Sensitivity computation in network reliability analysis

Gerardo Rubino

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P. 105
78153 Le Chesnay Cedex
France
Tél.: (1) 39 63 55 11

Rapports de Recherche

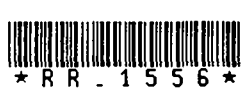
N° 1556

Programme 1
Architectures parallèles, Bases de données,
Réseaux et Systèmes distribués

SENSITIVITY COMPUTATION IN NETWORK RELIABILITY ANALYSIS

Gerardo RUBINO

Novembre 1991



IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTEMES ALEATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CEDEX FRANCE
Tél. : 99 84 71 00 - Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

Sensitivity Computation in Network Reliability Analysis

Gerardo Rubino, INRIA

Programme I
Publication Interne No. 612, Octobre 91, 38 pages.

Abstract

We analyze the computation of sensitivities in network reliability analysis using stochastic graphs models. These models are graphs whose components are weighted by probabilities (their reliabilities) and they are widely used, for instance, in the design of communication networks. The paper deals with the sensitivities of usual reliability network metrics, with respect to the reliabilities of the components. These metrics quantify some aspect of the behaviour of the whole system from the reliability point of view. The importance of sensitivities in this context is discussed and it is shown how to modify some of the existing procedures in network reliability theory to obtain the gradient of the reliability function as a byproduct, without a significant increasing in complexity. To illustrate the approach, we have chosen a basic reliability function in this area, the so called *2-terminal* (or *source-to-terminal*) one. The different methods considered include direct approaches based on boolean techniques, factoring algorithms and polynomial time reductions such as *series-parallel* and *polygon-to-chain* simplifications. The ideas presented here can be applied to other classes of reliability problems and/or methods.

Calcul de sensibilité dans l'analyse de la fiabilité des réseaux

Résumé

Nous étudions le problème du calcul des sensibilités dans l'analyse de la fiabilité d'un réseau modélisé par un graphe stochastique. Ces modèles sont des graphes dont les éléments sont pondérés par des probabilités (leurs fiabilités) et ils sont largement utilisés, par exemple, dans la conception de réseaux de communications. Le sujet d'étude est la sensibilité des functions donnant la fiabilité du système par rapport aux fiabilités des composants. Nous discutons de l'importance des sensibilités dans ce contexte et nous montrons comment modifier certaines méthodes existantes pour le calcul de la fiabilité de manière à obtenir aussi les sensibilités comme un sous-produit, sans une augmentation significative de la complexité. Pour illustrer l'approche, nous avons choisi une fonction de base dans le domaine, la fiabilité *source-terminal*. Les différentes méthodes considérées incluent les approches directes basées dans des techniques booléennes, les algorithmes de factorisation et les réductions polynomiales telles que les simplifications *série-parallèle* et *polygone-chaîne*. Les idées présentées ici peuvent être appliquées à d'autres classes de problèmes de fiabilité ainsi qu'à d'autres méthodes de calcul.

Contents

1 Introduction **7**
- 1.1 The model . . . 7
- 1.2 Motivation for sensitivity analysis . . . 7

2 Sensitivity of the reliability function **9**
- 2.1 Definitions and basic properties . . . 9
- 2.2 The case of "direct" methods . . . 11

3 Sensitivity analysis with series–parallel reductions **12**
- 3.1 Series-parallel reductions . . . 12
- 3.2 Series-parallel reductions with sensitivity analysis . . . 14
- 3.3 Example (see also Subsection 4.3) . . . 16

4 Sensitivity analysis with polygon–to–chain reductions **17**
- 4.1 Polygon–to–chain reductions . . . 17
- 4.2 Polygon–to–chain reductions with sensitivity analysis . . . 21
- 4.3 Example . . . 24

5 Sensitivity analysis in factoring algorithms **30**
- 5.1 Factoring algorithms . . . 30
- 5.2 Factoring algorithms with sensitivity analysis . . . 31
- 5.3 Example . . . 32

6 Conclusions **34**

Principal notation

- Pr( ) is the probability measure and E( ) denotes expectation.
- $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is the given graph:

  $\mathcal{V}$ is the set of nodes, $|\mathcal{V}| = N$,

  $\mathcal{E}$ is the set of edges, $|\mathcal{E}| = M$.
- r_e is the (elementary) reliability of line e.
- MP is the number of minpaths between nodes S and T,

  π_i is the ith minpath in a fixed (arbitrary) order,

  P_i is the event that "every line in the minpath π_i is working."
- MC is the number of mincuts separating nodes S and T,

  C'_h is the event that "every line in hth mincut separating S and T (in a fixed but arbitrary order) is not working."
- P_{ST} is the event "there exists at least one path between nodes S and T in which all the links are working."
- $R = \Pr(P_{ST})$ is the reliability of the network (the reliability of the connection between nodes S and T at the instant of interest.)
- $\sigma_e = \partial R / \partial r_e$ is the sensitivity of R with respect to r_e.
- $\nabla R = (\ldots \partial R / \partial r_e \ \ldots)$ is the vector of sensitivities (the gradient of R.)
- $\mathcal{G}_e^c$ is the graph obtained from $\mathcal{G}$ by contraction of edge e and R_e^c is the reliability of the connection between S and T in $\mathcal{G}_e^c$ (if $e = \{S, T\}$ then $R_e^c = 1$.) Also,

$$R_e^c = R|_{r_e=1} .$$

- $\mathcal{G}_e^d$ is the graph obtained from $\mathcal{G}$ by deletion of edge e and R_e^d is the reliability of the connection between S and T in $\mathcal{G}_e^c$. Also,

$$R_e^d = R\Big|_{r_e=0} .$$

1 Introduction

1.1 The model

The most widely used models to analyze the behaviour of complex systems subject to component failures are stochastic graphs, that is, graphs whose elements are wheighted by probabilities. The output usually obtained from this type of structure is the probability of events representing a desired behaviour of the system modelled. In this paper we consider the problem of the computation of reliability measures and we focus on a basic one, the so called 2-terminal (or source-to-terminal) reliability, in the following version: The input is a stochastic undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ is the set of edges (in some cases, we will explicitly consider *multigraphs* where there can be more than one edge between two vertices.) The graph may represent, for instance, a communication network. In this case, the vertices correspond to nodes sending and receiving information and the edges model communication lines. This application is used as a reference in the paper. For instance, the terms *vertex* and *node* are used as synonymous, and the same with *edge* or *line* or *link*. At a fixed time t, each line is in one of two states: either working (and working perfectly) or completely down (and it behaves as if it did not exist in the network.) A working line will also be called *operational*. In general, the reliability of a node in a communication network is much higher than the reliability of a line. According to this and to simplify the presentation, we assume that nodes are perfect, that is, not subject to failures. In the same way, the graph is pressumed to be undirected, that is, the lines of the communication network represented are bi-directional (messages can pass in both directions.) Of course, we assume that the graph is connected and without loops.

The state of line e at time t is a binary random variable X_e with value 1 if the line is working (and working perfectly) or 0, if the line does not work (and can be considered absent from the graph.) A basic assumption now is that (X_e) is a family of independent random variables. The 2-terminal problem is the following: Given the probability r_e of the event $\{X_e = 1\}$ for each line e of the network and two different nodes S and T, compute the probability R that, at time t, the two nodes can communicate with each other. In other words, if the first graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is fixed, the set $\mathcal{E}'$ of operational lines at time t defines a subgraph $\mathcal{G}' = (\mathcal{V}, \mathcal{E}')$ of the previous one. The number R is the probability that there exists at least one path between S and T in $\mathcal{G}'$ (we will also say that $\mathcal{G}'$ is S,T connected.) It will be called *reliability* of the connection between S and T, or in short, the *reliability of the network*. The number r_e is the *reliability* of line e. The set $\{r_e,\ e \in \mathcal{E}\}$ is the set of *elementary* reliabilities.

This problem (and several other related reliability problems) has received considerable attention from the research community. Numerous published papers have been devoted to its solution (see, for instance, [1] for references) mainly due to two reasons: the general use of these models in the communication network area and the fact that in the general case, the problem is in the #P-complete class [2], a family of *NP*-hard problems not known to be in *NP*. Recall that a #P-complete problem is equivalent to counting the number of solutions to a *NP*-complete one. This implies that a #P-complete problem is at least as hard as a *NP*-complete one. This last fact justifies the continuous effort to find more efficient solution methods. Even in very restricted (and important) families of graphs, the problem remains in the #P-complete class. For instance, it is shown in [3] that it remains #P-complete in the family of S,T planar graphs (that is, planar graphs where nodes S and T are in the frontier) having vertex degree at most equal to three.

1.2 Motivation for sensitivity analysis

In this paper, we adress a complementary problem. If we consider the reliability R of the network as a function of the individual reliabilities r_e of the lines, it is of interest to know the sensitivity σ_e of R

with respect to each independent variable r_e which are defined as

$$\sigma_e \stackrel{\text{def}}{=} \frac{\partial R}{\partial r_e}.$$

Several reasons make the computation of this set of derivatives (that is, the vector ∇R) interesting. First, it is involved in optimizations when, for instance, each elementary reliability is a function of some local cost measure c_e: $r_e = r_e(c_e)$. Assume that a fixed capital C is available to improve an existing communication network by investing into the lines. If the criteria is to increase the reliability of the network, we have to maximize $R = R(c_e,\ e \in \mathcal{E})$ under the constraint $\sum_e c_e = C$. The rate $\rho_e = \partial R/\partial c_e$ is the instantaneous variation of R with respect to the variation of the local cost of line e. Since only r_e depends on c_e, we have

$$\rho_e = \sigma_e \frac{dr_e}{dc_e}.$$

Once the vector ∇R is computed, the rates ρ_e can be obtained using only the local derivates dr_e/dc_e, i.e., without explicitly constructing and computing the function $R = R(c_e,\ e \in \mathcal{E})$. This is a rather classical application of sensitivity analysis. It can be mentionned that many authors propose the computation of a symbolic expression of the function $R(r_e,\ e \in \mathcal{E})$ (that is, a formal expression) in order to perform optimizations (see for instance [4].) Of course, symbolic expressions are not only used for this purpose: When the topology is fixed and the user wants to evaluate its model for different values of the reliabilities of its components, it is very useful to obtain such a symbolic expression first and to perform just variable substitutions by different numerical values. However, the price to pay is a considerably higher computational cost. It will be noted that in the subject discussed here, there is place to perform formal computations also, if the size of the model allows that. Since our goal is to show how to obtain ∇R as a byproduct of known algorithms for the computation of R itself, our remarks can be applied if these computations are performed symbolically.

The computation of ∇R may also be useful if there is a temporal dimension in the problem. Assume that each individual reliability is considered at different instants t, that is, $r_e = r_e(t)$. The instantaneous variation of R with time can be obtained without considering explicitly $R = R(t)$. Applying elementary differentiation rules, we have

$$\frac{dR}{dt} = \sum_e \sigma_e \frac{dr_e}{dt}.$$

If we know how to calculate ∇R efficiently, this approach is better than the computation of dR/dt directly from the function $R = R(t)$.

A more specific and relevant application of the computation of sensitivities in our context is the following. The reliability measure R quantifies the ability of the whole network to transmit messages between the two selected nodes at a fixed point in time. This number depends, of course, on the precision of input data which consists also of measured (or calculated) reliabilities. The possible (and probable) errors in the input induce an error in the output R. Although this numerical output R may not be very precise (or in other words, may not be very "reliable" itself), the order induced on the set of edges by sensitivities is a much more robust information about the properties of the network. It allows the identification of most critical parts of the structure, taking into account both the topology and the stochastic behaviour of the system [5]. It is clear that this order is signifiantly less dependent on the errors arising from each individual reliability r_e. As an example, consider the network of Figure 1. We have $R = r_a r_b + r_c - r_a r_b r_c$, $\sigma_a = r_b(1-r_c)$, $\sigma_b = r_a(1-r_c)$ and $\sigma_c = 1 - r_a r_b$. If $r_a = r_b = r_c = 0.9$ then $\sigma_a = \sigma_b = 0.09$ and $\sigma_c = 0.19$, that is, link c is the critical one. It is clear that c will remain the critical link if the "real" values of the elementary reliabilities are "near" 0.9. If we let $r_a = r_b = 0.9$ and we decrease the value of r_c, while $r_c > 71/90 \approx 0.7889$ we will have $\sigma_c > \sigma_a = \sigma_b$. For instance, if $r_c = 0.7$, σ_a (and σ_b) changes to 0.27 and c is the least critical link in

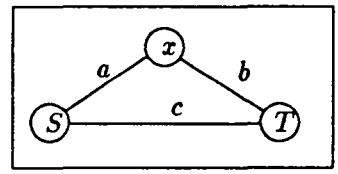


Figure 1: A triangle

the network. To illustrate the robustness, the reader can verify for instance that if $r_a < 0.5$, then for any $r_b < r_a$ and for any r_c we have $\sigma_a < \sigma_b < \sigma_c$.

In [5], the analysis of ∇R is considered in the general framework of coherent binary systems. In the particular case of reliability network theory, the importance of sensitivity analysis is observed in [6] and the sensitivities are computed by means of an extension of the well known Abraham algorithm [7]. The rest of this paper is devoted to some of the most successful techniques applied to the computation of R and the extension necessary to calculate ∇R without a drastic change in complexity. Some of these techniques have been developped in (or extended to) more general frameworks (directed graphs, arcs and nodes subject to failures, etc.) but these generalizations are not essential here. The observations made in this paper can be considered in those contexts. Also, these ideas can be easily adapted to other reliability measures such as the *all-terminal* reliability or the general $\mathcal{K}$-terminal reliability: given a subset $\mathcal{K}$ of vertices, the reliability (in the undirected context) is the probability that there exists at least one path between every pair of nodes in $\mathcal{K}$. As particular cases we have the 2-terminal problem ($\mathcal{K} = \{S, T\}$) and the all-terminal one ($\mathcal{K} = \mathcal{V}$.)

The paper is organized as follows. The next section introduces basic definitions and properties, as well as the case of *direct* techniques to compute R since in these methods the extensions to calculate ∇R as a byproduct are obtained in a straightforward manner. Sections 3 and 4 deal with two reduction techniques, namely series–parallel and polygon–to–chain reductions, which are important tools to diminish the computation time in reliability algorithms. The idea is to develop low cost (actually polynomial cost) simplifications that can be applied as preprocessors, in order to reduce the size of the original problem. These methods are not necessarily effective, that is, the graphs can be irreducible with respect to them, but the overhead is negligible compared to the total time required to solve the initial models. In many cases, simplifications can be performed and the gain in the total solution time is significantly higher than the associated cost to apply them. In fact, the polygon–to–chain reduction is an extension of the series–parallel one. Both simplification procedures are presented here because (i) the first one is probably most widely used and (ii) the technical modifications required to deal with sensitivities are slightly more complicated in the second. Section 5 considers *factoring* or *pivoting* algorithms, an important family of methods which perform in general very fast. The last section is devoted to conclusions and possible research directions.

2 Sensitivity of the reliability function

2.1 Definitions and basic properties

Let us denote by N the number of nodes and by M the number of edges or lines. Let P_{ST} be the event that "there exists at least one path between S and T in which all the links are working." The reliability of the network is $R = \Pr(P_{ST})$ where Pr is the probability measure. The sensitivity vector is

$$\nabla R \stackrel{\text{def}}{=} \left(\dots \frac{\partial R}{\partial r_e} \dots \right) = (\dots \sigma_e \dots).$$

As this model is a particular case of a *binary monotone system*, it is well known that for every line e we have $\sigma_e \geq 0$ (see the basic reference [5].) This means that we cannot improve the reliability of the

whole system by decreasing the (elementary) reliability of one of its components.

The *contraction* of line $e = \{x, y\}$ of $\mathcal{G}$ consists of eliminating the line e and "merging" the two extremities x and y: the resulting graph has $N - 1$ nodes and $M - 1$ edges and it is denoted by $\mathcal{G}_e^c$. Just deleting line e gives $\mathcal{G}_e^d$ with the same node set and $M - 1$ edges. Let us define

$$R_e^c \stackrel{\text{def}}{=} R|_{r_e=1} \quad \text{and} \quad R_e^d \stackrel{\text{def}}{=} R|_{r_e=0}. \tag{1}$$

The number R_e^c (respectively R_e^d) is the reliability of the connection between S and T in $\mathcal{G}_e^c$ (respectively in $\mathcal{G}_e^d$.) If line e is $\{S, T\}$, then R_e^c is defined as 1. Between these quantities the following order relation holds:

$$\text{for all line } e, \quad 0 \leq R_e^d \leq R \leq R_e^c \leq 1. \tag{2}$$

This is simply a consequence of the fact that we deal with a *monotone* system. Conditionning with respect to the state of line e we obtain the following useful relation:

$$\text{for all line } e, \quad R = r_e R_e^c + (1 - r_e) R_e^d. \tag{3}$$

Relation (3) is usually called the "factoring theorem" and it is the basis of a family of algorithms considered in Section 5. Deriving it with respect to r_e gives

$$\text{for all line } e, \quad \sigma_e = R_e^c - R_e^d. \tag{4}$$

From relations (2) and (4) we have the following trivial bounds on the sensitivity σ_e:

$$\text{for all line } e, \quad \sigma_e \leq R_e^c \quad \text{and} \quad \sigma_e \leq 1 - R_e^d.$$

Now, the formula (4) together with (3) give the two following expressions of σ_e:

$$\text{for all line } e \text{ such that } 0 < r_e < 1, \quad \sigma_e = \frac{R_e^c - R}{1 - r_e} \tag{5}$$

$$= \frac{R - R_e^d}{r_e}. \tag{6}$$

Relations (5) or (6) allow us to compute ∇R by means of $M + 1$ reliability computations. The main objective of this paper is to show how to modify some of the most widely used approaches in network reliability theory in order to avoid this significant increment in complexity.

For completeness, let us mention another measure of the importance of a component on the reliability of the whole system, proposed in [8]. If we denote by Y the random variable with value in $\{0, 1\}$, defined by $Y = 1$ iff nodes S and T are connected in $\mathcal{G}'$ (and thus, $R = \Pr(Y = 1)$), the measure proposed in [8] is

$$\xi_e \stackrel{\text{def}}{=} \Pr(X_e = 0 \mid Y = 0),$$

that is, ξ_e is the probability that line e has failed, given that the network itself has failed. The author proposes this measure as a tool for diagnostic analysis, when the network is observed to fail. We can write

$$\begin{aligned} \xi_e &= \Pr(X_e = 0 \mid Y = 0) \\ &= \frac{\Pr(Y = 0 \mid X_e = 0) \Pr(X_e = 0)}{\Pr(Y = 0)} \\ &= \frac{(1 - R_e^d)(1 - r_e)}{1 - R} \end{aligned}$$

and using (6),

$$\xi_e = \frac{(1 - R + r_e\sigma_e)(1 - r_e)}{1 - R},$$

so that this measure can be directly derived from sensitivities.

Last, remark that, as in physics, it can be useful to consider the *elasticity* h_e of R with respect to r_e, instead of the sensitivity. The elasticity is defined here by

$$h_e \stackrel{\text{def}}{=} \frac{r_e\sigma_e}{R}. \tag{7}$$

From (6), we have that

$$\sigma_e \leq \frac{R}{r_e}, \tag{8}$$

so that we always have $h_e \leq 1$ and it is immediate to verify that $h_e = 1$ if and only if line e is in series with the rest of the network. This quantity reflects the same kind of property as σ_e in a normalized way.

2.2 The case of "direct" methods

We call "direct methods" the techniques which do not create (explicitly or implicitly) any intermediate graph in the process of computing the reliabilty of the given network. To be more specific, let us denote by MP the number of minpaths between nodes S and T and $\{\pi_1, \ldots, \pi_{MP}\}$ the set of these minpaths in any given order. Let P_i denote the event "every line in the minpath π_i is working." From the independence assumption, we have

$$\Pr(P_i) = \prod_{e \in \pi_i} r_e. \tag{9}$$

Now, the reliability R is the probability of the union of all P_i's, that is

$$R = \Pr(P_1 \cup \ldots \cup P_{MP}) \tag{10}$$

but unfortunately, the events $P_1, P_2, \ldots$ are not mutually disjoint. Poincare's formula allows us to rewrite (10) as a more explicit function of the elementary reliabilities r_e's:

$$R = \sum_{h=1}^{MP} (-1)^{h-1} \sum_{i_1 < i_2 < \cdots < i_h} \Pr(P_{i_1}.P_{i_2}.\cdots.P_{i_h}) \tag{11}$$

$$= \sum_{h=1}^{MP} (-1)^{h-1} \sum_{i_1 < i_2 < \cdots < i_h} \prod_{e \in \pi_{i_1} \cup \pi_{i_2} \cup \cdots \cup \pi_{i_h}} r_e. \tag{12}$$

In fact, the direct application of this formula performs worse than the pure brute force approach which considers the whole state space of size 2^M. However, there are several papers taking (11) as a starting point. For instance, it can be seen that, in general, (11) leads to many terms being cancelled out. In [9] and [10], the authors give algorithms that generate only the resulting terms.

Another approach consists in applying the following transformation:

$$P_1 \cup P_2 \cup \cdots \cup P_{MP} = P_1 \cup \overline{P}_1.P_2 \cup \overline{P}_1.\overline{P}_2.P_3 \cup \cdots \cup \overline{P}_1.\overline{P}_2.\ldots.\overline{P}_{MP-1}.P_{MP} \tag{13}$$

in which we denote by $\overline{P}_i$ the complement of P_i. This relation involves MP terms instead of the 2^{MP} of (11). However, they are much more complex and difficult to obtain. A well known example is the Abraham algorithm [7]. See also [11] and [12]. The generation of the MP minpaths between S and T is an expensive problem itself (it belongs to the #P-complete class) and the computation of the

probability of each term in (13) is also expensive. The complexity of the 2-terminal reliability in the number of minpaths is still an open issue in the context of undirected graphs. In [13] it is shown that, for directed graphs, an algorithm to compute R having its time complexity polynomial in MP exists if and only if $P = NP$.

It is clear that a dual approach using mincuts between S and T is also possible. If C'_h denotes the event "all the links in the hth mincut separating nodes S and T are not working" then we have

$$R = 1 - \Pr\left(C'_1 \cup \ldots \cup C'_{MC}\right)$$

where MC is the number of mincuts separating S and T. It can be observed that there is an interesting asymetry between the minpath and the mincut approaches: In [13], the computation of R is shown to be polynomial in the number of mincuts.

In these methods, the reliability of the network is computed as a sum of polynomials in the variables r_e, $e \in \mathcal{E}$ and, usually, each polynomial has the property of containing each individual variable r_e as a factor at most once. The computation of $\partial R/\partial r_e$ can then be performed in a straightforward manner by deriving each term in the expression of R as soon as it is generated. The next section considers a first example of extension of existing methods to deal with sensitivities as well.

3 Sensitivity analysis with series–parallel reductions

3.1 Series-parallel reductions

A simple and useful technique to diminish the computational time of reliablity calculations is to perform series–parallel reductions before using a general algorithm to solve the network. These reductions can be performed in polynomial time which implies that the overhead is negligible with respect to the cost of reliability algorithms. On the other hand, the gain can be considerable. Before looking in detail at these reductions and the computation of the sensitivity vector, let us set some definitions.

Two links are in *series* if they are adjacent and they have only one common vertex with degree two. A *series reduction* consists of replacing two links in series by a single edge between the not shared vertices. This new edge receives as its elementary reliability the product of the reliabilities of the previous ones. A S,T series reduction is a series reduction in which the vertix shared by the two links in series does not belong to $\{S,T\}$. Two links e_1, e_2 are in *parallel* (that is, considering multigraphs) if they have the same vertices and a *parallel reduction* consists of replacing them by a single edge between the same vertices. The reliability of the new link is $r_{e_1} + r_{e_2} - r_{e_1} r_{e_2}$. These definitions extend immediately to more than 2 edges.

If in a (multi)graph $\mathcal{G}$ there are no parallel edges and no edges in series, exept perhaps two edges $\{x,y\}$ and $\{y,z\}$ with $y \in \{S,T\}$, it is S,T *series-parallel irreducible*; if $\mathcal{G}$ has only two vertices (S and T) and an edge between them, it is S,T *series-parallel irreducible* by definition. Otherwise, $\mathcal{G}$ is S,T *series-parallel reducible*. If we perform recursively parallel reductions and S,T series reductions on a S,T *series-parallel reducible* (multi)graph $\mathcal{G}$ until no more reductions are possible, the result is an unique graph $\mathcal{G}^{\text{IRR}}$ regardeless of the chosen successive pairs of edges in series or in parallel. We say that $\mathcal{G}^{\text{IRR}}$ is the result of a *complete* S,T *series-parallel* reduction. If $\mathcal{G}^{\text{IRR}}$ has only the two vertices S,T we say that the original (multi)graph $\mathcal{G}$ is S,T *series-parallel*. It is clear that the reliability R is invariant with respect to these transformations (and can be computed on the reduced graph $\mathcal{G}^{\text{IRR}}$.) See for instance [14] for general remarks on these conservation properties. If $\mathcal{G}$ is a S,T *series-parallel* (multi)graph then R is the elementary probability of the only edge of the reduced graph $\mathcal{G}^{\text{IRR}}$.

From an algorithmic point of view, assume that we have two functions called *lookForParallel* and *lookForSeries* taking as input data a given (multi)graph $\mathcal{G}$ with its elementary reliabilities and the distinguished nodes S and T. Assume that $\mathcal{G}$, S, T and the elementary reliabilities are all global variables. The first function returns a set $\{e_1, e_2\}$ where e_1, e_2 are two edges in parallel; if such a pair

of nodes does not exist, the empty set is returned. In any case, we write '$\{e_1, e_2\}$:= *lookForParallel*' to denote the use of this function. Also, a global boolean *foundParallel* is set to **true** in the first case and to **false** in the second. The function *lookForSeries* returns $\{e_1, e_2\}$ if (i) e_1, e_2 are two edges in series and (ii) the common vertex does not belong to $\{S, T\}$; otherwise, it returns the empty set. The global boolean *foundSeries* is set in the analogous way. Let us denote by $r[e]$ the variable containing the (elementary) reliability of edge e. Then, an algorithm to perform S, T series-parallel reductions on $\mathcal{G}$ is the following.

```
algorithm : S,T series parallel reduction
    parallelReductions
    loop
        seriesReductions
        if noReductions then break
        parallelReductions
        if noReductions then break
    end loop
end algorithm
```

The first call to *parallelReductions* is necessary in the case of a possible multigraph in the input. Observe that even if there is no parallel edges, we have to look for series before leaving the procedure. In the following procedures, the functions *replaceSeries* and *replaceParallel* perform the series and parallel reductions respectively, returning the new edge that replaces the previous ones.

```
procedure parallelReductions /* elimination of all parallel links */
    noReductions := true
    loop
        {e1,e2} := lookForParallel
        if foundParallel then
            noReductions := false
            e := replaceParallel(e1,e2) /* elementary parallel reduction */
            r[e] := r[e1] + r[e2] - r[e1] * r[e2]
        else
            return
        end if
    end loop
end procedure
```

```
procedure seriesReductions /* elimination of all S,T series */
    noReductions := true
    loop
        {e1,e2} := lookForSeries
        if foundSeries then
            noReductions := false
            e := replaceSeries(e1,e2) /* elementary series reduction */
            r[e] := r[e1] * r[e2]
        else
            return
        end if
    end loop
end procedure
```

It is clear that to implement this reduction method, we can improve the presented algorithm in several ways. For instance, note that after a series reduction (that is, after a call to *seriesReductions*) the only pairs of nodes x, y where it is possible to find edges in parallel are extremities of edges that have been generated in the series replacements. A similar remark holds for the possible nodes candidates to a series reduction.

3.2 Series-parallel reductions with sensitivity analysis

With respect to sensitivities, observe first that in the case of a three vertices series graph with node set $\{S, T, x\}$ and edge set $\{e_1 = \{S, x\},\ e_2 = \{x, T\}\}$, the sensitivities are

$$\begin{aligned} \sigma_{e_1} &= \frac{\partial R}{\partial r_{e_1}} = r_{e_2}, \\ \sigma_{e_2} &= \frac{\partial R}{\partial r_{e_2}} = r_{e_1}. \end{aligned}$$

In the same way, if $\mathcal{G}$ is a multigraph with two vertices S and T, and two edges between them, we have

$$\begin{aligned} \sigma_{e_1} &= \frac{\partial R}{\partial r_{e_1}} = 1 - r_{e_2}, \\ \sigma_{e_2} &= \frac{\partial R}{\partial r_{e_2}} = 1 - r_{e_1}. \end{aligned}$$

Suppose that we have a procedure to compute R and the sensitivity vector ∇R for any graph and that we perform a S, T series–parallel reduction as a preprocessing to attempt to diminish the computational time. Let $\mathcal{G}_0$ be the original graph and $\mathcal{G}_i$ be the current graph obtained after i elementary (single step) reductions from $\mathcal{G}_0$. With each link e of $\mathcal{G}_i$, $i \geq 0$, we associate a subset of links *containedBy*[e] of $\mathcal{G}_0$ and with each link f in *containedBy*[e] we associate a real number $w_{e,f}$, called the *weight* of f. We call these sets and the corresponding weights the "associated information" and it is computed as the following modification of the previous algorithm shows (in the algorithms, the weight $w_{e,f}$ is denoted by *weight*[e,f].)

```
algorithm : S,T series parallel reduction with sensitivity analysis
    for all e do    /* initializations */
        containedBy[e] := {e}
        weight[e,e] := 1.0
    end for
    parallelReductions
    loop
        seriesReductions
        if noReductions then break
        parallelReductions
        if noReductions then break
    end loop
end algorithm ____________________________________________

procedure parallelReductions /* elimination of all parallel links */
    noReductions := true
    loop
        {e1, e2} := lookForParallel
```

```
        if foundParallel then
            noReductions := false
            e := replaceParallel(e1, e2) /* elementary parallel reduction */
            r[e] := r[e1] + r[e2] − r[e1] * r[e2]
            containedBy[e] := containedBy[e1] ∪ containedBy[e2]
            for all f in containedBy[e1] do
                weight[e, f] := weight[e1, f] * (1 − r[e2])
            end for
            for all f in containedBy[e2] do
                weight[e, f] := weight[e2, f] * (1 − r[e1])
            end for
        else
            return
        end if
    end loop
end procedure ____________________________________________

procedure seriesReductions /* elimination of all S,T series */
    noReductions := true
    loop
        {e1, e2} := lookForSeries
        if foundSeries then
            noReductions := false
            e := replaceSeries(e1, e2) /* elementary series reduction */
            r[e] := r[e1] * r[e2]
            containedBy[e] := containedBy[e1] ∪ containedBy[e2]
            for all f in containedBy[e1] do
                weight[e, f] := weight[e1, f] * r[e2]
            end for
            for all f in containedBy[e2] do
                weight[e, f] := weight[e2, f] * r[e1]
            end for
        else
            return
        end if
    end loop
end procedure ____________________________________________
```

Let us first state some properties of the sets *containedBy*[.] that are quite obvious from the text of the algorithms. Assume that a link e of $\mathcal{G}_0$ has not been involved in any of the i reductions leading to $\mathcal{G}_i$. Then, $e \in \mathcal{G}_i$, *containedBy*$[e] = \{e\}$ and $w_{e,e} = 1.0$. If this is not the case, there is an edge e' in $\mathcal{G}_i$ (and only one) such that e belongs to the set *containedBy*$[e']$; if we denote by x and y the extremities of e', then the set of edges in *containedBy*$[e']$ define a partial x, y series-parallel graph of $\mathcal{G}$. The probability $r_{e'}$ is the reliability of the connexion between x and y in this partial graph and it is a function of the reliabilities of the links in *containedBy*$[e']$ only. In all cases, the following relation holds:

$$\text{for all } e' \in \mathcal{G}_i \text{ and for all } e \in \textit{containedBy}[e'], \qquad w_{e',e} = \frac{\partial r_{e'}}{\partial r_e}.$$

This property can be easily verified by induction. It obviously holds for $i = 0$. Assume it holds, for instance, after a parallel reduction which leads to the graph $\mathcal{G}_i$ and suppose that we replace two edges

e_1 and e_2 in S,T series in $\mathcal{G}_i$ by a new edge e'. We must only consider the edges $f \in \mathcal{G}_0$ belonging to *containedBy*$[e_1]$ or *containedBy*$[e_2]$. Assume that $f \in$ *containedBy*$[e_1]$. We have

$$\begin{aligned} \frac{\partial r_{e'}}{\partial r_f} &= \frac{\partial r_{e'}}{\partial r_{e_1}} \frac{\partial r_{e_1}}{\partial r_f} \\ &= r_{e_2} w_{e_1,f} \end{aligned}$$

which justifies the statement $weight[e,f] := weight[e_1,f] * r[e_2]$. In the same way we can check the validity of the corresponding statement after a parallel reduction.

Denote, as before, by $\mathcal{G}^{\text{irr}}$ the graph obtained from $\mathcal{G}$ after a complete S,T series–parallel reduction (that is, let $\mathcal{G}^{\text{irr}}$ be the output of the previous main algorithm.) If $\mathcal{G}^{\text{irr}}$ is reduced to the nodes S,T and a single link e' between them, the elementary reliability $r_{e'}$ of e' is equal to R, the set *containedBy*$[e']$ is the whole set $\mathcal{E}$ of edges of $\mathcal{G}$ and for any $e \in \mathcal{E}$ the number $w_{e',e}$ is the sensitivity of R with respect to the reliability of link e. In the case of $\mathcal{G}^{\text{irr}}$ containing more than 2 nodes (and then more than three links), let us denote by $\mathcal{E}'$ the set of edges in $\mathcal{G}^{\text{irr}}$. For any $e' \in \mathcal{E}'$, let us denote by $\sigma_{e'}$ the sensitivity of the reliability between S and T in $\mathcal{G}^{\text{irr}}$ with respect to the reliability of e'. To obtain the sensitivities of R with respect to the reliabilities of the links of $\mathcal{G}$, we perform the following loop.

```
for all e' in E' do
    for all e in containedBy[e'] do
        σ[e] := weight[e', e] * σ[e']
    end for
end for
```

3.3 Example (see also Subsection 4.3)

Let us consider the graph in Figure 2 where all the elementary reliabilities are equal to 0.5.

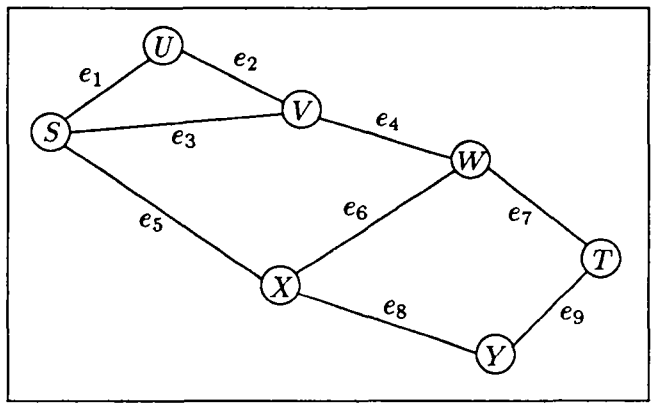


Figure 2: A S,T series–parallel reducible graph

It is not irreducible with respect to S,T series–parallel structures. The reduction procedure replaces first the two edges e_1 and e_2 by a new one, then this new edge is in parallel with e_3, etc. The resulting graph is given in Figure 3.

The reliabilities of edges e_{10} and e_{11} are

$$r_{e_{10}} = 0.3125,\ r_{e_{11}} = 0.25$$

and the associated information needed to perform sensitivity computations is given in the following Table.

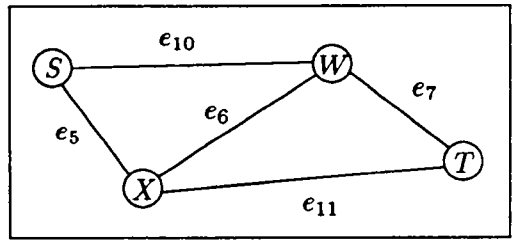


Figure 3: The S,T series–parallel reduction of the graph in Figure 2

$e' \in \mathcal{E}'$	e_5	e_6	e_7	e_{10}				e_{11}	
$containedBy[e']$	e_5	e_6	e_7	e_1	e_2	e_3	e_4	e_8	e_9
$w_{e',.}$	1	1	1	0.125	0.125	0.375	0.625	0.5	0.5

Table 1: The associated information

For instance, the sensitivity of R with respect to $r_{e_{10}}$ (so, in the reduced graph) is equal to 0.375 and we have:

$$\begin{aligned} \sigma_{e_1} &= w_{e_{10},e_1} \frac{\partial R}{\partial r_{e_{10}}} \\ &= 0.125 * 0.375 \\ &= 0.046875. \end{aligned}$$

4 Sensitivity analysis with polygon–to–chain reductions

4.1 Polygon–to–chain reductions

The *polygon-to-chain* reduction can be seen as a generalization of the series–parallel one. It has been introduced in [15] for the K–terminal reliability problem. As stated before, we limit ourselves to the case of $K = \{S,T\}$ to clarify the presentation. In this case, the procedure can be easily simplified and improved but we will present it exactly as in the general case since we just want to illustrate the extensions that are needed in order to compute the sensitivities. The idea is basically the same if the general case (any subset K of V) is considered.

Let us first define the meaning of the words *polygon* and *chain* in this context. A *polygon* is a cycle with at least three nodes and such that all its nodes except two of them have degree two. A *chain* is a sequence of adjacent and distinct nodes containing at least two elements and such that all the intermediate nodes have degree two.

The first result leading to this method states that if $\mathcal{G}$ is a S,T series–parallel irreducible network, then if it has a polygon, necessarily the polygon is of one of the four types illustrated in Figure 4 where $\{U,V\} \equiv \{S,T\}$ (in the general case of K–terminal reliability, there are seven possible cases [15].) This is evident given that the graph is S,T series–parallel irreducible. The next step states that in any of the four previous cases, the polygon can be changed by a chain conserving the same node set according to the scheme given in Figure 5, with the following property: It is possible to assign values to the elementary reliabilities of the links in the introduced chain depending only on the reliabilities of the edges in the replaced polygon such that between R and the reliability of the connexion between S and T in the transformed graph, which is denoted here by R', the following relation holds:

$$R = \Omega R' \tag{14}$$

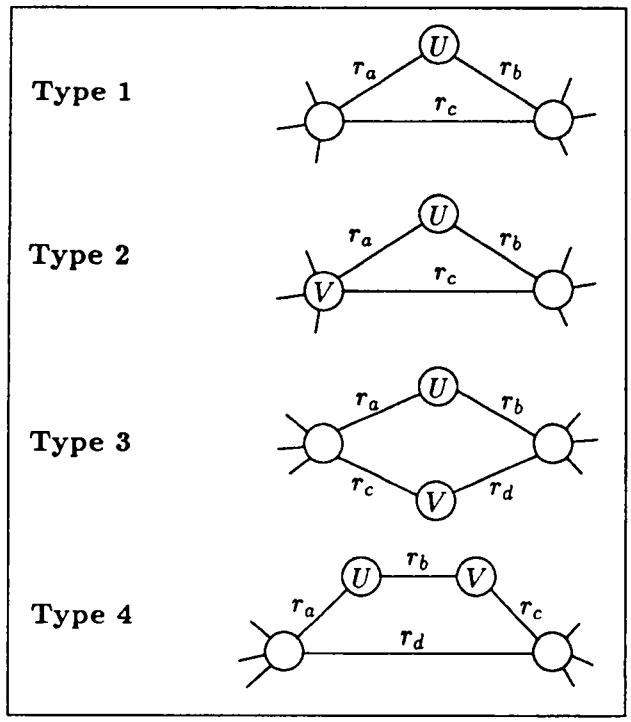


Figure 4: The four possible polygons in a S,T series–parallel irreducible graph.

where the factor Ω depends also on the reliabilities of the links of the polygon only. Formulae giving

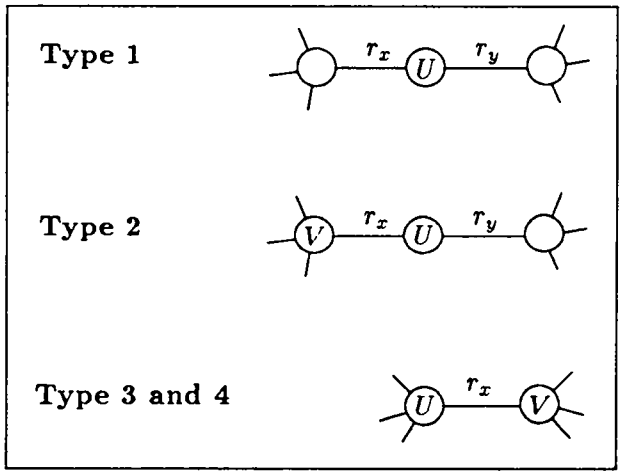


Figure 5: The chains corresponding to each polygon in Figure 4.

these functions are the following:

- **Type 1**
  Let $u = (1 - r_a)r_b(1 - r_c)$,
  $v = r_a(1 - r_b)(1 - r_c)$,
  $\Delta = r_a r_b r_c \left(1 + \frac{1 - r_a}{r_a} + \frac{1 - r_b}{r_b} + \frac{1 - r_c}{r_c}\right)$;
  then $r_x = \frac{\Delta}{u + \Delta}$,
  $r_y = \frac{\Delta}{v + \Delta}$,

$$\Omega = \frac{(u+\Delta)(v+\Delta)}{\Delta}.$$

- **Type 2**
  Exactly as in Type 1 polygons.

- **Type 3**
  $r_x = \frac{r_a r_c + r_b r_d - r_a r_b r_c r_d}{\Omega}$
  and $\Omega = (r_a + r_b - r_a r_b)(r_c + r_d - r_c r_d)$.

- **Type 4**
  $r_x = \frac{r_b + r_a(1 - r_b) r_c r_d}{\Omega}$
  and $\Omega = r_b + r_a(1 - r_b) r_c$.

There is a minor precision to be made in the case of the configuration shown in Figure 6 where, according to Figure 4, there are two polygons of the type 1 and a polygon of the type 3. The case of

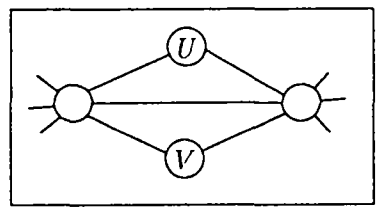


Figure 6: A case of two polygons of the type 1 and a polygon of the type 3

this configuration will be considered below in the presentation of the procedure *polygonToChainStep*. The main algorithm can be written as follows.

```
algorithm : S,T polygon to chain reduction
    seriesParallelReductions /* perform a complete S,T series-parallel reduction */
    if bond then stop /* the original graph was S,T series-parallel */
    Ω := 1.0 /* initialize the global variable Ω */
    loop
        polygonToChainStep /* replace all the polygons */
        if bond then break /* the previous graph was a bridge (see Figure 7) */
        if noReductions then break /* there is no polygons */
        seriesReductions /* perform all S,T series reductions (at most 2) */
        if noReductions then break /* the graph is S,T series-parallel irreducible */
    end loop
end algorithm
```

A *bond* is a graph having only one edge. We assume that the boolean *bond* in the text of the algorithm is a global variable set to **true** if the current graph is a bond. The procedure *polygonToChainStep* changes all the polygons of the current graph by the corresponding chains. The comment saying that the previous graph was a *bridge* means that the only possibility for reducing a S,T series-parallel graph with more than one edge (so, with more than four edges) to a single edge just by replacing all the polygons by chains (that is, without any series reduction) is that the graph was the *bridge* shown in Figure 7. The procedure *polygonToChainStep* can be presented as follows, taking into account the possible situation of Figure 6.

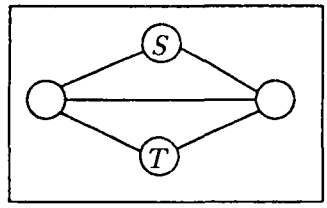


Figure 7: A S,T series-parallel irreducible bridge.

```
procedure polygonToChainStep /* replacing all polygons (at most 2) */
    noReductions := true
    p := lookForPolygon(1) /* first, it looks for a type 1 polygon */
    if p = ∅ then
        p := lookForPolygon() /* looking for a polygon (any type) */
    end if
    if p = ∅ then break
    noReductions := false
    update(p)
    if type(p) = 1 then
        p4 := lookForPolygon(4)
        if p4 ≠ ∅ then
            update(p4)
        else
            p := lookForPolygon(1)
            if p ≠ ∅ then
                update(p)
            end if
        end if
    end if
end procedure
```

The procedure *update* is

```
procedure update(p) /* perform all update operations */
    c := chain(p)
    Ω' := factor(p)
    replace(p,c)
    Ω := Ω * Ω'
end procedure
```

As in series-parallel reductions, the function *lookForPolygon*() returns a polygon if there is at least one, the empty set otherwise. With integer k as a parameter, it looks for a type k polygon. The procedure looks first for a type 1 polygon. If such a polygon exists, it is possible after its reduction to have a type 4 one and if there is no type 4 polygon, another type 1 may exists. In the updating procedure, the function *chain*(p) returns the chain replacing the polygon p together with the reliabilities of its edges computed from those of the edges of p; *replace*(p,c) replaces the polygon p by the chain c. The function *factor*(p) returns the factor corresponding to the polygon p according to the type of the polygon, using the formulae given before.

Again, we simply outline a description of the polygon-to-chain procedure. A real implementation can be more efficient (see the last paragraph of Subsection 3.1.)

4.2 Polygon–to–chain reductions with sensitivity analysis

Assume that the graph $\mathcal{G}$ has a type 1 polygon with the notation of Figure 4 for the concerned elementary reliabilities. Let us denote here by $\mathcal{G}'$ the reduced graph in which the polygon has been replaced by a chain according to the scheme of Figure 5. From (14) we have that for any edge e which does not belong to the polygon,

$$\frac{\partial R}{\partial r_e} = \Omega \frac{\partial R'}{\partial r_e}$$

and that, for instance,

$$\begin{aligned} \frac{\partial R}{\partial r_a} &= R' \frac{\partial \Omega}{\partial r_a} + \Omega \frac{\partial R'}{\partial r_a} \\ &= R' \frac{\partial \Omega}{\partial r_a} + \Omega \left(\frac{\partial R'}{\partial r_x} \frac{\partial r_x}{\partial r_a} + \frac{\partial R'}{\partial r_y} \frac{\partial r_y}{\partial r_a} \right). \end{aligned}$$

The expressions $\partial\Omega/\partial r_a$, $\partial r_x/\partial r_a$, $\partial r_x/\partial r_b$, etc., are obtained from the formulae shown before. Consider the case of a type 1 polygon. We have

$$\frac{\partial r_x}{\partial r_a} = \frac{r_b(1-r_c)(r_b+r_c-r_b r_c)}{(r_b+r_a r_c-r_a r_b r_c)^2}, \quad \frac{\partial r_x}{\partial r_b} = \frac{r_a r_b(1-r_a)(1-r_c)}{(r_b+r_a r_c-r_a r_b r_c)^2}, \text{ etc.}$$

The partial derivatives of Ω are obviously longer rational functions. One can write

$$\Omega = \frac{\Delta}{r_x r_y}$$

and thus, for instance,

$$\frac{\partial \Omega}{\partial r_a} = \frac{\frac{\partial \Delta}{\partial r_a} r_x r_y - \Delta \left(\frac{\partial r_x}{\partial r_a} r_y - \frac{\partial r_y}{\partial r_a} r_x \right)}{(r_x r_y)^2}.$$

Let us analyse in detail the steps that are needed to be able to perform a polygon–to–chain reduction as a preprocessing task, in order to carry out sensitivity computations too. Let us denote by $\mathcal{G}_0$ the initial graph. We perform first a complete S,T series–parallel reduction leading to the graph $\mathcal{G}_1$ and let us assume that $\mathcal{G}_1$ has more than one edge (so, more than four edges.) Since we are interested also in sensivities, with each edge e' in $\mathcal{G}_1$ we have constructed a set *containedBy*$[e']$ of edges of $\mathcal{G}_0$ and with each edge e in *containedBy*$[e']$, a weight $w_{e',e}$ such that

$$\text{for any } e \in \textit{containedBy}[e'] \text{ we have } \frac{\partial R}{\partial r_e} = w_{e',e} \frac{\partial R}{\partial r_{e'}}.$$

See Section 3 for the details. Observe that for any edge e on $\mathcal{G}_0$ there is an unique e' in $\mathcal{G}_1$ such that e belongs to *containedBy*$[e']$ (if e does not belong to any x,y series–parallel partial graph of $\mathcal{G}_0$ then $e'=e$, *containedBy*$[e]=\{e\}$ and $w_{e,e}=1.0$.)

Now, assume that a polygon p is found in $\mathcal{G}_1$ and that it is replaced by the chain c defining the new graph $\mathcal{G}_2$. If R_i denotes the reliability between S and T in $\mathcal{G}_i$, we have that

$$R = R_0 = R_1 = \Omega_2 R_2$$

with the factor Ω_2 depending on the reliabilities of the edges in the polygon p only. Let $e \in \mathcal{G}_0$ and let e' be the unique edge of $\mathcal{G}_1$ such that $e \in$ *containedBy*$[e']$ (eventually, $e'=e$.) The sensitivity of R

with respect to e can be written

$$\begin{aligned}\sigma_e = \frac{\partial R}{\partial r_e} &= w_{e',e}\frac{\partial R_1}{\partial r_{e'}} \\ &= w_{e',e}\frac{\partial(\Omega_2 R_2)}{\partial r_{e'}} \\ &= w_{e',e}\left(R_2\frac{\partial \Omega_2}{\partial r_{e'}} + \Omega_2\frac{\partial R_2}{\partial r_{e'}}\right) \\ &= w_{e',e}\left(R_2\frac{\partial \Omega_2}{\partial r_{e'}} + \Omega_2\sum_{e''\in\mathcal{G}_2}\frac{\partial R_2}{\partial r_{e''}}\frac{\partial r_{e''}}{\partial r_{e'}}\right).\end{aligned}$$

If $e \in \mathcal{G}_2$ then

$$e' = e,\ \frac{\partial \Omega_2}{\partial r_{e'}} = 0,\ \frac{\partial r_{e''}}{\partial r_{e'}} = 1_{(e''=e')},\ w_{e',e} = 1.0$$

$$\text{and } \sigma_e = \Omega_2\frac{\partial R_2}{\partial r_e},$$

where 1_B is the indicator function of the boolean B. If $e \notin \mathcal{G}_2$, then if $e \neq e'$ and $e' \in \mathcal{G}_2$ we have

$$e' \notin p,\ \frac{\partial \Omega_2}{\partial r_{e'}} = 0,\ \frac{\partial r_{e''}}{\partial r_{e'}} = 1_{(e''=e')}$$

$$\text{and } \sigma_e = w_{e',e}\Omega_2\frac{\partial R_2}{\partial r_{e'}},$$

else ($e' \notin \mathcal{G}_2$),

$$e' \in p,\ \frac{\partial r_{e''}}{\partial r_{e'}} \neq 0 \Longleftrightarrow e'' \in c$$

$$\text{and } \sigma_e = w_{e',e}\left(R_2\frac{\partial \Omega_2}{\partial r_{e'}} + \Omega_2\sum_{e''\in c}\frac{\partial R_2}{\partial r_{e''}}\frac{\partial r_{e''}}{\partial r_{e'}}\right).$$

Every edge $e' \in p$ has disappeared in the reduction leading from $\mathcal{G}_1$ to $\mathcal{G}_2$; the new edges in the graph are those of the chain c. For any edge $e'' \in c$, we define

$$containedBy[e''] \stackrel{\text{def}}{=} \bigcup_{e'\in p} containedBy[e']$$

and for every $e \in containedBy[e'']$, let us set

$$w_{e'',e} \stackrel{\text{def}}{=} w_{e',e}\Omega_2\frac{\partial r_{e''}}{\partial r_{e'}}$$

where e' is the unique edge of p (i.e. of $\mathcal{G}_1$) such that $e \in containedBy[e']$. Last, for every $e \in \mathcal{G}_0$, a real variable ϕ_e is initialized by

$$\phi_e := \begin{cases} 0 & \text{if } e \in containedBy[e'] \text{ with } e' \notin p \text{ (eventually } e' = e) \\ w_{e',e}\dfrac{\partial \Omega_2}{\partial r_{e'}} & \text{if } e \in containedBy[e'] \text{ with } e' \in p \end{cases}$$

If the reliability R_2 is known and if for all $e'' \in \mathcal{G}_2$ the sensitivities $\partial R_2/\partial r_{e''}$ are also known, the sensitivities σ_e, $e \in \mathcal{G}$, can be obtained by

$$\sigma_e = \phi_e R_2 \ + \sum_{\substack{e''\in\mathcal{G}_2/ \\ e\in containedBy[e'']}} w_{e'',e}\frac{\partial R_2}{\partial r_{e''}}. \tag{15}$$

The proof follows from the preceeding remarks, by taking into account successively the cases (i) $e \in \mathcal{G}_2$, (ii) $e \in containedBy[e']$, $e' \in \mathcal{G}_1$, $e' \neq e$ and $e' \in \mathcal{G}_2$, (iii) $e \in containedBy[e']$, $e' \in \mathcal{G}_1$ and $e' \in p$.

Assume that after i reduction steps, we have constructed a graph $\mathcal{G}_i$ and the following associated information: the current factor Ω, for each edge $e' \in \mathcal{G}_i$ the set $containedBy[e']$ of edges of the original graph $\mathcal{G} = \mathcal{G}_0$ together with the weights $w_{e',e}$ associated with each $e \in containedBy[e']$ and, for each $e \in \mathcal{G}$, the numbers ϕ_e, such that

$$R = \Omega R_i$$

and that, for any $e \in \mathcal{G}$,

$$\sigma_e = \phi_e R_i \; + \sum_{\substack{e' \in \mathcal{G}_i / \\ e \in containedBy[e']}} w_{e',e} \frac{\partial R_i}{\partial r_{e'}}. \tag{16}$$

Let us show that the same expression allows to compute the measures of $\mathcal{G}$ from those of $\mathcal{G}_{i+1}$ once a supplementary reduction step is performed, if the convenient update operations are carried out on the associated information.

Let us suppose that either a polygon or a S,T series are deleted from $\mathcal{G}_i$ and replaced by a chain (eventually a single edge) to obtain $\mathcal{G}_{i+1}$. In any case, let us denote by p the deleted partial graph and by c the replacing chain. If the ith step is a S,T series reduction, then Ω is not changed, otherwise, it is updated by $\Omega := \Omega_{i+1}\Omega$, where Ω_{i+1} is the factor correponding to the poygon p. In all cases, we can update Ω by $\Omega := \Omega_{i+1}\Omega$ if $\Omega_{i+1} = 1.0$ in the case of a S,T series reduction. We can write

$$\begin{aligned}
\sigma_e &= \phi_e R_i \; + \sum_{\substack{e' \in \mathcal{G}_i / \\ e \in containedBy[e']}} w_{e',e} \frac{\partial R_i}{\partial r_{e'}} \\
&= \phi_e \Omega_{i+1} R_{i+1} \; + \sum_{\substack{e' \in \mathcal{G}_i / \\ e \in containedBy[e']}} w_{e',e} \left(R_{i+1} \frac{\partial \Omega_{i+1}}{\partial r_{e'}} + \Omega_{i+1} \frac{\partial R_{i+1}}{\partial r_{e'}} \right) \\
&= \left(\phi_e \Omega_{i+1} \; + \sum_{\substack{e' \in \mathcal{G}_i / \\ e \in containedBy[e']}} w_{e',e} \frac{\partial \Omega_{i+1}}{\partial r_{e'}} \right) R_{i+1} \; + \sum_{\substack{e' \in \mathcal{G}_i / \\ e \in containedBy[e']}} w_{e',e} \Omega_{i+1} \sum_{e'' \in \mathcal{G}_{i+1}} \frac{\partial R_{i+1}}{\partial r_{e''}} \frac{\partial r_{e''}}{\partial r_{e'}}.
\end{aligned}$$

This leads to the following updating formulae: For every edge $e'' \in c$, we set, as in the reduction from $\mathcal{G}_1$ to $\mathcal{G}_2$,

$$containedBy[e''] := \bigcup_{e' \in p} containedBy[e'].$$

For all $e'' \in c$ and for all $e \in containedBy[e'']$,

$$w_{e'',e} := \sum_{\substack{e' \in p / \\ e \in containedBy[e']}} w_{e',e} \Omega_{i+1} \frac{\partial r_{e''}}{\partial r_{e'}}.$$

Last, for every $e \in \mathcal{G}$, the factor ϕ_e is updated by

$$\begin{aligned}
\phi_e &:= \phi_e \Omega_{i+1} \; + \sum_{\substack{e' \in \mathcal{G}_i / \\ e \in containedBy[e']}} w_{e',e} \frac{\partial \Omega_{i+1}}{\partial r_{e'}} \\
&= \phi_e \Omega_{i+1} \; + \sum_{\substack{e' \in p / \\ e \in containedBy[e']}} w_{e',e} \frac{\partial \Omega_{i+1}}{\partial r_{e'}}.
\end{aligned}$$

In algorithmic form, we only have to rewrite the procedure *update*, which we do in the following manner:

```
procedure update(p) /* perform all update operations */
    c := chain(p)
    Ω' := factor(p)
    replace(p,c)
    Ω := Ω * Ω'
    S := ∅ /* auxiliary set */
    for all e' in p do
        S := S ∪ containedBy[e']
    end for
    for all e'' in c do
        containedBy[e''] := S
        for all e in containedBy[e''] do
            weight[e'', e] := 0.0
        end for
    end for
    for all e' in p do
        for all e in containedBy[e'] do
            for all e'' in c do
                weight[e'', e] := weight[e'', e] + weight[e', e] * ∂r_{e''}/∂r_{e'}
            end for
        end for
    end for
    for all e in G do
        φ[e] := φ[e] * Ω'
    end for
    for all e' in p do
        for all e in containedBy[e'] do
            φ[e] := φ[e] * weight[e', e] ∂Ω'/∂r_{e'}
        end for
    end for
end procedure
```

4.3 Example

(The reader is suggested to look at the example of series–parallel reductions with sensitivity analysis, Subsection 3.3, before reading this subsection.) To illustrate these procedures, let us consider the graph shown in Figure 8, taken from [16]. It represents a version of the well known Arpanet communication network (see [16] for details on the model.) It has 21 nodes and 26 edges. Assume that all the elementary reliabilities are equal to $r = 0.9$. Let ud analyze in detail the first steps of a polygon–to–chain reduction on this graph.

Since the graph is S,T series–parallel reducible, the first step in the polygon–to–chain procedure is to perform all the S,T series–parallel reductions. The result of this first step is shown in Figure 9 and the associated information necessary to perform sensitivity computations is presented in Table 2.

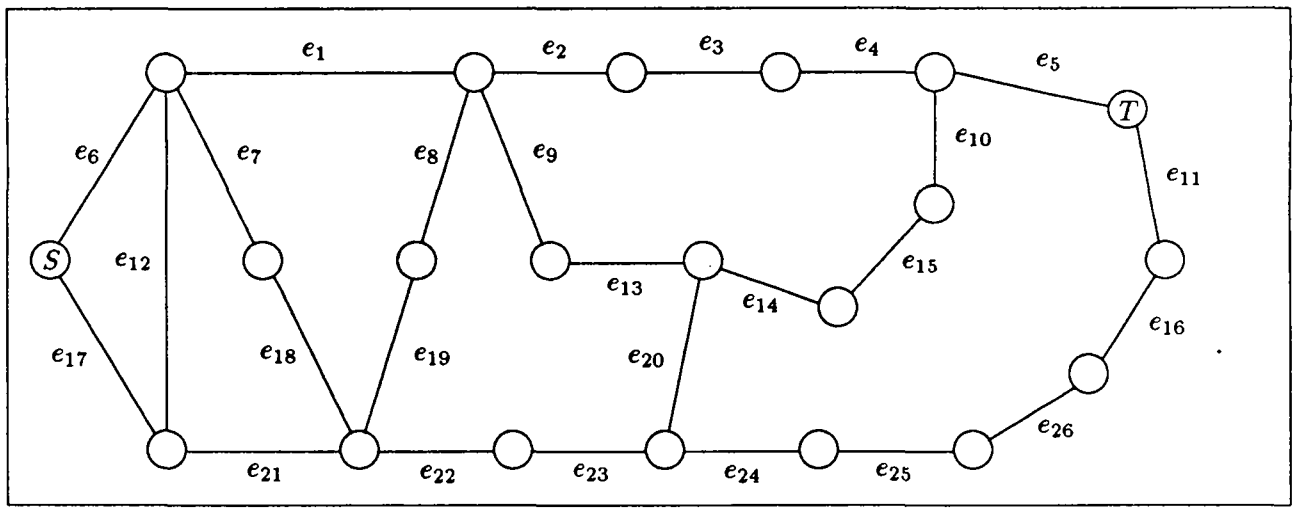


Figure 8: A S,T series-parallel reducible graph $\mathcal{G}_0$.

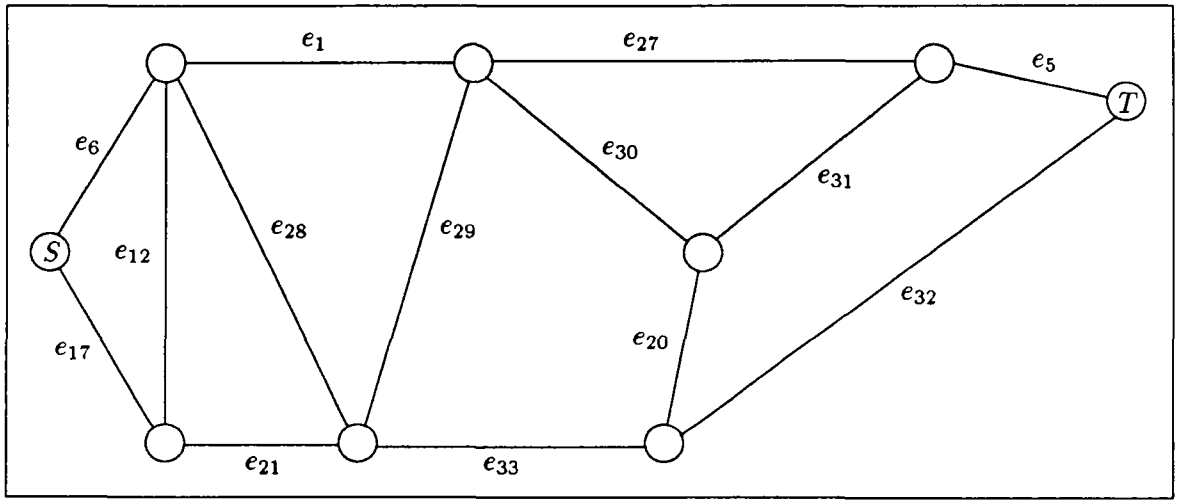


Figure 9: The complete S,T series-parallel reduction $\mathcal{G}_1$ of the graph $\mathcal{G}_0$ in Figure 8.

The reliability of the new edges are the following:

$$\begin{aligned} r_{e_{27}} &= 0.729, \\ r_{e_{28}} &= 0.81, \\ r_{e_{29}} &= 0.81, \\ r_{e_{30}} &= 0.81, \\ r_{e_{31}} &= 0.729, \\ r_{e_{32}} &= 0.59049, \\ r_{e_{33}} &= 0.81. \end{aligned}$$

The next step is to replace the polygon defined by the set of edges $\{e_6, e_{17}, e_{12}\}$ by a chain whose edges are labeled e_{34} and e_{35}. The resulting graph $\mathcal{G}_2$ is shown in Figure 10.

The reliabilities of the introduced edges and the factor Ω_2 are

$$\begin{aligned} r_{e_{34}} = r_{e_{35}} &\approx 0.990826, \\ \Omega_2 &\approx 0.990083. \end{aligned}$$

e_{27}		e_{28}		e_{29}		e_{30}		e_{31}		e_{32}		e_{33}	
e_2	0.81	e_7	0.9	e_8	0.9	e_9	0.9	e_{10}	0.81	e_{11}	0.6561	e_{22}	0.9
e_3	0.81	e_{18}	0.9	e_{19}	0.9	e_{13}	0.9	e_{14}	0.81	e_{16}	0.6561	e_{23}	0.9
e_4	0.81							e_{15}	0.81	e_{24}	0.6561		
										e_{25}	0.6561		
										e_{26}	0.6561		

Table 2: The sets *containedBy*[.] and the associated weights, after the complete S,T series-parallel reduction step leading from $\mathcal{G}_0$ to $\mathcal{G}_1$. Only the sets of cardinality ≥ 2 are shown. For instance, $containedBy[e_{33}] = \{e_{22}, e_{23}\}$ and $w_{e_{33},e_{22}} = w_{e_{33},e_{23}} = 0.9$. For every $e \notin \{e_{27}, e_{28}, e_{30}, e_{31}, e_{32}, e_{33}\}$, we have $containedBy[e] = \{e\}$ and $w_{e,e} = 1.0$.

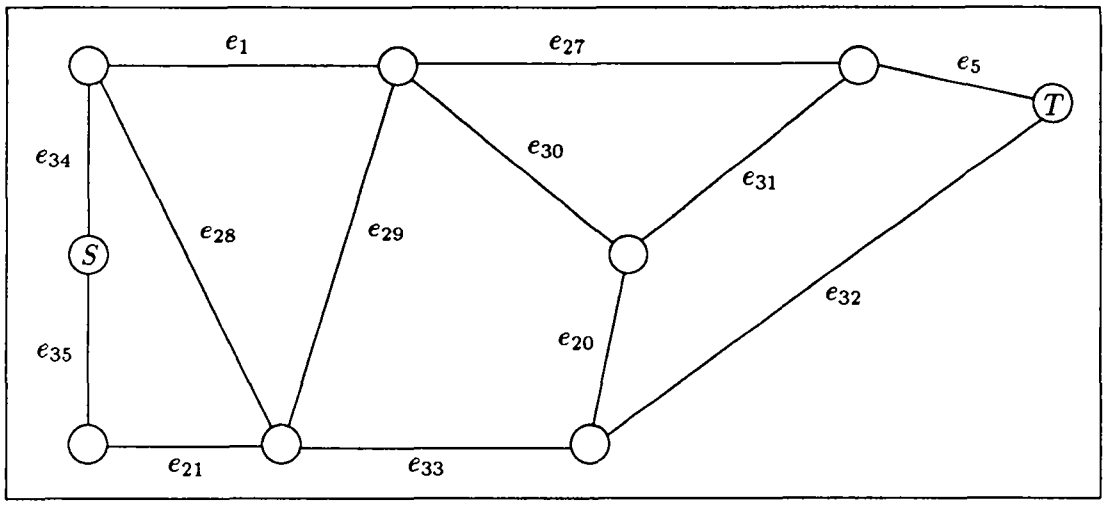


Figure 10: The result $\mathcal{G}_2$ of replacing the polygon $\{e_6, e_{17}, e_{12}\}$ in the graph $\mathcal{G}_1$ of Figure 9

The complementary information associated with the new edges in $\mathcal{G}_2$ is:

- *containedBy*[ ] sets:

  $containedBy[e_{34}] = \{e_6, e_{12}, e_{17}\}$,
  $w_{e_{34},e_6} = w_{e_{34},e_{12}} \approx 0.916667\,10^{-1}$,
  $w_{e_{34},e_{17}} \approx -0.833333\,10^{-2}$,
  $containedBy[e_{35}] = \{e_6, e_{12}, e_{17}\}$,
  $w_{e_{35},e_6} \approx -0.833333\,10^{-2}$,
  $w_{e_{35},e_{12}} = w_{e_{35},e_{17}} \approx 0.916667\,10^{-1}$.
  for all $e' \in \mathcal{G}_2$, $e' \notin \{e_{34}, e_{35}\}$, for all $e \in containedBy[e']$, $w_{e',e} := w_{e',e}\Omega_2$.

- ϕ function:

  $\phi_{e_6} = \phi_{e_{17}} \approx 0.992438\,10^{-1}$, $\phi_{e_{12}} \approx -0.168210\,10^{-2}$,
  and for all $e \notin \{e_6, e_{12}, e_{17}\}$, $\phi_e = 0.0$.

For instance, to check the value of w_{e_{34},e_6}, the reader can verify that

$$\begin{aligned}\frac{\partial r_{e_{34}}}{\partial r_{e_6}} &= \left.\frac{\partial r_x}{\partial r_a}\right|_{r_a=r_b=r_c=r} \\ &= \frac{(1-r)(2-r)}{(1+r-r^2)^2} \\ \text{and, in the same way, } \Omega_2 &= \frac{(1+r-r^2)^2}{3-2r}, \\ \text{so that } w_{e_{34},e_6} &= \frac{(1-r)(2-r)}{3-2r}.\end{aligned}$$

A new series consisting of edges e_{35} and e_{21} is then reduced and the result $\mathcal{G}_3$ is shown in Figure 11.

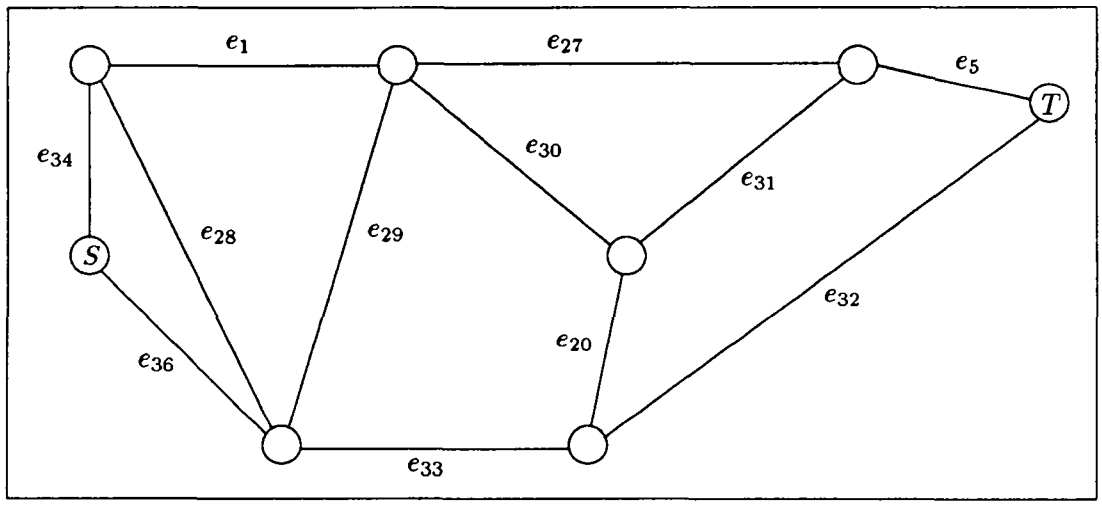


Figure 11: The result $\mathcal{G}_3$ of replacing the series $\{e_{21}, e_{35}\}$ in the graph of Figure 10.

The reliability of the new edge e_{36} is

$$r_{e_{36}} = r_{e_{21}} r_{e_{35}} \approx 0.891743$$

and the complementary information is the following:

- *containedBy*[ ] sets:

$$\begin{aligned}\mathit{containedBy}[e_{36}] &= \{e_6, e_{12}, e_{17}, e_{21}\}, \\ w_{e_{36},e_6} &= r_{e_{21}} w_{e_{35},e_6} = -0.0075, \\ w_{e_{36},e_{12}} &= r_{e_{21}} w_{e_{35},e_{12}} = 0.0825, \\ w_{e_{36},e_{17}} &= r_{e_{21}} w_{e_{35},e_{17}} = 0.0825, \\ w_{e_{36},e_{21}} &= r_{e_{35}} w_{e_{21},e_{21}} = 0.981.\end{aligned}$$

- Function ϕ is not modified.

A new type 1 polygon appears (edges e_{28}, e_{34}, e_{36}) whose replacement introduces edges e_{37}, e_{38}, the first one being in series with e_1. This last series is replaced by edge e_{39} and the result is shown in Figure 12.

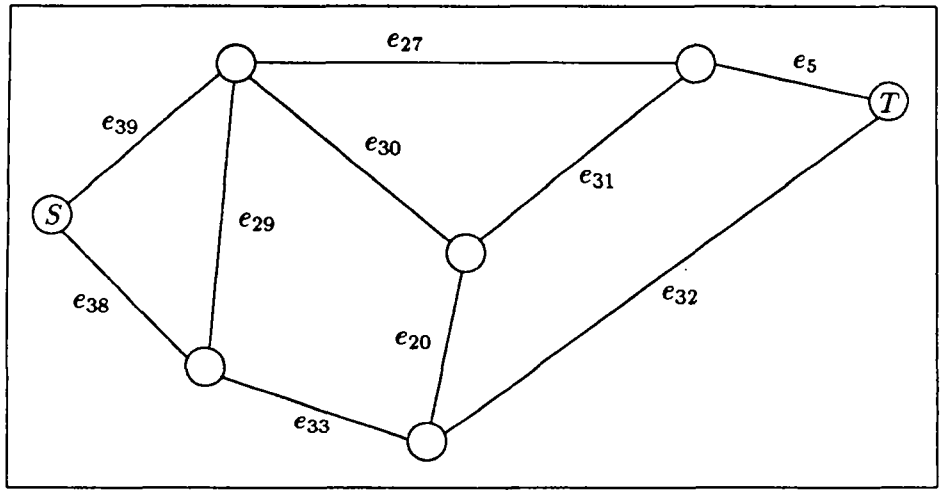


Figure 12: The S, T polygon–to–chain reduction of the graph in Figure 9.

The reliabilities of the edges in this graph are the following:

$$\begin{aligned} r_{e_5} &= 0.9, \\ r_{e_{20}} &= 0.9, \\ r_{e_{27}} &= 0.729, \\ r_{e_{29}} &= 0.81, \\ r_{e_{30}} &= 0.81, \\ r_{e_{31}} &= 0.729, \\ r_{e_{32}} &= 0.59049, \\ r_{e_{33}} &= 0.81, \\ r_{e_{38}} &\approx 0.979568, \\ r_{e_{39}} &\approx 0.898571, \end{aligned}$$

and the new factor, Ω_4, is

$$\Omega_4 \approx 0.999039$$

which gives a total factor

$$\Omega = \Omega_2 \Omega_4 \approx 0.989132.$$

The complementary information relying this last graph to the initial one is:

- *containedBy*[ ] sets

$$\begin{aligned} &\mathit{containedBy}[e_5] = \{e_5\}, \\ &\quad w_{e_5,e_5} \approx 0.989132, \\ &\mathit{containedBy}[e_{20}] = \{e_{20}\}, \\ &\quad w_{e_{20},e_{20}} \approx 0.989132, \\ &\mathit{containedBy}[e_{27}] = \{e_2, e_3, e_4\}, \\ &\quad w_{e_{27},e_2} = w_{e_{27},e_3} = w_{e_{27},e_4} \approx 0.801197, \\ &\mathit{containedBy}[e_{29}] = \{e_8, e_{19}\}, \\ &\quad w_{e_{29},e_8} = w_{e_{29},e_{19}} \approx 0.890219, \end{aligned}$$

$containedBy[e_{30}] = \{e_9, e_{13}\}$,
$w_{e_{30},e_9} = w_{e_{30},e_{13}} \approx 0.890219$,
$containedBy[e_{31}] = \{e_{10}, e_{14}, e_{15}\}$,
$w_{e_{31},e_{10}} = w_{e_{31},e_{14}} = w_{e_{31},e_{15}} \approx 0.801197$,
$containedBy[e_{32}] = \{e_{11}, e_{16}, e_{24}, e_{25}, e_{26},\}$,
$w_{e_{32},e_{11}} = w_{e_{32},e_{16}} = w_{e_{32},e_{24}} = w_{e_{32},e_{25}} = w_{e_{32},e_{26}} \approx 0.648970$,
$containedBy[e_{33}] = \{e_{22}, e_{23}\}$,
$w_{e_{33},e_{22}} = w_{e_{33},e_{23}} \approx 0.890219$,
$containedBy[e_{38}] = \{e_6, e_7, e_{12}, e_{17}, e_{18}, e_{21}\}$,
$w_{e_{38},e_6} = -0.278286\ 10^{-2}, w_{e_{38},e_7} = 0.958812\ 10^{-1}$,
$w_{e_{38},e_{12}} = 0.142009\ 10^{-1}, w_{e_{38},e_{17}} = 0.156928\ 10^{-1}$,
$w_{e_{38},e_{18}} = 0.958812\ 10^{-1}, w_{e_{38},e_{21}} = 0.185123$,
$containedBy[e_{39}] = \{e_1, e_6, e_7, e_{12}, e_{17}, e_{18}, e_{21}\}$,
$w_{e_{39},e_1} = 0.987561, w_{e_{39},e_6} \approx 0.142912\ 10^{-1}$,
$w_{e_{39},e_7} \approx 0.683734\ 10^{-2}, w_{e_{39},e_{12}} \approx 0.141730\ 10^{-1}$,
$w_{e_{39},e_{17}} \approx -0.140666\ 10^{-2}, w_{e_{39},e_{18}} \approx 0.683733$,
$w_{e_{39},e_{21}} \approx -0.128846\ 10^{-2}$,

- ϕ function:

$\phi_{e_6} \approx 0.108684, \phi_{e_7} \approx -0.307528\ 10^{-3}$,
$\phi_{e_{12}} \approx 0.865596\ 10^{-2}, \phi_{e_{17}} \approx 0.990102\ 10^{-1}$,
$\phi_{e_{18}} \approx -0.307528\ 10^{-3}, \phi_{e_{21}} \approx 0.873549\ 10^{-2}$,
and for all $e \notin \{e_6, e_7, e_{12}, e_{17}, e_{18}, e_{21}\}$, $\phi_e = 0.0$.

The reduction can be pursuived further since there is again a type 1 polygon in the last graph (lines e_{29}, e_{38}, e_{39}), etc. Stopping here for shortness, let us illustrate the computation of some sensitivities in the original graph. Assume that we compute the reliability and the sensitivities in $\mathcal{G}_4$. We obtain $R_4 \approx 0.914516$ and then $R = \Omega R_4 \approx 0.904577$. Concerning the sensitivities, let us consider the line e_1. Since e_1 belongs only to the set $containedBy[e_{39}]$, we have

$$\begin{aligned} \sigma_{e_1} &= \phi_{e_1} R_4 + w_{e_{39},e_1} \frac{\partial R_4}{\partial r_{e_{39}}} \\ &\approx 0.987651 \times 0.0564664 \\ &\approx 0.0558. \end{aligned}$$

For e_5, since it belongs to the last graph in the reduction process, we can write

$$\begin{aligned} \sigma_{e_5} &= w_{e_5,e_5} \frac{\partial R_4}{\partial r_{e_5}} \quad (\phi_{e_5} = 0) \\ &\approx 0.989132 \times 0.386059 \\ &\approx 0.3819. \end{aligned}$$

Last, considering e_6 which belongs to the sets $containedBy[e_{38}]$ and $containedBy[e_{39}]$, we have

$$\begin{aligned} \sigma_{e_6} &= \phi_{e_6} R_4 + w_{e_{38},e_6} \frac{\partial R_4}{\partial r_{e_{38}}} + w_{e_{39},,e_6} \frac{\partial R_4}{\partial r_{e_{39}}} \\ &\approx 0.108684 \times 0.914516 - 0.00278286 \times 0.09685340 + 0.0141730 \times 0.0564664 \\ &\approx 0.0999. \end{aligned}$$

5 Sensitivity analysis in factoring algorithms

5.1 Factoring algorithms

We consider here a family of algorithms based on relation (3). If we make explicit the dependency of R with respect to $\mathcal{G}$, we can rewrite (3) as

$$R(\mathcal{G}) = r_e R(\mathcal{G}_e^c) + (1 - r_e) R(\mathcal{G}_e^d). \tag{17}$$

Relation (17) says that we can reduce the problem of computing the reliability of $\mathcal{G}$ to the computation of the reliabilities of two smaller graphs, $\mathcal{G}_e^c$ and $\mathcal{G}_e^d$. Of course, applying (17) recursively leads to a virtual binary tree (also called *associated binary search structure*) with $2^M - 1$ nodes (recall that M is the number of edges of the original graph.) The idea is then the following. Before the recursive application of (17), we use a simplification technique as those seen in the previous sections (which are polynomial in time.) As deleting and contracting operations lead to smaller graphs, sooner or later simplifications will appear reducing the size of the constructed tree. To be more specific, let us formalize the method in the form of a recursive procedure.

```
procedure FACT(G) /* factoring procedure */
    reduce(G)
    if bond(G) then
        return fiabBond(G)
    else
        e := chooseEdge(G)
        return r[e]*FACT(G_e^c) + (1 - r[e])*FACT(G_e^d)
    end if
end procedure
```

The `reduce()` procedure may be replaced for instance by `seriesParallelReduction()` or by `polygonToChainReduction()`. A *bond* is a graph having two vertices and a single edge; the function $bond(\mathcal{G})$ returns `true` iff $\mathcal{G}$ is a bond and $fiabBond(\mathcal{G})$ returns the reliability of its single edge. Observe that in the case of a S,T series–parallel reductible node in the binary search structure, the reduced graph is a bond. The key point is the `chooseEdge()` procedure since it is easily seen that different strategies for selecting the next edge on which we will condition leads to different binary trees. There is a mathemathical tool, *domination theory*, that provides the support to develop optimal selection procedures in the case of the 2-terminal reliability measure and in the more general K–terminal one for some sizes of the K set. The reader is suggested to see [14] to have a general view on the subject.

Reported experiences say that in general this approach performs fast. Of course, in the worst case, the reductions arrive late in the construction of the tree and the computational time is exponential in the size of the network. Another important property of factoring algorithms is that they need a storage of about $O(M^2)$ since the stack used to implement the recursive procedure has, at most, the size $M - 1$. In contrast, boolean techniques usually need to start from the set of minpaths or mincuts which are exponential in number.

Considering the 2-terminal reliability measure, the theory was developped in [17] in the case of series–parallel reductions and it leads to an optimal selection `chooseEdge()` procedure. Optimal means that this procedure generates the minimum possible number of leaf nodes in the generated binary tree, that is, the size of the tree is the minimal possible size. This is the case that will be considered in this paper. The more general polygon–to–chain reductions were analysed in [18] (see the end of this subsection.)

To present the basic principles, let us recall briefly some definitions from the theory of graphs. If x is a vertex of $\mathcal{G}$, we denote by $\mathcal{G} - x$ the subgraph of $\mathcal{G}$ obtained by deleting node x and its

adjacent edges. Vertex x of $\mathcal{G}$ is an *cutvertex* of $\mathcal{G}$ if $\mathcal{G}$ has more than two vertices and $\mathcal{G} - x$ is not connected. Let u, v be two different vertices of $\mathcal{G}$. The graph $\mathcal{G}$ is a u, v coherent graph if any edge of $\mathcal{G}$ belongs to a minpath between nodes u and v. Let $\mathcal{G}$ be a u, v coherent graph. A family of minpaths $\mathcal{F} = \{\mu_1, \ldots, \mu_L\}$ between u and v is a u, v *formation* of $\mathcal{G}$ if the union of all $\mu \in \mathcal{F}$ is $\mathcal{G}$. The *signed domination* $d_{u,v}$ of $\mathcal{G}$ is the number of u, v formations with an odd number of minpaths minus the number of u, v formations with an even number of minpaths. The *domination* $D_{u,v}$ is the absolute value of $d_{u,v}$. The definition of domination or signed domination can be extended to any graph $\mathcal{G}$ by setting $d_{u,v} = 0$ and $D_{u,v} = 0$ if $\mathcal{G}$ is not a u, v coherent graph. It can be shown that if $\mathcal{G}$ is a u, v coherent graph, then $D_{u,v} > 0$. Also, $\mathcal{G}$ is u, v series–parallel reducible iff $D_{u,v} = 1$. Domination has many interesting properties. Two fundamental ones are that for eny edge e of $\mathcal{G}$,

$$D(\mathcal{G}) = D(\mathcal{G}_e^c) + D(\mathcal{G}_e^d) \tag{18}$$

and that domination is invariant under S, T series–parallel reductions.

The main result in [17] stays that if $\mathcal{G}$ is not a bond, if it is a S, T coherent graph and S, T series–parallel irreducible, then there exists an edge e such that both graphs $\mathcal{G}_e^c$ and $\mathcal{G}_e^d$ are S, T coherent. Moreover, if `chooseEdge(`$\mathcal{G}'$`)` returns such an edge e when $\mathcal{G}'$ is as in the enonced result, then the binary tree is optimal in size (and its number of leaves is exactly equal to $D(\mathcal{G})$, the domination of the original graph $\mathcal{G}$, assuming that $\mathcal{G}$ is S, T coherent.) The key property to prove this result is relation (18). The paper is not very explicit in the efficient construction of such an `chooseEdge()` procedure. Details can be found in [17] and [14]. In [18] an extension of these results was done for polygon–to–chain reductions. The theory uses another invariant called *minimal domination* but the principles are similar for our purposes in this paper.

5.2 Factoring algorithms with sensitivity analysis

First, observe that for all edge $f \neq e$,

$$\sigma_f = r_e \frac{\partial R_e^c}{\partial r_f} + (1 - r_e) \frac{\partial R_e^d}{\partial r_f}. \tag{19}$$

We have already seen that (relation (4)) $\sigma_e = R_e^c - R_e^d$. This allows to obtain the sensitivities in $\mathcal{G}$ from the sensitivities in the graphs R_e^c and R_e^d. Since the contraction and deletion of edges do not create new ones, there is no need to manage pointers as in the previous sections (*containedBy* sets.) The following modification of the previous procedure allows to deal with sensitivities. To simplify the presentation, let us modify the syntax putting all the parameters after the name of the `FACT` procedure, whose call is now `FACT(G,`σ`[],R)`. In the same way, to perform a S, T series–parallel reduction of $\mathcal{G}$ we use *seriesParallelReduction*($\mathcal{G}$,$\mathcal{G}'$,*containedBy*[],*weight*[,]). Let us add the function *edge*($\mathcal{G}'$) returning the only edge of $\mathcal{G}'$ when $\mathcal{G}'$ is a bond.

```
procedure FACT(G,σ[],R) /* factoring procedure with sensitivity analysis */
    seriesParallelReduction(G,G',containedBy[],weight[,])
    if bond(G') then
        e' := edge(G')
        R := r[e']
        for all e in G do
            σ[e] := weight[e', e]
        end for
    else
        e := chooseEdge(G')
```

```
            FACT(G'_e^c, σ_1, R_1)
            FACT(G'_e^d, σ_2, R_2)
            R := r[e] * R_1 + (1 - r[e]) * R_2
            for all f in G', f ≠ e do
                σ[f] := r[e] * σ_1[f] + (1 - r[e]) * σ_2[f]
            end for
            σ[e] := R_1 - R_2
            if G' ≠ G then
                for all e' in G' do
                    for all e in containedBy[e'] do
                        σ[e] := weight[e', e]
                    end for
                end for
            end if
        end if
    end procedure
```

5.3 Example

Consider the following graph:

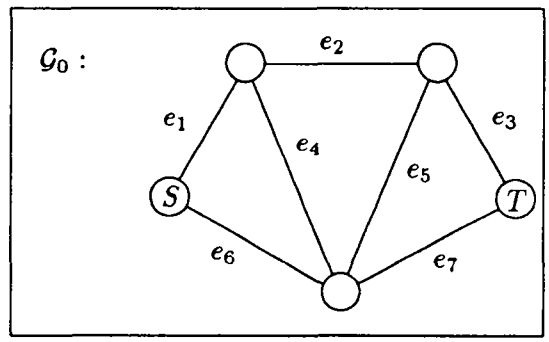


Figure 13: A S,T series-parallel irreducible example: $\mathcal{G}_0$.

Assume that the first conditionning is done with respect to edge e_2. Figure 14 shows the resulting graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ together with their respective S,T series-parallel reductions.

It remains graph $\mathcal{G}_3$ to continue the factoring procedure. Choosing edge e_1, the result is shown in Figure 15.

Let us illustrate the computation of the sensitivities in symbolic form. To simplify the text, we denote by r_i the reliability r_{e_i}. Calculating first the reliability of the network, we start with the nodes $\mathcal{G}_5$ and $\mathcal{G}_6$. They are S,T series-parallel graphs and we have:

$$\begin{aligned} R(\mathcal{G}_5) &= r_3 + r_7(r_6 + r_8 - r_6 r_8) - r_3 r_7(r_6 + r_8 - r_6 r_8), \\ \text{and } R(\mathcal{G}_6) &= r_6(r_7 + r_3 r_8 - r_3 r_7 r_8). \end{aligned}$$

The reliability of $\mathcal{G}_1 = \mathcal{G}_3$ comes from

$$R(\mathcal{G}_3) = r_1 R(\mathcal{G}_5) + (1 - r_1) R(\mathcal{G}_6). \tag{20}$$

To write it as a function of the reliabilities of the edges in $\mathcal{G}_1$ only, we have to replace r_8 in $R(\mathcal{G}_3)$ by $r_4 + r_5 - r_4 r_5$. As $\mathcal{G}_2$ is S,T series-parallel, we compute

$$R(\mathcal{G}_2) = (r_6 + r_1 r_4 - r_1 r_4 r_6)(r_7 + r_3 r_5 - r_3 r_5 r_7).$$

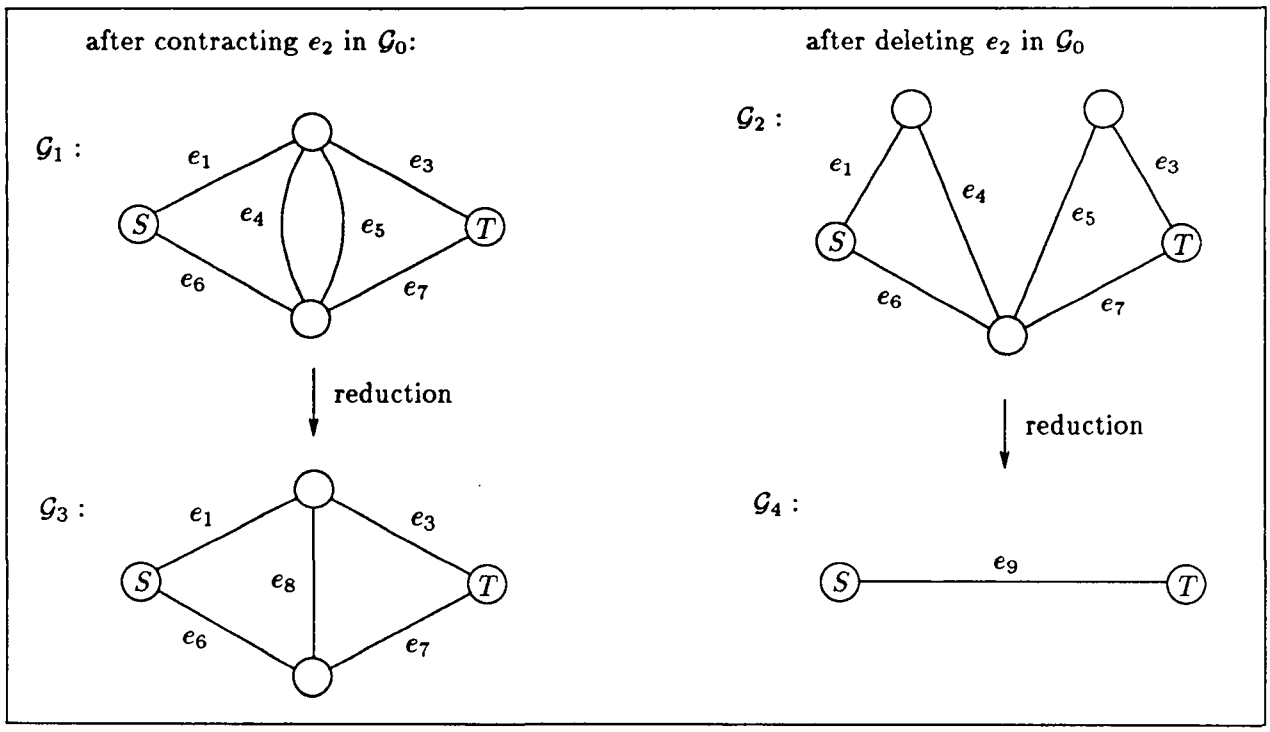


Figure 14: The first level in the virtual binary tree induced by the factoring algorithm starting from $\mathcal{G}_0$ and choosing edge e_2.

Last, from

$$R(\mathcal{G}_0) = r_2 R(\mathcal{G}_1) + (1 - r_2) R(\mathcal{G}_2) \tag{21}$$

we obtain the reliability $R = R(\mathcal{G}_0)$ of the network. Evaluated in the point $r_i = r$, $1 \leq i \leq 7$, this gives

$$R = r^2(1 + 3r + r^2 - 12r^3 + 11r^4 - 3r^5).$$

Concerning the sensitivities, we have from (21),

$$\frac{\partial R(\mathcal{G}_0)}{\partial r_i} = \begin{cases} r_2 \dfrac{\partial R(\mathcal{G}_1)}{\partial r_i} + (1 - r_2) \dfrac{\partial R(\mathcal{G}_2)}{\partial r_i} & \text{if } i \neq 2, \\ R(\mathcal{G}_1) - R(\mathcal{G}_2) & \text{if } i = 2. \end{cases}$$

In graph $\mathcal{G}_1$,

$$\frac{\partial R(\mathcal{G}_1)}{\partial r_i} = \begin{cases} \dfrac{\partial R(\mathcal{G}_3)}{\partial r_i} & \text{if } i \neq 4, 5, \\ (1 - r_5) \dfrac{\partial R(\mathcal{G}_3)}{\partial r_4} & \text{if } i = 4, \\ (1 - r_4) \dfrac{\partial R(\mathcal{G}_3)}{\partial r_5} & \text{if } i = 5. \end{cases}$$

For instance, with respect to r_1, we have

$$\begin{aligned} \frac{\partial R(\mathcal{G}_0)}{\partial r_1} &= r_2 \frac{\partial R(\mathcal{G}_1)}{\partial r_1} + (1 - r_2) \frac{\partial R(\mathcal{G}_2)}{\partial r_1} \\ &= r_2 \frac{\partial R(\mathcal{G}_3)}{\partial r_1} + (1 - r_2) \frac{\partial R(\mathcal{G}_2)}{\partial r_1} \end{aligned}$$

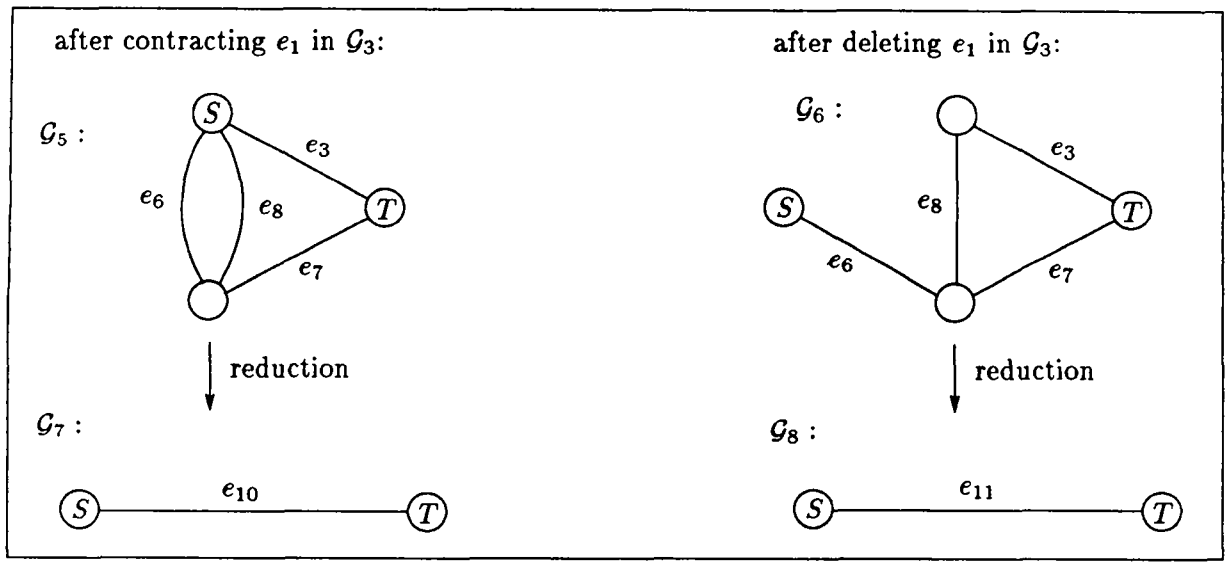


Figure 15: The result of conditionning with respect to e_1 in $\mathcal{G}_3$.

$$= r_2(R(\mathcal{G}_5) - R(\mathcal{G}_6)) + (1 - r_2)\frac{\partial R(\mathcal{G}_2)}{\partial r_1}.$$

It remains to obtain $\frac{\partial R(\mathcal{G}_2)}{\partial r_1}$ as shown in Section 3. We have

$$\frac{\partial R(\mathcal{G}_2)}{\partial r_1} = (r_4 - r_4 r_6)(r_7 + r_3 r_5 - r_3 r_5 r_7).$$

Let us present the complete list of formal expressions for the sensitivities evaluated in the point $r_i = r$, $1 \leq i \leq 7$:

$$\begin{aligned}
\frac{\partial R(\mathcal{G}_0)}{\partial r_1} &= r^2(1-r)(2+3r-7r^2+3r^3) = \frac{\partial R(\mathcal{G}_0)}{\partial r_3}, \\
\frac{\partial R(\mathcal{G}_0)}{\partial r_2} &= r^2(1-r)^2(1+4r-3r^2), \\
\frac{\partial R(\mathcal{G}_0)}{\partial r_4} &= r^2(1-r)^2(1+3r-3r^2) = \frac{\partial R(\mathcal{G}_0)}{\partial r_5}, \\
\frac{\partial R(\mathcal{G}_0)}{\partial r_6} &= r(1-r)(1+2r+r^2-6r^3+3r^4) = \frac{\partial R(\mathcal{G}_0)}{\partial r_7}.
\end{aligned}$$

6 Conclusions

In this paper, we pointed out the practical importance of sensitivity analysis in network reliability computations. We showed that it is possible to extend already known algorithms that calculate reliabilities to compute sensitivities as well. Moreover, these byproduct results can be obtained without a signifiant modification of the complexity. More precisely, we explicitly considered several representative methods used in this context and we analyzed the technical problems that are to be solved in order to carry out such an extension.

The exploration of other reliability computation techniques is obviously an immediate possible direction for further research. Another basic extension of this work is to consider more general contexts

or less restrictive assumptions: In particular, the case of directed graphs seems particular important and also the models in which links and/or nodes can fail. In the same way, more general reliability functions can be adressed as those of source–to–K problems or K–terminal problems.

References

[1] M.O.Locks and A.Satyarayana editors. Network reliability – the state of the art. *IEEE Trans. Reliab.*, R-35(3), 1986.

[2] L.G.Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8:410–421, 1979.

[3] J.S.Provan. The complexity of reliability computations in planar and acyclic graphs. *SIAM J. Comput.*, 15(3), Aug. 1986.

[4] S.Hariri and C.S.Raghavendra. Syrel: a symbolic reliability algorithm based on path and cutset methods. *IEEE Trans. on Comput.*, C-36(10):1224–1232, october 1987.

[5] B.E.Barlow and F.Proschan. *Statistical Theory of Reliability and Life Testing.* Holt, Rinehart & Winston, New York, 1975.

[6] Y.Correc. *Vulnérabilité et fiabilité d'un réseau.* Technical Report LA No. 57, CELAR, Mar. 1983. Address: CELAR, 35170 Bruz, France.

[7] J.A.Abraham. An improved algorithm for network reliability. *IEEE Trans. on Reliab.*, R-28(1):58–61, April 1979.

[8] J.B.Fussell. How to hand-calculate system reliability characteristics. *IEEE Trans. Reliab.*, R-24(3), 1975.

[9] A.Satyarayana and A.Prabhakar. New topological formula and rapid algorithm for reliability analysis of comples networks. *IEEE Trans. Reliab.*, R-27:82–100, 1978.

[10] A.Satyarayana. A unified formula for the analysis of some network reliability problems. *IEEE Trans. Reliab.*, R-31:23–32, 1982.

[11] S.Ahmad. A simple technique for computing network reliability. *IEEE Trans. Reliab.*, R-31(1), Apr. 1982.

[12] M.O.Locks. Recursive disjoint products: a review of three algorithms. *IEEE Trans. Reliab.*, R-31, 1982.

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

[14] K.Wood. Factoring algorithms for computing k–terminal network reliability. *IEEE Trans. Reliab.*, R-35(3), 1986.

[15] A.Satyanarayana and R.K.Wood. *Polygon–to–chain Reductions and Network Reliability.* Technical Report ORC 82-4, Operations Research Center, University of California, Berkeley, USA, 1982.

[16] A.Satyanarayana and R.K.Wood. A linear–time algorithm for computing k–terminal in series–parallel networks. *SIAM J. Comput.*, 14(4), Nov. 1985.

[17] A.Satyarayana and M.K.Chang. Network reliability and the factoring theorem. *Networks*, 13, 1983.

[18] K.Wood. A factoring algorithm using polygon–to–chain reductions for computing k–terminal network reliability. *Networks*, 15:173–190, 1985.

PI 604 A GENERAL METHOD TO DEFINE QUORUMS
Mitchell L. NEILSEN, Masaaki MIZUNO, Michel RAYNAL
Septembre 1991, 20 pages.

PI 605 OBTENTION DES EQUATIONS DYNAMIQUES D'UN SYSTEME PHYSIQUE A PARTIR DE SON MODELE BOND GRAPH
Bénédicte EDIBE
Septembre 1991, 26 pages.

PI 606 CONSTRUCTIVE PROBABILITY AND THE SIGNalea LANGUAGE : BUILDING AND HANDLING RANDOM PROCESSES VIA PROGRAMMING
Albert BENVENISTE
Septembre 1991, 60 pages.

PI 607 ABOUT LOGICAL CLOCKS FOR DISTRIBUTED SYSTEMS
Michel RAYNAL
Octobre 1991, 16 pages.

PI 608 UNE NOUVELLE APPROCHE REALISTE DE SIMULATION D'ECLAIRAGE DANS UN ENVIRONNEMENT DIFFUS
Eric LANGUENOU, Kadi BOUATOUCH, Pierre TELLIER
Octobre 1991, 70 pages.

PI 609 INTEGRATION D'UN CORRECTEUR ORTHOGRAPHIQUE DANS L'EDITEUR STRUCTURE GRIF
Patrice FRISON, Eric PICHERAL, Hélène RICHY
Octobre 1991, 22 pages.

PI 610 SYNCHRONIZATION AND CONCURRENCY MEASURES FOR DISTRIBUTED COMPUTATIONS
Michel RAYNAL
Octobre 1991, 20 pages.

PI 611 MALI v06 - TUTORIAL AND REFERENCE MANUAL
Olivier RIDOUX
Octobre 1991, 86 pages.

PI 612 SENSITIVITY COMPUTATION IN NETWORK RELIABILITY ANALYSIS
Gerardo RUBINO
Octobre 1991, 38 pages.

PI 613 OPAC : A FLOATING-POINT COPROCESSOR DEDICATED TO COMPUTE-BOUND KERNELS
André SEZNEC
Karl COURTEL
Octobre 1991, 28 pages.

PI 614 CONTROLLING AND SEQUENCING AN HEAVILY PIPELINED FLOATING-POINT OPERATOR
André SEZNEC
Karl COURTEL
Octobre 1991, 28 pages.

ISSN 0249 - 6399