P. J. DOULLIEZ
M. R. RAO

Non-planar network with edges subject to failure and enumeration of proper cuts

Revue française d'automatique, informatique, recherche opérationnelle. Recherche opérationnelle, tome 7, nº V1 (1973), p. 85-96

<http://www.numdam.org/item?id=RO_1973__7_1_85_0>

R.A.I.R.O.
(7e année, V-1, 1973, p. 85-96)

NON-PLANAR NETWORK WITH EDGES SUBJECT TO FAILURE AND ENUMERATION OF PROPER CUTS

by P. J. Doulliez [1] et M. R. Rao [2]

Summary. — *An undirected non planar network consisting of several demand vertices each with an associated demand is considered. The requirement at each demand vertex is an increasing function of time. It is required to find the maximum time up to which all demands can be satisfied when k (> 1) edges are subject to failure. The number k is pre-specified but not the edges themselves. For source — sink planar networks an efficient algorithm is available to solve this problem. However, no such algorithm is available for non-planar networks. A method is given here for small non-planar networks. This method is based on enumerating all proper cuts of a network. A procedure is suggested for enumerating all proper cuts under the hypothesis that each vertex is on at least one elementary chain from source to sink. A method for locating vertices, if any, which are not on any elementary chain from source to sink is also given.*

1. INTRODUCTION

An undirected non-planar network [3] G with several demand vertices is considered. Each demand vertex is connected to one or more supply vertices through intermediate vertices. The requirement at each demand vertex is an increasing function of time. Some or all edges of the network have two capacities — the normal and the reduced capacity. An edge is said to fail if its capacity is equal to the lower value which is strictly less than the normal capacity. It is required to find the maximum time up to which all demands can be satisfied when k (> 1) edges fail; only the number of arcs is pre-specified and not the arcs themselves.

All the supply vertices can be joined to a common source vertex by adding edges of appropriate capacity. Similarly the demand vertices can be connected to a common sink vertex by introducing demand edges whose capacities are increasing functions of time. Let the network obtained after the addition of

(1) Société de Traction et d'Electricité, Brussels.
(2) University of Rochester.
(3) Section 2 contains a definition of a network and other related ideas.

these edges be G^*. The problem under consideration is equivalent to finding the maximum time such that the capacity of a cut (in G^*) consisting of only the demand edges is equal to the capacity of a minimal cut separating the common source and the common sink vertex when k edges fail.

An efficient algorithm is given in [1] for solving the problem when the network (G^*) is source-sink planar. A network is planar if it can be represented on a plane or a sphere with no two edges intersecting except at a vertex and it is source-sink planar if it remains planar after an edge connecting the source and sink has been added [2, 3]. For a source-sink planar network (primal) there exists a dual network such that there is a one-to-one correspondence between the proper cuts of the primal network and the elementary chains between two specified vertices in the dual network [4, 7]. The problem then reduces to one of finding a shortest chain between two specified vertices of the dual network when k of its edges may have reduced distances.

For non-planar networks no good algorithm is available (1). One approach would be to consider all possible combinations of k edges that may fail. Then associated with each set of k edges that fail, the problem of finding the maximum time up to which all demands can be satisfied may be solved by any one of the methods given in [5]. The minimum of these values represents the solution. The main difficulty with such an approach is that the number of combinations of k edges that may fail would be prohibitively large.

In this paper (section 6) a method is given for undirected non-planar networks in which the number of vertices is small. This method is based on enumerating all proper cuts of a network. Needless to say the number of proper cuts in a network increases very rapidly with the number of vertices. Hence the method is useful only if the number of vertices in the network is small. The main advantage of this method over the previous one is that its applicability is limited by the number of vertices in the network and not by the number of edges that are potential candidates for failure. Furthermore when the number of vertices is small enumeration of all elementary cuts is considerably easier than solving a separate problem associated with each set of k edges that may fail. Consequently this method is usually preferable if the number of vertices in the network is less than the number of edges that are candidates for failure.

A procedure is given in section 5 for enumerating all proper cuts when each vertex of the network is on at least one elementary chain from source to sink. This procedure is based on two theorems which are stated and proved in section 3. A method for locating vertices, if any, which are not on any elementary chain from source to sink is given in section 4.

Throughout this paper we consider only undirected networks. Some of the definitions given in the next section are from [6].

(1) An efficient algorithm is given in [5] for non-planar networks with $k = 1$.

2. DEFINITIONS

An undirected network $G = (N, A)$ consists of a collection N of elements x, y, ..., together with a subset of unordered pairs $[x, y]$ of elements taken from N. The elements of N are the vertices and the members of A are the edges of the network.

A chain is a sequence $x_1, x_2, \ldots, x_n$ of distinct vertices such that $[x_i, x_{i+1}]$ $i = 1, 2, \ldots, n - 1$, is an edge of the network.

A cycle is a chain with $x_n = x_1$.

An elementary chain is a chain with no cycles.

Given X_1 which is a subset of N, a subnetwork $G_1 = (X_1, A_1)$ is defined such that edge $[x, y] \in A_1$ if $[x, y] \in A$ and $x \in X_1, y \in Y_1$.

A network (or a subnetwork) $G = (N, A)$ is connected if for any two vertices $i, j \in N, i \neq j$, there is a chain between i and j.

If X_1 and X_2 are subsets of N, let $[X_1, X_2]$ represent the set of all edges $[x_1, x_2]$ such that $x_1 \in X_1$ and $x_2 \in X_2$.

Vertex s represents the source vertex and vertex d the sink vertex.

A cut in G separating a vertex s and a vertex d is a set of edges $[X_1, X_2]$ where $s \in X_1, d \in X_2, X_1 \cap X_2 = \emptyset$ and $X_1 \cup X_2 = N$.

A proper cut is one that does not contain another cut as one of its proper subsets.

The capacity of a cut is the sum of the capacities of the edges that belong to that cut.

A cut separating s and d is minimal if its capacity is not larger than the capacity of any other cut separating s and d.

3. BASIC THEOREMS

The procedure for enumerating all proper cuts is based on the following two theorems. The first one gives a necessary and sufficient condition for a proper cut. The second theorem shows that a proper cut $[X_1, X_2]$ exists for each possible number of vertices in X_1 (or in X_2) when each vertex of the network is on at least one elementary chain from source to sink.

Theorem 1

A cut $[X_1, X_2]$ of a connected network is a proper cut if and only if the subnetworks $G_1 = (X_1, A_1)$ and $G_2 = (X_2, A_2)$ are connected.

Proof. Let vertex s be in X_1 and vertex d in X_2.

(*i*) If the cut $[X_1, X_2]$ is a proper cut, then $G_1 = (X_1, A_1)$ and $G_2 = (X_2, A_2)$ are connected :

Suppose that one of the subnetworks (G_1) is disconnected. Then, $X_1 = X_1' \cup X_1''$ where $X_1' \cap X_1'' = \varnothing$ and $[X_1', X_1''] = \varnothing$. Let s be in X_1'. The cut $[X_1, X_2]$ contains a cut $[X_1', X_1'' \cup X_2]$ and hence cannot be a proper cut.

(*ii*) If the two subnetworks defined by a cut $[X_1, X_2]$ are connected, then $[X_1, X_2]$ is a proper cut (1).

Suppose $[X_1, X_2]$ is not a proper cut. Then there exists a cut

$$[X_3, X_4] \subset [X_1, X_2].$$

Consider an edge $[x, y] \in [X_1, X_2]$ and $\notin [X_3, X_4]$. Since the subnetworks defined by X_1 and X_2 are connected there exists a chain $s, i, ..., x, y, j, ... d$ where the vertices s, i, ..., x belong to X_1 and the vertices y, j, ... d belong to X_2. But the cut $[X_3, X_4]$ does not block this chain form s to d since by assumption $[x, y] \notin [X_3, X_4]$ and all other edges of this chain are not in $[X_1, X_2]$ and hence not in $[X_3, X_4]$. Since a cut separating s and d has the property that it blocks all chains from s to d, $[X_3, X_4]$ cannot be a cut. This leads to a contradiction. Hence $[X_1, X_2]$ is a proper cut.

Theorem 2

If each vertex of the network is on at least one elementary chain from the source $s \in X_1$ to the sink $d \in X_2$, then there exists for each value of $r (r = 1, ..., n - 1)$ a proper cut $[X_1, X_2]$ with r elements in X_1.

Proof (by induction). For $r = 1$, the source s is the only vertex in X_1. Consequently the subnetwork $G_1 = (X_1, A_1 = \varnothing)$ is connected. Each vertex in X_2 is connected to the sink d for otherwise each vertex of the network would not be on at least one elementary chain from the source to the sink. Thus the subnetwork $G_2 = (X_2, A_2)$ is also connected and hence the theorem holds for $r = 1$.

Now suppose that the theorem is true for $r = k$ $(k < n - 1)$, we will prove that it is true for $r = k + 1$.

A proper cut $[X_1, X_2]$ with k elements in X_1 and $n - k$ elements in X_2 defines two connected subnetworks $G_1 = (X_1, A_1)$ and $G_2 = (X_2, A_2)$. There must be an edge (belonging to the set A) from a vertex in X_1 to a vertex $j \neq d$ in X_2 since each vertex of the network is on at least one elementary chain from the source to the sink (fig. 1 *a*). Form the two mutually exclusive and exhaustive sets $X_1' = X_1 \cup j$ and $X_2' = X_2 - j$ with $k + 1$ and $n - k - 1$ elements respectively. The subnetwork $G_1' = (X_1', A_1')$ is connected since

(1) Our original proof of this part of the theorem is somewhat longer. We are indebted to B. Gordon for this simple proof.

$G_1 = (X_1, A_1)$ is connected and vertex j has an edge to a vertex in X_1. If $G'_2 = (X'_2, A'_2)$ is also connected, the theorem follows. Now suppose G'_2 is not connected. Let $X'_2 = X_2 - j = X_{21} \cup X_{22} \cup \ldots \cup X_{2t}$ where the sets X_{2i}, $i = 1, 2, \ldots, t$ are mutually exclusive and each set X_{2i} defines a connected subnetwork $G_{2i} = (X_{2i}, A_{2i})$ with the property that there exists no edge (belonging to the set A) from a vertex in X_{2i} to a vertex in X_{2j}, $i \neq j$.

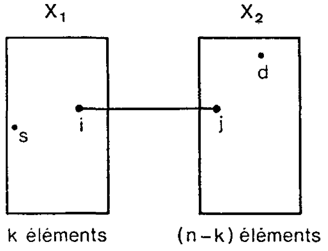

Figure 1 *a*

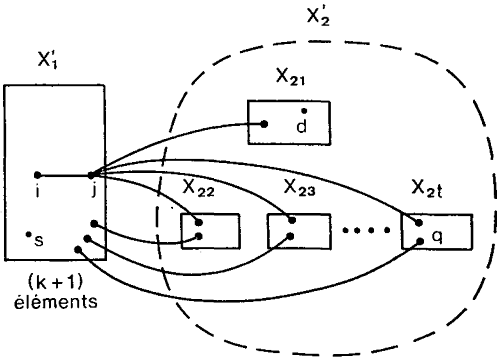

Figure 1 *b*

Let the sink $d \in X_{21}$. Since $G_2 = (X_2, A_2)$ was connected, there exists at least one edge from vertex j to vertices in each of the sets X_{2i}, $i = 1, 2, \ldots, t$. In addition, since each vertex of the network is on at least one elementary chain from s to d, there exists at least one edge from a vertex in X_1 to vertices in X_{2i}, $i = 2, 3, \ldots, t$. Let q be a vertex in X_{2t} that has an edge to a vertex in X_1 (fig. 1 *b*). Now, if X'_1 is taken to be $X_1 \cup q$ and $X'_2 = X_2 - q$, the subnetwork $G'_1 = (X'_1, A'_1)$ is still connected. Again, if $G'_2 = (X'_2, A'_2)$ is connected the theorem follows. But suppose G'_2 is not connected. Then, we have

$$X'_2 = j \cup X_{21} \cup \ldots \cup X_{2,t-1} \cup X^1_{2t} \cup X^2_{2t} \cup \ldots \cup X^v_{2t}$$

where $X_{2t} - q = X^1_{2t} \cup X^2_{2t} \cup \ldots \cup X^v_{2t}$ and the sets X^i_{2t}, $i = 1, 2, \ldots, v$ are mutually exclusive. In addition each set X^i_{2t} defines a connected subnetwork $G^i_{2t} = (X^i_{2t}, A^i_{2t})$ with the property that there exists no edge (belonging to the set A) from a vertex in X^i_{2t} to a vertex in X^j_{2t}, $i \neq j$. We may assume (without any loss in generality) that a vertex in each of the sets X^i_{2t}, $i = 1, 2 \ldots, p < v$ has an edge to vertex j.

Let $X''_2 = j \cup X_{21} \cup \ldots \cup X_{2,t-1} \cup X^1_{2t} \cup \ldots \cup X^p_{2t}$. The subnetwork $G''_2 = (X''_2, A''_2)$ is connected since vertex j has an edge to each X_{2i}, $i = 1, 2, \ldots, t-1$ and to X^w_{2t}, $w = 1, 2, \ldots, p$. By a similar argument as before there exists at least one edge from vertex q to vertices in each of the sets X^b_{2t}, $b = p + 1, \ldots, v$. Also there exists at least one edge from a vertex in X_1 to vertices in

$$X^b_{2t}, b = p + 1, \ldots, v.$$

Furthermore there must be at least one edge from vertex q to vertices in X_2''. We note that the subset X_{2t}^v has fewer elements than X_{2t} which in turn has less than $n - k - 1$ elements. Now, choose an element u from X_{2t}^v that has an edge to a vertex in X_1. By following the same arguments as before, we should either form a subnetwork G_2' that is connected or be able to form subsets of X_{2t}^v that have fewer elements than X_{2t}^v. Since the successive subsets chosen have fewer elements, we will reach a stage when G_2' is connected. Thus the theorem is proved and we can also conclude that a proper cut $[X_1, X_2]$ with r vertices in X_1 ($r = 2, 3, ..., n - 1$) can always be constructed from a proper cut with $r - 1$ vertices in X_1.

Corollary

If there is a proper cut $[X_1, X_2]$ with r vertices in X_1 ($r < n - 1$), then there is a proper cut $[X_3, X_4]$ with $r + 1$ vertices in X_3 such that $X_1 \subset X_3$ and, $X_4 \subset X_2$.

The proof of this corollary is very similar to that given for theorem 2 where a cut with $r + 1$ elements was constructed from a cut with r elements.

The converse of this corollary is also true. That is, if there is a proper cut $[X_3, X_4]$ with $r + 1$ elements in X_3, then there is a proper cut $[X_1, X_2]$ with r elements in X_1 such that $X_1 \subset X_3$ and $X_4 \subset X_2$.

4. FINDING VERTICES NOT ON ANY ELEMENTARY CHAIN FROM SOURCE TO SINK

The procedure given in the next section for enumerating all proper cuts of a network is valid only if each vertex of the network is on at least one elementary chain from source to sink. In this section, an algorithm is presented for finding the vertices, if any, that are not on any elementary chain from source to sink. If we drop such vertices and the associated edges, the value for a minimum cut separating the source and the sink is not altered. Thus we can obtain a network that satisfies the condition for which the procedure in section 5 is valid.

In the algorithm below, the vertices which are on at least one elementary chain from s to d are identified by a labeling procedure.

Three labels are associated with each vertex i :

the label $h(i)$ indicates whether an elementary chain has already been found through vertex $i (h(i) = i)$ or not $(h(i) = 0)$;

the label $p(i) = j$ refers to the vertex j from which vertex i has been labeled;

the label $q(i)$ refers to a vertex u which is a predecessor of i (that is u can be reached by making use of the labels $j = p(i)$, $k = p(j)$ ... $u = p(t)$. Vertex u is known to be on at least one elementary chain from s to d ($h(u) = u$).

The algorithm

Let m be the number of edges and n the number of vertices. Let the vertices of the network be numbered 1 to n with vertex s numbered as one and vertex d as n. Let the edges of the network be numbered 1 to m.

Step 1. Find an elementary chain from s to d. This is easily done by any labeling procedure. The vertices of that chain are of course on an elementary chain from s to d. For each vertex i of that chain, let $h(i) = p(i) = q(i) = i$. For other vertices, let $h(i) = p(i) = q(i) = 0$.

Step 2. Let $r = 1$.

Step 3. If $r > m$, go to step 5. Otherwise, let edge r be between vertices i and j.

Step 4. One of the following cases occurs :

1° $h(i) \neq 0$; $h(j) \neq 0$.

An elementary chain has already been found through both vertices i and j. Let $r = r + 1$ and go to step 3;

2° $h(i) = 0$; $h(j) = 0$; $p(i) \neq 0$; $p(j) = 0$.

Vertex j has not yet been labeled. Since vertex i is an immediate predecessor of j, let $p(j) = i$. A vertex u with $h(u) = u$ is a predecessor of i and $q(i) = u$. Since u is also a predecessor of j, let $q(j) = q(i)$. Let $r = r + 1$ and go to step 3;

3° $h(i) = 0$; $h(j) = 0$; $p(i) \neq 0$; $p(j) \neq 0$.

a) $q(i) \neq q(j)$ (breakthrough case).

The predecessors u of i and v of j are different. We have $q(i) = u$ and $q(j) = v$. Since u and v are known to be on at least one elementary chain (not necessarily the same elementary chain) from s to d, vertices i and j and also their predecessors are on an elementary chain from s to d. The predecessors of i can be traced by making use of the labels $d = p(i)$, $c = p(d)$ ... $u = p(e)$. Let $h(i) = p(i) = q(i) = i$, $h(d) = p(d) = q(d) = d$, ... $h(e) = p(e) = q(e) = e$.

Similarly, if $f, g, ... w, v$ are the predecessors of j, let $h(j) = p(j) = q(j) = j$, $h(f) = p(f) = q(f) = f$, $h(g) = p(g) = q(g) = g$, ... $h(w) = p(w) = q(w)$. If $h_i \neq 0$, $i = 1, 2, ..., n$ then terminate since each vertex of the network is on at least one elementary chain from s to d. Otherwise for any vertex i of the network if $h(i) = 0$, let $p(i) = q(i) = 0$. Go to step 3.

b) $q(i) = q(j)$.

Both i and j have the same predecessor u. Hence no conclusions can be drawn. Let $r = r + 1$ and go to step 3.

4° $h(i) = 0$; $h(j) = 0$; $p(i) = 0$; $p(j) = 0$.

Vertices i and j are both unlabeled. Nothing can be said about the existence of an elementary chain through i or j. Let $r = r + 1$ and go to step 3.

5° $h(i) = 0$; $h(j) = 0$; $p(i) = 0$; $p(j) \neq 0$.

This case is equivalent to case 2°, except that j plays the role of i and vice-versa.

6° $h(i) \neq 0$; $h(j) = 0$; $p(j) = 0$.
If $h(i) \neq 0$, we necessarily have $h(i) = p(i) = q(i) = i$. This case is equivalent to case 2° with $q(i) = i$. Vertex i has no predecessors.

7° $h(i) = 0$; $h(j) \neq 0$; $p(i) = 0$.

This case is equivalent to case 6° except that j plays the role of i and vice-versa.

8° $h(i) \neq 0$; $h(j) = 0$; $p(j) \neq 0$.

Since $h(i) \neq 0$, we have $h(i) = p(i) = q(i) = i$. This case is equivalent to case 3° with $q(i) = i$. Vertex i has no predecessors.

9° $h(i) = 0$; $h(j) \neq 0$; $p(i) \neq 0$;.

This case is equivalent to case 3° with $q(j) = j$. Vertex j has no predecessors.

Step 5. If during the last scanning of the list of edges there was a break-through or at least one new label $p(i)$ was assigned to a vertex i, then go to step 2. Otherwise, go to step 6.

Step 6. All vertices i such that $h(i) = 0$ are not on any elementary chain from s to d. END.

A justification of the algorithm together with a proof of its finiteness could be easily established.

5. PROCEDURE FOR ENUMERATING ALL THE PROPER CUTS

The procedure enumerates all proper cuts with $r + 1$ vertices in X_1 utilizing the previously found proper cuts $[X_1, X_2]$ with r vertices in X_1. The fact that all proper cuts are enumerated follows from theorem 2 and the converse of its corollary.

The following notation is used in the algorithm.

Y_{ri} refers to a subset of vertices with r elements. The subscript i denotes the i th such subset. $[Y_{ri}, Z_{ri}]$ with $Z_{ri} = N — Y_{ri}$ is a proper cut.

$p(r)$ is the number of proper cuts such that X_1 has r elements. Thus the index i in Y_{ri} goes from 1 to $p(r)$;

the vertices are numbered from 1 to n with vertex s as 1 and vertex d as n.

Step 1. Let $Y_{11} = 1$; $Z_{11} = \{2, 3, ..., n\}$; $p(1) = 1$; $p(r) = 0$, $r = 2, 3, ... n - 1$. Let $r = 1$.

Step 2. Let $i = 1; j = 2; q = 1$.

Step 3. If $j \in Z_{ri}$ and j has an edge to a vertex in Y, go to step 4. Otherwise, go to step 8.

Step 4. If the subnetwork defined by $(Z_{ri} - j)$ is connected (1), go to step 5. Otherwise, go to step 8.

Step 5. If $q = 1$, go to step 7. Otherwise, go to step 6.

Step 6. If $(Y_{ri} \cup j) \neq Y_{r+1,t}$ for $t = 1, 2, ..., q - 1$, go to step 7. Otherwise, go to step 8.

Step 7. Let $Y_{r+1,q} = (Y_{ri} \cup j)$; $Z_{r+1,q} = (Z_{ri} - j)$; $p(r + 1) = p(r + 1) + 1$ $q = q + 1$.

Step 8. Let $j = j + 1$. If $j = n$, go to step 9. Otherwise, go to step 3.

Step 9. Let $i = i + 1$. If $i > p(r)$, go to step 10. Otherwise, let $j = 2$ and go to step 3.

Step 10. Let $r = r + 1$. If $r < n - 1$, go to step 2. If $r = n - 1$, then the lastproper cut is obtained, i.e., $Y_{n-1,1} = \{1, 2, ..., n - 1\}$ and $Z_{n-1.1} = \{n\}$. END.

EXAMPLE

It is required to find all the proper cuts of the following network (fig. 2).

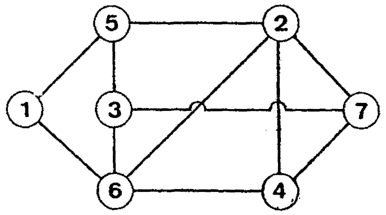

Figure 2

The vertices are numbered as shown. $N = \{1, 2, ..., 7\}$. If vertex 1 (vertex 7) is the source (the sink), then it must be in every subset $X_1(X_2)$. Subsets X_1 with 1, 2, ... $n - 1$ vertices are successively generated as in figure 3. The corresponding subset X_2 is given by $N - X_1$ and the proper cut by $[X_1, X_2]$.

It should be noted that the subsets $X_1 = (1, 6, 2, 3)$; $X_1 = (1, 5, 2, 3, 4)$; $X_1 = (1, 6, 2, 4, 3)$ are not retained because the corresponding subsets X_2 define sub-networks which are not connected. In this example there are 20 proper cuts whereas the total number of cuts is 32.

(1) This can be verified for instance by considering a vertex $u \in (Z_{ri} - j)$ such that edge $(u, j) \in A$. The subnetwork defined by $(Z_{ri}\ j)$ is connected if and only if vertex u has a chain to every vertex$v \in (Z_{ri} - j)$ such that edge $[v, j] \in A$.

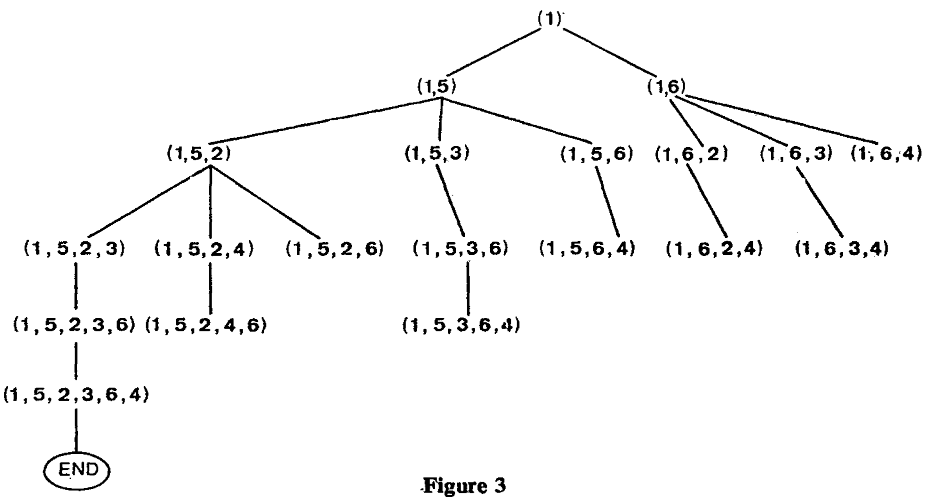

Figure 3

If a network with n vertices is complete, i.e., there exists an edge between every pair of vertices, all the 2^{n-2} cuts are proper cuts and they can be enumerated directly. In many applications, the network is not complete and there may be a substantial difference between the number of cuts and the number of proper cuts. However, the number of proper cuts increases rapidly with the number of vertices and hence, the procedure given is applicable only if the network is not large.

6. CAPACITY OF A NETWORK WITH EDGES SUBJECT TO FAILURE

An undirected network (G) is given with several demand vertices connected to one or more supply vertices through intermediate vertices. The demand at each sink vertex is an increasing function of time. It is required to find the maximum time time up to which all demands can be satisfied even if k edges of the network fail. The given network G is augmented by adding a source (s) and a sink (d) vertex. All the supply vertices are joined to the source vertex by introducing edges of appropriate capacity. Similarly, all demand vertices are joined to the sink vertex by the addition of demand edges. Let the demand vertices be denoted as d_1, d_2, ... d_p. The capacity of the edge between demand vertex d_i and the sink d is a given function $g_i(t)$ where t is the time parameter. $g_i(t)$ represents the demand at demand vertex d_i and it is an increasing function of time. Let the augmented network be denoted as $G^* = (X^*, A^*)$. The above problem is now equivalent to finding the maximum value of t such that $[X^* - d, d]$ represents a minimum cut separating s and d in G^* when k edges fail.

If $[X_1, X_2]$ is a proper cut separating s and d in G^* let $C'[X_1, X_2]$ be its capacity with k largest capacity reductions (1) taken into account. Corresponding to each proper cut $[X_1, X_2]$ we have an upper bound for t given by the relation

$$C'[X_1, X_2] = \sum_{i=1}^{p} g_i(t)$$

where p is the number of demand vertices.

The least of these upper bounds gives the maximum time (t^*) up to which all demands can be satisfied even when k edges fail. The proper cut $[\bar{X}_1, \bar{X}_2]$ corresponding to this least upper bound gives a set of edges with the property that the capacity of at least one member of this set should be increased if the value of t^* is to be increased. Thus if all proper cuts are known it is easy to find the value of t^* and the corresponding proper cut $[\bar{X}_1, \bar{X}_2]$ separating s and d in G^*.

7. CONCLUSIONS

The problem of finding the capacity of a network with edges subject to failure arises as a sub-problem in finding an optimum sequence of investments to be made on the edges of a network over a given time horizon [8]. An investment on an edge increases the capacity of that edge by a specified amount. Since investment costs are time discounted, an investment on an edge should be delayed as much as possible.

Enumeration of all proper cuts is done only once and the capacity of each cut is calculated easily when some or all of the initial edge capacities are changed. However, since all proper cuts have to be stored in memory, only small non-planar networks can be handled.

ACKNOWLEDGEMENT

The authors are indebted to Professor G. de Ghellinck for many helpful discussions.

(1) If an edge never fails, it is considered to have zero capacity reduction. If the cut has only $k_1 < k$ edges then k_1 capacity reductions are taken into account.

REFERENCES

[1] DOULLIEZ P. and RAO M. R., « Capacity of a Network with Increasing Demands and Arcs Subjects to Failure » *Operations Research*, 19 (4), 905-915 (1971).

[2] L. R. FORD and D. R. FULKERSON, « Maximum Flow Through a Network », *Canadian Journal of Mathematics*, 8 (May 1956), 399-404.

[3] R. WOLLMER, « Removing Arcs from a Network », *Operations Research*, 12 (1964), 934-940.

[4] G. de GHELLINCK, «Aspects de la notion de dualité en théorie des graphes», Cahiers du centre de recherche opérationnelle, 3, 2 (1961).

[5] P. DOULLIEZ and M. R. RAO, « Maximal Flow in a Multiterminal Network with any One Arc Subject to Failure », Management Science, 18-1 (1971).

[6] L. R. FORD and D. R. FULKERSON, « Flows in Networks », Princeton University Press, Princeton, New Jersey, 1962.

[7] C. BERGE, *Théorie des graphes et applications*, Dunod, Paris, 1958, p. 209.

[8] P. DOULLIEZ, « Optimal Capacity Planning of Multi-terminal Networks », Thèse de Doctorat, Université Catholique de Louvain, Louvain, Belgium, 1970.