A Linear-Time Algorithm for Finding a Central Vertex of a Chordal Graph *

Victor Chepoi and Feodor Dragan

Department of Mathematics & Cybernetics,
Moldova State University, A. Mateevici str., 60,
Chişinău 277009, Moldova

Abstract. In a graph $G = (V, E)$, the eccentricity $e(v)$ of a vertex v is $\max\{d(v, u) : u \in V\}$. The center of a graph is the set of vertices with minimum eccentricity. A graph G is chordal if every cycle of length at least four has a chord. We present an algorithm which computes in linear time a central vertex of a chordal graph. The algorithm uses the metric properties of chordal graphs and Tarjan and Yannakakis linear–time test for graph chordality.

1 Introduction

All graphs in this paper are connected and simple, i.e. finite, undirected, loopless and without multiple edges. In a graph $G = (V, E)$ the *length* of a path from a vertex v to a vertex u is the number of edges in the path. The *distance* $d(u, v)$ from vertex u to vertex v is the length of a minimum length path from u to v and the *interval* $I(u, v)$ between these vertices is the set

$$I(u, v) = \{w \in V : d(u, v) = d(u, w) + d(w, v)\}.$$

The *eccentricity* $e(v)$ of a vertex v is the maximum distance from v to any vertex in G. Denote by $D(v)$ the set of all *farthest from* v vertices, i.e. $D(v) = \{w \in V : d(v, w) = e(v)\}$. The *radius* $r(G)$ is the minimum eccentricity of a vertex in G and the *diameter* $d(G)$ the maximum eccentricity. The *center* $C(G)$ of G is the subgraph induced by the set of all *central vertices* , i.e. vertices whose eccentricities are equal to $r(G)$. A *clique* of G is a set of pairwise adjacent vertices. A vertex v of G is called *simplicial* if its neighborhood is a clique.

A graph G is *chordal* (*triangulated*) if every cycle of length greater than three possesses a chord, i.e. an edge joining two nonconsecutive vertices on the cycle. Chordal graphs arise in the study of Gaussian elimination of sparse symmetric

*This work was partially supported by the VW–Stiftung Project No. I/69041
e-mail addresses: chepoi@university.moldova.su, dragan@university.moldova.su

matrices. They were first introduced by Hajnal & Suranyi [10] and then studied extensively by many people, see [6,9] for general results. Recently, the metric properties of chordal graphs were also investigated, see [1–7,13,15]. In particular, the inequality $2r(G) \geq d(G) \geq 2r(G) - 2$ was proven [1,3,4] and the centers of chordal graphs were characterized [4]. The class of chordal graphs contains trees, block graphs, maximal outerplanar graphs, k–trees, interval graphs and strongly chordal graphs.

In this paper we will present a linear time algorithm for finding a central vertex of a chordal graph. Note that for general graphs with n vertices and m edges the upper bound on the time complexity of this problem is $O(nm)$ and the lower bound is $\Omega(m)$. Hence the presented algorithm is optimal. Linear time algorithms for finding the central vertices are known for trees [11,12], 2–trees and maximal outerplanar graphs [8], strongly chordal graphs [5] and interval graphs [14].

The key idea of our algorithm is that with a few applications of breadth first search, it is possible to find two vertices y, z such that the distance $d(y, z)$ is at most two less than the diameter of the chordal graph G. Intuitively, it makes sense to look for a central vertex in the vicinity of the middle of shortest y, z–paths. We either find a central vertex of G in this set or we replace the pair y, z by a new pair of vertices at distance one step closer to the diameter of G.

2 Metric Properties of Chordal Graphs

In this section we present the metric properties of chordal graphs, used in our algorithms.

Lemma 1 [1] *For any two vertices x and y of a chordal graph G and integer $k \leq d(x, y)$ the set $L(x, k, y) = \{z \in I(x, y) : d(x, z) = k\}$ is a clique.*

Lemma 2 [1] *In a chordal graph G, if C is a clique and x is a vertex not in C such that $d(x, y) = k$ is a constant for all $y \in C$ then there exists a vertex $x^* \in \cap\{I(x, y) : y \in C\}$ adjacent to all vertices from C.*

Such a vertex x^* we will call a *gate* of a vertex x in C.

Lemma 3 [3] *If $x, y, u, v \in V$ are distinct vertices of a chordal graph G, such that $x \in I(u, y)$,*
$y \in I(x, v)$ and $d(x, y) = 1$ then $d(u, v) \geq d(u, x) + d(y, v)$. The equality holds if and only if there is a vertex $w \in I(x, v) \cap I(u, y)$ adjacent to x and y.

For any subset $M \subset V$ and any vertex v we denote by

$$Pr(v, M) = \{y \in M : d(v, y) = d(v, M)\}$$

the *metric projection* of v on M (recall that $d(v, M) = \min\{d(v, z) : z \in M\}$).

Lemma 4 *In a chordal graph G, for any clique C and any adjacent vertices $u, v \notin C$ the metric projections $Pr(u, C)$ and $Pr(v, C)$ are comparable, i.e. either $Pr(u, C) \subseteq Pr(v, C)$ or $Pr(v, C) \subseteq Pr(u, C)$.*

Proof Since vertices u and v are adjacent $\mid d(u,C)-d(v,C)\mid\leq 1$. Without loss of generality assume that $d(u,C)\geq d(v,C)$. If $d(u,C)=d(v,C)+1$ then for any vertex $w\in Pr(v,C)$ we have $d(u,w)\leq d(u,C)$ and therefore $w\in Pr(u,C)$, i.e. $Pr(v,C)\subseteq Pr(u,C)$. Now assume that $d(u,C)=d(v,C)$, but the sets $Pr(u,C)$ and $Pr(v,C)$ are incomparable. Then we find the vertices

$$x\in Pr(v,C)\backslash Pr(u,C), y\in Pr(u,C)\backslash Pr(v,C).$$

Since $x\in I(v,y)$ and $y\in I(u,x)$, by Lemma 3 we have

$$d(u,v)\geq d(u,y)+d(x,v)=2d(u,C)\geq 2,$$

thus yielding a contradiction. □

Lemma 5 *For any graph G if $d(x,y)=d(G)=2r(G)$ then* $C(G)\subset L(x,r(G),y)$.

Lemma 6 *For any vertex v of a chordal graph G and any vertex u that is farthest from v we have $e(u)\geq 2r(G)-3$.*

Proof Assume the contrary and among the vertices which fail our assertion choose a vertex v with minimal eccentricity. Let $u\in D(v)$ be a vertex for which $e(u)<2r(G)-3$. From our assumption we deduce that for any vertex $x\in L(v,1,u)$ we have $u\notin D(x)$, i.e. $e(x)\geq e(v)$. If $e(x)>e(v)$ for some vertex $x\in L(v,1,u)$ then $v\in I(x,y)$ for any vertex $y\in D(x)$. By Lemma 3

$$d(u,y)\geq e(v)-1+e(x)-1\geq 2e(v)-1\geq 2r(G)-1.$$

Hence all vertices from $L(v,1,u)$ have the same eccentricity $e(v)$. Now, if for some vertex $x\in L(v,1,u)$ there is a vertex $z\in D(x)\backslash D(v)$, then $v\in I(z,x)$ and by Lemma 3

$$d(u,z)\geq e(v)-1+e(x)-1\geq 2r(G)-2.$$

Finally assume that $\cup\{D(x):x\in L(v,1,y)\}\subset D(v)$. Let x^* be a vertex from $L(v,1,u)$ having a minimum number of farthest vertices and let $y^*\in D(x^*)$. Since $d(v,y^*)=d(x^*,y^*)$ by Lemma 2 there is a vertex $x\in I(v,y^*)\cap I(x^*,y^*)$ adjacent to v and x^*. If $x\notin I(v,u)$ then $x^*\in I(x,u)$ and again, by Lemma 3 $d(u,y^*)\geq 2e(v)-2\geq 2r(G)-2$. Therefore $x\in I(v,u)$. From our choice of the vertex x^* there exists a vertex $y\in D(x)\backslash D(x^*)$. Since the vertices x and x^* are adjacent and equidistant from u, according to Lemma 2 there is a vertex $w\in I(x,u)\cap I(x^*,u)$ adjacent to x and x^*. Since $x^*\in I(x,y)$ and $x\in I(x^*,y^*)$ $d(w,y^*)\geq e(v)-1$ and $d(w,y)\geq e(v)-1$.

If $x\in I(w,y^*)$ or $x^*\in I(w,y)$ then by Lemma 3 at least one of the following inequalities holds

$$d(u,y^*)\geq d(x,y^*)+d(w,u)=2e(v)-3\geq 2r(G)-3,$$

$$d(u,y)\geq d(x^*,y)+d(w,u)=2e(v)-3\geq 2r(G)-3.$$

So $d(w,y^*)=d(x,y^*)$ and $d(w,y)=d(x^*,y)$. Again, by Lemma 2 there exist vertices $s\in I(x,y^*)\cap I(w,y^*)$ and $t\in I(x^*,y)\cap I(w,y)$ adjacent to x,w

and x^*, w correspondingly. Note that $w \in I(s,u) \cup I(t,u)$, otherwise $s,t,w \in L(v,2,u)$ and by Lemma 1 the vertices s and t must be adjacent. The obtained cycle (t,x^*,x,s,t) has one of the chords (x,t) or (x^*,s), thus violating that $x \in I(x^*,y^*)$ and $x^* \in I(x,y)$. Without loss of generality assume that $w \in I(t,u)$. By Lemma 3 $d(u,y) \geq d(u,w) + d(t,y) \geq 2r(G) - 4$ and equality holds if and only if there exists a vertex $z' \in I(w,y) \cap I(t,u)$ adjacent to w and t. We distinguish two cases.

Case 1 $d(s,u) = d(w,u)$.

By Lemma 2 there is a common neighbour $u'' \in I(s,u) \cap I(w,u)$ of vertices s and w; see Fig. 1. Since vertices w and z' are equidistant from u they have a common neighbour $u' \in I(w,u) \cap I(z',u)$. Since $u', u'' \in I(w,u)$ the vertices u' and u'' are adjacent or coincide. In any case we obtain the cycle $(x^*,x,s,u'',u',z',t,x^*)$; see Fig. 1. From the previous conditions on vertices v,x,x^*,y,y^*,u we deduce that vertices x and x^* do not have a common neighbour among the vertices of this cycle. Therefore there is no triangulation of this cycle in which the edge (x,x^*) belongs to some triangle, contradicting the chordality of the graph G.

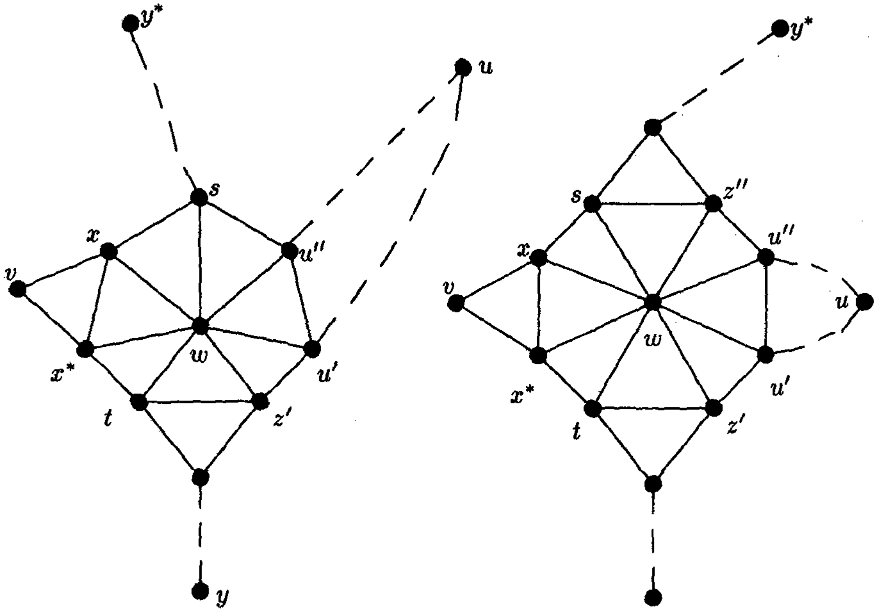

Fig. 1 **Fig. 2**

Case 2 $d(s,u) = d(w,u) + 1$.

By Lemma 3 $d(u,y^*) \geq d(y^*,s) + d(w,u) \geq 2r(G) - 4$ and equality holds only if there is a vertex $z'' \in I(w,y^*) \cap I(s,u)$ adjacent to w and s; see Fig.

2. As in the first case, since $d(z', u) = d(w, u) = d(z'', u)$, there exist vertices $u' \in I(w, u) \cap I(z', u)$ and $u'' \in I(w, u) \cap I(z'', u)$ adjacent to z', w and z, w respectively. Then u' and u'' either are adjacent or coincide. In any case we obtain the cycle $(x^*, x, s, z'', u'', u', z', t, x^*)$. As in the first case there is no common neighbour of x and x^* among the vertices of this cycle, yielding a contradiction.

So the assumption that for some vertex $u \in D(v)$ we have $e(u) < 2r(G) - 3$ leads to a contradiction. □

The following example (see Fig. 3) show the sharpness of the inequality $e(u) \geq 2r(G) - 3$.

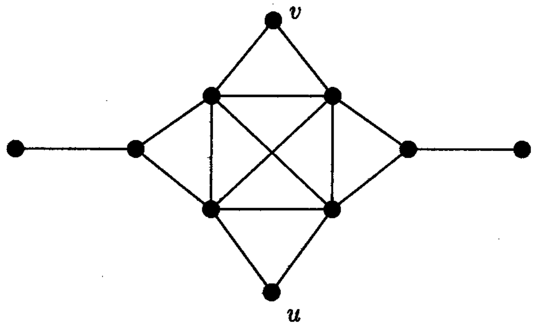

Fig. 3 ($u \in D(v)$ and $e(u) = 3 = 2r(G) - 3$)

Corollary *For any vertex v of a chordal graph G and any vertex u that is farthest from v we have $e(u) \geq d(G) - 2$.*

Proof By Lemma 6 it is enough to consider only the case when $d(G) = 2r(G)$ and $v \in C(G)$. Then according to Lemma 5 $v \in L(x, r(G), y)$ for any vertices $x, y \in V$ such that $d(x, y) = d(G)$. For any $u \in D(v)$ let t be a vertex from $I(v, u)$ adjacent to v. As in the proof of Lemma 6 we can show that vertices v and t are equidistant from both vertices x and y, otherwise we received the required inequality for u. By Lemma 2 there are vertices $s' \in I(v, y) \cap I(t, y)$ and $s'' \in I(v, x) \cap I(t, x)$. Again, as in Lemma 6 vertices v, s' and s'' are equidistant from u and therefore $s', s'' \in I(v, u)$. By Lemma 1 these vertices must be adjacent. Since $s' \in L(x, r(G) + 1, y)$ and $s'' \in L(x, r(G) - 1, y)$ this is impossible. □

3 Computing the Gates and Distances of Vertices to Clique

In this section we present a linear–time algorithm for computing the distances from all vertices to any fixed clique C of a chordal graph G. For any vertex $v \in V$ besides the distance *dist(v)*$= d(v, C)$ we find some gate

$$gate(v) \in \cap\{I(v, u) : u \in Pr(v, C)\},$$

and also the size *num(v)* of this projection $Pr(v, C)$. Note that the existence of such a vertex *gate(v)* follows from Lemma 2. The algorithm is based on a modification of a *maximum cardinality search*; see Tarjan and Yannakakis [16]. Their linear–time test for graph chordality numbers the vertices of a graph from n to 1. A vertex adjacent to the largest number of previously numbered vertices will be the next selected vertex. In our algorithm we process the vertices of the graph beginning with the vertices of the clique C. As in [16] we maintain an array of sets *set(i)* for $0 \leq i \leq m-1$. We store in *set(i)* all unnumbered vertices adjacent to exactly i numbered vertices. Initially *set(0)* contains all the vertices. After steps (1)–(13) any vertex $w \notin C$ adjacent to some vertex of the clique C is included in *set(size(w))*, where *size(w)* is the number of vertices from C adjacent to w, all other vertices stay in *set(0)*. Further we maintain the largest index j such that *set(j)* is nonempty. Then we remove a vertex v from *set(j)* and number it. For each unnumbered vertex w adjacent to v, we move w from the set containing it, say *set(i)*, to *set(i+1)* (steps (16)–(21)). Among the numbered vertices adjacent to v we find a vertex z with minimal *dist(z)*= $d(z, C)$ and if there are several such vertices a vertex z with maximal *num(z)* is chosen. In other words this is a vertex with the largest projection on the clique C (steps (22)–(23)). Finally for the vertex v define

$$dist(v) = dist(z) + 1, num(v) = num(z), gate(v) = gate(z)$$

(steps (24)–(26)). The correctness of these steps follows from Lemma 4. Note that v is a simplicial vertex of a subgraph induced by all numbered vertices, i.e. the set $C(v)$ of numbered vertices adjacent to v is a clique. According to Lemma 4 the projections of any two vertices from $C(v)$ are comparable. Hence in $C(v)$ there is a vertex, whose projection covers any other projection on C. Obviously, any vertex with maximal projection, in particular the vertex z, has this property. Since the subgraph induced by numbered vertices is a distance–preserving subgraph of a graph G, *dist(v)* computed in steps (25) or (26) is a distance from v to the clique C. After all this we add one to j and while *set(j)* is empty we update the value of j. As in the maximum cardinality search we represent each set by a doubly linked list of vertices and maintain for each vertex v the index of the set containing it, denoted by *size(v)*. Since the complexity of the updates of j is bounded by the total number of times j is incremented and the access to a set element requires a constant time, the whole complexity of the algorithm is $O(m)$.

Procedure *distance_to_clique*

Input: A chordal graph and its clique C;
Output: For all vertices of the graph the pairs *dist(v)* and *gate(v)*;

begin
(1) **for** $i \in \{0, 1, \ldots, n-1\}$ **do** $set(i) := \emptyset$;
(2) **for** $v \in vertices$ **do**
(3) **if** $(v \in C)$ **then**

```
      begin size(v) := −1; dist(v) := 0; num(v) := 1; gate(v) = ^ end;
    else
      begin size(v) := 0; add v to set(0) end;
  j := 0; k := 0;
  for v ∈ C do begin
    for (v, w) ∈ E such that size(w) ≥ 0 do
      begin
        delete w from set(size(w));
        size(w) := size(w) + 1; k := 1; num(w) := size(w);
        add w to set(size(w))
      end;
    if (k = 1) then
      begin j := j + 1; k := 0 end;
           end;
  i :=| C | +1; (* | C | is the number of vertices in clique C *)
  while (i ≤ n) do begin
    v := delete any from set(j); size(v) := −1;
    d := ∞; nm := 0;
    for (v, w) ∈ E do
      if (size(w) ≥ 0) then begin
        delete w from set(size(w)); size(w) := size(w) + 1;
        add w to set(size(w))
                          end;
      else (* size(w) = −1 *)
        if (dist(w) < d or (dist(w) = d and num(w) > nm)) then
          begin z := w; d := dist(w); nm := num(w) end;
    if (dist(z) = 0) then
      begin dist(v) := 1; gate(v) := v end;
    else
      begin
        dist(v) := dist(z) + 1; num(v) := num(z); gate(v) = gate(z)
      end;
    i := i + 1; j := j + 1;
    while (j ≥ 0 and set(j) = ∅) do j := j − 1;
                end;
end;
```

4 Computing the Radius and a Central Vertex of a Chordal Graph

The algorithm presented below for finding a central vertex is based on the properties of chordal graphs stated in Lemmas 3, 5 and 6.

Procedure *central_vertex*

Input: A chordal graph G;
Output: A central vertex and the radius of G;

```
begin
(1)   x := any vertex of graph;
(2)   d := 0;
(3)   y := the farthest vertex from x;
(4)   delta := d(x, y);
(5)   while (d < delta ) do begin
(6)     z := y; d :=delta ;
(7)     y := the farthest vertex from z;
(8)     delta := d(y, z)
                                    end;
(9)   C:= the set L(y, ⌊delta/2⌋, z);
(10)  call the procedure distance_to_clique for the clique C;
(11)  R := max{dist(u) : u ∈ vertices};
(12)  for v ∈ vertices do num(v) := 0;
(13)  param := 0;
(14)  for v ∈ vertices do
(15)    if (dist(v) = R) then
(16)      begin param := param + 1; num(gate(v)) := num(gate(v)) + 1 end;
(17)  for v ∈ C do size(v) := 0;
(18)  for v ∈ C do
(19)    for (v, w) ∈ E do size(v) := size(v) + num(w);
(20)  c := vertex in C with maximal size();
(21)  if (size(c) = param) then  (* e(c) = R *)
(22)    begin c is central vertex of G; r := R; stop end;
      else  (* size(c) < param and so e(c) = R + 1 *)
(23)    if ((delta is even) and (R=delta /2)) then
(24)      begin c is a central vertex of G; r := R + 1;   stop end;
        else
(25)      begin x:= the farthest vertex from c; d:=delta ;
(26)        goto step (3)
          end;
end;
```

Theorem *The central_vertex algorithm correctly finds a central vertex of a chordal graph G in time $O(m)$.*

Proof We begin with the description of the algorithm. First we find a pair of mutually farthest vertices (steps (1)–(8)). To find these vertices, we choose an arbitrary vertex x, find a farthest vertex y from x, and then find a farthest vertex z from y. Put $delta = d(y, z)$. By Corollary $delta \geq d(G) - 2$. If $e(z) > e(y)$ then we add one to *delta* and choose the farthest vertex from z. Denote this vertex by y and repeat the same operations until $delta = e(y) = e(z)$. Since $delta \leq d(G)$ and initially $delta \geq d(G) - 2$ there are at most two improvements of the value of *delta*. Hence the procedure of finding farthest vertices and future improvement of this pair requires a total computational effort of $O(m)$.

In step (9) in time $O(m)$ we find the clique

$$C = L(y, \lfloor delta/2 \rfloor, z),$$

where y and z is the pair of mutually farthest vertices computed in steps (3)–(8). In order to do this, by the breadth first search algorithm we compute the distances from y and z to all other vertices of G and select whose vertices $v \in V$ for which the equalities $d(v, y) = \lfloor delta/2 \rfloor$ and $d(v, z) = d(y, z) - \lfloor delta/2 \rfloor$ hold. At the following step using the procedure *distance_to_clique* we compute *dist(v)* and *gate(v)* for the clique C and all vertices $v \in V$. Using the list *dist*() we compute the maximal distance R from vertices of G to C (step (11)). Put $D^* = \{v \in V : d(v, C) = R\}$. At the steps (12)–(16) for each vertex from the list *gate()* we compute *num(w)* : the number of vertices v from D^* for which *gate(v)=w*. These steps can be done in $O(n)$ time. At the following step for each vertex $v \in C$ find *size(v)*, which is the number of vertices from D^* whose projections on C contain vertex v (this operation takes $O(m)$ time). Further we choose a vertex $c \in C$ with maximal *size(c)*, i.e. a vertex of C which belongs to projections of a maximum number of vertices from D^*. As we will show below either c is a central vertex of G or for any vertex $x \in D(c)$ $e(x) > delta$ and we improve the value of *delta*. Since initially $delta \geq d(G) - 2$ there are at most two returns from step (26) to the step (3). All steps of the algorithm require linear time and therefore the algorithm computes a central vertex of G in $O(m)$ time.

We now finally come to proving the correctness of our algorithm.

Claim 1 If *delta* is even ($delta = 2k$) then $R \leq k + 1$.

Proof of Claim 1 Assume the contrary and let $u \in D^*$, i.e. $d(u, C) = R \geq k + 2$. Denote by v some vertex from metric projection of u on C and by w some vertex of the interval $I(u, v)$ adjacent to v. Then either $w \in I(v, y) \cup I(v, z)$ or $w \notin I(y, z)$. In the first case, if, say $w \in I(v, y)$, then by Lemma 3 $d(z, u) \geq d(z, v) + d(w, u) = k + R - 1 > delta$, contradicting our assumption that y and z are mutually farthest vertices. Now assume that $w \notin I(y, z)$. Then $v \in I(w, y) \cup I(w, z)$. Therefore if, say $v \in I(y, w)$, then again by Lemma 3

$$d(y, u) \geq d(y, v) + d(w, u) = k + R - 1 > delta,$$

which is a contradiction. □

Claim 2 If *delta* is odd ($delta = 2k - 1$) then $R = k$, i.e. $d(u, C) \leq k$ for any $u \in V$.

Proof of Claim 2 Assume that there exists a vertex for which $d(u, C) \geq k + 1$, and let $v \in Pr(u, C)$. Recall that $C = L(y, k - 1, z)$, i.e. $d(y, v) = k - 1, d(z, v) = k$. Denote by w some vertex of the interval $I(v, u)$ adjacent to v . We distinguish two cases.

Case 1 $w \in I(y, z)$, i.e. $w \in I(y, v) \cup I(v, z)$.

If $w \in I(y, v)$, then $v \in I(w, z)$ and by Lemma 3

$$d(z, u) \geq d(z, v) + d(w, u) \geq 2k,$$

in contradiction with the assumption that y and z are mutually farthest vertices. Now assume that $w \in I(v, z)$. Then from Lemma 3

$$d(y, u) \geq d(y, v) + d(w, u) \geq 2k - 1.$$

Since $z \in D(y)$ the equality holds $d(y,u) = 2k-1$. From the second part of the same lemma this equality holds iff there is a vertex $x \in I(v,u) \cap I(w,y)$ adjacent to v and w. Since $I(w,y) \subset I(z,y)$ we get $x \in L(y,k-1,z) = C$ and $d(u,x) < d(u,v)$, contradicting our assumption that $v \in Pr(u,C)$.

Case 2 $w \notin I(y,z)$, i.e. $v \in I(w,y) \cup I(w,z)$.

If $v \in I(w,z)$ then by Lemma 3 $d(u,z) \geq 2k$, contradicting our assumption that $e(z)$=*delta*. Therefore $v \in I(y,w)$ and $d(w,z) = d(v,z)$. From this equality and Lemma 2 we deduce that there exists a vertex $x \in I(w,z) \cap I(v,z)$ adjacent to w and v. Note that $w \in I(u,x)$, otherwise $x \in I(v,u)$ and we get in conditions of Case 1. Since in addition $v \in I(y,w)$, from Lemma 3 we deduce that

$$d(u,y) \geq d(u,w) + d(v,y) \geq 2k-1,$$

$$d(z,u) \geq d(z,x) + d(w,u) \geq 2k-1.$$

Since $e(y) = e(z) = 2k-1$ we obtain that $d(y,u) = d(z,u) = 2k-1$. By the second part of Lemma 3 there exist two vertices $s \in I(w,y) \cap I(v,u)$ and $t \in I(x,u) \cap I(w,z)$ adjacent to w,v and w,x correspondently (Fig. 4). Vertices s,t and w are equidistant from u. So by Lemma 2 there exist vertices $u' \in I(s,u) \cap I(w,u)$ and $u'' \in I(w,u) \cap I(t,u)$ adjacent to s,w and w,t respectively. As $u',u'' \in L(w,1,u)$ then these vertices either are adjacent or coincide. In any case we obtain the cycle (v,s,u',u'',t,x,v). In every triangulation of this cycle the edge (v,x) belongs to at least one triangle. Since $u',u'' \in L(v,2,u)$ the third vertex of such a triangle is one of the vertices s or t. This means that either $s \in I(v,u) \cap I(z,y)$ or $t \in I(v,u) \cap I(y,z)$ and we get in the conditions of the preceding case. □

Claim 3 If $e(c) > \lfloor delta/2 \rfloor + 1$ then for any vertex $u \in D(c)$ $e(u) > delta$.

Proof of Claim 3 Put $p = \lfloor delta/2 \rfloor + 1$. By Claims 1 and 2 $d(u,C) \leq p$ and therefore $e(c) = p+1$ and there is a vertex $w \in I(c,u) \cap C$. Since $u \in D^*$ and c is a vertex which belongs to a maximum number of projections of vertices from D^* to the clique C, there is such a vertex $u' \in D^*$ that $c \in I(w,u')$. By Lemma 3 $d(u,u') \geq d(u,w) + d(c,u') = 2p$ and so $e(u) > delta$. □

Our final claim will complete the proof of the theorem.

Claim 4 $c \in C(G)$, except the case when $e(c) > \lfloor delta/2 \rfloor + 1$.

Proof of Claim 4 First we note that $r(G) \geq \lfloor delta + 1 \rfloor/2$. If *delta* is odd then either $e(c) = R = \lfloor delta + 1 \rfloor/2$ and so $c \in C(G)$ or by Claim 3 for any vertex $u \in D(c)$ $e(u) > delta$. Now assume that *delta* is even. If $e(c) > delta/2+1$ then by Claim 3 $e(u) > delta$ for any $u \in D(c)$. Hence suppose that $e(c) \leq delta/2+1$. In order to show that $c \in C(G)$ it is enough to consider only the case when $r(G) = delta/2$. Then $d(G) = d(y,z) = 2r(G)$ and by Lemma 5

$$C(G) \subset L(y, r(G), z) = C.$$

On the other hand $R = r(G)$ and all metric projections of vertices from D^* on C have a non-empty intersection. From our choice the vertex c must be in this intersection. Hence $e(c) = r(G)$ and therefore $c \in C(G)$ □

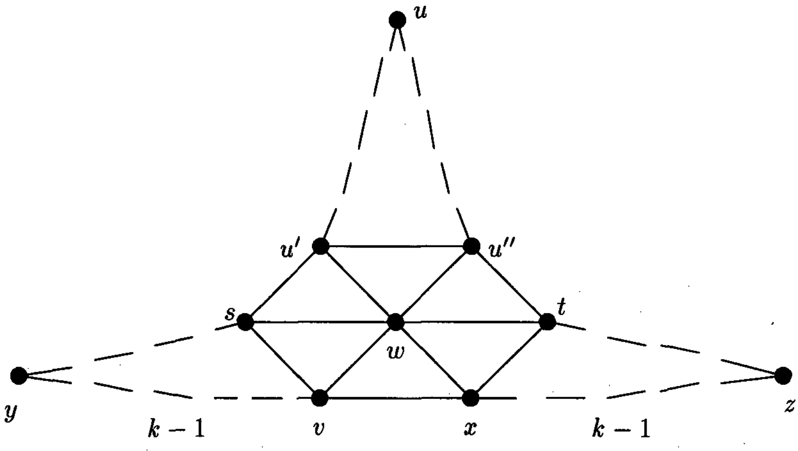

Fig. 4

Remark 1 Using this algorithm we can solve the following more general center problem :

> for a given subset $M \subset V$ of vertices of a chordal graph G find a vertex $v \in V$ such that $e_M(v)$ is minimal, where
>
> $$e_M(v) = \max\{d(v,u) : u \in M\}.$$

Remark 2 As a consequence of our algorithm we obtain that the interval $I(u,v)$ between any diametral vertices u and v intersects the center $C(G)$ of a chordal graph G. According to the algorithm either $L(u, \lfloor d(u,v)/2 \rfloor, v) \cap C(G) \neq \emptyset$ or we find a pair of vertices with a larger distance that is impossible in this case.

Open problem Find subquadratic time algorithms for computing the diameter $d(G)$ and the whole center $C(G)$ of a chordal graph G.

References

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

[2] G.J. Chang, Centers of chordal graphs, *Graphs and Combinatorics* 7 (1991) 305–313.

[3] V.D. Chepoi, Some properties of d–convexity in triangulated graphs (in Russian), *Mathematical Researches (Chişinău)* 87 (1986) 164–177.

[4] V.D. Chepoi, Centers of triangulated graphs (in Russian), *Matematiceskie Zametki (English Transl.: Mathematical Notes)* 43 (1988) 143–151.

[5] F.F. Dragan, Centers of graphs and Helly property (in Russian), *PhD Thesis, Moldova State University,* 1989.

[6] P. Duchet, Classical perfect graphs: an introduction with emphasis on triangulated and interval graphs, *Ann. Discrete Math.* 21 (1984) 67–96.

[7] M. Farber and R.E. Jamison, Convexity in graphs and hypergraphs, *SIAM J. Algebraic Discrete Meth.* 7 (1986) 433–444

[8] A.M. Farley and A. Proskurowski, Computation of the center and diameter of outerplanar graphs, *Discrete Appl. Math.* 2 (1980) 85–191.

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

[10] A. Hajnal and J. Suranyi, Über die Auflösung von Graphen in vollständige Teilgraphen, *Ann. Univ. Sci. Budapest Eötvös Sect. Math.* 1 (1958) 113–121.

[11] G. Handler, Minimax location of a facility in an undirected tree graph, *Transportation Sci.* 7 (1973) 287–293.

[12] S.M. Hedetniemi, E.J. Cockayne and S.T. Hedetniemi, Linear algorithms for finding the Jordan center and path center of a tree, *Transportation Sci.* 15 (1981) 98–114.

[13] R. Laskar and D. Shier, On powers and centers of chordal graphs , *Discrete Appl. Math.* 6 (1983) 139–147.

[14] S. Olariu, A simple linear-time algorithm for computing the center of an interval graph *Tech. Rept. TR-89-20 Old Dominion University, Norfolk* (1989).

[15] A. Proskurowski, Centers of 2–trees, *Ann. Discrete Math.* 9 (1980) 1–5.

[16] R.E. Tarjan and M. Yannakakis, Simple linear–time algorithms for test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs, *SIAM J. Comput.* 13 (1984) 566–579.