Indian J. pure appl. Math., **12**(1) : 41–54, January 1981

PARTITIONING ACYCLIC STOCHASTIC PRECEDENCE GRAPH

A. K. CHAKRAVARTY

School of Mathematics, University of Science, Penang, Malaysia

(*Received* 6 *October* 1979; *after revision* 17 *September* 1980)

The problem of partitioning an acyclic directed graph with stochastic transition from node to node is very important in many different fields such as computer program software and logic circuit design. We analyze the problem by creating segments (partitions) of the graph in such a way that on any route from the initial to the terminal node on the graph (a) no segments are accessed more than once and (b) the expected number of segments accessed is a minimum.

1. INTRODUCTION

Consider an acyclic directed graph $G = (N, A)$, where $N = \{N_i\}$, is the set of all nodes and $A = \{a_{ij}/i \neq j)$, is a set of all directed arcs between nodes N_i and N_j. Note that $a_{ij} \in A$ only if N_j can be reached from N_i without reaching any other node.

We define a partition (S, T) of the graph G such that

$$S, T \subset N$$
$$S \cup T = N$$
$$S \cap T = \phi$$

We call such a partition feasible if for any node $N_i(N_i \in S)$ and $N_j(N_j \in T)$, N_j does not precede N_i, i.e., there is no directed arc from N_j to N_i (arc $a_{ji} \notin A$).

Such a feasible partition will be known as a feasible segment. If the feasible segments minimize or maximize a certain objective function then they are also optimal.

The above problem becomes stochastic when, although the precedence relation and hence the possible successors of nodes N_i are properly defined, the exact route taken from N_i(i.e. transition from N_i to one of the possible successor node) depends on the state of the system at the Node N_i. Examples of such problems are a computer program and a logic circuit among others. We elaborate the problem further for a computer program.

A computer program is a set of instruction which can be represented by the nodes on a graph. The logic of the program connects these nodes with directed

*Present address: School of Business and Organizational Sciences, Division of Management, Florida International University, Tamiami Trail, Miami, Florida 33199.

arcs so that instructions are executed in a certain sequence. However, since the program is only a general representation of the problem not all the instructions in a program need be executed in any processing run of the program. That is, any path from the initial node to the final node is a possible sequence of operation and as shown in Fig. 1 only instructions 1, 2, 5 and 6 will be executed, as a possible sequence of operations, in a particular run. Branching of control occurs at node 2 (and node 3) such that the next instruction to be executed would be 4 or 5. The actual route from node 2, however, depends on the state of data, after the instruction 2 is executed. This can be expressed by the transition possibilities p_{24} and p_{25}, i.e., the probability of transition to node 4 or node 5, given the system is at node 2.

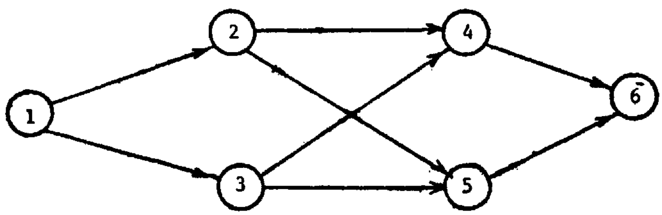


FIG. 1.

Now each of the instruction of a computer program requires storage space in the main memory and since this space, which may also have to be shared by other programs, is expensive and limited, partitioning of the program in smaller segments becomes necessary so as not to exceed the stipulated storage limit.

Usually, the whole program is stored in the cheap mass memory and transporated to the main memory one at a time for execution, the number of segments required to be transported depending on the branching probabilities at the nodes and the way the segments are constructed (feasibility). Obviously, it is desired that the number of segments transported to the main memory is minimized (since all the segments may not be required for a particular run) but at the same time the segments should not be too large to exceed the specified segment size limit.

The partitioning problem as defined above is modelled, first, by establishing an algorithm for the generation of feasible segments and then regrouping them using dynamic program to minimize the expected number of segments accessed in a run.

In what follows we have used the words "partition" and "segment" synonymously.

2. Generation of Feasible Partition

Note : $N_i \prec N_j$ implies "N_i precedes N_j" as defined in section 1.

Definitions

Let $E_0 = N_e$, the initial node and let $E_i(s)$ be any other set of nodes (to be defined) at stage i and state s, i.e., the path connecting N_e and any node in $E_i(s)$ passes through exactly $i - 1$ other nodes. In other words, any node in $E_i(s)$ can be reached from N_e in exactly i "transitions". The state s of $E_i(s)$ can be defined as a possible choice of nodes from all the nodes that can be reached in i transitions from N_e. The state variable is defined more rigorously using power set transformation shortly.

Let $C_0 = E_0$ and let $C_i(s)$ be a feasible segment as defined in section 1, i.e., for all $N_i \in C_i(s)$ and $N_j \in N - C_i(s)$, N_j is not contained in nor equal to N_i. The $C_i(s)$ and $E_i(s)$ can then be linked by the following recurrence relation,

$$C_{i+1}(k) = C_i(s) + E_{i+1}(k) \qquad \text{...(1)}$$

Thus if C_0 and all the $E_i(s)$ are known, all the $C_i(s)$ can be generated. We now demonstrate how to generate the $E_i(s)$ starting from E_0 such that the recurrence relation, above, generates all the feasible segments. (The validity of the procedure of this section is established in two theorems in Appendix I).

Let D_i be a set of successor nodes of E_i at stage i and state s (E_i would have been generated at stage $i - 1$) such that

$$D_i(s) = \{N_k/N_j \prec N_k, \text{ for } N_k \in N \text{ and } N_j \in E_i(s)\}.$$

Let P_j be the set of the predecessor nodes of the node N_j, i.e.

$$P_j = \{N_k/N_k \prec N_j, N_k \in N\}.$$

Let $L_i(s)$ be a subset of $D_i(s)$ such that

$$L_i(s) = \{N_k/N_k \in D_i(s) \text{ and } P_k \subset C_i(s)\}.$$

Then $E_{i+1}(k)$ is the subset of the power set of $L_i(s)$ by the transformation, Γ_k, such that

$$\Gamma_k : L_i(s) = E_{i+1}(k)$$

corresponding to the binary number k in the boolean lattice isomorphic to the power set of $L_i(s)$, i.e., $E_{i+1}(k)$ is the kth choice of x elements ($x = 1, 2, 3, ..., m$) from $L_i(s)$ where $m = | L_i(s) |$; Max $k = 2^m - 1$.

Example — Let $L_{i-1}(s) = (N_1, N_2, N_3)$.

Here $m = 3$, Max $k = 2^3 - 1 = 7$.

Then, $E_i(1) = N_1$, $E_i(2) = N_2$, $E_i(3) = (N_1, N_2)$, $E_i(4) = N_3$, $E_i(5) = (N_1, N_3)$, $E_i(6) = (N_2, N_3)$, $E_i(7) = (N_1, N_2, N_3)$.

The Algorithm

Step 1 : Add dummy nodes N_e and N_t. Connect N_e (directed away from N_e)

to all nodes with no incident arcs. Connect N_t (directed towards N_t) to all nodes which have no arcs emerging from them.

Step 2 : Set $i = 0$, $E_0(1) = C_0(1) = N_e$, $s = 1$.

Step 3 : Let the set $Z = E_i(s)$, label $E_i(s)$.

Step 4 : Generate set D, the successor nodes of the node(s) in set Z.

Step 5 : Generate L, a subset of D, such that the nodes in D whose predecessors are not contained in $C_i(s)$ are excluded from L. Let the number of nodes in $L = m$.

Step 6 : Select $r(r = 1, 2, \ldots, m)$ nodes from L at a time. Denote the kth choice of r nodes as $E_{i+1}(k)$, $k = 1, 2, \ldots, 2^m - 1$.

Step 7 : Generate $C_{i+1}(k) = C_i(s) + E_{i+1}(k)$.

Step 8 : If $N_t \in L$, go to Step 11; else go to step 9.

Step 9 : Scan $E_i(k)$, $k = 1, 2, \ldots$. If an unlabelled $E_i(k)$ is found, set $s = k$ and go to step 3.

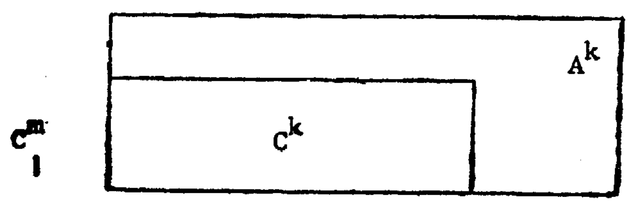
Step 10 : Update i to $i + 1$, reset $s = 1$ and go to step 3.

Step 11 : Terminate.

For a specified example, the $C_i(s)$ are shown in section 4.

3. Optimization

Consider a feasible segment C^m of a graph which is composed of 2 sub-segments C^k and A^k as shown



such that C^k is a feasible segment

$$A^k = C^m - C^k, m = 1, 2, 3, \ldots, M$$

where M = Total number of feasible segments.

It follows that A^k will never precede C^k.

3.1 *Number of Segments Accessed*

Let the probability of reaching $C^m = p_m$.

Let the probability of reaching *only* C^k given that C^m is reached $= p_{k/m}$

Let the probability of reaching *only* A^k given that C^m is reached $= p_{A^k/m}$

$\therefore$ Probability of reaching both C^k and A^k given that C^m is reached $= 1 - (p_{k/m} + p_{A^k/m})$

Let $n(C^m) =$ Expected number of segments need to be accessed given that C^m is accessed

i.e. $n(C^m) = E\,[\text{Segments accessed}/C^m \text{ is reached}]$

$$\therefore \quad n(C^m) = p_{k/m} \,.\, n(C^k) + p_{A^k/m} \,.\, n(A^k) + \{(1 - (p_{k/m} + p_{A^k/m})\}\,\{n(C^k) + n(A^k)\}$$

$$= (1 - p_{A^k/m}) \,.\, n(C^k) + (1 - p_{k/m}) \,.\, n(A^k).$$

Let the value of k be such that

Size of $A^k \leqslant$ size constraint of a segment

$$\therefore \quad n(A^k) = 1$$

$$\therefore \quad n(C^m) = (1 - p_{A^k/m}) \,.\, n(C^k) + (1 - p_{k/m}).$$

Now $1 - p_{A^k/m} =$ Probability of reaching C^k given that C^m is reached

and $1 - p_{k/m} =$ Probability of reaching A^k given that C^m is reached

$$\therefore \quad n(C^m) = P(\text{reaching } C^k/C^m \text{ is reached}) \,.\, n(C^k) + P(\text{reaching } A^k/C^m \text{ is reached})$$

which can be written as

$$n(C^m) = P(C^k/C^m) \,.\, n(C^k) + P(A^k/C^m).$$

In the context of the whole graph, let $N(C^m)$ denote the expected number of segment accesses contributed by C^m

$$\therefore \quad N(C^m) = P(C^m) \,.\, n(C^m)$$

where

$$P(C^m) = p_m.$$

$$\therefore \quad N(C^m) = P(C^m)\,[P(C^k/C^m) \,.\, n(C^k) + P(A^k/C^m)]$$

$$= P(C^k \cap C^m) \,.\, n(C^k) + P(A^k \cap C^m).$$

But since $C^k \subset C^m$ and $A^k \subset C^m$,

$$P(C^k \cap C^m) = P(C^k) = p_k$$

$$P(A^k \cap C^m) = P(A^k) = p_{A^k}$$

$$\therefore \quad N(C^m) = P(C^k) \, . \, n(C^k) + P(A^k)$$

$$= N(C^k) + P(C^m - C^k).$$

If C^m represents the whole graph, then the problem can be simply stated as Minimize $N(C^m)$, which is the same as

$$\text{Minimize } N(C^k) + P(C^m - C^k) \qquad \ldots(2)$$

where $C^k \in I(C^m)$ and $I(C^m)$ is the set of all feasible segments C^k, which are inclusive in C^m and which do not render $C^m - C^k$ too large.

The problem of minimization of the recurrence relation (2) can be solved by Bellman's dynamic programming technique.

All the segments $A^k = C^m - C^k$ thus generated for the optimum solution will be the optimal segments.

4. Example

Consider the example shown in Fig. 2. The size of each node and the transition probabilities from node i to j are labelled on the diagram. Assume that the upper limit of the segment size is 1500 units.

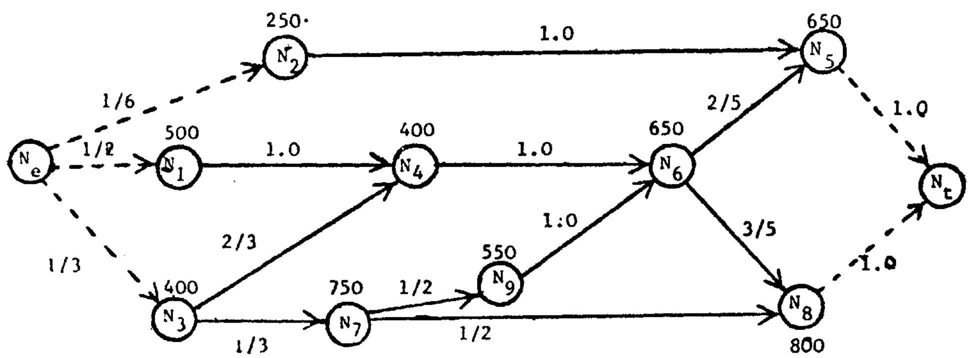


Fig. 2.

First the feasible segments (so as not to violate the precedence relation) are generated as shown in Fig. 3. The size of the feasible segments are labelled in the diagram.

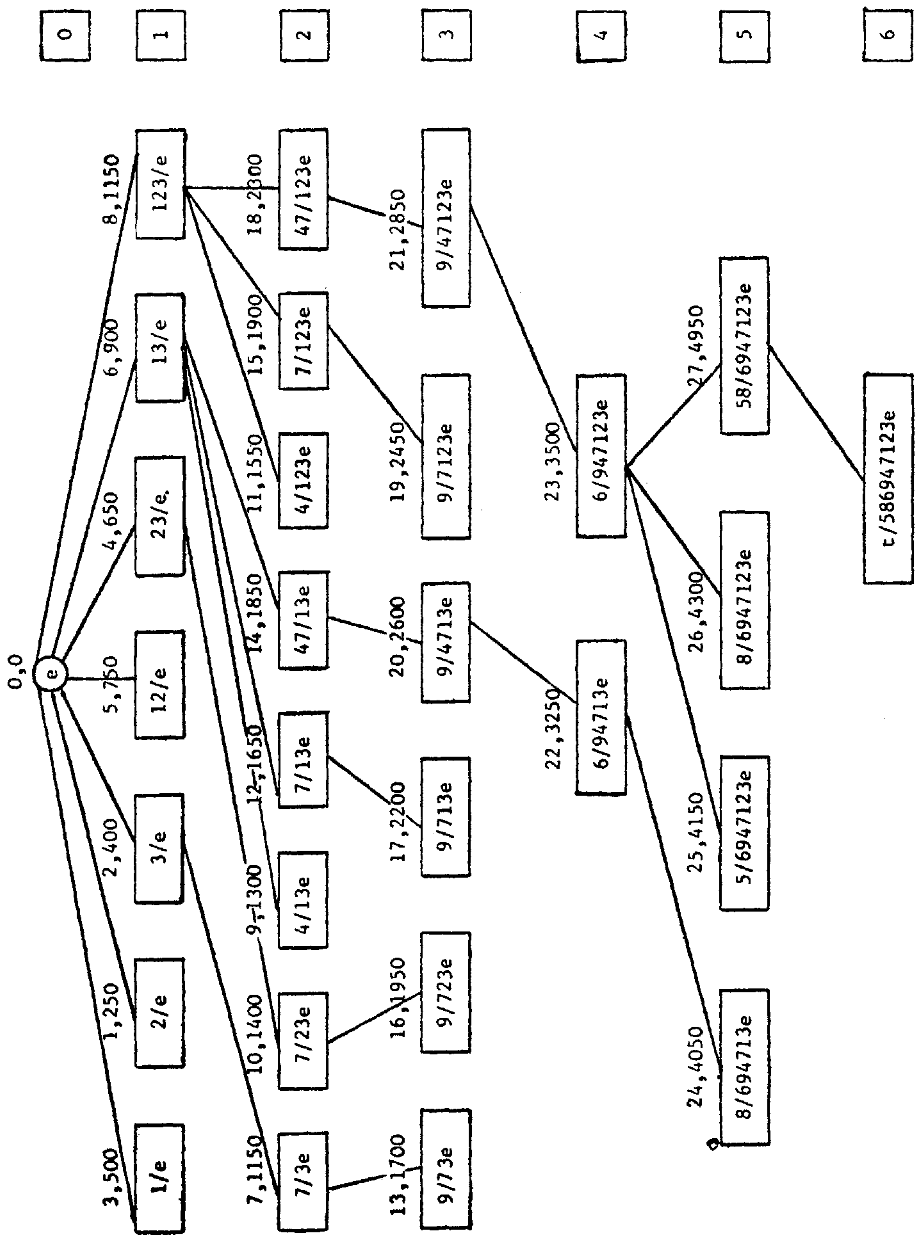

0
1
2
3
4
5
6
0,0
e
8,1150
123/e
6,900
13/e
4,650
23/e
5,750
12/e
2,400
3/e
1,250
2/e
3,500
1/e
18,2300
47/123e
15,1900
7/123e
11,1550
4/123e
14,1850
47/13e
12,1650
7/13e
9,1300
4/13e
10,1400
7/23e
7,1150
7/3e
21,2850
9/47123e
19,2450
9/7123e
20,2600
9/4713e
17,2200
9/713e
16,1950
9/723e
13,1700
9/73e
23,3500
6/947123e
22,3250
6/94713e
27,4950
58/6947123e
26,4300
8/6947123e
25,4150
5/6947123e
24,4050
8/694713e
t/586947123e

FIG. 3.

Explanatory notes relating to the tree-diagram, Figure 3.

1. The node N_i is represented by $i (i = 1, 2, 3, \ldots)$
2. The cluster of nodes in a rectangle represents a feasible segment.
3. The set $E_{i+1}(k)$ is separated from $C_i(s)$ as $E_{i+1}(k)/C_i(s)$. For example, the feasible segment 4, 7/1, 3, *e* has

$$E_2(k) = (4, 7)$$
$$C_1(s) = (1, 3, e).$$

4. The numbers in the square on the extreme right represents the stage j of the segment.
5. The numbers on top of the rectangular boxes are the segment index m and its storage size. The segments are indexed in the increasing order of storage size.

The segments C^k which are subsets of a large segment C^m and which do not violate the segment size constraint ($A^k < 1500$) are shown in Table I and are determined as follows :

(a) If the size of $C^m = \alpha$ then all feasible segments of size less than $\alpha - 1500$ are disqualified.

(b) From those remaining, determine whether they are proper sub-sets of C^m or not. If not disqualify.

The reaching probabilities of, $P(C^m - C^k)$ are also shown in Table I. The method of computation of these probabilities are shown in Appendix II.

The dynamic programming calculations are shown in Table II. A few steps of the calculation procedure are shown in Appendix II.

The optimal segments

$$C^9 - C^0, C^{20} - C^9, C^{24} - C^{20}, C^{27} - C^{24}$$

are shown in Fig. 4.

TABLE I

Estimation of $P(C^m - C^k)$, the reaching probabilities

For segments numbered 1 to 10 in Fig. 3, all the inclusive segments are disqualified because of the storage constraint of 1500. The reaching probabilities for these segments are:

Segment No.	1	2	3	4	5	6	7	8	9	10
$P(C^m - C^0)$	1/6	1/3	1/2	1/2	2/3	5/6	1/3	1	5/6	1/2

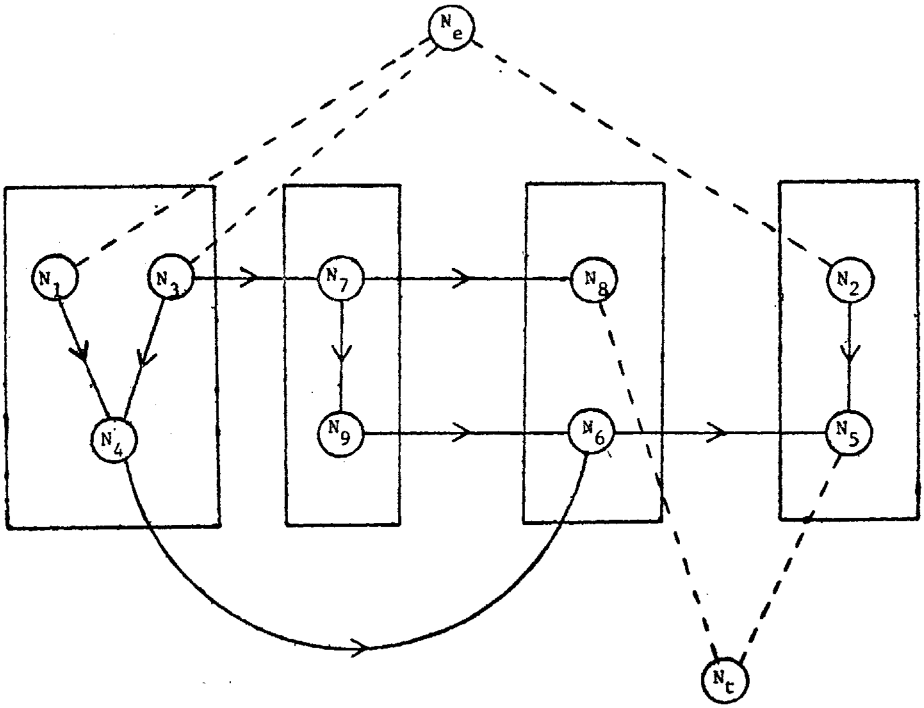


FIG. 4.

For the remaining segments the $P(C^m - C^k)$ values are shown below:

Index No. of Segments $C^m(m)$	Index No. k, of $C^k \in C^m$ such that size of $A^k < 1500$	$P(C^m - C^k)$
11	1, 2, 3, 4, 5, 6, 8, 9	5/6, 16/18, 1, 13/18, 5/6, 16/18, 13/18, 1/6
12	2, 3, 6, 7	11/18, 1/3, 1/9, 1/2
13	2, 7	1/9, 1/18
14	2, 3, 6, 7, 9, 12	15/18, 15/18, 15/18, 13/18, 1/9, 13/18
15	2, 3, 4, 5, 6, 7, 8, 10, 12	14/18, 1/2, 11/18, 1/3, 5/18, 4/6, 1/9, 1/2, 1/6
16	4, 7, 10, 13	1/9, 14/18, 1/18, 1/6
17	6, 7, 12, 13	1/9, 10/18, 1/18, 1/2
18	6, 7, 8, 9, 10, 11, 12, 14, 15	1, 16/18, 15/18, 5/18, 13/18, 1/9, 16/18, 1/6, 13/18
19	7, 8, 10, 12, 13, 15, 16, 17	13/18, 1/9, 10/18, 14/18, 4/6, 1/18, 1/2, 1/6
20	7, 9, 12, 13, 14, 17	14/18, 1/9, 14/18, 13/18, 1/18, 13/18
21	10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20	14/18, 1/9, 17/18, 16/18, 4/18, 14/18, 13/18, 16/18, 1/18, 13/18, 1/6
22	14, 17, 20	7/9, 7/9, 7/9
23	17, 18, 19, 20, 21, 22	17/18, 7/9, 7/9, 17/18, 7/9, 1/6
24	20, 22	5/6, 47/90
25	21, 22, 23	17/18, 43/90, 43/90
26	21, 22, 23, 24	5/6, 62/90, 47/90, 1/6
27	23, 24, 25, 26	1, 43/90, 47/90, 43/90

TABLE II

Solution of Min $N(C^m)$

For $m = 1$ to 10, Min $N(C^m) = P(C^m - C^0)$, since the C^m have no inclusive segments in Table I. For $m = 11$ to 27 the values of Min $N(C^m)$ are shown below:

m	$N(C^m) = P(C^m - C^k) + NC^k$	Min $N(C^m)$	K for Min $N(C^m)$
10	1/2	1/2	—
11	1, 11/9, 3/2, 22/18, 9/6, 31/18, 31/18, 1	1	1, 9
12	17/18, 15/18, 17/18, 15/18	15/18	3, 7
13	8/18, 7/18	7/18	7
14	21/18, 24/18, 30/18, 19/18, 17/18, 28/18	17/18	9
15	20/18, 1, 20/18, 1, 20/18, 1, 10/9, 1, 1	1	3, 5, 7, 10, 12
16	11/18, 20/18, 10/18, 10/18	10/18	10, 13
17	17/18, 16/18, 16/18, 16/18	16/18	7, 12, 13
18	33/18, 22/18, 33/18, 21/18, 22/18, 20/18, 31/18, 20/18, 31/18	20/18	11, 14
19	19/18, 20/18, 19/18, 29/18, 19/18, 19/18, 19/18, 19/18	19/18	7, 10, 13, 15, 16, 17
20	20/18, 17/18, 29/18, 20/18, 1, 29/18	17/18	9
21	23/18, 20/18, 32/18, 23/18, 21/18, 32/18, 23/18, 32/18, 21/18, 32/18, 20/18	20/18	11, 20
22	31/18, 30/18, 31/18	30/18	17
23	33/18, 34/18, 33/18, 34/18, 34/18, 34/18	33/18	17, 19
24	32/18, 197/90	32/18	20
25	37/18, 193/90, 208/90	185/90	21
26	175/90, 212/90, 212/90, 175/90	175/90	21, 24
27	255/90, 203/90, 240/90, 218/90	203/90	24

From the computations shown in Tables I and II, the expected number of segments transferred = 203/90.

The accuracy of the above result can be checked as follows. The various routes from N_e to N_t with their probabilities and the number of segments accessed are as below:

Routes	Nodes	Probability	No. of segments accessed
1	N_2, N_5	15/90	1
2	N_1, N_4, N_6, N_5	18/90	3
3	N_1, N_4, N_6, N_8	27/90	2
4	N_3, N_4, N_6, N_5	8/90	3
5	N_3, N_4, N_6, N_8	12/90	2
6	N_3, N_7, N_9, N_6, N_5	2/90	4
7	N_3, N_7, N_9, N_6, N_8	3/90	3
8	N_3, N_7, N_8	5/90	3

$$\therefore \quad E\,(\text{Number hit}) = 1/90\,[15.1 + 18.3 + 27.2 + 8.3 + 12.2 + 2.4 + 3.3 + 5.3] = 203/90.$$

References

Ferrari, D. (1974). Improving locality of critical working set. *CACM*, **17**, No. 11.

Hastings, N. A. J. (1973). Dynamic Programming for Managerial Applications. Butterworth Scientific Publications, New York.

Kerninghan, B. W. (1971). Optimal sequential partition of graphs. *JACM*, **18**, No. 1.

Lowe, T. C. (1970). Automatic segmentation of cyclic program structure based on connectivity and processor timing. *Comm. ACM*, 13, No. 1.

Morin, T. L., and Marsten, R. E. (1976). Branch and bound strategies for dynamic programming. *Op. Res.*, **24**, No. 4.

Appendix I

Theorem 1 — Any set $C_i(s)$ as generated by the algorithm of section 2 is a feasible segment.

Proof (by induction) : Assume that for some i and s, $C_i(s)$ is not a feasible segment. Then, for at least one $N_k \in C_i(s)$ and one $N_j \in \overline{C_i(s)}$, $N_j \prec N_k$.

$$\therefore\ P_k \cap N_j \neq \emptyset. \quad \text{...(1)}$$

Now from the relationship (1) of section 2, $C_i(s) = C_{i-1}(s') + E_i(s)$ $(i = 1, 2, 3, ...)$ where s and s' are defined by

$$\Gamma_s : L_{i-1}(s') = E_i(s). \quad \text{...(2)}$$

Therefore, $N_k \in C_i(s) \Rightarrow N_k \in C_{i-1}(s')$ or $N_k \in E_i(s)$.

Consider the two cases separately

(a) $N_k \in E_i(s)$

This implies, by definition (2), $N_k \in L_{i-1}(s')$

$$\therefore\ P_k \subset C_{i-1}(s') \quad \text{...(3)}$$

From (1) it follows that $P_k \not\subset C_{i-1}(s')$ (not a subset of) which is in contradiction with (3). Therefore, the assumption must be false. Hence $C_i(s)$ is a feasible segment.

(b) $N_k \in C_{i-1}(s')$

This implies that $N_k \in C_{i-2}(s'')$ or $N_k \in E_{i-1}(s')$. Now if $N_k \in E_{i-1}(s')$, then the theorem can be proved as in (a) above and if $N_k \in C_{i-2}(s'')$, then $N_k \in C_{i-3}(s''')$ or $N_k \in E_{i-2}(s'')$ and so on .... If the logical process is carried on to the limit it can be shown that either the theorem is true as proved in case (a) or; $N_k \in C_0 \triangleq \{N_e\}$. But by definition for all N_j, N_j is not contained in nor equal to N_e. Hence the theorem is true.

Theorem 2 — The set of all $C_i(s)$ is the exhaustive set of the feasible segments of the graph (i.e. there cannot exist a feasible segment which is not generated by $C_i(s)$.

PROOF : Let there exist a feasible segment Y^0 which is not generated by $C_i(s)$. Therefore, $Y^0 + \overline{Y^0} = N$. Also, $N_e \in Y^0$ and $N_t \in \overline{Y^0}$.

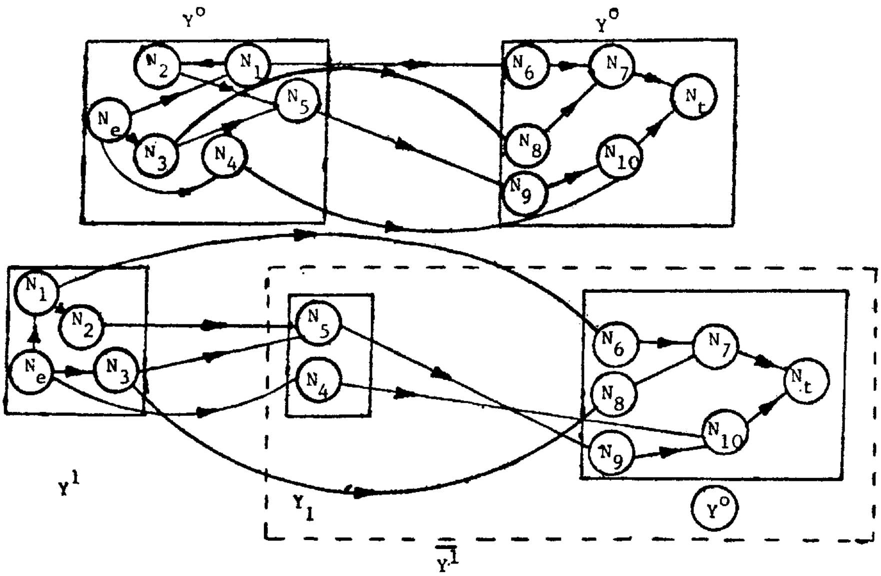


FIG. 5.

Construction

Construct X_1, the set of predecessor nodes such that

$$X_1 = \{N_k / N_k \in Y^0 \text{ and } (\exists N_m \in \overline{Y^0})\ N_k \prec N_m\} \qquad \ldots(4)$$

In the example shown in Fig. 5, $X_1 = (N_1, N_3, N_4, N_5)$.

Construct the subset Y_1 of X_1 (Y_1 is the set of boundary nodes of X_1), such that

$$Y_1 = \{N_k / N_k \in X_1 \text{ and } (\exists N_j \in Y^0)\ N_k \prec N_j\}. \qquad \ldots(5)$$

Then, by construction $Y^1 = Y^0 - Y_1$ is a feasible segment. For the same example, as in Fig. 5, $Y_1 = (N_5, N_4)$ and $Y^1 = (N_1, N_2, N_3, N_e)$. Similarly, by construction $Y^2 = Y^1 - Y_2 = (N_e, N_1)$ is a feasible segment, and in general,

$$Y^r = Y^{r-1} - Y_r \text{ is a feasible segment, for } r = 1, 2, 3, \ldots, n. \qquad \ldots(6)$$

Now, from the construction (5) it follows that $N_e \notin Y_r$ and as $N_e \in Y^0$, $N_e \in Y^r$. As the graph is connected and there is at least one directed chain from N_e to N_j and from N_j to N_t, Y^n, the last feasible segment of the series (6) is given by $Y^n = N_e = C_0$. Therefore, $C_0 = Y^{n-1} = Y_n$ and

$$Y^{n-1} = C_0 + Y_n. \qquad \text{...(7)}$$

From the relation (1) of section 2 it can be written $C_1(k) = C_0 + E_1(k)$. It therefore follows from (7) that either (a) $Y^{n-1} = C_1(k)$ for some k or (b) For some node $N_j \in Y_n$, $N_j \notin E_1(k)$ for any k. Considering case (b), it follows that $N_j \notin L_0$. From the construction of L_i, it also follows that $P_j \notin C_0 = Y^n$. This implies that there is some predecessor of N_j, say N_l, such that either $N_l \in \overline{Y^n}$ or $N_l \in Y_n$. But from the construction (4), it is clear that $N_l \notin \overline{Y^n}$ and similarly from construction (5), $N_l \notin Y_n$. Therefore, for all $N_j \in Y_n \Rightarrow N_j \in L_0$. Thus the deduction (b) above is false. Hence $Y^{n-1} = C_1(k)$, for some k. Similarly, it can be shown that $Y^{n-2} = C_2(k)$, for some k and, in general, $Y^{n-r} = C_r(k)$, ($r \leqslant n$, in the notation). Therefore, when $r = n$, $Y^0 = C_n(k)$, for some k.

APPENDIX II

Calculation of $P(C^m - C^k)$

The probability of reaching the segment $A^k = C^m - C^k$ is given by

$$P(C^m - C^k) = p_{A^k} = \sum_{N_i \in A^k} w_i - \sum_{\substack{N_i \in A^k \\ N_j \in A^k}} w_i \cdot p_{ij}$$

where w_i is the reaching probability of the node N_i, having reached the inital node N_e.

Considering the example in section 4, the reaching probabilities can be calculated as below:

$$w_1 = p_{e1} \cdot w_e,\ w_2 = p_{e2} \cdot w_e,\ \ldots,\ w_6 = p_{46} \cdot w_4 + p_{96} \cdot w_9,\ \ldots \text{ etc.}$$

Now, $w_e = 1$. Using the values of p_{ij} from section 4, the w_i's are,

$$w_1 = \frac{1}{2},\ w_2 = \frac{1}{6},\ w_3 = \frac{1}{3},\ w_4 = 1 \cdot \frac{1}{2} + \frac{2}{3} \cdot \frac{1}{3} = \frac{13}{18},$$

$$w_7 = \frac{1}{3} \cdot \frac{1}{3} = \frac{1}{9},\ w_9 = \frac{1}{2} \cdot \frac{1}{9} = \frac{1}{18},\ w_6 = 1 \cdot \frac{13}{18} + 1 \cdot \frac{1}{18} = \frac{14}{18},$$

$$w_5 = 1 \cdot \frac{1}{6} + \frac{2}{5} \cdot \frac{14}{18} = \frac{43}{90},\ w_8 = \frac{3}{5} \cdot \frac{14}{18} + \frac{1}{2} \cdot \frac{1}{9} = \frac{47}{90}.$$

Calculation of p_{A^k} for a few of the $A^k s$ are shown below:

(a) $A^k = C^{27} - C^{24} = (N_2, N_5)$

$$p_{A^k} = w_2 + w_5 - (w_2 \cdot p_{25} + w_5 \cdot p_{52})$$

$$= \frac{1}{6} + \frac{43}{90} - \left(\frac{1}{6} \cdot 1 + 0\right) = \frac{43}{90}$$

(b) $A^k = C^{23} - C^{17} = (N_2, N_4, N_6)$

$$p_{A^k} = w_2 + w_4 + w_6 - (w_2 \cdot p_{24} + w_2 \cdot p_{26} + w_4 \cdot p_{46} + w_4 \cdot p_{42} + w_6 \cdot p_{62} + w_6 \cdot p_{64})$$

$$= \frac{1}{6} + \frac{13}{18} + \frac{14}{18} - \left(0 + 0 + \frac{13}{18} \cdot 1 + 0 + 0 + 0\right)$$

$$= \frac{17}{18}$$

Solution of the Dynamic Program

The dynamic program can be stated as

$$\underset{C^k}{\text{Minimize}}\ N(C^k) + P(C^m - C^k).$$

Consider C^m, for $m = 23$.

The inclusive sets of C^{23} are

C^{17}, C^{18}, C^{19}, C^{20}, C^{21}, and C^{22}.

The expected numbers of segments hit for various values of k (in C^k) are shown below:

C^k	$N(C^k)$	$P(C^{23} - C^k)$	$N(C^k) + P(C^{23} - C^k)$	
C^{17}	16/18	17/18	33/18	←
C^{18}	20/18	14/18	34/18	
C^{19}	19/18	14/18	33/18	←
C^{20}	17/18	17/18	34/18	
C^{21}	20/18	14/18	34/18	
C^{22}	30/18	3/18	33/18	←

∴ Minimum expected number of segments hit is 33/18 and the possible values of k which would achieve this minimum are 17, 19 or 22.

∴ $N(C^{23}) = 33/18$ and the possible segments are

$C^{23} - C^{17}$ and the optimal segments of C^{17}, or

$C^{23} - C^{19}$ and the optimal segments of C^{19}, or

$C^{23} - C^{22}$ and the optimal segments of C^{22}.

If C^{23} happens to be the largest feasible segment of the program, then the solution would be completed at this stage. If not, the solution steps must continue with C^{24}, C^{25} etc.