An Algorithm for Identifying All Hinge Vertices on a Circular Permutation Graph

Hirotoshi HONMA[1] Yoko NAKAJIMA[1]

Abstract: Let $G_s = (V_s, E_s)$ be a simple connected graph. A vertex $u \in V_s$ is called a hinge vertex if there exist any two vertices x and y in G_s whose distance increase when u is removed. Finding all hinge vertices of a given graph is called the hinge vertex problem. These problems can be applied to improve the stability and robustness of communication network systems. In this study, we propose a linear time algorithm for the hinge vertex problem of a circular permutation graph.

Key words: Graph algorithms, Circular Permutation Graphs, Hinge Vertices;

1 Introduction

Let $G_s = (V_s, E_s)$ be a simple connected graph with $|V| = n$ and $|E| = m$. A vertex $u \in V_s$ is called a *hinge vertex* if there exist any two vertices x and y in G_s whose distance increase when u is removed. A graph without hinge vertices is called a *self-repairing graph*. Articulation vertices are a special case of hinge vertices in that the removal of an articulation vertex u changes the finite distance of some nonadjacent vertices x and y to infinity. Finding all hinge vertices of a given graph is called the hinge vertex problem. There exists an $O(n^3)$ time algorithm for solving the hinge vertex problem of a simple graph. These problems can be applied to improve the stability and robustness of communication network systems [1].

In many cases, more efficient algorithms can be developed by restricting the classes of graphs. Ho et al. [2] presented an $O(n)$ time algorithm for the hinge vertex problem on permutation graphs, whose minor error was corrected by [3]. Furthermore, for *interval graphs*, Hsu et al. [4] presented an $O(n)$ time algorithm for the hinge vertex problem.

Let $V_p = [1, 2, \ldots, n]$ be a vertex set and $P = [p(1), p(2), \ldots, p(n)]$ be a permutation of V_p. A permutation graph G_p is visualized by its corresponding *permutation model* M_p, which consists of two horizontal parallel lines called the *top channel* and *bottom channel*, respectively. Place the vertices $1, 2, \ldots, n$ on the top channel, ordered from left to right, and similarly, place $p(1), p(2), \ldots, p(n)$ on the bottom channel. Next, for each $i \in V_p$, draw a straight line from i on the top channel to i on the bottom channel. Then, an edge (i, j) in G_p exists if and only if lines i and j intersect in M_p. An example of a permutation model M_p and its corresponding permutation graph G_p is shown in Fig. 1. Permutation graphs are an important subclass of perfect graphs, and they are used for modeling practical problems in many areas, such as biology, genetics, very large scale integration (VLSI) design, and network planning [5].

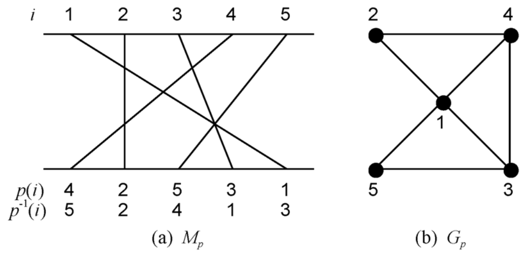

Figure 1: Permutation model M_p and graph G_p.

Circular permutation graphs properly contain a set of permutation graphs as a subclass. Rotem and Urrutia first introduced circular permutation graphs and provided an $O(n^{2.376})$ time recognition algorithm [6]. Lou and Sarrafzadeh showed that circular permutation graphs and their models have several applications in VLSI layout design [7]. They presented an $O(\min(\delta n \log\log n, n \log n) + |E|)$ time algorithm for finding a maximum independent set of a circular permutation model, where δ is the minimum degree of vertices in the corresponding circular permutation graph. Furthermore, they presented an $O(n \log\log n)$ time algorithm for finding the maximum clique and the chromatic number of a circular permutation model. Subsequently, the recognition algorithm was improved in $O(m + n)$ time by Sritharan [8].

In this study, we propose a linear time algorithm for the hinge vertex problem of a circular permutation graph. The rest of this paper is organized as follows. Section 2 describes some definitions of circular permutation graphs and models. Section 3 introduces the extended circular permutation model and its properties. Sections 4 consider algorithms that address hinge vertex problem and the complexity of this algorithm. Section 5 concludes this paper.

[1]Department of Information Engineering, National Institute of Technology, Kushiro College

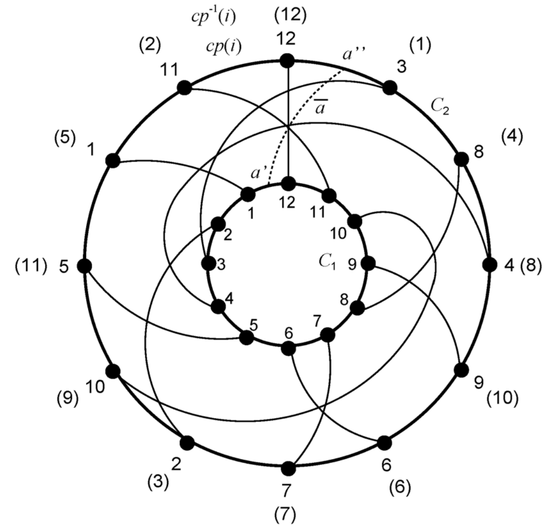

Figure 2: Circular permutation model CM.

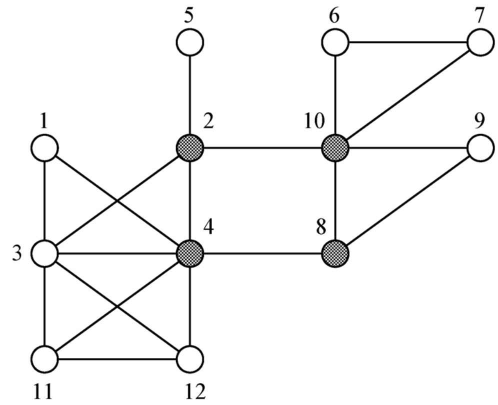

Figure 3: Circular permutation graph G.

2 Circular Permutation Model and Graph

We first illustrate the *circular permutation model* before defining the circular permutation graph. There exist inner and outer circles C_1 and C_2 with radii $r_1 < r_2$. Let $CP = [cp(1), cp(2), \ldots, cp(n)]$ be a permutation of integer sequence $[1, 2, \ldots, n]$. Furthermore, $cp^{-1}(i)$, $1 \leqslant i \leqslant n$, denotes the position of the number i in CP. Consecutive integers i, $1 \leqslant i \leqslant n$, are set to be counter-clockwise on C_1. Similarly, $cp(i)$, $1 \leqslant i \leqslant n$, is set to be counter-clockwise on C_2. For each i, $1 \leqslant i \leqslant n$, draw a chord joining the two i's, one on C_1 and the other on C_2, denoted as *chord i*. The geometric representation described above is called a circular permutation model CM. Figure 2 illustrates an example of CM with 12 chords constructed by $CP = [11, 1, 5, 10, 2, 7, 6, 9, 4, 8, 3, 12]$. This model is considered to be *proper* if any two chords i and j intersect at most once in the CM. In this paper, we consider only proper circular permutation graphs and models, and therefore, the word "proper" is omitted henceforth.

Next, we introduce circular permutation graphs. An undirected graph G is a circular permutation graph if it can be represented by the following circular permutation model CM: each vertex of the graph corresponds to a chord in the annular region between two concentric circles C_1 and C_2, and two vertices are adjacent in G if and only if their corresponding chords intersect exactly once [6]. Figure 3 illustrates the circular permutation graph G corresponding to CM shown in Fig. 2. In this example, $\{2, 4, 8, 10\}$ is a hinge vertex set.

Next, we consider a fictitious chord $\overline{a}$ which connects the point a' that is placed between 1 and 12 on C_1 and point a'' on C_2. A chord that intersects $\overline{a}$ is called a *feedback chord*. The set of all feedback chords is denoted by F. Moreover, a set of feedback chords that intersect $\overline{a}$ in clockwise is defined as F^-, and a set of feedback chords that intersect $\overline{a}$ counterclockwise is defined as F^+. We must place point a'' on C_2 so that $|F^-| = |F^+|$ is satisfied. In the example shown in Fig. 2, point a'' is placed between 3 and 12 on C_2. Consequently, $F = \{3, 4, 11, 12\}$, $F^- = \{3, 4\}$ and $F^+ = \{11, 12\}$. If a fictitious chord $\overline{a}$ exists that does not intersect any chord in CM, a model formed by opening CM along $\overline{a}$ is equivalent to a permutation model. This problem can be solved by applying Ibarra et al.'s algorithm [9] because this problem is the same as that of permutation graphs. In this paper, we assume that any fictitious chord intersects at least one chord.

3 Extended Circular Permutation Model

In this section, we introduce an *extended circular permutation model ECM* that is constructed from a CM.

Let n be the number of chords in CM. First, a point a' is fixed between 1 and n on C_1. Next, we consider a fictitious chord $\overline{a}$ with $|F^-| = |F^+|$. In Fig. 2, we obtain $|F^-| = |F^+| = 2$ by placing point a'' between 3 and 12 on C_2. ECM is formed by opening CM along $\overline{a}$. ECM consists of two horizontal parallel lines L_1 and L_2, called top and bottom channels, respectively. The top channel L_1 is assigned the consecutive number i, $-n+1 \leqslant i \leqslant 2n$, from left to right. The bottom channel L_2 is assigned $p(i)$, $-n+1 \leqslant i \leqslant 2n$, from left to right. Here, $p(i)$, $1 \leqslant i \leqslant n$, on L_2, is assigned a cp value on C_2 in the counter-clockwise direction from point a''. Next, $p(i)$, $1 \leqslant i \leqslant n$, changes to $p(i) - n$ if $i \in F^+$. Furthermore, $p(i)$, $1 \leqslant i \leqslant n$, changes to $p(i) + n$ if $i \in F^-$. We execute $p(i - n) = p(i) - n$ and $p(n + i) = p(i) + n$ for $1 \leqslant i \leqslant n$. For each

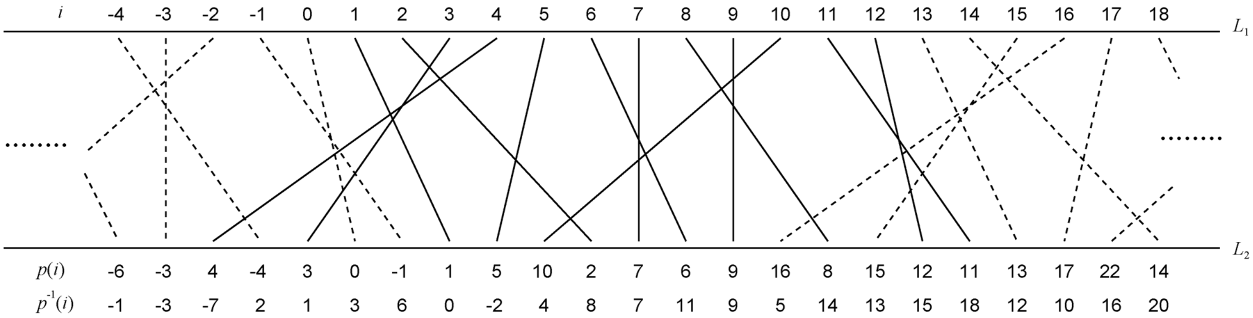

Figure 4: Extended circular permutation model *ECM*.

$-n+1 \leqslant i \leqslant 2n$, a straight line is drawn from i on L_1 to i on L_2. After executing the above process, *ECM* is constructed from *CM*. Figure 4 illustrates *ECM* constructed from *CM* shown in Fig. 2. Here, $p^{-1}(i)$ denotes the position of i on L_2.

Circular permutation and circular-arc graphs are circular versions of permutation and interval graphs, respectively. Moreover, as mentioned in Section 1, circular permutation and circular-arc graphs are superclasses of permutation and interval graphs, respectively. Efficient algorithms have been developed that address various problems concerning permutation and circular-arc graphs. However, in general, problems for circular graphs tend to be more difficult than those for non-circular graphs. One of the reasons is that we can not uniquely determine the starting position of an algorithm for a circular graph due to the existence of feedback elements although it can be fixed for a non-circular graphs.

For several problems, we can develop circular versions of the existing algorithms by constructing extended intersection models of the problems. By using extended intersection models, we can determine a start position of algorithm uniquely and apply partially the algorithms of the non-circular versions. For instance, this method has been applied to develop efficient algorithms for the shortest path query problem [10, 4], the articulation vertex problem [11] on circular-arc graphs, maximum clique and chromatic number problems [7], the spanning forest problem [12] on circular-permutation graphs. In this paper, we use *ECM* to construct an efficient algorithm for the hinge vertex problem.

Properties 1 and 2 stated below, can be derived in a straightforward manner from the processes of constructing *ECM*.

Property 1 *Lines $i-n$, i, and $i+n$ in ECM correspond to the vertex i in G.*

Property 2 *Let i and j $(i<j)$ be two vertices in G. Then, vertex i is adjacent to j if and only if lines i and j, or lines i and $j-n$, or lines $i+n$ and j intersect in ECM.*

Some notations that form the basis of the algorithms in sections 4 and 5 are defined as follows: The set of all lines that intersect line i in *ECM* is denoted by $N(i)$. In addition, $N[i] = N(i) \cup \{i\}$. For line i in *ECM*, the following functions are defined: $TR(i) = \max\{\ j \mid j \in N[i]\}$ and $STR(i) = \max\{\ j \mid j \in (N[i] \setminus TR(i)) \cup \{i\}\}$. $D_R(i) = \{\ k \mid STR(i) < k < TR(i)\}$. $TL(i) = \min\{\ j \mid j \in N[i]\}$ and $STL(i) = \min\{\ j \mid j \in (N[i] \setminus TL(i)) \cup \{i\}\}$. $D_L(i) = \{\ k \mid TL(i) < k < STL(i)\}$. $BR(i) = k$ such that $p^{-1}(k) = \max\{\ p^{-1}(j) \mid j \in N[i]\}$. $BL(i) = k$ such that $p^{-1}(k) = \min\{\ p^{-1}(j) \mid j \in N[i]\}$. Table 1 shows $TR(i)$, $STR(i)$, $D_R(i)$, $TL(i)$, $STL(i)$, $D_L(i)$, $BR(i)$, and $BL(i)$ for *ECM* shown in Fig. 4.

4 Algorithm for Hinge Vertex Problem

In this section, we present Algorithm HVC for finding all hinge vertices of circular permutation graphs. A vertex u is considered to be a hinge vertex if there exist any two vertices x and y in G whose distance increase by removing u.

4.1 Properties of Hinge Vertex

The following Lemma 1 proposed by Chang et al. [13] characterizes the hinge vertices of a simple graph G_s.

Lemma 1 ([13]) *For a simple graph G_s, a vertex u is a hinge vertex of G_s if and only if there exist two nonadjacent vertices $x<y$ such that u is the only vertex adjacent to both x and y in G_s.*

Lemma 2 provides the necessary and sufficient condition for hinge vertices in a permutation graph presented by Ho et al. [2].

Lemma 2 ([2]) *Let G_p be a permutation graph corresponding to a permutation model M_p. A vertex u is a hinge vertex of G_p if and only if there exist two vertices $x<y$; such that either of the following conditions holds in M_p:*

Table 1: Example of $TR(i)$, $STR(i)$, $TL(i)$, $STL(i)$, $BR(i)$, $BR(i)$, $A(i)$ and $B(i)$.

i	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$p(i)$	-3	4	-4	3	0	-1	1	5	10	2	7	6	9	16	8	15	12	11	13	17
$p^{-1}(i)$	-3	-7	2	1	3	6	0	-2	4	8	7	11	9	5	14	13	15	18	12	10
$TR(i)$	-2	-2	4	4	4	10	4	4	5	10	10	16	10	10	16	16	16	22	16	16
$STR(i)$	-3	-2	3	3	3	5	3	4	5	7	7	10	9	10	15	15	15	17	15	16
$D_R(i)$						6...9				8,9	8,9	11...15								
$TL(i)$	-4	-10	-1	-1	1	2	-1	-4	2	6	6	8	8	2	11	11	13	14	11	8
$STL(i)$	-3	-6	-1	0	1	2	0	-1	5	6	7	8	9	6	11	12	13	14	14	14
$D_L(i)$								-3,-2	3,4					3...5						
$BR(i)$	-4	-4	-1	-1	1	2	2	2	2	6	6	8	8	8	11	11	13	14	12	11
$BL(i)$	-2	-2	4	4	4	4	4	4	5	10	10	10	10	10	16	16	16	16	16	16

(1) $u = TR(x)$ for $y \in D_R(x)$ and $p^{-1}(BR(x)) < p^{-1}(y)$,

(2) $u = TL(y)$ for $x \in D_L(y)$ and $p^{-1}(x) < p^{-1}(BL(y))$.

Let $G = (V, E)$, $|V| = n$ be a circular permutation graph corresponding to a circular permutation model *CM*, and *ECM* be an extended circular permutation model constructed from *CM*. Lemmas 3 and 4 follow from Lemmas 1 and 2, respectively.

Lemma 3 *A vertex $u = TR(x)$ is a hinge vertex of G if there exist two vertices $x < y \in V$ satisfying $y \in D_R(x)$, $p^{-1}(BR(x)) < p^{-1}(y)$, $TR(y) < x + n$, and $p^{-1}(BR(y)) < p^{-1}(x + n)$ in ECM.*

(Proof) ($\Rightarrow$) If u is a hinge vertex of G, by Lemma 2, $u = TR(x)$, $STR(x) < y < TR(x)$, and $p^{-1}(BR(x)) < p^{-1}(y)$ in *ECM*. This indicates that line x does not intersect line y and u is the only line intersecting both lines x and y in *ECM* (Fig. 5). Assume that $TR(y) > x+n$ or $p^{-1}(BR(y)) > p^{-1}(x+n)$. If $TR(y) > x+n$, the line $TR(y)$ intersects both lines y and $x+n$. Note that line $x+n$ is a copy of line x. That is, both lines $x+n$ and x correspond to the same vertex x. This contradicts the assumption that u is the only vertex adjacent to vertices x and y in G. Furthermore, if $p^{-1}(BR(y)) > p^{-1}(x+n)$, line $BR(y)$ intersects both y and $x+n$. This is found to be contradictory to the assumption. Thus, necessity is satisfied.
($\Leftarrow$) By Lemma 2, if $u = TR(x)$, $y \in D_R(x)$, and $p^{-1}(BR(x)) < p^{-1}(y)$, u is the only line that intersects both lines x and y in *ECM*. Furthermore, as $TR(y) < x+n$ and $p^{-1}(BR(y)) < p^{-1}(x+n)$, no line intersects both lines y and $x+n$. This implies that vertex u is the only vertex adjacent to both vertices x and y in G. Therefore, sufficiency is satisfied. □

Lemma 4 *A vertex $u = TL(y)$ is a hinge vertex of G if there exist two vertices $x < y \in V$ satisfying $x \in D_L(y)$, $p^{-1}(x) < p^{-1}(BL(y))$, $y - n < TL(x)$, and $p^{-1}(y - n) < p^{-1}(BL(x))$ in ECM.*

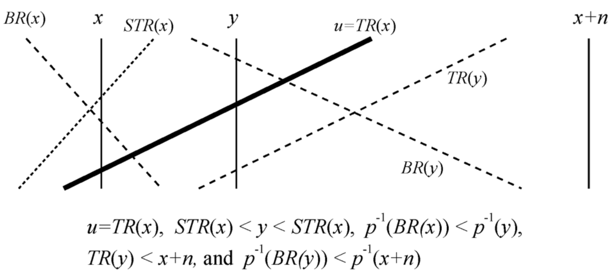

Figure 5: Example of Lemma 3.

(Proof) Lemma 4 is symmetric to Lemma 3. Hence, its proof is similar to that of Lemma 3. □

We show an example where vertex 4 is recognized as a hinge vertex by applying Lemma 3. In Fig. 4, for $x = 8$ and $y = 13$, $y = 13 \in D_R(x) = \{11, 12, 13, 14, 15\}$, $p^{-1}(BR(x)) = 14 < p^{-1}(y) = 15$, $TR(y) = 16 < (x + n) = 20$, and $p^{-1}(BR(y)) = 15 < p^{-1}(x + 2) = 23$ hold. Hence, $TR(x) = TR(8) = 16$ is a hinge vertex for 8 and 13 by Lemma 3. Normalization indicates that vertex 4 is a hinge vertex for 8 and 1.

4.2 Analysis of Algorithm HVC

The algorithm for finding all articulation vertices of a circular permutation graph is described formally in Algorithm HVC.

Next, we analyze the complexity of Algorithm HVC. In Step 1, we construct a circular permutation model *ECM* that can be executed in $O(n)$ time. $TR(i)$, $STR(i)$, and $BR(i)$ are computed in Step 2. $TL(i)$, $STL(i)$, and $BL(i)$ are computed in Step 3. Preprocessing steps 2 and 3 take $O(n)$ time [13]. Steps 4 and 5 find all hinge vertices by applying Lemmas 3 and 4, respectively, and they run in $O(n)$ time. After executing Step 5, all hinge vertices of a circular permutation graph are correctly found. Hence, we have the following theorem:

Theorem 1 *Algorithm HVC can solve the hinge vertex problem of a circular permutation graph in $O(n)$*

Algorithm 1: Algorithm HVC

Input: $CP = \{p(1), p(2), \ldots, p(n)\}$ of a circular permutation graph G.
Output: Hinge vertices of G.
(Step 1)
Construct ECM and compute $p^{-1}(i)$;
(Step 2)
Compute $TR(i)$, $STR(i)$, $BR(i)$ for $1 \leqslant i \leqslant n$;
(Step 3)
Compute $TL(i)$, $STL(i)$, $BL(i)$ for $1 \leqslant i \leqslant n$;
(Step 4) /* Compute hinge vertices */ ;
for *each* $y \in D_R(x)$ **do**
  if $p^{-1}(BR(x) < p^{-1}(y)$, $TR(y) < x + n$ *and* $p^{-1}(BR(y)) < p^{-1}(x + n))$ **then** Normalize $TR(x)$ to obtain the hinge vertex ;
end
(Step 5)
for *each* $x \in D_L(y)$ **do**
  if $p^{-1}(x) < p^{-1}(BL(y))$, $y - n < TL(x)$, *and* $p^{-1}(y - n) < p^{-1}(BL(x))$ **then** Normalize $TL(y)$ to obtain the hinge vertex ;
end

Function Normalize $v\{$
if $v < 1$ **then** $v := v + n$;
if $v > n$ **then** $v := v - n$;
return v ;
$\}$

time.

5 Concluding Remarks

In this study, we presented an algorithm that runs in $O(n)$ time to find all hinge vertices of a circular permutation graph. Our algorithm partially uses Ho's algorithm [2]. In future, we will continue this research by extending the results to other classes of graphs.

References

[1] H. Honma and S. Masuyama, "A parallel algorithm for finding all hinge vertices of an interval graph," IEICE Trans. Inf. & Syst., vol.E84-D, no.3, pp.419–423, 2001.

[2] T.Y. Ho, Y.L. Wang, and M.T. Juan, "A linear time algorithm for finding all hinge vertices of a permutation graph," Inf. Process. Lett., vol.59, no.2, pp.103–107, 1996.

[3] H. Honma, K. Abe, and S. Masuyama, "Erratum and addendum to "a linear time algorithm for finding all hinge vertices of a permutation graph" [Information Processing Letters 59 (2) (1996) 103–107]," Inf. Process. Lett., vol.111, no.18, pp.891–894, 2011.

[4] F.R. Hsu, K. Shan, H.S. Chao, and R.C. Lee, "Some optimal parallel algorithms on interval and circular-arc graphs," J. Inf. Sci. Eng., vol.21, pp.627–642, 2005.

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

[6] D. Rotem and J. Urrutia, "Circular permutation graphs," Networks, vol.12, no.4, pp.429–437, 1982.

[7] R.D. Lou and M. Sarrafzadeh, "Circular permutation graph family with applications," Discrete Appl. Math., vol.40, no.3, pp.433–457, 1992.

[8] R. Sritharan, "An linear time algorithm to recognize circular permutation graphs," Networks, vol.27, no.3, pp.171–174, 1996.

[9] O.H. Ibarra and Q. Zheng, "Finding articulation points and bridges of permutation graphs," International Conference on Parallel Processing, St. Charles, Illinois, pp.77–81, 1993.

[10] D. Chen, D.T. Lee, R. Sridhar, and C. Sekharam, "Solving the all-pair shortest path query on interval and circular-arc graphs," Networks, vol.31, pp.249–258, 1998.

[11] T.W. Kao and S.J. Horng, "Optimal algorithms for computing articulation points and some related problems on a circular-arc graph," Parallel Computing, vol.21, no.6, pp.953–969, 1995.

[12] H. Honma, S. Honma, and S. Masuyama, "An optimal parallel algorithm for constructing a spanning tree on circular permutation graphs," IEICE Trans. Inf. & Syst., vol.E92-D, no.2, pp.141–148, 2009.

[13] J.M. Chang, C.C. Hsu, Y.L. Wang, and T.Y. Ho, "Finding the set of all hinge vertices for strongly chordal graphs in linear time," Inf. Sci., vol.99, no.3-4, pp.173–182, 1997.