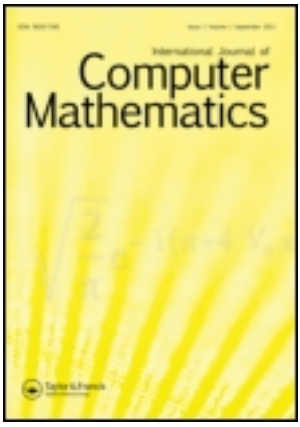

International Journal of Computer Mathematics
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/gcom20

An Optimal Algorithm to Solve 2-Neighbourhood Covering Problem on Interval Graphs

Sukumar Mondal [a], Madhumangal Pal [a] & Tapan K. Pal [a]

[a] Department of Applied Mathematics with Oceanology and Computer Programming, Vidyasagar University, Midnapore, 721 102, India

Available online: 15 Sep 2010

To cite this article: Sukumar Mondal, Madhumangal Pal & Tapan K. Pal (2002): An Optimal Algorithm to Solve 2-Neighbourhood Covering Problem on Interval Graphs, International Journal of Computer Mathematics, 79:2, 189-204

To link to this article: http://dx.doi.org/10.1080/00207160211921

Intern. J. Computer Math., 2002, Vol. 79(2), pp. 189–204

AN OPTIMAL ALGORITHM TO SOLVE 2-NEIGHBOURHOOD COVERING PROBLEM ON INTERVAL GRAPHS

SUKUMAR MONDAL[b], MADHUMANGAL PAL[a,*] and TAPAN K. PAL[a]

[a]*Department of Applied Mathematics with Oceanology and Computer Programming, Vidyasagar University, Midnapore–721 102, India;*
[b]*Department of Mathematics, Y. S. Palpara Mahavidyalaya, Palpara, Midnapore–721 458, India*

(Received 29 November 2000)

Let $G=(V, E)$ be a simple graph and k be a fixed positive integer. A vertex w is said to be a k-neighbourhood-cover of an edge (u, v) if $d(u, w) \leq k$ and $d(v, w) \leq k$. A set $C \subseteq V$ is called a k-neighbourhood-covering set if every edge in E is k-neighbourhood-covered by some vertices of C. This problem is NP-complete for general graphs even it remains NP-complete for chordal graphs. Using dynamic programming technique, an $O(n)$ time algorithm is designed to solve minimum 2-neighbourhood-covering problem on interval graphs. A data structure called interval tree is used to solve this problem.

Keywords: Design of algorithms; Analysis of algorithms; 2-neighbourhood-covering; Interval tree; Interval graph

C. R. Categories: E1, F2, G2.2, I1.2

1. INTRODUCTION

An undirected graph $G=(V, E)$ is an *interval graph* if the vertex set V can be put into one-to-one correspondence with a set I of intervals on the real line such that two vertices are adjacent in G iff their corresponding intervals have non-empty intersection. The set I is called an interval representation of G and G is referred to as the intersection graph of I [4].

*Corresponding author. e-mail: madhumangal@lycos.com

ISSN: 0020-7160 DOI: 10.1080/00207160290004838

Interval graphs arise in the process of modeling real life situations, specially involving time dependencies or other restrictions that are linear in nature. This graph and various subclass thereof arise in diverse areas such as archeology, molecular biology, sociology, genetics, traffic planning, VLSI design, circuit routing, psychology, scheduling, transportation and others. Recently, interval graphs have found applications in protein sequencing [7], macro substitution [3], circuit routine [9], file organization [1], job scheduling [1], routing of two points nets [5] and many others. An extensive discussion of interval graphs also appears in [4]. Thus interval graphs have been studied intensely from both the theoretical and algorithmic point of view.

The k-neighbourhood-covering (k-NC) problem is a variant of the domination problem. Domination is a natural model for location problems in operations research, networking *etc.*

The graphs, considered in this paper are simple *i.e.*, finite, undirected and having no self-loop or parallel edges. In a graph $G=(V, E)$, the *length* of a path is the number of edges in the path. The *distance* $d(x, y)$ from vertex x to vertex y is the minimum length of a path from x to y, and if there is no path from x to y then $d(x, y)=\infty$.

A vertex x k-dominates another vertex y if $d(x, y) \leq k$. A vertex z k-neighbourhood-covers an edge (x, y) if $d(x, z) \leq k$ and $d(y, z) \leq k$ *i.e.*, the vertex z k-dominates both x and y. Conversely, if $d(x, z) \leq k$ and $d(y, z) \leq k$ then the edge (x, y) is said to be k-neighbourhood-covered by the vertex z. A set of vertices $C \subseteq V$ is a k-NC set if every edge in E is k-NC by some vertex in C. The k-NC number $\rho(G, k)$ of G is the minimum cardinality of all k-NC set.

For $k=1$, Lehel *et al.* [8] have presented a linear time algorithm for computing $\rho(G, 1)$ for an interval graph G. Chang *et al.* [2] and Hwang *et al.* [6], have presented linear time algorithms for computing $\rho(G, 1)$ for a strongly chordal graph G provided that strong elimination ordering is known. Hwang *et al.* [6] also proved that k-NC problem is NP-Complete for chordal graphs.

To the best of our knowledge this problem is not solved for $k \geq 2$ for an interval graphs. Using dynamic programming technique, an $O(n)$ time algorithm is designed to solve minimum 2-neighbourhood-covering problem on interval graphs. A data structure called *interval tree* (IT) [10, 11] is used to solve this problem.

2. PRELIMINARIES AND INTERVAL TREE

Let $I=\{I_1, I_2, \ldots, I_n\}$, where $I_j=[a_j, b_j]$, $j=1, 2, \ldots, n$, be the interval representation of an interval graph $G=(V, E)$, $V=\{1, 2, \ldots, n\}$; a_j and b_j are

respectively left and right endpoints of the interval I_j. Without any loss of generality, we assume that each interval contains both of its endpoints and that no two intervals share a common endpoint. If the intervals have common end points then the algorithm **CONVERT** [13] may be used to convert the intervals of I into intervals of distinct end points. We consider intervals in the set I rather than the vertices in G. Further we assume that the graph G is connected, the sorted endpoints list is given and the intervals in I are indexed by increasing right end points *i.e.*, $b_1 < b_2 < \cdots < b_n$. This indexing is known as interval graph (IG) ordering. This ordering is obviously unique when a representation by a set of intervals is provided and fixed.

An interval graph and its interval representation are shown in Figure 1.

For any interval graph $G = (V, E)$ a very important result is stated below.

LEMMA 1 *If the vertices $u, v, w \in V$ are such that $u < v < w$ in the IG ordering and u is adjacent to w, then v is also adjacent to w. But v is not necessarily adjacent to u.*

For each vertex $v \in V$ let $H(v)$ be the highest numbered adjacent vertex of v. If there is no vertex adjacent to v and greater than v then $H(v)$ is assumed to be v. In other words

$$H(v) = \max\{u : (u, v) \in E, u \geq v\}.$$

The array $H(v)$, $v \in V$ satisfies the following important result.

LEMMA 2 *If $u, v \in V$ and $u < v$ then $H(u) \leq H(v)$* [10].

For an interval graph $G = (V, E)$ let an interval tree (IT) with root n be defined as $T(G) = (V, E')$ where $E' = \{(u, v) : u \in V \text{ and } v = H(u), u \neq n\}$.

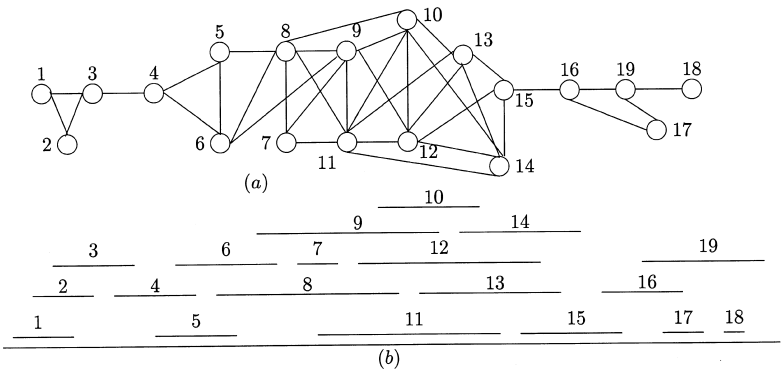

FIGURE 1 An interval graph and its interval representation.

In [10], it is proved that for a connected interval graph there exists a unique interval tree.

The interval tree $T(G)$ of the interval graph of Figure 1 is shown in Figure 2.

Since the tree $T(G)$ is built from the vertex set V and the edge set $E' \subseteq E$, $T(G)$ is a spanning tree of G. Let N_j be the set of vertices which are at a distance j from the vertex n in IT. Thus

$$N_j = \{u : d(u,n) = j\} \text{ and } N_0 \text{ is the singleton set } \{n\}.$$

For each vertex u of IT, we define *level* of u to be the distance of u from the vertex n in the tree IT, *i.e.*, $level(u) = d(u, n)$. If $u \in N_j$ then $d(u, n) = j$ and the vertex u is at level j of IT. Thus the vertices at level j of IT are the vertices of N_j.

The property that the vertices at any level of IT are the consecutive integers, is proved in [10] as the following lemma.

LEMMA 3 *The vertices of N_j are consecutive integers and if v is equal to $min\{u : u \in N_j\}$ then $max\{u : u \in N_{j+1}\}$ is equal to $v-1$* [10].

The following result is also proved in [10].

LEMMA 4 *If level $(u) < level(v)$ then $u > v$.*

If the level of a vertex v of IT is j then it should be adjacent only to the vertices at levels $j-1$, j and $j+1$ in G. This observation is proved in [10] as the following lemma.

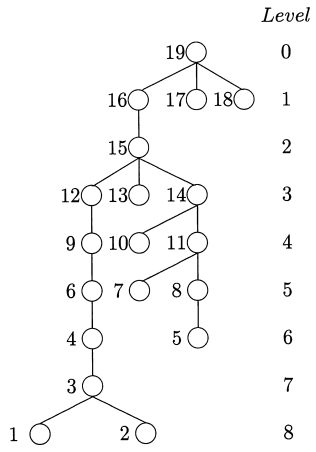

FIGURE 2 Interval tree of the interval graph of Figure 1.

LEMMA 5 *If* $u, v \in V$ *and* $|level\ (v) - level\ (u)| > 1$ *then there is no edge between the vertices* u *and* v.

The distance $d(u, v)$ between any two vertices u and v of same level is either 1 or 2, which is proved in [10] as follows.

LEMMA 6 [10] *For* $u, v \in V$ *if* $level\ (u) = level\ (v)$ *then*

$$d(u, v) = \begin{cases} 1, & (u, v) \in E \\ 2, & otherwise. \end{cases}$$

Let the notation $u \to v$ be used to indicate that there is a path from u to v of length one.

The path in IT from the vertex 1 to the root n is called *main path*. Throughout the paper, we denote the vertex at level l on the main path by u_l^* for all l. From the definition of IT and its level it is obvious that *level*(1) is equal to the height (h) of the tree IT. In [12], it is shown that the length of the main path is h if all the vertices of N_1 are adjacent to u_1^* otherwise it is $h+1$.

3. 2-NEIGHBOURHOOD-COVERING SET

Let C be the minimum 2-neighbourhood-covering (2NC) set of the given interval graph G. To find a 2NC set on interval graphs, an IT is to be constructed.

The main basic idea to compute 2NC is described below. If there exists at least one vertex of N_1 which is not adjacent to u_1^*, we take u_1^* as a member of C otherwise we select the vertex u_2^* as a member of C. Let the first selected vertex (either u_1^* or u_2^*) be at level l. After selection of first member of C, we are to consider two vertices u_{l+3}^* and u_{l+4}^* on the main path at level $l+3$ and $l+4$ respectively. Now, either u_{l+3}^* or u_{l+4}^* (not both) will be a member of C. This selection is to be made according to some results, discussed in the following. After selection of second member of C, we set $l+3$ to l, if u_{l+3}^* is selected, otherwise we set $l+4$ to l. This selection is to be continued till new $l+3$ becomes greater than the height of the tree IT.

The condition to select u_1^* as a first member of C is obtained in the following lemma.

LEMMA 7 *If there exists at least one vertex of* N_1 *which is not connected with* u_1^*, *then* u_1^* *is a possible member of* C.

Proof From the construction of IT it is clear that n is the parent of u_1^*. By hypothesis there exist at least one vertex at level 1, *i.e.*, in N_1 which is not

connected with u_1^*. Let v_1' be any such vertex. Then $d(u_1^*, v_1') = 2$ (as $u_1^* \rightarrow n \rightarrow v_1'$) and $d(u_1^*, n) = 1$, *i.e.*, the vertex u_1^* is a 2NC of the edge (v_1', n). If v_1'' be any vertex of N_1 connected with u_1^* then $d(v_1'', u_1^*) = 1$. As $d(n, u_1^*) = 1$, u_1^* is also a 2NC of the edge (v_1'', n). Hence u_1^* is a 2NC of (v_1, n) for each $v_1 \in N_1$. ■

If u_1^* is connected with all vertices of N_1 then the vertex u_1^* may also be a member of C. But in this case, the vertex u_2^* is to be selected as a member of C. This result is proved in the following lemma.

LEMMA 8 *If u_1^* is connected with all vertices of N_1 then u_2^* is a possible member of C.*

Proof Let u_1^* be connected with all vertices of N_1. Therefore $d(u_1^*, v_1) = 1 = d(u_1^*, n)$ for all $v_1 \in N_1$. Hence the path from u_2^* to any $v_1, v_1 \in N_1$ is $u_2^* \rightarrow u_1^* \rightarrow v_1$ (since u_1^* is adjacent with all vertices of N_1), so $d(u_2^*, v_1) = 2$. But, u_2^* may be adjacent to some vertices of N_1. In this case $d(u_2^*, v_1) = 1$. Hence $d(u_2^*, v_1) \leq 2$, for all $v_1 \in N_1$. Also, $d(u_2^*, n) = 2$. Thus, the edges (n, v_1), $v_1 \in N_1$ are 2NC by u_2^*.

Again, if $v_2 \in N_2$ then $d(u_2^*, v_2) \leq 2$ (by Lemma 6). Therefore, $d(u_2^*, v_1) \leq 2$ and $d(u_2^*, v_2) \leq 2$ for $v_1 \in N_1$ and $v_2 \in N_2$. Thus each edge $(v_1, v_2) \in E$ is 2NC by u_2^*. Hence u_2^* may be selected as a member of C. ■

From Lemma 8, it is observed that either u_1^* or u_2^* may be selected as a member of C. But our aim is to find C with minimum cardinality. So, under the condition of Lemma 8 u_2^* is to be selected instead of u_1^*.

If u_l^* be selected as a member of C at any stage then in the next stage either u_{l+3}^* or u_{l+4}^* is to be selected as a member of C. The selection depends on some results which are considered below.

Here we introduce some notations which are used throughout the remaining part of the paper.

parent if $H(u) = v$ then the *parent*$(u) = v$ in IT,

gparent if *parent*(*parent*$(u)) = v$ then *gparent*$(u) = v$,

l an integer representing the level number at any stage,

u_l^* represents the vertex on the main path at level l,

X_l the set of vertices at level l of IT which are greater than u_l^*, *i.e.*,

$$X_l = \{v : v > u_l^* \text{ and } v \in N_l\}.$$

Y_l the set of vertices at level l of IT which are less than u_l^*, *i.e.*,

$$Y_l = \{v : v < u_l^* \text{ and } v \in N_l\}.$$

w_l the least vertex of the set Y_l, *i.e.*, $w_l = min\{v : v \in Y_l\}$.

It may be noted that $X_l \cap Y_l = \phi$ and $N_l = X_l \cup Y_l \cup \{u_l^*\}$. As the vertices of N_l are consecutive integers, the vertices of X_l and Y_l are also consecutive integers.

LEMMA 9 *If v be any member of $\cup_{i=0}^{2} X_{l+i}$ then $d(v, u_l^*) \leq 2$.*

Proof To prove this lemma, consider the IT of Figure 3. From definition of X_l it follows that $u_l^* < v$ for all $v \in X_l$, and for all l.

Let v_1 be any vertex of X_{l+2}. Then $u_{l+2}^* < v_1 < u_{l+1}^*$. Since $(u_{l+2}^*, u_{l+1}^*) \in E$, by Lemma 1 $(v_1, u_{l+1}^*) \in E$. Therefore, $d(v_1, u_l^*) = 2$ (as $u_l^* \to u_{l+1}^* \to v_1$). If v_1'' be any vertex of X_{l+1} then $u_{l+1}^* < v_1'' < u_l^*$. Since $(u_{l+1}^*, u_l^*) \in E$, $(v_1'', u_l^*) \in E$ (by Lemma 1) and hence $d(u_l^*, v_1'') = 1$.

Again, if $v \in X_l$ then $d(v, u_l^*) \leq 2$ (by Lemma 6). Thus $d(u_l^*, v) \leq 2$ for all $v \in \cup_{i=0}^{2} X_{l+i}$. ■

LEMMA 10 *If t be any member of $\cup_{i=0}^{2} Y_{l+i}$ then either $d(t, u_l^*) \leq 2$ or $d(u_{l+3}^*, t) \leq 2$.*

Proof To prove this lemma consider the IT of Figure 3. Let t_1 and t_1' be any two vertices of Y_{l+2} and Y_{l+1} respectively. Let u_l' be any vertex at level l and $u_l' < u_l^*$. There are two cases arise. Case 1: $u_l' = u_l^*$ and Case 2: $u_l^* \neq u_l^*$.

Case 1 $u_l' = u_l^*$. In this case $d(u_l^*, t_l') = 1$ and $d(u_l^*, t_1) = 2$. Also, $d(u_l^*, t) \leq 2$ (by Lemma 6) for all $t \in Y_l$. Therefore $d(u_l^*, t) \leq 2$ for all $t \in \cup_{i=0}^{2} Y_{l+i}$.

Case 2 $u_l' \neq u_l^*$. Without loss of generality we assume that *parent* $(t_1) = t_1'$ and *parent* $(t_1') = u_l'$. Since *parent* $(t_1) = t_1'$ *i.e.*, $H(t_1) = t_1' < u_{l+1}^*$, $(t_1, u_{l+1}^*) \notin E$. Similarly, $H(t_1') = u_l' < u_l^*$ implies $(t_1', u_l^*) \notin E$. Thus, $d(u_l^*, t_1') = 2$ and $d(u_l^*, t_1) = 3$ (as $u_l^* \to u_l' \to t_1'$ or $u_l^* \to u_{l+1}^* \to t_1' \to t_1$).

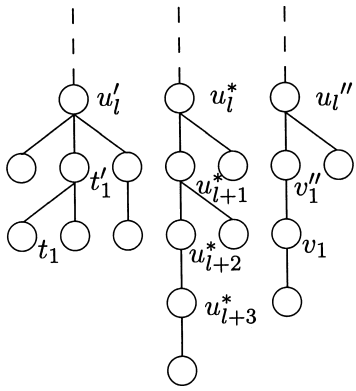

FIGURE 3 A part of an IT.

Now, $u^*_{l+3} < t_1 < u^*_{l+2} < t'_l$, $(u^*_{l+3}, u^*_{l+2}) \in E$ and $(t_1, t'_1) \in E$ implies $(t_1, u^*_{l+2}) \in E$ and $(u^*_{l+2}, t'_1) \in E$. Thus, $d(u^*_{l+3}, t_1) \leq 2$ and $d(u^*_{l+3}, t'_1) = 2$. Hence, either $d(u^*_l, t) \leq 2$ or $d(u^*_{l+3}, t) \leq 2$ for all $t \in \cup_{i=0}^{2} Y_{l+i}$. ■

We have $d(v, u^*_{l+3}) \leq 2$ for all $v \in N_{l+3}$ (by Lemma 6). Combining the results of Lemmas 9 and 10 and their proofs one can conclude the following result.

LEMMA 11 *All edges* $(x, y) \in E$ *where* $x, y \in \cup_{i=0}^{3} N_{l+i}$ *are 2NC by either* u^*_l *or* u^*_{l+3} *or both.*

From above lemma, it follows that if u^*_l is selected as a member of C at any stage then one can select u^*_{l+1} or u^*_{l+2} or u^*_{l+3} as a next member of C.

Also from the Lemmas 9 and 10 one may conclude another result which is stated below.

COROLLARY 1 *If* $gparent(w_{l+2}) = u^*_l$ *then the edge* (x, y) *where* $x, y \in \cup_{i=0}^{2} N_{l+i}$ *is 2NC by* u^*_l.

LEMMA 12 *If* $gparent(w_{l+2}) \neq u^*_l$ *then* u^*_{l+4} *can not be a member of* C.

Proof The condition $gparent(w_{l+2}) \neq u^*_l$ implies that the IT has a branch on the left of the main path. So we consider the IT of Figure 4. It may be noted that existence of w_{l+2} implies $Y_{l+2} \neq \phi$.

In this case, $gparent(w_{l+2}) < u^*_l$, *i.e.*, $parent(parent(w_{l+2})) = H(parent(w_{l+2})) < u^*_l$. So, $(parent(w_{l+2}), u^*_l) \notin E$. Since, $gparent(u^*_{l+3}) < gparent(w_{l+2}) < u^*_l$ and $(gparent(u^*_{l+3}), u^*_l) \in E$ then by Lemma 1,

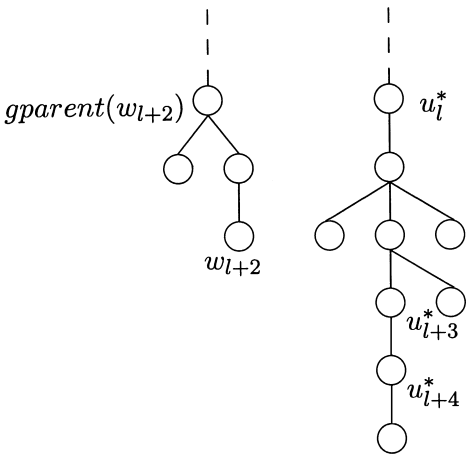

FIGURE 4 A part of an IT.

$(gparent(w_{l+2}), u_l^*) \in E$. Therefore, $d(parent(w_{l+2}), u_l^*) = 2$ and $d(w_{l+2}, u_l^*) = 3$. Thus, the edge $(w_{l+2}, parent(w_{l+2}))$ is not 2NC by u_l^*. Since, $d(w_{l+2}, u_{l+3}^*) \leq 2$ as $u_{l+3}^* < w_{l+2} < parent(u_{l+2}^*)$. Therefore, $d(u_{l+4}^*, parent(w_{l+2})) \geq 3$. Again, the edge $(w_{l+2}, parent(w_{l+2}))$ is not 2NC by u_{l+4}^*. Hence, u_{l+4}^* can not be a member of C. ■

But, if $gparent(w_{l+2})$ is equal to u_l^* then some time one can select the vertex u_{l+4}^* as a member of C. The selection depends on the nature of the IT of the given interval graph.

LEMMA 13 *If* $gparent(w_{l+2}) = u_l^*$ *and* $X_{l+3} = \phi$ *then* u_{l+4}^* *be a possible member of* C.

Proof To prove this lemma, consider the IT of Figure 5. The relation $gparent(w_{l+2}) = u_l^*$ implies that $d(u_l^*, v) \leq 2$ for all $v \in \cup_{i=0}^{2} N_{l+i}$ (by Corollary 1). So the edge (x, y), $x \in N_{l+2} \cup N_{l+1}$ and $y \in N_{l+2} \cup N_{l+1}$ is 2NC by u_l^*.

As $X_{l+3} = \phi$, $v \leq u_{l+3}^*$, for all $v \in N_{l+3}$, *i.e.*, $u_{l+4}^* < v < u_{l+3}^*$, for all $v \in N_{l+3}$. Again $(u_{l+3}^*, u_{l+4}^*) \in E$, so by Lemma 1, $(v, u_{l+3}^*) \in E$. Thus, $d(v, u_{l+4}^*) \leq 2$ for all $v \in N_{l+3}$. Also, $d(v, u_{l+4}^*) \leq 2$ for all $v \in N_{l+4}$. So the edge (x, y), $x \in N_{l+3} \cup N_{l+4}$ and $y \in N_{l+3} \cup N_{l+4}$ is 2NC by u_{l+4}^*. Hence the vertex u_{l+4}^* may be selected as a member of C. ■

From the above lemma it follows that if $X_{l+3} = \phi$ then one can select u_{l+4}^* as a member of C. But, if $X_{l+3} \neq \phi$ then some times one can select u_{l+4}^* as a member of C. The conditions for selecting u_{l+4}^* as a next possible member of C are discussed below.

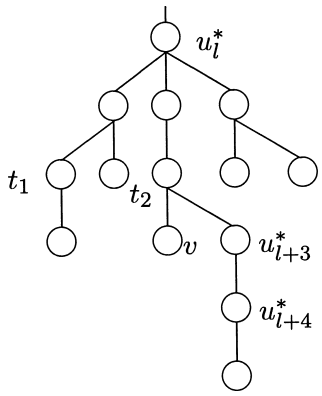

FIGURE 5 A part of an IT.

LEMMA 14 *If* $gparent(w_{l+2}) = u_l^*$ *and if* $(u_{l+3}^*, v) \notin E$ *for at least one* $v \in X_{l+3}$ *where* $X_{l+3} \neq \phi$ *then* u_{l+4}^* *can not be a member of* C.

Proof To prove this lemma, consider the IT of Figure 6. The relation $gparent(w_{l+2}) = u_l^*$ implies that $d(u_l^*, v) \leq 2$ for all $v \in \cup_{i=0}^{2} N_{l+i}$ (by Corollary 1). So all edges (x, y), $x \in N_{l+2} \cup N_{l+1}$ and $y \in N_{l+2} \cup N_{l+1}$ are 2NC by u_l^*. But the edge (x, y), $x \in N_{l+2}$ and $y \in N_{l+3}$ is not 2NC by u_l^* as $d(u_l^*, y) \not\leq 2$. Now, if $(u_{l+3}^*, v) \notin E$ for at least one $v \in X_{l+3}$ then the shortest path from u_{l+4}^* to v is $u_{l+4}^* \rightarrow u_{l+3}^* \rightarrow parent(v) \rightarrow v$ (since $parent(u_{l+4}^*) = u_{l+3}^* = H(u_{l+4}^*)$ and $v > u_{l+3}^*$, $v \in X_{l+3}$ so $(u_{l+4}^*, v) \notin E$) and hence $d(u_{l+4}^*, v) = 3$. Thus, the edge $(v, parent(v))$, $v \in X_{l+3}$ is not 2NC by u_{l+4}^*. Therefore, u_{l+4}^* can not be a member of C. ■

LEMMA 15 *If* $gparent(w_{l+2}) = u_l^*$ *and* $(u_{l+3}^*, v) \in E$ *for all* $v \in X_{l+3}$ *but* $parent(v) \neq parent(u_{l+3}^*)$ *for at least one* $v \in X_{l+3}$ *then* u_{l+4}^* *can not be a member of* C.

Proof Let v be a vertex of X_{l+3} such that $parent(v) \neq parent(u_{l+3}^*)$. In this case, the edge $(v, parent\ (v))$ is not 2NC by u_l^*, as $d(u_l^*, parent(v)) = 2$ but $d(u_l^*, v) = 3$ (see Fig. 6).

Now, if $(u_{l+3}^*, v) \in E$ then $d(u_{l+4}^*, v) = 2$ but $d(u_{l+4}^*, parent(v)) = 3$ (because, $(u_{l+3}^*, parent(v)) \notin E$ as $H(u_{l+3}^*) = parent(u_{l+3}^*) < parent(v)$, so the shortest path from u_{l+4}^* to $parent(v)$ is $u_{l+4}^* \rightarrow u_{l+3}^* \rightarrow v \rightarrow parent(v)$). Therefore the edge $(v, parent(v))$ is not 2NC by u_{l+4}^*. Hence u_{l+4}^* can not be a member of C. ■

LEMMA 16 *If* $gparent(w_{l+2}) = u_l^*$ *and* $(u_{l+3}^*, u) \in E$ *for all* $u \in X_{l+3} \cup Y_{l+2}$, $(v, t) \in E$ *for at least one* $v \in X_{l+3}$ *and* $t \in Y_{l+2}$ *and* $parent(v) = parent(u_{l+3}^*)$ *for all* $v \in X_{l+3}$ *then* u_{l+4}^* *is a possible member of* C.

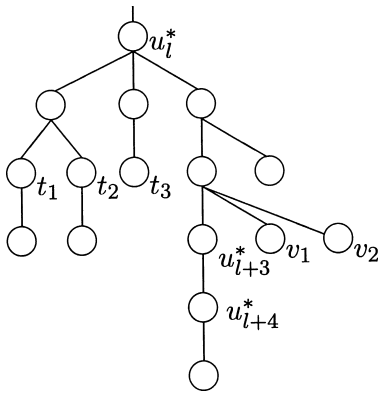

FIGURE 6 A part of an IT.

Proof To prove this lemma consider the IT of Figure 6. Since $(u^*_{l+3}, u) \in E$ for all $u \in X_{l+3} \cup Y_{l+2}$ the edge (x, y), $x \in N_{l+2} \cup N_{l+3}$ and $y \in N_{l+2} \cup N_{l+3}$ is 2NC by u^*_{l+4} (as $u^*_{l+4} \to u^*_{l+3} \to x$ and $u^*_{l+4} \to u^*_{l+3} \to y$). Also, the edge $(parent(u^*_{l+3}), v)$, $v \in X_{l+3}$ is 2NC by u^*_{l+4} (as $d(parent(u^*_{l+3}), u^*_{l+4}) = 2$, $d(u^*_{l+4}, v) = 2$). Again, the edge (t, t'), $t \in Y_{l+2}$, $t' \in Y_{l+3}$ is 2NC by u^*_{l+4} (as $d(u^*_{l+4}, t) = 2$ and $d(u^*_{l+4}, t') \leq 2$) and by the result of Corollary 1, the lemma follows. ■

The converse case of the above result is proved in the following lemma.

LEMMA 17 *If* $gparent(w_{l+2}) = u^*_l$ *and* $(u^*_{l+3}, v) \in E$ and $parent(v) = parent(u^*_{l+3})$ *for all* $v \in X_{l+3}$, $(v, t) \in E$ *for all* $v \in X_{l+3}$ *and* $t \in Y_{l+2}$ *and* $(u^*_{l+3}, t) \notin E$ *for at least one* $t \in Y_{l+2}$ *then* u^*_{l+4} *can not be a member of* C.

Proof Since $(u^*_{l+3}, v) \in E$, $v \in X_{l+3}$, there is a path $u^*_{l+4} \to u^*_{l+3} \to v$ from u^*_{l+4} to v and hence $d(u^*_{l+4}, v) = 2$. But, the shortest path from u^*_{l+4} to t is $u^*_{l+4} \to u^*_{l+3} \to parent(u^*_{l+3}) \to t$ (since $(u^*_{l+3}, t) \notin E$, but $(parent(u^*_{l+3}), t) \in E$). So, $d(u^*_{l+4}, t) = 3$. Therefore the edge (v, t), $v \in X_{l+3}$, $t \in Y_{l+2}$ is not 2NC by u^*_{l+4}. Also, the edge (t, v) is not 2NC by u^*_l. Hence u^*_{l+4} can not be a member of C. ■

LEMMA 18 *If* $gparent(w_{l+2}) = u^*_l$ *for all* $v \in X_{l+3}$, $(u^*_{l+3}, v) \in E$ *and* $parent(v) = parent(u^*_{l+3})$ *and* $(v, t) \notin E$ *for all* $v \in X_{l+3}$, $t \in Y_{l+2}$ *then* u^*_{l+4} *is a possible member of* C.

Proof Consider Figure 6 to proof this lemma. Since, $(u^*_{l+3}, v) \in E$ for all $v \in X_{l+3}$, the edge (v_1, v_2), $v_1, v_2 \in X_{l+3}$ is 2NC by u^*_{l+4} (as $d(v, u^*_{l+4}) \leq 2$). Let $u \in Y_{l+3}$. Since $u < u^*_{l+3}$ and $(u^*_{l+4}, u^*_{l+3}) \in E$, therefore, $(u, u^*_{l+3}) \in E$. Also, $u < u^*_{l+3} < t$, $t \in Y_{l+2}$ and if $(u, t) \in E$ then $(u^*_{l+3}, t) \in E$ and for this t, $d(u^*_{l+4}, t) = 2$. Hence (u, t) is 2NC by u^*_{l+4} and by Corollary 1 u^*_{l+4} may be a member of C. ■

LEMMA 19 *If* $X_{l+3} = \phi$ *and* $Y_{l+2} = \phi$ *then* u^*_{l+4} *is a possible member of* C.

Proof For this case, a possible IT is shown in Figure 7. Let $t \in Y_{l+3}$ and $t' \in Y_{l+4}$. As $u^*_{l+4} < t < u^*_{l+3}$ and $(u^*_{l+4}, u^*_{l+3}) \in E$ then $(t, u^*_{l+3}) \in E$ and hence $d(t, u^*_{l+4}) \leq 2$.

Also, $d(t', u^*_{l+4}) \leq 2$ (by Lemma 6). Thus the edge (t, t'), if any, is 2NC by u^*_{l+4}. And by Corollary 1, the lemma follows. ■

LEMMA 20 *If* $Y_{l+2} = \phi$ *and* $(u^*_{l+3}, v) \notin E$ *for at least one* $v \in X_{l+3}$ *then* u^*_{l+4} *can not be a member of* C.

Proof To prove this lemma consider the IT of Figure 8. If $(u^*_{l+3}, v) \notin E$ for at least one $v \in X_{l+3}$ then the shortest path from u^*_{l+4} to v is

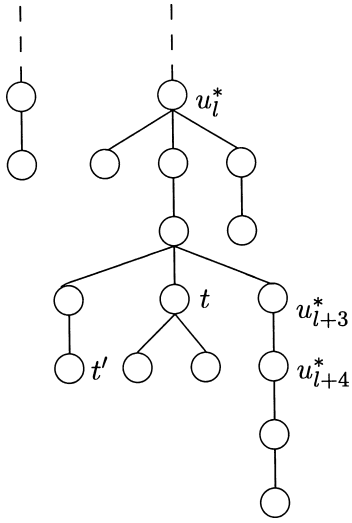

FIGURE 7 A part of an IT.

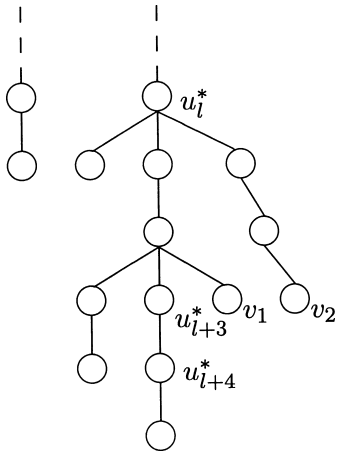

FIGURE 8 A part of an IT.

$u_{l+4}^* \rightarrow u_{l+3}^* \rightarrow parent(u_{l+3}^*) \rightarrow v$. Therefore, $d(u_{l+4}^*, v) = 3$. Hence the edge (u, v), $u \in X_{l+2}$, $v \in X_{l+3}$ is not 2NC by u_{l+4}^*. Thus u_{l+4}^* can not be a member of C. ■

LEMMA 21 *If* $Y_{l+2} = \phi$, $(u_{l+3}^*, v) \in E$ *for all* $v \in X_{l+3}$ *and* $parent(v) \neq parent(u_{l+3}^*)$ *for at least one* $v \in X_{l+3}$ *then* u_{l+4}^* *can not be a member of* C.

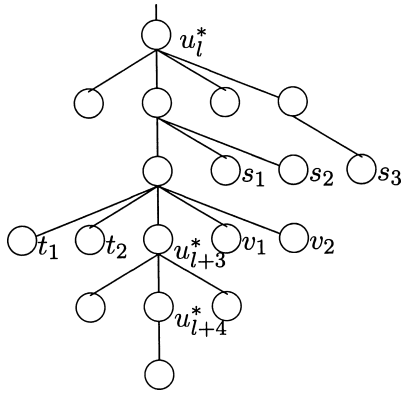

FIGURE 9 A part of an IT.

Proof Without loss of generality, we assume that $(u_{l+3}^*, v_2) \in E$ and $parent(v_2) \neq parent(u_{l+3}^*)$, $v_2 \in X_{l+3}$ (see Fig. 8). Since $parent(v_2) \neq parent(u_{l+3}^*)$, $(u_{l+3}^*, parent(v_2)) \notin E$ as $parent(u_{l+3}^*) = H(u_{l+3}^*) < parent(v_2)$. In this case the path from u_{l+4}^* to $parent(v_2)$ is $u_{l+4}^* \to u_{l+3}^* \to v_2 \to parent(v_2)$.

Therefore, $d(u_{l+4}^*, parent(v_2)) = 3$ and $d(u_{l+4}^*, v_2) = 2$. Hence the edge $(v_2, parent(v_2))$ is not 2NC by u_{l+4}^*. Thus, u_{l+4}^* can not be a member of C. ■

LEMMA 22 *If $Y_{l+2} = \phi$ and $(u_{l+3}^*, v) \in E$ for all $v \in X_{l+3}$ and $parent(v) = parent(u_{l+3}^*)$ for all $v \in X_{l+3}$ then u_{l+4}^* may be a possible member of C.*

Proof For this case the possible form of IT is shown in Figure 9. Since, $(u_{l+3}^*, v) \in E$ for all $v \in X_{l+3}$, $d(u_{l+4}^*, v) = 2$ (as $u_{l+4}^* \to u_{l+3}^* \to v$). Also, $d(u_{l+4}^*, t) \leq 2$ for all $t \in Y_{l+3}$. Again, $Y_{l+2} = \phi$ and $parent(v) = parent(u_{l+3}^*)$ for all $v \in X_{l+3}$, so the edge $(parent(u_{l+3}^*), u)$, $u \in N_{l+3}$ is 2NC by u_{l+4}^*.

Again, the edge (v', v), $v' \in X_{l+4}$, $v \in X_{l+3}$ also 2NC by u_{l+4}^* (since $d(u_{l+4}^*, v') \leq 2$ and $d(u_{l+4}^*, v) = 2$). Hence u_{l+4}^* may be a possible member of C. ■

4. ALGORITHM AND ITS COMPLEXITY

From the above lemmas it is observed that if u_l^* is selected as a member of C at any stage then either u_{l+3}^* or u_{l+4}^* will be selected as a member of C at next stage. Also, we observed that the vertex u_{l+3}^* may be selected at any stage. But our aim is to find the set C such that $|C|$ is minimum. To find C with minimum cardinality we will select u_{l+4}^* if

possible. All possible cases for selection of the members of C are already presented in terms of lemmas. Now a procedure FINDNEXT is formally presented in the following which computes the level L of next vertex u_L^* of C, if the level l of the currently selected vertex u_L^* is supplied.

Procedure FINDNEXT (l, L)

//The procedure computes the level L such that u_L^* will be the next member of C where as u_l^* is the currently selected vertex of C. The sets X_i, Y_i and the array u_i^*, $i=1,2,\ldots,h$, h is the height of the tree $T(G)$, are known globally.//

Initially $L=l+3$;
If $Y_{l+2}=\phi$ then
  if $X_{l+3}=\phi$ then $L=l+4$; (Lemma 19)
  elseif for all $v\in X_{l+3}$, $parent(v)=parent(u_{l+3}^*)$ and $(u_{l+3}^*, v)\in E$ then
    $L=l+4$; (Lemma 22)
  endif;
else //$Y_{l+2}\neq\phi$//
  if $gparent(w_{l+2})=u_l^*$ then
    if $X_{l+3}=\phi$ then $L=l+4$; (Lemma 13)
    elseif for all $v\in X_{l+3}$, $parent(v)=parent(u_{l+3}^*)$, $(u_{l+3}^*, v)\in E$ and
      if $(v, t)\in$ E for some $v\in X_{l+3}$, $t\in Y_{l+2}$ and
        $(u_{l+3}^*, t)\in E$ then $L=l+4$; (Lemma 16)
      elseif $(v, t)\notin$ E for all $v\in X_{l+3}$ and $t\in Y_{l+2}$
        then $L=l+4$; (Lemma 18)
      endif;
    endif;
  endif;
endif;
return L;
end FINDNEXT

Now, we are in the stage to present the complete algorithm to find a minimum size 2NC set on interval graphs. Using repeatedly, the procedure FINDNEXT, one can compute the 2NC set.

The complete algorithm to compute 2NC set is presented below.

ALGORITHM **TWONC**

Input: An interval graph with its interval representation.
Output: Minimum cardinality 2-neighbourhood-covering set C.
Initially $C=\phi$;

Step 1 Construct the interval tree $T(G)$.
Step 2 Compute the vertices on the main path of the tree $T(G)$ and let them u_i^*, $i=1,2,\ldots,h$, h is the height of the tree $T(G)$.
Step 3 Compute the sets X_i, Y_i, $i=1,2,\ldots,h$.
Step 4 If $(u_1^*, v)\in E$ for all $v\in X_1\cup Y_1$ then
$l=1$ else $l=2$;
endif;
$C=C\cup\{u_l^*\}$
Step 5 Repeat
Call FINDNEXT (l, L); //Find level L for the next vertex of C//
$l=L$;
$C=C\cup\{u_l^*\}$;
Until ($|h-l|\leq 2$);

end TWONC.

The vertices of $T(G)$ are the vertices of G. The sets N_i, $i=1,2,\ldots,h$ are mutually exclusive and the vertices of each N_i are consecutive integers. Again, the sets X_i and Y_i, $i=1,2,\ldots,h$ are also mutually exclusive, *i.e.*, $X_i\cap X_j=\phi$, $Y_i\cap Y_j=\phi$, for $i\neq j$ and $i,j=1,2,\ldots,h$ and $X_i\cap Y_j=\phi$, i, $j=1,2,\ldots,h$. Moreover, $N_i=X_i\cup Y_i\cup\{u_i^*\}$, $i=1,2,\ldots,h$. The vertices of each X_i and Y_i are also consecutive integers. So only the lowest and highest numbered vertices are sufficient to maintain the sets X_i, Y_i, N_i, $i=1,2,\ldots,h$. So, we will store only the lowest and highest numbered vertices corresponding to the sets X_i, Y_i, N_i instead of all vertices. If any set is empty then the lowest and highest numbered vertices may be taken as 0 and 0. It is obvious that $|\cup_{i=1}^{n} N_i|=n$. In the procedure FINDNEXT, only the vertices of the sets N_l, N_{l+1}, N_{l+2} and N_{l+3} are considered to process them. The total number of vertices of these sets is $|\cup_{i=0}^{3} N_{l+i}|$ and the subgraph induced by the vertices $\cup_{i=0}^{3} N_{l+i}$ is a part of the tree $T(G)$ so the total number of edges in this portion is less than or equal to $|\cup_{i=0}^{3} N_{l+i}|-1$. Hence one can conclude the following result.

THEOREM 1 *The time complexity of the procedure* FINDNEXT(l, L) *is* $O(|\cup_{i=0}^{3} N_{l+i}|)$.

Now we can compute the time complexity of the algorithm **TWONC**.

THEOREM 2 *The 2-neighbourhood covering set of an interval graph can be computed in* $O(n)$ *time.*

Proof For a given interval representation of an interval graph, the unique interval tree $T(G)$ can be constructed in $O(n)$ time [10, 11]. Since the main

path starting from the vertex 1 and ending at the vertex n, all the vertices u_i^*, $i=1,2,\ldots,h$ on the main path can be identified in $O(n)$ time. By computing the level of each vertex one can compute the sets X_i and Y_i, $i=1,2,\ldots,h$ in $O(n)$ time. Step 3 of Algorithm **TWONC** can be computed in $O(n)$ time. Each iteration of repeat-until loop takes only $O(|\cup_{i=0}^{3} N_{l+i}|)$ time for a given l. The algorithm **TWONC** calls the procedure FINDNEXT for $|C|$ times and each time the value of l is increased by either 3 or 4. Also, if the vertices of the set $\cup_{i=0}^{3} N_{l+i}$ (or $\cup_{i=0}^{3} N_{l'+i}$) are considered to find the kth (or k'th) member of C then $\cup_{i=0}^{3} N_{l+i}$ and $\cup_{i=0}^{3} N_{l'+i}$ are disjoint. Therefore, Step 5 takes $O(|\cup_{i=0}^{h} N_i|) = O(n)$ time. Hence overall time complexity is of $O(n)$. ■

The following theorem gives the space complexity of Algorithm **TWONC**.

THEOREM 3 *The space complexity to of algorithm* ***TWONC*** *is* $O(n)$.

Proof By storing the endpoints of intervals one can store an interval graph using only $O(n)$ space. The tree $T(G)$, the sets X_i, Y_i and the vertices u_i^*, $i=1,2,\ldots,h$ can be stored using $O(n)$ space. Also, $|C|$ is equal to $O(n)$. Hence the total space complexity is of $O(n)$. ■

References

[1] Carlisle, M. C. and Loyd, E. L. (1991). On the k-coloring of intervals, *LNCS*, *497*, *ICCI'91*, pp. 90–101.

[2] Chang, G. J., Farber, M. and Tuza, Z. (1993). Algorithmic aspects of neighbourhood numbers, *SIAM J. Discrete Math.*, **6**, 24–29.

[3] Fabri, J., *Automatic Storage Optimization*, (UMI Press Ann Arbor, MI, 1982).

[4] Golumbic, M. C., *Algorithmic graph theory and perfect graphs* (Academic Press, New York, 1980).

[5] Hashimoto, A. and Stevens, J. (1971). Wire routing by optimizing channel assignment within large apertures, *Proc., 8th IEEE Design Automation Workshop*, pp. 155–169.

[6] Hwang, S. F. and Chang, G. J. (1998). k-neighbourhood-covering and independence problems for chordal graphs, *SIAM J. Discrete Math.*, **11**(4), 633–643.

[7] Jungck, J. R., Dick, O. and Dick, A. G. (1982). Computer assisted sequencing, interval graphs and molecular evolution, *Biosystem*, **15**, 259–273.

[8] Lehel, J. and Tuza, Z. (1986). Neighbourhood perfect graphs, *Discrete Math.*, **61**, 93–101.

[9] Ohtsuki, T., Mori, H., Khu, E. S., Kashiwabara, T. and Fujisawa, T. (1979). One dimensional logic gate assignment and interval graph, *IEEE Trans. Circuits and Systems*, **26**, 675–684.

[10] Pal, M. and Bhattacharjee, G. P. (1997). A data structure on interval graphs and its applications, *J. Circuits, Systems, and Computers*, **7**(3), 165–175.

[11] Pal, M. and Bhattacharjee, G. P. (1997). An optimal parallel algorithm for all-pairs shortest paths on unweighted interval graphs, *Nordic J. Computing*, **4**, 342–356.

[12] Pal, M. and Bhattacharjee, G. P. (1995). Optimal sequential and parallel algorithms for computing the diameter and centre of an interval graph, *Intern. J. Computer Math.*, **59**, 1–13.

[13] Pal, M. and Bhattacharjee, G. P. (1995). The parallel algorithms for determining edge-packing and efficient edge dominating sets in interval graphs, *Parallel Algorithms and Applications*, **7**, 193–207.