International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:1, No:3, 2007

Networks with Unreliable Nodes and Edges: Monte Carlo Lifetime Estimation

Y. Shpungin

***Abstract*—**Estimating the lifetime distribution of computer networks in which nodes and links exist in time and are bound for failure is very useful in various applications. This problem is known to be NP-hard. In this paper we present efficient combinatorial approaches to Monte Carlo estimation of network lifetime distribution. We also present some simulation results.

***Keywords*—**Combinatorial spectrum, Monte Carlo, Network lifetime, Unreliable nodes and edges.

I. Introduction

IN this paper we consider efficient combinatorial Monte Carlo (MC) approaches for estimating network lifetime distribution. It is a well known fact that most network reliability problems are NP-hard and therefore only using Monte Carlo methods will produce computational results for large networks. Most MC applications are actually Crude Monte Carlo (CMC) applications, which are very inefficient and have a major drawback. Their drawback is the unbounded growth of the relative error as a result of increase in reliability. A detailed description of the CMC scheme and essentially not-CMC scheme with its advantages can be found in [22]. We skip a detailed survey of the background and only note several main research directions from our point of view.

- Algorithms for reliability computation [1]-[5].
- Reliability estimation by means of simulation [6]-[17], [22].
- Constructing tractable lower and upper bounds on the network reliability [18]-[20].

In this paper we develop a combinatorial MC approach for a network with unreliable nodes and edges, which is more complicated and realistic than the common case of either unreliable nodes or edges. Our approach utilizes networks combinatorics, which enhances the performance of these methods.

The rest of this paper is organized as follows. In section II, we give an MC schema of lifetime distribution function estimation for a network with non-identical nodes and non-identical edges. This schema is a generalization of a method developed in [17] for the case of a network with unreliable edges. The main tool of this method is the use of Lemma 1, which extremely accelerates the simulation process. In section III, we consider a network with identical nodes and identical edges. In this case, we use a very effective two-dimensional spectrum, proposed in [22]. In section IV, we give an approximation formula for dynamic network reliability function based on famous Burtin-Pittel theorem [23]. Some simulation results are given in section V.

Manuscript received March 10, 2007.

Y. Shpungin is with Negev Monte Carlo Research Center (NMCRC) and Department of Software Engineering, Sami Shamoon College of Engineering (SCE), Beer Sheva, 84100, Israel (e-mail: yosefs@sce.ac.il).

A. Basic Notations and Definitions

All networks have *vertices* (*nodes*) and *edges*. There are many types of networks varying on their performance definitions and therefore with different concepts of 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 $|V| = n$ and $|E| = m$. A node which is a terminal will be called a *terminal*, while other nodes will be referred to as *nodes.*

In our model, terminals can never fail, while nodes and edges can. If an element (either node or edge) fails, we say that it is *down*; otherwise we say it is *up*. A state of a network, (X, Y), where $X \subseteq E, Y \subseteq V$ is defined as being *Good* if any two terminals are connected by a path consisting of edges from X and nodes from *Y.* Otherwise it is *Bad*.

There are two network reliability models: *static* and *dynamic*. The standard static network reliability problem is defined as follows; assume that edges and nodes fail independently. By p_e and p_v we denote the respective probabilities if edge *e* and node *v* not being *up*. We wish to compute $R(N) = P$ (the network N is in the *Good* state).

There are two variants of the dynamic network model. The first type is when the network elements (both nodes and edges) are nonrenewable. At *t=0* each element is *up.* Edge *e* fails at time τ_e and node v fails at time τ_v. The lifetime of the network τ^* is defined as the instant at which the network becomes *Bad*.

The second type is when the network elements are renewable. This time, each element behaves independently according to a two-state Markov process. The network lifetime, by definition, is the first instant when the state of the network becomes *Bad* (assuming the initial state was *Good*). Similarly it can be defined as the first instant when the network becomes *Good* (assuming the initial state was *Bad*).

The common ground for investigation of the networks above in the case of unreliable nodes and edges is the use of combinatorial approaches developed in [8]-[9], [14]-[17] and [21]-[22] (most of which are for static networks). In this paper we restrict our attention to the first type of dynamic networks.

II. Lifetime Estimation for Network with Non-Identical Edges and Non-Identical Nodes

The purpose of this section is to estimate the network lifetime, i.e. estimate the function $F_N(t) = P(\tau^* \le t)$. Recall that at $t = 0$ all elements are *up*. The network fails if at least one of the terminals gets disconnected from the others. The straightforward estimation of the function $F_N(t) = P(\tau^* \le t)$ may be carried out as follows.

A. Simulation Scheme 1

Step1. Simulate element according to its lifetime distribution: edge *e* with $F_e(t)$ and node *v* with $F_v(t)$.

Step 2. Order the elements lifetimes in an increasing order: $t_{i_1} < t_{i_2} < ... < t_{i_{n+m}}$.

Step 3. Determine the instant of network failure $\tau^* = \tau_{i_r}$.

Repeat Steps 1-3 *N* times.

Step 4. Order *N* replicas of network lifetime $\tau_1^* < \tau_2^* < ... < \tau_N^*$.

Step 5. Estimate $F_N(t)$ as follows:

$$\hat{F}_N(t) = \frac{\#of\ (\tau_i^*(t))}{N}, t = \Delta, 2\Delta, ..., r \cdot \Delta.$$

Step 6. Estimate the $Var[F_N(t)]$ as

$$\hat{Var}[\hat{F}_N(t)] = \frac{\hat{F}_N(t)(1 - \hat{F}_N(t))}{N}, t = \Delta, 2\Delta, ..., r \cdot \Delta.$$

Remark. Let the *critical element* be an element which disrupts the terminal connectivity at time τ^*. A straightforward implementation of step 3 means a repeated operation of checking network connectivity at times $t_{i_1}, t_{i_2}, ... <, t_{i_{n+m}}$, which is very time consuming. We propose a very efficient method which will determine the instant of network failure.

B. Algorithm for Network Failure Determination

Given a network $N = (V, E, K)$, let $t_1 < t_2 < ... < t_{n+m}$ be the lifetimes of network elements (nodes and edges). Denote by x_i the element which corresponds to lifetime t_i. For every network edge $x_i = (x_l, x_m)$, where x_i is an edge and x_l, x_m are nodes, assign $w_i = \min\{t_i, t_l, t_m\}$. For every network node x_j, assign $w_j = t_j$. Let T_V be the *maximal spanning tree* of V. We say a network element x_i is *terminal-irrelevant* in T_V if its removal does not disconnect the terminals. Let T_K be a *terminal-relevant-subtree of* T_V obtained by removing all *terminal-irrelevant* elements. Next we find the network lifetime and the critical element by using the following Lemma.

Lemma 1. Let w_i be the minimal weight element in T_K. Then the network lifetime is $\tau^* = w_i$ and we consider two cases:

1. If x_i is a node, then the corresponding node is critical.
2. If $x_i = (x_l, x_m)$ is an edge, then the critical element is the one which corresponds to the minimal lifetime $\min\{t_i, t_l, t_m\}$ (it can be either the edge x_i, or one of the nodes x_l, x_m).

Proof. Suppose that τ^* is not the true network lifetime. Let $\tau' \ne \tau^*$ be the true network lifetime. Therefore, either $\tau' < \tau^*$ or $\tau' > \tau^*$. We will show both cases are not possible.

Case $\tau' < \tau^*$: The network is still in the *Good* state at time $\tau' < \tau^*$, since all the elements in T_K are still up – a contradiction to τ' being the network lifetime.

Case $\tau' > \tau^*$: In this case, there exists some *spanning tree* T_V' and a *terminal-relevant-subtree* T_K' so that for each element x_i' in T_K' it holds $w_i' > \tau^*$. Note that in T_V there is at least one edge with weight τ^*. In a similar way as in the famous Kruskal algorithm for MST, we will remove the minimal edge *e* in T_V (the one with weight τ^*) which will result in two disconnected components. Then we would take an edge *e'* in T_K' which connects this two components. The resulting graph is a tree and has a weight which is greater than the weight of T_V - a contradiction to T_V being the *maximal spanning tree* of V. ■

Remark. Lemma 1 in [8] and [17] addressed the case of reliable nodes and unreliable edges. The algorithm described above is a generalization of this Lemma to the case of both the nodes and the edges being unreliable.

Example. Let us illustrate the proposed algorithm. In Fig. 1.a there is a network with 20 elements, 3 of which are terminals (A,D,F). And let

$$t_2 < t_{12} < t_{11} < t_B < t_3 < t_{10} < t_6 < t_9 <$$
$$< t_C < t_1 < t_4 < t_5 < t_8 < t_7 < t_E < t_{13} < t_G$$

be the order of elements lifetimes. For simplicity, suppose all time increments are 1. In Fig. 1.b the weight assignment stage is illustrated. In Fig. 1.c a *maximal spanning tree* is constructed. Nodes B, E and edges 5,8 are *terminal-irrelevant*. The terminal irrelevant edges are emphasized. As a result, the critical element is node C and the network lifetime is $\tau^* = 8$.

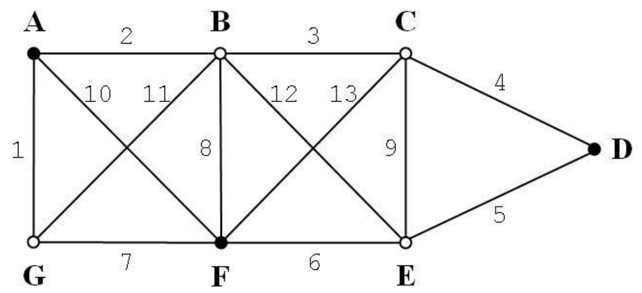

Fig. 1(a)

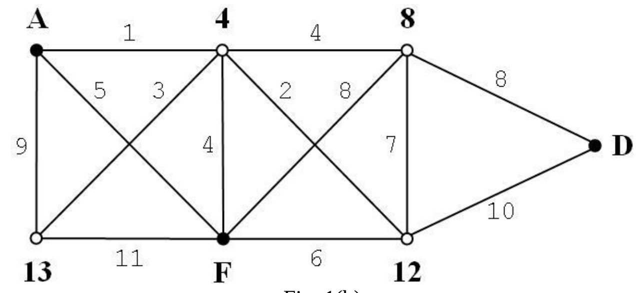

Fig. 1(b)

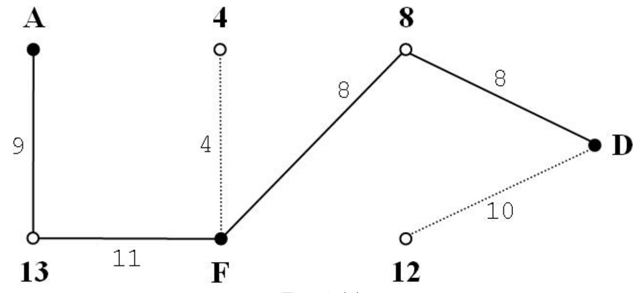

Fig. 1 (c)

C. Simulation Scheme 2

We alter the simulation scheme to make use of the algorithm in B.

Step1. Simulate element according to its lifetime distribution: an edge *e* with $F_e(t)$ and a node *v* with $F_v(t)$.

Step 2. Use the algorithm in B to obtain the critical element and its lifetime τ^*.

Repeat Steps 1-2 *N* times.

Step 3. Order *N* replicas of network lifetime $\tau_1^* < \tau_2^* < ... < \tau_N^*$.

Step 4. Estimate $F_N(t)$ as follows:

$$\widehat{F}_N(t) = \frac{\#of\ \ (\tau_i^*(t))}{N}, t = \Delta, 2\Delta, ..., r \cdot \Delta.$$

III. Using the Combinatorial Spectrum to Estimate the Network Lifetime

The combinatorial invariant of the network – its combinatorial spectrum is very useful and was used in [17] and [22] for different purposes. Here we use it for evaluating the lifetime distribution for a network with unreliable and identical edges and unrelable and identical nodes. Using this highly efficient method provides a significant gain in the simulation time. Since the spectrum concept is relatively new, we start by giving a short description of it, based on our previous paper [22].

Let Π_V be a set of all possible node permutations of nodes in *V* and let Π_E be a set of all edge permutations of edges in *E*. Define the following Cartesian product to be $\Pi = \Pi_{V\backslash K} \times \Pi_E$. Every permutation $\pi \in \Pi$ is a pair $(\pi_{V\backslash K}, \pi_E)$, where $\pi_{V\backslash K} \in \Pi_{V\backslash K}, \pi_E \in \Pi_E$. By sub-permutation $\pi(i, j)$ of π we denote a sequence constructed out of the first *i* nodes from $\pi_{V\backslash K}$ and first *j* edges from π_E. For each sub-permutation $\pi(i, j)$ we define a network state $N(\pi(i, j))$, where all the nodes and edges in $\pi(i, j)$ are *up* and all the other nodes and edges are *down.*

For example, $N(\pi(i,m))$ is a state with first *i* nodes being *up* with all other nodes being *down*. Recall that *m* is the total number of edges; therefore all the edges are *up* as well.

Next we define an *anchor*. This definition slightly differs from the one in [22]. The *anchor* plays a central role in all combinatorial approaches.

Definition 2.1. Associate with each permutation $\pi \in \Pi$ a set of indexes pairs $A(\pi)$ as follows. Let $r = r(\pi)$ be the first index in permutation π so that $N(\pi(r,m))$ is *Good* and for a fixed index *r*, let $s = s(\pi)$ be the first index in π so that $N(\pi(r,s))$ is *Good.* Then,

$$A(\pi) = \{(i, j) \mid i \geq r, j \leq s\},$$

where *j* is the first index so that $N(\pi(i, j))$ is *Good* .

Each state $N(\pi(i, j))$ with $(i, j) \in A(\pi)$ is called the *anchor* of permutation π (there may be several anchors defined by a single permutation).

Definition 2.2. Denote by $x_{i,j}$ the number of all permutations π such that $N(\pi(i, j))$ is an *anchor* of π. We say that the set

$$SP = \{\{x_{i,j}\}, 1 \leq i \leq n, 1 \leq j \leq m\}$$

is the *combinatorial spectrum* of the network.

The following example demonstrates these definitions.

Example. In Fig. 2 there is a simple network with 4 nodes. Nodes S and T are terminals.

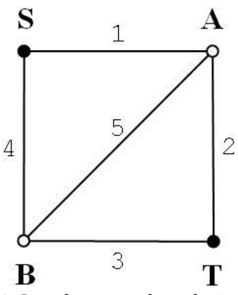

Fig. 2 Simple network with 4 nodes

We consider the following permutation: $\pi = ((\pi_V = A, B), \pi_E = (1,3,4,2,5))$. First we identify the index r so that $N(\pi(r,5))$ is good and obtain $r = 1$. As a result, the first anchor of the permutation is $N(\pi(1,4))$, i.e. the state in which node A and edges 1,3,4,2 are *up* and the rest of the elements are *down*. The second (and the last) anchor of the permutation is $N(\pi(2,3))$, in which nodes A, B and edges 1,3,4 are *up* and the rest of the elements are *down*. It is easy to get the combinatorial spectrum of this network:

$$SP = \begin{Bmatrix} x_{1,2} = 24, x_{1,3} = 48, x_{1,4} = 72, x_{1,5} = 96, \\ x_{2,2} = 48, x_{2,3} = 144, x_{2,4} = 48 \end{Bmatrix}.$$

For all other values of (i, j) we get $x_{i,j} = 0$.

Based on the combinatorial spectrum we obtain the static network reliability:

$$R(N) = \sum_{r=1}^{n} \sum_{s=1}^{m} x_{r,s} f(r,s),$$

where

$$f(r,s) = \frac{1}{r!(n-r)!} p_v^r q_v^{n-r} \sum_{j=s}^{m} \frac{1}{j!(m-j)!} p_e^j q_e^{m-j}.$$

The appropriate Monte Carlo computational scheme may be constructed in a similar way as in [22]. If we put $p_v = 1 - F_v(t), q_v = F_v(t)$ and $p_e = 1 - F_e(t), q_e = F_e(t)$ we can obtain the expression for $F_N(t)$. Note that if we know the combinatorial spectrum of the network, then the computation of $F_N(t)$ is straightforward without any simulations.

Moreover, we can use the same combinatorial spectrum for computing different distribution functions of network elements.

Remark. We can ask why for the same purpose not to use the one-dimensional spectrum, which is much more comfortable in use rather than the two-dimensional one. In this one-dimensional case the permutation is a sequence of "intermixed" nodes and edges.

There are two reasons not to use this approach. First, note that the permutations are much longer: $(n+m)! > n! \cdot m!$, which essentially affects the method efficiency. The second reason is that we can consider only identical nodes and edges, that is $p_e = p_v$ for all edges and nodes, which is much less applicable.

IV. The Approximation Formula for Network Lifetime Distribution

In this section we describe a method for computing the approximation of network lifetime distribution in the case of elements lifetime exponential distribution. We would use one of the Burtin and Pittel theorem formulations described in [23].

Let S be a monotone system, which means that the superset of the *Good* state is also a *Good* state. Suppose that the lifetimes of all elements have exponential distribution $\tau_i \sim \exp(\lambda_i)$ and $\lambda_i = \alpha \cdot \theta_i$, where $\alpha \to 0$ (i.e. we consider very reliable systems). Denote by D_r the set of all minimal cuts of *minimal* size r. For some cut $C \in D_r$, we define by $I(C)$ the set of element indexes in C. Then the system reliability expression is formulated as follows:

$$R_S(t) = \exp(-\alpha^r \cdot t^r \cdot g(\theta)) + O(\alpha^r)$$

(the Weibull distribution), where $g(\theta) = \sum_{C \in D_r} \prod_{i \in I(C)} \theta_i$

Let us show an example of the above.

Example. Consider a network shown in Fig. 2. This network is a monotone system. The size of a minimal cut is $r = 2$, as a result

$D_2 = \{(1,4),(2,3),(A,B),(A,4),(A,3),(B,1),(B,2)\}$.

For simplicity, assume that all lifetimes distribute exponentially with the same $\lambda, \lambda = \alpha \cdot \theta, \alpha \to 0$. Therefore, $g(\theta) = 7 \cdot \theta^2$ and $R_N(t) \approx \exp(-7 \cdot \alpha^2 \cdot \theta^2)$.

The approximation formula described above is very useful for reliable networks. To use it, one has to enumerate all the minimal cuts of minimal size in conjunction with the appropriate values of θ_i.

In the case of identical nodes and identical edges it may be done very efficiently by using the combinatorial spectrum. Take some permutation π and one of its anchors $N(\pi(i, j))$. According to Definition 2.1 the set of nodes and edges $\{(x_{i+1},...,x_n),(y_j,...,y_m)\}$ is some minimal cut, where $x_{i+1},...,x_n$ are nodes and $y_j,...,y_m$ are edges. It follows that we would obtain a minimal cut of minimal size for the maximal value of $i + j$ and the size of the cut would be $r = n + m - i - j + 1$. If the spectrum element $x_{i,j}$ is of the maximal size $i + j$ then the number of minimal cuts of minimal size which are *related* to this pair of indexes is $\frac{x_{i,j}}{i! \cdot (n-i)!(j-1)(m-j+1)!}$. Therefore to get the number of all minimal cuts of minimal size we must consider all pairs of indexes (i, j) that are of a maximal size $i + j$.

Note that if we use the two-dimensional combinatorial spectrum we can obtain all the desired cuts of the form "nodes and edges" and "only edges", but not "only nodes". This is because in the anchors construction, the last element is always an edge and as a result, it is a part of the cut. Thus to get the desired cuts of all forms, we also have to construct a one-dimensional spectrum of nodes.

Example. Consider the network in Fig. 2 and the Example in section IV. We see that the values $x_{1,5} = 96$ and $x_{2,4} = 48$ are related to maximal size 6. The minimal cuts of the form "nodes and edges" are $(A,4)$, $(A,3)$, $(B,1)$ and $(B,2)$. Their number is computed in the following way: $\frac{96}{1!\cdot 4!} = 4$. The number of the minimal cuts of the form "only edges" is $\frac{48}{2!\cdot 3!} = 4$.

In order to get the desired values for large networks we have to simulate the combinatorial spectrum according to the Monte Carlo schemes from [22]. We denote by M the number of repetitions in the simulation scheme and by $y_{i,j}$ the values of the simulated spectrum. Then the values $\hat{x}_{i,j} = \frac{y_{i,j} \cdot n!m!}{M}$ are the unbiased estimators for the true values of $x_{i,j}$. Similarly we can use the one-dimensional spectrum for computing the cuts of the form "only nodes ".

V. Numerical Results

Here we present some simulation results obtained by the MC scheme described in sections II-IV based on 100000 replications. The results we provide are in the form of reliability: $R_N(t) = 1 - F_N(t)$.

In TABLE I we show the estimates for two networks; the first is a mesh network with 36 nodes and 75 edges; the second is a hypercube H_6 with 64 nodes and 192 edges. The computations were performed for $\lambda = 0.1$. The columns marked with II hold the results obtained by using a method described in Section II, while the results in columns marked with III are obtained using the method from Section III. We can see that both methods produce almost identical results.

In TABLE II we bring the results for a mesh network with 36 nodes and 75 edges obtained by using the approximation formula from Section IV, performed for $\lambda = 0.1, 0.5$. For comparison purposes we added computation results in columns marked with III obtained by using the method in section III. As expected, the approximation formula provides us with very good results for small values of t and λ.

VI. Conclusion

(1) To the best of our knowledge, very few works were conducted on dynamic networks, especially for the case of unreliable nodes and edges.

(2) The algorithm, suggested in [17] and in section II is very effective due to Lemma 1.

(3) The method suggested in section III is very effective since it makes use of the combinatorial spectrum. Once the combinatorial spectrum is computed it can be used for as many values of nodes and edges probabilities as desired.

(4) In our previous work [22] we already stressed out the main advantages of using combinatorial spectrum for a static network. Here we showed that it can be very useful for solving different reliability problems for dynamic networks.

TABLE I
Reliability Estimation by Two Methods

t	R of Mesh		R of Hypercube	
	II	III	II	III
0.1	0.9999	0.9999	0.9999	0.9999
0.2	0.9998	0.9999	0.9999	0.9999
0.3	0.9996	0.9996	0.9999	0.9999
0.4	0.9990	0.9991	0.9999	0.9999
0.5	0.9980	0.9982	0.9999	0.9999
0.6	0.9968	0.9970	0.9999	0.9999
0.7	0.9950	0.9953	0.9999	0.9999
0.8	0.9928	0.9930	0.9999	0.9999
0.9	0.9898	0.9902	0.9999	0.9999
1.0	0.9850	0.9867	0.9999	0.9999
1.5	0.9565	0.9567	0.9994	0.9994
2.0	0.8973	0.8978	0.9976	0.9972
2.5	0.8020	0.8018	0.9923	0.9917
3.0	0.6732	0.6711	0.9807	0.9799
3.5	0.5242	0.5224	0.9584	0.9575
4.0	0.3782	0.3784	0.9188	0.9179
4.5	0.2582	0.2567	0.8545	0.8532
5.0	0.1653	0.1645	0.7580	0.7575
5.5	0.1001	0.1006	0.6336	0.6333
6.0	0.0581	0.0593	0.4951	0.4945
6.5	0.0341	0.0339	0.3628	0.3607
7.0	0.0190	0.0189	0.2462	0.2472

TABLE II
Reliability Estimation by the Approximation Formula

t	$\lambda = 0.1$		$\lambda = 0.5$	
	III	IV	III	IV
0.1	0.9999	0.9999	0.9982	0.9978
0.2	0.9999	0.9999	0.9866	0.9822
0.3	0.9996	0.9995	0.9566	0.9411
0.4	0.9991	0.9988	0.8978	0.8659
0.5	0.9982	0.9978	0.8017	0.7548
0.6	0.9970	0.9961	0.6710	0.6151
0.7	0.9953	0.9938	0.5223	0.4622
0.8	0.9930	0.9908	0.3784	0.3160
0.9	0.9902	0.9870	0.2567	0.1939
1.0	0.9867	0.9821	0.1645	0.1054
1.5	0.9567	0.9411	0.0104	0.0005
2.0	0.8978	0.8659	0.0004	0.0000
2.5	0.8018	0.7548	0.0000	0.0000
3.0	0.6711	0.6151	0.0000	0.0000

References

[1] Agraval, R. E. Barlow, "A survey of network reliability and domination theory", *Operation Research*, vol. 32, 1984, pp. 478-492.
[2] M.O. Ball, "Complexity of network reliability computation", *Networks*, 10, 1980, pp. 153-165.

[3] R. E. Barlow, F. Proschan, *Statistical Theory of Reliability and Life Testing*, 1975. Holt, Rinehart & Winston.
[4] C. J. Colbourn, *The Combinatorics of Network Reliabilty*. Oxford Univ. Press.
[5] C.J. Colbourn, A. Satyanarayana, C. Suffel, K. Sutner, "Computing residual connectedness reliability for restricted networks", *Discrete Applied Mathematics*, 44, 1993, pp. 221-232.
[6] Lucia I. P. Resende, "Implementation of factoring algorithm for reliability evaluation of undirected networks", *IEEE,Trans. Reliability*, vol 37, 1988, pp. 462-468.
[7] M. C. Easton, C. K. Wong, "Sequential destruction method for Monte Carlo evaluation of system reliability", *IEEE Trans. Reliaility*, vol R-29, 1980, pp. 27-32.
[8] T. Elperin, I. Gertsbakh, M. Lomonosov, "Estimation of network reliability using graph evolution models", *IEEE Transactions on Reliability*, 40(5), 1991, pp. 572-581.
[9] T. Elperin, I. Gertsbakh, M. Lomonosov, "An evolution model for Monte Carlo estimation of equilibrium network renewal parameters", *Probability in the Engineering an Informational Sciences*, 6, 1992, pp. 457-469.
[10] G. S. Fishman, "A Monte Carlo sampling plan for estimating network reliability", *Operations Research*, vol 34, 1986, pp. 581-592.
[11] G. S. Fishman, "A Monte Carlo sampling plan for estimating reliability parameters and related functions", *Networks*, vol. 17, 1987, pp. 169-186.
[12] K-P. Hui, N. Bean, M. Kraetzl, and D P. Kroese, "The Tree Cut and Merge Algorithm for Estimation of Network Reliability", *Probability in the Engineering and Informational Sciences*, 17(1):24-25, 2003.
[13] H. Kumamoto, T. Kazuo, I. Koichi, E. J. Henley, "State transition Monte Carlo for evaluating large, repairable systems", *IEEE Trans. Reliability*, vol R-29, 1980, pp. 376-380.
[14] M. Lomonosov, "On Monte Carlo estimates in network reliability", *Probability in the Engineering and Information Sciences, 8*, 1994,
[15] M Lomonosov, Y. Shpungin, "Combinatorics of Reliability Monte Carlo", *Random Structures & Algorithms*, 14,1999, No. 4, pp. 329-343.
[16] Y.Shpungin, "Combinatorial and Computational Aspects of Monte Carlo Estimation of Network Reliability*"*, PhD. Dissertation, Dept. of Mathematics and Computer Science, Ben Gurion University of the Negev, 1996.
[17] Gertsbakh and Y. Shpungin, "Combinatorial approaches to Monte Carlo estimation of network lifetime distribution", *Appl. Stochastic Models Bus. Ind*., 2004, 20:49-57.
[18] C. J. Colbourn, D. D. Harms, "Bounding all-terminal reliability in computer networks", *Networks*, 1988, pp. 1-12.
[19] Z. He, Y. Tian and Y. Chen, "Simulating the Reliability of Distributed Systems with Unreliable Nodes", *I. J. of Simulation*, vol 3, 1-2, pp. 68-78.
[20] K-P. Hui, "Reliability Estimation", Ph D. dissertation, Faculty of Engineering, Computer and Mathematical Sciences, University of Adelaide, 2005.
[21] Y. Shpungin, Combinatorial Approaches to Monte Carlo estimation of dynamic systems reliability parameters, *Communication of Dependability and Quality Management: An International Journal, Volume* 9, *Number* 1, 2006, pp. 69-75.
[22] Y. Shpungin, Combinatorial Approach to Reliability Evaluation of Network with Unreliable Nodes and Unreliable Edges, *International Journal of Computer Science, Vol* 1, *num*. 3, 2006, pp. 177-183.
[23] I. Gertsbakh, Reliability Theory, Springer-Verlag, 2000.