Performance Improvement of Edge Expansion Technique for BDD-based Network Reliability Analysis

Ronggen Chen, Yuchang Mo, Zhusheng Pan
Department of Computer Science and Technology, Zhejiang Normal University, Jinhua, China
Email: chen081215@126.com

Abstract—**The network reliability analysis based on Binary Decision Diagram (BDD) consists of three steps: edge ordering, BDD generation and BDD evaluation. The BDD generation process using edge expansion technique should recursively decompose the network and construct the edge expansion subnet in a top-down manner. There is large number of useless or redundant subnets generated in this decomposition, which causes numerous inefficient computations. Thus, it is extremely important to optimize the edge expansion technique. In this paper, the notation of useless edge expansion and redundant edge expansion is formally defined. The original reason of them being created is identified, and the improvement algorithms based on graph traversal are used to eliminate all these inefficient edge expansion. According to the experimental data, compared with the unimproved BDD generation process, our proposal can dramatically reduce the running time and memory usage and makes possible the analysis of large network.**

Index Terms—**Binary Decision Diagram (BDD), Network Reliability, Edge Expansion**

I. Introduction

Many complex physical, technological, social, biological and economical systems can be represented in the form of networks, where vertices are the entities of the system and the edges represent the relational links among the entities. Since complex networks display a high degree of tolerance to random failures, errors and attacks, network reliability has always been a challenging task in system reliability analysis for last three decades. Reliability means probability of success between two nodes connected by several branches. Knowing the reliability of each branch, one can determine the terminal reliability between any two nodes.

The network reliability has been approached and resolved with different methods in the literature. Two kinds of computations exist for network reliability: exact, and approximate. The exact methods provide an exact reliability; in contrast, the simulation methods provide an approximate result.

In literature, two classes of exact methods are often used to compute the network reliability. The first class deals with the enumeration of all minimal paths or cuts. A *path* is defined as a set of network components so that if these components are all reliable, the system is up. A path is *minimal* if it has no proper sub-paths. Conversely, a *cut* is a set of network components such that if these components fail, the system is down. If the probability of failure of nodes and edges is known, the reliability expression can be calculated using different techniques, like *inclusion-exclusion* or *sum of disjoint products (SDP)* methods because this enumeration provides non-disjoint events. Numerous works related to this kind of analysis have been done, [1]-[3]. However, in min-paths and min-cuts methods, as the number of edges become large, the number of minimal paths and cuts will be large and SDPs or inclusion-exclusion expression for min-paths and min-cuts will be increased too large to be stored and be processed.

In the second class, the algorithms are based on reducing process in graph topology. The first process consists of reducing the graph by replacing some special substructures by smaller ones. The replaced substructures can be elementary, such as two adjacent edges (series-parallel reductions), or more complex as a sub-graph [4], and delta-to-star reductions [5]. Thee reductions allow us to compute the reliability of series-parallel networks in linear time; the reductions are recursively applied until resulting in a single edge. Nevertheless, for general networks, such substructure can not be found. In this case, the factoring process is applied. The idea is to choose a component, and decompose the problem into two sub-problems: the first assumes the component has failed, and the second assumes it is functioning. Satarayana & Chang [6], and Wood [7], have shown that the factoring algorithms with reductions are more efficient at solving this problem than the classic path or cut enumeration methods. In factoring algorithm the number of factored reliability graphs will increase exponentially with the number of edges increases thereby computational time becomes prohibitive for large networks.

The increased complexity of the real-life networks such as infrastructure and lifeline networks requires new analytical methods to evaluate their behavior. In the last decades, Binary Decision Diagrams (BDD) has provided an extraordinary efficient method to represent and manipulate Boolean functions. Akers [8] proposed BDD as a powerful representation of truth tables with an easy implementation. Since his work, there has been a tremendous amount of literature in which BDD have been utilized for different applications. Bryant [9] applied the BDD concept to Boolean function manipulation. Rauzy [10], Coudert & Madre [11] and Sinnamon & Andrews [12] applied BDD to reliability analysis of fault trees.

To the best of our knowledge, two classes of algorithms have been proposed in the literature to derive directly the reliability function in the form of a BDD, without deriving the corresponding logical expression. The first class of algorithms presented by Kuo et al. [18]-[20] where the success path function of a given graph is constructed based on BDD by traversing the graph with edge expansion diagram and the graph reliability is obtained by directly evaluating on this BDD recursively. Performing several case studies, they showed that their algorithm has an efficient performance even for a *2×100* lattice network. The second class of algorithms presented by Sekine & Imai[13]-[15], revised by Hardy et al. [16], [17], are based on a suitable factorization algorithm, consisting in pivoting along a complete sequence of edges and deriving the right sub-graph when the edge functions and the left sub-graph when the edge fails. Both papers indicate a lattice of *12×12* (*144 nodes and 264 edges*) as an upper limit to computational power of the method.

In this paper, we focus on the first line of research, i.e., improve the performance of network reliability analysis based on BDD and edge expansion technique. The aim of this paper is to try to answer following question:

Question: will there be useless or redundant expansion by applying the edge expansion technique? What is the original reason of these useless or redundant expansions being created? And how can we remove them efficiently?

Compared with previous work on the network reliability analysis based on BDD and edge expansion technique [18]-[20], this research has the following advantages:

Advantage #1: The notation of useless edge expansion is formally defined. The original reason of them being created is identified, and the connectivity testing algorithm (*i.e., graph traversal algorithm*) is used to eliminate all the useless edge expansion.

Advantage #2: The notation of redundant edge expansion is formally defined. The original reason of them being created is identified, and a modified depth first traversal algorithm is used to eliminate all the redundant edge expansion.

The rest of this paper is organized as follows: Section II gives a brief introduction to terminal pair reliability of network and binary decision diagram. Section III introduces the edge expansion technique and formally defines the notation of useless edge expansion and redundant edge expansion. Section IV introduces the method to eliminate all the useless edge expansion and proposes our experimental results to show the significant performance improvement. Section V introduces the method to eliminate all the redundant edge expansion and illustrates its efficiency by our experimental results. Finally, section VI concludes our paper.

II. Preliminaries

A. Terminal-pair reliability

Assume the natural numbers set ***V*** is finite and non-empty. ***E*** is a relationship between ***V*** and ***V***. *P* is a function from ***E*** to the interval *(0, 1), P:* ***E****->(0, 1).*

Definition 1 (network): A two-tuple *G= (V, E)* is a network, if

- *V*Í ***V*** is a node set
- *E*Í ***E*** is a edge set.

For convenience, we name the node pair <*a*, *b*> as edge e_i.

Definition 2 (terminal-pair network): A four-tuple *G= (V, E, s, t)* is a terminal-pair network, if

- *V*Í ***V*** is a node set
- *E*Í ***E*** is a edge set.
- $s \in V$ is the source node
- $t \in V$ is the sink node.

The terminal-pair network *G* in Fig.1 is correspond to the four-tuple ($\{ n_0, n_1, n_2, n_3, n_4, n_5 \},\{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9\}, n_3, n_5$).

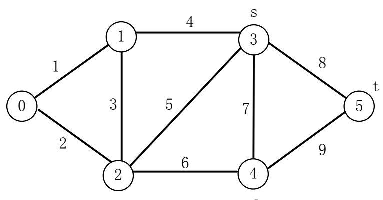


Figure1. Network G

Each edge e_i of the network *G* is subject to failure with known probability q_i ($q_i \in [0, 1]$). The probability that edge e_i functions can be obtained from $p_i = 1 - q_i$. In the following, we consider the nodes as perfect. In classical enumerative methods, all the states of the network are generated, evaluated as a failing state or as a functioning state, then probabilistic methods are used to compute the resulting reliability. Since there are two states for each edge, there are 2^m ($m=|V|$) possible states for the network *G*. Let *Xi* be the binary random variable state of the edge e_i in *G*, defined by $X_i = 1$ when edge e_i is operational, and when $X_i = 0$ edge e_i is down. $X = (X_1, X_2,...,X_m)$ is the random network state vector. A state *x* of *G* is denoted by $x = (x_1, x_2, ..., x_m)$ where x_i stands for the state of edge e_i, $x_i = 0$ if e_i is down and $x_i = 1$ if it works. Probability of *x* is can be computed by:

$$\Pr(X = x) = \prod_{i=1}^{m} (x_i \cdot p_i + (1 - x_i) \cdot q_i) \quad (1)$$

The terminal-pair network reliability is defined by:

$$R_K(p;G) = \sum_{x \text{ is a functioning state}} \Pr(X = x) \quad (2)$$

B. BDD

BDD is a compact encoding of the truth tables of Boolean formulae. The BDD representation is based on the Shannon decomposition: Let *F* be a Boolean function that depends on the variable *x*, then the following equality holds.

$$\begin{aligned} F &= (x \wedge F[x \leftarrow 1]) \vee (\neg x \wedge F[x \leftarrow 0]) \\ &= ite(x, F[x \leftarrow 1], F[x \leftarrow 0]) \end{aligned} \qquad (3)$$

BDD has two sink nodes, labeled *0* and *1*, representing the two corresponding constants *0* and *1*. Each non-sink node is labeled with a Boolean variable *x* and has two outgoing edges that represent the two corresponding expressions in the Shannon decomposition. These two edges are called *0-edge* (or *else-edge*) and *1-edge* (or *then-edge*), respectively. The node linked by the *1-edge* represents the Boolean function *F[x←1]*; *0-edge* represents the Boolean function *F[x←0]*. Thus, each non-sink node in a BDD encodes an *ite* format.

An ordered BDD is a BDD with the constraint that the variables are ordered and every *source-to-sink* path in the ordered BDD visits the variables in ascending order. A reduced ordered BDD is an ordered BDD where each node represents a distinct Boolean expression.

In the Boolean function to equivalent BDD conversion, following ordinary BDD operation BDDOp is recursively performed on the higher priority edge x as

$$\begin{aligned} & BDDOp(F, G, <op>) \\ & = BDDOp(ite(x, F_1, F_0), ite(y, G_1, G_0), <op>) \\ & = \begin{cases} ite(x, BDDOp(F_1, G_1, <op>), BDDOp(F_0, G_0, <op>)) & if\ x = y \\ ite(x, BDDOp(F_1, G, <op>), BDDOp(F_0, G, <op>)) & if\ x < y \end{cases} \end{aligned} \qquad (4)$$

where <*op*> is an *AND* or *OR* Boolean operator.

Fig.2 shows the reduced ordered BDD for the Boolean function *(x₁∧x₂) ∨(¬x₁∧x₃)* under the orderings $x_1 < x_2 < x_3$.

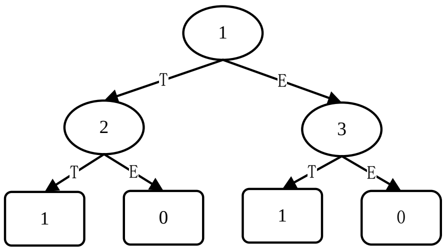


Figure2. The equivalent BDD for *(x1∧x2) ∨(¬x1∧x3)*

The reliability measure can be calculated by recursively evaluating the probabilities of all the nodes in the BDD as

$$\Pr(F) = \Pr(ite(x, F_1, F_0)) = p * \Pr(F_1) + (1-p) * \Pr(F_0) \qquad (5)$$

where *p* is the successful probability of variable *x*.

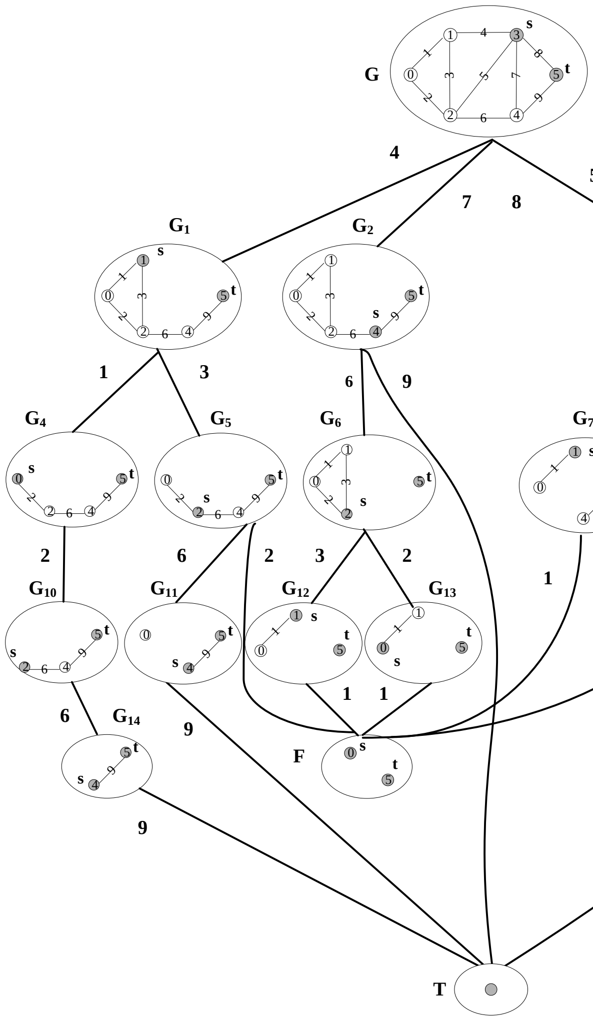


Figure3. Edge expansion diagram of *G*

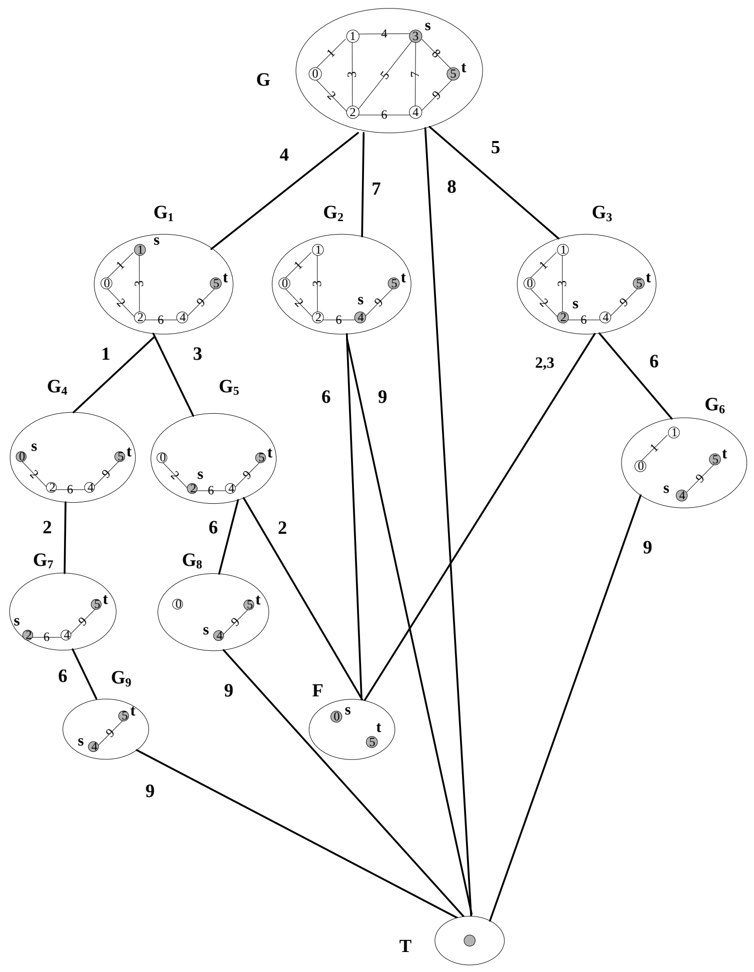


Figure4. Edge expansion diagrams of *G* with the useless subnet removement

III. Edge expansion Subnet

The network reliability analysis basing on edge expansion should recursively decompose the network and construct the edge expansion subnet in a *top-down* manner.

Definition 3 (edge expansion subnet): Given a terminal-pair network *G=(V, E, s, t),* for any source pair *<s, s'>* belonging to the edge set *E*, the network *G* has a edge expansion subnet *G'=(V', E', s', t)*, where

- *V'=V-{s}*
- *E'=E-{(x, y) | (x, y)∈E, x=s ∨ y=s}.*

Assume the edge e_i (*i=1, 2, ..., k*) is connective to the source *s*. Traverse the network with e_i (*i=1, 2, ..., k*) and constructing *k* subnets $G{*}e_i$ (*i*=1,2,...,*k*). According to the definition 3, the construction rule is merging the source *s* and the other end of e_i. Meanwhile, delete all the edges connecting to the source *s* originally.

The edge expansion process of the network *G* in Fig.1 is shown in Fig.3. For example, by expanding the edge e_4, we can get the subnet $G_1=(\{ n_0, n_1, n_2, n_4, n_5 \},\{e_1, e_2, e_4, e_5, e_9\}, n_1, n_5)$. But when it turns to e_9, the corresponding subnet $T=(\{ n_0, n_1, n_2, n_4, n_5 \}, \{e_1, e_2, e_4, e_5, e_9\}, n_5, n_5)$.

From the edge expansion diagram, we can find two kinds of special subnets. They can be defined formally as follows:

Definition 4 (s-t-combined edge expansion subnet): A edge expansion subnet *G'=(V', E', s', t)* is *s-t-combined* s-ubnet, if *s'= t.*

For convenience, we denote all *s-t-combined* subnets as *T*.

Definition 5 (s-isolated edge expansion subnet): A ed-ge expansion subnet *G'=(V', E', s', t)* is *s-isolated* subnet, if

- *G'* is not a *s-t-combined* subnet
- *Degree(s') =0*, where *Degree(s')* is the number of nodes connected to *s'*.

For convenience, we denote all *s-isolated* subnets as *F*.

With the definition of *s-isolated* subnet, useless and redundant edge expansion subnets can be defined as follows:

Definition 6 (useless edge expansion subnet): A edge expansion subnet *G'=(V', E', s', t)* is useless, if

- *G'* is not a *s-isolated* subnet
- no *s-t-combined* subnet *T* can be expanded from it.

Consider edge expansion diagram shown in Fig.3 of the network *G* in Fig.1. Subnets G_6, G_7, G_8, G_{12}, and G_{13} are useless.

Denote all the expansion paths leading to *s-t-combined* subnet *T* starting from *Gi'* with *PSi*, and all the paths leading to *s-t-combined* subnet *T* starting from *Gj'* with *PSj*. For any path *Pi* in *PSi*, there is a path *Pj* in *PSj* and that: any subnet *G'* (after discarding those nodes with 0 degree) included by *Pi* has an identical subnet *G''* (after discarding those nodes with 0 degree) in *Pj*, and vice versa. Then, we say *PSi* and *PSj* are the equivalent.

Definition 7 (redundant edge expansion subnet): Given two edge expansion subnet G_i' and G_j', if the two set of expansion paths leading to *s-t-combined* subnet *T* starting from *Gi'* and *Gj'* are equivalent, these two subnets are redundant with each other.

Consider edge expansion process shown in Fig.3 of the network *G* in Fig.1. Subnets G_5 and G_{10} are redundant with each other, and subnets G_9, G_{11}, and G_{14} are redundant with each other.

IV. Removement of Useless Subnets

In order to improve the performance of reliability analysis, some removement algorithms should be designed to fulfill the elimination of the computation of useless edge expansion subnets.

By checking these useless subnets we can simply find that: the original reason of these useless subnets being created is that they are *s-t-unconnected* subnets. A network is said to be *s-t-unconnected* if the network does not contains a path from source *s* to sink *t*.

Table I gives the classic graph depth first traversal algorithm. Depth first traversal tends to create very long, narrow trees. It is a generalization of preorder traversal. Starting at some node *n*, we process *n* and then recursively traverse all nodes adjacently. This traversal creates a spanning tree that can be used to determine if a network is *s-t-unconnected*.

TABLE I.
GRAPH DEPTH FIRST TRAVERSAL ALGORITHM

```
1  Algorithm DFS(G, s):
2      mark s
3      for all nodes v connected to s do
4          if v is not marked:
5              DFS (G, v)
```

Using the depth first traversal algorithm *DFS*, we can determine if a subnet *G'* is useless or not. If *G'* is useless (*i.e., s-t-unconnected*), the *s-isolated* subnet *F* is used to replace it and the following expansion starting from *G'* can be avoided. With the removement of useless subnets, the edge expansion process for the network in Fig.1 is illustrated in Fig.4. Notice that there is no useless subnet at all.

In order to study the performance improvement achieved by the removement of useless subnets, we choose *3×N* lattice networks to analyze the terminal-pair network reliability. All network nodes are numbered *left-to-right, top-to-bottom,* starting with 0. Assume that source *s* has a number *m*n/2-2* and the sink *t* has a number *m*n/2*. We use the breadth-first search strategy to get the edge ordering.

The experimental results of the *3×N* network are listed in Table II. The columns give the value of network size parameter *N*, the calculated reliability measures (*Relib*), the performance improvement measure $\varDelta T/T_1$, $\varDelta T=T_1-T_2$, where T_1 is the running time of reliability analysis without any improvement and T_2 is the running time with the useless subnet removement, and $\varDelta NG/NG_1$, $\varDelta NG=NG_1-NG_2$, where NG_1 is the number of generated subnets without any improvement and NG_2 is the number of generated subnets with the useless subnet removement. From the results in Table II, it can be concluded that:

Conclusion #1: The application of useless subnet removement technique can significantly reduce the running time of reliability analysis and the number of generated edge expansion subnets. With the increase of network size, the performance improvement measure $\varDelta T/T_1$ and $\varDelta NG/NG_1$ increases quickly from 3.13% and 20.00% to 77.84% and 81.06%.

TABLE II
PERFORMANCE IMPROVEMENT WITH THE USELESS SUBNET REMOVEMENT

Net	Relib	⊿T/T1 (%)	⊿NG/NG1(%)
5	0.99651766656	3.13	20.00
6	0.99669526401	12.06	31.17
7	0.99687151399	19.77	40.56
8	0.99688815593	32.35	49.46
9	0.99690478515	41.87	54.30
10	0.99690639111	50.83	61.21
11	0.99690799696	56.98	65.69
12	0.99690815233	63.42	70.67
13	0.99690830769	69.32	74.47
14	0.99690832273	74.00	78.04
15	0.99690833776	77.84	81.06

TABLE III
REDUNDANT NODE REMOVEMENT ALGORITHM

```
1  Algorithm RemoveNode(G, s):
2      mark s
3      for all nodes n connected to s do
4          if n is not marked:
5              RemoveNode (G, n)
6      if G. degree (n)=1
7          delete n
```

V. Removement of Redundant Subnets

By checking the edge expansion diagram in Fig.4, we can find that subnets G_5 and G_7 are redundant with each other, and subnets G_6, G_8, and G_9 are redundant with each other. That is to say, they have the same reliability measures, though they have different structures. Further, we can simply find that: the original reason of this kind of redundant expansion being created is the appearance of redundant nodes included by subnets.

Definition 8 (redundant node): A node n of *G=(V, E, s, t)* is redundant, if

- $n \neq s$ and $n \neq t$
- *Degree(n)=1.*

In order to improve the performance of reliability analysis, some removement algorithm should be designed to remove all the redundant nodes included by a subnet and make possible to merge redundant subnets.

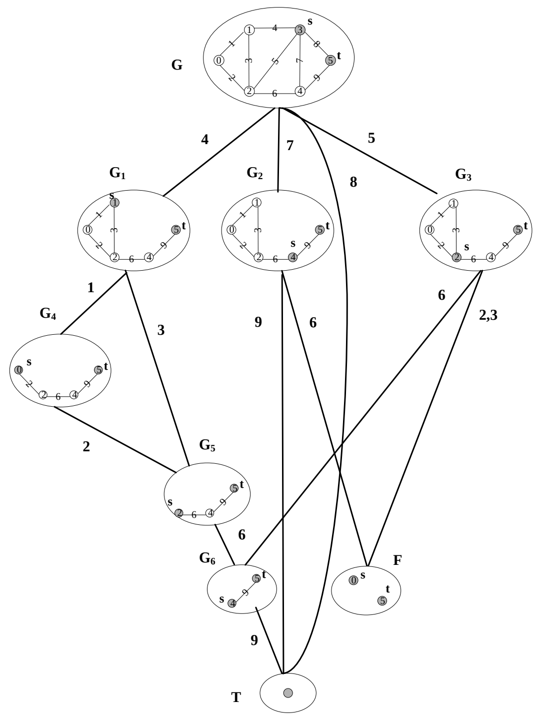


Figure5. Edge Expansion Diagrams of *G* with redundant subnet and node removement

Table III gives the *RemoveNode* algorithm, slightly revised from the graph depth first traversal algorithm, to remove all the redundant nodes within one-pass traversal.

With the removement of both useless subnets and redundant subnets, the edge expansion process for the network in Fig.1 is illustrated in Fig.5. Notice that the redundant subnets are identified and merged. The total number of subnets decreases from 12 to 9.

Now, consider the performance improvement achieved by redundant subnet and redundant node removement. The experimental results of the $3 \times N$ network are listed in Table IV. From the results in Table IV, it can be concluded that:

Conclusion #2: By removing all the redundant nodes included by a subnet to enhance the subnet isomorphism, the performance can be improved further. With the increase of network size, the performance improvement measure $\triangle T/T_1$ and $\triangle NG/NG_1$ increases quickly from 35.02% and 66.25% to 99.33% and 99.84%. It makes possible the analysis of large network.

TABLE IV
PERFORMANCE IMPROVEMENT WITH REDUNDANT SUBNET AND NODE REMOVEMENT

Net	Relib	$\triangle T=T_1-T_2$ (%)	$\triangle NG/NG_1$(%)
5	0.99651766656	35.02	66.25
6	0.99669526401	35.08	76.11
7	0.99687151399	58.55	86.55
8	0.99688815593	66.58	91.37
9	0.99690478515	81.00	94.83
10	0.99690639111	86.64	97.11
11	0.99690799696	92.96	98.22
12	0.99690815233	95.23	99.08
13	0.99690830769	97.68	99.44
14	0.99690832273	98.61	99.72
15	0.99690833776	99.33	99.84

VI. CONCLUSION

Edge expansion technique is one widely-used BDD-based network reliability analysis technique proposed by Kuo et al. Experiments have shown that the terminal-pair network reliability analysis method based on edge expansion has a lot of advantages, such as more efficient calculation, more accurate results and easy implementation.

The network reliability analysis basing on edge expansion should recursively decompose the network and construct the edge expansion subnet in a top-down manner. There is large number of useless or redundant subnets appearing in this decomposition. In order to avoid the numerous inefficient computations and improve the performance, it is extremely important to optimize the edge expansion technique. In this paper, the notation of useless edge expansion and redundant edge expansion is formally defined. The original reason of them being created is identified, and the removement algorithm based on graph traversal algorithm is used to eliminate all these inefficient edge expansion. According to the experimental data of the $3\times N$ network, with the increase of network size, the performance improvement of running time and number of subnets increases quickly from 3.13% and 20.00% to 77.84% and 81.06%. It makes possible the analysis of large network.

ACKNOWLEDGEMENT

This work was supported in part by the NSF of China under Grant No. 61272130, the NSF of Zhejiang Province under Grant No. Y1100689, the China Scholarship Council under Grant No. 2011833135.

REFERENCES

[1] Ahmad, S.H. "Simple enumeration of minimal cutsets of acyclic directed graph," IEEE Trans. Reliability, R-27(5): 484–487, 1988.

[2] Locks, M.O. "A minimizing algorithm for sum of disjoint products," IEEE Trans. Rel., R-36(4): 436–445, 1987.

[3] Hariri S. & Raghavendra, C. S. SYREL: "A symbolic reliability algorithm based on path and cutset methods," IEEE Trans. Comput., C-36(10): 1224–1232, 1987.

[4] Choi, M.S. & Jun, C.H. "Some Variant of Polygon-Chain- Reductions in Evaluating Reliability of Undirected Net-work," Microelectron. Reliab., 35(No1): 1-11, 1995.

[5] Gadani, J.P. "System Effectiveness Evaluation using Star and Delta Transformations," IEEE Trans. Rel., R-30(No1): 43-47, 1981.

[6] Satyanarayana, A. & Chang, M. K. "Network reliability and the factoring theorem. Networks,"13:107–120, 1983.

[7] Wood, R.K. "A factoring Algorithm using Polygon-to-Chain Reductions for Computing k-Terminal Network reliability,"Networks, 15: 173-190,1985.

[8] Akers, S.B. "Binary Decision Diagrams," IEEE Trans. Comput. C-27(6): 509-516,1978.

[9] Bryant, R. "Graph Based Algorithms for Boolean Function Manipulation," IEEE Trans. Comput., 35(8): 677-691, 1986.

[10] Rauzy. A. "New Algorithms for Fault Tolerant Trees Analysis," Reliability Engineering and System Safety, 50(59): 203-211, 1993.

[11] Coudret, O. & Madre, J.C. Metaprime: "An Interactive Fault-Tree Analyzer," IEEE Trans. Rel., 43(1): 121-127, 1994.

[12] Sinnamon, R.M. & Andrew, J.D. "Improved Accuracy in Quantitative Fault Tree Analysis," Quality and Reliability Engineering International, 13: 285-292, 1997.

[13] Sekine, K. & Imai, H. "A unified approach via BDD to the network reliability and path number," Technical Report TR-95-09, Department of Information Science, University of Tokyo, 1995.

[14] K. Sekine and H. Imai, Computation of the network reliability (extended abstract).Technical report, Department of Information Science,University of Tokyo, 1998.

[15] H.Imai, K. Sekine, and K. Imai, "Computational investigations of all termina network reliability via BDDs," IEICE Transactions on Fundamentals vol. E82-A, no. 5, pp. 714–721, May 1999.

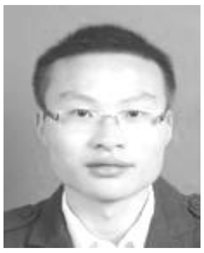
[16] G. Hardy, C. Lucet, and N. Limnios, "Computing all-terminal reliability of stochastic networks with binary decision diagrams," 11th International Symposium on Applied Stochastic Models and Data Analysis, May. 2005.

[17] G. Hardy, C. Lucet, and N. Limnios, "K-Terminal Network Reliability Measures With Binary Decision Diagrams," IEEE Trans. Reliability, vol. 56 no. 3, pp.506 – 515, 2007.

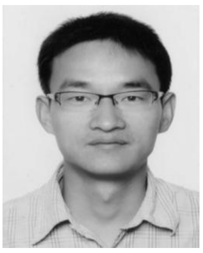
[18] F.-M. Yeh and S.-Y. Kuo, "OBDD-based network reliability calculation," Electronics Letters, vol. 33, no. 9, pp. 759–760, Apr. 1997.

[19] S. Y. Kuo, S. K. Lu, and F. M. Yeh, "Determining terminal-pair network reliability based on edge expansion diagrams using OBDD," IEEE Trans. Reliability, vol. 48, no. 3, pp. 234–246, Sep. 1999.

[20] F.-M. Yeh, S.-K. Lu, and S.-Y. Kuo,"OBDD-based evaluation of k-terminal network reliability," IEEE Trans. Reliability, vol. R-51, no. 4, 2002.



Chen Ronggen is a postgraduate in the College of Mathematics, Physics and Information Engineering at the Zhejiang Normal University. He was born in Shaoxing, China, 1988. He received his B. Sc. degree in the College of Mathematics and Physics at the Lishui University. His research interest is analysis of complex systems and networks using combinatorial models.



Mo Yuchang is an Associate Professor in the College of Mathematics, Physics and Information Engineering Director of the Dependable Computing Lab and at the Zhejiang Normal University. He received his B. Sc., M. Sc. and Ph. D. degrees in computer science from Harbin Institute of Technology, Harbin, China, in 2002, 2004 and 2009, respectively. He is currently a Visiting Scholar with the Department of Electrical and Computer Engineering, University of Massachusetts (UMass), Dartmouth, North Dartmouth. He has published papers on IEEE Transactions on Reliability, Reliability Engineering and System Safety, Quality and Reliability Engineering International, and others. His research has been supported by the National Science Foundation of China and Zhejiang Province, and Sci&Tech Development Plan of Zhejiang Province. His research interests include dependable computing and networking, fault tolerant computing, and reliability analysis of complex systems and networks using combinatorial models.

Pan Zhusheng is a lecture in the College of Mathematics, Physics and Information Engineering at the Zhejiang Normal University. He received his M. Sc. degree in computer science from Shanghai University, Shanghai, China, 2005. His research interests include reliability analysis of complex systems and networks and embedded system.