AN APPLICATION OF REGULAR ALGEBRA TO THE ENUMERATION OF CUT SETS IN A GRAPH

Alberto Martelli

Nota Interna B73-20

Dicembre 1973

ABSTRACT

Many path-finding problems have been formulated in a suitable algebra and, in terms of this algebra, they have been reduced to the solution of a system of linear equations. In this paper, these results are extended to the problem of enumerating all minimal i-j cut sets between all pairs of nodes in a directed graph. An i-j cut set is a set of arcs such that, by removing these arcs, there is no path from node i to node j. A definition of sum and multiplication is given, in such a way that the problem can be represented by a system of linear equations. Gaussian elimination provides an efficient solution of this system.

1. Introduction

Several path-finding problems for graphs have been formulated in a suitable regular algebra, i.e. an algebra satisfying the axioms of regular expressions [1]. For example, regular algebras have been given for finding the shortest path between all pairs of nodes in a weighted graph, for finding the transitive closure of a directed graph and for finding all simple paths [2,3]. In terms of a regular algebra, the problem can be posed as that of solving a system of linear equations, and it was demonstrated that such equations can be solved by variants of classical methods of linear algebra. For example, both Floyd's method for finding the shortest path [4] and Warshall's algorithm for finding the transitive closure [5] can be interpreted as a solution of the given system through Gaussian elimination.

This paper shows that these concepts can be applied to problems other than path-finding problems and, precisely, to the problem of enumerating all i-j cut sets between all pairs of nodes i and j in a graph. An i-j cut sets is a set of arcs such that, if these arcs are removed from the graph, there is no path between node i and j. The enumeration of all i-j cut sets is often the first step of important procedures, like, for example, the computation of the terminal reliability in a communication network [6,7,8].

In sections 2 and 3 we introduce the basic concepts of regular algebra and explain a few simple methods for solving a system of linear equations. In section 4 we give the algebra for cut sets and we show that the problem of enumerating all i-j cut sets can be reduced to that of solving a system of linear equations in this algebra. Finally, in section 5, we discuss how to implement our method in an efficient way and make some comparisons with other methods which have been proposed for solving the same problem.

2. Regular Algebra

A regular algebra $R = (S, +, ., ^*)$ [1,3] consists of a set S on which are defined two binary operations, addition and multiplication [1], and one unary operation *, and for which the following axioms are valid:

A1 $(\alpha + \beta) + \gamma = \alpha + (\beta + \gamma)$

A2 $\alpha(\beta\gamma) = (\alpha\beta)\gamma$

A3 $\alpha + \beta = \beta + \alpha$

A4 $\alpha(\beta + \gamma) = (\alpha\beta) + (\alpha\gamma)$

A5 $(\alpha + \beta)\gamma = (\alpha\gamma) + (\beta\gamma)$

A6 $\alpha + \alpha = \alpha$

---

(1) Multiplication of α and $\beta \in S$ will be denoted by $\alpha\beta$

where $\alpha, \beta, \gamma \in S$.
The set S contains a zero element $\emptyset$ such that

A7 $\alpha + \emptyset = \alpha$
A8 $\emptyset \alpha = \emptyset = \alpha \emptyset$

By denoting $\emptyset^*$ by e, we have

A9 $e \alpha = \alpha = \alpha e$
A10 $\alpha^* = e + \alpha \alpha^*$
A11 $\alpha^* = (e + \alpha)^*$

We can define a partial ordering $\leq$ on the set S by

$$\alpha \leq \beta \iff \alpha + \beta = \beta$$

Finally, the following rule of inference is valid:

R1 $\alpha = \beta \alpha + \gamma \Rightarrow \alpha \geq \beta^* \gamma$

This rule of inference states that $\alpha = \beta^* \gamma$ is the minimal solution of the equation $\alpha = \beta \alpha + \gamma$.

The above axiom system was originally defined for regular expressions [1].

Given any regular algebra R, we can form a new regular algebra consisting of all n x n matrices whose elements belong to R. In this algebra addition and multiplication have the usual meaning, i.e., if $A = [a_{ij}]$ and $B = [b_{ij}]$ are two n x n matrices, then

$$A + B = [a_{ij} + b_{ij}] \quad \text{and} \quad AB = \left[\sum_{k=1}^{n} a_{ik}\, b_{kj} \right]$$

Moreover, $A \leq B$ if and only if $a_{ij} \leq b_{ij}$ for all i,j. The <u>unit matrix</u> $E = [e_{ij}]$ is the n x n matrix with $e_{ij} = e$ if $i = j$ and $e_{ij} = \emptyset$ if $i \neq j$. The <u>zero matrix</u> is a matrix all of whose entries are $\emptyset$. The powers of a matrix are

$$A^{o} = E, \; A^{k} = A^{k-1} A \qquad (k = 1,2,...)$$

Finally, the <u>closure</u> of A is

$$A^{*} = \sum_{k=o}^{\infty} A^{k}$$

It can be verified that this algebra of n x n matrices is a regular algebra, i.e. that axioms A1 - A11 and rule of inference R1 are valid.

An n x n matrix $A = [a_{ij}]$ can always be visualized as a n-node labelled directed graph as follows. If $a_{ij} \neq \emptyset$, there is an arc from node i to node j whose label is a_{ij}, otherwise, if $a_{ij} = \emptyset$, there is no arc from node i to node j. A <u>path</u> p_{ij} from node i to node j in this graph is a sequence of arcs $(i\ i_1)\ (i_1\ i_2)...(i_m\ j)$, where the first node of each arc is equal to the second node of the preceding arc. The <u>length</u> of a path is the number of its arcs. The <u>path product</u> $w(p_{ij})$ of the path p_{ij} is the product of the labels of the arcs belonging to p_{ij}:

$$w(p_{ij}) = a_{i,i_1}\ a_{i_1,i_2} \cdots a_{i_m,j}$$

The element a^r_{ij} of the matrix A^r corresponds to the sum of the path products of all paths from node i to node j of length r. For example, the following matrix

$$A = \begin{vmatrix} \emptyset & c & \emptyset & d \\ \emptyset & \emptyset & \emptyset & f \\ a & b & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \end{vmatrix}$$

is represented by the graph in Fig. 1. We have $a^2_{34} = ad + bf$, which correspond to the two paths (3 1) (1 4) and (3 2) (2 4) of length 2.

In many cases, we have an algebra for which the additional axiom

A12 $\quad e + \alpha = e$

holds. As a consequence of this axiom we have $\alpha^* = e$ and the operator * can be eliminated from the algebra. Let us denote such an algebra by R_Q. Its axioms are given by A1-A9, A12.

If we now consider the algebra of all n x n matrices over R_Q, the following property of the closure of a matrix can be proved:

$$(2.1) \qquad A^* = E + A + A^2 + \dots + A^r = E + A + A^2 + \dots A^{n-1}$$
$$\text{for all } r \geq n-1$$

This algebra has been applied to several path-finding problems, such as the determination of shortest paths or the enumeration of all simple paths on a directed graph [2,3].

3. Finding the closure of a matrix

Once a path-finding problem has been formulated in a regular algebra, it reduces to the problem of determining the closure A^* of a matrix A, whose elements belong to the given algebra. By inference rule R1 we know that A^* is the minimal solution of the equation

$$Y = AY + E .$$

This equation suggests an analogy with linear algebra and, in fact, it has been shown [3] that it is possible to define algorithms for solving it, similar to the methods of linear algebra.

In this section we shall consider only the case of n x n matrices over a regular algebra R_Q, defined in the previous section. One well-known method of obtaining A^* in this case, is to set M = E + A and then to compute successively M^2, M^4, ..., M^{2^r}, where r is the first integer such that $2^r \geq n-1$. We know, by (2.1), that $M^{2^r} = A^*$.

Another way of obtaining A^* is Gaussian elimination, for which the following algorithm can be given.

<u>Algorithm G</u>

(1) Set

$$b^o_{ij} = a_{ij} \qquad (i,j = 1,\dots,n;\ i \neq j)$$

$$b^o_{ii} = e \qquad (i = 1,\dots,n)$$

(2) Repeat next step for h = 1,...,n

(3) Set

$$b^h_{ij} = b^{h-1}_{ij} + b^{h-1}_{ih}\ b^{h-1}_{hj} \qquad (i,j = 1,\dots,n;\ i \neq h;\ j \neq h)$$

$$b^h_{ij} = b^{h-1}_{ij} \qquad \text{otherwise}$$

(4) Set

$$a^*_{ij} = b^n_{ij} \qquad (i,j = 1,\dots,n)$$

Notice that the elements on the main diagonal are initially set to e, and they always keep the same value because of axiom A12.

This algorithm has been used in [2] for enumerating all simple paths in a graph and it is practically equivalent to Floyd's algorithm for finding the shortest path between all pairs of nodes in a weighted graph [4].

If only a submatrix of matrix A^* is required, a more economical version of algorithm G can easily be obtained. For example, the following algorithm will compute the submatrix $[a^*_{ij}]$ with $r \leq i \leq n$, $c \leq j \leq n$.

<u>Algorithm G'</u>

(1) Set

$$b^o_{ij} = a_{ij} \qquad (i,j = 1,\dots,n;\ i \neq j)$$

$$b^o_{ii} = e \qquad (i = 1,\dots,n)$$

(2) Repeat next step for $h = 1,\dots,n$

(3) Set

$$b^h_{ij} = b^{h-1}_{ij} + b^{h-1}_{ih}\, b^{h-1}_{hj} \qquad \begin{pmatrix} \text{if } h \leq r \text{ then } h < i \leq n \\ \qquad \text{else } r \leq i \leq n;\ i \neq h \\ \text{if } h \leq c \text{ then } h < j \leq n \\ \qquad \text{else } c \leq j \leq n;\ j \neq h \end{pmatrix}$$

$$b_{ij}^{h} = b_{ij}^{h-1} \quad \text{otherwise}$$

(4) Set

$$a_{ij}^{*} = b_{ij}^{n} \qquad (r \leq i \leq n;\ c \leq j \leq n)$$

4. Enumeration of all minimal cut sets

Given a directed graph G, an i-j cut set is a set of arcs, such that, by removing from G these arcs, there is no path between node i and node j. A minimal i-j cut set is an i-j cut set such that no subset of it is an i-j cut set.

In this section, we introduce a regular algebra for cut sets and we show that the problem of enumerating all minimal i-j cut sets can be reduced to the problem of finding the closure of a certain matrix. This problem can then be solved with the methods described in the previous section.

We assume that an arc is uniquely identified by a label attached to it. Let $L = \{l_1, \ldots, l_m\}$ be the set of such labels. Cut sets will be denoted by sets of labels: e.g. (l_1, l_2, l_3). Sets of cut sets will be denoted by sets of sets of labels: e.g. $\{(l_1, l_2, l_3),\ (l_4, l_5)\}$.

Now, we can define our algebra $C = (S, +, .)$. An element α of S is defined as a set of sets of labels $\in L$, such that if a set of labels s is an element of α, there is no other element of α which is a subset of s. For example, $\{(l_1, l_2), (l_2, l_3)\}$ is an element of S and $\{(l_1, l_2), (l_1, l_2, l_3)\}$ is not. The last requirement has been introduced because we want to consider only minimal cut sets.

If α and β are two elements of S, addition and multiplication are defined as follows:

$\alpha + \beta$ is a set obtained by making the union of each element of α with each element of β, and then deleting all elements which are a superset of some other element;

$\alpha\beta$ is a set obtained by making the union of α and β, and then deleting all elements which are a superset of some other element.

For example, if $\alpha = \{(l_1), (l_2, l_3)\}$ and $\beta = \{(l_1, l_2), (l_2, l_3)\}$, we have $\alpha + \beta = \{(l_1, l_2), (l_2, l_3)\}$ and $\alpha\beta = \{(l_1), (l_2, l_3)\}$.

The zero element $\emptyset$ is the set whose only element is the empty set, and the unit element e is the empty set.

It is possible to show that this algebra C is an algebra R_Q, i.e. that it satisfies the axioms A1-A9, A12. In order to prove this, we introduce another algebra

$C' = (S', \circledast, \circleddash)$. The elements of S' are the sets of sets of labels $\in L$ $(S' \supset S)$ and two operations are defined as

$\alpha \circledast \beta$ is a set obtained by making the union of each element of α with each element of β

$\alpha \circleddash \beta$ is the union of α and β .

Now, we can prove the following lemma.

Lemma 1 . Let $\eta \in S$ be the result of the computation of an expression consisting of additions and multiplications of elements of S in the algebra C. Let $\eta' \in S'$ be the result of the computation of the same expression in the algebra C'. Then

$$\eta' = \eta \cup \eta_1$$

where every element of η_1 is a superset of some element of η .

Proof. Given $\alpha, \beta \in S'$, we have

$$\alpha = \alpha_1 \cup \alpha_2 \quad \text{and}$$

$$\beta = \beta_1 \cup \beta_2$$

where $\alpha_1, \beta_1 \in S$ and every element of α_2(or β_2) is a

superset of some element of α_1 (or β_1).
We want to show that

$$(4.1) \qquad \gamma = \alpha \oplus \beta = (\alpha_1 + \beta_1) \cup \gamma_2$$

$$(4.2) \qquad \delta = \alpha \odot \beta = (\alpha_1 \beta_1) \cup \delta_2$$

where every element of γ_2 (or δ_2) is a superset of some element of $\alpha_1 + \beta_1$ (or $\alpha_1 \beta_1$).

If we prove (4.1) and (4.2), we can extend the same result to any expression containing additions and multiplications and the lemma is proved.

We have

$$\gamma = \alpha \oplus \beta = \alpha_1 \oplus \beta_1 \cup \alpha_1 \oplus \beta_2 \cup \alpha_2 \oplus \beta_1 \cup \alpha_2 \oplus \beta_2$$

and, according to the definition of addition in C,

$$\alpha_1 \oplus \beta_1 = (\alpha_1 + \beta_1) \cup \gamma'$$

where every element of γ' is a superset of some element of $\alpha_1 + \beta_1$.
Let us consider the set $\alpha_1 \oplus \beta_2$. According to the definition of β_2, every element of $\alpha_1 \oplus \beta_2$ is a superset of some element of $\alpha_1 \oplus \beta_1$ and, therefore, of $\alpha_1 + \beta_1$.

Analogously, for $\alpha_2 \oplus \beta_1$ and $\alpha_2 \oplus \beta_2$. Thus, by taking

$$\gamma_2 = \gamma' \cup \alpha_1 \oplus \beta_2 \cup \alpha_2 \oplus \beta_1 \cup \alpha_2 \oplus \beta_2$$

we have proved (4.1).

Now we have

$$\delta = \alpha \oplus \beta = \alpha_1 \cup \alpha_2 \cup \beta_1 \cup \beta_2$$

and

$$\alpha_1 \cup \beta_1 \doteqdot \alpha_1 \beta_1 + \delta'$$

If we take

$$\delta_2 = \delta' \cup \alpha_2 \cup \beta_2$$

every element of δ_2 is a superset of some element of $\alpha_1 \beta_1$ and (4.2) is proved.

Q.E.D.

As a consequence of this lemma, we see that any expression in our algebra C can be computed by first computing it in the algebra C' and then deleting from the resulting set all elements which are a superset of some other element. In this way, it is easy to prove the validity of axioms A1-A9, A12.

Let G be a directed graph, with n nodes, for which we want to find all minimal i-j cut sets for every pair of nodes i and j. Let a label l_{ij} be attached to every arc (i j) of G. An n x n matrix A, whose elements belong to the above defined algebra C, is constructed by setting $a_{ij} = \emptyset$, if there is no arc between node i and node j, and $a_{ij} = \{(l_{ij})\}$ otherwise. We can prove the following theorem.

<u>Theorem 1</u>. The element a^*_{ij} of the closure A^* of matrix A gives the set of all minimal i-j cut sets for graph G.

<u>Proof</u>. From (2.1) we recall that

$$A^* = E + A + A^2 + \dots + A^{n-1}$$

Thus, the element a^*_{ij} consists of a sum of path products, corresponding to all simple paths between node i and node j:

$$a^*_{ij} = \sum_{k=1}^{n-1} \sum_{p \in P^k_{ij}} w(p)$$

where P^k_{ij} is the set of all simple paths between i and j of length k, and w(p) is the path product of path p.

If we have $p = (i\ i_1)\ (i_1\ i_2)\ \ldots\ (i_{k-1}\ j)$, the path product of p is

$$w(p) = \{(l_{i,i_1})\}\ \{(l_{i_1,i_2})\}\ \ldots\ \{(l_{i_{k-1},j})\} =$$

$$= \{(l_{i,i_1}),\ (l_{i_1,i_2}),\ \ldots,\ (l_{i_{k-1},j})\}$$

According to Lemma 1, we can add all path products in algebra C'. Every element of the resulting set η' will contain an arc label for every path from i to j, therefore it is certainly a cut set. Moreover, these sets are obtained by taking all possible combinations of arcs from simple paths; hence, the set η' will contain all minimal cut sets. Finally, the value of a^*_{ij} is obtained by deleting from η' all elements which are a superset of some other element, hence, a^*_{ij} will contain only minimal cut sets.

Q.E.D.

The closure of matrix A can be computed using the methods described in section 3. Let us consider, for example, the graph in Fig. 1. If we want to obtain a^*_{34}, we can use algorithm G' with $r = 3$ and $c = 3$.

$$B^{0} = \begin{vmatrix} e & \{(c)\} & \emptyset & \{(d)\} \\ \emptyset & e & \emptyset & \{(f)\} \\ \{(a)\} & \{(b)\} & e & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \end{vmatrix}$$

$$B^{1} = \begin{vmatrix} \text{---} & \text{---} & \text{---} & \text{---} \\ \text{---} & e & \emptyset & \{(f)\} \\ \text{---} & \{(a,b),(b,c)\} & e & \{(a),(d)\} \\ \text{---} & \emptyset & \emptyset & \emptyset \end{vmatrix}$$

$$B^{4}=B^{3}=B^{2} = \begin{vmatrix} \text{---} & \text{---} & \text{---} & \text{---} \\ \text{---} & \text{---} & \text{---} & \text{---} \\ \text{---} & \text{---} & e & \{(a,b),(a,f),(b,c,d),(d,f)\} \\ \text{---} & \text{---} & \emptyset & \emptyset \end{vmatrix}$$

We have indicated with dashes those elements of a matrix B^{i}, which are not used in the computation of subsequent matrices.

The results of this section can easily be extended to the case of non-directed graphs, by transforming them into directed graphs. To each non-directed arc i-j we substitute two directed arcs (i j) and (j i), and we attach the same label l_{ij} to both arcs.

5. Experimental results

A computer program implementing algorithm G' has been written in LISP 1.5. Every cut set is represented by a word specifying with a bit pattern which arcs are present in the cut set. Implementation of addition requires some care, since a simple implementation according to its definition might be too expensive. In fact, if we add two sets of cardinality n and m, we first have to perform n x m unions and then we must delete all non-minimal cut sets. If the result is a set of cardinality r, the time necessary to perform addition is roughly proportional to n x m x r. However, the computing time can be reduced to be proportional to n x m, by taking into account some properties of cut sets.

For example, the program has been applied to the graph of Fig. 2, and the cut sets between the pairs of nodes $N_5 - N_8$, $N_6 - N_8$, $N_7 - N_8$ are given in Table I.

The total computing time of the algorithm is greatly dependent on the structure of the graph and, in general, it is exponential, since the number of minimal cut sets can be exponential. Furthermore, for a given graph, the computing time depends on the order of elimination of nodes, as in the case of the solution of a sparse system of equations in linear algebra [9].

Among other methods proposed to solve this problem, the one described in [7] first computes all simple paths and then finds all combinations of arcs cutting all paths. Since the time necessary to compute all paths is comparable with the time needed to compute all cut sets, this method can be considerably less efficient than ours. Another method [8], for non-directed graphs, searches the given graph starting from node i and constructs a tree whose terminal nodes are the minimal i-j cut sets.

The main difference between these methods and the one proposed in this paper is that they compute the minimal cut sets only for one pair of nodes and, if we consider several pairs of nodes, they have to be used separately for each pair. Instead, with our method, we can compute simultaneously several entries of the matrix. For example, the computation of $a^*_{n-2,n}$ and $a^*_{n-1,n}$ with algorithm G' requires only one extra addition and multiplication with respect to the computation of $a^*_{n-1,n}$.

REFERENCES

[1] - Salomaa, A., Theory of Automata, Pergamon Press, 1969.

[2] - Fratta, L. and Montanari, U., All Simple Paths in a Graph by Solving a System of Linear Equations, Internal Report B71-11, I.E.I., Pisa, October 1971.

[3] - Backhouse, R.C. and Carré, B.A., Regular Algebra Applied to Path-Finding Problems, to appear in J. Inst. Maths. Applics.

[4] - Floyd, R.W., Algorithm 97 Shortest Path, CACM 5, June 1962, p. 345.

[5] - Warshall, S., A Theorem on Boolean Matrices, JACM 9, 1962, pp. 11-12.

[6] - Fratta, L. and Montanari, U., A Boolean Algebra Method for Computing the Terminal Reliability in a Communication Network, IEEE Trans. on Circuit Theory, May 1973.

[7] - Nelson, A.C., Batts, J.R. and Beadles, R.L., A Computer Program for Approximating System Reliability, IEEE Trans. on Reliability, R-19, 1970, pp. 61-65.

[8] - Jensen, P.A. and Bellmore, M., An Algorithm to Determine the Reliability of a Complex System, IEEE Trans. on Reliability, R-18, 1969, pp. 169-174.

[9] - Martelli, A. and Montanari, U., An Application of Heuristically Guided Search to Elimination Processes, International Computing Symposium, Venice, 12-14 April 1972, pp. 59-72.

TABLE I

(5-8 CUTSETS)
(12 9 6 1)
(12 9 5 4)
(12 7 6 5 1)
(10 6 1)
(3 2 1)
(11 2 1)
(10 8 7 5)
(8 7 6 5 3 2)
(11 8 7 6 5 2)
(10 9 8)
(9 8 6 3 2)
(11 9 8 6 2)
(12 8)
(10 5 4)
(6 5 4 3 2)
(11 6 5 4 2)
(12 7 4)
(10 9 7 4)
(9 7 6 4 3 2)
(11 9 7 6 4 2)

(6-8 CUTSETS)
(10 6 1)
(11 10 6 2)
(10 6 3 2)
(10 5 4)
(12 7 6 5 1)
(12 11 7 6 5 2)
(12 7 6 5 3 2)
(12 7 4)
(12 8)
(12 9 6 1)
(12 11 9 6 2)
(12 9 6 3 2)
(12 9 5 4)
(10 8 7 5)
(10 9 8)
(10 9 7 4)

(7-8 CUTSETS)
(11 9 7 6 4 2)
(10 9 7 4)
(12 7 4)
(11 6 5 4 2)
(10 5 4)
(12 8)
(11 9 8 6 2)
(10 9 8)
(11 8 7 6 5 2)
(10 8 7 5)
(11 2 1)
(10 6 1)
(11 3)
(10 6 3 2)
(12 9 6 1)
(12 9 6 3 2)
(12 9 5 4)
(12 7 6 5 3 2)
(12 7 6 5 1)

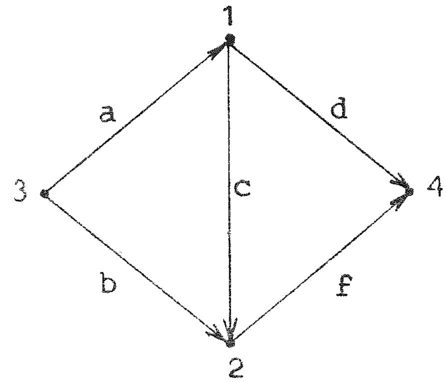

Fig. 1

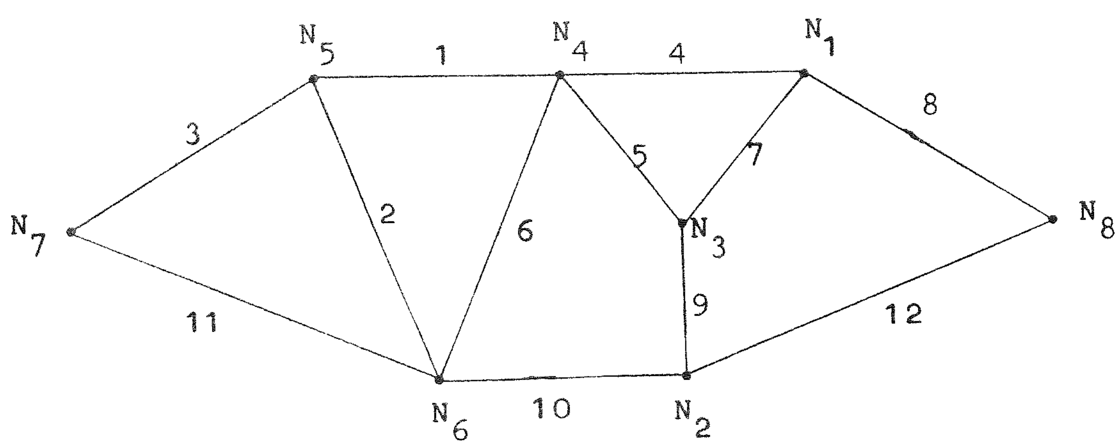

Fig. 2