SIAM J. DISCRETE MATH.
Vol. 0, No. 0, pp. 000–000

UNIT CIRCULAR-ARC GRAPH REPRESENTATIONS AND FEASIBLE CIRCULATIONS[*]

MIN CHIH LIN[†] AND JAYME L. SZWARCFITER[‡]

Abstract. In a recent paper, Durán et al. [*J. Algorithms*, 58 (2006), pp. 67–78] described an algorithm of complexity $O(n^2)$ for recognizing whether a graph G with n vertices and m edges is a unit circular-arc (UCA) graph. Furthermore, the following open questions were posed in the above paper: (i) Is it possible to construct a UCA model for G in polynomial time? (ii) Is it possible to construct a UCA model, whose extremes of the arcs correspond to integers of polynomial size? (iii) If (ii) is true, could such a model be constructed in polynomial time? In the present paper, we describe a characterization of UCA graphs, based on network circulations. The characterization leads to a different recognition algorithm and to answering these questions in the affirmative. We construct a UCA model whose extremes of the arcs correspond to integers of size $O(n)$. The proposed algorithms, for recognizing UCA graphs and constructing UCA models, have complexities $O(n+m)$. Furthermore, the complexities reduce to $O(n)$, if a proper circular-arc (PCA) model of G is already given as the input, provided the extremes of the arcs are ordered. We remark that a PCA model of G can be constructed in $O(n+m)$ time, using the algorithm by Deng, Hell, and Huang [*SIAM J. Comput.*, 25 (1996), pp. 390–403]. Finally, we also describe a linear time algorithm for finding feasible circulations in networks with nonnegative lower capacities and unbounded upper capacities. Such an algorithm is employed in the model construction for UCA graphs.

Key words. algorithm, circular-arc graph, circular-arc model, circulations, networks, proper circular-arc graph, unit circular-arc graph

AMS subject classifications. 05C62, 05C85, 05C05, 68R10, 90B10

DOI. 10.1137/060650805

1. Introduction.

In a recent paper, Durán et al. [2] described an algorithm of complexity $O(n^2)$ for recognizing whether a graph G, with n vertices and m edges, is a unit circular-arc (UCA) graph. Furthermore, the following open questions were posed in [2]:

(i) Is it possible to construct a UCA model for G in polynomial time?
(ii) Is it possible to construct a UCA model, whose extreme points of the arcs correspond to integers of polynomial size?
(iii) If (ii) is true, could such a model be constructed in polynomial time?

Problems (ii) and (iii) were also proposed in the book by Spinrad [10]. As for problem (i), the proof of the characterization of UCA graphs in terms of forbidden subgraphs by Tucker [13], actually contains an algorithm for constructing a UCA model. However, due to the possible manipulation of large integers, the complexity of this algorithm is unknown, and so far the construction of a UCA model in polynomial time remains unsolved [2, 10].

[*]Received by the editors January 24, 2006; accepted for publication (in revised form) November 12, 2007; published electronically DATE.
http://www.siam.org/journals/sidma/x-x/65080.html

[†]Departamento de Computación, Universidad de Buenos Aires, Facultad de Ciencias Exactas y Naturales, Buenos Aires 1428, Argentina (oscarlin@dc.uba.ar). This author's research was partially supported by UBACyT grants X184 and X212, PICT ANPCyT grant 11-09112, and CNPq under PROSUL project Proc. 490333/2004-4.

[‡]Universidade Federal do Rio de Janeiro, Instituto de Matemática, NCE and COPPE, Caixa Postal 2324, 20001-970 Rio de Janeiro, Brazil (jayme@nce.ufrj.br). This author's research was partially supported by the Conselho Nacional de Desenvolvimento Científico e Tecnológico, CNPq, and Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro, FAPERJ, Brazil.

In the present paper, we propose a characterization for UCA graphs, which leads to a different recognition algorithm and to answering in affirmative the previous three questions. An extended abstract of the present paper appeared in [7]. The proposed algorithm recognizes UCA graphs and constructs UCA models whose extremes of the arcs correspond to integers of $O(n)$ size, in overall time $O(n+m)$. Furthermore, if the graph G is already given by a proper circular-arc (PCA) model of it, then the complexity of the proposed algorithm reduces to $O(n)$, provided the extremes of the arcs are ordered. Observe that a PCA model for G, when given by its vertices and edges, can be constructed in $O(n+m)$ time, employing the algorithm by Deng, Hell, and Huang [1]. Recall that UCA graphs are PCA graphs.

Our method employs network circulations. We formulate an algorithm finding a feasible circulation in a network with nonnegative lower capacities and unbounded upper capacities. The algorithm has linear time complexity in the size of the network.

Denote by G an undirected graph, with vertex set $V(G)$ and edge set $E(G)$. For an edge $e \in E(G)$, write $e = uv$, where u and v are the *extremes* of e. Denote by $d_G(v)$ for the *degree* of v in G. We may also simply write $d(v)$, instead. Use a similar notation for a digraph D. For a directed edge $e = uv \in E(D)$, say that u is the *start* and v the *end* of e. Write $d^-_G(v)$ and $d^+_G(v)$ for the *indegree* and *outdegree* of vertex v, respectively. Again, also write $d^-(v)$ and $d^+(v)$, simply. Moreover, $E^-(v)$, $E^+(v) \subseteq E(D)$, represent the set of edges of D entering and leaving v, respectively. Also, D is *connected* when its underlying (undirected) graph is connected. Say that D is *strongly connected* when D contains paths from u to v and from v to u, for any $u, v \in V(D)$. Finally, the *bridge* of D is an edge contained in no (directed) cycle of D.

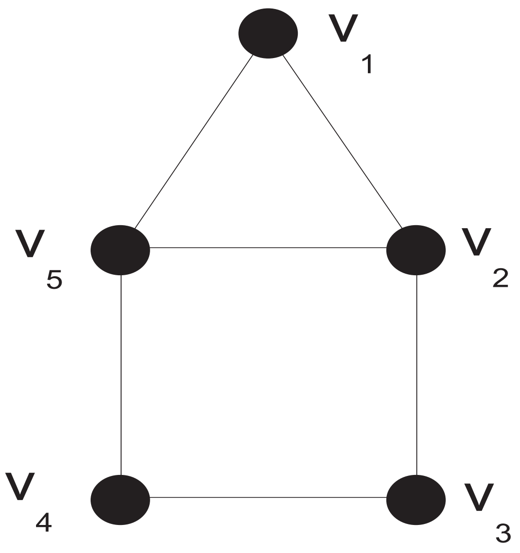
An *out-arborescence* (*in-arborescence*) T of D is a spanning connected subdigraph of D such that there is a distinguished vertex $v^* \in V(T)$, called the *root* of T, satisfying $d^-_T(v^*) = 0$ ($d^+_T(v^*) = 0$), while $d^-_T(v) = 1$ ($d^+_T(v) = 1$) for all of the remaining vertices $v \neq v^*$ of T. Clearly, a strongly connected digraph admits both an out-arborescence and in-arborescence, with root at any arbitrary vertex. When $d^+_T(v) = 0$ ($d^-_T(v) = 0$), call v a *leaf* of T. For $u, v \in V(T)$, if T contains a path from u to v, then u is an *ancestor* of v, and v a *descendant* of u. A *leaf-root ordering* of T is a sequence $v_1, \ldots, v_n$ of its vertices, such that $i < j$ implies v_i is not an ancestor (descendant) of v_j in T.

A *network* is a digraph D, having real values b_j, c_j associated to each edge $e_j \in E(D)$. Call b_j the *lower capacity* of e_j, while c_j is the *upper capacity* of e_j. A *circulation* of D is a function W assigning a real number w_j to each edge e_j of D, called the *flow of* e_j. Denote $w^-(v_i) = \sum_{w_j \in E^-(v_i)} w_j$ and $w^+ = \sum_{w_j \in E^+(v_i)} w_j$, for each $v_i \in V(D)$. A circulation is *feasible* when it satisfies

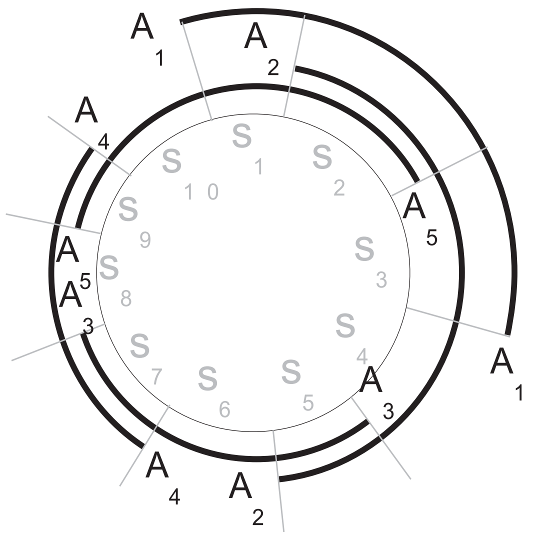
$$w^-(v_i) = w^+(v_i), \text{ for each } v_i \in V(D), \text{ and } b_j \leq w_j \leq c_j, \text{ for each } e_j \in E(D).$$

For our purposes, we only consider networks with finite nonnegative lower capacities and flow values, and unbounded the upper capacities. In this case, the condition $w_j \leq c_j$ is always satisfied.

A *circular-arc* (CA) model for a graph G is a pair $(C, \mathcal{A})$, where C is a circle and $\mathcal{A}$ is a collection of arcs of C, such that each arc $A_i \in \mathcal{A}$ corresponds to a vertex $v_i \in V(G)$, and any A_i, A_j intersects precisely when v_i, v_j are adjacent in G, $1 \leq i, j \leq n$, and $i \neq j$. A CA graph is one admitting a CA model. When no arc of $\mathcal{A}$ properly contains another arc of $\mathcal{A}$ then $(C, \mathcal{A})$ is a PCA model, while if all arcs of $\mathcal{A}$ have the same length, then $(C, \mathcal{A})$ is a UCA model. A PCA (UCA) graph is one admitting a PCA (UCA) model. A *normal* model is a PCA model where no two



(a)



(b)

Fig. 1. *A PCA graph and a normal model of it.*

arcs cover the circle. Figure 1(a) depicts a PCA graph and a normal model of it is in Figure 1(b). When traversing the circle C, always choose the clockwise direction. If s, t are points of C, then write (s, t) to mean the arc of C defined by traversing the circle from s to t. Call s, t the *extremes* of (s, t), while s is the *start* and t the *end* of the arc. The *extremes* of $\mathcal{A}$ are the extremes of $A_i \in \mathcal{A}$. For $A_i \in \mathcal{A}$, write $A_i = (s_i, t_i)$. We assume the labelling $A_1, \ldots, A_n$ of the arcs is such that the sequence of the start points $s_1, \ldots, s_n$ is in the circular ordering of C. For $p, q \in A_i$, if $(s_i, p) \subseteq (s_i, q)$, then p *precedes* q, and q *succeeds* p in A_i. For $p, q, t \in C$, write $\max\{p, q\}_t$ to represent the point, p or q, which is farthest from t, in the circular ordering of C. If p is an extreme of $\mathcal{A}$, then denote by $PRED(p)$ and $SUC(p)$, the extreme of $\mathcal{A}$, which immediately precedes and succeeds p, in the circular ordering, respectively. We assume that all arcs of C are open, no two extremes of distinct arcs of $\mathcal{A}$ coincide, and no single arc covers C. An arc of C defined by two consecutive extremes of $\mathcal{A}$ is a *segment* of $(C, \mathcal{A})$. Clearly, C is the union of the $2n$ segments of $(C, \mathcal{A})$ and the extreme points. Figure 1(b) shows the ten segments of the corresponding CA model.

The first characterization of CA graphs leading to a polynomial time recognition algorithm is due to Tucker [12, 14]. Subsequently, faster algorithms have been described by Hsu [6] and by Eschen and Spinrad [3]. More recently, a linear time recognition algorithm for CA graphs has been formulated by McConnell [8]. Deng, Hell, and Huang [1] described an algorithm for recognizing PCA graphs in linear time. This algorithm also constructs a corresponding PCA model for the graph. As for UCA graphs, the first polynomial time recognition algorithm is that by Durán et al. [2].

Given a graph G, the algorithm [2] initially employs algorithm [1] to construct a PCA model $(C, \mathcal{A})$ of G, if any. Clearly, G is not UCA if such a model does not exist. Afterwards, this algorithm proceeds with two main phases: The first of them is to transform $(C, \mathcal{A})$ into a normal model. The last phase is the actual algorithm for deciding if G is a UCA graph, employing the normal model constructed in the first phase. The complexities of the algorithms corresponding to these two phases are both $O(n^2)$. The method proposed in the present paper is also composed by similar phases. However, the complexities of the two corresponding algorithms are $O(n)$.

In section 2, we describe the characterizations in which are based the proposed algorithms. The characterization of UCA graphs relates them to circulations in special networks. Section 3 describes the algorithm for finding feasible circulations in general networks, having nonnegative lower capacities and unbounded upper capacities. Section 4 presents the algorithm for constructing normal models. Section 5 contains the actual algorithm for recognizing UCA graphs and constructing UCA models. Further comments form the last section.

2. A characterization for UCA graphs.

In this section, we characterize UCA graphs in terms of circulations of a special network. The following two theorems are basic to our purposes.

THEOREM 1 (see Golumbic [4] and Tucker [13]). *Let G be a PCA graph. Then G admits a normal model.*

THEOREM 2 (see Tucker [13]). *Let G be a UCA graph and $(C, \mathcal{A})$ a normal model of it. Then G admits a UCA model, such that its extremes are in the same circular ordering as those of $(C, \mathcal{A})$.*

Let G be a CA graph and $(C, \mathcal{A})$ a CA model of it. Denote by $S_1, \dots, S_{2n}$ the segments of $(C, \mathcal{A})$ in circular ordering, where the start of S_1 and that of A_1 coincide. Denote by l_j the length of S_j. Clearly, any arc $A_i \in \mathcal{A}$ may be decomposed into the segments which form it. The length of A_i equals $\sum_{S_j \subseteq A_i} l_j$. The decomposition allows one to represent relations between lengths of arcs by relations between lengths of the corresponding segments. Consequently, the condition of equality between any two arc lengths required by a UCA model can be expressed by a system of $n-1$ linear equations q_i, together with $2n$ inequalities, called the *full system* of $(C, \mathcal{A})$:

$$q_i : \sum_{S_j \subseteq A_i} l_j = \sum_{S_j \subseteq A_{i+1}} l_j, \ i = 1, \dots, n-1, \tag{2.1}$$

$$l_j > 0,\ j = 1, \ldots, 2n. \tag{2.2}$$

Clearly, equation q_i corresponds to the condition that the length of arc A_i equals that of A_{i+1}. The segment lengths l_j are the variables of such a system of equations. Our aim is to find a solution of the full system, if existing. The values of l_j, obtained from the solution of the system, would define a UCA model for G. In equation q_i, call the term $\sum_{S_j \subseteq A_i} l_j$ the *left side* of q_i, while $\sum_{S_j \subseteq A_{i+1}} l_j$ is its *right side*. Each equation q_i can possibly be simplified if the length l_j of a same segment appears in both the left and right sides of q_i. In this case, subtract l_j from both sides of q_i. The equations so obtained form the *reduced system* of $(C, \mathcal{A})$. The following lemma describes a useful property of these systems.

Lemma 3. *Let G be a PCA graph, $(C, \mathcal{A})$ a normal model of it, and R the reduced system of $(C, \mathcal{A})$. Then each segment length l_j of $(C, \mathcal{A})$ appears at least once in R, unless S_j is a segment contained in all arcs of $\mathcal{A}$ or in none of them. Furthermore, l_j appears at most twice in R. In the latter situation, the two instances of l_j in R occur in different sides of their equations.*

Proof. Let S_j be any segment of the normal model $(C, \mathcal{A})$. Denote by $\mathcal{A}'$ the set of arcs of $\mathcal{A}$ covering S_j. If $\mathcal{A}' = \emptyset$, then no arc of $\mathcal{A}$ contains S_j, implying that l_j does not appear in R, and the lemma is satisfied. Consider $\mathcal{A}' \neq \emptyset$. We know that $\mathcal{A}'$ does not entirely cover C, otherwise there would be two arcs of $\mathcal{A}$ which would do so, contradicting $(C, \mathcal{A})$ to be normal. Consequently, the circular ordering of the extremes of $(C, \mathcal{A})$, when restricted to the arcs of $\mathcal{A}'$, is a linear ordering. Let A_i and A_k be the first and last arcs of $\mathcal{A}'$, respectively, in the considered linear ordering.

First, let $i \leq k$. Observe that l_j is not on the right side of any of the equations $q_i, \ldots, q_k$, of R. Because if l_j is on the right side of q_t, then $k \geq t \geq i$ in the full system. However, l_j is also on the left side of q_t, hence it gets simplified in R. However, l_j is on the right side of q_{i-1} in R provided $i > 1$. Similarly, l_j is not on the left side of any $q_i, \ldots, q_{k-1}$ but it is at the left of q_k, provided $k < n$. On the other hand, $i = 1$ and $k = n$ imply that S_j is common to all arcs of $\mathcal{A}$. Consequently, each l_j appears at least once in R, unless S_j is contained in all arcs of $\mathcal{A}$, or in none of them.

Consider the alternative $i > k$. If $i - 1 > k$, then l_j appears on the left of q_{i-1} and on the right of q_k. There are no other occurrences of l_j. Finally, when $i - 1 = k$ it follows that S_j is common to all arcs of $\mathcal{A}$. Again, each l_j always appears at least once in R, except when S_j is contained in all arcs of $\mathcal{A}$. Furthermore, it appears at most twice in R. In the latter situation, l_j occurs on different sides of the corresponding equations. $\square$

Aiming to solve R efficiently, we describe a graph theoretical model for it. Define the *segment digraph* D, as follows. There is a vertex $v_i \in V(D)$ for each equation q_i of R, and one edge e_j for each segment S_j of $(C, \mathcal{A})$. In addition, there is a distinguished vertex $v_0 \in V(D)$. Each edge $e_j \in E(D)$ is directed according to:

(i) If l_j appears at the left of some equation q_i of R, then e_j starts at v_i; otherwise it starts at v_0.

(ii) If l_j appears at the right of some equation q_k, then e_j ends at v_k; otherwise it ends at v_0.

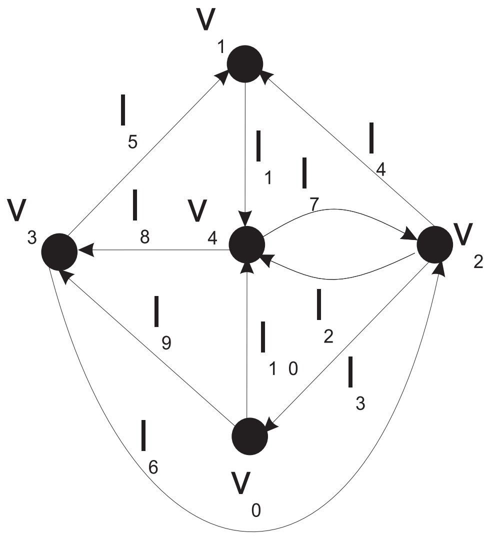
The description of D is complete. Clearly, D has n vertices and $2n$ edges and no loops, except possibly at v_0.

By adding to each edge $e_j \in E(D)$ a lower capacity $b_j = 1$ and an unbounded upper capacity c_j, we obtain the *segment network* of D.

As an example, the reduced system of the PCA model of Figure 1(b) is illustrated in Figure 2(a), while the corresponding segment digraph is in Figure 2(b). Similarly,

$$R \quad \begin{array}{ll} q_1: & l_1 = l_4 + l_5 \\ q_2: & l_2 + l_3 + l_4 = l_6 + l_7 \\ q_3: & l_5 + l_6 = l_8 + l_9 \\ q_4: & l_7 + l_8 = l_{10} + l_1 + l_2 \end{array}$$

(a)



(b)

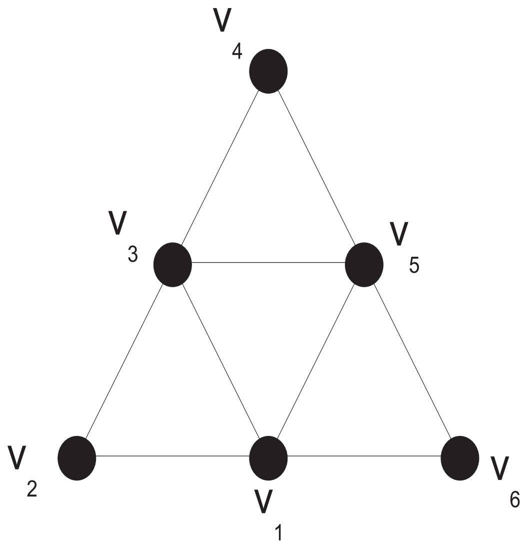
FIG. 2. *The reduced system of Figure* 1(b) *and its segment digraph.*

the reduced system and segment digraph corresponding to the PCA model of Figure 3(b) appear in Figures 4(a) and 4(b), respectively.

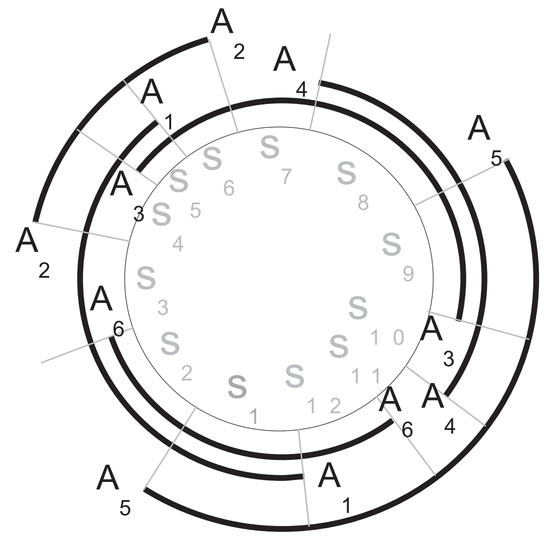
The following theorem characterizes UCA graphs in terms of feasible circulations.

THEOREM 4. *Let G be a PCA graph, $(C, \mathcal{A})$ a normal model of it, and D its segment network. Then G is a UCA graph if and only if D admits a feasible circulation.*

Proof. Let G be a UCA graph, and $(C, \mathcal{A})$ a normal model of G. By Theorem 2, there exists a UCA model $(C', \mathcal{A}')$ of G such that the endpoints of the arcs of $\mathcal{A}$ and $\mathcal{A}'$ are in the same ordering. That is, $(C, \mathcal{A})$ and $(C', \mathcal{A}')$ differ only by the lengths of the corresponding segments. Let R be the reduced system of $(C, \mathcal{A})$ and D be the segment network. Define a circulation W by assigning a flow w_j to each edge e_j of D, equal to the length l'_j of the segment of $(C', \mathcal{A}')$, corresponding to S_j. We show that such an assigment W is a circulation of D. Let $v_i \in V(D)$, $1 \le i \le n-1$, and q_i the corresponding equation in R. Then $w^-(v_i) = \sum_{S_j \subseteq A_i \setminus A_{i+1}} l'_j$. Similarly, $w^+(v_i) = \sum_{S_j \subseteq A_{i+1} \setminus A_i} l'_j$. Because $(C', \mathcal{A}')$ is a UCA model, the lengths of arcs A'_i and A'_{i+1} of $\mathcal{A}'$ are the same, implying that $w^-(v_i) = w^+(v_i)$. It remains to show that this equality also holds for v_0. This assertion follows from the fact that $\sum_{0 \le i \le n-1} w^-(v_i) = \sum_{0 \le i \le n-1} w^+(v_i)$, because every edge e_j counts l'_j units both in $\sum_{0 \le i \le n-1} w^-(v_i)$ and $\sum_{0 \le i \le n-1} w^+(v_i)$. Since $w^-(v_i) = w^+(v_i)$, for each $1 \le i \le n-1$, it follows that $w^-(v_0) = w^+(v_0)$. Consequently, W is indeed a feasible circulation of D.



(a)



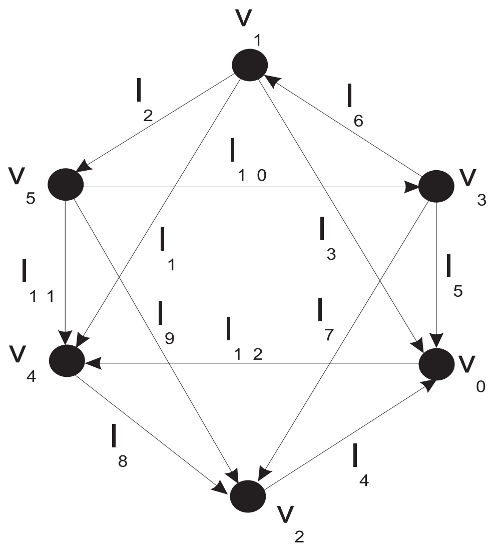
(b)

FIG. 3. *Another PCA graph with its normal model.*

Conversely, by hypothesis, D admits a feasible circulation W. We prove that G is UCA by constructing a UCA model $(C', \mathcal{A}')$ for G. First, consider the situation where no segment S_j of $(C, \mathcal{A})$ is contained in all arcs of $\mathcal{A}$ or in none of them. Let w_j be the flow of edge e_j of D. Construct the model $(C', \mathcal{A}')$ of G, by maintaining the endpoints of the arcs of $\mathcal{A}'$ in the same circular ordering as those in $\mathcal{A}$, while possibly modifying the lengths of the segments. The length of C' is defined as $\sum_{e_j \in E(D)} w_j$, while the length of the segment of $(C', \mathcal{A}')$ corresponding to S_j is set to the flow w_j of e_j. We show that the model so constructed is a UCA model.

$$R \quad \begin{array}{ll} q_1: & l_1 + l_2 + l_3 = l_6 \\ q_2: & l_4 = l_7 + l_8 + l_9 \\ q_3: & l_5 + l_6 + l_7 = l_{10} \\ q_4: & l_8 = l_{11} + l_{12} + l_1 \\ q_5: & l_9 + l_{10} + l_{11} = l_2 \end{array}$$

(a)



(b)

FIG. 4. *The reduced system of Figure* 3(b) *and its segment digraph.*

Let R be the reduced system of $(C, \mathcal{A})$. We show that the assigment $l_j := w_j$ for each segment S_j of $(C, \mathcal{A})$ is a solution to R. Let q_i be an equation of R and v_i the corresponding vertex of D. By the construction of D, the left side of q_i corresponds to the flow values of the edges of D ending at v_i, while the right side of q_i corresponds to the flow values of the edges, starting at v_i. Because D is feasible, $w^-(v_i) = w^+(v_i)$, implying that $\sum_{S_j \subseteq A_i} l_j = \sum_{S_j \subseteq A_{i+1}} l_j$. Moreover, all l_j are positive, because the lower capacities are satisfied. Consequently, W is a solution to R. Clearly, any solution to R is also a solution to the full system. On the other hand, because no segment is contained in all arcs of $\mathcal{A}$ or in none of them, Lemma 3 implies that all segment lengths appearing in the full system also appears in the reduced system, and consequently have been assigned a required value, The latter implies that all arc lengths of $\mathcal{A}'$ are equal, meaning that $(C', \mathcal{A}')$ is a UCA model.

Finally, examine the situation where there is a segment S_j contained in all arcs of $\mathcal{A}$ or in none of them. Construct $(C', \mathcal{A}')$ as above, and assign any positive length to S_j. Clearly, all arc lengths of $\mathcal{A}'$ are equal and $(C', \mathcal{A}')$ is a UCA model. ∎

The above theorem implies that the problem of recognizing if a given graph G is UCA is equivalent to deciding if the corresponding segment network admits a feasible circulation. We describe a characterization for the existence of circulations in general networks, with arbitrary real nonnegative lower capacities and unbounded upper capacities. This is a special case of Hoffman's circulation theorem, as below. See [9]. For a subdigraph B of D, write $b^-(B) = \sum_{e_j \in E^-(B)} b_j$ and $c^+(B) = \sum_{e_j \in E^+(B)} c_j$

THEOREM 5 (see Hoffman [5]). *Let D be a network having an arbitrary real lower capacity b_j and upper capacity c_j, for each edge $e_j \in E(D)$. Then D admits a feasible circulation W if and only if $b^-(B) \leq c^+(B)$, for each subdigraph B of D. Moreover, if b and c are integers, then the flow values can be taken as integers.*

COROLLARY 6. *Let D be a network with real nonnegative lower capacities and unbounded upper capacities. Then D admits a feasible circulation if and only if all bridges of D have lower capacity zero. In this case, a circulation with integer flow values always exists.*

Proof. Suppose D admits a feasible circulation W. To the contrary, assume that D has a bridge e_j with positive lower capacity. Then $e_j \in E^+(B), E^-(B')$ for some distinct strongly connected components B, B' of D. In this situation, we know that $w(E^-(B')) > 0$ and $w(E^+(B')) > 0$. The latter implies that there exists some edge e_k leaving B' and enters a strongly connected component $B'' \neq B, B'$, such that $w_k > 0$. Repeating this argument over and over leads to the contradiction that D is not finite. Consequently, no such e_j may exist.

Conversely, by hypothesis all bridges of D have lower capacity equal to zero. Let B be any strongly connected component of D. Then any proper subdigraph B' of B satisfies $E^-(B'), E^+(B') \neq \emptyset$. Since the upper capacities of D are unbounded and the lower capacities are not, it follows that $b^-(B') < c^+(B')$. Using Theorem 5, we conclude that B admits a feasible circulation, then repeat this argument for the remaining strongly connected components of D. Finally, assign the flow value equal to zero to all bridges of D. We have so obtained a feasible circulation. By simply replacing each lower capacity b_j by the value $\lceil b_j \rceil$, we obtain a feasible circulation with integer flow values. $\square$

As examples, observe that the segment digraph of Figure 2(b) is strongly connected, while that of Figure 4(b) is not. Consequently, the graph of Figure 1(a) is a UCA graph, while that of Figure 3(a) is not.

3. Finding feasible circulations. Let D be a network having finite real nonnegative lower capacities and unbounded upper capacities. We describe an algorithm which finds a feasible circulation W of D, or reports that such circulation does not exist. For each edge e_j of D, let b_j be the given nonnegative lower capacity. The algorithm below computes integer flow values w_j of the feasible circulation W of D.

1. Find the strongly connected components and the bridges of D. If any bridge has a positive lower capacity, then report that no feasible circulation exists and stop. Otherwise, set the flow value $w_j := 0$ for each bridge edge e_j of D. Then perform step 2 for each strongly connected component B of D. At termination, stop.
2. Assign the initial flow values $w_j := \lceil b_j \rceil$, for each edge $e_j \in E(B)$. Then compute $w^-(v_i) = \sum_{e_j \in E^-(v_i)} w_j$ and $w^+(v_i) = \sum_{e_j \in E^+(v_i)} w_j$ for each $v_i \in V(B)$. Finally, run *OUT PROCEDURE* and afterwards run *IN PROCEDURE.*

   OUT PROCEDURE. Choose an arbitrary vertex $v^* \in V(B)$ and find an out-arborescence T of B, with root v^*. Repeat the following operations, for each vertex $v \in V(T)$, $v \neq v^*$, in leaf-root ordering: if $w^-(v) < w^+(v)$ then set $w^+(u) := w^+(u) + w^+(v) - w^-(v)$, $w_j := w_j + w^+(v) - w^-(v)$ and $w^-(v) := w^+(v)$, where $e_j = uv$ is the edge of T ending at v.

IN PROCEDURE. Construct an in-arborescence T of B, with the same root v^*, as chosen in the computation of the *OUT PROCEDURE* for B. Repeat the following operations, for each vertex $v \in V(T)$, $v \neq v^*$, in leaf-root ordering: if $w^+(v) < w^-(v)$, then set $w^-(u) := w^+(u) + w^-(v) - w^+(v)$, $w_j := w_j + w^-(v) - w^+(v)$, and $w^+(v) := w^-(v)$, where $e_j = vu$ is the edge of T starting at v.

Next, we assert the correctness of the algorithm.

THEOREM 7. *The above algorithm finds a feasible circulation for D.*

Proof. From Corollary 6, it follows that the algorithm correctly reports the nonexistence of a circulation, whenever there exists a bridge having a positive lower capacity. Otherwise, let B be any strongly connected component of D. We show that the flow values assigned by the algorithm to the edges of B define a feasible circulation for B. Furthermore, since the bridges are assigned a zero flow, the latter implies that the circulation of D so obtained is also feasible.

First, observe that since B is strongly connected it admits both an out-arborescence and an in-arborescence, rooted at the same arbitrary vertex $v^* \in V(B)$. Classify the vertices v_i of D into three types, according to the initial values of $w^-(v_i)$ and $w^+(v_i)$. Say that a vertex v is *in-deficient* when $w^-(v) < w^+(v)$, *out-deficient* when $w^+(v) < w^-(v)$, and *balanced* when $w^-(v) = w^+(v)$.

Consider the computation of *OUT PROCEDURE* and examine the vertices of the chosen out-arborescence T of B, in the ordering chosen by the algorithm. Let $v \neq v^*$ be the vertex next to be visited, in the considered ordering. If v is in-deficient, then the algorithm chooses the edge e_j of T ending at v and modify its weight from w_j to the value $w_j + w^+(v) - w^-(v)$. It is clear that v becomes balanced. Moreover, since T contains exactly one edge e_j entering v, we can assure that v remains balanced until the end of *OUT PROCEDURE.* Consequently, B does not contain in-deficient vertices at the completion of the procedure, except possibly v^*. Similarly, after the completion of *IN PROCEDURE*, B does not contain out-deficient vertices, except possibly v^*. Furthermore, the computation of *IN PROCEDURE* cannot create in-deficient vertices. Consequently, after completing the visits to the second tree, we can assure that all vertices are balanced, except possibly v^*. However, $w_j = 0$, for every bridge e_j of D. Consequently, $\sum_{v \in V(B)} w^-(v) = \sum_{v \in V(B)} w^+(v) = \sum_{e_j \in E(B)} w_j$. That is, $w^-(v^*) = \sum_{v \in V(B)} w^-(v) - \sum_{v \neq v^*} w^-(v) = \sum_{v \in V(B)} w^+(v) - \sum_{v \neq v^*} w^+(v) = w^+(v^*)$, meaning that v^* is automatically balanced. Finally, the vertices of B' outside B are clearly balanced, because all edges entering and leaving them have flow value zero. Therefore W is indeed a feasible circulation. ∎

It is simple to determine the complexity of the algorithm. Finding strongly connected components can be done in $O(n + m)$ time [11], so as for the initial flow value assignments. The computation of *OUT PROCEDURE* and *IN PROCEDURE* require $O(n)$ time. Consequently, the overall time is $O(n + m)$.

Finally, observe that the flow values of the feasible circulation obtained by the algorithm are all integers of size $w_j < 2 \sum_{e_j \in E(D)} \lceil b_j \rceil$.

4. Constructing normal models.

Let G be a PCA graph and $(C, \mathcal{A})$ a PCA model of it. We describe an algorithm for transforming $(C, \mathcal{A})$ into a normal model. The algorithm is based on [2, 4, 13]. For each $A_i \in \mathcal{A}$, the idea is to traverse C, searching for an arc A_j of $\mathcal{A}$ which together with A_i would cover C. If such an arc is found, then the arc A_j is conveniently shrunk with the purpose that A_i, A_j would no longer cover C. Clearly, it has to be assured that the new model so obtained is still a PCA model for G. Recall from Theorem 1 that any PCA graph admits a normal model.

The proposed method considers the arcs A_i in the circular ordering $A_1, \ldots, A_n$. For each i, only a fraction of the arcs of $\mathcal{A}$ would be examined in general, with the above goal of searching for an arc A_j, which together with A_i, cover C. The algorithm consists of the computation of a procedure $NORMAL(i)$, for $i = 1, \ldots, n$. Each computation $NORMAL(i)$ examines the extreme points $s'_i, \ldots, t'_i \in A_i \cup \{s_i, t_i\}$, consecutive in the circular ordering, where s'_i is the *first point* and t'_i the *last point* of $NORMAL(i)$, respectively. Each iteration inside $NORMAL(i)$ considers one of the extreme points $s'_i, \ldots, t'_i$, in circular ordering. Let p be the current point under examination by $NORMAL(i)$. Clearly, p is either a start point or an end point. The following is the action taken, according to these alternatives. If p is a start point, then nothing is done and the examination of p is terminated. Otherwise, $p = t_j$ for some j, and check whether A_i and A_j cover C. In the affirmative case, shrink A_j in such a way that A_i and A_j would no longer cover C, and terminate the examination of p. The actual shrinking operation consists of moving t_j counterclockwise in C, so that it becomes an interior point of the segment $(PRED(s_i), s_i)$. When A_i and A_j do not cover C, then terminate the computation $NORMAL(i)$, disregarding all possible extreme points of $(C, \mathcal{A})$ contained in A_i which lie after $p = t_j$, when traversing A_i in the circular ordering. Consequently, such an extreme point is the last point t'_i of $NORMAL(i)$. The first point s'_i of $NORMAL(i)$ is determined at the moment $NORMAL(i)$ starts, as follows. When $i = 1$, define $s'_1 = s_1$. For $i > 1$, let $s'_i = \max\{s_i, t'_{i-1}\}_{s_{i-1}}$. Similarly as above, the algorithm disregards all extreme points contained in A_i, which lie before s'_i in A_i.

The procedure is formulated below. If p and q are points of C, then denote by $POINT(p, q)$ an arbitrarily chosen point of the (open) arc (p, q). The external calls are $NORMAL(i)$, for $i = 1, \ldots, n$.

```
procedure $NORMAL(i)$
        if $i = 1$ then $p := s_1$ else $p := \max\{p, s_i\}_{s_{i-1}}$
        repeat
                if $p$ is an end point $t_j$ then
                   if $A_i \cup A_j = C$ then $t_j := POINT(PRED(s_i), s_i)$
                   else return
                $p := SUC(p)$
```

The following theorem assures that $NORMAL(i)$ terminates, and $(C, \mathcal{A})$ is a normal model after the computation of $NORMAL(n)$.

THEOREM 8. *The algorithm is correct.*

Proof. The algorithm consists of the computation of $NORMAL(i)$, for $i = 1, \ldots, n$. We prove that after completing $NORMAL(i)$, no two arcs A_k and A_j cover C, with $1 \leq k \leq i$. Furthermore, we ought to prove that the shrinking operations maintain $(C, \mathcal{A})$ as a PCA model for G.

Clearly, if there is an arc A_j such that A_i and A_j cover C, then $s_j, t_j \in A_i$, and t_j precedes s_j in A_i. Suppose the point $p = t_j$ is reached during $NORMAL(i)$. Then $A_i \cup A_j = C$ implies that t_j will be moved to $POINT(PRED(s_i), s_i)$. Let A^*_j be the new arc A_j, after shrinking. We know that A_i, A^*_j no longer cover C. We show that the configuration of $(C, \mathcal{A})$ still reflects the adjacencies of G. Clearly, A_i and A^*_j still intersect, otherwise A_i, A_j would not cover C. We discuss the intersections between A^*_j and any other arc $A_k \neq A_i$. We can restrict to arcs A_k which intersect A_i and have an extreme point, s_k or t_k, inside (s_i, t_j), since the intersections of A^*_j and the remaining arcs are not affected by the shrinking of A_j. The following are the possible alternatives.

Case 1. $s_k \in (s_i, t_j)$. If $t_k \in (s_i, t_i)$, then t_k precedes s_k in A_i, otherwise $(C, \mathcal{A})$ is not a PCA model. Consequently, A_k and A_j^* also intersect. When $t_k \notin (s_i, t_i)$ it follows that A_j^* and A_k again intersect, because $s_j \in (s_i, t_i)$.

Case 2. $s_k \notin (s_i, t_j)$. By the initial assumption, $t_k \in (s_i, t_j)$. Examine the possibilities for s_k. The situation $s_k \in (s_i, t_j)$ is in Case 1. Let $s_k \notin (s_i, t_j)$. Then $s_k \in (t_j, s_j)$, otherwise A_j contains A_k. Consequently, A_j^* and A_k again intersect.

Consequently, shrinking A_j in the manner $NORMAL(i)$ does actually preserve adjacencies. In what follows, we show that the modified model is still a PCA model. It is sufficient to prove that A_j^* cannot be contained in any other arc of $\mathcal{A}$, except for A_j which has been removed. Suppose the contrary, that is, A_j^* is contained in some arc A_k. In this case, the circular ordering of the considered extreme points must be $t_j^*, s_i, t_k, t_j, s_k, s_j, t_i$; otherwise A_k also contains A_j, or A_k does not contain A_j^*. Consequently, $A_k \cup A_i = C$. Since t_k precedes t_j in A_i, it follows that A_k would have been shrunk before A_j in $NORMAL(i)$. The latter contradicts A_k to contain A_j^*.

It remains to show that any arc A_k, which together with A_i covers C, is detected by the algorithm. Suppose the contrary. That is, $NORMAL(i)$ terminates and does not shrink A_k. Then $t_k \notin (s'_i, t'_i)$. Because $t_k \in (s_i, t_i)$, there are two possibilities.

Case 1. $t_k \in (s_i, s'_i)$. Then $s_i \neq s'_i$, which implies $i > 1$ and $t'_{i-1} \in A_i$. We also know that $s'_i \equiv t'_{i-1}$ is an end point t_j satisfying $A_{i-1} \cup A_j \neq C$. Observe that A_{i-1} and A_j may (or may not) coincide. Because $(C, \mathcal{A})$ is a PCA model, the sequence $s_j, s_{i-1}, s_i, t_k, t_j, t_{i-1}, t_i$ is in the circular ordering. Try to locate s_k. Because $A_i \cup A_k = C$, $s_k \in (t_k, t_i)$. However, $s_k \notin (t_k, t_{i-1})$ because A_k and A_{i-1} would also cover C, implying that $NORMAL(i-1)$ would also have moved t_k from the considered position, by an inductive argument. On the other hand, $s_k \notin (t_{i-1}, t_i)$ because in this case $A_{i-1} \cup A_k \neq C$, implying that t_k would have been reached by $NORMAL(i-1)$ before t_j. In the latter situation, $NORMAL(i-1)$ would have terminated at t_k, contradicting the assumption.

Case 2. $t_k \in (t'_i, t_i)$. We know there is an arc A_j, such that $t_j = t'_i$ and $A_i \cup A_j \neq C$. Because $A_i \cup A_k = C$, it follows that $s_k \in (t_k, t_i)$. In this situation, A_k contains A_j, contradicting $(C, \mathcal{A})$ to be a PCA model. Hence, Case 2 also does not occur.

Therefore every arc, which together with A_i covers C, is shrunk by the algorithm, completing the proof of correctness. ☐

THEOREM 9. *The algorithm terminates within $O(n)$ time. Moreover, there are less than $6n$ iterations of the* repeat *block during the entire process.*

Proof. The number of steps performed by the algorithm corresponds to the number of iterations of the *repeat* block of the described formulation. Such a number is equivalent to the number of assignments of extreme points to p during the entire process. We will prove that the total number of such assignments is less than $6n$. The first value of p is s_1 and the last one, in the worst case, is t_n. We know that p never moves counterclockwise in C. First, assume for a while that p does not stay stationary in two consecutive iterations, and also disregard the shrinking of arcs. Any of the $2n$ extreme points can be assigned to p. If $t_n \notin A_1$, then the assignments would correspond at most to a complete turn around the circle, because the model is PCA. That is, $2n$ assignments. On the other hand, if $t_n \in A_1$, then any extreme lying between s_1 and t_n may be assigned again to p, but at most once more, besides its assignment in $NORMAL(1)$. This corresponds to an additional n units in our account.

Next, examine the case when p stays stationary in two consecutive iterations. Such a situation may only occur, possibly at the first iteration within $NORMAL(i)$, when $s'_i = t'_{i-1}$, for $i > 1$. Overall, it counts to additional $n-1$ units, at most, in the total number of assignments to p.

Finally, examine the shrinking operations. Each time an arc A_j is shrunk, an extreme point t^*_j is placed in a different position of $(C, \mathcal{A})$, offering the possibility to t^*_j itself to be assigned to p. Hence, the shrinking operations may contribute to our account. In fact, if an arc gets shrunk k times during the algorithm, this may represent additional k assigments to p, in the worst case. Furthermore, there might be $O(n)$ distinct arcs, each of them covering C, together with A_j. However, the next assertion assures that even in this case, the number of times A_j gets shrunk is low.

Claim. Any arc of $(C, \mathcal{A})$ may be shrunk at most twice during the entire algorithm.

Proof. Let A_j be an arc of $(C, \mathcal{A})$. Clearly, if A_j together with some other arc does not cover C, then A_j is not shrunk at all. Otherwise, consider the model $(C, \mathcal{A})$ at the beginning of the computation $NORMAL(k)$, where A_j gets shrunk for the first time, if any. Denote by $A_{j_1}, \ldots, A_{j_l}$ the arcs of $(C, \mathcal{A})$, each of them covers C, together with A_j. This implies that all extreme points of the arcs $A_{j_1}, \ldots, A_{j_l}$ belong to (s_j, t_j). Furthermore, s_{j_k} precedes t_{j_k} in A_j, $1 \le k \le l$. Without loss of generality, the starting points $s_{j_1}, \ldots, s_{j_l}$ are in the circular ordering. Clearly, the arc of least index among $A_j, A_{j_1}, \ldots, A_{j_l}$ corresponds to one of this collection which is first handled by the algorithm. Let $i = \min\{j, j_1, \ldots, j_l\}$. The following are the alternatives for i:

Case 1. $i = j$. We know that $NORMAL(j)$ does not shrink A_j. Furthermore, Theorem 8 implies that $NORMAL(j)$ would shrink all arcs $A_{j_1}, \ldots, A_{j_l}$. Consequently, A_j never gets shrunk.

Case 2. $i = j_1$. Because A_j and A_{j_1} cover C, during $NORMAL(j_1)$ the assignment $p := t_j$ necessarily occurs, since Theorem 8 assures that $NORMAL(j_1)$ shrinks A_j. Then t_j moves to the position t^*_j, becoming the predecessor of s_1. The arc A^*_j so obtained does not intersect any of $A_{j_1}, \ldots, A_{j_l}$, and consequently, it cannot get shrunk. Consequently, A_j is shrunk only once during the process.

Case 3. $i = j_k$, for some $1 < k \le l$. Then $NORMAL(j_k)$ moves t_j to t^*_j, where t^*_j, this time, is the new predecessor of s_{j_k}. Following the circular ordering of $(C, \mathcal{A})$, the first subsequent computation which can possibly shrink A_j is $NORMAL(j_1)$. By Case 2, the latter would shrink A_j, exactly once more. Consequently, A_j gets shrunk twice.

The conclusion is that any arc may get shrunk at most twice during the process. $\square$

Consequently, the shrinking operations contribute at most $2n$ units to our account. Therefore the algorithm performs less than $6n$ iterations of the *repeat* block. Each of these iterations requires no more than constant time. That is, the overall complexity is $O(n)$. $\square$

5. Recognition and model construction. In this section, we describe the algorithms for recognizing UCA graphs and for constructing UCA models. Also, we determine the size of the obtained model by showing that its extremes correspond to integers of size $O(n)$.

The algorithm is as follows. Let G be the given graph.

1. Verify if G is a PCA graph, using the algorithm [1]. In the affirmative case, let $(C, \mathcal{A})$ be the PCA model so obtained. In the negative case, G is not UCA.
2. Transform $(C, \mathcal{A})$ into a normal model $(C', \mathcal{A}')$, using the algorithm of section 4.
3. Construct the segment digraph D of $(C', \mathcal{A}')$. Find the connected components of D. Verify if all connected components are strongly connected. In the negative case, report that G is not UCA and stop. Otherwise, report that G is UCA and construct a UCA model $(C'', \mathcal{A}'')$ of it by running Step 4.
4. Construct the segment network of $(C', \mathcal{A}')$ by assigning to each edge $e_j \in E(D)$ a unit lower capacity and an unbounded upper capacity. Then find the flow values of a feasible circulation W of D, using the algorithm of section 3. Let $p'_1, \ldots, p'_{2n}$ be the extremes of $(C', \mathcal{A}')$, in circular ordering, with p_1 corresponding to the start point of the first arc of the model. Then construct $(C'', \mathcal{A}'')$ as follows. The length of C'' is $\sum_{e_j \in E(D)} w_j$. The extremes $p''_1, \ldots, p''_{2n}$ of $(C'', \mathcal{A}'')$ are defined by the following conditions: p''_1 is an arbitrary point of C'', and the segment (p''_j, p''_{j+1}) has length w_j, $1 \leq j < 2n$.

The correctness follows basically from Theorems 4, 7, and 8, and Corollary 6. We evaluate the complexity of the recognition algorithm. Step 1 requires $O(n+m)$ time [1]. Step 2 takes $O(n)$ time by Theorem 9. As for Step 3, note that for constructing the segment digraph, we need to construct the full and reduced systems of $(C', \mathcal{A}')$. The first of them is a family of $n-1$ equations, whose variables are segment lengths. In each side of the equations, the segment lengths are consecutive. Consequently, we need $4n-4$ indices to represent these equations, overall. On the other hand, in the reduced system, each segment length appears at most twice. That is, it requires at most $4n$ indices. The segment digraph has n vertices and at most $2n$ edges. There is no difficulty to construct D in $O(n)$ time. The connected and strongly connected components can be constructed in linear time [11]. Step 4 also requires linear time in the size of D, according to section 3. Consequently, the overall complexity of algorithm is $O(n+m)$. Furthermore, Steps 2, 3, and 4 require $O(n)$ time, meaning that the complexity of the algorithm reduces to $O(n)$, if the input is a PCA model of G, with ordered extremes of the arcs.

Finally, we verify the size of the UCA model obtained by the algorithm.

Corollary 10. *Let $(C, \mathcal{A})$ be a UCA model constructed by the algorithm. Then the extremes of $(C, \mathcal{A})$ correspond to integers of size $< 4n$.*

Proof. Let $p_1, \ldots, p_{2n}$ be the extremes of $(C, \mathcal{A})$ in circular ordering. We associate the segment length l_j to the extreme p_{j+1}, $1 \leq j < 2n$. From Step 4 of the algorithm, we know that the length l_j of each segment of $(C, \mathcal{A})$ equals the flow value w_j of the corresponding edge e_j of D. On the other hand, we know from section 3 that $w_j < 2m$. Since $m < 2n$, it follows that $l_j < 4n$. □

6. Conclusion.

The proposed algorithms for recognizing UCA graphs and constructing UCA models terminate within $O(n+m)$ time. Furthermore, the complexity reduces to $O(n)$ when the input is a PCA model. It should be noted that the algorithms require the extreme points of the given model to be ordered. If the input is a graph given by its vertices and edges, then there is a preprocessing step, which consists of running the algorithm by Deng, Hell, and Huang [1], for constructing a PCA model of the graph. The algorithm [1] can be implemented so as to construct a PCA model with ordered extreme points within $O(n+m)$ time. On the other hand, if the input is already a PCA model and its extreme points are not ordered, then we need an additional $O(n \log n)$ steps for ordering.

The UCA model constructed by the proposed algorithm is such that its extreme points correspond to integers of size $O(n)$. Regarding this feature, we leave the four related questions below.

Given a UCA graph G, find a UCA model whose extremes of the arcs correspond to integers, satisfying

(i) the maximum segment length is minimized.

(ii) the circle length is minimized.

More generally, for given n, find the least l, such that any UCA graph with n vertices admits a UCA model whose extremes of the arcs correspond to integers, such that

(iii) the maximum segment length is $\leq l$.

(iv) the circle length is $\leq l$.

Corollary 10 implies that $l < 4n$ for Question (iii), and that $l < 4n^2$ for Question (iv).

Acknowledgments. We are grateful to Professor Pavol Hell and an anonymous referee for their most valuable suggestions and comments, which lead to improvements in different parts of this paper.

REFERENCES

[1] X. Deng, P. Hell, and J. Huang, *Linear time representation algorithms for proper circular-arc graphs and proper interval graphs*, SIAM J. Comput., 25 (1996), pp. 390–403.

[2] G. Durán, A. Gravano, R. M. McConnell, J. P. Spinrad, and A. Tucker, *Polynomial time recognition of unit circular-arc graphs*, J. Algorithms, 58 (2006), pp. 67–78.

[3] E. Eschen and J. P. Spinrad, *An $O(n^2)$ algorithm for circular-arc graph recognition*, in Proceedings of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms, Austin, TX, ACM, New York, SIAM, Philadelphia, 1993, pp. 128–137.

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

[5] A. J. Hoffman, *Some recent applications of the theory of linear inequalities to extremal combinatorial analysis*, in Proceedings of the Symposia in Applied Mathematics, American Mathematical Society, Providence, RI, 1960, pp. 113–117.

[6] W. L. Hsu, *$O(mn)$ algorithms for the recognition and isomorphism problems on circular-arc graphs*, SIAM J. Comput., 24 (1995), pp. 411–439.

[7] M. C. Lin and J. L. Szwarcfiter, *Efficient construction of unit circular-arc models*, in Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms, Miami, FL, 2006, pp. 309–315.

[8] R. M. McConnell, *Linear-time recognition of circular-arc graphs*, Algorithmica, 37 (2003), pp. 93–147.

[9] A. Schrijver, *Combinatorial Optimization, Vol A: Polyhedra and Efficiency*, Springer-Verlag, Berlin, 2003.

[10] J. P. Spinrad, *Efficient Graph Representations*, Fields Institute Monographs 19, AMS, Providence, RI, 2003.

[11] R. E. Tarjan, *Depth-first search and linear graph algorithms*, SIAM J. Comput., 1 (1972), pp. 146–160.

[12] A. Tucker, *Characterizing circular-arc graphs*, Bull. Amer. Math. Soc., 76 (1970), pp. 1257–1260.

[13] A. Tucker, *Structure theorems for some circular-arc graphs*, Discrete Math., 7 (1974), pp. 167–195.

[14] A. Tucker, *An efficient test for circular-arc graphs*, SIAM J. Comput., 9 (1980), pp. 1–24.