Journal of Taiwan Normal University: Mathematics, Science & Technology
1999, 44(1), 31-42

Constant-Time Algorithms for Dominating Problem on Circular-Arc Graphs

Shun-Shii Lin Ching-Fung Lee

Department of Information and Computer Education
National Taiwan Normal University

Abstract

The objective of this paper is to solve the dominating problem on circular-arc graphs in O(1) time. This problem has not been solved in O(1) time before, even on the ideal PRAM model. In this paper, we take advantage of the characteristics of the PARBS (processor arrays with reconfigurable bus systems), which can connect the inner buses in O(1) time. We use $O(n^2)$ processors in the study. By combining the characteristics of PARBS and improving the methods of [14][15], we are able to derive constant-time algorithms for this problem.

Keywords: circular-arc graphs, dominating problem, PARBS (processor arrays with reconfigurable bus systems).

Introduction

A graph is an ordered pair G=(V, E), where V is a finite set of n=|V| elements called *vertices* and $E \subseteq \{(x, y) | x, y \in V, x \neq y\}$ is a set of m=|E| unordered vertex pairs called *edges*. Let $S=\{S_0, S_1, S_2, \cdots, S_{n-1}\}$ be a family of sets with each S_i $(0 \leq i \leq n-1)$ being a set. A graph G is *an intersection graph* of S if there is a one-to-one correspondence between V and S such that the vertices in V are adjacent if and only if their corresponding sets have a nonempty intersection [5]. There are many applications for circular-arc graphs, such as genetics [6], course scheduling [6], the channel assignment problem in computer-aided design [6], and so on. These applications rise some interesting problems on circular-arc graphs. There are many related researches as can be found in [2] [4] [7] [11] [12] [13]. The set S is then called the *intersection model of G* [1] [3] [7] [8]. When S is a set of circular-arcs on a circle, G is called a *circular-arc graph* [5]. A circular-arc graph is called a *proper circular-arc graph* if there is no circular-arc containing the other arcs or contained by the other arcs, in the given set of circular-arcs. For instance, Fig. 1 gives a set of proper circular-arcs. If a circular-arc graph is not proper, it is called a general circular-arc graph. For instance, Fig. 2 gives a set of general circular-arcs. Given a set A of circular-arcs, *LARGE(A)* denotes the set of arcs which are not contained by any other arcs [15]. For instance, in Fig. 2, LARGE(A)={1, 2, 3, 4, 8, 10, 11}. Note that arc 5 doesn't belong to LARGE(A), because arc 5 is contained by arc 4.

The *processor arrays with reconfigurable bus systems* model (abbreviated to PARBS) consists of a VLSI array of processors connected to a reconfigurable bus system which can be used to dynamically obtain various interconnection patterns between the processors. Each processor of PARBS has four inner ports and outer ports. The four inner ports could be connected dynamically in O(1) time. The four outer ports I^+, I^-, J^+ and J^- connect with its four neighborhood. In Fig. 3, we show some possible connections inside a processor. In this paper, the notation $\{q_1, q_2, \cdots, q_t\}$ is used to represent the local connection within a processor. This means a group of ports $q_1, q_2, \cdots, q_t$ is connected together within the processor. For example, in Fig. 3(h), the local connection within the processor can be represented by $\{J^+, I^+\}$ and $\{J^-, I^-\}$, where ports J^+ and I^+ are connected, and ports J^- and I^- are connected.

A set $B \subseteq V$ is called a dominating set if each vertex in A is adjacent to at least one member of B. *The dominating problem* is to find a dominating set with the minimum number of elements in it [10][11]. In [10], Hsu and Tsai give an O(n) time algorithm for the dominating set problem on circular-arc graphs. For this problem, Yu, Chen and Lee give an O(log n)-time parallel algorithm with O(n) processors on PRAM models for circular-arc graphs [14]. For solving the dominating set problem on interval graphs, Olariu and Schwing present an algorithm in constant time on an n*n PARBS model [13].

In this paper, we will develop O(1) time algorithms on a (2n)*n PARBS model to solve the dominating set problem on circular-arc graphs. In section II, we will introduce SMDS relations and its relative algorithm. SMDS[i] = j if arc j is the farthest arc in clockwise direction such that those arcs from arc i to j can be dominated by arc i or arc j. This relation is helpful for finding an answer. According to the SMDS relations, we can construct a SMDS relation graph. In section III, we classify SMDS relation graphs into two patterns, Pattern 1 and Pattern 2. In order to find one answer, we have to study the properties of SMDS relation graphs. In section IV, we can use these properties to find one answer for proper circular-arc graphs. In section V, we will introduce the algorithms for general circular-arc graphs. In section VI, we shall state our conclusions.

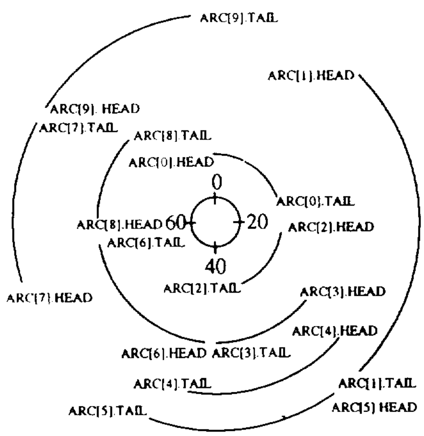


Fig.1 A set of proper circular-arcs

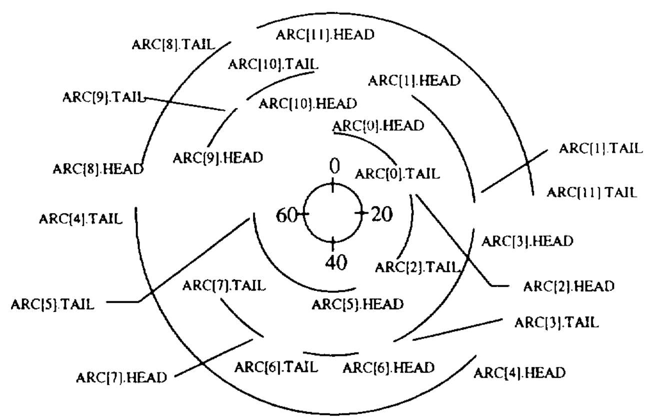


Fig.2 A set of general circular-arcs

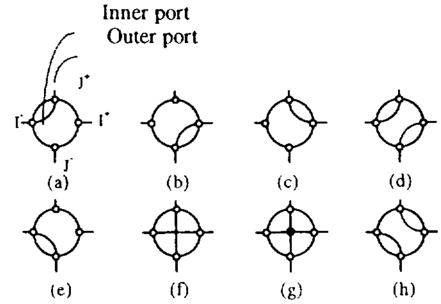


Fig.3 Some possible connections inside a processor

SMDS algorithm

There are two phases for solving dominating problem on proper circular-arc graphs. In the first phase, we find relation SMDS[i] for each arc i. In the second phase, we find a dominating set. In other words, if arc i belongs to a dominating set, arc *SMDS[i]* is the best next arc to be put into the dominating set. SMDS[i]=j if arc j is the farthest arc in clockwise direction such that those arcs from arc i to j can be dominated by arc i or arc j. For instance, in Fig. 1, SMDS[1]=6 because arcs 2, 3, 4 and 5 are dominated by arc 1 or 6. SMDS[1] $\neq$ 7 because arc 5 can not be dominated by arc 1 or 7.

In [8], we have SMIS and SMCC algorithms to find relations SMIS and SMCC, where *SMIS[i]*=j if arc j owns the nearest ending point among those arcs which have empty intersection with arc i in clockwise direction [15], and *SMCC[i]*=j if arc j owns the farthest ending point among those arcs which have nonempty intersection with arc i in clockwise direction [15]. We will apply SMIS and SMCC algorithms [9] in our SMDS algorithm.

Theorem 1. For a set of n circular-arcs, if there is no arc which dominates all the arcs, then SMDS[i]=SMCC[SMIS[i]] for $0 \leqslant i \leqslant$ n-1 [8].

The following algorithm will find the SMDS relation for each arc in O(1) time on an O(n^2) PARBS. Table 1 shows the values of the SMIS, SMCC and SMDS relations for the graph in Fig. 1 after this algorithm.

Table 1 The values of SMIS[j], SMCC[j] and SMDS[j] for the graph in Fig. 1

j	0	1	2	3	4	5	6	7	8	9
SMIS[j]	2	5	6	6	7	7	8	9	0	0
SMCC[j]	1	4	5	5	6	6	7	8	9	9
SMDS[j]	5	6	7	8	8	8	9	9	1	1

SMDS Algorithm

(We explain the algorithm by using the example in Fig. 1.)

Input: ARC[j].HEAD and ARC[j]. TAIL in P(0, j) for $0 \leqslant j \leqslant$ n-1, where ARC[j].HEAD and ARC[j].TAIL record the addresses of beginning and ending points of arc j.

Output: SMDS[j] in P(0, j) for $0 \leqslant j \leqslant$ n-1.

Step 1: Sort the n arcs ARC[j], $0 \leqslant j \leqslant$ n-1, according to the coordinates ARC[0].HEAD, ARC[1]. HEAD, $\cdots$, ARC[n-1].HEAD in increasing order, which are stored in P(0, j), $0 \leqslant j \leqslant$ n-1, respectively. This step can be computed in O(1) time on a 2-D n*n PARBS [5].

Step 2: Apply SMIS algorithm and SMCC algorithm in [9] to find SMIS[j] and SMCC[j], which are stored in P(0, j) for $0 \leqslant j \leqslant$ n-1.

Step 3: For $0 \leqslant i \leqslant$ n-1 and $0 \leqslant j \leqslant$ n-1, if i = j, P(i, j) makes a connection $\{I^+, I^-, J^+, J^-\}$; otherwise, P(i, j) makes connections $\{I^+, I^-\}$ and $\{J^+, J^-\}$. And then P(0, j) broadcasts SMCC[j] and SMIS[j] to port I^+. P(i, 0) sets the value received from port J+ to be SMCC[i]. (See Fig. 4(a))

Step 4: For $0 \leqslant i \leqslant n-1$ and $0 \leqslant j \leqslant n-1$, if SMIS[j] =i, P(i, j) makes a connection $\{I^+, I^-, J^+, J^-\}$; otherwise, P(i, j) makes connections $\{I^+, I^-\}$ and $\{J^+, J^-\}$. And then P(i, 0) broadcasts SMCC[i] to port J^+. P(0, j) sets the value received from port I^+ to be SMDS[j]. (See Fig. 4(b))

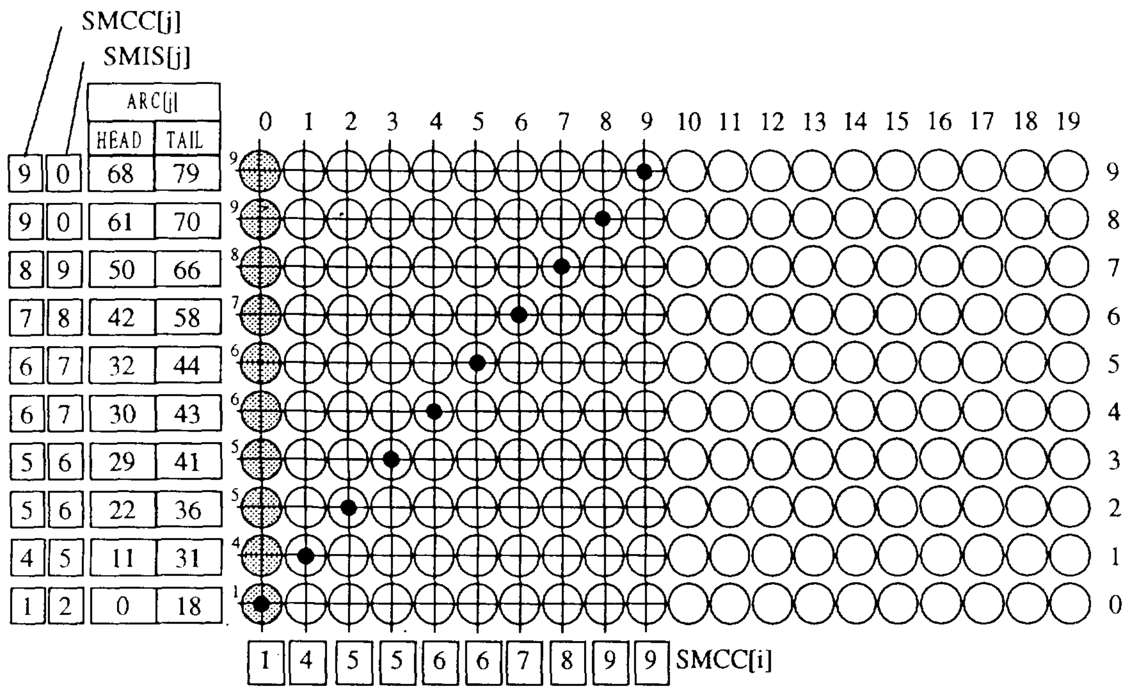


Fig. 4(a) After Steps 1, 2 and 3 of SMDS algorithm

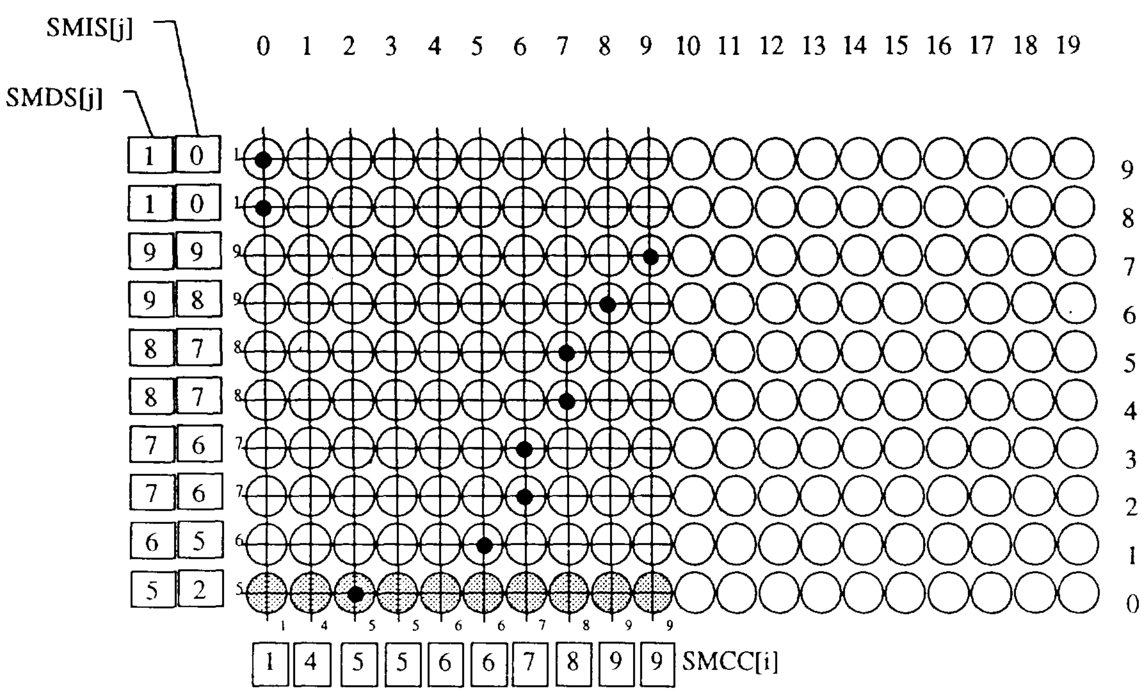


Fig. 4(b) After Step 4 of SMDS algorithm

Two patterns of the SMDS relation graphs

In the above section, we have found the relation SMDS. But how to use it to find the minimum dominating set? There is an idea about finding the minimum dominating set on proper circular-arc graphs. According to the SMDS relation, we can construct a 1-1 corresponding graph, and define this graph to be an *SMDS relation graph.* In an SMDS relation graph, a directed edge from node i to node j means that SMDS[i]=j. Define $SMDS^1[i]$ = SMDS[i] and $SMDS^k[i] = SMDS^{k-1}[SMDS[i]]$ if $k>1$. See Fig. 5 for a depiction. Note that the nodes with the indices between $SMDS^k[0]$ and $SMDS^{k+1}[0]$ are put at the same level as node $SMDS^k[0]$, $k=0, 1, 2, \cdots$. For instance, in Fig. 5, nodes 1, 2, 3, 4 are put at level 0 because SMDS[0] = 5. After we convert the SMDS relation for a proper circular-arc graph into an SMDS relation graph, the SMDS relation graph will help us to find one solution for this problem.

Let DS[i] denote a dominating set formed from arc i. For example, in Fig. 5, DS[0] = {0, 5, 8}, DS[1] = {1, 6, 9}, DS[2] = {2, 7, 9, 1}, DS[3] = {3, 7, 9, 1}, DS[4] = {4, 8, 1}, DS[5] = {5, 8, 1, 6}, DS[6] = {6, 9, 1}, DS[7] = {7 , 9, 1, 6}, DS[8] = {8, 1, 6}, DS[9] = {9, 1, 6}. Then there exists a minimum dominating set in $\{DS[i] \mid 0 \leq i \leq SMDS[0]-1\}$ because any set without the arcs between 0 and SMDS[0]-1 doesn't have the possibility to be a dominating set. Hence, we just need to consider the dominating sets formed by arcs 0, 1, $\cdots$, SMDS[0]-1. We observe that the difference of sizes between any two elements in $\{DS[i] \mid 0 \leq i \leq SMDS[0]-1\}$ is at most 1.

Now we want to find out the dominating set with minimum size. At first, we don't consider the edges between i and SMDS[i] if i > SMDS[i]. For the dominating problem, we have to find a shortest paths formed by node i, $0 \leq i \leq SMDS[0]-1$. At last, we construct DS[i]. If arc j is an element of DS[i], we let MDS[j]=1 to denote the final solution.

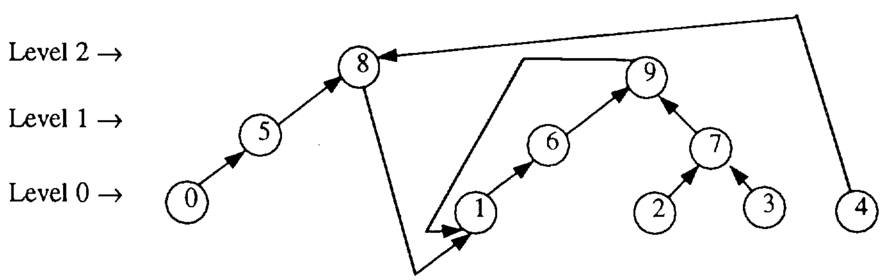
The SMDS relation graph has some properties. They are shown in the following lemmas. Due to page limit, we omit the proofs.

Lemma 1: Given an SMDS relation graph, the following properties will be satisfied:

(1) if $i_1 < i_2$, $i_1 < SMDS[i1]$ and $i_2 < SMDS[i_2]$, then $SMDS[i_1] \leq SMDS[i_2]$,

(2) if $i_1 < SMDS[i_1]$, $i_2 < SMDS[i_2]$ and $SMDS[i_1] < SMDS[i_2]$, then $i_1 < i_2$,

(3) the indices of the elements in $A_j = \{i \mid SMDS[i] = j, j > i\}$ are continuous. That is $A_j = \{\min(A_j), \min(A_j)+1, \min(A_j)+2, \cdots, \max(A_j)\}$.



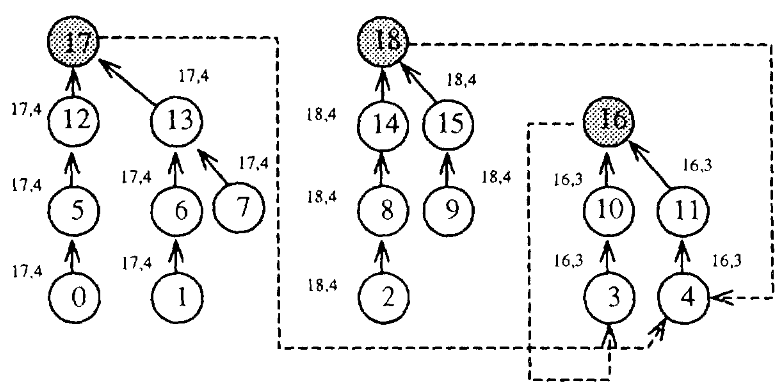
j	0	1	2	3	4
ROOT[j]	8	9	9	9	8
ROOT_SMDS[j]	1	1	1	1	1
GOOD[j]	1	1	0	0	0

Fig. 5 An SMDS relation graph for Fig.1

Lemma 2: Given an SMDS relation graph, we first remove the edges between i and SMDS[i] if i > SMDS[i]. Let $A_j = \{i \mid SMDS[i] = j, j > i\}$. Then the indices of the elements in $A_j \cup A_{j+1}$ are continuous.

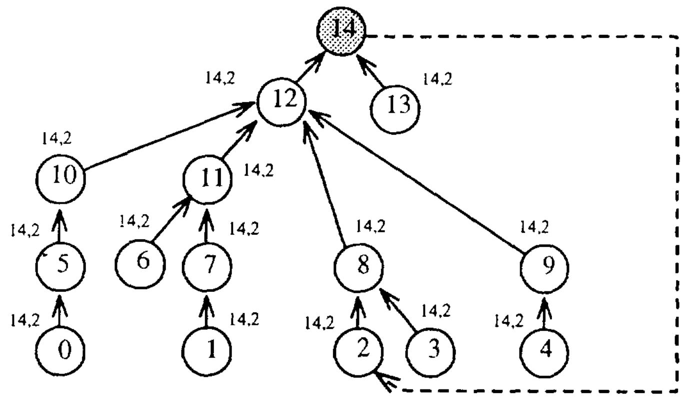
At first, we don't consider the edges between i and SMDS[i] if i > SMDS[i]. Now the SMDS relation graphs for proper circular-arc graphs can be classified into two patterns in general. See Fig. 6 and Fig. 7 for a depiction. *Pattern 1* consists of one or more trees $T_1, T_2, \cdots, T_t$. In each tree T_k, the lengths of the paths from the root to its leaves excluding nodes SMDS[0], $\cdots$, n-1 are the same. For instance, Fig. 6 shows a graph of Pattern 1. There are 3 trees. For the first tree T_1, the lengths of the paths formed by nodes 0 and 1 are 4. For the second tree T_2, the length of the path formed by node 2 is also 4. For the third tree T_3, the lengths of the paths formed by nodes 3 and 4 are 3. Note that the height of each tree may not be the same, but there are at most two kinds of heights. *Pattern 2* consists of only one tree, but with two kinds of lengths of the paths from the root to its leaves excluding nodes SMDS[0], $\cdots$, n-1. Fig. 7 shows a SMDS relation graph of Pattern 2. The lowest common ancestor of nodes 0, 1, 2, 3, and 4 is node 12. Note that, in Pattern 2, we have exactly two kinds of lengths in the subtrees of the lowest common ancestor. Let j be the lowest common ancestor of nodes 0, 1, $\cdots$, SMDS[0]-1. In Lemma 4, we show that if $0 \leq i < i+1 < m <$ SMDS[0] and the lengths of the paths formed by i and i+1 are different, then the length of the path formed by node m is the same as that formed by i+1. For example, in Fig. 7, the lengths of the paths formed by nodes 1 and 2 are different, so the length of the paths formed by node 3 is the same as that formed by node 2. Fig. 8 shows another example of Pattern 2.

Lemma 3: Given a proper circular-arc graph with n circular-arcs, if its SMDS relation graph belongs to Pattern 2, then there will exist a lowest common ancestor j of the nodes 0, 1, $\cdots$, SMDS[0]-1. In addition, there will exist a node i such that $SMDS^k[0] = SMDS^k[1] = \cdots = SMDS^k[i] = SMDS^{k-1}[i+1] = \cdots = SMDS^{k-1}[SMDS[0]-1] = j$.



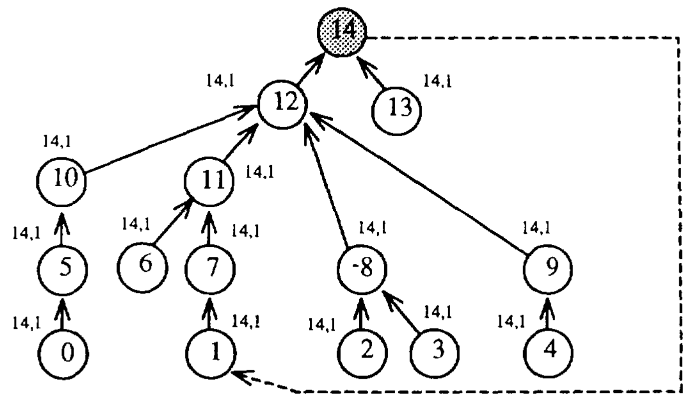
j	0	1	2	3	4
ROOT[j]	17	17	18	16	16
ROOT_SMDS[j]	4	4	3	3	3
GOOD[j]	1	1	1	1	0

Fig. 6 Pattern 1



j	0	1	2	3	4
ROOT[j]	14	14	14	14	14
ROOT_SMDS[j]	2	2	2	2	2
GOOD[j]	1	1	1	0	0

Fig. 7 Pattern 2



j	0	1	2	3	4
ROOT[j]	14	14	14	14	14
ROOT_SMDS[j]	1	1	1	1	1
GOOD[j]	1	1	0	0	0

Fig. 8 Pattern 2: another example

Find one dominating set

On PARBS models, we cannot find several optimal solutions concurrently, because some congestion may happen. What we can do is to just find one solution. Now we want to know which paths have the possibility. If a path formed by node j is the shortest one, it may contain one solution and we let GOOD[j] = 1 to denote this situation, where j=0, 1, ⋯ , SMDS[0]-1; otherwise, GOOD[j] = 0. Note that the difference of sizes between any two elements in {DS[0], DS[1], ⋯ , DS[SMDS[0]-

1]} is at most 1. How to find a path with the shortest length? We let each root broadcast its index and its SMDS value downward. For $0 \leq j \leq$ SMDS[0]-1, node j sets ROOT[j] to be the index of its root and sets ROOT_SMDS[j] to be SMDS[ROOT[j]]. If ($0 \leq j \leq$ SMDS[0]-1) and (ROOT_SMDS[j] $\geq$ j), the leaf node j sets GOOD[j] = 1; otherwise sets GOOD[j] = 0. For instances, in Fig. 5, ROOT_SMDS[0] = 1 $\geq$ 0, ROOT_SMDS[1] = 1 $\geq$ 1, so we let GOOD[0] = GOOD[1] = 1. In Fig. 6, ROOT_SMDS[0] = 4 $\geq$ 0, ROOT_SMDS[1] = 4 $\geq$ 1, ROOT_SMDS[2] = 4 $\geq$ 2 and ROOT_SMDS[3] = 3 $\geq$ 3. So we let GOOD[0] = GOOD[1] = GOOD[2] = GOOD[3] = 1. In Fig. 7, ROOT_SMDS[0] = 2 $\geq$ 0, ROOT_SMDS[1] = 2 $\geq$ 1 and ROOT_SMDS[2] = 2 $\geq$ 2. Then we let GOOD[0] = GOOD[1] = GOOD[2] = 1.

At last, we find the largest j such that GOOD[j] = 1 and then the path formed by node j is the minimum dominating set. In [13], Olariu and Schwing have shown that, given a collection of trees containing n nodes altogether, the nodes lying on the unique path joining a given node and the root of some tree in the collection, can be identified in O(1) time on a PARBS of size n*n. Since ROOT_SMDS[ROOT[0]] $\geq$ 0 (i.e. GOOD[0] = 1), we can guarantee that we can find one j such that GOOD[j] = 1 in this process. For instance, in Fig. 6, we find that j=3 because GOOD[3] = 1, so we choose {3, 10, 16} as our final solution. In Fig. 7, GOOD[0] = GOOD[1] = GOOD[2] = 1, so we choose {2, 8, 12, 14} as our final solution.

Theorem 2: Given a proper circular-arc graph with n arcs, the dominating set problem can be solved in O(1) time on a (2n)*n PARBS.

The following algorithm shows how to find the solution on PARBS. Table 2 shows the final values of MDS[j] for Fig. 1.

Algorithm for finding the minimum dominating set

(We explain the algorithm by using the example in Fig. 1.)

Input: SMDS[j] in P(0, j), $0 \leq j \leq$ n-1.

Output: $\text{MDS[i]} = \begin{cases} 1 & \text{if arc i belongs to the minimum dominating set,} \\ 0 & \text{otherwise,} \end{cases}$

in P(i, 0) for $0 \leq i \leq$ n-1.

Step 1: For $0 \leq i \leq$ n-1 and $0 \leq j \leq$ n-1, P(i, j) makes a connection {I^+, I^-}. Then P(0, j) broadcasts SMDS[j] to port I^+. (See Fig. 9(a))

Step 2: For $0 \leq i \leq$ n-1 and $0 \leq j \leq$ n-1, if (i = j) or ((SMDS[j] = i) and (SMDS[j] > j)), P(i, j) makes a connection {I^+, I^-, J^+, J^-}; otherwise, P(i, j) makes connections {I^+, I^-}and {J^+, J^-}. And then P(j, j) broadcasts j and SMDS[j] to port J^- if SMDS[j] $\leq$ j. P(0, j) sets the value received from port I^+ to be ROOT[j] and ROOT_SMDS[j]. (See Fig. 9(b))

Step 3: P(0, 0) broadcasts SMDS[0] to all processors. For $0 \leq j \leq$ n-1, if ($0 \leq j \leq$ SMDS[0]-1) and (ROOT_SMDS[j] $\leq$ j), P(0, j) sets GOOD[j] = 1; otherwise sets GOOD[j] = 0.

Step 4: For $0 \leq j \leq$ n-1, if GOOD[j] = 0, P(0, j) makes a connection {J^+, J^-}; otherwise, P(0, j) makes no connections. And then P(0, n-1) broadcasts a signal "#" to port J^+. (See Fig. 9(c))

Step 5: Find a path from node j to node ROOT[j] if if P(0, j) received the signal "#" from port J^+ but didn't receive the signal "#" from port J^- in Step 4. This step can be computed in O(1) time on a 2-D n*n PARBS [13]. (In [13], Olariu and Schwing have shown that, given a collection of trees containing n nodes altogether, the nodes lying on the unique path joining a given node and the root of some tree in the collection, can be identified in O(1) time on a PARBS of size n*n.) Now, for $0 \leq j \leq$ n-1, P(0, j) sets MDS[j] =1 if node j is on the path; otherwise P(0, j) sets MDS[j] =0. (See Fig. 9(c))

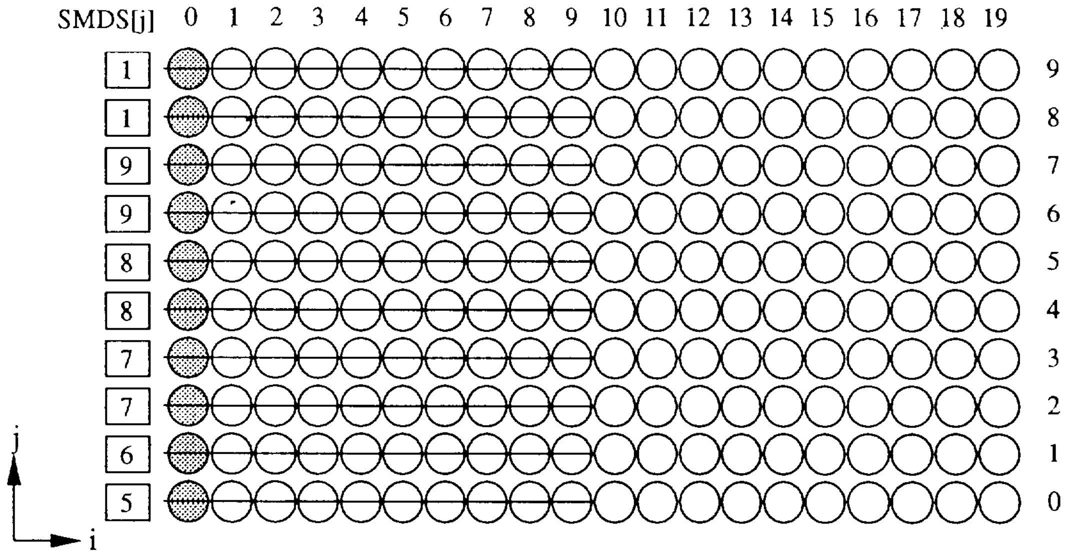


Fig. 9(a) After Step 1 of Algorithm for finding the minimum dominating set

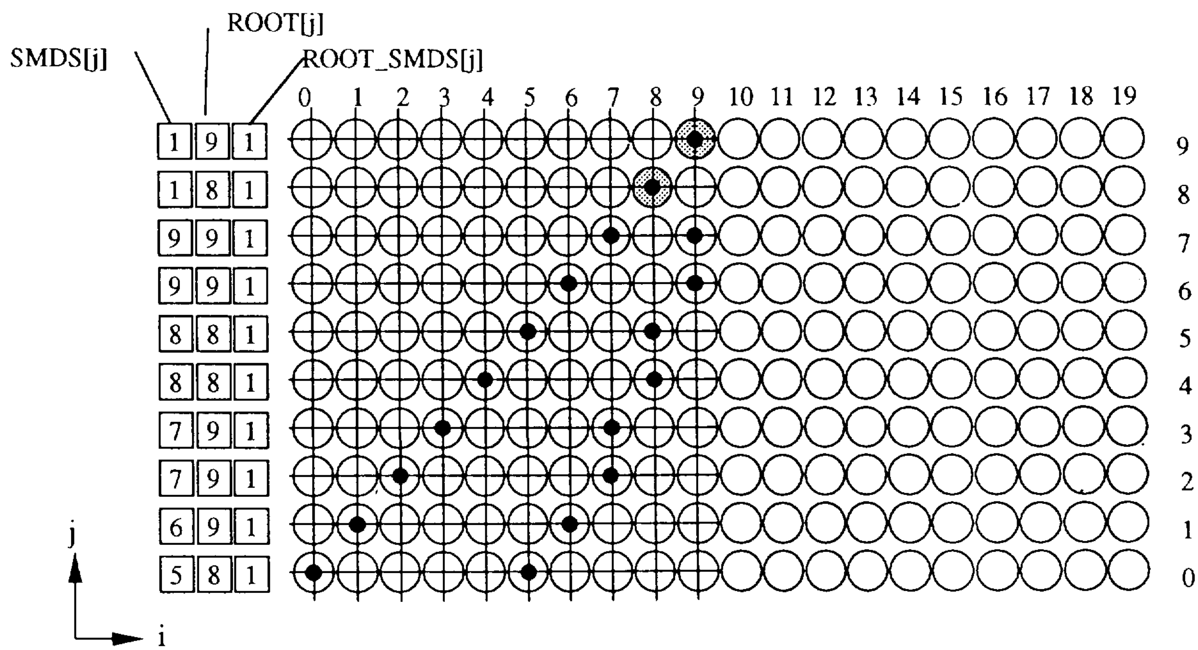


Fig. 9(b) After Step 2 of Algorithm for finding the minimum dominating set

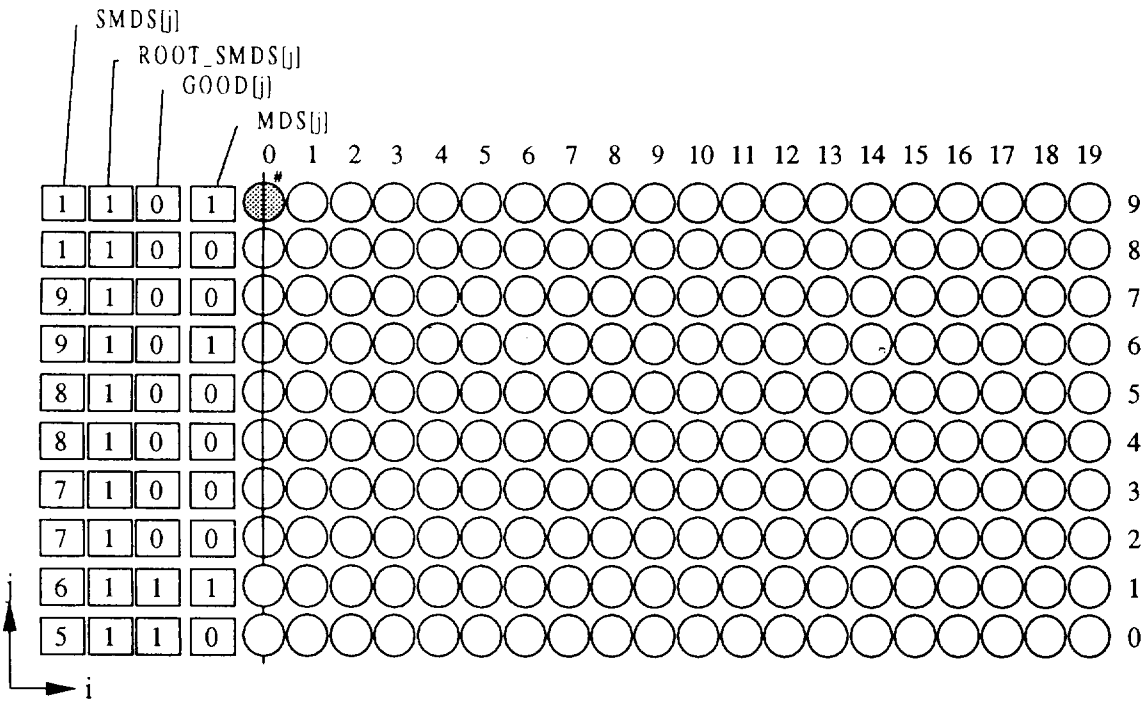


Fig. 9(c) After Steps 3, 4 and 5 of Algorithm for finding the minimum dominating set

Table 2 The values of MDS[j] for Fig. 1

j	0	1	2	3	4	5	6	7	8	9
MDS[j]	0	1	0	0	0	0	1	0	0	1

Find the dominating set on general circular-arc graphs

Note that if arc j doesn't belong to LARGE(A), the dominating set DS[j] formed by arc j will not be the minimum dominating set. This has been proven in [15]. Hence we can find the LARGE(A) first. For example, in Fig. 2, LARGE(A) = {1, 2, 3, 4, 8, 10, 11}. Then we take advantage of the algorithm for proper circular-arc graphs to find the minimum dominating set of LARGE(A) = {1, 2, 3, 4, 8, 10, 11}. Hence we find that {1, 6, 9} is the minimum dominating set of LARGE(A). Then {1, 6, 9} is the dominating set in Fig. 2. It is easy to show that LARGE(A) can be found in O(1) time on a PARBS with $O(n^2)$ processors.

Theorem 3: Given a general circular-arc graph with n arcs, the minimum dominating set problem can be solved in O(1) time on a (2n)*n PARBS.

Conclusion

In the previous literatures, the best sequential algorithm solves this problem in O(n) time [10]. At present, our algorithm is not cost-optimal. Apparently, the gap between our cost and the sequential cost is still large. Hence, in the future, we hope that we can improve the cost to be $O(n^{1+\varepsilon})$, or

O(n log n), while keeping the time complexity in O(1). Additionally, it seems that devising a constant time algorithm to solve the "weighted" version of this problem is difficult. This promises to be a very interesting topic for further research.

References

Apostolico, A. and Hambrusch, S. E. (1987). Finding maximum cliques on circular-arc graphs, *Information Processing Letters, 26,* 209-251.

Arikati, S. R. and Rangan, C. P. (1990). Linear algorithm for optimal path cover problem on interval graphs, *Information Processing Letters, 35,* 149-153.

Booth, K. S. and Johnson, J. H. (1982). Dominating sets in chordal graphs, *SIAM Journal on Computing, 11,* 191-199.

Chang, M. S. and Liu, Y. C. (1993). Polynomial algorithms for the weighted perfect domination problems on chordal graphs and split graphs, *Information Processing Letters, 48,* 205-210.

Chen, Y. C. and Chen, W. T. (1994). Constant time sorting on reconfigurable mesh, *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems,* 43, 749-751.

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

Gupta, U. I., Lee, D. T. and Leung, J. Y. T. (1982). Efficient algorithms for interval graphs and circular-arc graphs, *Networks, 12,* 459-467.

Hsu, W. L. (1985). Maximum weight clique algorithms for circular-arc graphs and circle graphs, *SIAM Journal on Computing, 14,* 224-231.

Lee, C. F. The Design and Researches of Constant-Time Algorithms on Circular-Arc Graphs and Interval Graphs, Master Thesis, Department of Information and Computer Education National Taiwan Normal University, Taiwan, 1998.

Hsu, W. L. and Tsai, K. H. (1991). Linear time algorithms on circular-arc graphs, *Information Processing Letters, 40,* 123-129.

Lin, S. S. (1996). Constant-time algorithms for minimum circle-cover problem on circular-arc graphs, *Bull. National Taiwan Normal University, 41,* 133-176.

Manacher, G. K. and Mankus, T. A. (1996). Finding a domatic partition of an interval graph in Time O(n), *SIAM Journal on Discrete Mathematics, 9,* 167.

Olariu, S., Schwing, J. L. and Zhang, J. (1995). Interval graph problem on reconfigurable meshes, *ORSA Journal on Computing, 7,* 333-348.

Yu, M. S., Chen, C. L. and Lee, R. C. T. (1989). Optimal algorithms for the minimum dominating set problem on circular-arc graphs, *Proc. First Ann. IEEE Symp. on Parallel and Distributed Computing,* 3-10.

Yu, M. S. (1989). Parallel algorithms on circular-arc graphs, Ph.D Thesis, Department of Computer Science and Information Engineering, National Taiwan University, Taiwan.

收稿日期：民國87年9 月1 日
修正日期：民國87年11月26日
接受日期：民國87年11月26日

環弧圖上支配問題之常數時間演算法

林順喜 李青芳

國立台灣師範大學資訊教育研究所

在本篇論文中，我們利用可重組態匯流排之處理器陣列(PARBS, processor arrays with reconfigurable bus systems)具有在O(1)時間中動態連結內部不同的匯排流之特性，在常數時間內解決在環弧圖(circular-arc graphs)上的支配問題(the dominating problem)。關於環弧圖上的支配問題，以前尚未有人能在常數時間內解決，即使是在理想的PRAM模型上也是如此，在本論文中，我們改良Yu [14] [15]中提到的相關演算法，然後再自行發展出一套理論，並且將之推廣至可重組態匯流排之處理器陣列模型上。我們以$O(n^2)$個處理器之可重組態匯流排處理器陣列作為主要的計算模型，使得我們能在常數時間內解決支配問題。

關鍵詞：環弧圖、支配問題、可重組態匯流排之處理器陣列