SIAM J. DISCRETE MATH.
Vol. 10, No. 1, pp. 109–127, February 1997
© 1997 Society for Industrial and Applied Mathematics
009

CLIQUE r-DOMINATION AND CLIQUE r-PACKING PROBLEMS ON DUALLY CHORDAL GRAPHS*

ANDREAS BRANDSTÄDT†, VICTOR D. CHEPOI‡, AND FEODOR F. DRAGAN‡

Abstract. Let $\mathcal{C}$ be a family of cliques of a graph $G = (V, E)$. Suppose that each clique C of $\mathcal{C}$ is associated with an integer $r(C)$, where $r(C) \geq 0$. A vertex v *r-dominates* a clique C of G if $d(v, x) \leq r(C)$ for all $x \in C$, where $d(v, x)$ is the standard graph distance. A subset $D \subseteq V$ is a *clique r-dominating set* of G if for every clique $C \in \mathcal{C}$ there is a vertex $u \in D$ which r-dominates C. A *clique r-packing set* is a subset $P \subseteq \mathcal{C}$ such that there are no two distinct cliques $C', C'' \in P$ r-dominated by a common vertex of G. The *clique r-domination problem* is to find a clique r-dominating set with minimum size and the *clique r-packing problem* is to find a clique r-packing set with maximum size. The formulated problems include many domination and clique-transversal-related problems as special cases. In this paper an efficient algorithm is proposed for solving these problems on dually chordal graphs which are a natural generalization of strongly chordal graphs. The efficient algorithm is mainly based on the tree structure and special vertex elimination orderings of dually chordal graphs. In some important particular cases where the algorithm works in linear time the obtained results generalize and improve known results on strongly chordal graphs.

Key words. graphs, hypergraphs, tree structure, hypertrees, covering and packing problems, transversal and matching problems, dually chordal graphs, clique hypergraphs, generalization of strongly chordal graphs

AMS subject classifications. 05C65, 05C70, 68Q25, 68R10

PII. S0895480194267853

1. Introduction.

Strongly chordal graphs introduced in [13, 9] and [19] are a well-known subclass of chordal graphs [10] for which several problems remaining NP complete for chordal graphs are efficiently solvable. Among them are the problems of r-domination [7], clique transversal [8], and k-neighborhood covering [17] which are solved in linear time if a *strong elimination ordering* of a strongly chordal graph is given together with the input graph. The best algorithms for finding strong elimination orderings are not linear time.

A very natural generalization of strong elimination orderings is given by *maximum neighborhood orderings* [2]. These orderings lead to *dually chordal graphs* [5, 12, 24]—a generalization of strongly chordal graphs. For a dually chordal graph a maximum neighborhood ordering can be computed in linear time.

For dually chordal graphs in [4] a linear time solution of the r-domination problem is given using only a maximum neighborhood ordering.

In this paper we present a unified method to solve different types of clique r-domination and clique r-packing problems on dually chordal graphs. In some particular cases the obtained results generalize and improve results of [7, 8] and [17] on strongly chordal graphs.

2. Problem formulations.

Let $G = (V, E)$ be a finite connected simple (i.e., without loops and multiple edges) and undirected graph. A *clique* is a subset of pairwise adjacent vertices of V. A *maximal clique* is a clique that is not a proper

* Received by the editors May 4, 1994; accepted for publication (in revised form) February 20, 1996.
http://www.siam.org/journals/sidma/10-1/26785.html
† Universität Rostock FB Informatik, D 18051 Rostock, Germany (ab@informatik.uni-rostock.de).
‡ Department of Mathematics and Cybernetics, Moldova State University, A. Mateevici Str. 60, Chişinău 277009 Moldova (chepoi@cinf.usm.md, dragan@cinf.usm.md). The research of these authors was supported by the VW–Stiftung Project No. I/69041.

subset of any other clique. The *distance* $d(u,v)$ between vertices $u,v \in V$ is the length (i.e., number of edges) of a shortest path connecting u and v. The *disk* centered at vertex v with radius k is the set of all vertices having distance at most k to v:

$$N^k[v] = \{u \in V : d(v,u) \leq k\}.$$

For a clique C and vertex $v \in V$ we denote by

$$\delta(v,C) = \max\{d(v,u) : u \in C\}$$

the *deviation* of v from C. Evidently, $\delta(v,C)$ is the smallest integer $k \geq 0$ such that $C \subseteq N^k[v]$.

Let $\mathcal{C}$ be a family of cliques of a graph G. Suppose that each clique C of $\mathcal{C}$ is associated with an integer $r(C)$, where $r(C) \geq 0$ and $r(C) > 0$ for cliques with size $|C| > 1$. A vertex v *r-dominates* a clique C of G if $\delta(v,C) \leq r(C)$, i.e., $C \subseteq N^{r(C)}[v]$. (For cliques of size $|C| > 1$ and $r(C) = 0$ there is no vertex v which r-dominates C.) A subset $D \subseteq V$ is a *clique r-dominating set* of G if for every clique $C \in \mathcal{C}$ there is a vertex $u \in D$ which r-dominates C. A *clique r-packing set* is a subset $P \subseteq \mathcal{C}$ such that there are no two distinct cliques $C', C'' \in P$ r-dominated by a common vertex of G. The *clique r-domination problem* is to find a clique r-dominating set with minimum size $\gamma_{\mathcal{C},r}(G)$, and the *clique r-packing problem* is to find a clique r-packing set with maximum size $\pi_{\mathcal{C},r}(G)$. Then $\gamma_{\mathcal{C},r}(G)$ and $\pi_{\mathcal{C},r}(G)$ are called the *clique r-domination* and *clique r-packing numbers* of G.

The formulated problems include many domination and clique-transversal-related problems as special cases. First, if $\mathcal{C}$ is the family of maximal cliques of G and $r(C) = 1$ for all $C \in \mathcal{C}$ then we obtain the *clique-transversal* and the *clique-independence problems* [8]. If $\mathcal{C}$ is a family of edges of G and $r(C) = k$ (where k is an integer) for all $C \in \mathcal{C}$ then we obtain the *k-neighborhood covering* and *k-neighborhood independence problems* considered in [17]. Finally, if $\mathcal{C}$ consists only of single vertices of G then we obtain the *r-domination* and *r-packing problems* [22, 19, 7, 12, 11, 4] whose particular instances are the *domination, k-domination, packing*, and *k-packing problems* [14, 9].

3. Dually chordal graphs. In this section we recall the definitions and some results and algorithms on dually chordal graphs, which we use in what follows.

For a vertex $v \in V$ of a graph $G = (V,E)$ we denote by $N(v)$ the *open neighborhood* $N(v) = \{v : uv \in E\}$ and by $N[v] = N(v) \cup \{v\}$ the *closed neighborhood*. For $Y \subseteq V$ let $G(Y)$ be the *subgraph induced by* Y. For a graph G with the vertex set $V = \{v_1, \ldots, v_n\}$ let $G_i = G(\{v_i, v_{i+1}, \ldots, v_n\})$ and $N_i[v]$ ($N_i(v)$) be the *closed (open) neighborhood* of v in G_i.

A vertex v is *simplicial* if and only if (iff) $N[v]$ is a clique. The ordering $(v_1, \ldots, v_n)$ of V is a *perfect elimination ordering* iff for all $i \in \{1, \ldots, n\}$ the vertex v_i is simplicial in G_i. The graph G is *chordal* iff G has a perfect elimination ordering [15].

The ordering $(v_1, \ldots, v_n)$ is a *strong elimination ordering* iff for all $i \in \{1, \ldots, n\}$ $N_i[v_j] \subseteq N_i[v_k]$ when $v_j, v_k \in N_i[v_i]$ and $j < k$. The graph G is *strongly chordal* iff G has a strong elimination ordering [13].

A vertex $u \in N[v]$ is a *maximum neighbor* of v iff for all $w \in N[v]$ the inclusion $N[w] \subseteq N[u]$ holds (note that $u = v$ is not excluded). The ordering $(v_1, \ldots, v_n)$ is a *maximum neighborhood ordering* if for all $i \in \{1, \ldots, n\}$ there is a maximum neighbor $u_i \in N_i[v_i]$:

$$\text{for all } w \in N_i[v_i], N_i[w] \subseteq N_i[u_i] \text{ holds.}$$

The graph G is *dually chordal* [5] iff G has a maximum neighborhood ordering. The graph G is *doubly chordal* [20] iff G is chordal and dually chordal.

There is a close connection between chordal and dually chordal graphs which can be expressed in terms of hypergraphs (for hypergraph notions we follow [3]). Let $\mathcal{N}(G) = \{N[v] : v \in V\}$ be the (*closed*) *neighborhood hypergraph* of G, and let $\mathcal{C}(G) = \{C : C \text{ is a maximal clique of } G\}$ be the *clique hypergraph* of G. By $\mathcal{D}(G) = \{N^k[v] : v \in V, k \text{ a nonnegative integer}\}$ we denote the *disk hypergraph* of G.

Now let $\mathcal{E}$ be a hypergraph with underlying vertex set V, i.e., $\mathcal{E}$ is a set of subsets of V. The *dual hypergraph* $\mathcal{E}^*$ has $\mathcal{E}$ as its vertex set and $\{e \in \mathcal{E} : v \in e\}$ $(v \in V)$ as its edges. The *underlying graph* (or *two-section graph*) $\Gamma(\mathcal{E})$ of the hypergraph $\mathcal{E}$ has vertex set V and two distinct vertices are adjacent iff they are contained in a common edge of $\mathcal{E}$. The *line graph* $L(\mathcal{E}) = (\mathcal{E}, E)$ of $\mathcal{E}$ is the intersection graph of $\mathcal{E}$, i.e., $ee' \in E$ iff $e \cap e' \neq \emptyset$. A *partial hypergraph* of hypergraph $\mathcal{E}$ has V as the underlying vertex set and some edges of $\mathcal{E}$.

A hypergraph $\mathcal{E}$ is a *hypertree* (called *arboreal hypergraph* in [3]) iff there is a tree T with vertex set V of $\mathcal{E}$ such that every edge $e \in \mathcal{E}$ induces a subtree in T. Equivalently, $\mathcal{E}$ is a hypertree iff the line graph $L(\mathcal{E})$ is chordal and $\mathcal{E}$ has the *Helly property*, i.e., any pairwise intersecting subfamily of edges of $\mathcal{E}$ has a common vertex; see [3]. A hypergraph $\mathcal{E}$ is a *dual hypertree* (*α-acyclic hypergraph*) iff there is a tree T with vertex set $\mathcal{E}$ such that for all vertices $v \in V$ $T_v = \{e \in \mathcal{E} : v \in e\}$ induces a subtree of T, i.e., $\mathcal{E}^*$ is a hypertree.

THEOREM 3.1 (see [12], [5]). *Let $G = (V, E)$ be a graph. Then the following conditions are equivalent:*

(i) *G is a dually chordal graph,*
(ii) *$\mathcal{N}(G)$ is a hypertree,*
(iii) *$\mathcal{D}(G)$ is a hypertree,*
(iv) *$\mathcal{C}(G)$ is a hypertree,*
(v) *G is the underlying graph of a hypertree.*

It is well known [6] that G is chordal iff $\mathcal{C}(G)$ is a dual hypertree, i.e., G is the underlying graph of some dual hypertree. Therefore the equivalence of parts (i) and (iv) of Theorem 3.1 justifies the name "dually chordal graphs" for graphs with maximum neighborhood ordering.

Since hypertrees are the dual hypergraphs of α-acyclic hypergraphs by Theorem 3.1 we immediately get that dually chordal graphs can be recognized in time proportional to the size of the corresponding hypergraph. This is a consequence of the linear time algorithm for testing α-acyclicity of hypergraphs [25]. It is easy to see that the hypergraph $\mathcal{N}(G)$ used in Theorem 3.1 has size proportional to the number of edges of the corresponding graphs. Therefore it can be tested in linear time $O(|E|)$ whether a graph $G = (V, E)$ is dually chordal.

Below we present a special algorithm for determining a maximum neighborhood ordering of dually chordal graphs. Its complexity is $O(|E|)$; for more details and a correctness proof we refer to [4].

ALGORITHM 3.2. **MNO** *(Find a maximum neighborhood ordering of G)*
Input: *A dually chordal graph $G = (V, E)$ with $|V| = n > 1$.*
Output: *A maximum neighborhood ordering of G.*

(0) *initially all $v \in V$ are unnumbered and unmarked*;
(1) *choose an arbitrary vertex $v \in V$, number v with n, i.e., $v_n = v$, and let $mn(v_n) := v$;*

repeat

(2) *among all unmarked vertices select a numbered vertex u such that $N[u]$ contains a maximum number of numbered vertices*;
(3) *number all unnumbered vertices x from $N[u]$ consecutively with maximal possible numbers between 1 and $n-1$ which are still free*;
for all of them let $mn(x) := u$;
(4) *mark u*;

until *all vertices are numbered*

The meaning of $mn(x)$ is a maximum neighbor of x. Note that the algorithm also yields a maximum neighbor for each vertex, and all vertices of $N[v_n]$ occur consecutively in the ordering on the left of v_n and have v_n as their maximum neighbor. Furthermore, since G is assumed to be connected, for all v_i with $i \leq n-1$, $mn(v_i) \neq v_i$ holds.

Maximum neighborhood orderings of graphs produced by the MNO algorithm immediately lead to optimal algorithms for computing the distance matrix for all graphs having such orderings. Let $(v_1, \ldots, v_n)$ be a maximum neighborhood ordering of a graph G produced by the MNO algorithm. Subsequently we only use such maximum neighborhood orderings. This assumption is of crucial importance for the correctness of the main algorithm. The maximum neighbor $mn(v_i)$ of the vertex v_i in $G(\{v_i, v_{i+1}, \ldots, v_n\})$ has an important metric property: for every vertex $v_j, j > i$, which is nonadjacent to v_i there exists a shortest path of $G(\{v_i, v_{i+1}, \ldots, v_n\})$ between v_i and v_j which passes through $mn(v_i)$.

To see this assume by way of contradiction that all shortest paths between v_i and v_j contain vertices not in G_i. Among these paths choose a path P whose leftmost vertex u with respect to the maximum neighborhood ordering $(v_1, \ldots, v_n)$ has rightmost position. Let v, w be the neighbors of u in P. Since P is a shortest path the distance of v, w is 2. Now the maximum neighbor $mn(u)$ which according to the MNO algorithm is distinct from u is also adjacent to v and w and on the right of u. Thus replacing u by $mn(u)$ in P we obtain a shortest path P' between v_i and v_j whose leftmost vertex is on the right of u—a contradiction.

In particular, we obtain that every graph $G(\{v_{i+1}, v_{i+2}, \ldots, v_n\})$ is a distance-preserving subgraph of $G(\{v_i, v_{i+1}, \ldots, v_n\})$. Thus it follows that $G(\{v_i, v_{i+1}, \ldots, v_n\})$ is a distance-preserving subgraph of G for all $i \in \{1, \ldots, n\}$.

Let $G = (V, E)$ be a dually chordal graph, and let $D(G) = (d(v_i, v_j))_{i,j \in \{1,\ldots,n\}}$ denote the distance matrix of G. By $D_{i+1}(G)$ we denote the submatrix of $D(G)$ which contains the distances between the vertices $v_{i+1}, \ldots, v_n$. The next submatrix $D_i(G)$ is obtained from $D_{i+1}(G)$ by adding the ith row and ith column according to the following rule:

for all $k > i$ define

$$d(v_i, v_k) = d(v_k, v_i) = \begin{cases} 1 & \text{if } v_i \text{ and } v_k \text{ are adjacent,} \\ d(mn(v_i), v_k) + 1 & \text{otherwise.} \end{cases}$$

Evidently, this procedure correctly finds the whole matrix $D(G)$ in optimal time $O(n^2)$. Moreover, the maximum neighborhood ordering of G for every two query vertices u and v allows us to find in time $O(c \cdot d(u, v))$ a shortest path between u and v (c is the necessary time to verify the adjacency of two vertices). Let $num(v) = i$ if $v = v_i$ in the maximum neighborhood ordering of G.

PROCEDURE 3.3 (**sh–path**(u, v)**).**
if *u and v are adjacent* **then return** (u, v)
else

if $num(u) < num(v)$ **then**
  return $(u,sh\text{–}path(mn(u), v))$
else
  return $(sh\text{–}path(u, mn(v)), v))$

According to [4] the shortest path between two vertices u and v of a dually chordal graph G computed by procedure $sh\text{–}path(u, v)$ is called a *maximum neighbor path*. Such a path $P(u, v)$ has an important property: it splits into two subpaths $P' = (u_0, u_1, \ldots, u_p)$ and $P'' = (v_0, v_1, \ldots, v_q)$, where $u_0 = u$ and $v_0 = v$ such that each u_i is the maximum neighbor of u_{i-1} and each v_i is the maximum neighbor of v_{i-1} and vertices u_p, v_q are adjacent.

We conclude this section with the following property of cliques of dually chordal graphs.

LEMMA 3.4. *Let G be a dually chordal graph and C be a clique of G. If the vertex v r-dominates C then there exists a vertex v^* with $d(v, v^*) = \delta(v, C) - 1$ which 1-dominates C.*

Proof. If $d(v, u) < \delta(v, C)$ for some vertex $u \in C$, then u is the required vertex. Thus assume that all vertices of C are equidistant from v. Applying the Helly property to the family of pairwise intersecting disks consisting of $N^{\delta(v,C)-1}[v]$ and closed neighborhoods of vertices of C, we obtain a vertex v^* adjacent to all vertices of C and satisfying the required equality $d(v, v^*) = \delta(v, C) - 1$. □

4. Hypergraph approach to clique r-domination and clique r-packing. For a family of cliques $\mathcal{C}$ of a graph G and a function $r : \mathcal{C} \to N \cup \{0\}$ with $r(C) > 0$ for cliques of size $|C| > 1$ define the hypergraph $\mathcal{D}_{\mathcal{C},r}(G)$ as follows:

$$\mathcal{D}_{\mathcal{C},r}(G) = \left\{ \bigcap_{v \in C} N^{r(C)}[v] : C \in \mathcal{C} \right\}$$

where $\bigcap_{v \in C} N^{r(C)}[v] = \{x : x \text{ } r\text{-dominates } C\}$. (Note that this notation can be used in the same way for families of arbitrary vertex sets instead of cliques.)

Using this notation, the clique r-domination and clique r-packing problems on G may be formulated as the transversal and matching problems on the hypergraph $\mathcal{D}_{\mathcal{C},r}(G)$. Recall that a *transversal* of a hypergraph $\mathcal{E}$ is a subset of vertices which meets all edges of $\mathcal{E}$. A *matching* of $\mathcal{E}$ is a subset of pairwise disjoint edges of $\mathcal{E}$. For a hypergraph $\mathcal{E}$, the *transversal problem* is to find a transversal with minimum size $\tau(\mathcal{E})$, and the *matching problem* is to find a matching with maximum size $\nu(\mathcal{E})$. From the definitions we obtain the following lemma.

LEMMA 4.1. *D is a clique r-dominating set of a graph G iff D is a transversal of $\mathcal{D}_{\mathcal{C},r}(G)$. P is a clique r-packing set of a graph G iff P is a matching of $\mathcal{D}_{\mathcal{C},r}(G)$. Thus $\tau(\mathcal{D}_{\mathcal{C},r}(G)) = \gamma_{\mathcal{C},r}(G)$ and $\nu(\mathcal{D}_{\mathcal{C},r}(G)) = \pi_{\mathcal{C},r}(G)$ hold for every graph G and every function $r : \mathcal{C} \to N \cup \{0\}$ defined on every family $\mathcal{C}$ of cliques.*

The parameters $\gamma_{\mathcal{C},r}(G)$ and $\pi_{\mathcal{C},r}(G)$ are always related by a min–max duality inequality $\gamma_{\mathcal{C},r}(G) \geq \pi_{\mathcal{C},r}(G)$. The next result shows that for dually chordal graphs the converse inequality holds.

LEMMA 4.2. *For every family of cliques $\mathcal{C}$ of a dually chordal graph G and every function $r : \mathcal{C} \to N \cup \{0\}$ the hypergraph $\mathcal{D}_{\mathcal{C},r}(G)$ is a hypertree. In particular, the equality $\gamma_{\mathcal{C},r}(G) = \pi_{\mathcal{C},r}(G)$ holds.*

Proof. By definition each edge of the hypergraph $\mathcal{D}_{\mathcal{C},r}(G)$ is the intersection of some disks of G. By Theorem 3.1 the disk hypergraph $\mathcal{D}(G)$ of G is a hypertree. This means that there exists a tree T such that each edge of $\mathcal{D}(G)$ is a subtree of T.

Since the intersection of subtrees is a subtree too, all edges of $\mathcal{D}_{\mathcal{C},r}(G)$ are subtrees of T. Therefore the hypergraph $\mathcal{D}_{\mathcal{C},r}(G)$ is a hypertree. It is well known [3] that the equality $\tau(\mathcal{E}) = \nu(\mathcal{E})$ holds for every hypertree. By Lemma 4.1 we obtain the required equality. □

Note that Lemma 4.2 is true not only for families of cliques but also for arbitrary families of vertex sets of a dually chordal graph assuming only that the given r-values do not lead to empty hyperedges. (For hypergraphs with empty edges there is no transversal.)

It is known that in a hypertree $\mathcal{E}$ the transversal and matching problems can be solved in time proportional to the size of $\mathcal{E}$ [25]. The hypergraph $\mathcal{D}_{\mathcal{C},r}(G)$ can be computed in $O(|V|^2 + |V|\sum_{i=1}^{p}|C_i|)$ time for $\mathcal{C} = \{C_1, C_2, \ldots, C_p\}$. This can be done by applying the distance matrix of G. As we already mentioned this matrix can be computed in optimal time $O(|V|^2)$. In the worst case the hypergraph $\mathcal{D}_{\mathcal{C},r}(G)$ has size $O(|V||\mathcal{C}|)$. Thus the whole time to solve the clique r-domination and clique r-packing problems is $O(|V|^2 + |V|\sum_{i=1}^{p}|C_i|)$. Below we present an algorithm of the same complexity for solving these problems, which avoids the construction of the hypergraph $\mathcal{D}_{\mathcal{C},r}(G)$. For two particular cases when $\mathcal{C}$ consists only of maximal cliques or only of vertices of G its complexity becomes linear. The algorithm simultaneously finds a clique r-dominating set D and a clique r-packing set P such that $|D| = |P|$. This provides an algorithmic proof of duality results between these two problems on dually chordal graphs.

We refer also to [1], where similar duality results were used to provide efficient algorithms for three covering and packing problems on families of subtrees of a tree.

5. The algorithm. Let $G = (V, E)$ be a dually chordal graph, $\mathcal{C}$ be an arbitrary family of cliques of G, and $(v_1, \ldots, v_n)$ be the ordering of V generated by the MNO algorithm. By $r(C_1), \ldots, r(C_p)$ we denote the dominating radii of the corresponding cliques of $\mathcal{C}$. The algorithm processes the vertices in the order from v_1 to v_n. In iteration i the algorithm decides whether the vertex v_i has to be put into the clique r-dominating set D. If v_i is included in D then a certain clique C which is r-dominated by v_i is included in the clique r-packing set P. Initially, both sets D and P are empty. After processing, vertex v_i is deleted from the graph and information concerning whether or not v_i was included in D is given to its maximum neighbor $mn(v_i)$ and/or to its other neighbors and cliques from $N_i(v_i)$.

For technical reasons we extend the initial family of cliques $\mathcal{C}$ by including in $\mathcal{C}$ as one-vertex cliques all vertices $v \in V$ such that $\{v\} \notin \mathcal{C}$. For each of them initially $r(\{v\}) = \infty$. Let $\mathcal{C}(v)$ be a set of all cliques of $\mathcal{C}$ that contain the vertex v. At the next step we redefine the r-dominating radii of one-vertex cliques by putting $r(\{v\}) = \min\{r(C) : C \in \mathcal{C}(v)\}$. As in [7, 4] we associate to each clique $C \in \mathcal{C}$ the dominating radius $r(C)$ and the nonnegative integer $a(C)$. Initially $a(C) = \infty$ for all $C \in \mathcal{C}$. $a(C)$ keeps decreasing during the execution of the algorithm, while the dominating radii decrease only for one-vertex cliques. At each step $r(\{v\})$ becomes the current radius within which the clique $\{v\}$ and all other yet undominated cliques from $\mathcal{C}(v)$ must be r-dominated in the remaining graph. Unlike the dominating radii of cliques the value $a(C)$ indicates an upper bound for the distances of vertices $v \in D$ from the current clique r-dominating set D to the clique C; i.e., there is a $v \in D$ such that $d(v, C) \le a(C)$. The value of $r(\{v_i\})$ decreases in the case where v_i is the maximum neighbor of a vertex v_j, $j < i$, such that there is a clique $C \in \mathcal{C}(v_j)$ that is not properly r-dominated by a vertex of D within distance $r(C)$ in iteration j. Then necessarily $r(C) = r(\{v_j\})$; for a proof see Lemma 5.5. In this case, $r(\{v_i\})$ is set to

be $r(\{v_j\})-1$. Similarly, in a previous iteration, $r(\{v_j\})$ is set to be $r(\{v_k\})-1, k<j$. Continuing this argument, we find that there is a smallest (i.e., leftmost) vertex v_{i^*} and a clique $C^* \in \mathcal{C}(v_{i^*})$ that forces $\ldots, k, j, i$ to decrease their $r(\cdot)$ values, although $r(\{v_{i^*}\})$ never changes. We use $fn(\{v_i\})$ (*clique furthest neighbor* of $\{v_i\}$) to denote this initial clique C^* from which $r(\{v_i\})$ decreases.

In step i the algorithm processes vertex v_i according to the following rules. When $r(\{v_i\})=0$ then v_i must be in D because no other vertex r-dominates $\{v_i\}$. Moreover, we include $fn(\{v_i\})$ in P and set $a(\{v_i\})=0$. Otherwise, if $r(\{v_i\})>0$ then we distinguish between two cases. Either we find a clique $C \in \mathcal{C}(v_i)$ which is not yet r-dominated (lines (8) and (9)) or we establish that all cliques of $\mathcal{C}(v_i)$ are properly r-dominated by vertices of D in iteration i. In the second case we do nothing. So suppose that the first case holds. Then either C coincides with $\{v_i\}$ or C contains v_i as the smallest vertex in the MNO ordering and $r(C)=r(\{v_i\})$. In both cases we update $r(\{mn(v_i)\})$ by

$$r(\{mn(v_i)\}) = \min\{r(\{mn(v_i)\}), r(\{v_i\})-1\}.$$

At each step we have to update $a(C)$ for all $C \subseteq N_i(v_i)$:

$$a(C) = \min\{a(C), a(\{v_i\})+1\}.$$

The algorithm solves the transversal and matching problems on the hypergraph $\mathcal{D}_{\mathcal{C},r}(G)$ without constructing this hypergraph and works in linear time when $\mathcal{C}$ consists only of maximal cliques or only of vertices of G.

ALGORITHM 5.1. **(CRDP)** *(Find a minimum clique r-dominating set and a maximum clique r-packing set of a dually chordal graph G)*

Input: *A dually chordal graph $G=(V,E)$ with a maximum neighborhood ordering $(v_1,\ldots,v_n)$ obtained by the MNO algorithm and a family $\mathcal{C}=\{C_1,\ldots,C_p\}$ of cliques in G with radii $r(C_1),\ldots,r(C_p)\geq 0$ and $r(C_i)>0$ for $|C_i|>1$*

Output: *A minimum clique r-dominating set D and a maximum clique r-packing set P of G*

(1) $D:=\emptyset; P:=\emptyset;$
(2) **for all** $v \in V$ **such that** $\{v\} \notin \mathcal{C}$ **do begin** $\mathcal{C}:=\mathcal{C}\cup\{\{v\}\}$; $r(\{v\}):=\infty$ **end**;
(3) **for all** $C \in \mathcal{C}$ **do begin** $a(C):=\infty$; $fn(C):=C$ **end**;
(4) **for all** $v \in V$ **do**
  begin
    choose a clique C from $\mathcal{C}(v)$ with minimal radius $r(\cdot)$;
    $r(\{v\}):=r(C)$; $fn(\{v\}):=C$
  end;
(5) **for** $i:=1$ **to** $n-1$ **do**
  begin
    $par:=1$;
(6) **if** $r(\{v_i\})=0$ **then**
      begin
(7) $D:=D\cup\{v_i\}; P:=P\cup\{fn(\{v_i\})\}; a(\{v_i\}):=0$
      end
    else
      begin
(8) **if** $a(\{v_i\})>r(\{v_i\})$ **and** $\forall v \in N_i(v_i)\ a(\{v\})+1>r(\{v_i\})$ **and** $r(\{v\})>0$
        then $C:=\{v_i\}$
      else

(9) **if** $\exists C' \in \mathcal{C}(v_i)$ **such that** $[i = \min\{num(v) : v \in C'\}$ **and** $a(C') > r(C')$
and $(\forall v \in N_i[v_i]$
$(a(\{v\}) + 1 > r(C')$ **or** $(a(\{v\}) + 1 = r(C')$ **and** $C' \not\subseteq N[v]))$
and $(r(\{v\}) > 0$ **or** $r(\{v\}) = 0$ **and** $r(C') = 1$ **and** $C' \not\subseteq N[v]))]$
then $C := C'$
else
(10) $par := 0$;
(11) **if** $par = 1$ **and** $r(\{mn(v_i)\}) \geq r(\{v_i\})$ **then**
begin
(12) $r(\{mn(v_i)\}) := r(\{v_i\}) - 1$;
(13) $fn(\{mn(v_i)\}) := fn(C)$
end;
end;
(14) **for all** $C \in \mathcal{C}$ **such that** $C \subseteq N_i(v_i)$ **do** $a(C) := \min\{a(C), a(\{v_i\}) + 1\}$;
end;
(15) **if** $r(\{v_n\}) < a(\{v_n\})$ **then** $D := D \cup \{v_n\}$ **and** $P := P \cup \{fn(\{v_n\})\}$

Subsequently for one-vertex cliques the brackets { and } are omitted.

THEOREM 5.2. *Algorithm CRDP is correct.*

Proof. In order to prove that the set D constructed by the algorithm is clique r-dominating we use the following reformulation of the clique r-domination problem in terms of $r(C)$ and $a(C)$:

find a minimum size set $D \subseteq V$ such that for every clique $C \in \mathcal{C}$
(a) $a(C) \leq r(C)$, or
(b) $\delta(v, C) \leq r(C)$ for some $v \in D$, or
(c) $\delta(u, C) + a(u) \leq r(C)$ for some $u \in V$ (D dominates C via vertex u).

(Note that the cases (a),(b),(c) do not exclude each other—this case distinction is useful subsequently for technical purposes.)

The proof of the theorem is based on some auxiliary results. In all of them let $\mathcal{C}_1 = \mathcal{C}$ and $\mathcal{C}_{i+1} = \mathcal{C}_i \setminus \mathcal{C}(v_i)$, where $\mathcal{C}(v_i)$ are all cliques of $\mathcal{C}$ which contain the vertex v_i.

LEMMA 5.3. *If $r(v_i) = 0$ then D is a clique r-dominating set of $\mathcal{C}_i$ in G_i iff $D = D' \cup \{v_i\}$, where D' is a clique r-dominating set of $\mathcal{C}_{i+1}$ in G_{i+1} with $a(C) := 1$ for all cliques $C \subseteq N_i(v_i)$ and $a(C) := a(C)$ otherwise, and $r(C) := r(C)$ for all cliques $C \in \mathcal{C}_{i+1}$.*

Proof. If $r(v_i) = 0$ then the one-vertex clique $\{v_i\}$ is not r-dominated by any other vertex of G_i. So, necessarily v_i belongs to every clique r-dominating set of G_i, in particular $v_i \in D$. Then v_i r-dominates every clique $C \subseteq N_i[v_i]$ with $r(C) > 0$. Let $D' = D \setminus \{v_i\}$. Assume that D' is not a clique r-dominating set for $\mathcal{C}_{i+1}$, i.e., some clique $C \in \mathcal{C}_{i+1}$ is not dominated by D'. Evidently, $C \not\subseteq N_i[v_i]$. Then in both graphs G_i and G_{i+1} the clique C has the same values for $r(C)$ and $a(C)$. Since C is r-dominated by D this is possible only if $\delta(u, C) + a(u) \leq r(C)$ for some u of G_i or $\delta(v_i, C) \leq r(C)$. Consider the second case. By Lemma 3.4 there exists a vertex v^* with $d(v_i, v^*) = \delta(v_i, C) - 1$ which dominates C. Let w be a neighbor of v_i which belongs to a shortest path between v_i and v_i^*. Since $C \not\subseteq N_i[v_i]$ such a vertex w always exists. Then in G_{i+1} $a(w) = 1$ and $\delta(w, C) + a(w) \leq r(C)$. This means that C is r-dominated by D'. Next suppose that $\delta(u, C) + a(u) \leq r(C)$. Necessarily $u \in N_i(v_i)$ or $u = v_i$. In the first case we obtain a similar inequality $\delta(u, C) + a(u) \leq r(C)$ in G_{i+1} too because $a(u)$ does not increase in G_{i+1}. Otherwise, if $\delta(v_i, C) + a(v_i) \leq r(C)$ then for the vertex $w \in N_i(v_i)$ introduced above we have $\delta(w, C) = \delta(v_i, C) - 1$ and

$a(w) \leq a(v_i) + 1$. This means that $\delta(w, C) + a(w) \leq r(C)$. Thus D' is a clique r-dominating set of $\mathcal{C}_{i+1}$ in G_{i+1}. Conversely if $\delta(u, C) + a(u) \leq r(C)$ for some clique $C \in \mathcal{C}_{i+1}$ and some vertex $u \in N_i(v_i)$ then $\delta(v_i, C) \leq r(C)$ and C is r-dominated in G_i from v_i. Therefore if D' is a clique r-dominating set for $\mathcal{C}_{i+1}$ then $D' \cup \{v_i\}$ is a clique r-dominating set for $\mathcal{C}_i$ in G_i. □

LEMMA 5.4. *Suppose that $r(v_i) \geq 1$ and that the conditions of lines* (8) *and* (9) *are not fulfilled. A subset $D \subseteq \{v_{i+1}, \ldots, v_n\}$ is a clique r-dominating set of $\mathcal{C}_i$ in G_i iff D is a clique r-dominating set of $\mathcal{C}_{i+1}$ in G_{i+1} with $r(C) := r(C)$ for all $C \in \mathcal{C}_{i+1}$, and $a(C) := \min\{a(C), a(v_i) + 1\}$ when $C \subseteq N_i(v_i)$ and $a(C) := a(C)$ otherwise.*

Proof. Since the values of $a(\cdot)$ do not increase from left to right along the ordering $(v_1, \ldots, v_n)$ each clique r-dominating set $D \subseteq \{v_{i+1}, \ldots, v_n\}$ of $\mathcal{C}_i$ in G_i is clique r-dominating in G_{i+1} too.

Conversely, suppose that D is a clique r-dominating set of $\mathcal{C}_{i+1}$ in G_{i+1}. Pick an arbitrary clique C of $\mathcal{C}_i$. By the conditions of the lemma every clique of $\mathcal{C}(v_i)$ is already r-dominated: let $C \in \mathcal{C}(v_i)$. Recall that the conditions of (8) and (9) are not fulfilled. Condition (8) is not fulfilled iff $a(\{v_i\}) \leq r(\{v_i\})$ or there exists $v \in N_i(v_i)(a(\{v\}) + 1 \leq r(\{v_i\})$ or $r(\{v\}) = 0)$. Consequently $\{v_i\}$ is dominated by D. Condition (9) is not fulfilled iff

$$\forall C' \in \mathcal{C}(v_i)(a(C') \leq r(C'))$$

(i.e., D dominates C') or

$$\exists v \in N_i(v_i)(a(\{v\}) + 1 \leq r(C') \text{ and } (a(\{v\}) + 1 \neq r(C') \text{ or } C' \subseteq N[v]))$$

(if even $a(\{v\}) + 2 \leq r(C')$ holds then D dominates C' via v; if $a(\{v\}) + 1 = r(C')$ then $C' \subseteq N[v]$ and thus also D dominates C' via v) or

$$r(\{v\}) = 0 \text{ and } (r \neq 0 \text{ or } r(C') \neq 1 \text{ or } C' \subseteq N[v])$$

($r(\{v\}) = 0$ means that $v \in D$, $r(v) \neq 0$ does not hold; thus $r(C') > 1$ since $r(v_i) \leq 1$ by the suppositions of the lemma and thus because of $r(\{v\}) = 0$ also C' is dominated or ($r(C') = 1$ and $C' \subseteq N[v]$) in which case C' is also dominated).

Thus it is enough to consider only the case when $C \in \mathcal{C}_{i+1}$. If $a(C) \leq r(C)$ in G_{i+1} then the same inequality holds in G_i too, except the case when $C \subseteq N_i(v_i)$. Then $a(v_i) + \delta(v_i, C) \leq a(v_i) + 1 \leq a(C) \leq r(C)$. If the clique C is r-dominated by a vertex $v \in D$ in G_{i+1} then v dominates C in G_i too. This is so because G_{i+1} is a distance preserving subgraph of G_i. Therefore it is sufficient to consider only the case when $\delta(u, C) + a(u) \leq r(C)$ holds in G_{i+1} for some vertex $u \in N_i(v_i)$ (i.e., for some one-vertex clique $\{u\} \subseteq N_i(v_i)$). If $\min\{a(u), a(v_i) + 1\} = a(u)$ then we are done. Otherwise, since $\delta(v_i, C) \leq \delta(u, C) + 1$ we obtain that

$$\delta(v_i, C) + a(v_i) \leq \delta(u, C) + a(v_i) + 1 \leq r(C).$$

Hence, D is a clique r-dominating set of $\mathcal{C}_i$ in G_i. □

LEMMA 5.5. *Assume that C^+ is a clique of $\mathcal{C}(v_i)$ obtained in lines* (8), (9) *of the algorithm. A subset $D \subseteq (\{v_{i+1}, \ldots, v_n\} \setminus \{v \in N_i(v_i) : r(v) \neq 0\}) \cup \{mn(v_i)\}$ is a clique r-dominating set of $\mathcal{C}_i$ in G_i iff D is a clique r-dominating set of $\mathcal{C}_{i+1}$ in G_{i+1} with $a(C) := \min\{a(C), a(v_i) + 1\}$ for $C \subseteq N_i(v_i)$ and $a(C) := a(C)$ for all other cliques of $\mathcal{C}_{i+1}$, and $r(C) := r(C)$ for all $C \in \mathcal{C}_{i+1} \setminus \{\{mn(v_i)\}\}$ and $r(mn(v_i)) := \min\{r(v_i) - 1, r(mn(v_i))\}$.*

Proof. 1. "$\Longrightarrow$": From our conditions we immediately get that if D' is a clique r-dominating set of G_{i+1} then replacing all vertices of $D' \cap \{v \in N_i(v_i) : r(v) \neq 0\}$ by $mn(v_i)$ we obtain a clique r-dominating set D with $|D| \leq |D'|$.

First we prove that if the clique C^+ is different from $\{v_i\}$ then $r(C^+) = r(v_i)$. Assume the contrary. Then $r(C^+) > r(v_i)$ because for every vertex v_i and every clique $C \in \mathcal{C}(v_i)$ during the first $i-1$ steps of the algorithm the inequality $r(C) \geq r(v_i)$ holds; see lines (4) and (12) of the algorithm. If $a(v_i) \leq r(v_i)$ then

$$a(v_i) + \delta(v_i, C) = a(v_i) + 1 \leq r(C^+).$$

Next suppose that there exists a vertex $v \in N(v_i)$ such that either $a(v) + 1 \leq r(v_i)$ or $r(v) = 0$. In the first case we immediately get

$$\delta(v, C^+) + a(v) \leq 2 + a(v) \leq r(C^+).$$

Otherwise, if $r(v) = 0$ then by the choice of C^+ we obtain that $r(C^+) = 1$. But then $r(v_i) = 0$, which is impossible. In all cases we get a contradiction with the choice of the clique C^+. Thus $r(C^+) = r(v_i)$.

Let $D \subseteq (\{v_{i+1}, \ldots, v_n\} \setminus \{v \in N_i(v_i) : r(v) \neq 0\}) \cup \{mn(v_i)\}$ be a clique r-dominating set of $\mathcal{C}_i$ in G_i and let C be an arbitrary clique of $\mathcal{C}_{i+1}$. First suppose that $C \neq \{mn(v_i)\}$. Since C is r-dominated by D in G_i in order to establish the same property in G_{i+1} it is enough to assume that $\delta(v_i, C) + a(v_i) \leq r(C)$ in G_i; otherwise we immediately get this. If $C \not\subseteq N_i(v_i)$ then as in Lemma 5.4 we choose a vertex $w \in N_i(v_i)$ such that $\delta(w, C) = \delta(v_i, C) - 1$. In this case we have

$$\delta(w, C) + a(w) \leq \delta(v_i, C) + a(v_i) \leq r(C).$$

If $C \subseteq N_i(v_i)$ then as $a(C) \leq a(v_i) + 1$ and $\delta(v_i, C) = 1$ in G_{i+1} we obtain that $a(C) \leq r(C)$.

Next consider the one-vertex clique $\{mn(v_i)\}$ of G_{i+1}. We can suppose that

$$r(mn(v_i)) = r(v_i) - 1 = r(C^+) - 1;$$

otherwise, we can apply the preceding arguments. In G_i the clique C^+ is r-dominated by D. By the choice of C^+ this means that in G_i either $\delta(u, C^+) + a(u) \leq r(C^+)$ for some vertex $u \notin N_i[v_i]$ or $\delta(u, C^+) \leq r(C^+)$ for some $u \in D \setminus N_i[v_i]$. In both cases the vertex $mn(v_i)$ is one step closer to u than the vertices of C^+. This allows us to conclude that in G_{i+1} either

$$d(mn(v_i), u) + a(u) \leq r(C^+) - 1 = r(mn(v_i))$$

or

$$d(mn(v_i), u) \leq r(C^+) - 1 = r(mn(v_i)).$$

Therefore D is a clique r-dominating set of $\mathcal{C}_{i+1}$ in G_{i+1}.

2. "$\Longleftarrow$": Conversely, let $D \subseteq \{v_{i+1}, \ldots, v_n\}$ be a clique r-dominating set of $\mathcal{C}_{i+1}$ in G_{i+1}. The arguments for proving that D is a clique r-dominating set of $\mathcal{C}_i$ in G_i are the same as in Lemma 5.4 except when C is a clique of $\mathcal{C}(v_i)$. For all such cliques we have $r(C) \geq r(v_i)$. Then every such clique is r-dominated by the same vertex as the clique $\{mn(v_i)\}$ is r-dominated in G_{i+1}, except the case when $a(mn(v_i)) \leq r(mn(v_i))$

in G_{i+1}. Then $mn(v_i)$ r-dominates all cliques of $\mathcal{C}(v_i)$ because for all $C \in \mathcal{C}(v_i)$ we have

$$\delta(mn(v_i), C) + a(mn(v_i)) = 1 + a(mn(v_i)) \leq 1 + r(mn(v_i)) \leq r(v_i) \leq r(C).$$

Thus, D is a clique r-dominating set of $\mathcal{C}_i$ in G_i. □

LEMMA 5.6. *The set $D \subseteq V$ obtained by the algorithm is a clique r-dominating set of $\mathcal{C}$ in G.*

Proof. The proof follows from the preceding three lemmas by induction on the number of vertices and by the fact that the clique r-dominating set of a larger family of cliques remains clique r-dominating for every subfamily too. □

LEMMA 5.7. $|P| = |D|$.

Proof. By the algorithm we know that $|D| \geq |P|$. If $|P| < |D|$ then there are two vertices $u, v \in D$ and a clique $C \in P$ such that $fn(u) = C = fn(v)$. Let w be the vertex of C with the minimal index $num(w)$. By the algorithm, $fn(u)$ and $fn(v)$ reach vertices u and v along maximum neighbor paths connecting w, u and w, v, respectively. Let w^+ be the vertex belonging to both paths that is furthest from w. Denote by u' and v' the next neighbors of w^+ in these paths. Necessarily $u' = v'$ since both are the maximum neighbors $mn(w^+)$. Thus the unique maximum neighbor path from w^+ reaches the unique vertex $u = v$, which is a contradiction to the assumption that there are two vertices $u, v \in D$ with the property $fn(u) = C = fn(v)$. □

LEMMA 5.8. *Let $R = (u_0, u_1, \ldots, u_k)$ be a path between vertices u_0 and u_k such that $u_i = mn(u_{i-1}), i = 1, \ldots, k$. If u is a vertex with $num(u) > num(u_{k-1})$ then either the maximum neighbor path between u_0 and u contains R or u is adjacent to all vertices $u_j, \ldots, u_k$ for some $j < k$.*

Proof. By the conditions of the lemma we get that the procedure *sh–path* will include in the maximum neighbor path between u_0 and u all vertices of R until a neighbor u_j of u is achieved. If $j = k$ then the whole path R belongs to the constructed path. Otherwise, if $j < k$ then because $num(u_j) < \cdots < num(u_k) < num(u)$ and $u_{i+1} = mn(u_i)$ for each $i \leq k - 1$, we obtain that u must be adjacent to all vertices $u_j, u_{j+1}, \ldots, u_k$. □

LEMMA 5.9. *P is a clique r-packing set of $\mathcal{C}$ in G.*

Proof. First of all, note that P is a subset of the initial family of cliques $\mathcal{C}$; see line (4) of the algorithm.

Assume that P is not a clique r-packing set; i.e., there are two cliques $C', C'' \in P$ which are r-dominated by a common vertex of G. In particular we obtain that $d(w', w'') \leq r(C') + r(C'')$ for arbitrary vertices $w' \in C'$ and $w'' \in C''$. According to the algorithm there are vertices $x, y \in D$ such that $C' = fn(x)$ and $C'' = fn(y)$. Let $u \in C'$ and $v \in C''$ be vertices with minimal indices $num(u)$ and $num(v)$ in the cliques C' and C'', respectively. By the algorithm $fn(u) = C'$ and $fn(v) = C''$. Let R' and R'' be maximum neighbor paths between u, x and v, y. Both of these paths are increasing. By the algorithm we obtain that R' and R'' are disjoint; otherwise a common vertex of R' and R'' is a "bottleneck" for the transmission of cliques $C' = fn(x)$ and $C'' = fn(y)$ (see also Lemma 5.7). Moreover, the lengths of these paths are $r(C')$ and $r(C'')$, respectively.

Let R be the maximum neighbor path between vertices u and v. As we know already R splits into two increasing maximum neighbor paths $R_u = (u, \ldots, u^+)$ and $R_v = (v, \ldots, v^+)$ and an edge u^+v^+. Comparing the paths R', R_u and R'', R_v we conclude that they must be comparable with respect to $\subseteq$. Because of $d(u, v) \leq r(C') + r(C'') = r(u) + r(v)$ at least one of the inequalities $|R'| > |R_u|$ or $|R''| > |R_v|$ holds. In particular, at least one of the incidences $u^+ \in R'$ or $v^+ \in R''$ is satisfied.

Case 1. $u^+ \in R'$ and $v^+ \in R''$.

Let R'_0 and R''_0 be subpaths of R' and R'' which connect vertices u^+, x and v^+, y, respectively. Since $d(u,v) \leq r(C') + r(C'')$ at least one of these paths has an edge, i.e., $u^+ \neq x$ or $v^+ \neq y$. Among adjacent vertices $u' \in R'_0$ and $v' \in R''_0$ with $u' \neq x$ or $v' \neq y$, we choose adjacent vertices $u^* \in R'_0$ and $v^* \in R''_0$ whose sum $d(u^*, x) + d(v^*, y)$ is minimal. Assume without loss of generality that $num(u^*) < num(v^*)$.

We claim that $u^* \neq u$ or $v^* \neq v$ holds. Assume to the contrary that $u^* = u$ and $v^* = v$. Then necessarily $r(C') + r(C'') > 0$. If $r(C') > 0$ then the vertex $mn(u^*)$ of R' must be adjacent to v^*. We get a contradiction with the choice of u^* and v^* except when $mn(u^*) = x$ and $v^* = y$. Then $r(C') = 1$ while $r(C'') = 0$ and thus $C'' = \{v\}$. If v is not adjacent to some vertex $u' \in C'$ then $d(v,u') = 2 > r(C') + r(C'')$ in contradiction to the choice of the cliques C' and C''. Otherwise if v is adjacent to all vertices of C' then $C' \subseteq N[v]$ and according to the algorithm we cannot transmit the clique $C' = fn(u)$ to the vertex $mn(u)$. Next assume that $r(C') = 0$, i.e., $C' = \{u\}$ and $x = u^* = u$. Again, as in the preceding case, if $r(C'') = 1$ then u must be adjacent to all vertices of C''. Hence in step $num(v)$ we have $r(C'') = a(C'') = 1$, which is a contradiction. So let $r(C'') \geq 2$. But then in step $num(v)$ we have $\delta(v, C'') + a(v) = 1 + 1 \leq r(C'')$ and according to lines (8) and (9) of the algorithm we cannot insert the clique C'' in P. Thus $u^* \neq u$ or $v^* \neq v$ holds.

Since $num(u^*) < num(v^*)$ we get that either u^* coincides with u or x or v^* is adjacent to the maximum neighbor $mn(u^*)$ of u^*. In the last case we obtain a contradiction with our choice of an edge u^*v^* except the case when $mn(u^*) = x$ and $v^* = y$. In this case we conclude that y is adjacent to x. If $r(C'') = 0$ and $C'' = \{y\}$ then in step $num(u^*)$ we have $r(y) = 0$ for $y \in N(u^*)$. By the algorithm we cannot transmit $C' = fn(u^*)$ to x. So $r(C'') > 0$. Let y^+ be the neighbor of y in P''. If $num(y^+) < num(u^*)$ then in step $num(y^+)$ we obtain $r(y) = 0$. Therefore in step $num(u^*)$ we already have a neighbor of u^* which violates the condition in line (8) of the algorithm. Hence $num(y^+) > num(u^*)$; i.e., the vertex $x = mn(u^*)$ is adjacent to y^+. Then $r(x) = 0$ in step $num(y^+)$ and again we can apply the condition in line (8) in order to obtain a contradiction with $C'' = fn(v) = \cdots = fn(y)$.

Therefore we obtain that if $num(u^*) < num(v^*)$ then u^* coincides with x or u. Consider the first case, i.e., $u^* = x$. Then $v^* \neq y$. Before step $num(x)$ we have $r(x) = 0$; after this step we obtain $a(v^*) = 1$. Then in step $num(v^*)$ the condition in line (8) of the algorithm is violated except the case when $v^* = v$ and $r(C'') \leq 1$. By the choice of cliques C' and C''

$$d(u,v') \leq r(C') + r(C'') \leq r(C') + 1$$

holds for every vertex $v' \in C''$. Because of $num(x) = num(u^*) < num(v^*) = num(v) < num(v')$ we can apply Lemma 5.8 to path R' and every vertex $v' \in C''$. If $r(C'') = 0$ then necessarily $C'' = \{v\}$ and by this lemma we get that v is adjacent to some vertex $z \neq x$ of R'. Then in step $num(z)$ we have a neighbor v of z with $r(v) = 0$ and we can apply the condition in line (8) in order to obtain a contradiction with $C' = fn(u) = \cdots = fn(z) = \cdots = fn(x)$. So suppose that $r(C'') = 1$. If x is adjacent to all vertices of C'' then in step $num(x)$ we obtain $a(C'') = 1$. Then in step $num(v)$ we have $a(C'') = r(C'')$ and we cannot include C'' in P. So there is a vertex $v' \in C''$ nonadjacent with x. By Lemma 5.8 the whole path R' belongs to the maximum neighbor path between u and v'. Then $d(u,v') \geq d(u,x) + 2 > r(C') + r(C'')$ in contradiction with the choice of the cliques C' and C''.

Finally consider the case when $u^* = u$. Then $v^* \neq v$ and $u^* \neq x$; otherwise, the conditions of the preceding cases are fulfilled. In particular we obtain that $r(C') > 0$.

Since $num(u^*) < num(v^*)$ the vertex v^* must be adjacent to the vertex $mn(u^*)$. By the choice of vertices u^* and v^* and our conditions we get $v^* = y$ and the vertices $u^* = u$ and x are adjacent. Thus $r(C') = 1$. Let z be the neighbor of v^* in the subpath of R'' connecting the vertices v and v^*. If $num(u^*) < num(z)$, then from $v^* = mn(z)$ it follows that $v^* = mn(u^*)$. Then we get a contradiction with the disjointness of the paths R' and R''. So suppose that $num(u^*) > num(z)$. If y is adjacent to all vertices of the clique C' then in step $num(u)$ we have $r(y) = 0$ and $C' \subseteq N[y]$. This leads to a contradiction with the fact that $C' = fn(x)$ is included in P. So suppose that y is nonadjacent to some vertex $u' \in C'$. Because of

$$num(u') > num(u) > num(z) > \cdots > num(v)$$

we can apply Lemma 5.8 to the vertex u' and the path R''. Then we obtain that

$$d(v, u') \geq d(v, y) + d(y, u') \geq r(C'') + 2 > r(C'') + r(C')$$

in contradiction with the choice of the cliques C' and C''.

Case 2. $u^+ \in R'$ and $v^+ \notin R''$ (the case when $u^+ \notin R'$ and $v^+ \in R''$ is similar).

Since the paths R'' and R_v are comparable, the inclusion $R'' \subseteq R_v$ holds; i.e., the vertex y belongs to the path R_v. Let z be the neighbor of v^+ in R_v. Then $v^+ = mn(z)$ and z belongs to the subpath of R_v between v^+ and y. Let $d(u^+, x) = l'$ and $d(v^+, y) = l''$. Since $d(u, v) \leq r(C') + r(C'')$ and $d(u, v) = r(C') - l' + 1 + l'' + r(C'')$ holds, the inequality $l'' < l'$ necessarily is fulfilled. Moreover since $v^+ \neq y$ and $2 + r(C'') \leq d(u, v) \leq r(C') + r(C'')$ holds we have $r(C') \geq 2$.

If $num(u^+) < num(z)$ then the vertex $mn(u^+) \in R'$ must be adjacent to both v^+ and z. Then since u^+ and z are adjacent to both vertices $mn(u^+)$ and $v^+ = mn(z)$ by the MNO algorithm we conclude that $v^+ = mn(u^+)$. By the CRDP algorithm in step $num(v^+)$ we have $a(v^+) = l''$ and $r(v^+) = l' - 1$. Since $l'' < l'$ the inequality $a(v^+) \leq r(v^+)$ holds. Comparing with the conditions in lines (8) and (9) we get that $x = v^+$. But then

$$d(u, v) \geq r(C') + d(x, y) + r(C'') > r(C') + r(C''),$$

which leads to a contradiction.

Now assume that $num(u^+) > num(z)$. If $num(u^+) < num(v^+)$ then in step $num(u^+)$ for vertex v^+ we have $a(v^+) = l'' < l' = r(u^+)$. Therefore in step $num(u^+)$ $a(v^+) + 1 \leq r(u^+)$ and if $u \neq u^+$ we cannot transmit the value $fn(u^+) = C'$ to the maximum neighbor of u^+. Otherwise if $num(u^+) > num(v^+)$ and $u^+ \neq u$ then before step $num(u^+)$ we already have $a(u^+) \leq l'' + 1$. Then $a(u^+) \leq r(u^+)$ in step $num(u^+)$ and again we can apply the condition in line (8) in order to obtain a contradiction with $C' = fn(u) = \cdots = fn(u^+) = \cdots = fn(x)$.

Finally suppose that $u^+ = u$ and $num(u) = num(u^+) > num(z)$. First assume that $l'' + 1 = l'$. We claim that $C' \subseteq N[v^+]$. Assume the contrary and let u' be a vertex of C' nonadjacent with v^+. Since

$$num(u') > num(u) > num(z) > \cdots > num(v),$$

by applying Lemma 5.8 to the path R_v and the vertex u' we get

$$d(u', v) = d(v, v^+) + d(v^+, u') \geq r(C'') + l'' + 2 > r(C'') + r(C'),$$

which is a contradiction. So $C' \subseteq N[v^+]$. Then in step $num(u)$ we have $a(C') \leq l'' + 1 = l' = r(C')$ if $num(u) > num(v^+)$ and $a(v^+) + 1 \leq l'' + 1 = l' = r(C')$

and $C' \subseteq N[v^+]$ if $num(u) < num(v^+)$. In both cases we get a contradiction with $C' = fn(u) = \cdots = fn(x)$. So let $l''+1 < l'$. Then in step $num(u)$ we have $1+a(u) \leq l''+2 \leq l' = r(C'')$ if $num(u) > num(v^+)$ and $1 + a(v^+) = l'' + 1 < l' = r(C')$ if $num(u) < num(v^+)$. We obtain the same contradiction as in the preceding case. This concludes the proof of the lemma. □

From Lemmas 5.3–5.9 and the duality between the clique r-domination and clique r-packing problems we immediately obtain that the sets $D \subseteq V$ and $P \subseteq \mathcal{C}$ computed by the algorithm CRDP are minimum clique r-dominating and maximum clique r-packing sets. □

6. Time bounds for special cases. In this section we consider the time bounds for some important special input cases of the CRDP algorithm. The obtained results are collected in the following.

THEOREM 6.1. *Let $\mathcal{C} = \{C_1, \ldots, C_p\}$ be a family of cliques of a dually chordal graph $G = (V, E)$ and $r : \mathcal{C} \to N \cup \{0\}$ be the radius function on $\mathcal{C}$. Then the clique r-domination and clique r-packing problem for $\mathcal{C}$ can be solved in time*

(1) $O(|E| + |V| \sum_{i=1}^{p} |C_i|)$ *if $\mathcal{C}$ is an arbitrary family of cliques,*
(2) $O(|E| + \sum_{i=1}^{p} |C_i|)$ *if $\mathcal{C}$ is a family of maximal cliques,*
(3) $O(|E|)$ *if $\mathcal{C}$ is a family of one-vertex cliques,*
(4) $O(|V||E|)$ *if $\mathcal{C}$ is a family of edges,*
(5) $O(|E|)$ *if G is doubly chordal and $\mathcal{C}$ is a family of maximal cliques,*
(6) $O(|E|)$ *if G is doubly chordal without induced sun S_3 and $\mathcal{C} = E$ and $r(e) \equiv k$ for all $e \in E$.*

The running time of the algorithm for an arbitrary family $\mathcal{C} = \{C_1, \ldots, C_p\}$ of cliques of a dually chordal graph $G = (V, E)$ can be estimated as follows. For a vertex $v_i \in V$ let s_i be the degree of v_i and k_i be the number of cliques containing v_i (i.e., $k_i = |\mathcal{C}(v_i)|$) and l_i be the number of cliques of $\mathcal{C}$ which are dominated by v_i, i.e.,

$$l_i = |\{C \in \mathcal{C} : C \subseteq N[v_i]\}|.$$

Evidently lines (2) and (3) of the algorithm use $O(|\mathcal{C}| + |V|)$ operations, while line (4) takes $O(|V| + \sum_{i=1}^{p} |C_i|)$ time. The overall time bound of lines (5)–(8), (10)–(13), and (15) is $O(|E|)$. In order to implement lines (9) and (14) we compute in advance in $O(\sum_{i=1}^{|V|} k_i s_i + \sum_{i=1}^{|V|} l_i)$ time the 0-1-matrix $M = (m_{ij})_{j=1,\ldots,|\mathcal{C}|}^{i=1,\ldots,|V|}$, where $m_{ij} = 1$ if and only if $C_j \subseteq N[v_i]$. Using this matrix, lines (9) and (14) can be executed in time $O(\sum_{i=1}^{|V|} k_i s_i)$ and $O(\sum_{i=1}^{|V|} l_i)$, respectively. So the total time of the algorithm is

$$O\left(|E| + \sum_{i=1}^{|V|} k_i s_i + \sum_{i=1}^{|V|} l_i\right),$$

which in the worst case can be estimated as $O(|E| + |V| \sum_{i=1}^{|\mathcal{C}|} |C_i|)$ because $\sum_{i=1}^{|V|} k_i = \sum_{i=1}^{|\mathcal{C}|} |C_i|$ and $\sum_{i=1}^{|V|} l_i \leq |V||\mathcal{C}|$.

Now we consider the complexity of the algorithm CRDP in some important particular cases. First suppose that $\mathcal{C}$ consists only of all one-vertex cliques. Then we obtain the r-domination and r-packing problems. Since

$$\sum_{i=1}^{|\mathcal{C}|} |C_i| = |\mathcal{C}| \leq |V|, \quad \sum_{i=1}^{|V|} l_i = \sum_{i=1}^{|V|} s_i = 2|E|$$

and line (9) is omitted the total time complexity for these problems is $O(|E|)$.

Next let $\mathcal{C}$ be a family of maximal cliques of G. Then the time complexity of all lines is the same as in the general case except lines (9) and (14). Line (14) takes only $O(|E|)$ time because the condition $C \subseteq N_i(v_i)$ is fulfilled only for the additional one-vertex cliques and not for any maximal clique of G. Line (9) can be implemented directly without computing the matrix M. For this purpose recall that it is enough to verify the condition in this line only for cliques $C \subseteq N_i[v_i]$ such that v_i has minimal index in C and $r(C) = r(v_i)$; see the proof of Lemma 5.5. Using this fact first we can select all vertices $v \in N_i[v_i]$ such that either $a(v) + 1 = r(v_i)$ or $r(v) = 0$. This can be done in time $O(s_i)$. After that for each clique $C \in \mathcal{C}(v_i)$ such that $r(C) = r(v_i)$ and v_i has minimal index in C we decide whether $C \not\subseteq N[v]$ for some selected vertex v. This operation can be implemented in time $O(k_i + \sum_{C_i \in \mathcal{C}(v_i)} |C_i|)$. So for each $i, i \in \{1, \ldots, |V|\}$ the time complexity of line (9) is $O(k_i + l_i + \sum_{C_i \in \mathcal{C}(v_i)} |C_i|)$. Since each clique is considered only once in step $\min\{num(v) : v \in C\}$ we conclude that the overall time amount of line (9) and of the whole algorithm is $O(|E| + \sum_{i=1}^{|\mathcal{C}|} |C_i|)$. Assume that a dually chordal graph $G = (V, E)$ is also chordal; i.e., G is a doubly chordal graph. It is known that for chordal graphs $\sum_{i=1}^{|\mathcal{C}|} |C_i|$ does not exceed $O(|E|)$. So for doubly chordal graphs the algorithm requires only $O(|E|)$ time. Since strongly chordal graphs are doubly chordal [5] this result improves and generalizes the algorithm for the clique transversal and clique independence problem presented in [8].

Finally consider the case when $\mathcal{C}$ is a subset of edges of a dually chordal graph $G = (V, E)$. Since $\sum_{i=1}^{|\mathcal{C}|} |C_i| \leq 2|E|$ the time complexity of the algorithm is $O(|V||E|)$.

Next assume that $\mathcal{C} = E$ and $r(e) \equiv k$ for every edge $e \in E$, where k is a positive integer. As we already mentioned these problems are known as the k-neighborhood covering and k-neighborhood independence problems [17]. In [17] a linear time algorithm for solving these problems in strongly chordal graphs is presented under the assumption that a strong elimination ordering of such graphs is given. Unfortunately the fastest known algorithms for deriving such orderings have time complexity $O(|V|^2)$ [23] or $O(|E| \log |V|)$ [21]. Our algorithm can be modified in order to solve the k-neighborhood covering and k-neighborhood independence problems on strongly chordal graphs. In fact the approach presented below solves these problems on a more general subclass of dually chordal graphs, namely on doubly chordal graphs containing no induced sun S_3 (see Figure 1). This is mainly due to the next result.

Denote by $\gamma_{E,k}$ and $\pi_{E,k}$ the k-neighborhood covering and the k-neighborhood independence numbers of G.

LEMMA 6.2. *Let $G = (V, E)$ be a doubly chordal graph containing no induced sun S_3. Then*

$$\gamma_{E,k}(G) = \pi_{E,k}(G) = \pi_{\mathcal{C},k}(G) = \gamma_{\mathcal{C},k}(G),$$

where $\mathcal{C}$ is the family of all maximal cliques of G and $r(C) \equiv k$ for all $C \in \mathcal{C}$.

Proof. By Lemma 4.2 we have

$$\gamma_{E,k}(G) = \pi_{E,k}(G),\ \pi_{\mathcal{C},k}(G) = \gamma_{\mathcal{C},k}(G).$$

Moreover for arbitrary graphs $\pi_{E,k}(G) \leq \pi_{\mathcal{C},k}(G)$ holds. In order to show this we extend each edge of the k-neighborhood independent set to a maximal clique of G. The obtained family of cliques represents a clique k-packing set of $\mathcal{C}$. So it is enough to show the converse inequality. Let P be a maximal clique k-packing set of G. By

[18, Proposition 3] every maximal clique of a chordal graph G without induced sun S_3 has an edge which is not contained in any other maximal clique of G. Include in the set I all such representative edges of cliques from P. We claim that I is a k-neighborhood independent set of G. Let $e' = u'v'$ and $e'' = u''v''$ be edges which represent cliques $C', C'' \in P$. Assume that there exists a vertex w such that $\delta(w, e') \leq k$ and $\delta(w, e'') \leq k$. We will show that $\delta(w, C') \leq k$ and $\delta(w, C'') \leq k$. Assume the contrary and let $\delta(w, C') > k$. This means that $d(w, v) > k$ for some vertex $v \in C'$. Then necessarily $d(w, v') = d(w, u') = k$. By Lemma 3.4 there is a common neighbor w^* of vertices v' and u' which is at distance $k - 1$ from w. Since the edge e' is contained in the unique clique C' we have $w^* \in C'$ and thus the vertices w^* and v must be adjacent. This contradicts the assumption that $d(w, v) > k$. □

Thus in order to solve the k-neighborhood covering and k-neighborhood independence problems on a doubly chordal graph G without induced sun S_3 we can apply the algorithm CRDP for the family of all maximal cliques $\mathcal{C}$ of G when $r(C) \equiv k$ for all $C \in \mathcal{C}$. As we already mentioned for doubly chordal graphs this problem can be solved in time $O(|E|)$. Let D and P be the output of the algorithm. Evidently D is a minimum k-neighborhood covering set too while the set I defined in the proof of Lemma 6.2 is a maximum k-neighborhood independent set. Hence it remains only to efficiently compute such a set I in $O(|E|)$ time under the assumption that the set P is given.

In order to do this we use the perfect elimination ordering of a chordal graph G and obtain in $O(|E|)$ time all representative edges for the family of maximal cliques; see the procedure presented below. In each clique of the set P we select such an edge and obtain the required set I.

Let $G = (V, E)$ be a chordal graph and $v_1, \ldots, v_n$ be a perfect elimination ordering of G. As before $num(v)$ is the index of a vertex v in this ordering.

PROCEDURE 6.3 (**representative edges**).

$E^* := \emptyset$;
for all $i \in \{1, \ldots, n\}$ **do** $A_i := N_i(v_i)$;
for $i := 1$ **to** $n - 1$ **do**
  if A_i *is not empty* **then**
    begin
      $v :=$ *arbitrary vertex from* A_i;
      $E^* := E^* \cup v_i v$;
      for all $v_j \in N_i(v_i)$ **do** $A_j := A_j \setminus N_i(v_i)$;
    end

LEMMA 6.4. *Let $G = (V, E)$ be a chordal graph without induced sun S_3. Then the presented procedure correctly finds within $O(|E|)$ steps a set of representative edges for the family of all maximal cliques of G.*

Proof. The correctness proof is based on two claims.

(a) Each maximal clique of G contains an edge selected by the procedure.

(b) The family of selected edges E^* consists only of representative edges.

In order to prove (a) let C be an arbitrary maximal clique. By [18, Proposition 3], C has a representative edge $v_i v_j, i < j$. If this or any other edge of C is not selected by the procedure then in step i we must have $A_i = \emptyset$. This means in particular that in some step $t < i$ the vertex v_j must be deleted from A_i. By the procedure in this step some edge $v_t v_l$ must be included into E^* where $v_l \in N_t(v_t)$. Since the edge $v_i v_j$ is representative for the clique C and v_t is adjacent to both v_i and v_j, certainly v_t is a vertex of C. Moreover, since v_t is simplicial in $G(\{v_t, v_{t+1}, \ldots, v_n\})$ the vertex v_l is

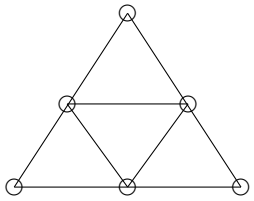

FIG. 1.

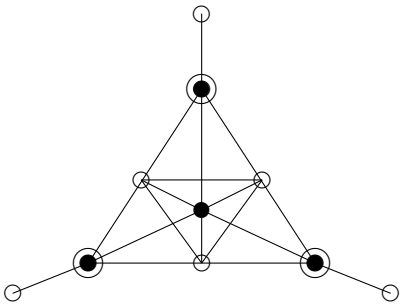

FIG. 2.

adjacent to v_i and v_j. Using the same argument for the edge v_iv_j, we conclude that $v_l \in C$ and thus the edge $v_tv_l \in E^*$ is contained in C.

Next suppose that in E^* there is an edge v_iv_j, $i < j$ which belongs to two maximal cliques C' and C''. Let v_t be the vertex with minimal index in $C' \cup C''$. Let, for example, $v_t \in C'$. Since both cliques C' and C'' are completely contained in $G(\{v_t, \ldots, v_n\})$ and v_t is a simplicial vertex of this graph, we obtain that $v_t \notin C''$. Then v_t is not adjacent to some vertex $v_l \in C''$. By the procedure the edge v_iv_j is included in E^* in step i. Therefore, before this step we have $v_j \in A_i$. In particular, v_j remains in A_i after step $t < i$. This is possible only if $A_t = \emptyset$. Since initially both vertices v_i and v_j belong to A_t they must be deleted from this set in some steps k' and k''. Necessarily $k' \neq k''$; otherwise we must delete the vertex v_j from A_i in step k', which is impossible. But in this case the vertices $v'_k, v_t, v_i, v_j, v''_k, v_l$ induce a sun S_3. This is the case because G is chordal and there are no edges between the vertex pairs (v_l, v_t), (v'_k, v_j) and (v''_k, v_i).

Time bound. Let $\sigma = (v_1, \ldots, v_n)$ again be the perfect elimination ordering of G. For every $i \in \{1, \ldots, n\}$ determine the position $c(v_i)$ of the leftmost neighbor of v_i in σ. Obviously this can be done in $2|E|$ steps.

Rearrange the vertices of V in increasing order with respect to the parameter $c(v_i)$, $i \in \{1, \ldots, n\}$. This can be done using bucket sort in $O(|E|)$ time, obtaining the ordering $\tau = (v_{j_1}, \ldots, v_{j_n})$. Note that the nonempty elements of the family $L_k = \{v_i : c(v_i) = k\}$, $k \in \{1, \ldots, n\}$, represent a clique partition of the chordal graph G along σ.

Using the ordering τ and the standard technique of how to get an ordered adjacency list from a nonordered adjacency list of G (see, e.g., [16]) we get an ordered representation of A_i as linearly linked list $L^i_{j_1}, \ldots, L^i_{j_i}$, ordered with respect to τ, of linearly linked lists $L^i_{j_k} \subseteq L_{j_k}$, $k \in \{1, \ldots, i\}$, called *segments* subsequently.

Then, having this structure for every A_i, the operation $A_j := A_j \setminus N_i(v_i)$ can be performed in the following way: due to claim (a) each maximal clique contains a representative edge, and hence we have to delete only the leftmost segment (corresponding to $c(v_i)$) from the current list A_j.

Since for fixed j this can be done in constant time the whole procedure takes linear time. □

Unfortunately the equality in Lemma 6.2 does not hold for all dually chordal graphs. Figure 2 provides an example of a doubly chordal graph G with $\gamma_{\mathcal{C},1}(G) = 4$ and $\gamma_{E,1}(G) = 3$.

7. Conclusions. In this paper we presented a unified approach to solve different types of clique r-domination and clique r-packing problems on dually chordal graphs. For three particular cases of these problems, namely for r-domination and r-packing, clique transversal and clique independence, and k-neighborhood covering and k-neighborhood independence we obtain linear time algorithms. The corresponding results generalize and improve results of [7, 8, 17] for the same problems on strongly chordal graphs.

Concerning the clique transversal and clique independence problems, the complexity of our algorithm is proportional to the sum of the sizes of all maximal cliques of a dually chordal graph. Unlike chordal graphs where the number of maximal cliques does not exceed the number of vertices, dually chordal graphs may have an exponential number of maximal cliques. The reason for this is that an arbitrary graph G can be transformed into a dually chordal graph by adding a new vertex adjacent to all vertices of G. Moreover, it is not known whether the clique transversal problem is in NP.

REFERENCES

[1] I. Bárány, J. Edmonds, and L. A. Wolsey, *Packing and covering a tree by subtrees*, Combinatorica, 6 (1986), pp. 221–233.

[2] H. Behrendt and A. Brandstädt, *Domination and the Use of Maximum Neighbourhoods*, Technical Report SM-DU-204, University of Duisburg, 1992.

[3] C. Berge, *Hypergraphs*, North–Holland, Amsterdam, The Netherlands, 1989.

[4] A. Brandstädt, V. D. Chepoi, and F. F. Dragan, *The Algorithmic Use of Hypertree Structure and Maximum Neighbourhood Orderings*, Technical Report SM-DU-244, University of Duisburg, 1994; in Graph-Theoretic Concepts in Computer Science, Lecture Notes in Computer Science 903, E. W. Mayr, G. Schmidt, and G. Tinhofer, eds., Springer-Verlag, Berlin, New York, 1995, pp. 65–80.

[5] A. Brandstädt, F. F. Dragan, V. D. Chepoi, and V. I. Voloshin, *Dually Chordal Graphs*, Technical Report SM-DU-225, University of Duisburg 1993; Graph-Theoretic Concepts in Computer Science, Lecture Notes in Computer Science 790, J. van Leeuwen, ed., Springer-Verlag, Berlin, New York, 1994, pp. 237–251.

[6] P. Buneman, *A characterization of rigid circuit graphs*, Discrete Math., 9 (1974), pp. 205–212.

[7] G. J. Chang, *Labeling algorithms for domination problems in sun–free chordal graphs*, Discrete Appl. Math., 22 (1988/89), pp. 21–34.

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

[9] G. J. Chang and G. L. Nemhauser, *The k-domination and k-stability problems on sun-free chordal graphs*, SIAM J. Alg. Disc. Meth., 5 (1984), pp. 332–345.

[10] G. A. Dirac, *On rigid circuit graphs*, Abh. Math. Sem. Univ. Hamburg, 25 (1961), pp. 71–76.

[11] F. F. Dragan, *Domination and packings in triangulated graphs*, Metody Diskret. Analiz., 51 (1991), pp. 17–36. (In Russian.)

[12] F. F. Dragan, C. F. Prisacaru, and V. D. Chepoi, *The location problem on graphs and the Helly problem*, Disk. Math., 4 (1992), pp. 67–73. (In Russian.) (The full version appeared as preprint: F.F. Dragan, C.F. Prisacaru, and V.D. Chepoi, r-Domination and p-Center Problems on Graphs: Special Solution Methods and Graphs for Which This Method is Usable, Kishinev State University, preprint MoldNIINTI, N. 948–M88, 1987 (in Russian).)

[13] M. Farber, *Characterizations of strongly chordal graphs*, Discrete Math., 43 (1983), pp. 173–189.

[14] M. Farber, *Domination, independent domination and duality in strongly chordal graphs*, Discrete Appl. Math., 7 (1984), pp. 115–130.

[15] D. R. Fulkerson and O. R. Gross, *Incidence matrices and interval graphs*, Pacific J. Math., 15 (1965), pp. 835–855.

[16] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

[17] S. H. Hwang and G. J. Chang, *k-Neighbourhood Covering and Independence Problems*, DIMACS Report 93-03, DIMACS Center, Rutgers University, New Brunswick, NJ, 1993.

[18] J. Lehel and Z. Tuza, *Neighbourhood perfect graphs*, Discrete Math., 61 (1986), pp. 93–101.
[19] A. Lubiw, *Doubly lexical orderings of matrices*, SIAM J. Comput., 16 (1987), pp. 854–879.
[20] M. Moscarini, *Doubly chordal graphs, Steiner trees and connected domination*, Networks, 23 (1993), pp. 59–69.
[21] R. Paige and R.E. Tarjan, *Three partition refinement algorithms*, SIAM J. Comput. 16 (1987), pp. 973–989.
[22] P. J. Slater, *R-domination in graphs*, J. Assoc. Comput. Mach., 23 (1976), pp. 446–450.
[23] J. P. Spinrad, *Doubly lexical ordering of dense* 0–1*–matrices*, Inform. Process. Lett., 45 (1993), pp. 229–235.
[24] J. L. Szwarcfiter and C. F. Bornstein, *Clique graphs of chordal and path graphs*, SIAM J. Discrete Math., 7 (1994), pp. 331–336.
[25] R. E. Tarjan and M. Yannakakis, *Simple linear time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs*, SIAM J. Comput., 13 (1984), pp. 566–579.