MINIMAL CUT SETS FOR TRANSPORTATION SYSTEM

Marija Mihova
Institute of Informatics,
Faculty of Natural Sciencies
and Mathematics
Skopje, Macedonia

Nataša Maksimova
Faculty of informatics,
University "Goce Delčev"
Štip, Macedonia

ABSTRACT

One of the most significant problems in the analysis of the reliability of multi-state transportation systems is to find the minimal path and minimal cut vectors. For that purpose there are proposed several algorithms that use the minimal path and cut sets of such systems. In this paper we give an approach to determine the minimal cut vectors. This approach directly finds all minimal cut sets. Also we will give an optimization of the basic idea, as a result of which, we will not get a candidates for minimal path sets, which actually are not a minimal cut sets.

I. INTRODUCTION

Traditionally, network reliability has been analyzed from a binary perspective, which assumes that the system and its components can be in either two states: completely functioning or failed. However, binary state theory does not fully describe some systems, as telecommunication systems, transportation systems, water distribution, and gas and oil production [1], since the elements of these systems usually operate in any of several intermediate states. So, its reliability is analyzed using multi-state reliability theory. One way to calculate their reliability is by using the minimal path or minimal cut vectors. In [1] - [4] are proposed algorithms for obtaining such vectors. They assumed that the minimal path or cut sets are known. Thus opens the question of obtaining the minimal path and cut sets for such system.

The problem of obtaining the minimal path sets of a network is easily solvable, but it is not the case for the minimal cut vectors. An algorithm for obtaining the minimal cut sets is given in [6], but this technique works only in the case of a directed network. A procedure for obtaining the minimal cut vectors for multi-state system from the corresponding minimal path vectors is given in [5], and it can be applied in the binary case. But this algorithm has a fairly high complexity. In this study we develop techniques for obtaining the minimal cut sets of an undirected network.

II. GENERAL

Let we have a two-terminal undirected network $G(V, E)$, where V is the set of nodes, and E is a set of links. Let s be the source node and t be the sink node. The cut set is defined as a set of links, such that, if there no flow through these links, then there no flow from the source to the sink. The cut set C is a minimal cut set if there is not another cut set C', such that $C' \subset C$. For an undirected network the following proposition is clear.

Proposition 1 Let $G(V, E)$ be an undirected connected network with source node s and sink node t. Then C is a cut set if and only if by removing the links from C, the graph $G(V, E)$ is divided into two subgraphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$, such that $V_1 \cap V_2 = \emptyset$, $V_1 \cup V_2 = V$, the source $s \in V_1$ and the sink $t \in V_2$.

We are interesting in minimal cut sets. The following proposition specifies the minimal cut sets.

Proposition 2 Let $G(V, E)$ be an undirected connected network with source node s and sink node t. If the graph $G(V, E)$ is divided into two connected subgraphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$, such that $V_1 \cap V_2 = \emptyset$, $V_1 \cup V_2 = V$, the source $s \in V_1$ and the sink $t \in V_2$, then $C = E/(E_1 \cup E_2)$ is a minimal cut set.

Proof: From the Proposition 1 it is clear that C is a cut set. We only need to prove that C is a minimal cut set.
Suppose the opposite, i.e. C is not a minimal. That means that there is a cut set C', such that $C' \subset C$. Also it is clear that $C = \{\{u, v\} | u \in V_1 \text{ and } v \in V_2\}$. We will proof that there is a path from s to t in the graph $G'(V, E/C')$. Since $C' \subset C$ there is a link $\{u, v\} \in E/C'$ such that $u \in V_1$ and $v \in V_2$. V_1 is connected, so there is a path p_1 from s to u, and since V_2 is connected there is a path p_2 from v to t. Now the path $p_1\{u, v\}p_2$ is a path from s to t, which is a contradiction with the proposition that C' is a cut set. □

Example 1: Let us regard a network shown on Figure 1.

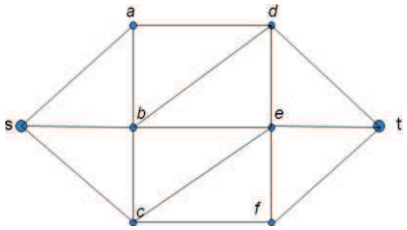

Figure 1

Figure 2 shows the set $C = \{(s, c), (c, b), (b, e), (e, d), (d, t)\}$ which is a minimal cut set.

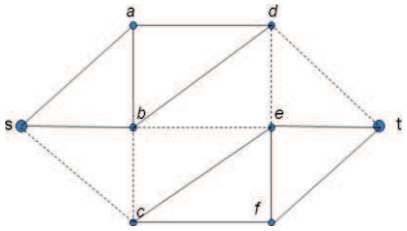

Figure 2

III. DESCRIPTION OF THE ALGORITHM

In the following section we give a new algorithm for obtaining minimal cut set. The idea of the algorithm is to obtain as less as possible candidates which are cut sets, but not a minimal cut sets.

Algorithm 1

Input: The set of nodes V and the set of links E, where the source s=1 and the sink t=$|V|$.

Output: The set of all minimal cut sets.

Step1: $A = \{\{\{\{s, a\} \mid \{s, a\} \in E\}, \{a \mid \{s, a\} \in E\}\}, \{s\}\}$
Step2: CutSet = $\{\{s, a\} \mid \{s, a\} \in E\}$
Step3: Until $A \neq 0$, take an element from the set A, $B = \{B_1, B_2, B_3\}$ and remove it from A.
Step 3.1: For each element $b \in B_2$, $b \neq t$, create
$B_3 = B_3 \cup \{b\}$
$B_2 = \{B_2 \setminus \{b\}\} \cup \{a \mid \{a, b\} \notin B_1\}$
$B_1 = \{B_1 \setminus \{\{a, b\} \mid \{a, b\} \in B_1\}\} \cup \{\{a, b\} \mid \{a, b\} \notin B_1\}$
CutSet = CutSet $\cup \{B_1\}$
$A = A \cup \{\{B_1, B_2, B_3\}\}$
Step 4: Delete all $Y \in$ CutSet, such that there is a set $X \in$ CutSet and $X \subseteq Y$.
Step 5: Print CutSet.

Lemma 1 In each iteration of Step 3.1 the graphs $G_1(B_3, \{\{u, v\} \mid u, v \in B_3\})$ and $G_2(V/B_3, \{\{u, v\} \mid u, v \in V/B_3\})$ separate the graph G into two unconnected components, such that s belong in G_1 and t belong in G_2.

Proof: It is clear that $B_3 \cup V/B_3 = V$ and $B_3 \cap V/B_3 = \emptyset$, so the graphs G_1 and G_2 separate graph G into two unconnected components. Because, in the Step 1 we initialize $B_3 = \{s\}$, and in each entry in Step 3.1 we are adding only one element in the set B_2, follows that $s \in G_1$. On the other side, we never add the sink into B_3, so $t \notin B_3$. This implies that s and t belong to different components. □

Theorem 1 For an input graph $G(V, E)$ with a source s and sink t, the elements of the set CutSet are cut sets.

Proof: Initially we have that CutSet = $\{B_1\}$, $B_1 = \{\{s, a\} \mid \{s, a\} \in E\}$, so before entry into the loop in Step 3.1, CutSet is consist of cut sets for the graph G. We are going to prove that the assertion "B_1 is a cut set" is an invariant of the loop in Step 3.1. Actually we will prove that

$$B_1 = \{\{u, v\} \mid u \in B_3, v \in V/B_3\} \quad (1)$$

is an invariant of the loop in Step 3.1.
Let us assume that before entry into the loop, B_1 is a cut set, precisely that $B_1 = \{\{u, v\} \mid u \in B_3, v \in V/B_3\}$. Let $B_3' = B_3 \cup \{b\}$, $B_2' = \{B_2 \setminus \{b\}\} \cup \{a \mid \{a, b\} \notin B_1\}$ and $B_1' = \{B_1 \setminus \{\{a, b\} \mid \{a, b\} \in B_1\}\} \cup \{\{a, b\} \mid \{a, b\} \notin B_1\}$. From the Lemma 1 follows that the graphs $G_1'(B_3', \{\{u, v\} \mid u, v \in B_3'\})$ and $G_2'(V/B_3', \{\{u, v\} \mid u, v \in V/B_3'\})$ separate the graph G into two components, such that $s \in B_3'$ and $t \in V/B_3'$. To prove that B_1' is cut set, we should to prove that $B_1' = \{\{u, v\} \mid u \in B_3', v \in V/B_3'\}$, i.e.

$$\{\{u, v\} \mid u \in B_3', v \in V/B_3'\} = \{B_1 \setminus \{\{a, b\} \mid \{a, b\} \in B_1\}\} \cup \{\{a, b\} \mid \{a, b\} \notin B_1\} \quad (2)$$

Let $\{u', v'\} \in \{\{u, v\} \mid u \in B_3', v \in V/B_3'\} \Leftrightarrow u' \in B_3' \wedge v' \in V/B_3' \Leftrightarrow u' \in B_3 \cup \{b\} \wedge v' \in V/\{B_3 \cup \{b\}\} \Leftrightarrow (u' \in B_3 \vee u' \in \{b\}) \wedge (v' \in V/\{B_3 \cup \{b\}\}) \Leftrightarrow (u' \in B_3 \wedge v' \in V/\{B_3 \cup \{b\}\}) \vee (u' = b \wedge v' \in V/(B_3 \cup \{b\}))$.
I case: $u' \in B_3 \wedge v' \in V/\{B_3 \cup \{b\}\} \Leftrightarrow u' \in B_3 \wedge v' \notin B_3 \cup \{b\} \Leftrightarrow u' \in B_3 \wedge \neg(v' \in B_3 \vee v' = b) \Leftrightarrow u' \in B_3 \wedge v' \in V/B_3 \wedge v' \neq b \Leftrightarrow (u' \in B_3 \wedge v' \in V/B_3) \wedge (u' \in B_3 \wedge v' \neq b) \Leftrightarrow (u', v') \in B_1 \wedge (u', v') \notin \{\{a, b\} \mid \{a, b\} \in B_1\}\} \Leftrightarrow \{u', v'\} \in \{B_1 \setminus \{\{a, b\} \mid \{a, b\} \in B_1\}\}$
II case: $u' = b \wedge v' \in V/\{B_3 \cup \{b\}\} \Leftrightarrow u' = b \wedge v' \in V/B_3 \wedge v' \neq b \Leftrightarrow u', v' \in V/B_3 \wedge v' \neq b \wedge u' = b$. Because of $B_1 = \{\{u, v\} \mid u \in B_3, v \in V/B_3\}$, $u', v' \in V/B_3$ if and only if $\{u', v'\} \notin B_1$. i.e. $\{u', v'\} \in \{\{a, b\} \mid \{a, b\} \notin B_1\}$.
From case I and case II follows that (2) is true. □

In order to prove that the Algorithm 1 provides all minimal cut sets we need the following proposition:

Proposition 3 Let $G(V, E)$ is a connected graph with $|V| > 1$. Then there is at least two nodes $x, y \in V$, such that graphs $G'(V/\{x\}, E / \{\{x, u\} \mid \{x, u\} \in E\})$ and $G''(V/\{x\}, E / \{\{y, u\} \mid \{y, u\} \in E\})$ are connected graphs.

Theorem 2 The Algorithm 1 provides all minimal cut sets for an input graph $G(V, E)$.

Proof: First we will prove that each set of nodes B_3, such that s belongs to it and $G_1(B_3, \{\{u, v\} \mid u, v \in B_3\})$ is connected, is received at last ones. It is clear that the only such set with only one node is $\{s\}$, $(G_1(\{s\}, \emptyset))$. We suppose that there are obtained all sets B_3, $|B_3| = k$, such that $s \in B_3$ and the subgraph $G_1(B_3, \{\{u, v\} \mid u, v \in B_3\})$ is connected, for k=1, 2,…, n-1.
Let Y be the set of nodes of G such that $|Y| = n$, $s \in Y$, and the graph $G'_1(Y, \{\{u, v\} \mid u, v \in Y\})$ is connected. From the Proposition 3 we have that there is node $b \neq s$, such that the graph $G''_1(Y', \{\{u, v\} \mid u, v \in Y'\})$, where $Y = Y' \cup \{b\}$ is a connected graph with $s \in Y'$. Since $|Y'| = n - 1$, from the inductive assumption we have that Y' is obtained by the Algorithm 1. Now, because there is a edge $\{u, b\} \in E$, $u \in Y'$, it is clear that $Y = Y' \cup \{b\}$ is obtain by the Algorithm 1.
At the end, since each minimal cut set divides the graph into two connected components and all such components are obtained by the Algorithm 1, is following that all minimal cut sets are obtained. □

A disadvantage of the Algorithm 1 is the fact that it is possible to obtain the same cut set many times. For that reason, we are going to propose an optimization of the previous algorithm. In order to do that, we add another set F. In this set F we keep all nodes $b \in B_2$, for which we have found nodes c, such that $\{b, c\} \in E$ and $\{b, c\}$ are not links from some cut set obtained before. In order to avoid

duplication of the same combination of nodes in the next iterations, we remove these nodes from the set B_2. With this we get improvement, so that the third component of the set A is obtained in one way only, and because of that, each element in A is obtained in one way only, i.e. each cut set is obtained only once. Because of that, the third component from the set A is not necessary to be kept.

The improved algorithm is:

Algorithm 2

Input: The set of nodes V and the set of links E, where the source s=1 and the sink t=$|V|$.

Output: The set of all minimal cut sets.

Step 1: $A = \{\{\{\{s, a\} \mid \{s, a\} \in E\}, \{a \mid \{s, a\} \in E\}\}\}$
Step 2: CutSet = $\{\{\{s, a\} \mid \{s, a\} \in E\}\}$
Step 3: While $A \neq \emptyset$ do steps 3.1 to 3.3.1.4.
Step 3.1: $B = A[1]$, $A = A \setminus \{B\}$
Step 3.2: $Z = B[2]$; n = Length (Z); $F = \emptyset$
Step 3.3: For i = 1 to n do
Step 3.3.1: If $Z[i] \neq t$ then
Step 3.3.1.1: $b = Z[i]$; $F = F \cup \{b\}$;
Step 3.3.1.2: $D = \{B[1] \setminus \{\{a, b\} \mid \{a, b\} \in B[1]\}\} \cup \{\{a, b\} \mid \{a, b\} \notin B[1]\}$, CutSet = CutSet $\cup$ $\{D\}$
Step 3.3.1.3: $H = \{Z \cup \{a \mid \{a, b\} \notin B[1]\}\} \setminus F$
Step 3.3.1.4: $A = A \cup \{\{D, H\}\}$
Step 4: Delete all $Y \in$ CutSet, such that there is a set $X \in$ CutSet and $X \subseteq Y$.
Step 5: Print CutSet.

IV. STEP-BY-STEP ILLUSTRATION

In this section we will illustrate the work of the Algorithm 2.

Example 2: Let we have the network in the Figure 1.
In the step 1 $A = \{\{\{\{s, a\}, \{s, b\}, \{s, c\}\}, \{a, b, c\}\}\}$ and CutSet = $\{\{s, a\}, \{s, b\}, \{s, c\}\}$.
From the set A, we found the set $B = \{\{\{s, a\}, \{s, b\}, \{s, c\}\}, \{a, b, c\}\}$ and the new A is $A = A \setminus B = \emptyset$. Figure 3 shows the cut set $\{\{s, a\}, \{s, b\}, \{s, c\}\}$.

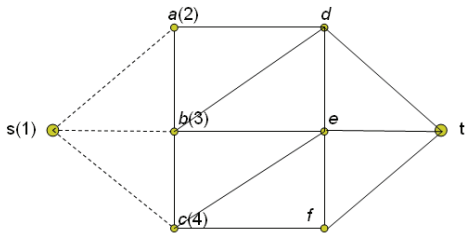

Figure 3

In this time $Z = \{a, b, c\}$, $n = 3$ and $F = \emptyset$.
For each element from Z, we obtain another cut set.
$Z[1] = a$, $F = \{a\}$
$D = \{\{s, b\}, \{s, c\}, \{a, b\}, \{a, d\}\}$ –cut set
$H = \{b, c, d\}$
$A = \{\{\{s, b\}, \{s, c\}, \{a, b\}, \{a, d\}\}, \{b, c, d\}\}\}$.
Figure 4 shows the cut set $\{\{s, b\}, \{s, c\}, \{a, b\}, \{a, d\}\}$.

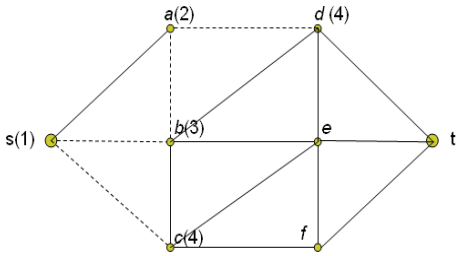

Figure 4

$Z[2] = b$, F = $\{a, b\}$
$D = \{\{s, a\}, \{s, c\}, \{b, a\}, \{b, d\}, \{b, c\}, \{b, e\}\}$-cut set
$H = \{c, d, e\}$
$A = \{\{\{s, b\}, \{s, c\}, \{a, b\}, \{a, d\}\}, \{b, c, d\}\}, \{\{\{s, a\}, \{s, c\}, \{b, a\}, \{b, d\}, \{b, c\}, \{b, e\}\}\}, \{c, d, e\}\}\}$
Figure 5 shows the cut set $\{\{s, a\}, \{s, c\}, \{b, a\}, \{b, d\}, \{b, c\}, \{b, e\}\}$

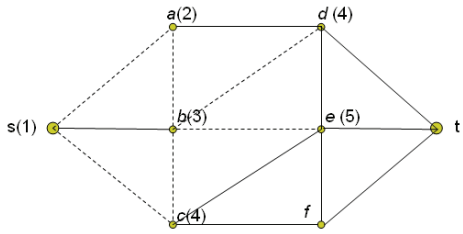

Figure 5

$Z[3] = c$. $F = \{a, b, c\}$
$D = \{\{s, a\}, \{s, b\}, \{b, c\}, \{c, e\}, \{c, f\}\}$-cut set
$H = \{e, f\}$
$A = \{\{\{s, b\}, \{s, c\}, \{a, b\}, \{a, d\}\}, \{b, c, d\}\}, \{\{\{s, a\}, \{s, c\}, \{b, a\}, \{b, d\}, \{b, c\}, \{b, e\}\}\}, \{c, d, e\}\}\}, \{\{\{s, a\}, \{s, b\}, \{b, c\}, \{c, e\}, \{c, f\}\}, \{e, f\}\}$.
Figure 6 shows the cut set $\{\{s, a\}, \{s, b\}, \{b, c\}, \{c, e\}, \{c, f\}\}$

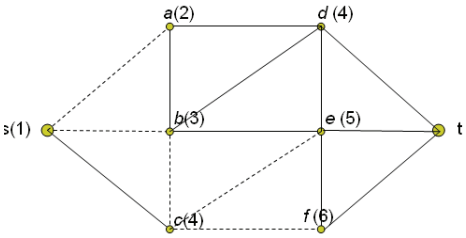

Figure 6

The algorithm continues in the same manner, until all cut sets are produced. At the end, all derived cut sets that are not minimal are deleted.

V. OPTIMIZATION OF THE BASIC ALGORITHM

By the preceding algorithm there are obtained candidates for minimal cut sets, which are not minimal cut sets. In order to avoid that appearance, we will analyze how that sets are obtained.

Example 3: Let us regard the cut set $\{\{\{s, a\}, \{s, c\}, \{b, c\}, \{b, a\}, \{b, d\}, \{b, e\}\}, \{c, d, e\}\} \in A$, Figure 5. Using this set there are obtained few new candidates for a minimal cut set.

$Z[1] = d$, $F = \{a, b, c, d\}$

$D = \{\{s, a\},\{s, c\},\{b, a\},\{b, c\},\{b, e\},\{a, d\}, \{d, e\}, \{d, t\}\}$ – cut set. $H = \{e\}$

The set $\{\{s, a\},\{s, c\},\{b, a\},\{b, c\},\{b, e\},\{a, d\}, \{d, e\}, \{d, t\}\},\{e\}\}$ is not minimal cut set, because the set $\{\{s, c\}, \{c, b\}, \{b, e\}, \{e, d\}, \{d, t\}\}$ is a smaller cut set.

For these reasons, in the Step 4 of the Algorithm 2 given in the previous part, we need to remove all candidates for minimal cut vectors, which are not minimal ones. It can be done with a minimum of $\ln(N)$ comparisons between sets, where N is the number of the elements in the CutSet, just before the Step 4. In order to remove that testing, we will make a modification of the algorithm. Note that in the already proposed algorithms, like in [6], there are always candidates for minimal cut sets that are not minimal.

From the Proposition 2 we have that when the obtained cut set is not a minimal cut set, the graph $G(V,E)$ is divided into more than two connected components, as it is shown in Figure 7.

The cut set $\{\{s, a\}, \{s, c\}, \{b, a\}, \{b, c\}, \{b, e\}, \{a, d\}, \{d, e\}, \{d, t\}\}$ is not a minimal cut set and it divides the graph $G(V, E)$ into three components, Figure 7.

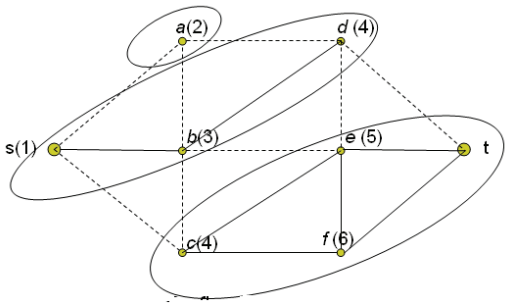

Figure 7

The idea of the new algorithm is to check whether by removing the links from the new candidate for the minimal cut set, the obtained subgraph is composed of two connected components, such that, the source is in one of them, and the sink in the other. So, from Proposition 1, we need to check that the graphs $(B_3, \{\{u, v\}| u, v \in B_3\}\}$ and $(V/B_3, \{\{u, v\}| u, v \in V/B_3\}\}$ are connected graphs.

Proposition 4 In each iteration of Step 3.1 the graph $G(B_3, \{\{u, v\}| u, v \in B_3\})$ is connected graph.

Proof: Initially we have that $B_3 = \{s\}$, from this it is clear that before entering in the loop in Step 3.1, the graph $G(B_3, \{\{u, v\}| u, v \in B_3\})$ is a connected graph. We are going to prove that assertion "$G(B_3,\{\{u, v\}| u, v \in B_3\})$ is a connected graph" is invariant of the loop in the Step 3.1.

Assume that before entering the loop in Step 3.1, $G(B_3, \{\{u, v\}| u, v \in B_3\})$ is a connected graph. We need to prove that the graph $G'(B_3',\{\{u, v\}| u, v \in B_3'\})$ is a connected graph.

Let $B_3' = B_3 \cup \{b\}$ and $B_2' = \{B_2 \setminus \{b\}\} \cup \{a \mid \{a, b\} \notin B_1\}$. From $b \in B_2$, follows that b is connected with at least one element of B_3, so $G'(B_3 \cup \{b\}, \{\{u, v\}| u, v \in B_3 \cup \{b\}\})$ is a connected graph. □

It remains to check that the graph $(V/B_3, \{\{u, v\}| u, v \in V/B_3\})$ is connected. For that reason, at the beginning we construct a tree with root in the sink. In the Step 3.3.1.1 from Algorithm 2, when we choose a new node b we check whether it is a leaf in the tree.

- When the node b is leaf, then it is clear that the subgraph has the required property. So, we construct another tree by removing that leaf, and we continue from Step 3.3.1.2 to Step 3.3.1.4 from Algorithm 2.
- When the node b is not a leaf, then we remove it from the tree and we check whether the nodes that are descendants of that node are connected with another node of the rest of the tree. If all nodes are connected with another node of the rest of the tree, then the obtained set is a minimal cut set, so we construct a new tree and we continue from Step 3.3.1.2 to Step 3.3.1.4 from Algorithm 2. If at least one node is not connected with the rest of the tree, then the set is not a minimal cut set, and we take another element from A.

Note that when all links of the graph are important, i.e. the graph as a system is a coherent system, the appropriate minimal cut set which is subset of some cut set, which is obtained by the algorithm will be obtained on the another way. This means that if the graph is coherent, then whenever we obtain a candidate for the minimal cut set, which is not a minimal cut set, the appropriate set may not be added in A.

Now, we give the new improved algorithm.

Algorithm 3

Input: The set of nodes V and the set of links E, where the source s=1 and the sink t=$|V|$.

Output: The set of all minimal cut sets.

Step 1: $A = \{\{\{s, a\} \mid \{s, a\} \in \text{E}\}, \{a \mid \{s, a\} \in E\}\}$, $T\}$, T is a tree with root in a sink.

Step2: CutSet = $\{\{s, a\} \mid \{s, a\} \in E\}$

Step3: Until $A \neq 0$, take an element from the set A, $B = \{B_1, B_2, T\}$ and remove it from A.

Step 3.1: For each element $b \in B_2$, $b \neq t$, check is b is a leaf

If b is leaf then

$B_2 = \{B_2 \setminus \{b\}\} \cup \{a \mid \{a, b\} \notin B_1\}$

$B_1 = \{B_1 \setminus \{\{a, b\}| \{a, b\} \in B_1\}\} \cup \{\{a, b\}| \{a, b\} \notin B_1\}$

$T = T / \{b\}$

CutSet = CutSet $\cup$ $\{B_1\}$

else

If each descendent nodes b' of the node b are connected with another node of the rest of the tree than

$B_2 = \{B_2 \setminus \{b\}\} \cup \{a \mid \{a, b\} \notin B_1\}$

$B_1 = \{B_1 \setminus \{\{a, b\} | \{a, b\} \in B_1\}\} \cup \{\{a, b\} | \{a, b\} \notin B_1\}$

CutSet = CutSet $\cup \{B_1\}$

Add the new edges in the tree T that connect b' with the rest of the tree, and remove the old edges.

else

If at least one node is not connected with the rest of the tree then go to Step 2.1

$A = A \cup \{\{B_1, B_2, T\}\}$

Step 4: Print CutSet.

The following example illustrates the work of Algorithm 3.

Example 4:

1. When the node is a leaf

$A = \{\{\{\{s, a\}, \{s, b\}, \{s, c\}\}, \{a, b, c\}\}\}$

$B_3 = \{s\}$, $V / B_3 = \{a, b, c, d, e, f\}$

Figure 8 shows appropriate tree $\{\{t, \{d, e, f\}\}, \{d, \{a, b\}\}, \{e, \{c\}\}, \{f, \{\}\}, \{a, \{\}\}, \{b, \{\}\}, \{c, \{\}\}\}$.

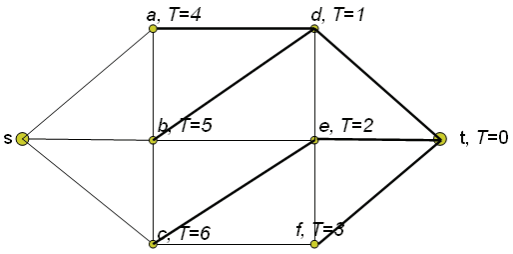

Figure 8

Because, the node f is leaf, it can be removed from the tree. It is obvious that the rest of the tree is connected.

2. When the node is not a leaf, but all its descendants are connected with another node from the rest of the tree.

Let the started cut set is $\{\{s, a\}, \{s, c\}, \{b, c\}, \{b, e\}, \{b, a\}, \{b, d\}\}, \{c, d, e\}$ and we consider a node e. The appropriate tree for this set is $\{t, \{d, e, f\}\}, \{d, \{a\}\}, \{e, \{c\}\}, \{f, \{\}\}, \{a, \{\}\}, \{c, \{\}\}\}$. After removing of the node e we obtain the tree $\{t, \{d, f\}\}, \{d, \{a\}\}, \{f, \{\}\}, \{a, \{\}\}\}$, Figure 9.

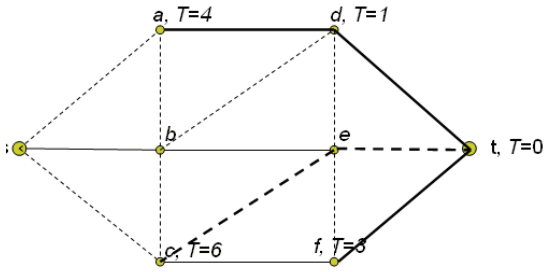

Figure 9

The node c is descendant of e. This node is connected with f. The new tree $\{t, \{d, f\}\}, \{d, \{a\}\}, \{f, \{c\}\}, \{a, \{\}\}, \{c, \{\}\}\}$ is shown on Figure 10. The obtained minimal cut set is $\{\{s, a\}, \{s, c\}, \{a, b\}, \{b, c\}, \{b, d\}, \{c, e\}, \{e, f\}, \{e, d\}, \{e, t\}$.

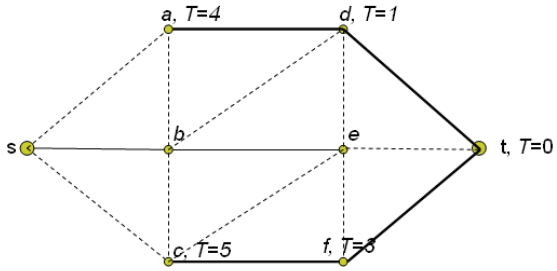

Figure 10

3. When the node is not a leaf and at least one node is not connected with the rest of the tree. Regard the set $\{\{s, a\}, \{s, c\}, \{b, c\}, \{b, e\}, \{b, a\}, \{b, d\}\}, \{c, d, e\}$ and let take the node d. Appropriate tree for this set is $\{t, \{d, e, f\}\}, \{d, \{a\}\}, \{e, \{c\}\}, \{f, \{\}\}, \{a, \{\}\}, \{c, \{\}\}\}$. After removing d, (the links $\{a, d\}$ and $\{d, t\}$) we obtain the tree $\{t, \{e, f\}\}, \{e, \{c\}\}, \{f, \{\}\}, \{c, \{\}\}\}$, Figure 11.

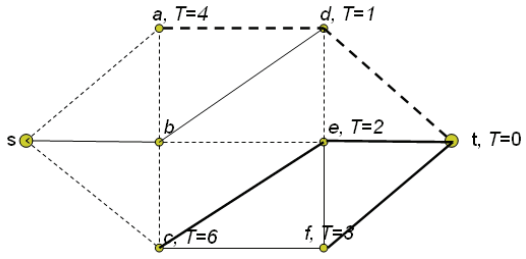

Figure 11.

The node a is descendant of the node d. This node is not connected with e and f, i.e. it is not connected with a node from the rest of the tree. So, this candidate for a minimal cut set is not a minimal cut set.

VI. CONCLUSION

These paper present new algorithms for calculating a minimal cut sets. With the first two proposed algorithms are obtained candidates for minimal cut sets, which are actually not the minimal cut sets. In order to avoid this, we make optimisation of the basic algorithms. With the optimized algorithm are obtained only the minimal cut set. The main advantage on this algorithm compared with the other known algorithm, is that we obtain only minimal cut sets, without additional request to make control whether these cut sets are minimal.

REFERENCES

[1] Ramirez-Marquez, J.E. and Coit, D. (2003), "Alternative Approach for Analyzing Multistate Network Reliability" *IERC Conference Proceedings 2003*.

[2] Ramirez-Marquez, J.E., Coit, D. and Tortorella, M., "Multi-state Two-terminal Reliability: *A Generalized Cut-Set Approach", Rutgers University IE Working Paper*.

[3] Mihova,M., Synagina,N., "An algorithm for calculating multi-state network reliability using minimal path vectors", The 6th international conference for Informatics and In-formation Technology (CIIT 2008)

[4] Mihova M., Maksimova N., Popeska ·Z. "An algorithm for calculating multi-state network reliability with arbitrary capacities of the links"- Fourth International Bulgarian-Greek Conference Computer Science'2008 ,170-175

[5] М.Михова, "Моделирање на надежност на непоправливи повеќе-состојбени системи со независни компоненти и хомогени маркови транзиции," докторска дисертација,*Природно-математички факултет,* Скопје 2008
[6] Bethel A. Gebre ,Jose E. Ramirez-Marquez ," Element substitution algorithm for general two-terminal network reliability analyses", IIE Transactions, Volume 39, March 2007, 265 - 275