Network reliability importance measures : combinatorics and Monte Carlo based computations

ILYA GERTSBAKH
Department of Mathematics
Ben Gurion University
P.O.B 653 Beer Sheva 84105
ISRAEL
elyager@bezeqint.net
YOSEPH SHPUNGIN
Department of Software Engineering
Negev Academic College of Engineering and NMCRC
P.O.B 950 Beer Sheva 84100
ISRAEL
yosefs@sce.ac.il

Abstract: - In this paper we focus on computational aspects of network reliability importance measure evaluation. It is a well known fact that most network reliability problems are NP-hard and therefore there is a significant gap between theoretical analysis and the ability to compute different reliability parameters for large or even moderate networks. In this paper we present two very efficient combinatorial Monte Carlo models for evaluating network reliability importance measures.

Key-Words: - Network, Reliability, Importance Measure, Monte Carlo, Combinatorial Approach

1 Introduction

One of the main goals of network reliability analysis is to identify the weak spots in the system and to quantify the impact of component reliabilities and their failures on the network reliability and its failure probability. The so called "reliability importance measures" are used for these purposes. The importance measures provide numerical indicators for determining which components are more important to network reliability improvement, or more critical for system failure. In spite of the fact that several well-known component importance measures exist in literature already many years, there is a wide gap between the theoretical analysis and the ability to compute component importance measures for large or even moderate networks. The reason for having this gap is that all theoretical important measures use an analytic expression for system (network) reliability as a function of its component reliabilities. In practice, obtaining such expressions in a closed form is an impossible task, except for several simple structures, such as series-parallel networks or *k*-out –of-*n* systems. In all other cases the reliability analysis relies mainly on various modifications of the Crude Monte Carlo (CMC) method. The main drawback of CMC is that it is very inefficient in two extreme (and probably most interesting) cases: highly reliable and highly unreliable networks (the so called rare event phenomenon).

Our purpose in this paper is to describe two very efficient Monte Carlo (MC) models for evaluating network reliability importance measures. The common feature for these two models is that the appropriate simulation schemes are *homogeneous*. Let us explain the latter notion in plain words. Consider an *urn U* with a large number of balls *b* in it. Suppose that each ball *b* is marked with some value $z(b)$ and we want to calculate the sum of $z(b)$ over *b* in *U*:

$$Z = \frac{1}{|U|}\sum_{b\in U} z(b) \qquad (1)$$

This completely matches the computation of network reliability. In this case, the balls *b* are the states, and $z(b)$ are defined as 0 for any *Bad* state and equals the probability of the state if it is *Good*. Therefore, *Z* becomes the reliability of the network. Since the number of balls in *U* is very large, the whole sum cannot be computed precisely, and we are forced to *estimate Z* by some MC scheme. We say that MC scheme is homogeneous, if the *balls* are drawn from the *urn* with probability which *does not depend* on the probabilities of the states (more on

homogeneous schemes see in [1]). The important feature of homogeneous schemes is that the relative error is bounded (a basic example of a non-homogeneous scheme is the CMC and its variations).

The paper is organized as follows. In Section 2, we give some basic notions and definitions. In Section 3, we present an efficient computation model [1,2] for evaluating reliability gradient vector, which is used for computing Birnbaum Importance Measure for a general case of networks with non-identical elements. As we mentioned above, our method allows obtaining numerical values of the component important measures **without** deriving the analytic expression for the network reliability function.

In Section 4, we propose a highly efficient spectrum approach [3-5] for networks with identical components. It's worth noting that this approach provides easily implemented computations and allows obtaining, with minimal difficulties, various topological features of the network. Section 5 presents a series of numerical examples.

2 Basic notions and definitions

2.1 Network and its reliability

All networks have *vertices* (*nodes*) and *edges*. There are many types of networks varying in their performance definitions and therefore with different concepts of their reliability. Let *K*-network be an undirected graph $N=(V,E,K)$ with a node-set V, an edge-set E and a set $K \subseteq V$ of special nodes called *terminals*. Also let $|V|=m$ and $|E|=n$. In our model, nodes can never fail, while edges can. Note that all of our results are valid also for the case of reliable edges and unreliable nodes. If an edge fails, we say that it is *down*; otherwise we say it is *up*. By state of a network we call a binary vector $(x_1,..,x_n)$, where each component $x_i=1$ if an edge e_i is *up* and $x_i=0$ otherwise. Denote by N^* the network induced by its state, i.e. with the same node-set V and the edge-set E^* consisting of all edges being *up*. A state of the network N is defined as being *Good* if any two terminals in the induced network N^* are connected by some path of its edges. Otherwise it is *Bad*. The *terminal connectivity* criterion has the property of being monotone: each subset of a *Bad* state is a *Bad* state and each superset of a *Good* state is a *G*ood state.

There are two network reliability models: *static* and *dynamic*. In this paper we restrict our attention to static networks. Each edge e_i is associated with probability p_i of being *up* and a probability $q_i=1-p_i$ of being *down.* We say that edges are identical if they all have the same probability of being *up*, that is for each $i \neq j$ we have $p_i = p_j = p$. We define the network reliability $R=R(p_1,...,p_n)$ as the probability that the network is in a *Good* state.

2.2 Reliability importance measures (IM's)

IM's aim at quantifying the contribution of components to the measure of system performance, which in our case is system reliability. We present below a short description of several prevalent IM's proposed in literature. IM's were first introduced by Birnbaum [6]. The Birnbaum Importance Measure (BIM) of element e_i is defined as

$$I_i^B = \frac{\partial R(p_1,...,p_n)}{\partial p_i} \qquad (2)$$

It expresses the rate of increase of the network reliability with respect to the *i*-th element reliability increase.

Remark. For equal $p_i \equiv p$, first the derivatives $\frac{\partial R}{\partial p_i}$ are computed and only afterwards all p_i set to be equal p.

Fussel and Vesely [7] later proposed another importance measure termed as Fussel-Vesely Importance Measure (FVIM):

$$I_i^{FV} = 1 - \frac{R(p_1,...,p_{i-1},0,p_{i+1},...,p_n)}{R(p_1,...,p_n)} \qquad (3)$$

It quantifies the decrement in system reliability caused by a particular component failure.

Among other popular important measures let us note the following two: CIM and JIM. The Criticality Importance Measure (CIM) [8, 9] is a natural extension of BIM and includes the component unreliability whereas the BIM does not. The three above measures (BIM, FVIM and CIM) are functionally different and measure slightly different properties of system behavior, and one can infer different information from each one of them [10].These IM's are united by the fact that quantify the importance rank of individual components. Contrary to them, the Joint Importance Measures (JIM's) quantify the importance of groups of components. They are very useful and have many applications [11, 12, 13]. Recently they became

popular in risk-informed applications [14, 15]. All the above mentioned IM's are of universal nature with respect to the object of their application. Another direction is a more detailed study of IM's characteristics for specific systems. Recently, the most studied systems were the so called consecutive-k-systems [16, 17].

As a rule, the use of IM's rests on the assumption that we have at our disposal the functional form of system reliability function. The fact that in practice we often don't have this functional description, especially for complex systems, stimulated the interest in developing computational and numerical methods for evaluating IM's, see for example [18, 19, 20, 21]. These works, however, demand computing system reliability parameters. The method we develop in the present paper is based on system structural description and allows components ranking by their IM's without knowing the analytic form of the reliability function.

3 Using Reliability Gradient Vector for BIM evaluation

3.1 Gradient and border states

The numerical evaluation of BIM will be done using some special properties of the reliability gradient vector. In this section we describe a special form of the gradient vector which allows using a highly efficient Graph Evolution Model [2] for its computation. Similar form of the gradient was outlined in [1].

Let us consider a monotone system [22] of n elements. Suppose that each element e_i may be in two states: *up* with probability p_i and *down* with probability q_i. The *state* of a system is defined as a binary vector $(x_1,..,x_n)$, where each component $x_i = 1$ if e_i is *up* and $x_i = 0$ otherwise. All 2^n binary states are divided into two classes: *Good* and *Bad*.

Definition 1. Reliability gradient vector ∇R is defined as $\nabla R = (\frac{\partial R}{\partial p_1},...,\frac{\partial R}{\partial p_n})$, i.e. component i of the reliability gradient vector is the BIM of element e_i.

Definition 2. System state $w = (w_1,...,w_n) \in Bad$ is called *direct neighbor* or simply *neighbor* of state $v = (v_1,...,v_n) \in Good$ if w differs from v in exactly one position (i.e. the Manhattan distance between vectors w and v equals 1). The set of all neighbor states of *Bad* is called the *border set* and is denoted as *DN**. Obviously, $DN^* \subseteq Bad$.

Surprisingly, it turns out that the reliability gradient vector is intimately related to the border states. To reveal this connection, we introduce an *artificial evolution process* on system elements. At $t = 0$ all elements are *down*. Element e_i is "born" after random time $\tau_i \sim \exp(\lambda_i)$, where λ_i is chosen so that the following equality takes place: $p_i = P(\tau_i \le 1) = 1 - e^{-\lambda_i}$. After the "birth", element e_i remains *up* forever. Consider two system states $v = (v_1,...,v_{i-1},0,v_{i+1},...,v_n)$ and $w = (v_1,...,v_{i-1},1,v_{i+1},...,v_n)$. Suppose that at time t the system is in state v. We look for the probability that during a small time interval Δt the system moves from v to w. Obviously, it will happen iff the element e_i is born during this interval, and all other components which are in state 0 will not become alive during the same interval. The first event has probability $\lambda_i \cdot \Delta t + o(\Delta t)$, and the second event has probability $1 - o(\Delta t)$. Then the probability that during $[t, t+\Delta t]$ there will be the transition $v \to w$ equals $\lambda_i \cdot \Delta t + o(\Delta t)$. Let v be a border state of the system, i.e. $v \in DN^*$. Denote by $\Gamma(v)$ the sum of λ_i over the set of all indices i such that $v + (0,...,1_i,...0) \in Good$. Call $\Gamma(v)$ the *flow* from v into *Good*. Formally, $\Gamma(v) = \sum_{\{v \in DN^*,\, v+(0,...,0,1_i,0,...,0) \in Good\}} \lambda_i$. We need two other notations. Let $R(p_1(t),..., p_n(t))$ be the probability that the system is in *Good state* at the instant t. Let $P(v;t)$ be the probability that the system is in state v at time t. Now let us consider the event "the system is in *Good* state at time $t + \Delta t$". This event takes place if at time t the system was already in the *Good* set or at time t it was in one of its border states and went during this interval from a border state to *Good*. All other possibilities which involve more than one transition during $[t, t+\Delta t]$ have probability $o(\Delta t)$. Formally,

$$R(p_1(t+\Delta t),..., p_n(t+\Delta t)) =$$
$$= R(p_1(t),..., p_n(t)) + \sum_{v \in DN^*} P(v;t) \cdot \Gamma(v) \cdot \Delta t + o(\Delta t).$$

Transfer $R(p_1(t),..., p_n(t))$ to the left-hand side, divide both sides by Δt and set $\Delta t \to 0$. We arrive at the following relationship:

$$\frac{dR(p_1(t),..., p_n(t))}{dt} = \sum_{v \in DN^*} P(v;t)\Gamma(v). \quad (4)$$

Now, represent the left-hand side of (4) in an alternative form:

$$\frac{dR(p_1(t),...,p_n(t))}{dt} \quad {}^{n}_{j\ 1}\frac{dR}{dp_j}\ \frac{dp_j(t)}{dt}$$

$$(p_j(t) \quad 1 \quad e^{\ j^t}, q_j(t) \quad e^{\ j^t}) \quad {}^{n}_{j\ 1}\frac{dR}{dp_j} q_j \quad {}_j$$

$$R\ \{q_{1\ 1},...,q_{n\ n}\}. \qquad (5)$$

Comparing (4) and (5) we arrive at the desired relationship between the gradient vector and the border state probabilities:

$$R\ \{q_{1\ 1},...,q_{n\ n}\} \quad {}_{v\ DN^*} P(v;t)\ \ (v) \qquad (6)$$

From the latter formula we can get the expression for BIM of system elements in the following manner. It follows from the above proof that if we set $\{q_{1\ 1},...,q_{n\ n}\}$ equal to $\{0,...,q_{i\ i},...,0\}$, then we arrive at the following important formula:

$$\frac{R}{p_i}\ q_{i\ i} \quad R\ \{0,...,q_{i\ i},...,0\}$$

$${}_{\{v\ DN^*, v\ \ (0,...,1_i,...,0)\ \ Good\}} P(v;1) \quad {}_i \qquad (7)$$

This relationship says that the sum of the probabilities of all border states which make a transition into the *Good* state by "activating" edge e_i is equal, up to a multiplier q_i, to the BIM of component i.

This formula will be the principal tool for the Monte Carlo evaluation of the gradient vector.

Example 1. Let us take the network given in Fig. 1 and compute the BIM for edge e_1. The corresponding border states v such that $v \quad (1,0,0,0) \quad Good$ are: $S_1 \quad (0,1,0,0)$, $S_2 \quad (0,1,1,0)$, $S_3 \quad (0,1,0,1)$. Then by (7) we get:

$$\frac{R}{p_1}\ q_{1\ 1} = \ {}_1\ (P(S_1) \quad P(S_2) \quad P(S_3)) =$$

$${}_1\ (p_2\ q_1\ q_3\ q_4 \quad p_2\ p_3\ q_1\ q_4 \quad p_2\ p_4\ q_1\ q_3).$$

Dividing the both sides of the latter expression by $q_{1\ 1}$, we arrive at the BIM of e_1.

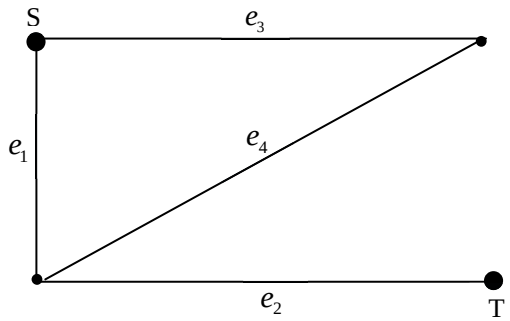

Fig. 1

The above example demonstrates computations via formula (6). It is obvious that the main technical difficulty lies in identifying the border states and finding their probabilities. Computations similar to shown in the above example are difficult to carry out for large or even moderate networks. There exists, however a powerful computational Monte Carlo technique based on introducing a special scheme called Evolution and Merging process, which allows an efficient estimation of expressions of type (6). It was first suggested in the principal work [2].

3.2 Lomonosov's turnip

To make the appropriate Monte Carlo scheme for evaluating BIM more transparent, we give here a short description of the Evolution and Merging process (EMP or Lomonosov's turnip) [1] which will be adopted to our needs. Let us consider the case of unreliable edges and reliable nodes The EMP uses two basic ideas. The first one is introducing an *artificial* random process associated with each edge. The second is defining the *trajectories* of a random process built on network states. Introduce now for each edge an *artificial creation process*, as it was described above in 3.1. Remind that the probability of being *down* at $t_0 \quad 1$ for each edge *coincides* with the *static down*-probability $q(e)$. At $t \quad 0$ the network is in *Bad* set (there are no edges).

Denote by (N) network's "birth" time, i.e. at random instant (N) the network becomes *Good* and remains *Good* forever. Let us denote by $P\{\ (N) \quad 1\}$ the probability that at moment $t_0 \quad 1$ in the creation process the network N is *Good*. Then we have:

$$R(N) \quad P\{\ (N) \quad 1\} \qquad (8)$$

In words: the static probability $R(N)$ that the network is *Good* coincides with the probability that in the edge creation process network state is *Good* at the instant $t_0 \quad 1$, i.e. N enters the *Good* set before or at $t_0 \quad 1$. On Fig. 2 we present the creation process for the network of Fig 1. The operational criterion is the two terminal connectivity with terminal nodes *S* and *T*.

Initially all edges are in the *down* state (the zero level of the turnip). The first level shows all possible evolution results from the zero level which appear as a result of a birth of a single edge. There are four such states. The second level of the turnip shows what happens when a second edge is born. We

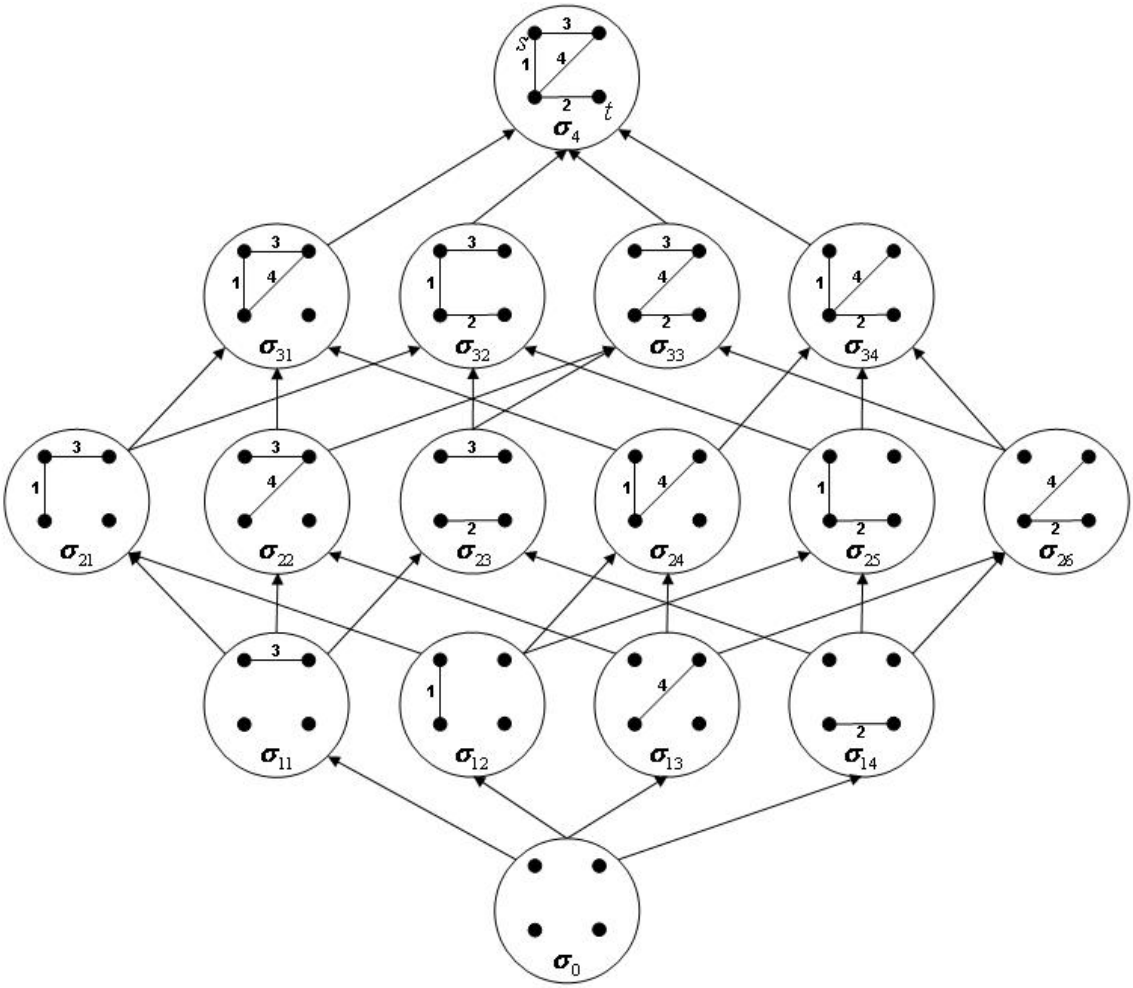

Fig. 2

distinguish six states, two of which are border states. At the second level, one of the states is *Good* and five others are border states. At the third level one of the states is a border state and the other three states belong to *Good*.

An important feature of the turnip is that simulating the transitions from one state to the following is very easy. Compute, for example, the probability $P(\sigma_{11} \to \sigma_{21})$. Let the birth rate of edge *k* be $\lambda(k)$, $k = 1,...4$. Then, by the well known property of the exponential distribution, we have:

$$P(\sigma_{11} \to \sigma_{21}) = \frac{\lambda(2)}{\lambda(2)+\lambda(3)+\lambda(4)} = \frac{\lambda(\sigma_{11})-\lambda(\sigma_{21})}{\lambda(\sigma_{11})}, \quad (9)$$

where $\lambda(\sigma_{ij})$ denotes the total birth rate for the state σ_{ij}. Indeed, the transition $\sigma_{11} \to \sigma_{21}$ takes place if and only if edge 2 is born out of three possibilities of the birth of edges 2, 3 or 4. Actually, the "turnip" diagram describes an artificial creation process, and the probability that the network in this process is in *Good* set at some moment t_0 coincides with the corresponding static probability of the network. For a more detailed description of the evolution process see [1,2].

Remark. The description of the creation process is given here in an elementary form, without using the *closure* operation. Closure [1] is one of most important advantages of the approach. Roughly speaking, the closure of some state *E* is an union of the state itself and all *irrelevant* edges (for given operational criterion). For example, in Fig. 2 the edge 4 is irrelevant for the state σ_{21} since the nodes associated with edge 4 are already connected by a path formed by edges 1 and 3. Therefore, the set σ_{31} is the closure for the state σ_{21} (and also for the states σ_{22} and σ_{24}). Using the operation of closure allows introducing so called *super-states* instead of the "microscopic" states, which extremely

accelerates the evolution process on turnip. For instance, if we have a network with 100 vertices and 2000 edges, using the super-states will result in a trajectory of maximal length 99. Without the closure operation the maximal length can be up to 2000.

Consider a random process $\sigma(t)$ whose states are the states of the described creation process. It was proved in [1] (for the super-states too) that

(i) $\sigma(t)$ is a Markov process;

(ii) The time spent by $\sigma(t)$ in a particular state σ' is distributed as $\exp(\lambda(\sigma'))$.

Define now a *trajectory*, which plays the central role in the described process and in the corresponding Monte Carlo scheme.

Definition 3. A trajectory is a sequence $u=(\sigma_0,\sigma_1,...,\sigma_r)$ of states such that σ_0 is the initial trivial state, each σ_i is the direct successor of σ_{i-1}, and σ_r is the first state belonging to *Good*. For example, the sequence $(\sigma_0,\sigma_{11},\sigma_{21},\sigma_{31},\sigma_4)$ on the Fig. 2, is a trajectory.

Now, in terms of the trajectories, the network is *Good* at moment *t* if there exists at least one trajectory that reaches *Good* before *t.* It is easy to calculate the probability *p*(*u*) of the trajectory u:

$$p(u)=\prod_{i=0}^{r-1} P(\sigma_i \to \sigma_{i+1}) \ . \qquad (10)$$

Denote by $P(t\,|\,u)$ the probability that the *Good* state will be reached before time *t* given that the creation process goes along trajectory *u.* Then by the above mentioned property, the process is sitting in each state σ_j an exponentially distributed random time $\tau(\sigma_j)$. Due to the Markov property, the total time along the trajectory *u* is a sum of the respective independent random variables, and

$$P(t\,|\,u)=P(\tau(\sigma_0)+...\tau(\sigma_{r-1})\le t\,|\,u). \qquad (11)$$

Note that $P(t\,|\,u)$ is a convolution and can be computed analytically [1] or simulated [23]. Now the probability that at moment *t* the creation process reaches *Good* equals:

$$R(N)=P(\xi(N)\le t)=\sum_{u\in U} p(u)\cdot P(t\,|\,u), \qquad (12)$$

where *U* is the set of all trajectories. The expression (12) has the form of an expectation and opens the way to estimating $R(N)$ by means of a Monte Carlo algorithm described in [1,2].

3.3 Numerical evaluation of BIM's

Let us now use the turnip for evaluating BIM. Remind that by the formula (6) we have

$$\nabla R\cdot\{q_1\lambda_1,...,q_n\lambda_n\}=\sum_{v\in DN^*} P(v)\Gamma(v)\,,$$

or using the turnip notations :

$$\nabla R\cdot\{q_1\lambda_1,...,q_n\lambda_n\}=\sum_{\sigma_j\in DN^*} P(\sigma_j)\Gamma(v)\,, \qquad (13)$$

where $\Gamma(v)=\Sigma_{\{\sigma_j\in DN^*,\,v+(0,...,0,1_i,0,...,0)\in UP\}}\lambda_i$. Now, for each border state σ we have following formula:

$P(\sigma)=P(\xi(\sigma)\le 1)-P(\xi(N)\le 1).$

Here the first term is the probability that the creation process is at $t_0=1$ in the state σ or is in *Good*. The second term is the probability that the *Good* state was reached before $t_0=1$. The difference is therefore the desired probability that at $t_0=1$the process is in σ. Considering all trajectories leading from σ_0 into *Good*, we obtain the following formula:

$$P(\sigma)=\sum_{u\in U} p(u)(P(\xi(\sigma)\le t_0\,|\,u)-P(\xi(N)\le t_0\,|\,u).$$

Substituting the latter expression into (13), we get:

$$\nabla R\cdot\{q_1\lambda_1,...,q_n\lambda_n\}=\sum_{\sigma_j\in DN^*} P(\sigma_j)\Gamma(v)=$$

$$\sum_{\sigma_j\in DN^*}\sum_{u\in U} p(u)(P(\xi(\sigma)\le t_0\,|\,u)-P(\xi(N)\le t_0\,|\,u)\Gamma(\sigma_j).$$

Changing the order of summation we arrive at the following formula:

$$\nabla R\cdot\{q_1\lambda_1,...,q_n\lambda_n\}= \qquad (14)$$

$$\sum_{i=1}^{m}\lambda_i\Big(\sum_{u\in U \text{ such that } e_i\in\sigma(u)^+} p(u)\cdot f(\sigma,u),$$

where $\sigma(u)$ is the border state defined by the trajectory u, $\sigma(u)^+$ is a set of all edges which "transfer" $\sigma(u)$ to the *Good* state and $f(\sigma,u)=$ $P(\xi(\sigma)\le t_0\,|\,u)-P(\xi(N)\le t_0\,|\,u)$.

The sum in (14) has the form of the expectation, which is the key for Monte Carlo numerical procedures on the turnip. We use the following simulation scheme to evaluate BIM's values simultaneously for all network edges. Define by ArB an array of size *n* so that ArB[*i*] will denote the BIM of edge *i.*

Simulation scheme.

Step1. Put $ArB[i]=0,\ 1\le i\le n.$

Step2. Generate trajectory *u* leading from the trivial state to the state in *Good.*

Step3. Let σ_j be a border state on this trajectory and let edge $e_i\in\sigma_j(u)^+$. Then

$$ArB[i]=ArB[i]+(P(\xi(\sigma_j)\le t_0\,|\,u)-P(\xi(N)\le t_0\,|\,u).$$

Step4. Repeat steps 2 and 3 *K* times.

Step5. For all *i* ($1\le i\le n$) put $ArB[i]=ArB[i]/K.$

(For the calculation of the difference of two

convolutions (Step 3) see [1] and [11]).

4 Spectral approach to computing network reliability importance measures

In this section we will derive the BIM and the FVIM for networks with identical elements by means of so called network combinatorial spectrum. This notion was introduced in [3] and [4,5] to estimate network lifetime distribution and /or its static reliability. For reader's convenience we remind shortly the principal idea of network combinatorial characteristic called spectrum. For simplicity we demonstrate the method for the case of reliable nodes and unreliable edges. It was shown in [4] that this approach is applicable also to the case of reliable edges and unreliable nodes, or – which is more complicated – to the case of both unreliable nodes and edges.

Let Π_E be the set of all edge permutations in *E*. Let π be a particular permutation. By sub-permutation $\pi(1...i)$ of π we denote a sequence constructed of the first i edges in π. For each sub-permutation $\pi(1...i)$ we define a network state $S(\pi(1...i))$, where all the edges in $\pi(1...i)$ are *up* and all other edges in π are *down.* For each edge e_j and permutation π denote by $\pi(e_j)$ the index of this edge in π.

Example 2. Let us take the network in Fig.1 and let our permutation be $\pi = (1,3,2,4)$. Then, for example, $\pi(1...3) = (1,3,2)$ and $S(\pi(1...3))$ is a state in which edges 1,3,2 are up, and edge 4 is down. We have also: $\pi(e_1) = 1$, $\pi(e_2) = 3$, $\pi(e_3) = 2$, $\pi(e_4) = 4$.

Next we define an *anchor*. This notion plays a central role in our reasoning.

Definition 4. Let $r = r(\pi)$ be the first index in permutation π so that $N(\pi(1...r))$ is *Good.* We say that $r(\pi)$ is the anchor of the permutation π.

Definition 5. Denote by x_i the number of all permutations π such that i is the *anchor* of π. We say that the set

$$SP = \{\{x_i\}, 1 \le i \le n\} \qquad (15)$$

is the *combinatorial spectrum* of the network.

Example 3. We demonstrate these definitions on a network given in Fig. 1. The total number of permutations of 4 edges in the network is 24. Let $\pi = (3,1,2,4)$. We see that the first index such that the network state becomes *Good* is 3. Therefore $r(\pi) = r(3,1,2,4) = 3$ is the anchor of this permutation. After going over all permutations we arrive at the following combinatorial spectrum of the given network:

i	1	2	3	4
x_i	0	4	14	6

It was shown in [4] that given a network spectrum $SP = \{\{x_i\}, 1 \le i \le n\}$, the network reliability may be expressed in the following form:

$$R = \sum_{r=1}^{n} x_r \sum_{i=r}^{n} \frac{p^i \cdot q^{n-i}}{i! \cdot (n-i)!} \qquad (16)$$

In our example, the network reliability is:

$$R = \sum_{r=1}^{4} x_r \sum_{i=r}^{4} \frac{p^i \cdot q^{n-i}}{i! \cdot (4-i)!} = p^4 + 3p^3 \cdot q + p^2 \cdot q^2 .$$

Remark. Sometimes it is more convenient to use the *cumulative* form of the spectrum:

$$SP^* = \{y_i : y_i = \sum_{k=1}^{i} x_i, 1 \le i \le n\} \qquad (17)$$

The value y_i expresses the number of permutations π such that $r(\pi) \le i$, or, in other words, that $N(\pi(i))$ is *Good.*

Example 4. For the network on Fig. 1, we have from the above example that there are 14 permutations with anchor $r = 3$ and 4 permutations with anchor $r = 2$. So, we get $y_3 = 18$.

It is easy to check from (16) that in the case of the cumulative spectrum the network reliability is given by

$$R = \sum_{i=1}^{n} y_i \cdot \frac{p^i \cdot q^{n-i}}{i! \cdot (n-i)!} \qquad (18)$$

Clearly, in the case of large or moderate networks we can not get the exact values of the spectrum. We can however try to estimate them by a Monte Carlo simulation [3,4]. It is worth to mention the main advantages of this combinatorial approach:

(a) eliminating the rare event phenomenon. This fact results in bounding the relative error, so the method is especially efficient for highly reliable networks.

(b) once computed, the *combinatorial spectrum* serves for as many values of nodes or edge failure probabilities as needed.

(c) possibility to use for solving different reliability problems in dynamic networks.

Definition 6. Denote by $z_{i,j}$ the number of all permutations π such that $S(\pi(1...i))$ is *Good* and $\pi(e_j) \le i$. We call the set $\{z_{i,j}, 1 \le i \le n, 1 \le j \le n\}$ - the *BIM spectrum.*

Definition 7. Denote by $v_{i,j}$ the number of all permutations π such that $S(\pi(1...i))$ is *Good* and

$\pi(e_j) > i$ .We call the set $\{v_{i,j}, 1 \le i \le n, 1 \le j \le n\}$ - the *FVIM spectrum.*
We see from these definitions that $z_{i,j} + v_{i,j} = y_i$.

i	y_1	$z_{i,1}$	$z_{i,2}$	$z_{i,3}$	$z_{i,4}$
1	0	0	0	0	0
2	4	4	4	0	0
3	18	12	18	12	12
4	24	24	24	24	24

Table 1

Example 5. Let us take the edge e_1 from the network in Fig. 1 and let us compute $z_{3,1}$. It is easy to see that there are 6 permutations π such that $S(\pi(1...3))$ is *Good* and $\pi(e_1) > 3$. So we get $y_3 - z_{3,1} = 6$. From the previous example, $y_3 = 18$. Hence, we get that $z_{3,1} = 12$. The BIM spectrum for the network is given in Table 1.

Claim 1.
(a) The BIM for edge *j* is given by the following formula:

$$I_j^{\,B} = \sum_{i=1}^{n} \frac{z_{i,j} \cdot p^{i-1} \cdot q^{n-i} - (y_i - z_{i,j}) \cdot p^i \cdot q^{n-i-1}}{i! \cdot (n-i)!} \qquad (19)$$

(b) The FVIM for edge *j* is given by the following formula:

$$I_j^{\,FV} = 1 - \frac{1}{R(p)} \sum_{i=1}^{n} \frac{v_{i,j} \cdot p^i \cdot q^{n-i-1}}{i! \cdot (n-i)!} \qquad (20)$$

Proof. (a) Remind that BIM for edge e_j equals

$$\frac{\partial R}{\partial p_j} = R(p_1, ...1_j, ..., p_n) - R(p_1, ..., 0_j, ..., p_n)$$

(see[12]).
The value $z_{i,j}$, by the definition 5, is the number of permutations π such that $S(\pi(1...i))$ is *Good* and the edge e_j is *up.* For fixed permutation π the probability of an appropriate state with e_j being *up,* equals $p^{i-1} \cdot q^{n-i}$. Take into account that a specific state with *i* edges being *up* and *n-i* edges being *down* we obtain $i! \cdot (n-i)!$ times (from different permutations). Then the summary probability of all *Good* states with *i* edges being *up* and *n-i* edges being *down* equals $\frac{z_{i,j} \cdot p^{i-1} \cdot q^{n-i}}{i! \cdot (n-i)!}$. For the case of the edge e_j being *down* we get the expression of the appropriate probability as $\frac{(y_i - z_{i,j}) \cdot p^i \cdot q^{n-i-1}}{i! \cdot (n-i)!}$, and (a) follows.
(b) Using (a), the definition of IS^{FV} and the above mentioned fact that $z_{i,j} + v_{i,j} = y_{i,j}$ we arrive at the desired expression.
In order to rank the elements according to their importance measure there is no need to compute the partial derivatives. The following simple claim takes place.

Claim 2. Let $\{z_{ij}, 1 \le i \le n\}$ and $\{z_{is}, 1 \le i \le n\}$ be the BIM spectrum elements for the edges e_j and e_s respectively. Then:
(a) If for all $1 \le i \le n$ the inequality $z_{ij} \ge z_{is}$ holds, then $\frac{\partial R}{\partial p_j} \ge \frac{\partial R}{\partial p_s}$. Moreover, if for at least one index *i* a strong inequality $z_{ij} > z_{is}$ holds, then $\frac{\partial R}{\partial p_j} > \frac{\partial R}{\partial p_s}$.
(b) Suppose that the condition of (a) does not take place. Then let k be the maximal index such that $z_{ij} \ne z_{is}$. Suppose that $z_{kj} > z_{ks}$. Then there exists some value p_0 such that for all $p \ge p_0$ the inequality $\frac{\partial R}{\partial p_j} > \frac{\partial R}{\partial p_s}$ holds.

Proof. (a) From (12) we have:

$$\frac{\partial R}{\partial p_j} - \frac{\partial R}{\partial p_s} = \sum_{i=1}^{n} \frac{z_{i,j} \cdot p^{i-1} \cdot q^{n-i} - (y_i - z_{i,j}) \cdot p^i \cdot q^{n-i-1}}{i! \cdot (n-i)!} -$$

$$\sum_{i=1}^{n} \frac{(z_{i,j} - z_{i,s}) \cdot p^{i-1} \cdot q^{n-i} - (z_{i,s} - z_{i,j}) \cdot p^i \cdot q^{n-i-1}}{i! \cdot (n-i)!} =$$

$$\sum_{i=1}^{n} \frac{(z_{i,j} - z_{i,s}) \cdot p^{i-1} \cdot q^{n-i-1}}{i! \cdot (n-i)!}$$ and (a) follows.

(b) From the definition of *k* and the latter expression we obtain:

$$\sum_{i=1}^{k} \frac{(z_{i,j} - z_{i,s}) \cdot p^{i-1} \cdot q^{n-i-1}}{i! \cdot (n-i)!} =$$

$$p^{k-1} \cdot q^{n-k-1} \cdot (\sum_{i=1}^{k} \frac{(z_{i,j} - z_{i,s})}{i! \cdot (n-i)!} \cdot (\frac{q}{p})^{k-i})$$ and for $p \to 1$, the assertion follows.
We use the following Monte Carlo scheme to obtain unbiased estimates for the y_i , $z_{i,j}$ and $v_{i,j}$.

Simulation scheme.
Step1. Initialize all a_i , $b_{i,j}$ and $c_{i,j}$ to be 0.
Step2. Simulate the permutation $\pi \in \Pi$.
Step3. Find $r = r(\pi)$ - the minimal index of edge in π so that the state $N(\pi(1...r))$ is *Good.*

Step4. Let $a_r := a_r + 1$.

Step5. For all j such that $\pi(e_j) \le r$ let $b_{r,j} := b_{r,j} + 1$.

Step6. For all s such that $\pi(e_s) > r$ let $c_{r,s} := c_{r,s} + 1$.

Step7. Let $r := r + 1$. If $r \le n$ Go to Step4.

Step8. Repeat steps 2-7 M times.

Computing $\hat{y}_i = \frac{a_i \cdot n!}{M}, \hat{z}_{i,j} = \frac{b_{i,j} \cdot n!}{M}, \hat{v}_{i,j} = \frac{c_{i,j} \cdot n!}{M}$ we can from (18), (19), (20) obtain, respectively, the unbiased estimates for R, I^B and I^{FV}.

5 Numerical examples

In this section we present several examples, which explain how we can rank network elements (edges and nodes) in accordance to their BIM's by using spectrum approach. For this purpose we choose some hypercubes (note that hypercube configurations are widely used in computer networks [24]).

Example 6. Consider a four-terminal hypercube H_4 with 2^4 =16 nodes and 32 edges. This hypercube is shown on Fig. 2. Let its terminals be nodes 1, 8, 9 and 16. Edges will be denoted by (*k*,*s*) , where *k* and *s* are node numbers.

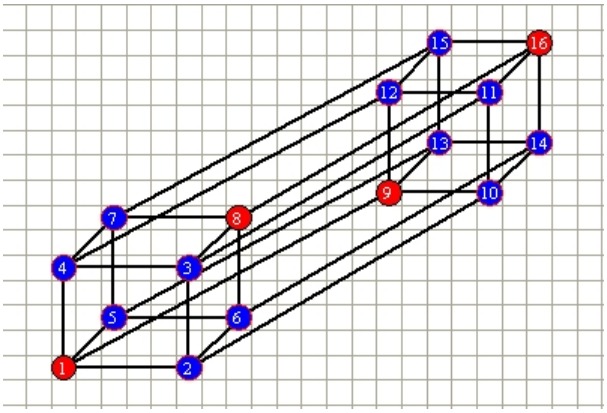

Fig. 3

Table 2 presents a fragment of simulation results, based on 10^4 replications. By a_i we marked the simulated value of spectrum and by $b_{i,(k,s)}$ - the simulated value of BIM spectrum IS^B for edge (*k*,*s*). Remind that for edge ranking there is no need to compute their BIM's. We see from Table 2 that the values of IS^B for edges (1,9) and (8,16) are very close to each other. On the other hand, these values are consistently greater than the BIM spectrum values for edge (1,2) and the latter – than those of (3,4). So, we rank the edges by their importance in the following order (read table 2 in horizontal direction) $(1,9) = (8,16) > (1,2) > (3,4)$.

i	a_i	$b_{i,(1,9)}$	$b_{i,(8,16)}$	$b_{i,(1,2)}$	$b_{i,(3,4)}$
6	2	**2**	**1**	0	1
7	15	**14**	**14**	2	3
8	59	**51**	**50**	12	12
9	154	**129**	**128**	40	38
10	350	**286**	**267**	109	91
11	679	**501**	**492**	235	206
12	1333	**886**	**904**	525	438
13	2385	**1478**	**1492**	996	892
14	3723	**2187**	**2210**	1625	1547
15	5230	**3012**	**3042**	2502	2363

Table 2

Note that from the whole data array (not presented here) one can infer that in our network there are three, by their BIM's, different groups of edges. The first consists of edges (1,9) and (8,16), which connect the pairs of terminals. The second is the group consisting of all other edges incident to one of the terminals 1, 8, 9 and 16, and the third – all other edges. The edges within each group have equal BIM's, the first group dominates the second, and the second dominates the third. This conclusion may seem to be intuitively obvious, but for the same hypercube with nonsymmetrical terminals, or for other nonsymmetrical networks, similar conclusions are not so obvious. More involved cases may need to use special statistical analysis for better discrimination between edges with close BIM's spectra..

Example 7. Consider now the same hypercube, but with three terminals: 1, 10 and 16. Table 3 presents a fragment of simulation results, based on 10^4 replications. The notation is the same as in the Table 2. This case is less obvious due to the non-symmetrical terminals. We can, by the results, rank the edges in the following order: (1,9)>(9,10)>(8,16)>(3,4)>(7,8). (A simulation run with 100,000 replications leads to the same conclusion). An interesting feature of the data in Table 3 is that here we can not unite in one group (as it was in Example 6) all edges incident to one of the terminals. For example, edge (1,9) is ranked higher than the edge (9,10). Note also that there exist edges which are not incident to any terminal and do not belong to one rank group (this was not

the case of the previous example). Such edges, for example, are (3,4) and (7,8).

i	a_i	$b_{i,(1,9)}$	$b_{i,(9,10)}$	$b_{i,(8,16)}$	$b_{i,(3,4)}$	$b_{i,(7,8)}$
6	13	11	11	2	0	2
7	41	33	29	8	5	4
8	99	78	72	37	23	20
9	193	163	159	93	79	68
10	322	311	310	208	167	150
11	547	583	553	441	356	320
12	795	970	923	818	660	614
13	1195	1576	1527	1416	1173	1123
14	1351	2288	2234	2151	1845	1777
15	1435	3027	3033	2969	2681	2590

Table 3

Example 8. Our initial hypothesis was that the type of inequality between BIM's of two elements remains unchanged for all values of *i* (see Claim 2(a)). We'll see from the following example that it is not so. Consider the same hypercube H_4 with three terminals: 1, 10, 16, but with **reliable** edges and **unreliable nodes.** (Remind that the terminal nodes are supposed to be reliable). Table 4 presents the results (also based on 10^4 replications). As it is seen from this table, the nodes 2, 9 and 15 are ranked as follows: (2) = (9)>15. It is interesting that all these three nodes are the "neighbors" of terminals, but the node 15 is less important than the first two.

i	a_i	$b_{i,(2)}$	$b_{i,(9)}$	$b_{i,(15)}$	$b_{i,(7)}$	$b_{i,(6)}$
6	1161	800	805	208	140	208
7	2045	1712	1718	964	837	812
8	2483	2926	2926	2307	2088	1986
9	2052	4161	4165	3821	3569	3497
10	1097	5200	5204	5038	4889	4871
11	450	6091	6088	6028	5970	5964
12	158	6912	6904	6894	6909	6879
13	49	7696	7694	7688	7694	7692
14	0	8460	8460	8458	8465	8459
15	0	9232	9228	9227	9230	9228

Table 4

Let us now pay attention to the last two columns, related to nodes 7 and 6. We see that $b_{1,(7)} < b_{1,(6)}$, but for all $i > 1$ the inequalities become reversed. We have checked this fact by increasing the number of simulation replications and got the same results. We have also combinatorial explanation: the number of short paths is greater for node 6, but the number of longer paths is greater for node 7. Here the conditions of the Claim 2(b) hold and therefore starting with some p_0, node 7 is more important than node 6.

Example 9. One of the important characteristics of Monte Carlo computations is the convergence rate of parameter estimates to their true values. Numerous simulation experiments reveal that the estimates of component importance rapidly become stable and change very little as the number of simulation runs increases. For example, BIM's were estimated for a hypercube H_7 with 128 nodes (5 of them were terminals) and 448 edges . Nodes are subject to failures and edges are reliable. Table 5 presents the estimated BIM's spectrum for node 2, divided by number of replications *K*, for $K=10^4$, 10^5, 10^6 ,see columns 2,3,4, respectively.

i	$b_{i,(2)}/10^4$	$b_{i,(2)}/10^5$	$b_{i,(2)}/10^6$
20	0	0	0.00005
30	0.01500	0.01610	0.01854
40	0.13500	0.13270	0.13559
50	0.29900	0.28900	0.29929
60	0.41300	0.41660	0.42126
70	0.51200	0.51400	0.51752
80	0.60100	0.60650	0.60722
90	0.69400	0.68610	0.69079
100	0.76500	0.77090	0.77191
110	0.84400	0.85160	0.85211
120	0.93800	0.93430	0.93405
125	0.97500	0.97580	0.97583

Table 5

6 Conclusions

(1) We have developed a new method of computing network component importance measures, for networks with different component reliabilities. (We deal with Birnbaum importance measure-the BIM). Our method is based on a connection between the so-called border state probabilities and the reliability gradient vector. The network component importance measures are the coordinates of the reliability gradient vector. Using a specially constructed evolution process on the network components, we have developed an efficient numerical procedure for estimating the border state probabilities and obtaining, therefore, the BIM's. Our numerical procedure avoids the rare event phenomenon and

allows obtaining all gradient coordinates in one simulation run.

(2) For the case of networks with equally reliable components (nodes, or edges or both), we have developed a new efficient computation scheme for evaluating the BIM's. Its main idea is based on calculating, via a Monte Carlo scheme, network combinatorial invariant called spectrum and on the connection between the spectrum and the component BIM's.

Spectrum based BIM computations allow obtaining results without calculating the gradient vector.

(3) The techniques developed in this paper can be viewed as a first step toward developing an algorithm for optimal network reliability design, since the latter has to be based on computing network reliability gradient vector.

References:

[1] T. Elperin, I. Gertsbakh, M. Lomonosov, An evolution model for Monte Carlo estimation of equilibrium network renewal parameters, Probability in the Engineering and Informational Sciences, Vol. 6, 1992, pp. 457-469.

[2] T. Elperin, I. Gertsbakh, M. Lomonosov, Estimation of network reliability using graph evolution models, IEEE Transactions on Reliability, Vol. 40, No.5, 1991, pp. 572-581.

[3] Gertsbakh and Y. Shpungin, Combinatorial approaches to Monte Carlo estimation of network lifetime distribution, Appl. Stochastic Models Bus. Ind., Vol 20, 2004, pp. 49-57.

[4] Y. Shpungin, Combinatorial Approach to Reliability Evaluation of Network with Unreliable Nodes and Unreliable Edges, International Journal of Computer Science, Vol.1, No.3, 2006, pp. 177-183.

[5] Y. Shpungin, Networks with unreliable nodes and edges: Monte Carlo lifetime Estimation, International Journal of Applied Mathematics and Computer Sciences,
Vol. 4, No. 1, 2007, pp. 168-173.

[6] Z. W. Birnbaum, On the importance of different components in a multicomponent system, Multivariate Analysis 2, New York, Academic Press, 1969.

[7] J. Fussel, How to calculate system reliability and safety characteristics, IEEE Transactions on Reliability, Vol. 24, No.3, 1975, pp. 169-174.

[8] F. C. Meng, Comparing the importance of system elements by some structural characteristics, IEEE Transctions on Reliability, Vol 45, No 1, 1996, pp. 59-65.

[9] L. M. Leemis, Reliability – Probabilistic models and Statistical Methods, Prentice Hall, Inc, 1995.

[10] J. F. Espiritu, D. W. Coit and U. Prakash, Component Criticality Importance Measures for the Power Industry, Electric Power Systems Research, Vol. 77, Issues 5-6, 2007, pp. 407-420.

[11] E. Zio and L. Podofillini, Accounting for components interactions in the differential importance measure, Reliability Engineering & System Safety, Vol. 91, Issues 10-11, 2006, pp. 1163-1174.

[12] J.S. Hong and C. H. Lie, Joint reliability-importance of two edges in an undirected network, IEEE Trans. on Reliab., Vol. 42, No.1, 1993, pp. 17-23.

[13] M. J. Armstrong, Joint reliability-importance of elements, IEEE Trans. on Reliab., Vol. 44, No. 3, 1995, pp 408-412.

[14] Bogronovo E., Apostolakis G. E., A new importance measure for risk-informed decision making, Reliab. Eng. And Sys. Safety, Vol. 72, 2001, pp. 193-212.

[15] Youngblood R. W., Risk significance and safety significance, Reliab. Eng. And Sys. Safety, Vol. 73, 2001, pp. 121-136.

[16] Hsun-Wen Chang, Jun-Da Chen, Joint Structural Importance in Consecutive-k-Systems, Proceedings of 7th WSEAS International Conference on Applied Computer Science, 2007, pp. 95-100.

[17] H. W. Chang, R. J. Chen and F. K. Hwang, The structural Birnbaum importance of consecutive-k systems, Journal of Combinatorial Optimization, Vol. 6, 2002, pp. 183-197.

[18] Yung-Ruei Chang, Amari S.V, Sy-Yen Kuo, OBDD-based evaluation of reliability and importance measures for multistate systems subject to imperfect fault coverage, IEEE Transactions on Dependable and Secure Computing, Vol. 2, Issue 4, 2005, pp. 336-347.

[19] Bogronovo E., Differential, criticality and Birnbaum importance measures: An applicationn to basic event, groups and SSCs in event trees and binary decision diagrams, Reliability Engineering & System Safety, Vol. 92, Issue 10, 2007, pp. 1458-1467.

[20] Hae Sang Lee, Chang Hoon Lie, Jung Sik Hong, A computation method for evaluating importance measures of gates in a fault tree,

IEEE Transactions on Reliability, Vol. 46, Issue 3, 1997, pp. 360-365.

[21] G. Rubino, Sensitivity analysis of network reliability using Monte Carlo, Proceedings of the 37th conference on Winter simulation, Orlando, Florida, 2005, pp. 491-498.

[22] I. Gertsbakh, Reliability Theory with Applications to Preventive Maintenance, Springer, 2000.

[23] I. Gertsbakh and Y. Shpungin, Product-Type Estimators of Convolutions, Semi-Markov Models and Applications, Kluwer Academic Publishers, 1999, pp. 201-206.

[24] M. Mitzenmacher, E. Upfal, Probability and Computing . Randomized Algorithms and Probabilistic Analysis, Cambridge University Press, 2005.