Analysis of Graphs by Connectivity Considerations

C. V. RAMAMOORTHY

Honeywell, Inc., Waltham, Massachusetts

Abstract. Discrete sequential systems, such as sampled data systems, discrete Markov processes, linear shift register generators, computer programs, sequential code generators, and prefixed comma-free codes, can be represented and studied in a uniform manner by directed graphs and their generating functions. In this paper the properties of the generating functions are examined from certain connectivity considerations of these graphs.

1. *Introduction*

This paper is a generalization and extension of the writer's earlier paper [9], and some of the results are reviewed here for the sake of continuity. Certain theoretical results are developed that pertain to the existence of strongly connected graphs, disconnected graphs, and reverse graphs, based on their generating and characteristic functions. Also, certain algorithms are developed that (i) test the "well-formation" of the graph with respect to a set of initial and terminal nodes, (ii) determine the inessential nodes, (iii) enumerate and determine all maximal strongly connected subgraphs, (iv) determine the entries and exits of all the maximal strongly connected subgraphs, (v) determine all the link subgraphs, and (vi) partition the graph into its component disjoint subgraphs.

All these algorithms are based on the connectivity matrix of the graph and have been devised with digital computer mechanization in mind; thus such operations as permutation of rows and columns of a matrix are carefully avoided. The connectivity of directed graphs has been considered in [1–6]. Since our approach is based on the generating functions of graphs, the results and methods in most cases are distinct and different, and this makes many physical applications meaningful. Since the generating function is based on a set of starting and terminating points, it provides a direct analog to computer programs and electrical circuits, which are characterized by entry and exit points, and sources and sinks, respectively. Thus the approach here relates the connectivity considerations of directed graphs to the quantitative aspects of physical systems represented by weighted graphs [7–14]. In the parts of this paper relating to the tests, the notion of essential nodes is due to [4], but no explicit tests for determining them were reported there. The method of enumeration and determination of strongly connected subgraphs in this paper is somewhat different from those of others. Our procedures for the determination of entries and exits to these subgraphs, the concept of link subgraphs and the tests to locate them in a connectivity matrix, and our method of partitioning a graph into unconnected subgraphs appear to be original.

2. *Generating Function*

A *weighted directed graph* is a set of nodes (vertices) connected by directed branches (edges) such that a complex number called a *branch transmission* is attached to each branch. Thus a graph of n nodes can be represented by a $n \times n$ branch transmission matrix $\mathbf{T}$, whose ijth element t_{ij} is the branch transmission

Journal of the Association for Computing Machinery, Vol. 13, No. 2 (April, 1966), pp. 211–222

from node i to node j [7, 8]. In many instances, the actual value t_{ij} is not important; it is merely important whether or not it is nonzero. One can define $\mathbf{C}$ as the *connectivity matrix* of a graph whose branch transmission matrix is $\mathbf{T}$ such that its ijth element $c_{ij} = 1$ if and only if its corresponding branching transmission t_{ij} is nonzero; otherwise $c_{ij} = 0$. That is, $c_{ij} = 1$ (or 0) if a branch exists (or does not exist) from node i to node j.

Given directed graphs, whether representing sequential processes, signal flows, or connectivities, there arises a need to know the transmission from one subset of nodes to another subset as a function of the number of branches traversed in series (path lengths). Such a function is called the *generating function* of the given graph between the given subsets of nodes, and one can get many interesting properties of the system by certain operations [7, 8].

3. *Determination of Generating Function*

Given the branch transmission matrix $\mathbf{T}$ of a graph, the generating function $G_{ik}(z) = \sum_{m=0}^{\infty} g_m z^m$ from node i (called the starting node) to node k (called the end node) is given explicitly by $G_{ik}(z) = A_{ki}/|A|$, where $\mathbf{A} = (\mathbf{I} - \mathbf{T}z)$, A_{ki} is the cofactor of the kith element of $\mathbf{A}$, $|\mathbf{A}|$ is the determinant of $\mathbf{A}$ and is nonsingular, and z is a separator variable [7, 8]. $\mathbf{I}$ is an identity matrix. Thus, g_m, the coefficient of z^m, is the total transmission from node i to node k, obtained by traversing exactly m branches in series. It can be shown that $G_{ik}(z)$ can also be obtained by signal flow graph methods [7, 8, 14]. The function $[\mathbf{I} - \mathbf{T}z]$ is called the characteristic function of the weighted graph and does not depend on the choice of the starting or the end nodes. When $\mathbf{A}$ is singular, the weighted directed graph representation is not valid [7]. The concepts of directed graphs and their generating functions have been generalized in [7–14], so that they are applicable to electric networks, discrete Markov processes, simultaneous linear equations, shift register sequences, sequential symbol generators, path enumeration problems, and comma-free codes.

4. *Definitions*

A *path* is a sequence of directed branches[1] connected in series between two nodes. A *loop* is a sequence of nodes $(1, 2, 3, \cdots, k, 1)$ such that a path exists from node 1 to itself. A *subgraph* of given graph of branch transmission matrix $\mathbf{T}$ is defined as consisting of a subset of nodes with all branches between these nodes retained. A node j is said to be *reachable* from node i if there is *at least one* directed path from node i to node j. Equivalently, one may say the $G_{ij}(z) \neq 0$. A node p is *essential* with respect to $G_{ij}(z)$, if (a) it is reachable from node i, and (b) it can reach the terminal node j. A graph consisting of a set of nodes and branches is said to be *strongly connected* if and only if any node is reachable from any other.

If $G_{ij}(z) \neq 0$, the starting node i and the end node j are also considered to be essential. Nodes that are not essential are redundant (inessential).

The *largest* strongly connected subgraph that contains a given node is defined as a *maximal* strongly connected subgraph (M.S.C.) on the graph. It is unique for any given node in its set.

[1] *Specified* directed branches.

THEOREM 1. *The generating function $G_{i,k}(z)$ from node i to node k can be obtained by considering only its essential nodes.*

PROOF. Only a sketch of the proof is presented here. For the sake of simplicity, interpret $G_{i,k}(z)$ as the generating function of path enumeration [7, 8]. Consider Figure 1, where a, b, c, d are redundant nodes.

(a) Consider the case where redundant nodes a and b are not reachable from i but can reach end node k. They cannot influence the number of ways of reaching node k from i.

(b) Consider the case where redundant nodes c and d are reachable from node i, but cannot reach node k. Their presence (or absence) does not influence the number of ways of reaching node j from i.

LEMMA. $G_{ik}(z) = 0$ *if and only if there is no path from node i to node k.*

THEOREM 2. *The largest exponent n in $G_{SE}(z) = \sum_{i=0}^{n} g_i z^i$ is a finite integer if and only if there are no loops in the graphs; otherwise, n is infinite.*

PROOF [8]. If there are no loops, the characteristic function is a constant.

THEOREM 3. *If $G_{SE}(z)$ and $G_{ES}(z)$ are not equal to zero, then nodes E and S are members of the same strongly connected subgraph.*

THEOREM 4. *If $G_{SE}(z)$ and $G_{ES}(z)$ exist ($\neq 0$), then any node essential (redundant) for $G_{SE}(z)$ is also essential (redundant) for $G_{ES}(z)$.*

PROOF. Let i be an essential node for $G_{SE}(z)$. Then i is reachable from S and it can reach E. Since $G_{ES}(z) \neq 0$, there is at least one path from E to S. If i is on this path it is essential. If not, let P_{ES} be a path from E to S. Then i is reachable from E because one can go from E to S (via P_{ES}) and S to i. Thus i can reach S by route i to E and E to S. By definition, i is an essential node of $G_{ES}(z)$.

5. *Graph "Well-Formation"*

When a sequential process is represented by its state diagram and is specified by its connectivity matrix, one would like to test whether there are any structural flaws in it. Two types of flaws can be detected: (a) there may not be a path from the initial or starting node to the terminal or end node, or (b) there may be redundant nodes in the graph with respect to the given set of starting and end nodes. We consider a directed graph to be well formed if (a) there exists a path from the starting node S to the end node E, i.e., $G_{SE}(z) \neq 0$, and (b) there are no inessential nodes in the graph.

6. *Test to Determine $G_{SE}(z) \neq 0$*

Let $\mathbf{C}$ and $\mathbf{C}^t$ be the connectivity matrix of the given graph and its transpose, respectively. Let S and E be the starting and the end nodes, respectively. Then

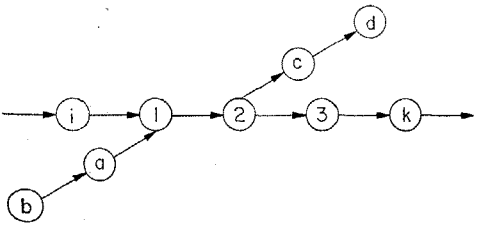


FIG. 1. A graph with inessential nodes

label the rows and columns of these matrices by node numbers 1 through n. S and E are among these nodes.

Step 1. Let $\mathbf{R}_S$ be a row vector of dimension n such that it is initially equal to $\mathbf{C}_S$, the row element of $\mathbf{C}$ corresponding to the starting node S, i.e., $\mathbf{R}_S = \mathbf{C}_S$.

Step 2. Examine the column elements of R_S. Let $(\mathbf{R}_S)_i$ be its ith column element, which is nonzero and not previously examined. Update $\mathbf{R}_S$ by performing a componentwise logical addition to the ith row of $\mathbf{C}$, i.e., $\mathbf{R}_S$ (new) $= \mathbf{C}_i \cup \mathbf{R}_S$ (old).

Step 3. Repeat step 2 above until no more changes take place in $\mathbf{R}_S$.

Step 4. Examine the Eth column of $\mathbf{R}_S$. If it is nonzero, there is at least one path from S to E, which implies that $G_{SE}(z) \neq 0$.

Let us define the final row vector $\mathbf{R}_S$ as the *reachability vector* of node S [2, 4], since a one in its kth column implies that node k is reachable from S. Thus $\mathbf{R}_S$ gives all nodes reachable from S. One can construct the *reachability matrix* $\mathbf{R}$ so that its kth row is the reachability vector of node k. Thus $G_{ik}(z)$ exists if and only if the ikth element of $\mathbf{R}$ is nonzero. This algorithm for obtaining $\mathbf{R}$ is equivalent to the older method [6], $\mathbf{R} = \lim_{N\to\infty} (\mathbf{C} + \mathbf{I})^N$. The new algorithm is efficient because it only involves logical addition operations between row vectors, rather than repeated matrix multiplication.

7. *Determination of Essential Nodes*

First, $\mathbf{R}_S$, the reachability vector of node S, is found by steps 1–4 of Section 6. If $G_{SE}(z) \neq 0$, go to step 5 below. Let $\mathbf{T}_E$ be a row vector of dimension n such that its kth column is nonzero if there is at least one path that can reach node E from node k. Designating $\mathbf{T}_E$ as the *reaching vector* of node E, $\mathbf{T}_E$ is found as follows:

Step 5. Find the reachability vector of node E for the graph whose connectivity matrix is given by $\mathbf{C}^T$. This is $\mathbf{T}_E$.

Step 6. Perform a componentwise logical multiplication between vectors $\mathbf{R}_S$ and $\mathbf{T}_E$, i.e., $\mathbf{R}_S \cap \mathbf{T}_E$. Let the resulting row vector be $\mathbf{L}_{S,E}$. The set of essential nodes are those that correspond to nonzero components of $\mathbf{L}_{S,E}$ together with nodes S and E. The rest are inessential. One can construct the reaching matrix $\mathbf{T}$ by finding the reaching vector for each node.

THEOREM 5. *The reaching matrix* $\mathbf{T}$ *is the transpose of reachability matrix* $\mathbf{R}$, *i.e.*, $\mathbf{T} = \mathbf{R}^T$.

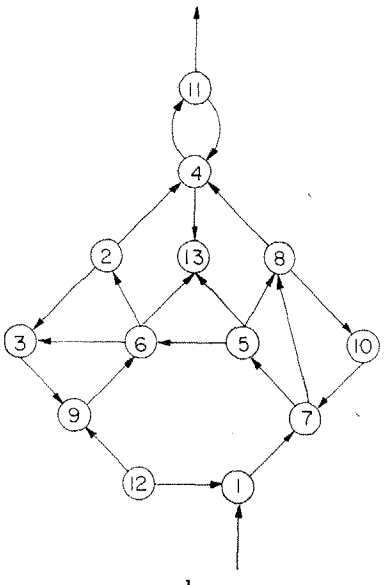
Proof of "Well-Formation." The reachability vector $\mathbf{R}_S$ contains all nodes reachable from S. If its Eth component is nonzero, then there exists a path $C_{si} \cdot C_{ik} \cdots C_{rE} = 1$ for some $i, k, \cdots, r$. Similarly $\mathbf{T}_E$ gives all nodes reaching E. Thus if the ith column of $\mathbf{L}_{SE} = \mathbf{R}_S \cap \mathbf{T}_E$ is not zero, this implies there is a path from S to i and another from i to E. Thus i is an essential node. This proves the algorithm. It could have been computationally simpler if after step 4, all rows and columns of $\mathbf{C}$ corresponding to zeros in $\mathbf{R}_S$ had been deleted and the transpose of the reduced connectivity matrix had been used to find $\mathbf{T}_E$ together with nodes S and E.

Example. Let Figure[2] 2a be the connectivity matrix of a graph, Figure 2b. Let the starting node be 1 and the end node be 11. Test for well-formation and determine all inessential nodes. The diagram (graph) is shown for convenience only.

[2] Underscored letter symbols in figures are same as boldface letters in text.

C =	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	0	0	0	0	0	1	0	0	0	0	0	0
2	0	0	1	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	1	0	1
5	0	0	0	0	0	1	0	1	0	0	0	0	0
6	0	1	1	0	0	0	0	0	0	0	0	0	1
7	0	0	0	0	1	0	0	1	0	0	0	0	0
8	0	0	0	1	0	0	0	0	0	1	0	0	0
9	0	0	0	0	0	1	0	0	0	0	0	0	0
10	0	0	0	0	0	0	1	0	0	0	0	0	0
11	0	0	0	1	0	0	0	0	0	0	0	0	0
12	1	0	0	0	0	0	0	0	1	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0

a



b

FIG. 2a. Connectivity matrix
Correction: The element in the 5th row, 13th column should be 1.
FIG. 2b. A given graph

Let

$$R_1^{(0)} = \mathbf{C}_1 = 0000001000000.$$

Since the 7th element of R_1 is nonzero, logical-add $\mathbf{C}_7$; i.e.,

$$\mathbf{R}_1^{(1)} = R_1^{(0)} \cup \mathbf{C}_7 = 0000101100000, \cdots, \text{etc.},$$

$$R_1(\text{final}) = \text{Reachability Vector of 1} = 0111111111101.$$

Since the 11th column of $\mathbf{R}_1 \neq 0$, $G_{1,11}(z) \neq 0$. Similarly,

$$\mathbf{T}_{11}^{(0)} = \mathbf{C}_{11}^T = 0001000000000.$$

The 4th column is nonzero; thus

$$\mathbf{T}_{11}^{(1)} = \mathbf{T}_{11}^{(0)} \cup \mathbf{C}_4{}^T = 0101000100100, \cdots, \text{etc.},$$

$$T_{11}(\text{final}) = \text{Reaching Vector of 11} = 1111111111110,$$

$$\mathbf{L}_{1,11} = \mathbf{R}_1 \cap \mathbf{T}_{11} = (0111111111101) \cap (1111111111110)$$
$$= (0111111111100).$$

The essential nodes are nodes 1 and 11 *together* with those corresponding to the nonzero columns of $\mathbf{L}_{1,11}$, i.e., nodes 2 through 10. The inessential nodes are 12 and 13.

8. *Strongly Connected Graphs*

THEOREM 6. *A graph is strongly connected if and only if* $G_{ij}(z) \neq 0$ *for all* i *and* j.

PROOF. $G_{ij}(z) \neq 0$ for all i and j if and only if i can reach j and vice versa.

THEOREM 7. *A graph is strongly connected if and only if there exists a node i such that $G_{ii}(z) \neq 0$ and all nodes are essential with respect to $G_{ii}(z)$.*

PROOF. (a) Assume $G_{ii}(z) \neq 0$ and all nodes are essential with respect to $G_{ii}(z)$. Starting at any node k, any other node m is reachable from k via node i.

(b) Assume a graph is strongly connected. Therefore, node i can be reached in one or more steps. Therefore, $G_{ii}(z) \neq 0$. In addition, all nodes can reach node i and can be reached from it. Hence, all nodes are irredundant with respect to $G_{ii}(z)$.

The last theorem is important, since it provides a test for strong connectivity. In addition, it can be used to find the largest subgraph that is strongly connected and that includes a given node (Section 9). The size of the graph is measured here by the number of nodes.

$G_{ij}(z) \neq 0$ if and only if node j is reachable from node i, which can be tested quite simply in the first part of the test in Section 6. The essential nodes again are determined by the algorithm in Section 9; thus the identification of the largest strongly connected subgraph that includes a particular node is quite simple. Theorem 8 below immediately follows from Theorem 7.

THEOREM 8. *A directed graph is strongly connected if and only if, for every node i, $G_{ii}(z) \neq 0$ and all other nodes are essential with respect to $G_{ii}(z)$.*

9. *Test for Strong Connectivity*

Since, by Theorem 6, a directed graph is stongly connected if and only if, for any node i in it, $G_{ii}(z) \neq 0$ and all nodes are essential with respect to $G_{ii}(z)$, one can pick any node i in the graph and apply the well-formation test from node i to itself. If node i is reachable from itself, and if all other nodes are essential with respect to $G_{ii}(z)$, then the diagram is strongly connected. In a nonstrongly connected graph, one can use the above test repetitively with respect to different nodes and determine all M.S.C. subgraphs. All M.S.C. subgraphs are nonintersecting; i.e., they do not have any common nodes. An equivalent test of strong connectivity of a graph will be to find the reachability matrix $\mathbf{R}$; if every element in it is a one, then it is strongly connected.

Example. Let us determine an M.S.C. subgraph that includes node 6 for the given graph of Figure 2b from its connectivity matrix. The starting and end node in this case is 6. Hereafter redundant nodes 12 and 13 are excluded from consideration. The connectivity matrix will be an 11×11 matrix.

By the well-formation test of Section 6, steps 1–4 yield the reachability vector of node 6, $\mathbf{R}_6 = (0\,1\,1\,1\,0\,1\,0\,0\,1\,0\,1)$. Since the 6th column of $\mathbf{R}_6$ is a one, the presence of a loop on the node is confirmed. Performing steps 5 and 6 with respect to end node 6, we get $\mathbf{T}_6 = (1\,1\,1\,0\,1\,1\,1\,1\,1\,1\,0)$. Thus the set of essential nodes with respect to $G_{66}(z)$ is given by the columns corresponding to the ones in $(\mathbf{R}_6 \cap \mathbf{T}_6) = (\mathbf{R}_6 \cap \mathbf{R}_6^T)$, i.e., nodes 2, 3, 6, and 9, which is also the M.S.C. subgraph containing node 6.

10. *Detection of Loops on a Given Node*

The first part of the test detects the presence of loops on a node. If node S is reachable from itself in at least one step, the presence of a loop is indicated. In that case, the reachability vector $\mathbf{R}_S$ will have a one in its Sth column.

11. *Determination of Strongly Connected (S.C.) Subgraphs*

Step 1. Compute the reachability matrix **R**.

Step 2. If any element in its main diagonal is nonzero, the presence of a loop on the corresponding node is confirmed. Otherwise, there are no loops. An equivalent but different test to detect the absence of loops in a graph is given by Marimont [4].

12. *Determination of All Maximal Strongly Connected (M.S.C.) Subgraphs*

Step 1. Construct the reachability matrix **R** for the given graph.

Step 2. Construct the reaching matrix **T** by either taking the transpose of **R** or determining the reachability matrix of $\mathbf{C}^T$.

Step 3. Construct $\mathbf{M} = \mathbf{R} \cap \mathbf{T} = \mathbf{R} \cap \mathbf{R}^T$. The number of M.S.C. subgraphs is given by the number of distinct nonzero row vectors of **M**. If $\mathbf{M}_i \neq \mathbf{0}$, a vector of all zeros, then the nodes of the M.S.C. subgraph correspond to the nonzero columns of $\mathbf{M}_i$. If all distinct nonzero row vectors are treated in a similar way, the memberships of every M.S.C. can be determined. The proof of the above algorithm is simple.

Example. Determine all the M.S.C. subgraphs for the connectivity matrix given in Figure 2a. (Note that nodes 12 and 13 have been excluded.) The reachability matrix **R** and the **M** matrix are obtained using procedures in Section 6.

$$\mathbf{R} = \begin{matrix} 0\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1 \\ 0\,1\,1\,1\,0\,1\,0\,0\,1\,0\,1 \\ 0\,1\,1\,1\,0\,1\,0\,0\,1\,0\,1 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1 \\ 0\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1 \\ 0\,1\,1\,1\,0\,1\,0\,0\,1\,0\,1 \\ 0\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1 \\ 0\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1 \\ 0\,1\,1\,1\,0\,1\,0\,0\,1\,0\,1 \\ 0\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1 \end{matrix} \qquad \mathbf{M} = \mathbf{R} \cap \mathbf{T} = \mathbf{R} \cap \mathbf{R}^T = \begin{matrix} 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,1\,1\,0\,0\,1\,0\,0\,1\,0\,0 \\ 0\,1\,1\,0\,0\,1\,0\,0\,1\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1 \\ 0\,0\,0\,0\,1\,0\,1\,1\,0\,1\,0 \\ 0\,1\,1\,0\,0\,1\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,0\,1\,1\,0\,1\,0 \\ 0\,0\,0\,0\,1\,0\,1\,1\,0\,1\,0 \\ 0\,1\,1\,0\,0\,1\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,0\,1\,1\,0\,1\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,1 \end{matrix}$$

The number of distinct nonzero rows, and hence the number of M.S.C. subgraphs, is equal to 3. The nodes of these M.S.C. subgraphs are {2, 3, 6, 9}, {5, 7, 8, 10}, and {4, 11}. Node 1 is not strongly connected.

13. *Some Properties of* **M** *and* **R** *Matrices*

THEOREM 9. *An S.C. graph has an* **R** *matrix of all ones.*

If the ith *row* (column) of **R** is all ones, then it can reach (can be reached from) all nodes including itself. $\mathbf{R}_{ii} = 1$ implies that there exists an M.S.C. subgraph that contains node i.

THEOREM 10. *If the main diagonal elements of* **R** *are all zero, the graph is loopless.*

THEOREM 11. *If any row of* **R** *and its corresponding column are all ones, then the graph is strongly connected.*

THEOREM 12. *The matrix* $M = \mathbf{R} \cap \mathbf{R}^T$ *is symmetric.*

THEOREM 13. *The number of M.S.C. subgraphs in a graph is given by the number of distinct nonzero rows (columns) of* **M**.

THEOREM 14. $\mathbf{M}_{ii} \neq 0$ *if and only if* $\mathbf{M}_i \neq 0$. Thus $\mathbf{M}_{ii} = 0$ if and only if $\mathbf{M}_i = \mathbf{0}$ (row vector of all zeros). This implies node i is not a member of an M.S.C. subgraph.

THEOREM 15. *The graph is loop-free if and only if* $\mathbf{M} = \mathbf{0}$, *or equivalently* $\mathbf{R}_{ii} = 0$ *for all* i.

THEOREM 16. $\mathbf{M}_{ij} \neq 0$ *if and only if* $\mathbf{M}_i = \mathbf{M}_j \neq \mathbf{0}$, *which implies* i *and* j *belong to the same M.S.C. subgraph.*

14. *Reverse Graphs*

The reverse of a directed graph is obtained by transposing its transmission (connectivity) matrix. The reaching and reachability matrices interchange their roles. Schematically, this implies the change of direction of the directed branches of the graph. The process of reversal leaves the loops intact. The characteristic function of the original graph is invariant under reversal.

THEOREM 17. *For any directed graph* $G_{SE}(z) = \hat{G}_{ES}(z)$, *where* G *and* $\hat{G}$ *are generating functions of the original graph and its reverse.*

$$G_{SE}(z) = \frac{\mathbf{A}_{ES}}{|\mathbf{A}|}, \quad \text{where} \quad \mathbf{A} = [\mathbf{I} - \mathbf{T}z],$$

$$G_{ES}(z) = \frac{A'_{SE}}{|\mathbf{A}'|}, \quad \text{where} \quad \mathbf{A}' = \mathbf{I} - \mathbf{T}^T z,$$

$$\mathbf{A}' = [\mathbf{I}^T - T^T z] = [\mathbf{I} - \mathbf{T}_z]^T = \mathbf{A}^T.$$

Thus,

$$\hat{G}_{ES}(z) = \frac{A^T_{SE}}{|\mathbf{A}^T|} = \frac{A_{ES}}{|\mathbf{A}|} = G_{SE}(z).$$

15. *Unconnected Graphs*

Let $N = \{n_1, n_2, \cdots, n_R\}$ and $M = \{m_1, m_2, \cdots, m_E\}$ be two sets of nodes in a graph. N and M are said to be *unconnected* (disconnected, disjoint) subgraphs if no node from N can reach a node in M, and vice versa.

THEOREM 18. *The subgraphs* M *and* N *are unconnected if and only if* $G_{m,n}(z) = G_{n,m}(z) = 0$ *for all* $m \in M$ *and* $n \in N$.

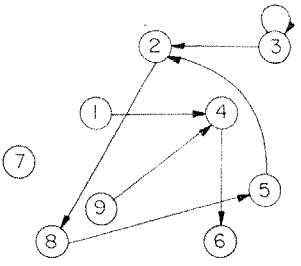
16. *An Algorithm for Partitioning a Given Graph Into Its Unconnected Subgraphs*

When a directed graph is specified by its connectivity matrix $\mathbf{C}$, it is possible to partition it into its unconnected subgraphs rapidly by the following algorithm.

Step 1. For any $\mathbf{C}_{ij} = 1$ in the graph make $\mathbf{C}_{ji} = 1$. Call this new connectivity matrix $\mathbf{C}'$, which is a symmetric matrix. This step makes each disjoint subgraph strongly connected within itself.

Step 2. Pick any node k. Find all the nodes that are reachable from it using the matrix $\mathbf{C}'$ and applying steps 1–4 of Section 6. Let the set of nodes be S_1.

Step 3. Pick a node from the remaining nodes. Repeat step 2 above and find another set of nodes S_2.



a

	1	2	3	4	5	6	7	8	9
1	0	0	0	1	0	0	0	0	0
2	0	0	0	0	0	0	0	1	0
3	0	1	1	0	0	0	0	0	0
4	0	0	0	0	0	1	0	0	0
5	0	1	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	0	0	0	1	0	0	0	0
9	0	0	0	1	0	0	0	0	0

b

	1	2	3	4	5	6	7	8	9
1	0	0	0	1	0	0	0	0	0
2	0	0	1	0	1	0	0	1	0
3	0	1	1	0	0	0	0	0	0
4	1	0	0	0	0	1	0	0	1
5	0	1	0	0	0	0	0	1	0
6	0	0	0	1	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0
8	0	1	0	0	1	0	0	0	0
9	0	0	0	1	0	0	0	0	0

c

FIG. 3a. A given graph
FIG. 3b. A given connectivity matrix = **C**
FIG. 3c. Modified symmetric connectivity matrix = **C**′

Repeat step 3 until all nodes of the graph are taken care of. Then the subsets of nodes $S_1, S_2, \cdots$, form subgraphs of the given graph that are mutually unconnected.

Example. Partition the directed graph shown in Figure 3a into unconnected subgraphs. The connectivity matrix is given in Figure 3b.

Step 1. Set $\mathbf{C}'_{ji} = \mathbf{C}'_{ij} = 1$ if $\mathbf{C}_{ji} = 1$. The matrix shown in Figure 3c is obtained.

Step 2. Pick node 1; all nodes strongly connected with it are (from $\mathbf{C}'$) equal to $S_1 = \{1, 4, 6, 9\}$.

Step 3. Pick node 2; $S_2 = \{2, 3, 5, 8\}$.

Step 4. Pick node 7; $S_3 = \{7\}$.

The given graph $= \{\{7\}, \{1, 4, 6, 9\}, \{2, 3, 5, 8\}\}$, i.e., the disjoint subgraphs are $\{7\}$, $\{1, 4, 6, 9\}$, and $\{2, 3, 5, 8\}$.

17. *Entries and Exits of S.C. Subgraphs and Their Relation to Directed Cut-Sets*

A *directed cut-set* can be defined as a set of those branches whose removal leaves the graph in disconnected subgraphs, such that these branches have their initial nodes in one subgraph and their terminal nodes in another, and such that no proper subset of these branches has this property [1].

It is of interest to determine the directed cut-sets of S.C. subgraphs in a given graph, which translates itself to the problem of finding all the entries or exits to M.S.C. subgraphs.

18. *Determination of Entries and Exits of M.S.C. Subgraphs*

(a) Algorithm to determine the entries:

Step 1. Compute matrix **M**. (See Section 12.)

Step 2. Compute matrix $\mathbf{S} = \overline{\mathbf{M}} \cap \mathbf{C}$.

Step 3. Let $J = j_1, j_2, \cdots, j_r$ be nodes that are not members of any M.S.C. subgraph. These correspond to all-zero rows of **M**. Set *columns* of **S** corresponding to set J to zero. The residual matrix $\mathbf{S}^N$ is the matrix of entries to S.C. subgraphs, i.e., $S^N_{ij} = 1$ if the branch (i, j) enters the M.S.C. subgraph containing j.

(b) Algorithm to determine the exits:

Step 1. Compute the matrix **M**.

Step 2. Compute the matrix $\mathbf{S} = \overline{\mathbf{M}} \cap \mathbf{C}$.

Step 3. Find set J as in step 3 in part (a). Set *rows* of **S** corresponding to nodes J to zero. The resulting matrix $\mathbf{S}^X$ is the matrix of exits such that $\mathbf{S}^X_{ij} = 1$ if and only if the branch (i, j) exits from the S.C. subgraph containing node i.

PROOF OF THE ALGORITHMS. $\mathbf{S} = \overline{\mathbf{M}} \cap \mathbf{C}$ is a matrix such that $\mathbf{S}_{ij} = 1$ if and only if (a) branch (i, j) *exists*, and (b) it either enters or exits from an M.S.C. subgraph or it either enters or exits from nodes that are not members of any S.C. subgraph, viz., $\{J\}$.

By setting column elements corresponding to set $\{J\}$ to zero, we have selected all branches (i, j) such that either nodes i and j are members of two different S.C. subgraphs, or i is an element of $\{J\}$ and j is an element of some S.C. subgraph. This proves the entry algorithm. The proof of the exit determination algorithm is similar.

Example. In the determination of the entries and exits of all S.C. subgraphs of the graph whose connectivity matrix is given in Figure 2, excluding nodes 12 and 13, it is found that $J = \{1\}$. Branches entering the S.C. subgraphs are those S_{ij}'s whose j's are not elements of $J = (1, 7), (5, 6), (2, 4)$, and $(8, 4)$.

Branches going out of S.C. subgraphs are equal to

$$S_{ij}\text{'s}, \quad \text{where } i \notin \{J\}, \text{ or}$$
$$(2, 4), (8, 4), \text{ and } (5, 6).$$

19. *Link Subgraphs*

A *link subgraph* of an M.S.C. subgraph G_1 is a set of *all* nonstrongly connected (N.S.C.) nodes such that there exists at least one path through them that emanates from G_1 and terminates in another M.S.C. subgraph.

In Figure 4 the subgraph containing the nodes 2, 3, 4, 5 is a link subgraph. When the source node (the end node) is not strongly connected, it can be assumed that there is a directed branch reaching (emanating from) it from (into) an M.S.C. subgraph. From these principles we note that the sequential flow from the start node to the end node goes from one M.S.C. subgraph to another either directly or via a link subgraph.

20. *Determination of the Link Subgraphs*

Step 1. Determine the set of all the N.S.C. nodes.

Step 2. Derive their reachability matrix $\mathbf{R}^{\text{N.S.C.}}$ *from their* connectivity matrix $\mathbf{C}^{\text{N.S.C.}}$.

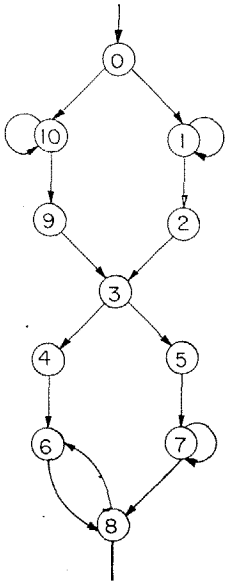


FIG. 4. A given graph

$\underline{C}^{N.S.C.}$

=	0	2	3	4	5	9
0	0	0	0	0	0	0
2	0	0	1	0	0	0
3	0	0	0	1	1	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0
9	0	0	1	0	0	0

a

$\underline{R}^{N.S.C.}$

	0	2	3	4	5	9	Row Wt.
0	1	0	0	0	0	0	1
2	0	1	1	1	1	0	4
3	0	0	1	1	1	0	3
4	0	0	0	1	0	0	1
5	0	0	0	0	1	0	1
9	0	0	1	1	1	1	4

b

FIG. 5a. N.S.C. connectivity matrix
FIG. 5b. $\mathbf{R}^{N.S.C.}$

Step 3. Make $\mathbf{R}_{ii}^{N.S.C.} = 1$ for all i.

Step 4. Count the number of ones in each row. Find the row $\mathbf{R}_i^{N.S.C.}$ with the largest weight. The nonzero columns of $\mathbf{R}_i^{N.S.C.}$ are the nodes in the link subgraph that starts at node i.

Step 5. Ignore all rows corresponding to nodes selected in step 4. Find from the remaining rows another row $\mathbf{R}_j^{N.S.C.}$ with the largest row weight. Its nonzero columns are the nodes in a link subgraph that starts at node j. Repeat step 5 until all rows are selected in $\mathbf{R}^{N.S.C.}$.

The proof of this algorithm is simple. The branching paths in a link subgraph of a given M.S.C. subgraph are given by its component N.S.C. node whose row weight in the N.S.C. connectivity matrix exceeds one.

Example. Find all the link subgraphs from the connectivity matrix of Figure 4. Using the previous tests, the N.S.C. nodes are 0, 2, 3, 4, 5, and 9. The connectivity matrix of N.S.C. nodes is given in Figure 5a.

The reachability matrix $\mathbf{R}^{N.S.C.}$ (Figure 5b) is derived from $\mathbf{C}^{N.S.C.}$. Also, $\mathbf{R}_{ii}^{N.S.C.}$ has been made equal to 1.

Referring to Figure 5b, select row 2 of $\mathbf{R}^{N.S.C.}$ to get the link subgraph {2, 3, 4, 5} emanating from M.S.C. subgraph {1}. Select row 9 to get the link subgraph {3, 4, 5, 9} emanating from M.S.C. subgraph {10}. Finally, row 0 gives the link subgraph {0}.

Acknowledgment. This paper is extracted from Chapter 6 of the writer's Doctoral thesis at Harvard. He wishes to acknowledge the help and encouragement he received from Professor D. W. Tufts of Harvard and Dr. R. B. Lawrance of Honeywell, Inc.

RECEIVED NOVEMBER, 1964; REVISED SEPTEMBER, 1965

REFERENCES

1. CHEN, Y. C. AND WING, O. Connectivity of directed graphs. Proc. of Allerton Conf. on Circuit and System Theory, U. of Illinois, Sept. 1964.
2. HARARY, F. Some historical and intuitive aspects of graph theory. *SIAM Rev. 2*, 2 (1960), 123–131.
3. —— ET AL. *Structural Models*. John Wiley & Sons, New York, 1965.
4. MARIMONT, R. B. A new method of checking the consistency of precedence matrices. *J. ACM 6* (1959), 164–171.
5. ——. Applications of graphs and Boolean matrices to computer programming. *SIAM Rev. 2*, 4 (1960), 259–268.
6. PROSSER R. Applications of Boolean matrices to the analysis of flow diagrams. Proc. Eastern Joint Comput. Conf. 1959. (Now available from Spartan Books, Washington, D.C.)
7. RAMAMOORTHY, C. V. Doctoral thesis, Harvard U., May 1964.
8. ——. Representation and analysis of discrete systems using generating functions of abstract graphs. IEEE Int. Conv. Rec. 1965, Pt. 6.
9. ——. Connectivity considerations of graphs representing discrete sequential systems. *Trans. IEEE EC-14* (Oct. 1965), 724–727.
10. ——. A unified approach for solving quantitative problems in discrete systems by generating functions of abstract graphs. Proc. IFIP Congr. 1965, Vol. 2, MacMillan Co., New York, 1965.
11. ——. Transitions in multi-parameter multi-level discrete systems. IEEE Publ. 4C9, Proc. IEEE Symp. on Signal Transmission and Processing, 1965.
12. ——. Discrete Markov analysis of computer programs. Proc. ACM 20th Nat. Conf., 1965, pp. 386–392.
13. ——. Signal coding for transmission and resolution by graph theoretic methods. Proc. First Annual IEEE Communications Conv., 1965, pp. 193–199.
14. —— AND TUFTS, D. W. Reinforced comma-free codes. Cruft Lab. Rep. No. 480, Harvard U., Aug. 1965; also Paper, IEEE Int. Symposium on Inf. Theory, 1966.