of u in $K_{\leq i}$. Equivalently, x is a *missing role* of u. Let $X_1, \ldots, X_t$ be the ordered twin sets of R. We denote the set of missing roles of u that are in X_k by $M_k(u)$. We call such a set a *missing role set* of u. Note that missing role sets are only defined for vertices in $K_i \cap K_{i+1}$, since by the definition of a starting role assignment, every vertex in $K_{\leq i} \setminus K_{i+1}$ has the required roles in its neighborhood. We say that r can be *finished by* r^* if r^* is an R-role assignment of G with $r^*(u) = r(u)$ for all $u \in K_{\leq i}$.

The following lemma is important for the correctness and the running time analysis of our algorithm.

Lemma 3.4. *Let G and R be two connected proper interval graphs, let $P = K_1 \cdots K_p$ be the clique path of G, and let $X_1, \ldots, X_t$ be the ordered twin sets of R. Let $r : K_{\leq i} \to V_R$ be a starting R-role assignment of $G[K_{\leq i}]$ for some $1 \leq i \leq p$. If r can be finished by an R-role assignment of G, then the following three statements hold.*

(i) $M_k(u) \subseteq M_k(v)$ for every two vertices $u, v \in K_i \cap K_{i+1}$ with $f_P(u) \leq f_P(v)$, for $1 \leq k \leq t$.

(ii) When a vertex $u \in K_{i+1} \setminus K_i$ gets a role, then its missing role sets can be computed in $\mathcal{O}(\deg_G(u))$ time in total.

(iii) When a vertex $v \in K_{i+1} \setminus K_i$ gets a role, then all the missing role sets of the vertices in K_{i+1} that have a role already can be updated in $\mathcal{O}(\deg_G(v))$ time in total.

Proof. We prove the three statements separately.

Proof of (i). Let $k \in \{1, \ldots, t\}$. Let u and v be two vertices in $K_{\leq i} \cap K_{i+1} = K_i \cap K_{i+1}$ with $f_P(u) \leq f_P(v)$ and let x be a role in $M_k(u)$. We will show that $x \in M_k(v)$. Since u misses role $x \in X_k$, and r can be finished by an R-role assignment, we know that u has a neighbor w in $K_{\geq i+1}$ that must get role x. Since $f_P(u) \leq f_P(v)$ and G is a proper interval graph, we know that $l_P(u) \leq l_P(v)$. Hence $N_G(u) \cap K_{\geq i+1} \subseteq N_G(v) \cap K_{\geq i+1}$. Consequently v is also adjacent to w. This means that v also misses role x, unless v already has a neighbor w' in $K_{\leq i}$ with $r(w') = x$.

Assume that v has such a neighbor w'. Since u misses role x, we find that u is not adjacent to w'. Because $u \in K_i \cap K_{i+1}$ and $w \in K_{\leq i}$, this means that $l_P(w') < f_P(u)$. Because v is adjacent to w', we find that $f_P(v) \leq l_P(w')$. Then we obtain $f_P(v) \leq l_P(w) < f_P(u)$. However, this is not possible because $f_P(u) \leq f_P(v)$ by our assumption. We conclude that v cannot have a neighbor $w' \in K_{\leq i}$ with role x. So, indeed v misses role x as well. This proves (i).

Proof of (ii). Observe first that when the twin sets of R are initially computed, we can store for each role which twin set it belongs to. Recall that these twin sets form a partition of V_R. Let $u \in K_{i+1} \setminus K_i$ get a role. We run through the neighbors of u in G and mark the roles that any of these have received. This takes $\mathcal{O}(\deg_G(u))$ time. Then we run through the neighbors of $r(u)$ in R to find the twin sets that these neighbors belong to. This takes $\mathcal{O}(\deg_R(r(u)) = \mathcal{O}(\deg_G(u))$ time, because $\deg_R(r(u)) \leq \deg_G(u)$ by the definition of a role assignment. Let these indices be $k_1, \ldots, k_j$. Now we run through the roles of each twin set $X_{k_1}, \ldots, X_{k_j}$ and put the unmarked roles into sets $M_{k_1}(u), \ldots, M_{k_j}(u)$, respectively. This takes $\mathcal{O}(\Sigma_{i=1}^{j}|X_{k_i}|)$ time. Observe that $r(u)$ is adjacent to every role in $X_{k_1} \cup \cdots \cup X_{k_j}$ because of the definition of a twin set and because u misses at least one vertex from each of these twin sets. Consequently, and since all twin sets are mutually disjoint, we find that $\Sigma_{i=1}^{j}|X_{k_i}| \leq \deg_R(r(u))$. Because $\deg_R(r(u)) \leq \deg_G(u)$, this means that the last step also takes $\mathcal{O}(\deg_G(u))$ time. Note that we did not consider any empty missing role set of u. We conclude that $M_1(u), \ldots, M_t(u)$ can be computed in $\mathcal{O}(\deg_G(u))$ time in total. This proves (ii).

Proof of (iii). For each role x of R we store the vertices of K_{i+1} that have a role already and that miss role x in a doubly linked list $L(x)$. This list can be created and updated without adding to the running time described in the proof of (ii); every time a vertex u receives its role and its missing role sets are computed, we can add u to the list $L(x)$ of every role x that u misses. Note that any missing role x of u appears in exactly one missing role set $M_k(u)$. Using this fact, we keep pointers in both directions between the corresponding entries of these lists, i.e., u in $L(x)$ points to x in $M_k(u)$ and vice versa.

Suppose that we decide to assign role y to a vertex $v \in K_{i+1} \setminus K_i$. Because v is in K_{i+1}, we find that v is adjacent to every vertex in $K_i \cap K_{i+1}$. Recall that the vertices in $K_{\leq i} \setminus K_{i+1}$ do not have any missing roles by definition of a starting role assignment. Hence $L(y)$ contains at most $\deg_G(v)$ vertices. Now we simply run through the vertices in the list $L(y)$. For each vertex u in this list, we follow the pointer that will lead us exactly to role y in a missing role set $M_k(u)$ of u. We delete y from $M_k(u)$, and we delete u from $L(y)$. This takes constant time for each vertex in $L(y)$. Because there are at most $\deg_G(v)$ vertices in $L(y)$, the claimed running time follows. This proves (iii), and the proof of Lemma 3.4 has now been completed. □

We are now ready to present our main result, namely a linear time algorithm that solves the Role Assignment problem on input pairs (G, R), where G is a proper interval graph, and R is an arbitrary connected graph. We emphasize that connectivity is the only restriction imposed on R. As a consequence, the running time of our algorithm only depends on G.

Theorem 3.5. Role Assignment *can be solved in $\mathcal{O}(n+m)$ time on input pairs (G, R) where G is a proper interval graph with n vertices and m edges, and R is a connected graph.*

Proof. We present an algorithm with running time $\mathcal{O}(n + m)$ that takes as input a connected proper interval graph G with n vertices and m edges and a connected graph R, and decides

whether G has an R-role assignment. If G is disconnected, then we run the described algorithm on each connected component of G separately, still giving a total running time of $\mathcal{O}(n+m)$.

We first check in constant time whether $|V_R| \leq n$ and $|E_R| \leq m$. If not, then the answer is `NO` as a result of Observation 2.4.

Suppose that $|V_R| \leq n$ and $|E_R| \leq m$. We check whether R is a proper interval graph. This can be done in $\mathcal{O}(|V_R| + |E_R|) = \mathcal{O}(n+m)$ time; see e.g. [3, 16]. If R is not a proper interval graph, then the answer is `NO` due to Theorem 2.6.

Suppose that R is a proper interval graph. Recall that G is connected, and let $P = K_1 \cdots K_p$ be the clique path of G. Our algorithm starts by constructing P and ordering the vertices of G as $w_1, \ldots, w_n$ such that $f_P(w_1) \leq \ldots \leq f_P(w_n)$, and $l_P(w_j) \leq l_P(w_{j+1})$ whenever $f_P(w_j) = f_P(w_{j+1})$, for $j = 1, \ldots, n-1$. Note that, as a consequence, $l_P(w_1) \leq \ldots \leq l_P(w_n)$, due to Theorem 2.2. Recall that P can be computed in $\mathcal{O}(n+m)$ time; during this computation, the ordering $w_1, \ldots, w_n$ can easily be generated within the same running time. Then we compute the clique path of R. Let R have clique path $P' = L_1 \cdots L_q$ and ordered twin sets $X_1, \ldots, X_t$. Because $|V_R| \leq n$ and $|E_R| \leq m$, we can compute P' and the ordered twin sets in $\mathcal{O}(|V_R|+|E_R|) = \mathcal{O}(n+m)$ time. Since Lemma 3.3 applies only when $q \geq 3$, we distinguish between the cases where $q = 1$, $q = 2$, and $q \geq 3$. These cases result, in principle, in three different algorithms, depending on the number of bags of R.

Case 1: $q = 1$.
In this case R is a complete graph. We check whether $|K_1| = |V_R|$. If not, then we output `NO` due to Theorem 3.1. Otherwise, we give each vertex in K_1 a different role. This yields a starting R-role assignment r of $G[K_1]$.

Suppose that $i \geq 1$ and that we have extended r to a starting R-role assignment of $G[K_{\leq i}]$. Let $u_1, \ldots, u_b$ be the ordering of the vertices of $K_i \cap K_{i+1}$ obtained from our initial ordering of the vertices of V_G. Hence, $f_P(u_a) \leq f_P(u_{a+1})$ and $l_P(u_a) \leq l_P(u_{a+1})$ for $a = 1, \ldots, b-1$. We assign different roles to the vertices of $K_{i+1} \setminus K_i$, where we first use the roles of $M_1(u_a)$ in an arbitrary order before using any roles of $M_1(u_{a+1})$ for $a = 1, \ldots, b-1$. This is because $l_P(u_a) \leq l_P(u_{a+1})$ for $a = 1, \ldots, b-1$. Consequently, we must first ensure that u_a gets its required roles in the neighborhood before considering u_{a+1}. We may apply this greedy way of assigning roles, because Lemma 3.4 (i) implies that $M_1(u_a) \subseteq M_1(u_{a+1})$ for $a = 1, \ldots, b-1$ if r can be finished by an R-role assignment of G.

If we have used all the roles and there are still vertices in K_{i+1} with no role yet, then we output `NO`. Suppose that all vertices in K_{i+1} received a role. Then we verify if the resulting mapping is a starting R-role assignment of $G[K_{\leq i+1}]$. If not, then we output `NO`, because the restriction of any R-role assignment of G to $K_{\leq i+1}$ is a starting role assignment of $G[K_{\leq i+1}]$. Suppose that we obtained a starting R-role assignment of $G[K_{\leq i+1}]$. We stop if $i+1 = p$, because a starting R-role assignment of $G[K_{\leq p}] = G$ is an R-role assignment of G. If $i+1 < p$, then we repeat the above procedure with $i = i+1$.

We are left to show that the total running time of Case 1 is $\mathcal{O}(n+m)$. We first recall that we never have to recompute an ordering of the vertices in a bag; we always use the initial ordering. This means that the computations necessary to assign roles to the vertices in G are of the following form:

(a) computing and updating missing role sets;

(b) checking if an obtained mapping is a starting role assignment.

By Lemma 3.4 (ii), we can compute the missing role set of a vertex u in $\mathcal{O}(\deg(u))$ time after we assign a role to u. Suppose that this happens when considering bag K_{i+1}. By Lemma 3.4 (iii), we can update the missing role sets of the vertices in K_{i+1} that have a role already in $\mathcal{O}(\deg(u))$ time as well. Because we only assign a role to a vertex u once, the total time to compute and update the missing role sets of all the vertices of G is $\sum_{u \in V_G} \mathcal{O}(\deg(u)) + \sum_{u \in V_G} \mathcal{O}(\deg(u)) = \mathcal{O}(m)$. Simultaneously we keep track of vertices whose missing role sets all get empty. We do this, because checking if an obtained mapping is a starting role assignment of $K_{\leq i+1}$ means checking if every vertex in $K_{i+1} \setminus K_{i+2}$ has an empty missing role set. Recall that every vertex is in $K_{i+1} \setminus K_{i+2}$ for exactly one value of i. This means that checking whether the obtained mappings are starting role assignments takes $\mathcal{O}(n)$ time in total. The total $\mathcal{O}(n+m)$ running time follows.

Case 2: $q = 2$.
In this case R has exactly two maximal cliques. We check in linear time whether $G[K_{\leq 2}] \simeq R$. If not, then we output `NO` due to Theorem 3.1. Otherwise, we assume without loss of generality that $G[K_{\leq 2}]$ has an R-role assignment r with $r(K_1) = L_1$ and $r(K_2) = L_2$. We note that r is a starting R-role assignment of $G[K_{\leq 2}]$.

From the above, we may assume that $i \geq 2$ and that we have extended r to a starting R-role assignment of $G[K_{\leq i}]$. We must extend r to $K_{\leq i+1}$ by assigning roles to $K_{i+1} \setminus K_i$. We say that a choice of r on $K_{i+1} \setminus K_i$ is *right* if our extension of r to $K_{\leq i+1}$ eventually leads to an R-role assignment of G.

Because R has exactly two maximal cliques L_1 and L_2, it has exactly three twin sets, namely $X_1 = L_1 \setminus L_2$, $X_2 = L_1 \cap L_2$, and $X_3 = L_2 \setminus L_1$. For $j = 1, 2, 3$, let $X'_j = r(K_i \cap K_{i+1}) \cap X_j$. Let $Y = r(K_{i+1} \setminus K_i)$ be the set of roles assigned to the vertices of $K_{i+1} \setminus K_i$ after we have extended r to $K_{\leq i+1}$. Before we explain how to do this, we first prove the following two claims.

Claim 1. If $X'_1 \cup X'_3 = \emptyset$, then all vertices in $K_i \cap K_{i+1}$ either all miss all roles in X_1 and no role in X_3, or else they all miss all roles in X_3 and no role in X_1.

We prove Claim 1 as follows. Assume that r is a right choice and that $X'_1 \cup X'_3 = \emptyset$. Then all vertices in $K_i \cap K_{i+1}$ have role in X'_2. First suppose that $K_i \cap K_{i+1}$ contains a vertex u, such that u has a neighbor $v \in K_{\leq i}$ with $r(v) \in X_1$ and a neighbor $w \in K_{\leq i}$ with $r(w) \in X_3$. Because a vertex with role in X_1 is not adjacent to a vertex with role in X_3, there is no bag containing both v and w. Then we may without loss of generality assume that $l_P(v) < f_P(w)$. Because $X'_3 = \emptyset$, we find that $l_P(u) \geq i + 1 > l_P(w)$. Because uv is an edge, there is a bag containing u and v. Hence $f_P(u) \leq l_P(v)$. This means that $f_P(u) \leq l_P(v) < f_P(w) \leq l_P(w) < l_P(u)$. Consequently, u transcends w. This is not possible due to Theorem 2.2. We conclude that every vertex in $K_i \cap K_{i+1}$ either misses all roles in X_1 and no role in X_3, or else misses all roles in X_3 and no role in X_1.

Now suppose that $K_i \cap K_{i+1}$ contains two vertices u, u' such that u misses all roles in X_1 and no role in X_3, whereas u' misses all roles in X_3 and no role in X_1. Then there exists a neighbor v of u with $l_P(v) \leq i - 1$ and role in X_3, and there exists a neighbor v' of u' with $l_P(v') \leq i - 1$ and role in X_1. Note that vu' is not an edge in G, because u' misses all roles in X_3 and $r(v) \in X_3$; similarly, $v'u$ is not an edge. Because roles in X_1 are not adjacent to roles in X_3, we may without loss of generality assume that $l_P(v) < f_P(v')$. Because u and v are neighbors, there is a bag of P that contains both of them. This means that $f_P(u) \leq l_P(v)$, and consequently, $f_P(u) < f_P(v')$. This, together with $l_P(v') < i < l_P(u)$, implies that u and v' are adjacent. This is not possible, since we concluded earlier that $v'u$ is not an edge. Consequently, we have proven Claim 1.

Claim 2. Let $u \in K_i \cap K_{i+1}$. If u misses a role in X_1, then Y contains no role in X_3, and if u misses a role in X_3, then Y contains no role in X_1; otherwise r is not a right choice.

We prove Claim 2 as follows. Suppose that $u \in K_i \cap K_{i+1}$ misses role $x \in X_3$, and Y contains a role in X_1. Then, by definition of Y, there is a vertex $u' \in K_{i+1} \setminus K_i$ with $r(u') \in X_1$. Note that $f_P(u') = i + 1$. Because u misses role x, it needs a neighbor v with $f_P(v) \geq i + 1$ and role x. Because a vertex with role in X_1 is not adjacent to a vertex with role in X_3 and $f_P(u') = i + 1$, we find that $l_P(u') < f_P(v)$. Because u and v are neighbors, we get $f_P(v) \leq l_P(u)$. Then $f_P(u) < i + 1 = f_P(u')$ and $l_P(u') < f_P(v) \leq l_P(u)$. Hence u transcends u'. This is not possible due to Theorem 2.2, and we have proven Claim 2.

We will now do as follows. Because a role in X_1 is not adjacent to a role in X_3, we know that Y will either contain no role from X_1 or no role from X_3. If $X'_1 \neq \emptyset$ then it is immediately clear that Y contains no role in X_3. If $X'_3 \neq \emptyset$ then it is immediately clear that Y contains no role in X_1. Below we show how to decide whether Y contains no role from X_1 or no role from X_3 in the case when $X'_1 = X'_3 = \emptyset$.

If $X'_3 = \emptyset$, then we pick a vertex $u \in K_i \cap K_{i+1}$ and check if u misses a role in X_1. We apply Claim 1 and either find that all vertices in $K_i \cap K_{i+1}$ miss all roles in X_1 and no roles in X_3, or they all miss all roles in X_3 and no role in X_1. We apply Claim 2 in order to find that, in the first case, Y contains no role in X_3, and in the second case, Y contains no role in X_1. From the above we deduce that there are three cases:

(i) $X'_1 \neq \emptyset$, and consequently, $X'_3 = \emptyset$ and Y contains no role from X_3;

(ii) $X'_3 \neq \emptyset$, and consequently, $X'_1 = \emptyset$ and Y contains no role from X_1;

(iii) $X'_1 = X'_3 = \emptyset$ and we found that Y contains no role from X_3;

(iv) $X'_1 = X'_3 = \emptyset$ and we found that Y contains no role from X_1;

We assume without loss of generality that we are in case (ii) or (iv). This means that Y only contains roles from $(X_2 \setminus X'_2) \cup (X_3 \setminus X'_3)$. Before we continue we need two new claims.

Claim 3. If Y contains a role from X_2, then Y contains all roles from $X_3 \setminus X'_3$; otherwise r is not a right choice.

We prove Claim 3 as follows. Let $v \in K_{i+1} \setminus K_i$ have role $r(v) \in X_2$, and assume, for contradiction, that role $x \in X_3 \setminus X'_3$ does not belong to Y. Note that $f_P(v) = i + 1$. Because Y contains no role from X_1, and $X'_1 = \emptyset$, we find that v has no neighbor in K_{i+1} that has its role in X_1. Then, as $f_P(v) = i + 1$ and $r(v) \in X_2$, we find that v will need a neighbor w with role $r(w) \in X_1$ in a later bag, so $f_P(w) \geq i+2$. Because $f_P(v) = i+1$ and $x \notin Y$, we find that v also needs a neighbor w' with role $r(w') = x$ in a later bag, so $f_P(w') \geq i + 2$. Because $r(w) \in X_1$ and $r(w') \in X_3$, we find that w and w' are not adjacent in G. Let u be a vertex in K_{i+1} with $l_P(u) = i + 1$. Because $l_P(u) = i + 1$ and $f_P(w) \geq i + 2$, we find that u and w are not adjacent. For the same reason, u and w' are not adjacent. Then we find that G contains an induced claw with center v and leaves u, w, w', which contradicts the assumption that G is proper interval. Hence we have proven Claim 3.

Claim 4. If $K_{i+1} \setminus K_i$ contains two vertices v and v' such that $r(v) \in X_2$ and $r(v') \in X_3$, then $l_P(v) > l_P(v')$; otherwise r is not a right choice.

We prove Claim 4 as follows. Suppose that v and v' are two vertices in $K_{i+1} \setminus K_i$ such that $r(v) \in X_2$ and $r(v') \in X_3$, and assume, for contradiction, that $l_P(v) \leq l_P(v')$. Because $f_P(v) = f_P(v')$ and $l_P(v) \leq l_P(v')$, all neighbors of v are neighbors of v'. This means that also a neighbor of v with role in X_1 is a neighbor of v'. Because v' has a role in X_3, this is not possible. Hence, we have proven Claim 4.

As we will explain, Claim 3 and Claim 4 enable us to assign roles to the vertices in $K_{i+1} \setminus K_i$. Recall that we already have an ordering of the vertices in $K_i \cap K_{i+1}$ as $u_1, \ldots, u_b$ such that $M_2(u_a) \subseteq M_2(u_{a+1})$ and $M_3(u_a) \subseteq M_3(u_{a+1})$, for $a = 1, \ldots, b-1$. Note that $M_2(u_b) \subseteq X_2 \setminus X_2'$ and $M_3(u_b) \subseteq X_3 \setminus X_3'$.

From our initial ordering of the vertices of G, we immediately obtain an ordering $v_1, \ldots, v_d$ of the vertices in $K_{i+1} \setminus K_i$ such that $l_P(v_i) \leq l_P(v_{i+1})$ for $i = 1, \ldots, d-1$. We consider the vertices of $K_{i+1} \setminus K_i$ in order $v_1, \ldots, v_d$ and try to assign different roles to them by first using the roles in $M_3(u_1)$ (in arbitrary order), then the roles in $M_3(u_2) \setminus M_3(u_1)$ (in arbitrary order), and so on, and finally the remaining roles in $(X_3 \setminus X_3') \setminus M_3(u_b)$ (in arbitrary order). The algorithm must do so, because of the following two reasons. First, we must use the roles in $X_3 \setminus X_3'$ before using any roles from $X_2 \setminus X_2'$ by Claim 3. Second, the above implies together with Claim 4 that we must consider the vertices of $K_{i+1} \setminus K_i$ in order $v_1, \ldots, v_d$. If necessary, i.e., if there are still vertices in $K_{i+1} \setminus K_i$ that have not received roles, we then continue to assign roles from $X_2 \setminus X_2'$ in the same way, i.e, starting with the roles from $M_2(u_1)$ and finishing with the roles from $(X_2 \setminus X_2') \setminus M_2(u_b)$. If this is not possible (i.e., there are too many vertices in $K_{i+1} \setminus K_i$) then we output `NO`. Otherwise we check if r is a starting R-role assignment of $G[K_{\leq i+1}]$. If this is not the case, then we output `NO`, because the restriction of any R-role assignment of G to $K_{\leq i+1}$ is a starting role assignment of $G[K_{\leq i+1}]$. If this is the case, we stop if $i+1 = p$, because a starting R-role assignment of $G[K_{\leq p}]$ is an R-role assignment of G. If $i+1 < p$, we repeat the above procedure with $i = i+1$. This completes the description and the correctness proof of the algorithm for the case when R has two maximal cliques.

For the running time analysis we can use exactly the same arguments as for Case 1. The only difference is that in Case 2 we may have to check if an arbitrary vertex $u \in K_i \cap K_{i+1}$ for some $1 \leq i \leq p-1$ misses a role in X_1 or a role in X_3 in case $X_3' = \emptyset$ or $X_1' = \emptyset$, respectively. This is equivalent to checking whether $M_1(u) = \emptyset$ or $M_3(u) = \emptyset$, respectively. As such it takes constant time, as we maintain these sets as explained in the proof of Lemma 3.4. Because there are at most n bags, this check is performed at most n times. Consequently, the extra running time is $\mathcal{O}(n)$, and we conclude with a total running time of $\mathcal{O}(n+m)$ for Case 2 as well.

Case 3: $q \geq 3$.
In this case R has at least three maximal cliques. First suppose that $p \leq 2q$. By Theorem 3.1, both $G[K_{\leq q}]$ and $G[K_{\geq p-q+1}]$ must be isomorphic to R and have an R-role assignment, in case G has an R-role assignment. Because $p \leq 2q$, every vertex of G is in $K_{\leq q} \cup K_{\geq p-q+1}$. Hence, there are just four possibilities of assigning roles to vertices of G, namely two possibilities for $K_{\leq q}$ combined with two possibilities for $K_{\geq p-q+1}$. We check if one of them leads to an R-role assignment of G. Verifying whether a mapping $V_G \to V_R$ is an R-role assignment of G can be done in $\mathcal{O}(n+m)$ time by following the procedure of computing and updating missing role sets as in Cases 1 and 2.

Now, suppose that $p \geq 2q+1$. The main idea of the algorithm is to map the first q bags of G to the bags of R, use Lemma 3.3 to identify any bags $q+1, ..., q+h$ of G to be "skipped", and then continue to find an isomorphism between R and the subgraph of G induced by the next q

bags of G. In the latter isomorphism, the bags of G will be mapped to the bags of the reversed clique path of R.

We first check if $G[K_{\leq q}]$ is isomorphic to R. This can be done in linear time [21]. If $G[K_{\leq q}]$ is not isomorphic to R, then we output `NO` due to Theorem 3.1. Suppose that $G[K_{\leq q}] \simeq R$ and that without loss of generality we have a starting R-role assignment r of $G[K_{\leq q}]$ with $r(K_j) = L_j$ for $j = 1, \ldots, q$. We now check whether we are in situation (i) or (ii) of Lemma 3.3. This means we have to check which vertices of K_q have roles in X_t and which of these belong to K_{q+1}. Note that we can look up the twin set of the role of each vertex of G in constant time, and decide whether a vertex v belongs to a particular bag K_j in constant time by checking whether $f_P(v) \leq j \leq l_P(v)$. Hence checking which situation of Lemma 3.3 applies can be done in $\mathcal{O}(|K_q|)$ time. Then in both situations we can determine the desired index i as follows.

As in Lemma 3.3, the set T denotes the subset of K_q that consists of all vertices with a role in X_t. We check if T contains a vertex that is not in K_{q+1}, as explained above. Suppose that such a vertex exists. Then, according to Lemma 3.3, there exists an index $i \geq q+1$ such that $K_{\geq q+1} \setminus K_q \subseteq K_{\geq i}$ and the restriction of r to $K_{\geq i}$ is an R-role assignment of $G[K_{\geq i}]$ with $r(K_i) = L_q$. Furthermore, if $i > q+1$ then $r(K_h) \subseteq X_t$ for $h = q+1, \ldots, i-1$. We can find this index i (if it exists) by checking the size of the bags K_h for $h \geq q+1$. We must have a sequence of bags $K_{q+1}, \ldots, K_{q+h^*}$ with $|K_h| \leq |X_t|$ for $h = 1, \ldots, q+h^*$ and $|K_{q+h^*}| = |L_q|$. During the initial computation of the clique path of G and the twin sets of R, the size of each set can be computed without increasing the running time, and stored for allowing constant time look up. Hence the existence of a sequence as described above can be checked in time $\mathcal{O}(h^*)$. Then the desired index is $i = q + h^*$ if $K_{\geq q+1} \setminus K_q \subseteq K_{\geq q+h^*}$ is true as well. This condition is equivalent to $\bigcup_{j=1}^{h^*-1} K_{q+j} \setminus K_q \subseteq K_{q+h^*}$, and hence can be checked in time $\mathcal{O}(\Sigma_{j=1}^{h^*}|K_{q+j}|)$ by going through all the vertices in these bags and checking in constant time whether each of them belongs to K_{q+h^*}. We observe that i is uniquely determined because $|L_q| > |X_t|$. If i does not exist, then we output `NO`.

Now suppose that T does not contain a vertex that is not in K_{q+1}, i.e., all vertices in T are in K_{q+1}. Then, according to Lemma 3.3, there exists an index $i \geq q+1$ such that $T = K_{i-1} \cap K_i$ and $T \cap K_{i+1} = \emptyset$, and the restriction of r to $K_{\geq i}$ is an R-role assignment of $G[K_{\geq i}]$ with $r(K_i) = L_q$. We scan through the bags $K_{q+1}, K_{q+2}, \ldots$, and we stop as soon as we find a bag K_{q+h} for some $h \geq 1$ that does not contain some vertex of T. Then we check if $T \cap K_{q+h} = \emptyset$ and if $T = K_{q+h-2} \cap K_{q+h-1}$. If one of these conditions is not true, then we output `NO`. Otherwise we choose $i = q + h - 1$. The described procedure can clearly be performed in time $\mathcal{O}(\Sigma_{j=1}^{h}|K_{q+j}|)$.

Now suppose that we have found the desired index i as described above. Then we consider $G[K_{\geq i}]$. By Lemma 3.3, we just have to check if $G[K_{\geq i}]$ has an R-role assignment r with $r(K_i) = L_q$. We can do this by using our algorithm; the only difference is that now the roles of the first bag are determined to go to L_q, whereas before we had two options (either L_1 or L_q). This completes the description and the correctness proof of the algorithm for the case when R has at least three maximal cliques.

To complete the running time analysis, note that finding the next index i according to Lemma 3.3 might have to be performed several times. However, each time this is done on a different, non-overlapping part of the clique path of G. Using the running time arguments given above, we then get $\mathcal{O}(n + \Sigma_{j=1}^{p}|K_j|)$ total running time, which is $\mathcal{O}(n+m)$, because the sum of the sizes of all maximal cliques is $\mathcal{O}(n+m)$ (see [24]). □

Below we show how to use the algorithm described in Theorem 3.5 to deal with the case in

which G is a proper interval graph and R is an arbitrary graph that is not necessarily connected. We observe that the running time now also depends on the number of connected components of R.

Corollary 3.6. Role Assignment *can be solved in $\mathcal{O}((n+m)\cdot c_R)$ time on input pairs (G,R) where G is a proper interval graph with n vertices and m edges and R is an arbitrary graph with c_R connected components.*

Proof. If $c_R > 1$, then R is disconnected. In this case we cannot assume that $|V_R| \leq |V_G|$. By the definition of a role assignment, G has an R-role assignment if and only if each connected component of G has an R'-role assignment for some connected component R' of R. Hence we can run our algorithm on G and every connected component of R. This gives a total running time $\mathcal{O}((n+m)\cdot c_R)$. □

The problem Proper Connected Coloring is to test whether a graph G allows a mapping $c : V_G \to \{1,\ldots,\ell\}$ for some $\ell < |V_G|$ such that $c(N_G(u)) = c(N_G(v))$ whenever $c(u) = c(v)$. This problem is equivalent to testing whether a graph has an R-role assignment for some smaller graph R. It is co-NP-complete in general [5]. Theorem 3.5 together with Corollary 3.2 has the following consequence.

Corollary 3.7. *The* Proper Connected Coloring *problem can be solved in polynomial time for proper interval graphs.*

Proof. Let G be a proper interval graph with n vertices and m edges. First assume that G is connected. If $n = 1$, then the answer is `No`. Suppose that $n \geq 2$. Let $P = K_1 \cdots K_p$ be the clique path of G. Note that $p \leq n$ by the definition of a clique path. By Corollary 3.2 we find that G only has an R-role assignment if $R \simeq G[K_{\leq i}]$ for some $1 \leq i \leq p$. This means that we need to apply the $\mathcal{O}(n+m)$ time algorithm for connected proper interval graphs of Theorem 3.5 at most $p \leq n$ times. Since G is connected, $m \geq n-1 \geq 1$ and $\mathcal{O}(n+m) = \mathcal{O}(m)$. Hence we find that testing whether G has an R-role assignment for some graph R with $|V_R| < |V_G|$ takes $\mathcal{O}(nm)$ time.

Now assume that G is disconnected, and let $G_1,\ldots,G_a$ be the connected components of G, $a \geq 2$. We define $n_j = |V_{G_j}|$ and $m_j = |E_{G_j}|$ for $j = 1,\ldots,a$. For increasing values of j from 1 to a, we consider G_j. If $n_j \geq 2$, we check if G_j has an R_j-role assignment for some role graph R_j with $|V_{R_j}| < n_j$. As soon as we find a value of j for which such a graph R_j exists, we replace the connected component G_j by the graph R_j in G, i.e., we output $R = G_1 \oplus \ldots \oplus G_{j-1} \oplus R_j \oplus G_{j+1} \oplus \ldots \oplus G_a$, where $\oplus$ denotes the disjoint union operation on graphs, which results in a graph that has as vertex set and edge set the union of the vertex sets and edge sets of the original graphs, respectively. If for none of the values of j we find a suitable role graph R_j in this way, then we output `NO`. Because we need $\mathcal{O}(n_j m_j)$ time for each G_j, and since $n = n_1 + \ldots + n_a$ and $m = m_1 + \ldots + m_a$, the total running time of this algorithm is $\mathcal{O}(nm)$. □

As a consequence, we have in fact a stronger result: given a proper interval graph G, we can list in polynomial time all graphs R (up to isomorphism) with $|V_R| < n$ such that G has an R-role assignment.

4 Complementary Results and an Open Question

A homomorphism r from a graph G to a graph R is *locally injective* if $|r(N_G(u))| = |N_G(u)|$ for every $u \in V_G$, and r is *locally bijective* if $r(N_G(u)) = N_R(r(u))$ and $|r(N_G(u))| = |N_G(u)|$ for every $u \in V_G$. Locally injective homomorphisms, also called *partial coverings*, have applications in frequency assignment [10] and telecommunication [11]. Locally bijective homomorphisms are also called *coverings* and have applications in topological graph theory [26] and distributed computing [1, 2]. The corresponding decision problems, called Partial Cover and Cover respectively, are NP-complete for arbitrary G even when R is fixed to be the complete graph on four vertices [11, 19].

In this section, to give a complete picture, we study the computational complexity of all three locally constrained homomorphisms on chordal, interval, and proper interval graphs. Our findings can be summarized in the table below, where the three problems have input (G, R) and the left column indicates the graph class that G belongs to. In the table, R is assumed to be an arbitrary graph.

	Partial Cover	Cover	Role Assignment
Chordal	NP-complete	GI-complete	NP-complete
Interval	NP-complete	Polynomial	?
Proper Interval	NP-complete	Polynomial	Polynomial

We start with the following result, which allows us to conclude several of the entries in the above table, and which can be viewed as interesting on its own.

Theorem 4.1. *Let G be a chordal graph and let R be a connected graph. Then there exists a locally bijective homomorphism from G to R if and only if every connected component of G is isomorphic to R.*

Proof. If G is disconnected then we consider each connected component of G separately. Assume that G is connected. If G is isomorphic to R, then the identity mapping from G to R is our desired locally bijective homomorphism.

For the reverse implication, suppose that there exists a locally bijective homomorphism r from G to R. Because any locally bijective homomorphism is also locally surjective, we can apply Theorem 2.6 in order to find that R is chordal. For the same reason we can apply Observation 2.4 in order to find that each vertex in R appears as a role of at least one vertex in G. We claim that each vertex in R appears as a role of exactly one vertex in G. In order to derive a contradiction, suppose that there exists a vertex $x \in V_R$ such that $r^{-1}(x)$ has size at least two.

Let v and v' be two different vertices of G belonging to $r^{-1}(x)$. Let P be a shortest path from v to v' in G. Because P is shortest, P is an induced path. From the definition of a locally bijective homomorphism we deduce the following two statements. Firstly, because two vertices with the same role cannot be adjacent, we find that $|V_P| \neq 2$. Secondly, because a vertex has no two neighbors with the same role, we find that $|V_P| \neq 3$. Hence, P is an induced path with $|V_P| \geq 4$. This, together with $r(v) = r(v') = x$, means that $r(P)$ forms an induced cycle D in R with $|V_D| = |V_P| - 1$. Because R is chordal, D must consist of three vertices, say $D = xyzx$. Consequently, $|V_P| = 4$ holds.

Let C be the connected component of $G[r^{-1}(\{x, y, z\})]$ that contains v and v'. By definition of a locally bijective homomorphism, every vertex is of degree two in D. This means that D

is an induced cycle in G. Because every vertex of P belongs to D, and $|V_P| = 4$, we find that $|V_D| \geq 4$. This contradicts our assumption that G is chordal. We conclude that indeed each vertex in R appears as a role of exactly one vertex in G. This means that r is an isomorphism between G and R, and we find that $G \simeq R$, as desired. □

It is known that Graph Isomorphism is Graph Isomorphism-complete even for pairs (G, R) where G and R are chordal graphs [21]. From Theorem 4.1 we get an immediate polynomial time reduction from Graph Isomorphism on chordal graphs to Cover on chordal graphs, and vice versa. Hence Cover is Graph Isomorphism-complete for pairs (G, R) where G and R are chordal graphs. On the other hand, Cover is polynomial time solvable on interval graphs, and hence also on proper interval graphs, since isomorphism between two interval graphs can be checked in polynomial time [21]. Because every locally bijective homomorphism is locally surjective, we can use Theorem 2.6 to deduce that these three results stay valid for input pairs (G, R) where only G is required to be chordal and R may be an arbitrary graph. This explains the three corresponding entries in the table.

Unfortunately, as indicated in the table, the problem Partial Cover remains NP-complete even on pairs (G, R) where G is a proper interval graph and R is an arbitrary graph. This is because Partial Cover is already NP-complete on pairs (G, R) where G is a complete graph and R is an arbitrary graph. In such cases, G allows a locally injective homomorphism to R if and only if R contains G as a subgraph. Deciding if a graph contains a complete graph as a subgraph is equivalent to the NP-compete problem Clique (see e.g. [15]).

We present one more complexity result on the Role Assignment problem. This result explains a corresponding entry in the table after applying Theorem 2.6.

Theorem 4.2. Role Assignment *is* NP*-complete for input pairs* (G, R) *where* G *and* R *are chordal graphs.*

Proof. We use a reduction from the following problem:

Clique Role Assignment
Instance: a graph G.
Question: does there exist an integer k such that G has a K_k-role assignment?

Lyle showed that Clique Role Assignment is NP-complete even for chordal graphs (Theorem 4.7 in [22]). Let G be a chordal graph. It is well known (cf. [16]) that every chordal graph has a *simplicial* vertex, i.e., a vertex whose neighbors induce a clique. Let u be a simplicial vertex of G, and let p be the number of neighbors of u. We claim that there exists an integer k such that G has a K_k-role assignment if and only if $k = p+1$, i.e., if and only if G has a K_{p+1}-role assignment. As the backward implication is trivial, we only have to show that $k = p+1$ in order to prove the claim. If $k < p+1$, then u misses at least one role in its neighborhood. If $k > p+1$, then u has two neighbors with the same role. Because the neighborhood of u is a clique, these two neighbors are adjacent. This is not possible. This completes the reduction and the proof of Theorem 4.2. □

Just as for Role Assignment, we denote the problems Cover and Partial Cover as R-Cover and R-Partial Cover, respectively, if R is fixed, i.e., not a part of the input. In that case we obtain the following result.

Proposition 4.3. *For any fixed* R*, the problems* R-Role Assignment*,* R-Cover*, and* R-Partial Cover *can be solved in linear time on chordal graphs.*

Proof. We first observe that a homomorphism from G to R maps the vertices in a clique of G to different vertices of R. Hence, in order to get a `YES` answer, each clique in G can have at most $|V_R|$ vertices. We compute the number of vertices in a largest clique of G in linear time. If this number is greater than $|V_R|$, we output `NO`. Otherwise, because the treewidth of a chordal graph is equal to the number of vertices in a largest clique minus 1, we find that G has treewidth bounded by $|V_R|$, which is a constant, as R is fixed. Since all three problems are expressible in monadic second order logic, linear time solvability follows from a well-known result of Courcelle [7]. □

We conclude with the following open question resulting from the table: what is the computational complexity of Role Assignment on input pairs (G, R) when G is an interval graph?

Acknowledgments. We thank an anonymous referee for some useful comments that helped us to improve the readability of our paper.

References

[1] D. Angluin, Local and global properties in networks of processors, *Proceedings of STOC 1980*, ACM (1980) 82–93.

[2] H. L. Bodlaender, The classification of coverings of processor networks, *Journal of Parallel and Distributed Computing* **6** (1989) 166–182.

[3] A. Brandstädt, V.B. Le, and J. Spinrad, Graph Classes: A Survey, SIAM, Philadelphia, 1999.

[4] J. Chalopin, Y. Métivier, and W. Zielonka, Local computations in graphs: the case of cellular edge local computations, *Fundamenta Informaticae* **74** (2006) 85–114.

[5] J. Chalopin and D. Paulusma, Graph labelings derived from models in distributed computing, *Proceedings of WG 2006*, LNCS **4271** (2006) 301–312.

[6] D. G. Corneil, H. Kim, S. Natarajan, S. Olariu, A. P. Sprague, Simple linear time recognition of unit interval graphs, Information Processing Letters 55 (1995) 99–104.

[7] B. Courcelle, The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs, *Information and Computation* **85** (1990) 12–75.

[8] X. Deng, P. Hell, J. Huang, Linear-time representation algorithms for proper circular-arc graphs and proper interval graphs, SIAM Journal on Computing 25 (1996) 390–403.

[9] M. G. Everett and S. Borgatti, Role colouring a graph, *Mathematical Social Sciences* **21** (1991) 183–188.

[10] J. Fiala J. Kratochvíl and T. Kloks, Fixed-parameter complexity of λ-labelings. *Discrete Applied Mathematics* **113** (2001) 59–72.

[11] J. Fiala and J. Kratochvíl, Partial covers of graphs, *Discussiones Mathematicae Graph Theory* **22** (2002) 89–99.

[12] J. Fiala, and D. Paulusma, A complete complexity classification of the role assignment problem, *Theoretical Computer Science* **349** (2005) 67–81.

[13] J. Fiala and D. Paulusma, Comparing universal covers in polynomial time, *Theory of Computing Systems* **46** (2010) 620–635.

[14] D. Fulkerson and O. Gross, Incidence matrices and interval graphs, *Pacific Journal of Mathematics* **15** (1965) 835–855.

[15] M. R. Garey and D. S. Johnson, *Computers and Intractability*, W. H. Freeman and Co., New York, 1979.

[16] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Annals of Discrete Mathematics Vol 57, Elsevier B.V., Amsterdam, 2004.

[17] P. Heggernes, P. van 't Hof, and D. Paulusma, Computing role assignments of proper interval graphs in polynomial time, *Proceedings of the 21st International Workshop on Combinatorial Algorithms* (IWOCA 2010), Lecture Notes in Computer Science 6460, 167–180, 2010.

[18] L. Ibarra, The clique-separator graph for chordal graphs, *Discrete Applied Mathematics* **157** (2009) 1737–1749.

[19] J. Kratochvíl, A. Proskurowski, and J. A. Telle, Covering regular graphs, *Journal of Combinatorial Theory, Series B* **71** (1997) 1–16.

[20] C. Lekkerkerker and D. Boland, Representation of finite graphs by a set of intervals on the real line, *Fundamenta Mathematicae* **51** (1962) 45–64.

[21] G. S. Lueker and K. S. Booth. A linear time algorithm for deciding interval graph isomorphism, *Journal of the ACM* **26** (1979) 183–195.

[22] J. Lyle, Homomorphisms of Graphs, PhD-Thesis, Clemson University, Clemson, South Carolina, 2008.

[23] P. J. Looges. S. Olariu, Optimal greedy algorithms for indifference graphs, Computers Math. Applic. 25 (1993) 15–15.

[24] Y. Okamoto, U Takeaki, and R. Uehara. Counting the number of independent sets in chordal graphs. *Journal of Discrete Algorithms* **6** (2008) 229–242.

[25] A. Pekeč and F. S. Roberts, The role assignment model nearly fits most social networks, *Mathematical Social Sciences* **41** (2001) 275–293.

[26] K. Reidemeister, Einführung in die kombinatorische Topologie. Braunschweig: Friedr. Vieweg. Sohn A.-G. XII, 209 S., 1932.

[27] Y. Rieck and Y. Yamashita, Finite planar emulators for $K_{4,5}-4K_2$ and $K_{1,2,2,2}$ and Fellows' conjecture, *European Journal of Combinatorics* **31** (2010) 903–907.

[28] F. S. Roberts, Indifference Graphs, In: *Proof Techniques in Graph Theory*, Academic Press, New York (1969) 139–146.

[29] L. Sheng, 2-Role assignments on triangulated graphs, *Theoretical Computer Science* **304** (2003) 201–214.