Algorithms to evaluate the reliability of a network

J. Galtier*, A. Laugier* and P. Pons†
*France Telecom, R&D Division
905 rue Albert Einstein, F-06921 Sophia Antipolis Cedex, France
Email: {jerome.galtier,alexandre.laugier}@francetelecom.com
†Ecole Normale Supérieure
45 rue d'Ulm, F-75005 Paris, France
Email: ppons@clipper.ens.fr

***Abstract*— We consider problems of network reliability: the two-terminal network reliability consists, given an undirected graph $G = (V, E)$, and a series of independent edge failure events, in computing the probability that two nodes remain connected. The all-terminal network reliability is the probability that the whole network remains connected. We present in the following two different approaches to compute two-terminal and all-terminal reliability, with various characteristics on the precision level of the result. We give an exact algorithm to compute the reliability in $O(|V|f(w)^2 + |E|f(w))$ with $f(x) = \left(\frac{x}{\ln x}\right)^x e^{-(1+o(1))x}$ and w is the tree-width of G. We also present polynomial methods to give bounds on the reliability. We discuss methods to optimize the *mean time to repair* of the components.**

Keywords : two-terminal reliability, all-terminal reliability, tree-width, frontal methods, polynomial approximations.

I. Introduction

Telecommunication network designers aim at drawing fault tolerant networks, but high connectivity solutions are generally too expensive. The usual client requirement for nowadays secured systems is 2-connectivity, which means that the network still works after one failure. However, this single constraint seems to be not enough to comply with the classical contractual conditions. For instance, the France Telecom Transfix 2.0 service guarantees a rate of reliability of $1.5.10^{-3}$, which means that the link has to be down for less than 6 hours and 35 minutes every six months. As a result, a client willing a higher reliability rate has to use at least 2-connectivity and combine properly the elements in order to achieve the required reliability rate. Since this criterion becomes an important element of the desired quality of service of a client network, some algorithms and methods are required to evaluate it as efficiently as possible.

A simple model consists in viewing the network as a graph where a ramdom function is defined on the different components, so the probability space emphasizes the different failure events. Different criteria can be considered in order to express the reliability of a network. The main ones are the four following :

- The first one is devoted to telecommunication network operators. It can be defined as the ratio between the expected traffic (the sum of the products of a demand by the probability that the demand can reach its destination) and the total desired traffic (the sum of the demands).
- The second considers for a given demand the ratio between the expected traffic serviced and the demand.
- The third considers for each demand the probability that there exists at least a path linking the origin of the demand to its destination.
- The fourth considers the probability that the network is connected.

Notice that the three latter criteria are more devoted to a customer approach than the first one. In the remainder we will focus on the third and fourth criterions.

Our problem can be formulated as follows: given an undirected graph $G = (V, E)$, we consider independent failures on the edges in E. Each failure is associated with its edge e and occurs randomly and independently to the other ones with probability p_e. We are given the probability space Ω where the failures are represented by a series of independent 0/1 variables Y_e with $P[Y_e = 0] = p_e$. Given an event ω in Ω we consider the graph $G_\omega = (V, E_\omega)$ defined by

$$E_\omega = \{e \in E, Y_e(\omega) = 1\}$$

and for any two vertices s and t of V we have the random variable $X_{s,t}$ defined for each probability event ω by

$$\begin{array}{rcll} X_{s,t}(\omega) & = & 1 & \text{if there exists a path} \\ & & & \text{between } s \text{ and } t \text{ in } G_\omega, \\ X_{s,t}(\omega) & = & 0 & \text{if not.} \end{array}$$

We aim at evaluating $E[X_{s,t}]$. This problem, called the two-terminal network reliability problem, is well-known for being $\#P$-hard (see [22] and [7]). A similar problem is to consider the random variable Z defined for each probability event ω by:

$$\begin{array}{rcll} Z(\omega) & = & 1 & \text{if the graph } G_\omega \text{ is connected} \\ Z(\omega) & = & 0 & \text{otherwise.} \end{array}$$

And our goal is to compute $E[Z]$. This problem is called the all-terminal network reliability problem.

In [1] M. Ball and J. Provan gave an upper bound and a lower bound of the reliability of systems and they showed that the bounds are equal if the system consists in a matroid in the following sense. Let K be the set of components and let $\mathcal{P}$ be a family of subsets of K and suppose the system operates if at least one element of $\mathcal{P}$ operates, then one says the system consists in a matroid if $\mathcal{P}$ is a matroid. More recently,

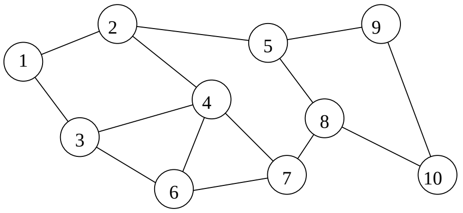
D. Karger investigates the all-terminal network reliability and gives a randomized polynomial time approximation scheme for this problem, [15]. R. Kannan mentions the network reliabilty as an important open problem for which an approximation would be useful, see [14]. Also, algorithms polynomial in the number of minimal cuts have been proposed [11], [23], but this number can grow tremendously with the size of the network. Also an estimation of the two-terminal reliability was recently given in the case where the graph is planar, [6]. We now give some insight in order to illustrate the problem of interest. We will call equipment any device which is useful for the proper functioning of networks, for instance a switch, a multiplexer or electrical energy suppliers, etc... To each equipement are associated a mean time separating two consecutive failures, say $Mtbf$, and a mean time to repair, say $Mttr$. We make the assumption that all the failures are independent. Thus the probability that an equipment i does not work is given by the ratio $\frac{Mttr_i}{Mtbf_i}$. We will call a network component every switching node or transmission link. A component is composed by several equipments. For instance, in an SDH transmission link, it may be composed by an optical fiber with optical multiplexers; in case of WDM networks, by some add-drop multiplexers. Thus it is possible to compute for each network component its failure probability.

The article is organized as follows. In the next section we detail our exact algorithm and analyze its complexity. In the third section we focus on various bounds derived in polynomial time. Finally, in section 4 we derive results on the practical problem of return *mean times to repair* for a real use of these results to manage repairing teams.

II. AN EXACT ALGORITHM FOR THE SIMPLIFIED PROBLEM

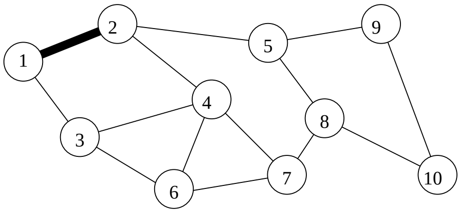
The basic idea of the algorithm presented here relies on the notion of a frontal description of a graph. We use the definition of tree-decomposition of Robertson and Seymour [24]. Many similar approaches have been developed in the past [3], [9], [13], [18], even in related domains such as resilience [20]. The originality of the present work is in the application to the two-terminal and all-terminal reliability problem with close analysis of the complexity depending on the tree-width.

In figures 1 to 9, we describe on a practical example how our algorithm works. Our goal is to compute the reliability of the connection of node 1 to node 10, given that each edge has a reliability of 50%. We present on the left part of each figure, the network with in bold font the edges that were already taken into consideration. On the right part, there is a partition table, that is, each possible partition with the elements of the front on the first column, and the associated probability on the second column. In all, in figure 1, there is only one node in the front, and therefore one possible partition, which occurs with a probability of 100%. In figure 2, an edge is taken into consideration, which connects the two nodes with a probability of 50%, and otherwise lefts them unconnected. The table grows up to figure 6 where node 3 is no more usefull for the rest of the computation. Some states need to be merged, such as the ones highlighted in figure 7. The simplified partition is presented in figure 8, and the computation so proceeds in figure 9. In the end, when all the intermediate nodes have been taken into account and eliminated, the partition table is based on the set $\{1, 10\}$, and our 2-terminal reliability is equal to the probability that the complete partition $\{\{1, 10\}\}$ occurs, as opposed to the separated partition $\{\{1\}, \{10\}\}$.



{1}	100%

Fig. 1. A frontal approach: front includes node 1.



{1}{2}	50%
{1,2}	50%

Fig. 2. A frontal approach: front includes nodes 1 and 2.

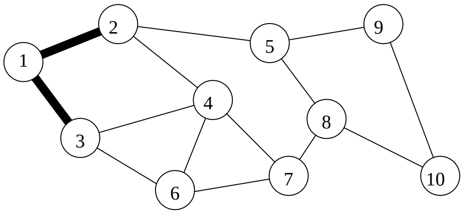
Definition 2.1: A tree-decomposition of G is a family $(X_i, i \in I)$ of subsets of V, together with a tree T with vertices indexed by I with the following properties:

(i) $\cup_{i \in I} X_i = V$.

(ii) Every edge of G has both its endpoints in some X_i ($i \in I$).

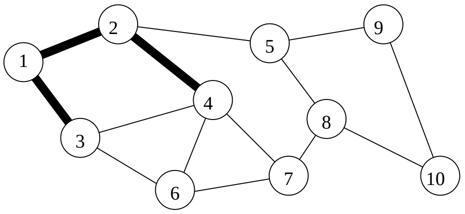
(iii) For i, j, and k in I, if j lies on the path of T from i to k then $X_i \cap X_k \subseteq X_j$.

Accordingly, the width w of the tree decomposition is



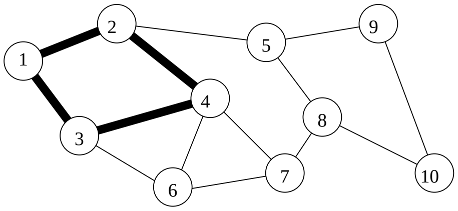
{1}{2}{3}	25%
{1,2}{3}	25%
{1}{2,3}	(0%)
{1,3}{2}	25%
{1,2,3}	25%

Fig. 3. A frontal approach: front includes nodes 1, 2 and 3.



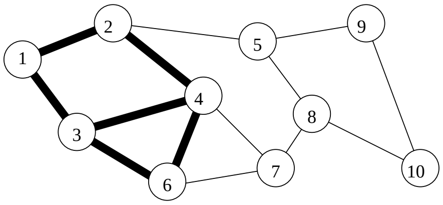
{1}{2}{3}{4}	12,5%
{1}{2,4}{3}	12,5%
{1,2,4}{3}	12,5%
{1,2}{3}{4}	12,5%
{1,3}{2}{4}	12,5%
{1,3}{2,4}	12,5%
{1,2,3}{4}	12,5%
{1,2,3,4}	12,5%

Fig. 4. A frontal approach: front includes nodes 1, 2, 3, and 4, edge {2,4}, but not edge {3,4}.



{1}{2}{3}{4}	6,25%
{1}{2,4}{3}	6,25%
{1,2,4}{3}	6,25%
{1,2}{3}{4}	6,25%
{1,3}{2}{4}	6,25%
{1,3}{2,4}	6,25%
{1,2,3}{4}	6,25%
{1,2,3,4}	31,25%
{1}{2}{3,4}	6,25%
{1}{2,3,4}	6,25%
{1,3,4}{2}	6,25%
{1,2}{3,4}	6,25%

Fig. 5. A frontal approach: front includes completely node 1, 2, 3, and 4.

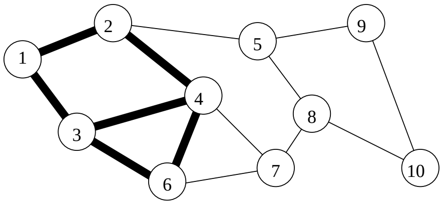


{1}{2}{3}{4,6}	1,56%
{1}{2,4,6}{3}	1,56%
{1,2,4,6}{3}	1,56%
{1,2}{3}{4,6}	1,56%
{1,3}{2}{4,6}	1,56%
{1,3}{2,4,6}	1,56%
{1,2,3}{4,6}	1,56%
{1,2}{3,4}{6}	1,56%

{1}{2}{3,6}{4}	1,56%
{1}{2,4}{3,6}	1,56%
{1,2,4}{3,6}	1,56%
{1,2}{3,6}{4}	1,56%
{1,3,6}{2}{4}	1,56%
{1,3,6}{2,4}	1,56%
{1,2,3,6}{4}	1,56%
{1,3,4}{2}{6}	1,56%

{1}{2}{3}{4}{6}	1,56%
{1}{2,4}{3}{6}	1,56%
{1,2,4}{3}{6}	1,56%
{1,2}{3}{4}{6}	1,56%
{1,3}{2}{4}{6}	1,56%
{1,3}{2,4}{6}	1,56%
{1,2,3}{4}{6}	1,56%
{1}{2,3,4}{6}	1,56%
{1}{2}{3,4}{6}	1,56%
{1}{2}{3,4,6}	6,25%
{1}{2,3,4,6}	6,25%
{1,3,4,6}{2}	6,25%
{1,2}{3,4,6}	6,25%
{1,2,3,4}{6}	7,81%
{1,2,3,4,6}	28,13%

Fig. 6. A frontal approach: front includes nodes 1, 2, 3, 4 and 6.

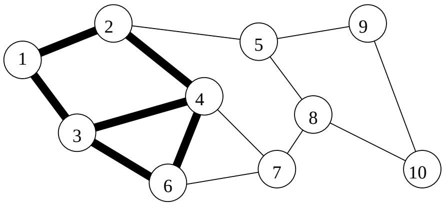


{1}{2}{3}{4,6}	1,56%
{1}{2,4,6}{3}	1,56%
{1,2,4,6}{3}	1,56%
{1,2}{3}{4,6}	1,56%
{1,3}{2}{4,6}	1,56%
{1,3}{2,4,6}	1,56%
{1,2,3}{4,6}	1,56%
{1,2}{3,4}{6}	1,56%

{1}{2}{3,6}{4}	1,56%
{1}{2,4}{3,6}	1,56%
{1,2,4}{3,6}	1,56%
{1,2}{3,6}{4}	1,56%
{1,3,6}{2}{4}	1,56%
{1,3,6}{2,4}	1,56%
{1,2,3,6}{4}	1,56%
{1,3,4}{2}{6}	1,56%

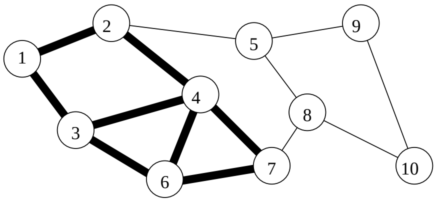
{1}{2}{3}{4}{6}	1,56%
{1}{2,4}{3}{6}	1,56%
{1,2,4}{3}{6}	1,56%
{1,2}{3}{4}{6}	1,56%
{1,3}{2}{4}{6}	1,56%
{1,3}{2,4}{6}	1,56%
{1,2,3}{4}{6}	1,56%
{1}{2,3,4}{6}	1,56%
{1}{2}{3,4}{6}	1,56%
{1}{2}{3,4,6}	6,25%
{1}{2,3,4,6}	6,25%
{1,3,4,6}{2}	6,25%
{1,2}{3,4,6}	6,25%
{1,2,3,4}{6}	7,81%
{1,2,3,4,6}	28,13%

Fig. 7. A frontal approach: some states equivalent when eliminating node 3.



{1,2,4,6}	29,68%
{1,2,4}{6}	10,94%
{1,2}{4,6}	9,37%
{1}{2}{4,6}	9,37%
{1}{2,4,6}	9,37%
{1}{2}{4}{6}	6,25%
{1}{2,4}{6}	6,25%
{1,2}{4}{6}	6,25%
{1,4,6}{2}	6,25%
{1,6}{2,4}	1,56%
{1,2,6}{4}	1,56%
{1,6}{2}{4}	1,56%
{1,4}{2}{6}	1,56%

Fig. 8. A frontal approach: reduced table after eliminating node 3.



{1,2,7}	29,29%
{1,2}{7}	29,29%
{1}{2}{7}	31,62%
{1,7}{2}	6,64%
{1}{2,7}	3,12%

Fig. 9. A frontal approach: simplified front containing nodes 1, 2, and 7.

$\max_{i\in I} |X_i| - 1$. The tree-width of a graph can be approximated within $O(\log(|V|)$, but there is no polynomial time algorithm that approximates w within a fixed constant unless $P = NP$ [4].

Now select an $r \in I$ to be a root in T. We say that $j \in I$ is a *descendant* of $i \in I$ if i lies on the path in T between r and j, and we note D_i the set of descendants of i. A neighboring descendant is also called a *son*. Given two vertices s and t, the front in i is given by

$$F_i = X_i \cup (\{s,t\} \cap \cup_{j\in D_i} X_j) .$$

According to definition 2.1, we arbitrarily choose one index i_e such that X_{i_e} contains both endpoints of e.

A. The algorithm for the two-terminal network reliability problem

We consider a bottom-up *visit* of T as follows. Suppose that we are visiting k. We aim at collecting in k all the information on failures of edges e such that i_e is a descendant of k. Any state of availibility/failure of the edges creates a partition among the vertices of F_k. Equivalence classes are given by the property "there exists a path made of available edges e between those two vertices such that i_e is a descendant of k". The probability that one equivalence class appears equals the sum of the probabilities of the reliability/failure states giving this partition.

We maintain during our visit the probabilities of appearance of any partition of F_k. In the end, only s and t remain in F_r, and two possible states: either they are connected or not. The propability of the case where they are connected gives our two-terminal reliability factor.

We store partitions with their probabilities of appearance in a partition table PT. The basic steps are as follows (we explain after the notions of merging, adding and removing elements).

Initialization

For any leaf i of T, we create a partition table $PT(i)$ of single vertices of $F_i = X_i$ with probability 1.

Visit of k

- If k is not a leaf, we merge in $PT(k)$ all the partition tables $PT(j)$ of the sons j of k.
- We add as singles in $PT(k)$ the vertices of F_k that are not already in.
- For any e such that $i_e = k$ we add e in $PT(k)$.
- If $k \neq r$, we remove from $PT(k)$ any vertex $v \notin \{s,t\}$ that is not in F_l, with k son of l.

A *merging* of two partition tables PT_1 and PT_2 consists in computing for each state s_3 of the resulting partition table

$$Pr(s_3) = \sum Pr(s_1)Pr(s_2)$$

where the summation is over all partitions s_1 and s_2 that combine to form s_3. Using a proper algorithm for identifying disjoint sets [8, pp. 440-464], s_3 can be obtained from s_1 and s_2 in $O(w \log^*(w))$ steps, given that no set is larger than w, where the log^* function calculates how many times one would need to take the log of a number before one would go below 2.

To *add* an edge e, we create a new partition table PT_{new} from PT_{old}. Each state s with probability π in PT_{old} is reconducted in PT_{new} with probability $\pi \cdot p_e$. Also for any state s with probability π of PT_{old} we merge the classes containing the two endpoints of e into s' and add to the probability of appearance of s' in PT_{new} the value $\pi \cdot (1 - p_e)$.

To *remove* a vertex v in a partition table PT_{old} we forget v and merge all the states that differ only in v by summing their probabilities.

B. Complexity

Claim 2.1: Our algorithm for computing the two-terminal reliability runs in $O(|V|f(w)^2 + |E|f(w))$ with $f(x) = \left(\frac{x}{\ln x}\right)^x e^{-(1+o(1))x}$ and w is the tree-width of G.

Proof: The algorithm uses extensively partitions of a set of n elements. The number B_n of such partitions is also known as the Bell number [2] and [19]. The asymptotic formula of de Bruijn [10] gives

$$\frac{\ln(B_n)}{n} = \ln(n) - \ln\ln(n) - 1 + o(1).$$

So the complexity of our algorithm is bounded by the operations of merging partitions tables, adding edges and removing vertices. There are at most $2|V|$ elements in T (making the assumption that two X_i, X_j with $i \neq j$ differ from at least one vertex), and therefore $|V|$ nodes with two or more sons. Moreover, each partition table has at most $\max_{i \in I} |X_i \cup \{s,t\}| \leq w + 3$ elements. A merging of k sons can then be done in $(k-1)B_{w+3}^2 O(w \log^*(w))$ operations, leading to a total of $O(|V|B_{w+3}^2 w \log^*(w))$ steps in merging. Also adding an edge can be done in $O(B_{w+3})$ operations. A node is removed only once in the visit of the tree with the same bound of $O(B_{w+3})$ operations per node. Therefore the algorithm takes $O(|V|B_{w+3}^2 w \log^*(w) + |E|B_{w+3})$ operations. Using the formula above, we obtain a bound of $O(|V|f(w)^2 + |E|f(w))$ with $f(x) = \left(\frac{x}{\ln x}\right)^x e^{-(1+o(1))x}$. ■

Other approaches, such as the one of [25], present a *nested dissection* algorithm (this vocabulary is derived from [12]), which is in our case less efficient. Indeed, a nested dissection approach has the consequence of *doubling* the size of the front, and recall that $f(2x)/f(x)^2 = 4^{x(1+o(1))}$. This guarantees the improvement of our method when w is more than $\log_2 |V|/2$. Even lesser values can lead to an improvement, since in fact the bisection width is increased when more merging is required in our algorithm.

C. The algorithm for the all-terminal network reliability problem

For sake of completeness, we describe here the same algorithm in the all-terminal problem case. The ultimate goal is to calculate the probability that *all* terminals are connected. The basic steps are as follows (we explain after the notions of merging, adding and removing elements).

Initialization

For any leaf i of T, we create a partition table $PT(i)$ of single vertices of $F_i = X_i$ with probability 1.

Visit of k

- If k is not a leaf, we merge in $PT(k)$ all the partition tables $PT(j)$ of the sons j of k.
- We add as singles in $PT(k)$ the vertices of F_k that are not already in.
- For any e such that $i_e = k$ we add e in $PT(k)$.
- If $k \neq r$, we remove from $PT(k)$ all vertices that are not in F_l, with k son of l.

A *merging* of two partition tables PT_1 and PT_2 consists in computing for each state s_3 of the resulting partition table

$$Pr(s_3) = \sum Pr(s_1)Pr(s_2)$$

where the summation is over all partitions s_1 and s_2 that combine to form s_3.

To *add* an edge e, we create a new partition table PT_{new} from PT_{old}. Each state s with probability π in PT_{old} is reconducted in PT_{new} with probability $\pi \cdot p_e$. Also for any state s with probability π of PT_{old} we merge the classes containing the two endpoints of e into s' and add to the probability of appearance of s' in PT_{new} the value $\pi \cdot (1 - p_e)$.

To *remove* a vertex v in a partition table PT_{old} we forget v, we drop from the list all the states where v appears as a singleton, and merge all the remaining states that differ only in v by summing their probabilities.

Once T has been entirely visited, the all-terminal reliability is given by the probability that all nodes of r are connected in $PT(r)$.

Since the front has at most a size of $w+1$, we easily deduce that the complexity is also given by $O(|V|f(w)^2 + |E|f(w))$ with $f(x) = \left(\frac{x}{\ln x}\right)^x e^{-(1+o(1))x}$.

III. Polynomial bounds

We will approximate the actual reliability of the network by computing a lower bound and an upper bound of the reliability using the fact that paths and cuts form a blocking pair of binary clutters (for definition and properties related to Lehman's theorem one can see [21, pages 86, 562, and 601]). In this part, the graph considered can be either undirected or directed. Note that the oriented case introduces in general more difficulty, see [7]. Therefore the bounds given in the following apply in a more general case. Although we consider only edge failures in this section our approach remains general in the oriented case since we can introduce node failures with probabilty p_n by operating a Mengerian transformation. Thus the problem is reduced to the previous one if we give a failure probability p_n to the edge arising after the transformation. Note also that a lower bound for the two-terminal reliability problem remains true for the all-terminal reliability problem.

Our approach for determining easily computable bounds for large networks is based on computing the reliability of monotone structures. Computing bounds of the reliability of such structures is extensively exposed in [17]. One will notice that the clutters we use are special cases of the clutters used by the previous authors in the sense that the clutters we use have the Maxflow-Mincut property, see [26].

A. Upper bound and lower bound of the reliability

Let p_e denote the failure probability of edge e and D_{st} the probability that there exists an available path between s and t. Consider a cut $\delta(S)$ separating t from s. As a definition of cut this set of edges intersects all the paths linking s to t. We will say that a cut is available if there is at least one of its edges working, thus its availability is given by $1 - \prod_{e\in\delta(S)} p_e$.

Claim 3.1: The availability of the st-cut minimizing $1 - \prod_{e\in\delta(S)} p_e$ is an upper bound of the st-reliability for the network :

$$D_{st} \leq Min\{1 - \prod_{e\in\delta(S)} p_e,\ S \subset V, S \neq \emptyset,\ S \neq V\}.$$

We will denote with d_{st} this measure of the reliability.

Proof: Immediate if we remark that considering the measure of the reliability yielded by the edges of the cut with minimum availability leads us to consider that all the other cuts are considered as infallible. ■

Now we check that the measure of the reliability induced by the cut with minimum availability can be computed in an easy way.

Claim 3.2: As an upper bound for the st-reliability of the network the availability of the cut minimizing $1-\prod_{e\in\delta(S)} p_e$ can be computed in polynomial-time of the size of the network. Precisely it can be done in $O(|V|^3)$.

Proof: Suppose that each edge $e \in E$ may collapse, this assumption leads to consider failure probabilities $p_e,\ \forall e \in E$ such that $0 < p_e \leq 1$. Thus $ln(p_e)$ is defined for all edge $e \in E$. Furthermore $ln(p_e) \leq 0,\ \forall e \in E$.
As a definition of the reliability criterion we have :

$$d_{st} \quad = Min\{1 - \prod_{e\in\delta(S)} p_e,\ s \in S,\ t \in V - S\}$$

Computing d_{st} can be reduced to compute $Max\{\prod_{e\in\delta(S)} p_e,\ \ s \in S,\ \ t \in V - S\}$. Since the ln function is strictly monotone increasing we may say that $Max\{\prod_{e\in\delta(S)} p_e,\ \ s \in S,\ \ t \in V - S\}$ is equivalent to $Max\{ln(\prod_{e\in\delta(S)} p_e),\ \ s \in S,\ \ t \in V - S\}$, in addition we have :

$$\begin{aligned} &Max\{ln(\prod_{e\in\delta(S)} p_e),\ s \in S,\ t \in V - S\} \\ &\quad = Max\{\sum_{e\in\delta(S)} ln(p_e),\ s \in S,\ t \in V - S\} \\ &\quad = Min\{\sum_{e\in\delta(S)} -ln(p_e),\ s \in S,\ t \in V - S\} \end{aligned}$$

Thus the problem is reduced to computing a minimum st-cut in a graph $G = (V, E)$, each edge having weight $-ln(p_e)$. We saw previously that $ln(p_e) \leq 0,\ \forall e \in E$. This allows us to say that the problem is reduced to compute a minimum weighted st-cut in a graph whose edges have non negative weights. Thus it can be done in $O(|V|^3)$ using Karzanov's max-flow algorithm, see [16]. ■

Now we look for a lower bound of the reliability. Clearly if we consider the subgraph induced by the set of disjoint paths linking s to t we will have a lower bound of the reliability of the network for the couple (s, t). Hence we will focus on the reliability when the set of st-paths is reduced to a set of edge disjoint paths.

Let C_{st} denote the set of the st-disjoint paths, P_{st} denotes the packing of st-cuts, π_i the probability that the path $i \in C_{st}$ is out of order and $d_i = 1 - \pi_i$ its availability, as previously p_e denotes the failure probability of the edge e. Then we may write :

$$\begin{gathered} D_{st} = 1 - \prod_{i\in C_{st}} \pi_i = 1 - \prod_{i\in C_{st}} (1 - d_i) = \\ 1 - \prod_{i\in C_{st}} (1 - \prod_{e\in i}(1 - p_e)) \end{gathered}$$

Now we suppose[1] that $\forall e \in E\ p_e = \epsilon$, this hypothesis is not so strong if we take into account that the failure probabilities of the network components are very small and in addition if the failure probability of a component is ϵ^q it may be replaced by q parallel components with failure probability ϵ. We obtain :

$$D_{st} = 1 - \prod_{i\in C_{st}} (1 - (1 - \epsilon)^{|i|})$$

where $|i|$ is the number of edges of the path $i \in C_{st}$. Let i_0 be the shortest path in the number of edges among those which

[1] this restriction is called ϵ-reliability in [7]

belong to C_{st}, then $(1-\epsilon)^{|i|} \le (1-\epsilon)^{|i_0|}$, $\forall i \in C_{st}$. Thereby we obtain :

$$1-(1-\epsilon)^{|i_0|} \le 1-(1-\epsilon)^{|i|},\ \forall i \in C_{st}$$

$$\prod_{i \in C_{st}} (1-(1-\epsilon)^{|i_0|}) = (1-(1-\epsilon)^{|i_0|})^{|C_{st}|}$$

$$\le \prod_{i \in C_{st}} (1-(1-\epsilon)^{|i|}).$$

which leads to : $D_{st} \le 1-(1-(1-\epsilon)^{|i_0|})^{|C_{st}|}$.

Consider the measure of the reliability defined by the product of the availabilities of the cuts of the packing P_{st}, then we have :

$$\prod_{\delta(S) \in P_{st}} (1-\prod_{e \in \delta(S)} p_e) = \prod_{\delta(S) \in P_{st}} (1-\epsilon^{|\delta(S)|}).$$

Notice that this definition of the reliability gives rise to an upper bound of the reliability when C_{st} is a set of disjoint paths. Indeed, as shown in the example below some edges are not covered by the packing, and so this is equivalent to consider these edges infallible. Since the number of disjoint paths equals the cardinality of a minimum cut and the cardinality of a maximum packing of disjoint cuts equals the number of edge of a shortest path we may write :

$$\prod_{\delta(S) \in P_{st}} (1-\epsilon^{|\delta(S)|}) \le \prod_{\delta(S) \in P_{st}} (1-\epsilon^{|C_{st}|}) = (1-\epsilon^{|C_{st}|})^{|i_0|}.$$

Let $\Delta(S)$ be the st cut which maximizes $\prod_{e \in \delta(S)} p_e$, then we obtain :

$$\prod_{\delta(S) \in P_{st}} (1-\epsilon^{|\delta(S)|}) \ge \prod_{\delta(S) \in P_{st}} (1-\epsilon^{|\Delta(S)|}) = (1-\epsilon^{|\Delta(S)|})^{|i_0|}.$$

Let $E(P_{st})$ be the set of edges covered by the packing P_{st} and $E(C_{st})$ the set of edges which appear in a path of C_{st}, then $\mathcal{D}_{st} = E(C_{st}) - E(P_{st})$ is the set of edges which belong to a path and are not covered by the packing of disjoint cuts P_{st}.

Claim 3.3: The quantity $\mathcal{A}(P_{st}) = (1-\epsilon^{|\Delta(S)|})^{|i_0|} \times (1-\epsilon)^{|\mathcal{D}_{st}|}$ is a lower bound of the reliability of the network

Proof: On one hand we consider the case where all the edges of $E(C_{st})$ are covered by the packing of disjoint cuts P_{st}, then since $|\mathcal{D}_{st}| = 0$ we have :

$$\mathcal{A}(P_{st}) = (1-\epsilon^{|\Delta(S)|})^{|i_0|} \times (1-\epsilon)^{|\mathcal{D}_{st}|} = (1-\epsilon^{|\Delta(S)|})^{|i_0|}.$$

Remark that under this assumption all the st-paths have a length equal to $|i_0|$. One of the edges of $\Delta(S)$ works with probability $1-\epsilon^{|\Delta(S)|}$. We need this to be true for all the $|i_0|$ cuts in the packing, therefore a lower bound appears of:

$$(1-\epsilon^{|\Delta(S)|})^{|i_0|}.$$

So we are done in this first case.

On the other hand consider that $\mathcal{D}_{st} \neq \emptyset$. Without loss of generality we may suppose that P_{st} is covering the $|i_0|$ first edges of any st-path. Now consider an extra vertex t' and carry out the transformation specified in figure 10.

Clearly the product $d_{st'} \times d_{t't}$ is a lower bound of the reliability of the network for the couple (s,t). Computing the availabilty between s and t' matches the case where all the edges of $E(C_{st})$ are covered by the packing of disjoint cuts, and $(1-\epsilon)^{|\mathcal{D}_{st}|}$ is a lower bound of the reliability of the network for the couple (t',t). Thus the result follows. ■

T.B. Brecht and C.J. Colbourn give a lower bound similar to $\mathcal{A}(P_{st})$ but quite different and not so easy to compute in [5]. This bound is the following : $1-\prod_{i=1}^{|\Delta(S)|} P_i$ where P_i is the probability that the i^{th} path fails.

Now we will show how to improve the lower bound. First of all, note that the network consisting of the edges of P_{st} has necessarily a lower reliability, so we restrict ourselves to the case where the set of edges is $E(C_{st})$. Consider the edges of $E(C_{st})$ which are not covered by the packing P_{st}. By definition each of these edges belongs to an st-path. So consider a path $p \in C_{st}$ and suppose that k of its edges are not covered by P_{st}, without loss of generality we may suppose that these edges are the k last ones of the path p. In order to cover all the edges of the path p by the packing P_{st} we shrink all these k edges and assign to those which comes before them the reliability corresponding to the reliability of the sub-path composed with these $k+1$ edges : $(1-\epsilon)^{k+1}$. Now we express the reliability over the packing of cuts, P'_{st}, obtained after we shrank the uncovered edges of any st path. Let $k_i,\ i \in \{1,\ldots,|C_{st}|\}$ be the number of uncovered edges of the path $p_i \in C_{st}$. We denote $\delta_0(S)$ the cut of P_{st} coming before the uncovered edges and $\delta'_0(S)$ the cut of P'_{st} that we obtain after we shrank the uncovered edges. Let $d(P'_{st})$ be the measure of the reliability of the network for the couple (s,t) involved by the product of the availabilities of the cuts of the packing P'_{st}, then we have :

$$d(P'_{st}) = \prod_{\delta(S) \in P_{st}} (1-\epsilon^{|\delta(S)|})(1-\prod_{e \in \delta'_0(S)} (1-(1-\epsilon)^{k_i+1})).$$

Then if we replace $\delta(S)$ by $\Delta(S)$ for all cuts we obtain :

$$d(P'_{st}) \ge \mathcal{A}(P'_{st}) =$$
$$(1-\epsilon^{|\Delta(S)|})^{|i_0|-1}(1-\prod_{e \in \delta'_0(S)} (1-(1-\epsilon)^{k_i+1})).$$

It follows from the claim 3.3 that $\mathcal{A}(P'_{st})$ is a lower bound of the reliability of the network for the couple (s,t).

Lemma 3.1: Both lower bounds $\mathcal{A}(P_{st})$ and $\mathcal{A}(P'_{st})$ can be computed in $O(|V|^3)$.

Proof: Straightforward if we remark that we just need to compute a minimum cut and a shortest path in order to reach the bounds. As seen previously the minimum cut can computed in $O(|V|^3)$ and the shortest path can be computed in $O(|V|^2)$ using the Dijkstra's algorithm. ■

B. Ratio between different measures of the reliability

In this section we express the ratios between the different measures of the reliability. On one hand we show that $\mathcal{A}(P'_{st})$ is a better lower bound than $\mathcal{A}(P_{st})$. On the other hand we express the ratio between the lower bound $\mathcal{A}(P'_{st})$ and the

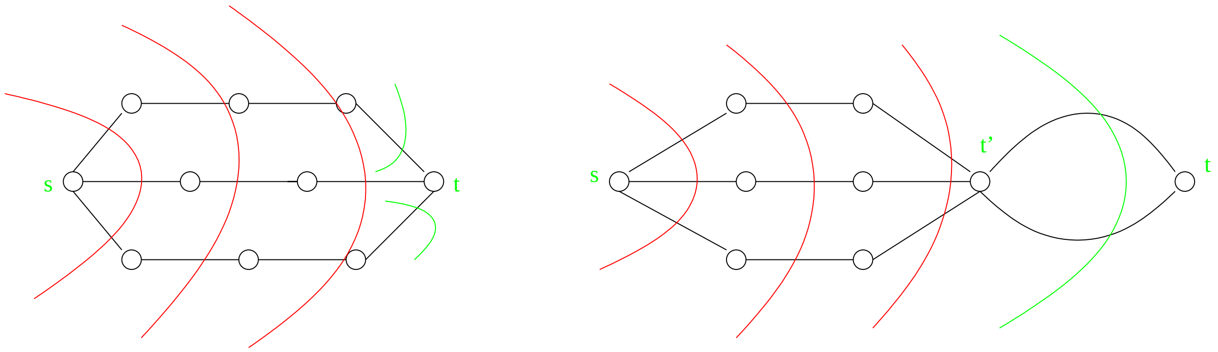


Fig. 10. Transformation used for the proof.

upper bound involved by the cut with the minimum reliability d_{st}.

Lemma 3.2: The lower bound $\mathcal{A}(P'_{st})$ is a better lower bound than $\mathcal{A}(P_{st})$.

Proof: Let $\mathcal{R}_0 = \frac{\mathcal{A}(P'_{st})}{\mathcal{A}(P_{st})}$, then $\mathcal{R}_0 = \frac{1-\prod_{e\in\delta'_0(S)}(1-(1-\epsilon)^{k_i+1})}{(1-\epsilon)^{|\mathcal{D}_{st}|}}$. Straightforward from the definitions of P_{st} and P'_{st} we have $\mathcal{R}_0 \geq 1$. ■

Now we will express the ratio between the lower bound and the upper bound : $\mathcal{R}_1 = \frac{\mathcal{A}(P'_{st})}{d_{st}}$. According to the definitions of the two bounds it arises that :

$$\mathcal{R}_1 = \frac{(1-\epsilon^{|\Delta(S)|})^{|i_0|}(1-\prod_{e\in\delta'_0(S)}(1-(1-\epsilon)^{k_i+1}))}{(1-\epsilon^{|C_{st}|})}$$

Since $|\Delta(S)| = |C_{st}|$ and $1 - \prod_{e\in\delta'_0(S)}(1-(1-\epsilon)^{k_i+1}) \geq (1-\epsilon)^{|\mathcal{D}_{st}|}$ it follows that :

$$\mathcal{R}_1 \geq (1-\epsilon^{|\Delta(S)|})^{|i_0|-1} \times (1-\epsilon)^{|\mathcal{D}_{st}|}.$$

This ratio confirms what can be seen intuitively : the lower bound becomes better as the set of uncovered edges becomes smaller, and the upper bound is better if the shortest path has just little few edges. If the latter conditions are fulfilled then the upper bound is closer to the lower bound.

IV. Further Work

We explained in the introduction that every network component e has a mean time before failure, $Mtbf_e$, and a mean time to repair, $Mttr_e$. Thus the failure probability of this component e is $p_e = \frac{Mttr_e}{Mtbf_e}$. Let γ_e be the savings done when the $Mttr_e$ is increased by one unit of time. The problem of interest, now, is to compute for every network component its best $Mttr_e$, or equivalently its failure probability p_e, in order to achieve the largest savings possible while reaching for every couple (s,t) the required level of reliability r_{st}. The total savings which can be done is given by the following objective function :

$$\sum_{e\in E}\gamma_e Mttr_e = \sum_{e\in E}\gamma_e Mtbf_e p_e.$$

Let $c = (\gamma_e Mtbf_e)_{e\in E}$ and $p \in (0,1]^{|E|}$ the failure probability vector of the network components, then we can formulate the problem previously described as follows :

Problem 4.1: Given $c \in \mathbb{R}_+^{|E|}$ find $p \in (0,1]^{|E|}$ that solves

$$\begin{cases} Max & c^t p \\ s.t. & d_{st}(p) \geq r_{st}\ \forall(s,t) \end{cases}$$

where $d_{st}(p)$ is the st-reliability of the random graph induced by p.

Under the assumption that $p_e = \epsilon\ \forall e \in E$ the problem collapses to maximizing ϵ under the constraints $d_{st}(\epsilon^{|E|}) \geq r_{st}\ \ \forall(s,t)$. Note that, since the parameter looked at is a scalar in this case, and the reliability is decreasing with it, a dichotomic approach on the algorithm of section II can work. Meanwhile, in a more general context, using the bounds seen in section III we define a relaxation of this latter problem. Since we want the required reliability to be reached, we want that $r_{st} \leq d_{st}$, where d_{st} is the actual measure of the st-reliability of the network for the couple (s,t). We previously saw that $d_{st} \geq \mathcal{A}(P_{st})$, thus if we take this last expression as the measure of the network reliability and if we impose that $r_{st} \leq \mathcal{A}(P_{st})$ we are sure that the required reliability will be reached. Thereby we obtain the following expression of the constraints of the problem :

$$\mathcal{A}(P_{st}) = (1-\epsilon^{|\Delta(S)|})^{|i_0|}(1-\epsilon)^{|\mathcal{D}_{st}|} \geq r_{st}\ \forall(s,t).$$

This is equivalent to : $|i_0|ln(1-\epsilon^{|\Delta(S)|}) + |\mathcal{D}_{st}|ln(1-\epsilon) \geq ln(r_{st})\ \ \forall(s,t)$. Now if we consider a first order expansion of $ln(1-\epsilon^{|\Delta(S)|})$ and $ln(1-\epsilon)$ and if we merge ϵ and its powers for small values of the exponent we obtain : $ln(1-\epsilon^{|\Delta(S)|}) \simeq -|\Delta(S)|\epsilon$ and $ln(1-\epsilon) \simeq -\epsilon$. This leads us to write the constraints of the problem in the following form : $-|i_0||\Delta(S)|\epsilon - |\mathcal{D}_{st}|\epsilon \geq ln(r_{st})\ \ \forall(s,t)$. Then the problem which consists in computing the values of the $Mttr_e$ leading to the largest savings and satisfying the reliability requirement can be formulated as the below linear program :

$$\begin{cases} Max & \epsilon \\ s.t. & -|i_0||\Delta(S)|\epsilon - |\mathcal{D}_{st}|\epsilon \geq ln(r_{st})\ \forall(s,t) \end{cases}$$

Once the $Mttr_e$ is computed for each equipment one determines the size of the sets of parts needed for servicing and the size of the maintenance teams. One other step in further development would be to compute lower and upper bounds as close as possible for large ranging failure probabilities of the equipments. This may be important if we consider the behavior of a network during ground operations since in this case some components may be more exposed than others.

REFERENCES

[1] M. O. Ball and J. S. Provan. Disjoint products and efficient computation of reliability. *Operations Research*, 36:703–715, 1988.

[2] E. T. Bell. Exponential numbers. *Amer. Math. Monthly*, 44:411–419, 1934.

[3] M. W. Bern, E. L. Lawler, and A. L. Wong. Linear time computation of optimal subgraphs of decomposable graphs. *J. Algorithms*, 8:216–235, 1987.

[4] H. L. Bodlaender, J. R. Gilbert, H. Hafsteinsson, and T. Kloks. Approximating treewidth, pathwidth, frontsize and shortest elimination tree. *J. Algorithms*, 18:238–255, 1995.

[5] T. B. Brecht and C. J. Colbourn. Lower bounds on two-terminal network reliability. *DAMATH: Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 21, 1988.

[6] M. Chari, T. Feo, and J. Provan. The delta-wye approximation procedure for two-terminal reliability. *Operations Research*, 44:745–755, 1996.

[7] C. J. Colbourn. *The combinatorics of network reliability*. Oxford University Press, 1987.

[8] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. The MIT Press, 1992.

[9] D. G. Corneil and J. M. Keil. A dynamic programming approach to the dominating set problem on k-trees. *SIAM J. Alg. Disc. Meth.*, 8:535–543, 1987.

[10] N. G. de Bruijn. Assymptotic methods in analysis. *New York:Dover*, pages 102–109, 1958.

[11] K. Dohmen. Inclusion-exclusion and network reliability. *The electronic journal of combinatorics*, 5, 1998.

[12] A. George. Nested dissection of a regular finite element mesh. *SIAM J. Numer. Anal.*, 10(2):345–363, 1973.

[13] K. Jansen and P. Scheffler. Generalized coloring for tree-like graphs. In *Proceedings 18th international workshop on graph-theoretic concepts in computer science (WG'92)*, LNCS vol 657, pages 50–59, Berlin, 1993.

[14] R. Kannan. Markov chains and polynomial time algorithms. *S. Goldwasser, ed. , Proceedings of the* 35^{th} *Annual Symposium on the Foundations of Computer Science, IEEE, IEEE Computer Society Press*, 1994.

[15] D. R. Karger. A Randomized Fully Polynomial Time Approximation scheme for the All-Terminal Network Reliability Problem. *SIAM Review*, 43(3):499–522, 2001.

[16] A. V. Karzanov. Determinig the maximal flow in a networkby the method of preflows. *Soviet Math. Dokl.*, 15:434–437, 1974.

[17] V. G. Krivoulets and V. P. Polesskii. Monotone Structures. The Best Possible Bounds of Their Reliability. *Information Processes*, Tom1, vol 2:188–198, 2001.

[18] J. Lagergren. *Algorithms and minimal forbidden minors for tree-decompsable graphs*. PhD thesis, Royal institute of technology, Stockholm, Sweden, 1991.

[19] L. Lovasz. *Combinatorial problems and exercises, 2nd ed.* North-Holland, Amsterdam, Netherland, 1993.

[20] E. Mata-Montero. Resilience of partial k-tree networks with edge and node failures. *Networks*, 21(321-344), 1991.

[21] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. Wiley Interscience, Series in Discrete Mathematics and Optimization, 1988.

[22] J. S. Provan and M. O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM J. Comput.*, 12(4):777–788, November 1983.

[23] J. S. Provan and M. O. Ball. Computing network reliability in time polynomial in the number of cuts. *Operations Research*, 32:516–526, 1984.

[24] N. Robertson and P. D. Seymour. Graph minors ii: algorithmic aspects of tree width. *J. Algorithms*, 7:309–322, 1986.

[25] A. Rosenthal. Computing the reliability of complex networks. *SIAM J. Appl. Math.*, 32(2):384–393, March 1977.

[26] A. Schrijver. *Theory of linear and integer programming*. Wiley Interscience, Series in Discrete Mathematics and Optimization, 1986.