Journal of Physical Sciences, Vol. 10, 2006, 201 – 217

A Sequential Algorithm to Solve Next-to-Shortest Path Problem on Circular-arc Graphs

Swagata Mandal and Madhumangal Pal*
Department of Applied Mathematics with Oceanology and Computer Programming, Vidyasagar University, Midnapore-721 102, India

Received 20 October, 2006 ; accepted 15 November, 2006

ABSTRACT

In this article, we study the problem of finding the next-to-shortest path in circular-arc graph. A next-to-shortest path between any pair of vertices in a shortest path amongst all paths between those vertices with length strictly greater than the length of the shortest path. The next-to-shortest path problem in a directed graph in NP-hard. Here we deigned a polynomial time algorithm to solve this problem for the circular-arc graph.

Keywords : *Design and analysis of algorithms, Shortest paths, Next shortest paths, Circular-arc graph.*

1. Introduction

1.1 Problem under consideration

A graph $G = (V, E)$ is called an intersection graph for a finite family F of a non empty set if there is a one-to-one correspondence between F and V such that two sets in F have non empty intersection if and only if their corresponding vertices in V are adjacent to each other. F is called an intersection model of G and G is called the intersection graph of F. If F is a family of arcs around a circle, then G is called an interval graph. V is the set of all vertices and E is the set of all edges of the graph G.

Circular-arc graph is a general form of interval graph [4, 11] and it is one of the most useful discrete mathematical structure for modelling problems arising in the real world. It has many applications in genetics, traffic control, cyclic scheduling and computer compiler design.

Turker [15] has proposed $O(n^3)$ time algorithm for recognizing a circular-arc graph and constructing in the affirmative case, a circular arc model. Hsu [5] has designed an

$O(nm)$ time algorithm for this problem. Eschen and Spinrad [3] have presented an $O(n^2)$ time algorithm for recognizing a circular-arc graph.

A *walk* is an ordered list of vertices and edges $v_0, e_1, v_1, e_2, v_2, \ldots e_k, v_k$ such that for $1 \leqslant i \leqslant k$, the endpoints of e_i and v_{i-1} and v_i. Depending on the context we may just specify a walk by giving either an ordered list of vertices or of edges. A *path* is a walk for which the vertices $v_0, e_1, v_1, e_2, v_2, \ldots, e_k, v_k$ are distinct. The *length* of a path is the number of edges in the path. A path from the vertex i to the vertex j is a *shortest path* if there is no other path from i to j with lower length. The shortest distance (i.e., the length of the shortest path) between the vertices i and j is denoted by $d(i,j)$. The next-to-shortest path from the vertex i to the vertex j is the shortest distance from the verted i to the verted j amongst those the distances strictly greater than the shortest distance $d(i, j)$. If no such path exist, we say that the distance of next-to shortest path is ∞. The next-to-shortest distance (i.e., the length of the next-to-shortest path) between the vertices i and j is denoted by $nd(i, j)$.

1.2. Survey of the previous work

Finding the k-shortest paths between two vertices has also been well-studied, in for instance [2, 6]. Problems involving finding a path between two vertices of length strictly greater than the length of the shortest such path have received much less attention due to the fact that in directed graphs, when we allow edges of length zero, the problem has been show to be NP-hard [8]. Krasikow and Noble [7] have solved the problem to find next-to-shortest path in a weighted graph in $O(n^3m)$ time, where m is the number of edges and n is the number of vertices. In [14], Seidel has given an $O(M(n)\log n)$ time sequential algorithm for all pair shortest path (APSP) problem for an undirected un-weighted arbitrary graphs with n vertices, $M(n)$ being the time necessary to multiply two $n \times n$ matrices of small integers, the best known time for $M(n)$ is of $O(n^{2.376})$. Atallah et al. [1] solved the single source shortest path problem on the weighted interval and circular-arc graphs in $O(n)$ time, if the intervals / circular arcs are given sorted and hence also solved the weighted all-pair shortest path problem in the optimal $O(n^2)$ time and $O(n^2)$ space. Pal and Bhattacharjee [12] have designed a parallel algorithm to solve APSP problem on interval graphs using $O(n^2/p + \log n)$ time and p processors on an EPEW PRAM. Mondal et al. have soloved APSP problem on permutation graph [10] and on trapezoid graphs [9] in $O(n^2)$ time. Saha et al. [13] have proposed $O(n^2)$ time algorithm for solving all-pairs shortest paths problem on circular-arc graphs.

1.3. Our result

In this paper, we designed an algorithm to find a next-to-shortest path between two vertices of a circular-arc graph. The proposed algorithm runs in $O(n^2)$ time.

2 Preliminaries

Let $A = \{A_1, A_2, \ldots, A_n\}$ be the circular arc family of a circular-arc graph $G = (V, E)$. The

family of circular arcs are located around a circle C. While going in a clockwise direction, the point at which we first encounter an arc will be called the *starting point* of the arc. Similarly, the point at which we leave an arc will be called the *starting point* of all arc. Similarly, the point at which we leave an arc will be called the *finishing point* of the arc. Every arc can be represented by their two endpoints e.g., A_i can be represented as $[s_i, f_i]$, wherre s_i is the starting point and f_i is the finishing point of the arc A_i on the circle C. Each endpoint of an arc is assigned to a positive integer called a *coordinate.* A ray is a straight line from the centre of C passing through any coordinate.

We consider a ray through straiting point of any arc. We label this arc by 1, then we start clockwise traversal from the ray. We label 2 to the next successive starting point. In this process, we label all the remaining arcs.

Without loss of generality, we assume the following :

1. An arc contains both its end points and that no two arcs share a common end point.
2. The graph G is connected and the list of sorted endpoints are given.
3. No single arc in A cover the entire circle C.
4. Arcs and vertices of a circular-arc graph are same thing.
5. The endpoints of the arcs in A are sorted according to the order in which they are visited during the anticlockwise traversal along circle by starting at an arbitrary arc called A_n.
6. The arcs are sorted in decreasing values of f_i's i.e., $f_i < f_j$ for $i < j$.
7. $U_{i=1}^{n} A_i = C$ (otherwise, the problem becomes one on interval graph).

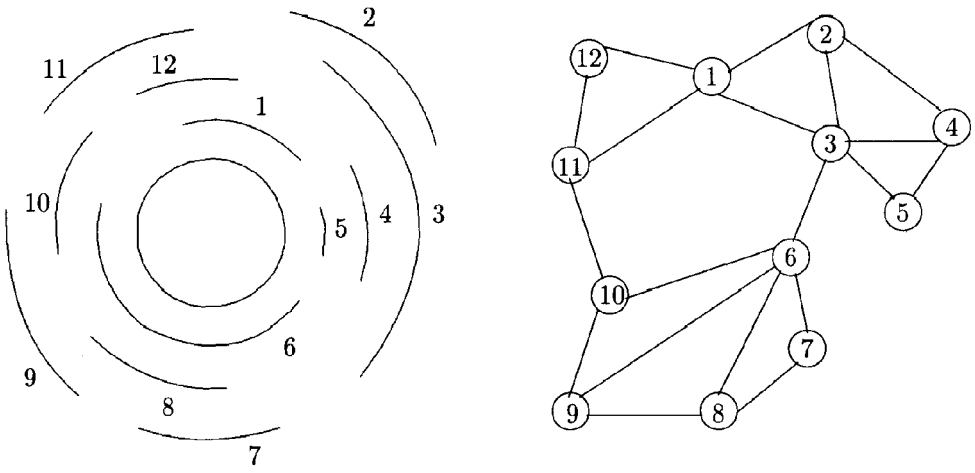

Figure 1 : Example of a circular-arc graph and its circular arc representation

The family of arcs A is said to be canonical if

(i) s_i's and f_i's for all i = 1, 2, ..., n are distinct integers between 1 and 2n, and

(ii) the endpont 2n is the finishing end point of the arc A_n.

If A is not canonical, using sorting one can construct a canonical family of arcs in $O(n \log n)$ time.

To find the next-to-shortest path of a circular-arc graph a main cycle is constructed from the set of arcs. The method to find such cycle is described in the next section.

3. Method to Find a Main Cycle

3.1. Definition of main cycle

A cycle of a circular-arc graph is the set of intersecting arcs those cover the whole circle C. That is, if $A_1, A_2 \ldots, A_r$ be a set of arcs of a cycle then $A_i \bigcap A_{i+1} \neq \phi, A_r \bigcap A_1 \neq \phi$ and $U_{i=1}^{r} A_i = C$. The *main cycle* is a cycle whose cardinality is minimum among all cycles. The main cycle is denoted by M'. Let M be the set of vertices corresponding to the arcs of M'. The set M is also regarded as the main cycle. The length of a cycle C is the number of arcs on the cycle C and it is denoted by $len(C)$. The length of the main cycle is the minimum among all other cycles. A circular-arc graph may have more that one main cycles, but their lengths are same. If a graph has multiple cycles then any one of them is taken as main cycle.

A method to find a main cycle is describe below :

First we draw a ray through the finishing point of any arc of A. Then, we consider the arcs which are intersected by this ray. Then we determine the arc which has right most finishing point. Thus is, the first verted of the set M. Again, we draw a ray from the centre and through the finishing point of the first vertex of M. Consider the arcs which are intersected by the second ray. Then we find the arc which has right most finishing point among the arcs which are intersected by second ray. Let the vertex corresponding to this arc be the second vertex of M. This process terminates when any vertex of M is repeated. Finally, the duplicate vertices are to be removed from M.

3.2. Algorithm and complexity to find the main cycle

Here, we present an algorithm which generates a main cycle, even the graph contains more than one main cycles.

ALGORITHM MC

Input	**:**	A set of arcs A of a circular-arc graph G.
Output	**:**	A set of vertices M which form a main cycle.
Step 1	**:**	Set M = ϕ. Choose any arc A_i from A.
Step 2	**:**	Draw a ray through the finishing point of the arc A_i.
Step 3	**:**	Consider the arcs which are intersected by the ray drawn in Step 2, and let these set of arcs be B.
Step 4	**:**	Find the arc which has right most finishing point of the arcs of B. Let this arc be A_i.

Step 5 : Set $M = M \cup \{i\}$.
Step 6 : Repeat Step 2 to Step 5 until any vertex of M is repeated.
Step 7 : Delete the repeated vertices from M.
END MC.

In this algorighm, the endpoints of each arc are consider to select the members of M. The total number of arcs of a circular-arc graph is taken as n. The time complexity of Algorithm MC is stated below.

Theorem 1. *A main cycle M of a circular-arc graph with n vertices can be computed in O(n) time.*

Throughout the paper, we mark the vertices of main cycle M by using the symbol of asterisk i.e., $v_1^*, v_2^*, ..., v_l^*, u^*, v^*$ etc. are the vertices of M. Let D be the set of the vertices which are not in M i.e., $D = V \backslash M$. The unmarked (by asterisk) vertices are taken as the vertices of D.

Lemma 1. *The Algorithm MC correctly computes the main cycle.*
Proof : Any vertex of M is adjacent with its next and previous vertices. The process of finding the vertices of M is terminate when any vertex of M is repeated. Let v_i^* be repeated. The vertex v_i^* and the next vertex of v_i^* for the first time are adjacent. Similarly, for second time v_i^* and the previous vertex of v_i^* are adjacent. So, if we delete the vertices from first vertex to the verted v_i^* (first occurance) from M, then any two consecutive vertices of the remaining vertices of M are adjacent. Also, any vertex is repeated in clockwise traversal, so the vertices of the remaining vertices of M are adjacent. Also, any vertex is repeated in clockwise traversal, so the vertices of M covers the whole circle i.e., vertices of M form a cycle.

If the length of the cycle is minimum the cycle becomes main cycle. If the removal of any vertex from M makes another cycle by the remaining vertices, then the cycle constructed by the Algorithm MC do not cover the circle C. Let $v_i^*, v_{i+1}^*, v_{i+2}^*$ be three consecutive vertices. So, v_i^* and v_{i+1}^* are adjacent, also v_{i+1}^* and v_{i+2}^* are adjacent. If v_i^*, v_{i+2}^* are adjacent then v_{i+2}^* must cover the finishing point of v_i^*. If v_{i+1}^* is the next vertex of v_i^*, then finishing point of v_{i+2}^* is less than finishing point of v_{i+1}^*. But it is impossible, because v_{i+2}^* is next vertex of v_{i+1}^*. So, many two non-consecutive vertices are not adjacent. Therefore, if we delete any vertex from M then vertices of M cannot form a cycle.

3.3. Properties of main cycle

A circular-arc graph may contain more than one main cycle. But, the length of all main cycles are equal whatever may be the starting arc. The Algorithm MC generates only one main cycle.

Lemma 2. *If* $u \in D$ *then u is adjacent to at least one vertex of M.*

Proof : By the definition of main cycle, the vertices of *M* form a cycle and cover the whole circle *C*. Since the graph is a circular-are graph, any are corresponding to the vertex u Î D must lie over the circle (see Figure 2). But, the vertices of M cover the whole circle. So, the arc corresponding to the vertex *u* must has non empty intersection with at least one arc corresponding to a vertex of *M.* Therefore, *u* is adjacent to at least one vertex of *M.*

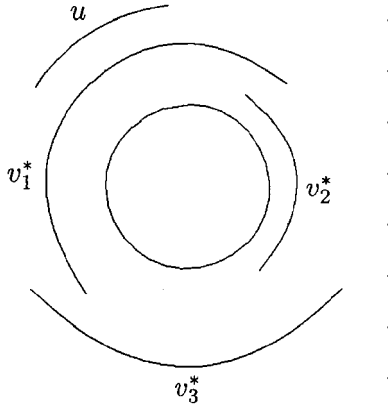

Figure 2 : Illustration of Lemma 2

Lemma 3. *Any vertex of D cannot be adjacent with more than three vertices of main cycle.*

Proof : Let A_k be the arc corresponding to the vertex n_k Î D. If possible, let the arc A_k intersect with four arcs $A_i^*, A_{i+1}^*, A_{i+2}^*, A_{i+3}^*$ (see Figure 3). Then the arc A_k covers, at least, the finishing point of the arc A_i^* and starting point of A_{i+3}^* is less than the finishing point of A_k *i.e.,* $s_{i+3}^* < f_k$. By definition, it is easy to see that A_{i+1}^*, A_{i+3}^* are non intersecting arcs. So, finishing point of A_{i+1}^* is less than starting point of A_{i+3}^* i.e., $f_{i+1}^* < s_{i+3}^*$. Therefore $f_{i+1}^* < f_k$. Both the arcs A_{i+1}^*, A_k cover the finishing point of A_i^* and finishing point of A_{i+1}^* is less than finishing point A_k. Thus the arc A_k cannot interesect with four arcs $A_i^*, A_{i+1}^*, A_{i+2}^*, A_{i+3}^*$.

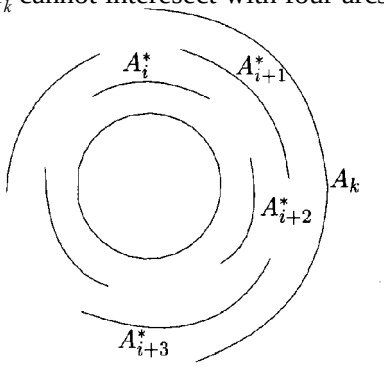

Figure 3 : Illustration of Lemma 3

Lemma 4. *If the vertex u Î D is adjacent to the non-consecutive vertices* ν^*_{i+2}*, then u is also adjacent to* A^*_{i+1}*, where* $\nu^*_i, \nu^*_{i+1}, \nu^*_{i+2}$ *are three consecutive vertices of M.*

Proof : If *u Î D* is adjacent to the vertices ν^*_i and ν^*_{i+2}, then arcs A_u and A^*_i have non-empty intersection and arcs A_u, A^*_{i+2} have non-empty intersection. Let the starting point of A_u be greater than the starting point of A^*_{i+2} [see Figure 4 (a)]. Also, the vertices $\nu^*_i, \nu^*_{i+1}, \nu^*_{i+2}$ of *M*, then the starting point of the arc A^*_{i+1} is less than finishing point of A^*_i and the finishing point of the arc A^*_{i+1} is greater than the starting point of the arc A^*_{i+2}. So, the arcs A_u and A^*_{i+1} must have non-empty intersection. The vertices *u* and ν^*_{i+1} are adjacent.

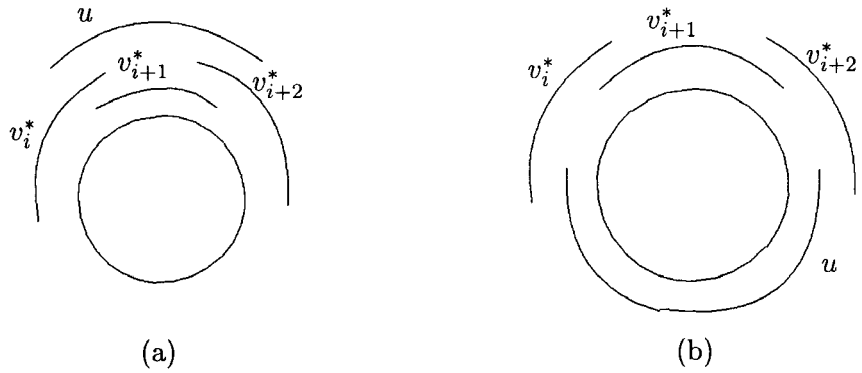

Figure 4 : (a) The possible case, and (b) the possible case of Lemma 4

Otherwise, is the starting point of arc A_u is less than finishing point of A^*_{i+2} and the finishing point of A_u is greater than the starting point of A^*_i [see Figure 4 (b)], then also the vertices *u,* ν^*_i are adjacent and *u,* ν^*_{i+2} are adjacent. There does not exist non-empty intersection of arcs A_u and A^*_{i+1}. But in this case, the vertex *u* becomes as a member of the main cycle. But we consider the vertex *u* and a mamber of the set *D.* Therefore, if *u* is a vertex of *D* and adjacent to the non-consectutive vertices ν^*_i and ν^*_{i+2} then *u* also adjacent to the vertex ν^*_{i+1}.

Lemma 5. *The length of a cycle other than main cycle is less than 4.*

Proof : If possible let $\nu^*_i \to u \to \nu^*_{i+2} \to \nu^*_{i+1} \to \nu^*_i$ be a cycle of length 4, where *u Î D* and $\nu^*_i, \nu^*_{i+1}, \nu^*_{i+2} \in M$. But by the Lemma 4, we know if *u* is adjacent to ν^*_i and ν^*_{i+2} then *u* is also adjacent to ν^*_{i+2}. The the cycle $\nu^*_i \to u \to \nu^*_{i+2} \to \nu^*_{i+1} \to \nu^*_i$ can be decomposed into two cycles $\nu^*_i \to u \to \nu^*_{i+1} \to \nu^*_i$ and $\nu^*_{i+1} \to u \to \nu^*_{i+2} \to \nu^*_{i+1}$ each of length 3.

Again, we let $\nu^*_i \to u \to \nu \to \nu^*_{i+1} \to \nu^*_i$ be a cycle of length 4, where *u, n Î D* and $\nu^*_i, \nu^*_{i+1} \in M$. If the vertices *u, n* are A_v be less than the finishing point of the arc A_v i.e., $s_v < f_u$. Then the portion $[s_v, f_u]$ must lies at least one of the arcs A_i and A^*_{i+1} or both the arcs

A_i and A^*_{i+1}. If the portion $[s_v, f_u]$ must lies at least one of the arcs A_i and A^*_{i+1} or both the arcs A_i and A^*_{i+1}. If the portion $[s_v, f_u]$ lies on the arc A^*_i then the vertices vertices ν^*_i and v are adjacent. If the portion $[s_v, f_u]$ lies on the arc A^*_{i+1} then the vertices ν^*_{i+1} and u are adjacent. Also if the portion $[s_v, f_u]$ lies on both the arcs A^*_i and A^*_{i+1} then vertices ν^*_{i+1} and u are adjacent and the vertices ν^*_i and v are adjacent. Therefore, if u and ν^*_{i+1} are adjacent then cycle $\nu^*_i \to u \to \nu \to \nu^*_{i+1} \to \nu^*_i$ is decomposed into two cycles $\nu^*_i \to u \to \nu^*_{i+1} \to \nu^*_i$ and $\nu^*_{i+1} \to \nu \to u \to \nu^*_{i+1}$ each of length 3. Also. if v is adjacent to ν^*_i then the cycle $\nu^*_i \to u \to \nu \to \nu^*_{i+1} \to \nu^*_i$ is the combination of cycles $\nu^*_i \to \nu \to \nu^*_{i+1} \to \nu^*_i$ and $\nu^*_i \to u \to \nu \to \nu^*_i$ each of length 3.

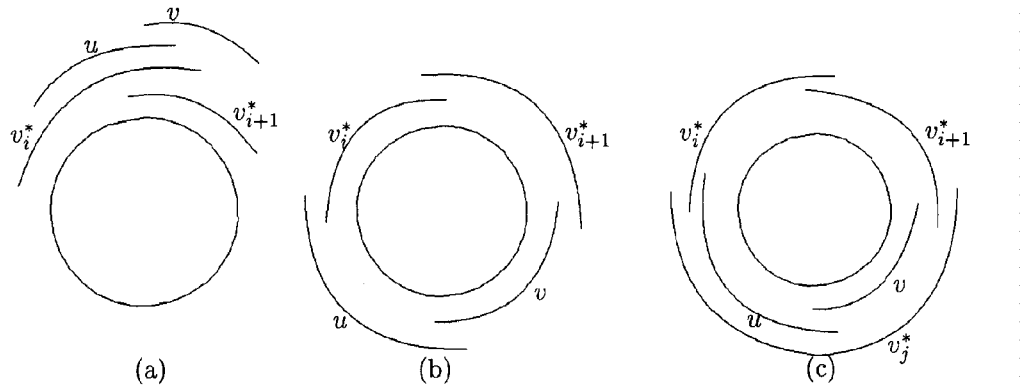

Figure 5 : (a) The possible cases of Lemma 5

Otherwise, if arcs A_u and A_v are connected in other endpoints i.e., starting point of A_u is less than finishing point of A_u [see Figure 5 (b)]. Then vertices u and v are adjacent but u may not be adjacent to ν^*_{i+1} and v may not be adjacent to ν^*_i. But, in this case vertices u and v becomes the member of main cycle.

Again, if there emist an arc A^*_j whose starting point is less than the finishing point of A^*_{i+1} and the finishing point is greater than the starting point of A^*_i [see Figure 5(c)] then $(\nu^*_i, \nu^*_j) \in E, (\nu^*_{i+1}, \nu^*_j) \in E$. Then u, v does not become the vertices of the main cycle. But, in this case, ν^*_j is the vertex of the main cycle and number of vertices of the main cycle is 3. So, the cycle $\nu^*_i \to u \to \nu \to \nu^*_{i+1} \to \nu^*_i$ is nothing but the combination of two cycles $\nu^*_i \to \nu^*_j \to \nu \to \nu^*_i$ and $\nu^*_{i+1} \to \nu \to \nu^*_j \to \nu^*_{i+1}$ each of length 3.

Therefore, the length of any cycle other than main cycle is always less than 4.

4. Next-to-Shortest Path

Computation of next-to-shortest path is mainly depends on all-pairs shortest paths on

circular arc graph. The all pairs shortest paths and their distances can be determined by using the algorithm of Saha et al. [13]. We assume that, all pairs shortest paths and distances are available. In the following, some results are presented which will help us to disign next-to-shortest path. In our proposed algorithm, the next-to-shortest path is determine in two stages. In first stage, we determine the next-to-shortest path between two vertices which belong to the main cycle *M.* In the next stage, we find the same for the vertices of *D.*

Throughout the paper, we denote a path between the vertices *u* and *v* of length more than one by the symbol $u \xrightarrow{*} v$.

If there is no alternating path between any two vertices *u* and *v* of length strictly greater than *d (u, v)* then we assumed that *nd (u, v)* = ∞.

The lemmas 6 and 7 give the value of next-t0-shortest distance between the vertices of *M.*

Lemman 6. *Let x be a vertex of D. If* $(v_i^*, v_{i+1}^*) \in E$ *and* $(v_{i+1}^*, x) \in E$ *then* $nd(v_i^*, v_{i+1}^*) = 2$.

Proof : Since v_i^* and v_{i+1}^* is $(v_i^*, v_{i+1}^*) \in E$. Thus the shortest path from v_i^* to v_{i+1}^* is $v_i^* \to v_{i+1}^*$ and hence $d(v_i^*, v_{i+1}^*) = 1$. If $(v_i^*, x) \in E$ and $(v_{i+1}^*, x) \in E$ for any vertex $x \hat{I} D$ then there exist a path $v_i^* \to x \to v_{i+1}^*$ from v_i^* to v_{i+1}^*. The length of this path is 2. This is greater than the shortest distance by 1. Hence the next-to-shortest distance from v_i^* to v_{i+1}^* is 2 i.e., and $nd(v_i^*, v_{i+1}^*) = 2$.

For the graph of the Figure 6, *nd(6,4) = 2.*

Lemma 7. *If* $(v_i^*, v_{i+1}^*) \in E$ *and there does not exist any vertex* $x \hat{I} D$ *such that* $(v_i^*, x) \in E$ *and* $(v_{i+1}^*, x) \in E$ *then* $nd(v_i^*, v_{i+1}^*) = |M| - 1$.

Proof : If v_i^*, v_{i+1}^* are two adjacent vertices in *M,* so the shortest path between v_{i+1}^* to v_{i+1}^* is $v_i^* \to v_{i+1}^*$. So, $d(v_i^*, v_{i+1}^*) = 1$. Since v_i^*, v_{i+1}^* are the vertices of *M* then there exist another path between v_i^* to v_{i+1}^* which is $v_i^* \to v_{i-1}^* \to v_{i-2}^* \xrightarrow{*} v_{i+2}^* \to v_{i+1}^*$. The length of this path is ($|M| - 1$). Let there exist a vertex $y \hat{I} D$ such that *y* is adjacent to another vertices v_{i-1}^* and v_{i-2}^* in *M* i.e, $(y, v_{1-1}^*) \in E$. Then there exist also a path between v_i^* to v_{i+1}^* which is $v_i^* \to v_{i-1}^* \to y \to v_{i-2}^* \to v_{i+2}^* \to v_{i+1}^*$. But the length of this path is $|M| - 1$. This is greater than ($|M| - 1$). By Lemma 5 there does not exist the path $v_i^* \to x \to y \to v_{i+1}^*$ for any two vertices *x, y* $\hat{I}$ *D.* Therefore, the next shortest distance between v_i^* and v_{i+1}^* is $|M| - 1$.

For the graph of the Figure 6, $nd(6,7) = |M| - 1 = 6 - 1 = 5$.

The next-to-shortest distance between two adjacent vertices vertices of *D* is obtain by the following lemmas.

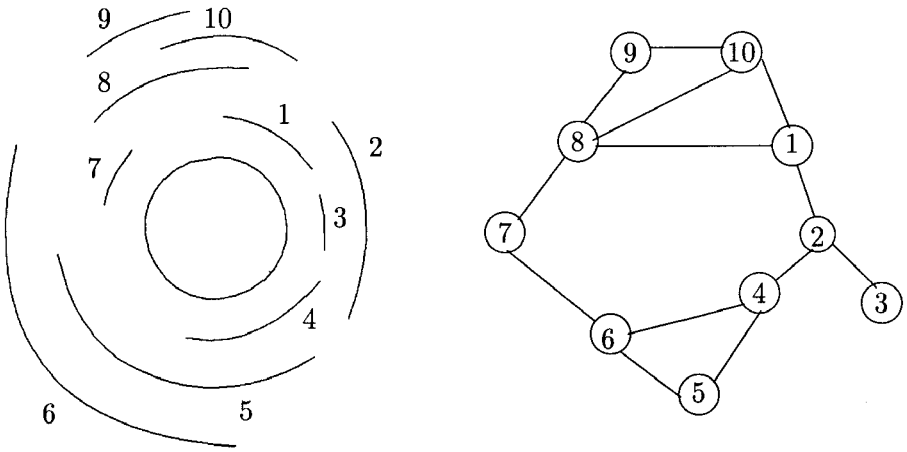

Figure 6 : Illustration of Lemmas 6, 8, 9

Lemma 8. *If (u,v) Î E where u, v Î D then nd (u,v)* = 2.

Proof : If *u* and *v* are adjacent then shortest path between them is *u* ® *v* and the shortest distance is 1. Let the starting point of the arc A_v corresponding to the vertex *v* is less than the finishing point arc A_u corresponding to the vertex *u*. Then the arc $[s_v, f_u]$ is over the arc A_i^* then *u* and *v* are both adjacent to the vertex v_i^*. So there exist the path *u* ® v_i^* ® *v* and the length of this path is 2. Therefore *nd(u,v)* = 2.

In the graph of the Figure 6, *nd* (9,10) = 2.

The next-to-shortest distance between two vertices does always not exist. This is proved in the following lemma.

Lemma 9. *If (u,* v_i^**) Î E where u Î D,* v_i^* *Î M and there does not exist any vertex x Î V such that (*v_i^**,x) Î E and (u,x) Î E then nd (u,v)* = ¥.

Proof : If there does mpt exist any vertex *x Î V* such that *(*v_i^**,x) Î E* and *(u,x) Î E* the by Lemma 7 there does not exist any path from v_i^* to *u* of length 2. Also, by Lemma 5 there does not exist any path *u* ® *x* ® v_{i+1}^* ® v_i^* or *u* ® *x* ® v_{i-1}^* ® v_i^* for any vertex *x Î D.* Also *u Ï M*, then *nd (u,* v_i^**)* is not equal to $|M|$ – 1. So there exist only one path between v_i^* and *u*. So, the next-to-shortest path does not exist between v_i^* and *u* i.e., *nd (*v_i^**,u)* = ¥.

In the graph of the Figure 6, *nd (2,3)* = ¥.

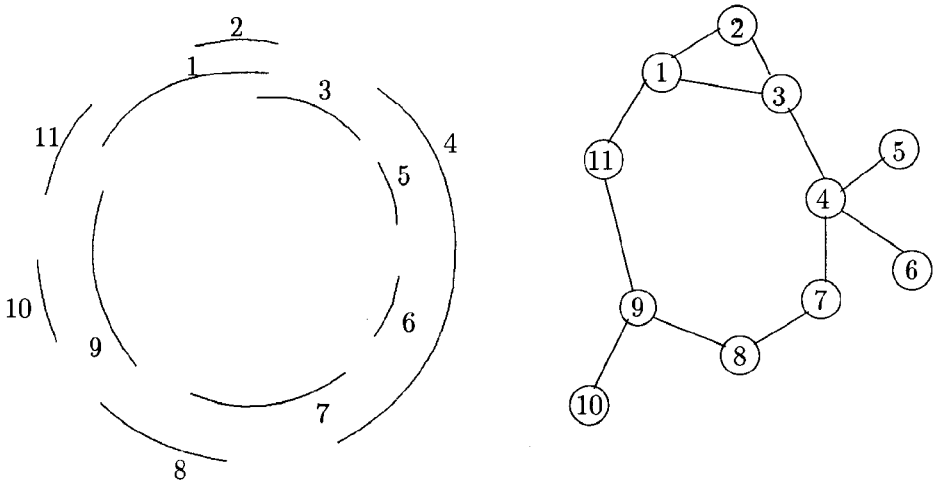

Figure 7 : Illustration of Lemmas 10, 11, 13, 14, 15

Lemma 10. *Let* v_i^* *and* v_j^* *be any two non-adjacent vertices on main cycle M. If there exist an edge* (u_i^*, u_{i+1}^*) *on shortest path between* v_i^* *to* v_j^* *such that* $nd(u_i^*, u_{i+1}^*) = 2$ *then* $nd(v_i^*, v_j^*) = d(v_i^*, v_j^*) + 1$.

Proof : Let (u_i^*, u_{i+1}^*) be an edge on the shortest path between v_i^* and v_j^* where next shortest distance between u_i^* to u_{i+1}^* is 2. Let the next shortest path between u_i^* to u_{i+1}^* be u_i^* ® v ® u_{i+1}^* for any vertex v Î D. If the shortest path between v_i^* to v_j^* is $v_i^* \to v_{i+1}^* \xrightarrow{*} u_i^* \to u_{i+1}^* \xrightarrow{*} v_j^*$ then there exist the path $v_i^* \to v_{i+1}^* \xrightarrow{*} u_i^* \to v \to u_{i+1}^* \xrightarrow{*} v_j^*$. The length of this path is $d(v_i^*, v_i^*)+1$ and this length is greater than the shortest distance by 1. So this path is next shortest path between v_i^* to v_j^*. Therefore, next shortest distance between v_i^* and v_j^* is $d(v_i^*, v_j^*)+1$.

In the graph of the Figure 7, $nd(11,4) = d(11,4)+1 = 3+1 = 4$.

Lemma 11. *Let* v_i^* and v_j^* *be any two non-adjacent vertices on main cycle M. If there does not exist an edge* (u_i^*, u_{i+1}^*) *on shortest path between* v_i^* and v_j^* *such that* $nd(u_i^*, u_{i+1}^*) = 2$ *then* $nd(v_i^*, v_j^*) = |M| - d(v_i^*, v_j^*)$.

Proof : Let the shortest path between v_i^* and v_j^* be $v_i^* \to v_{i+1}^* \xrightarrow{*} v_{i-1}^* \to v_j^*$. If there does not exist an edge (u_i^*, u_{i+1}^*) on shortest path between v_i^* and v_j^* such that $nd(u_i^*, u_{i+1}^*) = 2$ then there does not exist any path with length $d(v_i^*, v_j^*)+1$ between v_i^* and v_j^*. If v_i^* and v_j^* are the vertices of the set M then there exist another path $v_i^* \to v_{i-1}^* \xrightarrow{*} v_{j+2}^* \to v_{j+1}^* \to v_j^*$.

The length of this path is $|M| - d(v_i^*, v_i^*)$, if this is greater than $d(v_i^*, v_j^*)$. So the next shortest distance is $|M| - d(v_i^*, v_i^*)$.

In the graph of the Figure 7, $nd(9,7) = |M| - d(9,7) = 7 - 2 = 5$.

Lemma 12. *Let* v_i and v_j *be any two non-adjacent vertices on D. If there exist an edge (x,y) on shortest path between* v_i and v_j *such that* $nd(x,y) = 2$, *then* $nd(v_i, v_j) = d(v_i, v_j) + 1$ *where x, y Î V.*

Proof : Let (x,y) is an edge on the shortest path between v_i and v_j such that $nd(x,y) = 2$ for $x, y \hat{I} V$. Let the next shortest path between x and y is $x \to v \to y$ for any vertex $v \hat{I} D$. If the shortest path between v_i and v_j is $v_i \to v_i^* \xrightarrow{*} x \to y \xrightarrow{*} v_j^* \to v_j$ then there exist the path $v_i \to v_i^* \xrightarrow{*} x \to y \xrightarrow{*} v_j^* \to v_j$. The length of this path is $d(v_i, v_j) + 1$ and this length is greater than shortest distance by 1. So this path is next shortest distance between v_i and v_j. Therefore the next shortest distance between v_i and v_j is $d(v_i, v_j) + 1$.

Lemma 13. *Let* v_i and v_j^* *be any two non-adjacent vertices where* v_i *Î D and* v_j^* *Î M. If there does not exist an edge* (u_i^*, u_{i+1}^*) *on the shortest path between* v_i and v_j^* *such that* $nd(u_i^*, u_{i+1}^*) = 2$, *then* $nd(v_i, v_j^*) = nd(v_i^*, v_j^*) + 1$ *where* $(v_i, v_i^*) \in E$.

Proof : If there does not exist an edge (u_i^*, u_{i+1}^*) on the shortest path between the vertices v_i and v_j^* then there does not exist any path with distance $d(v_i, v_j^*) + 1$. If v_i *Ï M* then there does not exist any path with distance $|M| - d(v_i, v_j^*)$. If there does not exist an edge (u_i^*, u_{i+1}^*) on the shortest path between the vertices v_i and v_j^* then there does not exist any edge (u_i^*, u_{i+1}^*) on the shortest path between v_i and v_j^*. So by Lemma 11, the next shortest path from v_i^* to v_j^* is $v_i^* \to v_{i-1}^* \xrightarrow{*} v_{j+1}^* \to v_j^*$. Then there exist a path from v_i to v_j^* which is $v_i \to v_i^* \to v_{i-1}^* \xrightarrow{*} v_{j+i}^* \to v_j^*$. The length of this path is $nd(v_i^*, v_j^*) + 1$. Therefore the next shortest distance between v_i to v_j^* is $nd(v_i^*, v_j^*) + 1$.

Lemma 14. *Let* v_i, v_j *be two non-adjacent vertices of the set D. If there does not exist an edge* (u_i^*, u_{i+1}^*) *on the shortest path between* v_i *and* v_j *such that* $nd(u_i^*, u_{i+1}^*) = 2$, *when* $nd(v_i, v_j) = nd(v_i^*, v_j^*) + 2$ *where* $(v_i, v_i^*) \in E$ *and* $(v_j, v_{j+1}^*) \in E$ *for any two vertices* $v_i^*, v_j^* \in M$.

Proof : Let the shortest path between the vertices v_i *and* v_j is $v_i \to v_i^* \to v_{i+1}^* \xrightarrow{*} v_{j-1}^* \to v_j^* \to v_j$ where the shortest path between the vertices from v_i^* to v_j^* is $v_i^* \to v_{i+1}^* \xrightarrow{*} v_{j-1}^* \to v_j^*$. So $d(v_i, v_j) = d(v_i^*, v_j^*) + 2$. If there does not exist any edge (u_i^*, u_{i+1}^*) on the shortest path from v_i to v_j, such that $nd(v_i, v_j) = 2$. So the next shortest distance between the vertices v_i^* and v_j^* is

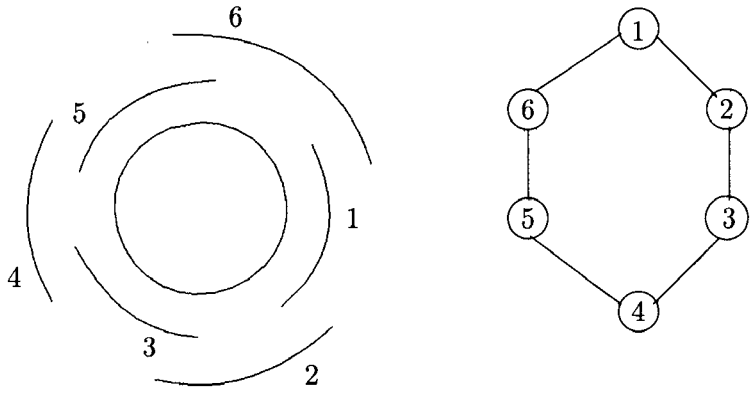

Figure 8

$v_i^* \to v_{i-1}^* \xrightarrow{*} v_{j+2}^* \to v_{j+1}^* \to v_j^*$ and the next shortest path from v_1 *to* v_j is $v_i \to v_i^* \to v_{i-1}^* \xrightarrow{*} v_{j+2}^* \to v_{j+1}^* \to v_j^* \to v_j$. The length of this path is $nd(v_i^*, v_j^*) + 2$. Therefore the next shortest distance from v_i to v_j is $nd(v_i^*, v_j^*) + 2$.

In the graph of the Figure 7, $nd(10,5) = d(9,4) + 2 = 4 + 2 = 6$.

Lemma 15. *If* v_i, v_j $\hat{I}$ D *are adjacent with only one vertex, say,* $v_k^* \in M$ *then* $nd(v_i, v_j) = \infty$.

Proof : If v_i, v_j $\hat{I}$ D are adjacent with only one vertex $v_k^* \in M$, then there exist only one path from v_i *to* v_j which is $v_i \to v_k^* \to v_j$. This is shortest path from v_i *to* v_j. Therefore the next shortest distance between the vertices v_i *to* v_j is ∞.

5. The Algorithm and its Complexity

In this section, we present an algorithm to find the paths and their distances between every pair of vertices of a circular-arc graph. The time complexity is also calculated here.

Here three procedures FINDNSP 1, FINDNSP 2 and FINDNSP 3 are formally presented in the following which computes the next shortest distances of adjacent vertices next shortest distance of non-adjacent vertices of the set M and the next shortest distances of all other

pair of vertices.

Using the results of the lemmas 6 to 9 we design the procedure FINDNSP 1.

Procedure FINDNSP 1

// This procedure computes the next-to-shortest distance between two adjacent vertices v_i, v_j Î V. //

Let $(v_i, v_j) \in E$.

If $(v_i, x) \in E$ and $(v_j, x) \in E$ for at least one x Î V then $nd(v_i, v_j) = 2$ (lemmas 6 and 8);

else if $(v_i, x) \notin E$ for all x Î V

if $v_i, v_j \in M$ then $nd(v_i, v_j) = |M| - 1$ (Lemma 7);

else if $v_i \notin M$ or $v_j \notin M$ then $nd(v_i, v_j) = \infty$;

endif;

endif;

endif;

end FINDNSP 1

Lemma 16. *The time complexity of the procedure* FINDNSP 1 *is O(n).*

Proof : In the procedure FINDNSP 1 we find the next-to-shortest distance between any tow adjacent vertices. First, we find a vertex of the set *D* which is adjacent with both the adjacent vertics. To find the set *M* or not. This testing can be done using *O(n)* time. So the total time complexity of the procedure FINDNSP 1 is *O(n).*

Based on the results of the lemmas 10 to 11 we have designed the procedure FINDNSP 2. This procedure deterines the next-to-shortest distance between two non-adjacent vertices of *M.*

Procedure FINDNSP 2

// This procedure computes the next-to-shortest distance between two vertices v_i^* and v_j^* of *M. //*

Let $(v_i^*, v_j^*) \notin E$.

If there exist an edge (u_i^*, u_{i+1}^*) on the shortest path between v_i^* and v_j^* such that $nd(u_i^*, u_{i+1}^*) = 2$ then $nd(v_i^*, v_j^*) = |M| - d(v_i^*, v_j^*)$; (Lemma 11)

otherwise $nd(v_i^*, v_j^*) = \infty$.;

endif;
endif;
endif;
end FINDNSP 1

Lemma 17. *The time complexity of the procedure* FINDNSP 2 *is O (n).*
Proof : In the procedure FINDNSP 2 we find the next-to-shortest distance between any two non-adjacent vertices of *M.* First we consider the shortest path between the vertices of *M.* The time complexity to find the shortest path between any pair of vertices is *O (n)* [13]. Also, we need to find next-to-shortest distance between any two adjacent vertices. We know the time complexity to find next-to-shortest distance between any two adjacent vertices is *O (n)* (Lemma 16). So, the time to find the next-to-shortest distance of any two non-adjacent vertices of *M* is *O (n).* Therefore, the time complexity of the procedure FINDNSP 2 is *O (n).*

The following procedure end FINDNSP 3 is designed based on the results of the lemmas 12 to 15. This procedure determines the next-to-shortest distance between two vertices *u, v Î D* and *(u,v) Ï E.*

Procedure FINDNSP 3

// This procedure determines the next-to-shortest between two vertices of *D,* when they are not adjacent. //

If $(v_i, v_j) \notin E$.

If there exist an edge (u_i^*, u_{i+1}^*) on the shortest path between v_i and v_j such that

$nd(u_i^*, u_{i+1}^*) = 2$ then $nd(v_i, v_j) = nd(v_i, v_j) + 1$; (Lemma 12)

else if there does not exist an edge (u_i^*, u_{i+1}^*) on the shortest path between v_i^* and v_j^*

such that $nd(u_i^*, u_{i+1}^*) = 2$

else if v_i, *Î D,* v_j *Î M* then $nd(v_i, v_j) = nd(v_i^*, v_j) + 1$ where $(v_i, v_i^*) \in E$ for $v_i^* \in M$; (Lemma 13)

else if v_i, v_j *Î D* then $nd(v_i, v_j) = nd(v_i^*, v_j^*) + 2$ where $(v_i, v_i^*) \in E$

$(v_j, v_j^*) \in E$ for $v_i^*, v_j^* \in M$; (Lemma 14)

else if $(v_i, x) \in E$ and $(v_j, x) \in E$ for a vertex v *Î M* then $nd(v_i, v_j) = \infty$; (Lemma 15)

endif;
endif;

endif;
end FINDNSP 3

Lemma 18. *The time complexity of the procedure* FINDNSP 3 *is O(n).*
Proof : In the procedure FINDNSP 3 we find the next-to-shortest distance between any two non-adjacent vertices of *D.* Here we determine the shortest path and their distance between the vertices of *D.* This can be done in *O(n)* time [13]. Also, we need to find next-to-shortest distance between any two adjacent vertices. We know the time complexity to find next-to-shortest distance between any two adjacent vertices is *O(n)* (Lemma 16). So, the time required to find the next-to-shortest distance of any two non-adjacent vertices of *D* is *O(n).* Hence, the time complexity of the procedure FINDNSP 3 is *O(n).*

Combining all these procedure and lemmas presented in this section, we have designed the algorthm NEXTSP between all pairs of vertices of a circular-arc graph.

Algorithm NEXTSP

Input : Afamily of circular arcs *A* of a circular-arc graph *G.*
Output : All pairs next-to-shortest distances *nd(u,v), u, v Î V.*
Step 1 : Find all pair shortest paths and their distances using the algorithm of Saha et al. [13].
Step 2 : Compute *M* and the vertices of it.
Step 3 : Find next-to-shortes distance between two adjacent vertices *u* and *v, u, v Î V.* using the proceding FINDNSP 1.
Step 4 : Find next-to-shortest distance between any two non-adjacent vertices of M, using the procedure FINDNSP 2.
Step 5 : Find next-to-shortest distance between *u, v Î D,* where (*u,v*) *Ï E,*using the procedure FINDNSP 3.

end NEXTSP

Lemma 19. *The time complexity of the algorithm* NEXTSP *is* $O(n^2)$.
Proof : In algorithm NEXTSP Step 1 takes $O(n^2)$ time by the algorithm of Saha et al. [13]. Step 2 takes *O(n)* by the Theorem 1. The time complexity to find next-to-shortest distance between any two adjacent vertices is *O(n).* So the time required to find next-to-shortest distance between all pair of adjacent vertices is $O(n^2)$. So, Step 3 takes $O(n^2)$ time. Again, to find the next-to-shortest distance between any two vertices of *M,* the time requires *O(n)* (Lemma 17). So the time complexity to find next-to-shortest distance between all pair non-adjacent vertices of *M is* $O(n^2)$. Therefore, Step 4 takes $O(n^2)$ time. Similarly, Step 5 takes $O(n^2 f)$ time (Lemma 18). Hence the total time complexity of the Algorithm NEXTSP $O(n^2)$.

By Lemma 19 we can conclude the following theorem.
Theorem 2. *The next-to-shortest distances between all-pairs of vertices of a circular-arc graph can be computed* $O(n^2)$ *time, where n is the number of vertices.*

REFERENCES

1. Atallah, M. J., Chen, D.Z. and Lee, D.T. (1995). An optimal algorithm for shortest paths on weighted interval and circular-arc graphs with applications, *Algorithmica,* **14,** 429-441.
2. Eppstein, D. (1999). Finding the k-shortest paths, *SIAM J. Comput,* **28**, 652-441.
3. Eschen, E. M. and Spinrad, J. P. (1993). An $O(n^2)$ algorithm for circular-arc graph recognition, *In Proc : 4th Annual ACM-SIAM Symposium on Discrete Algorithm, Austin, TX,* 128-137.
4. M. C. Golumbic, *Algorithm Graph Theory and Perfect Graphs,* (2004), Elsevier.
5. Hsu W. -L. and Tsai, K. -H. (1991). Linear time algorithms on circular-arc graphs, *Information Processing Letters,* **40**, 123-129.
6. Jimenez, V. M. and Marzaj, A. (1999). Computing the *k* shortest paths : a new algorithm and experimental comparison, in : J. S Vitter, C. D. Zaroliagis (Eds.), Algorithm Engineering : 3rd International Workshop, WAE '99, in : Lecture Notes in Compute. Sci., vol. 1668, *Springer*, Berlin, 15-29.
7. Krasikov, I. and Noble, S. D. (2004). Finding next-to-shortest paths in a graph, *Information Processing Letters*, **92**, 117-119.
8. Lalgudi, K. N. and Papaefthymiou, M. C. (1997). Computing strickly second shortest path, *Information Processing Letters*, **63**, 177-181.
9. Mondal, S., Pal, M. and Pal, T. k. and (2002). An optimal algorithm for solving all-pairs shortes paths on trapezoid graphs, *Inter. J. computational Engineering Science*, **3 (2)**, 103-116.
10. Mondal, S., Pal, M. and Pal, T. K. and (2003). An optimal algorithm to solve all-pairs shortest paths problem on permutation graphs, *Journal of Mathematical Modelling and Algorithms*, **2**, 57-65.
11. Pal, M. and Bhattacharjee, G. P. (1996). A sequential algorithm for finding a maximum weight *k*-independent set on interval graph, *Intern. J. Computer Math.*, **60**, 205-214.
12. Pal, M. and Bhattacharjee, G. P. (1997). An optimal parallel algorithm for all-pairs shortest paths on unweighted interval graphs, *Nordic J. Computing*, **4**, 342-356.
13. Saha, A., Pal, M. and Pal, T. K. (2005). An optimal parallel algorithm for solving all-pairs shortest paths problem on circular-arc graphs, *J. Applied Mathematics and Computing*, **17**, 1-23.
14. Seidel, R. (1992). *On the all pairs shortest path problem,* In Proceedings of 24th ACM STOC, ACM Press, 745-749.
15. Tucker, A. (1980). An efficient test for circular-arc graph, *SIAM J. Comput.,* **9**, 1-24.