以遞迴演算法作分散式系統可靠度評估
Recursive Algorithms for System Reliability Evaluation in Distributed Computer Systems

柯偉震
Wei-Jenn Ke

Chunghwa Telecommunication Laboratories, Taiwan R.O.C.
wjke@ms.tl.gov.tw

王勝德
Sheng-De Wang

Department of Electrical Engineering
National Taiwan University, Taiwan R.O.C.
sdwang@cc.ee.ntu.edu.tw

摘要

本論文提出兩種遞迴演算法作分散式系統可靠度評估. 第一種演算法運用負向保守策略而第二種則運用正向策略, 實驗結果證實第二種演算法效能較好.

關鍵字：終始阻塞率, 負(正)向保守策略.

Abstract

In this paper, two recursive and one-pass algorithms are presented to evaluate source-to-sink blocking probability, and hence reliability, in a distributed computer system (DCS). Algorithm 1 employs negative conservative policy (NCP) while algorithm 2 employs positive conservative policy (PCP) to generate exclusive and mutually disjoint decomposing events. A lot of experiments demonstrate that algorithm 2 using PCP performs better.

Key words: source-to-sink blocking probability, negative (positive) conservative policy.

1. Introduction

Advances in computer technology and the need to have the computers communicating with each other have led to an increased demand for a reliable distributed computer system (DCS). In the DCS network, one important performance measure is the congestion (blocking) probability for computer communications originated at the source node s and destined for the terminal node t. Blocking occurs when the computer traffic activated from s can not reach t. In this paper, the evaluation of exact node-to-node blocking probability or s-t terminal unreliability in the DCS is addressed. Such system unreliability refers to the probability that there exists at least one s-t cutset whose removal disconnects t from s in the DCS.

The terminal unreliability (reliability) problem is proven to be NP-hard [18] and has been studied by many algorithms [1-17]. These algorithms can be classified into two-pass [1-4] and one-pass [5-17] methods. In two-pass methods, all mincuts/minpaths from s to t must be derived at first, and then a complex disjoint process is applied to convert them into disjoint unreliability (reliability) expression. The enumeration of mincuts/minpaths is also NP-hard [18]. Hence, they grows exponentially with the size of network, yielding a burdensome computation in the case of large network. Besides this, since all outputs of the first pass must be saved as inputs to the second pass, the large temporary space must be allocated for them.

All the drawbacks of two-pass algorithms can be avoided by one-pass algorithms for the fact that no prior knowledge of all mincuts/minpaths are needed. In one-pass algorithms, mincuts/minpaths are generated in a way such that they are all disjoint with the previous ones, thereby the unreliability (reliability) expression of the network can be obtained by taking the direct sum of them. A famous one-pass method is known as pivotal decomposition [10-16] that, using the factoring theorem, the network is expressed in terms of a network with one fewer vertex and another with one fewer edge. The factoring theorem is applied recursively on the reduced networks without knowing any mincut/minpath. In this paper, another more efficient one-pass approach, based on a special state space decomposition theorem, extended from the factoring theorem, is derived to decompose the network into several subnetworks recursively, instead of only two by pivotal decomposition. The rationale under this approach is to choose a set of keystone elements of the network, instead of only one pivotal element in pivotal decomposition, to generate a set of exclusive mutually disjoint (EMD) events and decompose the network into successive smaller and disjoint subnetworks. Since all subnetworks are made smaller and disjoint, whose unreliabilities (reliabilities) are readily evaluated and then directly summed to get the system reliability.

Two variations of decomposing policy to generate a set of EMD events are discussed in [9], the first policy called negative conservative policy (NCP), which has been employed by Rai and Kumar [17] to compute system unreliability, and the second one named positive conservative policy (PCP), both policies are introduced and carried out in this paper by two decomposing algorithms, referred as algorithm 1 and 2, respectively. A lot of experiments are performed to measure and compare the efficiency between these two algorithms. It is shown that algorithm 2 with PCP outperforms algorithm 1 with NCP in terms of the less computation time as well as total number of generated disjoint terms.

2. Preliminaries

Assumptions

1. A DCS is modeled by graph G with no self loops, where nodes denoting computer sites and edges representing communication links.
2. The nodes in G are considered to be perfectly reliable and the links are either in working or failed states.
3. Blocking of all links are s-independent.

Notations

s, t the source node, the terminal node in the DCS.

X_i, N_j Link i, node j in the DCS.

x_i, $\overline{x_i}$ Boolean variable representing working, failed of link X_i.

p_i, q_i the probability that X_i is working, failed; where $p_i + q_i = 1$.

BP(G) the blocking probability for graph G.

Other notations are summarized in Section 4 for readily reference.

Nomenclature

minpath: a path with no proper subset is also a path in the DCS.

mincut: a cutset with no proper subset is also a cutset in the DCS.

exclusive and mutually disjoint (EMD): a set of product terms is EMD when they are disjoint each other.

negative conservative policy (NCP): Given a set of r Boolean variables $\{x_1, x_2, ..., x_r\}$, the policy yielding a set of r+1 EMD events $\{\overline{x_1}, x_1\overline{x_2}, ..., x_1x_2\cdots x_{r-1}\overline{x_r}, x_1x_2\cdots x_r\}$.

positive conservative policy (PCP): The dual policy of NCP; the policy yielding a set of r+1 EMD events $\{x_1, \overline{x_1}x_2, ..., \overline{x_1}\,\overline{x_2}\cdots\overline{x_{r-1}}x_r, \overline{x_1}\,\overline{x_2}\cdots\overline{x_r}\}$.

node to node blocking (terminal unreliability): the probability that the traffic originated at s can not reach t.

3. Background

3.1 State Space Decomposition Approach

The basic concept of the proposed algorithms is to decompose the state space which can be thought as an enumeration of the states in which the communication between s and t in the DCS is either working or blocked. The states of G can be partitioned into two sets with respect to the working or blocked state of link l, and consequently, the blocking probability for G can be expressed as [13]:

$$\mathrm{BP}(G) = p_l\,\mathrm{BP}(G|l) + q_l\,\mathrm{BP}(G|\bar{l}), \qquad (1)$$

where $G|l$ denotes G with l working or contracted, while $G|\bar{l}$ denotes G with l failed or cut. Eq. (1) is called the factoring theorem, and iterative substitution of it to compute the blocking probability yields the following theorem, which is the subject of this paper.

Theorem 1: Assuming that there are links $X_1, X_2, ..., X_e$ adjacent from s, BP(G) can be computed as:

$$\mathrm{BP}(G) = q_1\mathrm{BP}(G|\bar{1}) + p_1q_2\mathrm{BP}(G|1\bar{2}) + \cdots$$
$$+ p_1p_2\cdots p_{e-1}q_e\mathrm{BP}(G|12\cdots\bar{e}) + p_1p_2\cdots p_e\mathrm{BP}(G|12\cdots e),$$
(2)

where $G|12\cdots i-1\bar{i}$ denotes G with X_1, X_2, ..., X_{i-1} working but X_i failed.

proof: Eq. (2) is easily obtained by iterative substitution of Eq. (1). Q. E. D.

Theorem 1 is used as the rationale of algorithm 1 described in Section 4. The probability space for BP(G) is decomposed into e+1 subspaces that are further decomposed into several smaller probability space recursively. Since the computation problem is divided into e+1 subproblems, other than two, each subproblem are made smaller and converge to the termination condition more rapidly than by traditional pivotal decomposition methods. The coefficients in the right side of Eq. (2) corresponding to a set of EMD events $\overline{x_1}, x_1\overline{x_2}, ..., x_1x_2\cdots x_{e-1}\overline{x_e}, x_1x_2\cdots x_e$ are used to decompose the graph into a set of disjoint subgraphs. Such decomposition is named negative conservative policy (NCP) by Fratta and Montanari [9]. Rai and Kumar [17] also adopts it to decompose and reduce the directed network.

For the generated subgraph $G|12\cdots i-1\bar{i}$, it indicates that G is reduced by a series contractions of links X_1, X_2, ..., and X_{i-1} and a deletion of X_i. Since the contraction of any link l will result in the endnodes of l collapsed into a single node, named fused or coalescence node; hence s in conjunction with other endnodes of links X_1, X_2, ..., X_{i-1} are fused to form a new source node. The incident links of this new node are then computed to recursively decompose the subgraph. In the proposed algorithms, the incident links can simply be computed from the union of the adjacent links from endnodes of X_1, X_2, ..., X_{i-1}, but excluding the failure link X_i. Moreover, it is necessary to delete the loop link forming a loop in conjunction with some links of X_1, X_2, ..., X_{i-1}, because, in coalescing endnodes of X_1, X_2, .., X_{i-1}, such link is contracted into the new source node. The fusion and decomposing process is recursively applied until the terminating conditions occur that the generated subgraph fuses t or no incident links can be further found. In computing the blocking probability, fusing t implies a failure case wherein there exists a path reaching t, whereas no finding incident links indicates a success case wherein a cutset has been found.

All the computations involved in the recursion are carried out by a multi-ary state space tree whose internal nodes correspond to the reduced subgraphs while leaves represent success or failure ones. A four-node example along with the state space tree is shown in Fig. 1 to illuminate the stated concepts. First, G is decomposed into $G|\bar{1}$, $G|1\bar{3}$ and $G|13$ using incident links X_1 and X_3. The tree edges noted with $\bar{1}$, $1\bar{3}$ and 13 represent three EMD events of Eq. (2); the event that X_1 failed, the event that X_1 working but X_3 failed, and the event that both X_1 and X_3 working. For $G|\bar{1}$, since X_1 is cut, only adjacent link X_3 from s is used to decompose it, yielding $G|\bar{1}\bar{3}$ and $G|\bar{1}3$. Since no incident links from s can be found in $G|\bar{1}\bar{3}$, a cutset $\bar{x}_1\bar{x}_3$ has been found; thus we terminate it and add the probability term q_1q_3 to BP(G). For $G|1\bar{3}$, the failure link X_3 is removed and the working link X_1 yields the fusion of nodes N_1 and N_2 to generate a new source node; thus X_2 and X_5 are incident links with it. In our algorithms, X_2 and X_5 are directly computed by the adjacent links from N_1 and N_2, while excluding failure link X_3. Both X_2 and X_5 are then used to decompose $G|1\bar{3}$ into $G|1\bar{2}\bar{3}$, $G|12\bar{3}\bar{5}$, and $G|12\bar{3}5$. Since the working link X_2 yields the fusion of t, $G|12\bar{3}\bar{5}$ and $G|12\bar{3}5$ are determined failed and marked with 'F'. At last, for $G|13$, since X_1 and X_3 are working, N_1, N_2, and N_3 are fused into a new source node whose incident links are determined to be X_2 and X_4. Note that although X_5 is a incident link, yet it is excluded because a loop is formed by it in conjunction with X_1 and X_3.

The expansion order for the state space tree can either breadth first (BF) or depth first (DF). However, since BF order yields prohibitive memory requirement for the deeper tree with high branching factor in the large network, we employs DF order so that the space usage is only a linear function of the depth of the tree. Proceeding in the manner, there are six success leaves of twenty-five nodes in the order of $\bar{x}_1\bar{x}_3$, $\bar{x}_1x_3\bar{x}_4\bar{x}_5$, $\bar{x}_1\bar{x}_2x_3\bar{x}_4x_5$, $x_1\bar{x}_2\bar{x}_3\bar{x}_5$, $x_1\bar{x}_2\bar{x}_3\bar{x}_4x_5$, $x_1\bar{x}_2x_3\bar{x}_4$ generated. These success cases, in fact, representing disjoint cutsets, are converted into corresponding probability terms that are directly summed to get the node-to-node blocking as:

$$\begin{aligned}\mathrm{BP}(G) = \ & q_1q_3 + q_1p_3q_4q_5 + q_1q_2p_3q_4p_5 \\ & + p_1q_2q_3q_5 + p_1q_2q_3q_4p_5 + p_1q_2p_3q_4.\end{aligned}$$

3.2 The Dual Approach

In the previous decomposition approach, if there are n incident links found in the intermediate node of the state space tree, then the cutset with these links failed will be determined after successive n DF order expansions. As can be seen from Fig. 1, finding X_1 and X_3 incident with s in the root node, we can determine $\bar{x}_1\bar{x}_3$ to be a cutset; however, this is delayed until two more deeper level expansions. Similar situation with $G|1\bar{3}$, once incident links X_2 and X_5 are computed, the disjoint cutset $x_1\bar{x}_2\bar{x}_3\bar{x}_5$ can be determined immediately and it is not necessary to wait until two consecutive DF order expansions. Hence, the disjoint cutset must be determined as soon as the incident links are known. This fact motivates the proposition of the following theorem, which is regarded as the dual theorem of Theorem 1, and can avoid such clumsiness.

Theorem 2: With the same assumptions as Theorem 1, BP(G) is computed as:

$$\begin{aligned}\mathrm{BP}(G) = \ & p_1\mathrm{BP}(G|1) + q_1p_2\mathrm{BP}(G|\bar{1}2) + \cdots \\ & + q_1q_2\cdots q_{e-1}p_e\mathrm{BP}(G|\overline{12}\cdots\overline{e-1}e) + q_1q_2\cdots q_e.\end{aligned} \qquad (3)$$

proof: With the same deduction of Eq. (2), from the factoring theorem, the following equation is obtained:

$$\begin{aligned}\mathrm{BP}(G) = \ & p_1\mathrm{BP}(G|1) + q_1p_2\mathrm{BP}(G|\bar{1}2) + \cdots \\ & + q_1q_2\cdots q_{e-1}p_e\mathrm{BP}(G|\overline{12}\cdots\overline{e-1}e) \\ & + q_1q_2\cdots q_e\,\mathrm{BP}(G|\bar{1}\bar{2}\cdots\bar{e}).\end{aligned}$$

Since, in $G|\bar{1}\bar{2}\cdots\bar{e}$, all incident links with the source node have been failed, the source node is isolated; therefore, $\mathrm{BP}(G|\bar{1}\bar{2}\cdots\bar{e}) = 1$ and then Eq. (3) is obviously obtained. Q. E. D.

In comparison with Theorem 1, two differences are found. First, the set of EMD events x_1, $\bar{x}_1x_2$, ..., $\bar{x}_1\bar{x}_2\cdots\bar{x}_{e-1}x_e$, $\bar{x}_1\bar{x}_2\cdots\bar{x}_e$ in Eq. (3) is adopted to decompose the network, which is the dual policy of NCP, named positive conservative policy (PCP) [9]. This policy explains that Theorem 2 can be regarded as the dual theorem of Theorem 1. Secondly, it should be noted that the term $q_1q_2\cdots q_e$, implying a cutset with all incident links failed, is given at each decomposition step, without any delay for further decomposition; hence, the drawback of Theorem 1 is removed. Moreover, with this property, the number of generated subgraphs as well as the size of the state space tree both can be reduced.

Theorem 2 is the rationale of the proposed algorithm 2. Like Theorem 1, a state space tree is constructed by it and whose nodes are generated also in DF order. Fig. 2 takes the example in Fig. 1. At first, incident links X_1 and X_3 with s are found to decompose G into $G|1$ and $G|\bar{1}3$, yielding a cutset $\bar{x}_1\bar{x}_3$. For $G|1$, since incident links X_2, X_3 and X_5 are derived, another disjoint cutset $x_1\bar{x}_2\bar{x}_3\bar{x}_5$ and the decomposition of $G|1$ into $G|12$, $G|1\bar{2}3$ and $G|1\bar{2}\bar{3}5$ are obtained. $G|12$ is determined failed because of the fusion of t. Only one incident link X_4 is found in $G|1\bar{2}3$ and $G|1\bar{2}\bar{3}5$, yielding two disjoint cutsets $x_1\bar{x}_2x_3\bar{x}_4$ and $x_1\bar{x}_2\bar{x}_3x_5\bar{x}_4$ and two subgraphs $G|1\bar{2}34$ and $G|1\bar{2}\bar{3}45$ that are determined failed due to comprising t. Proceeding in the same manner as $G|1$, $G|_{\bar{1}3}$ is processed and another two disjoint cutsets $\overline{x_1}\, x_3\, \overline{x_4}\, \overline{x_5}$ and $\overline{x_1}\, \overline{x_2}\, x_3\, \overline{x_4}\, x_5$ will be obtained. Ultimately, six terms of eleven nodes, less than twenty-five in Fig. 1, are generated. In the last section, a lot of experiments reveal that using Theorem 2 is capable of reducing considerably the number of generated disjoint cutsets as well as the computation time as compared to Theorem 1.

Using Theorem 2, the expansion may terminate either on the failure condition that the reduced graph comprises t or success condition that no incident links can be further found. Fig. 3 is an example that gives two rectangular leaves where no incident links can be derived, indicating two cutsets $\bar{1}\,2\,\bar{3}\,4$ and $\bar{1}2\bar{3}\bar{4}5$ are found.

Since the computation cost is exponential with the network size, the size of the state space tree needs to be as small as possible. This can be accomplished by two techniques. First, if the expansion chooses all links that immediately form a termination condition to expand first, the tree size can be reduced [19]. Secondly, a set of graph reduction methods, such as serial, parallel, and polygon-to-chain reductions can be applied in every intermediate stage of the state space tree to dramatically reduce the size of the subgraphs and hence, the tree size [16, 20]. However, both techniques are not implemented in the proposed algorithms for simplification.

4. Algorithm Development

Notations

RS reduction status; a set of working or failure links stating the status of the reduced graph.

TN traversed nodes; a set of nodes traversed by the working links of RS.

IL incident links; a set of links incident with the source node.

$\overline{\text{IL}}$ failure set of IL; for example, if IL = {1,3}, then $\overline{\text{IL}}$ = {$\bar{1}$, $\bar{3}$}.

ME a set of mutually exclusive events generated by IL.

DC disjoint cutset; a disjoint product term in the expansion of BP(G).

In this section, we first try to formulate algorithm 2. To carry out the state space tree for algorithm 2, each node is devised to contain five sets RS, TN, IL, ME and DC, which must be updated at every stage to reflect the status of the new generated subgraph. RS is updated by the union of the decomposing event, say e, TN is then updated by the new expanded node incident with the working link of e, IL is computed by the adjacent links from TN but excluding the failure, working links of RS, and furthermore the loop links, while both ME and DC are directly derived from IL. We present the details of algorithm 2 in the following.

Algorithm 2

Input: graph G with (s, t) pair;

Output: the blocking probability BP(G);

BEGIN

1. Initialize RS to be empty and TN to be node s;
2. Find IL to be the incident links with node s;
3. Generate a set of mutually exclusive event ME from IL;
4. Compute DC = $\overline{\text{IL}}$, covert it into the corresponding probability value, and add the value to BP(G);
5. Generate the root node of the state space tree and perform *Procedure BP_Computing* for it;

END

Procedure BP_Computing

BEGIN

FOR each event e in ME DO

BEGIN

1. Update RS = RS ∪ {e};
2. Find the new expanded node incident on the working link of e;
3. IF the new expanded node is t THEN Continue;
   ELSE update TN = TN ∪ {the new expanded node};
4. Compute IL by the union of the adjacent links from nodes in TN while excluding the loop links and the working and failure links of RS;
5. IF IL = ∅ THEN convert RS into the corresponding probability value, add it to BP(G), and Continue;
6. Generate a set of mutually exclusive events ME from IL;
7. Update DC = RS ∪ $\overline{\text{IL}}$, convert it into the corresponding probability value, and add the value to BP(G);
8. Generate a new node of the state space tree and perform *Procedure BP_Computing* for it;

END

END

Fig. 4 demonstrates the snapshot of Fig. 2 to illuminate the proposed algorithm. At first, links X_1, X_3 incident with N_1, i.e., s are set to IL and TN, yielding the disjoint cutset $\bar{1}\,\bar{3}$ in DC and two events 1 and $\bar{1}3$ in ME. Two nodes, i.e., $G|1$ and $G|\bar{1}3$ in Fig. 2, are then expanded and whose RS are updated to be 1 and $\bar{1}3$, respectively. For the left node $G|1$, since X_1 is working, the new expanded node is N_2; hence, TN={N_1, N_2}, and the incident links are determined as X_2, X_3, and X_5 from TN, yielding another cutset DC = RS ∪ $\overline{\text{IL}}$ = $1\,\bar{2}\,\bar{3}\,\bar{5}$ and three events $2, \bar{2}3, \bar{2}\bar{3}5$ in ME to generate $G|12$, $G|1\,\bar{2}\,3$ and $G|1\,\bar{2}\,\bar{3}\,5$. Since the expansion of link X_2 yields the new expanded node N_4, i.e., t, $G|12$ terminates. For the right node $G|\bar{1}3$, since X_3 is expanded, TN is updated as {N_1, N_3} whose adjacent links X_4, X_5 yields the disjoint cutset $\bar{1}3\bar{4}\bar{5}$ and two events in ME, 4 and $\bar{4}5$, to get $G|\bar{1}34$ and $G|\bar{1}3\bar{4}5$.

Since algorithm 1 is the dual method of algorithm 2, they present almost the same. However, minor modifications must be done to get algorithm 1. First, ME must be generated according to NCP instead of PCP. Secondly, in algorithm 1, since all disjoint cutsets are equal to the RS of the leave nodes whose IL = $\varnothing$, thus set DC is not necessary. Therefore, to formulate algorithm 1, step 4 in the main program and step 7 in Procedure BP_Computing should be removed.

5. Experimental Results

We have run algorithm 1 and 2 over various benchmark networks on a SUN SPARC 10 machine. TABLE 1 presents the experimental results for benchmark networks G_i^j $(j \le i)$, where subscript i represents the number of nodes in the network, while superscript j denotes G with N_1 to N_j are completely connected. For all networks, s is located at N_1 whereas t is at $N_{\lfloor i/2 \rfloor}$. Fig. 5 shows an example of benchmark network, G_8^5. In TABLE 1, l represents the number of links, m_i provides the total number of disjoint cutsets for algorithm i, and t_i denotes the computation time in seconds. Assuming that each link has equal blocking probability of 0.2, the node-to-node blocking probability is also computed and illustrated in the field BP.

Making a comparison between algorithm 1 and 2 with respect to m_i and t_i reveals the superiority of algorithm 2 over algorithm 1. As far as m_i is concerned, it shows that m_2 is much less than m_1, indicating less numerical computation and smaller rounding error for the numerical computation of the blocking probability using algorithm 2. On the other hand, the computation time for networks G_8^6 to G_{10}^8 are plotted in Fig. 6, showing there is a gradual increasing time for algorithm 2 as compared to algorithm 1.

6. Conclusions

In this paper, the proposed state space decomposing algorithms outperform traditional pivotal decomposition algorithms because they use the concept of keystone elements decomposition and conservative policy. Moreover, we find that the algorithm using PCP performs better than other algorithms using NCP such as algorithm 1 in this paper and Rai and Kumar's algorithm [17]. Therefore, we strongly recommended that it is efficient and effective to calculate the blocking probability of the DCS using the proposed second algorithm.

References

[1] S. Rai, K. K. Aggarwal, "An efficient method for reliability evaluation of a general network," IEEE Trans. Reliability, vol. R-27, pp. 206-211, Aug. 1978.

[2] K. K. Aggarwal, K. B. Misra, and J. S. Gupta, "A fast algorithm for reliability evaluation," IEEE Trans. Reliability, vol. R-24, pp. 83-85, Apr. 1975.

[3] S. Hariri, C. S. Raghavendra, "SYREL: A symbolic reliability algorithm based on path and cutset methods," IEEE Trans. Computers, vol. C-36, no. 10, pp.1224-1232, Oct. 1987.

[4] S. Soh, S. Rai, "CAREL: Computer aided reliability evaluation for distributed computing networks," IEEE Trans. Parallel and Distributed Systems, vol. 2, no. 2, pp. 199-212, Apr. 1991.

[5] W. P. Dotson, J. O. Gobien, "A new analysis technique for probabilistic graphs," IEEE Trans. Circuits and Systems, vol. 26, pp. 855-865, Oct. 1979.

[6] S. H. Ahmad, "A simple technique for computing network reliability," IEEE Trans. Reliability, vol. R-31, no. 1, pp. 41-44, Apr. 1982.

[7] S. H. Ahamd, A. T. M. Jamil, "A modified technique for computing network reliability," IEEE Trans. Reliability, vol. R-36, no.5, pp. 554-556, Dec. 1987.

[8] Y. B. Yoo and Narsingh Deo, "A comparison of algorithms for terminal-pair reliability," IEEE Trans. Reliability, vol. 37, no. 2, pp. 210-215, June 1988.

[9] L. Fratta and U. G. Montanari, "A recursive method based on case analysis for computing network terminal reliability," IEEE Trans. Communications, vol. COM-26, no.8, pp. 1166-1177, Aug. 1978.

[10] H. Nakazawa, "Bayesian decomposition method for computing the reliability of an oriented network," IEEE Trans. Reliability, vol. R-25, pp. 77-80, Apr. 1976.

[11] M. O. Ball, "Computing network reliability," Operation Research, vol. 27, pp. 823-837, 1979,.

[12] A. Satyanarayana, A. Prabhakar, "New topological formula and rapid algorithm for reliability analysis of complex networks," IEEE Trans. Reliability, vol. R-27, pp. 82-100, June 1978.

[13] A. Satyanarayana, M. K. Chang, "Network reliability and the factoring theorem," Networks, vol. 13, pp. 107-120, 1983.

[14] T. A. Feo and R. Johnson, "Partial factoring: An efficient algorithm for approximating 2-terminal reliability on complete graphs," IEEE Trans. Reliability, vol. 39, no. 3, pp. 290-295, Aug. 1990.

[15] W. R. Wood, "Factoring algorithms for computing K-terminal reliability," IEEE Trans. Reliability, vol. R-35, pp. 269-278, Apr. 1986.

[16] L. Resende, "Implementation of a factoring algorithm for reliability evaluation of undirected networks," IEEE Trans. Reliability, vol. 37, no. 5, pp. 462-468, Dec. 1988.

[17] S. Rai and K. Kumar, "Recursive technique for

computing system reliability," IEEE Trans. Reliability, vol. R-36, no. 1, pp. 38-44, Apr. 1987.

[18] M. O. Ball, "Complexity of network reliability computations," Networks, vol. 10, pp. 153-165, 1980,.

[19] A. Kershenbaum, Telecommunications network design algorithms, McGraw-Hill, p.342, 1993.

[20] A. Satyanarayana, R. K. Wood, "A linear-time algorithm for computing K-terminal reliability in series-parallel networks," SIAM J. Computing, vol. 14, no. 4, pp. 818-832, Nov. 1985.

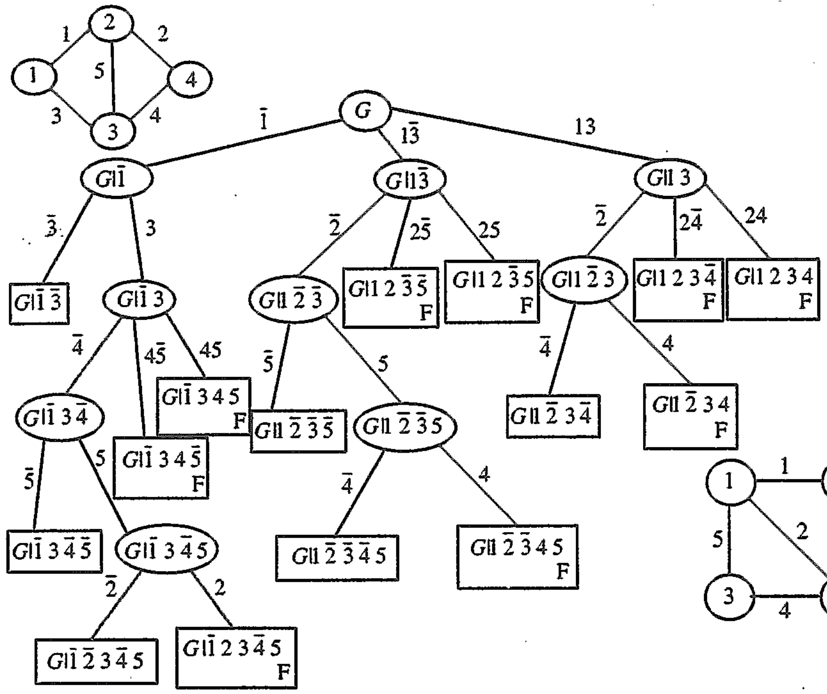

Fig. 1 The state space tree for a four-node DCS using Theorem 1.

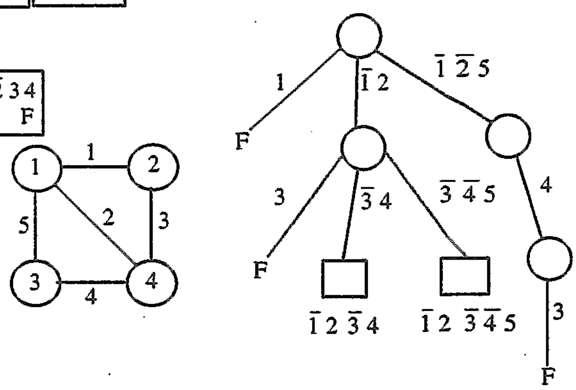

Fig. 3 Another DCS example and its state space tree constructed by Theorem 2.

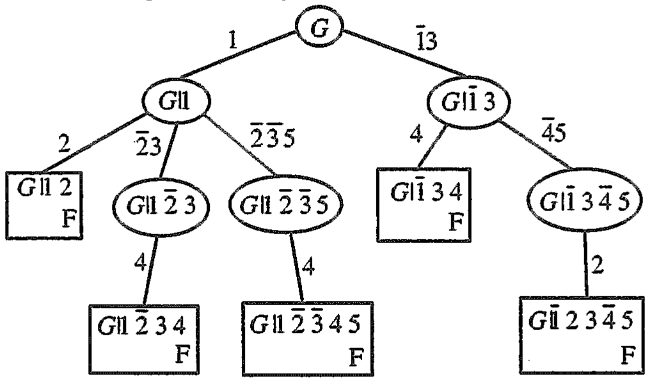

Fig.2 A state space trree constructed by Theorem 2.

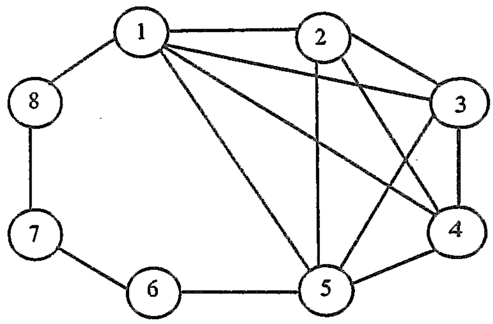

Fig. 5 The benchmark network G_8^5.

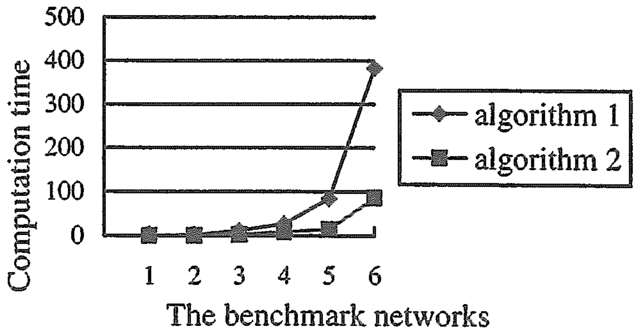

Fig. 6 Plots of computation time for six networks G_8^6 to G_{10}^8.

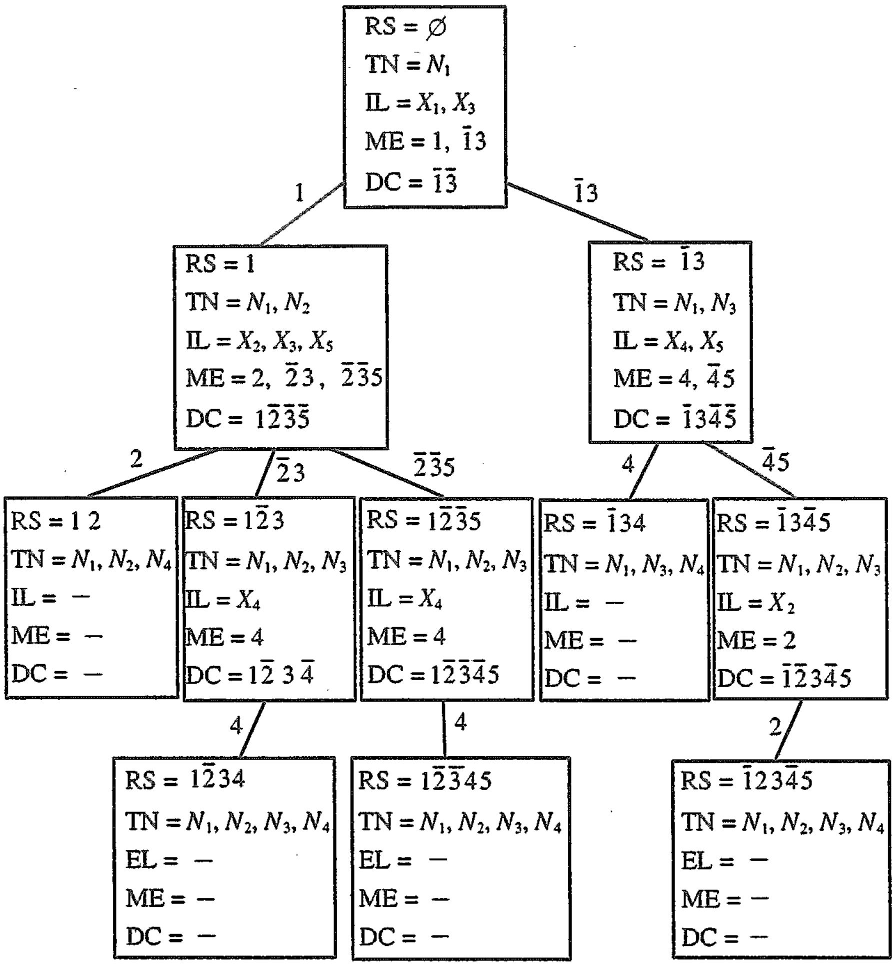

Fig. 4 The snapshot of the state space tree of Fig. 2.

TABLE 1: The experimental results by running Algorithm 1 and 2.

Network	l	m_1	m_2	t_1	t_2	BP
G_6^4	9	36	33	0.013	0.005	0.0134026
G_6^5	12	110	94	0.061	0.019	0.0023763
G_6^6	15	270	212	0.223	0.064	0.0006582
G_8^5	14	287	267	0.140	0.068	0.0027708
G_8^6	18	1657	1266	1.171	0.372	0.0004901
G_{10}^6	20	3398	2971	2.317	0.992	0.0005506
G_8^7	23	11170	6828	11.723	2.638	0.0000877
G_{10}^7	25	28620	20587	27.179	8.693	0.0001025
G_8^8	28	53091	26830	84.792	14.701	0.0000256
G_{10}^8	31	286306	162182	382.003	86.658	0.0000191