A Parallel Manipulation Method for Zero-suppressed Binary Decision Diagram

Jin Wang, Shanqi Chen, Liqin Hu*, Rongxiang Hu, Fang Wang, FDS Team
Institute of Nuclear Energy Safety Technology, Chinese Academy of Sciences, Hefei Anhui China

liqin.hu@fds.org.cn

Abstract: A parallel algorithm for the manipulation of Zero-suppressed Binary Decision Diagrams (ZBDDs) on a shared memory multi-processor system was described. Theoretical analysis showed that parallel manipulation of ZBDD has a better time performance than the sequential operation of ZBDD. Since the parallel ZBDD algorithm uses much less time than the ordinary ZBDD algorithm, a real-time calculation can be done in the risk monitoring of a nuclear power plant, which would do a favor of accelerating emergency response and improving safety of nuclear power plants.

Key words: Parallel, Zero-suppressed Binary Decision Diagram, Reliability, Probabilistic Safety Assessment

1. INTRODUCTION

Fault tree analysis is one of the main methods used in Probabilistic Safety Assessment (PSA) of a nuclear power plant. For solving a large fault tree, many new methods were introduced in the past years, such as Binary Decision Diagram (BDD) [1-3], Zero-suppressed Binary Decision Diagram (ZBDD) [4-6], variable ordering [7-10] and truncation technology [11], functional decomposition [12]. BDD Algorithm made the large fault tree analysis based on computer possible, and ZBDD algorithm improved the calculation speed greatly. However, in some cases like risk monitor which needs a real-time analysis, traditional ZBDD algorithm still need further improvement to meet such demand.

ZBDDs are very efficient representations of a factorized structure of minimal cut sets (MCSs), and are widely used for solving a fault tree. The ZBDD algorithm is known as an efficient replacement of a cutset-based algorithm that is based on traditional Boolean algebra, since that logic operations on logic functions, such as AND and OR, are reduced to operations on ZBDDs with a set of new ZBDD operation formulae developed in 2004 by Woo Sik Jung. However, operations on ZBDDs are time-consuming in some cases, and a fast manipulation method is needed.

Algorithm implemented by serial program traditionally cannot be accelerated in Multi-core computer widely used nowadays, so hardware performance advantage was constrained. There are two approaches transforming serial program to parallel program, one of which is transformed automatically by complier and the other is re-write the code. The second one was adopted by the paper because it can adapt the hardware environment of Multi-core computer more appropriately. A parallel ZBDD manipulation method was proposed and its performance was theoretically analyzed. This method would be applied in the Reliability and Probabilistic Safety Analysis Program RiskA [13-21] in later work.

2. TRADITIONAL MANIPULATION

ZBDD was proposed by S. Minato, and a set of new operation formulas shown below based on ZBDD was developed by Woo Sik Jung. In the four formulas (1)~(4), $Y=ite(y,Y_1,Y_2)$ where the variable ordering of $y>x$. X_1 and Y_1 are the left sub-trees of ZBDD, and X_2 and Y_2 are the right sub-trees.

$$ite(x,X_1,X_2)ite(x,Y_1,Y_2)=ite(x,(X_1Y_1+X_1Y_2+X_2Y_1),X_2Y_2) \quad (1)$$

$$ite(x,X_1,X_2)+ite(x,Y_1,Y_2)=ite(x,X_1+Y_1,X_2+Y_2) \quad (2)$$

$$ite(x,X_1,X_2)ite(y,Y_1,Y_2)=ite(x,X_1Y,X_2Y) \quad (3)$$

$$ite(x,X_1,X_2)+ite(y,Y_1,Y_2)=ite(x,X_1,X_2+Y) \quad (4)$$

As with the manipulation, the construction of a ZBDD from a fault tree was focused on. Transformation from basic events in fault tree to variable in ZBDD was done one by one from bottom to up as post-order traversal in traditional implementation of ZBDD algorithm.

Take the fault tree illustrated in Fig.1 for example, all fault tree nodes form a sequence as D, E, G4, A, B, G2, C, F, G, G5, G3, G1 according to the post-order traversal. Based on this sequence, ZBDD manipulations were done from D to G1, and the construction of a ZBDD from the whole fault tree was completed when the last operation on G1 was made. Suppose the variable ordering was A<B<C<D<E<F<G, and the corresponding variable in ZBDD was identified as a, b, c, d, e, f, g. The operation sequence would be shown by a set of expressions below. The result shown in Fig.2 of the last operation on G1 was the corresponding ZBDD of fault tree in Fig.1.

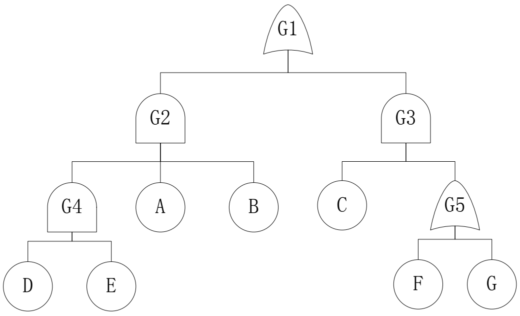

Fig.1 An example of a fault tree

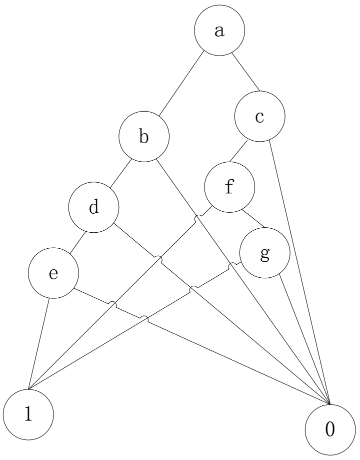

Fig. 2 ZBDD of the fault tree in fig. 1

Table 1: ZBDD operation sequence for the fault tree in fig.1

Order	Fault tree node	ZBDD operation
1	D	ite(d,1,0)
2	E	ite(e,1,0)
3	G4=DE	ite(d,1,0) ite(e,1,0)=ite(d,ite(e,1,0),0)
4	A	ite(a,1,0)
5	B	ite(b,1,0)
6	G2=G4AB	ite(d,ite(e,1,0),0) ite(a,1,0) ite(b,1,0)=ite(a,ite(b,ite(d,ite(e,1,0),0),0),0)
7	C	ite(c,1,0)
8	F	ite(f,1,0)
9	G	ite(g,1,0)
10	G5=F+G	ite(f,1,0)+ite(g,1,0)=ite(f,1,ite(g,1,0))
11	G3=CG5	ite(c,1,0) ite(f,1,ite(g,1,0))=ite(c,ite(f,1,ite(g,1,0)),0)
12	G1=G2+G3	*ite(a,ite(b,ite(d,ite(e,1,0),0),0),0)+ite(c,ite(f,1,ite(g,1,0)),0)=ite(a,ite(b, ite(d,ite(e,1,0),0),0),ite(c,ite(f,1,ite(g,1,0)),0))*

In this case, suppose each operation lasted t_i (i = 1, 2…, 12) seconds, so the whole process of ZBDD construction needed t seconds which can be obtained by the following formulae.

$$t = \sum_{i=1}^{12} t_i \tag{5}$$

3. PARALLEL MANIPULATION

As mentioned in section 2, we focus on the construction of a ZBDD from a fault tree, and devised a parallel algorithm for the construction. The construction was just done through a set of operations on ZBDDs. In the construction, there are many logic operations to be processed, and some of them can be processed in parallel. At first, we introduce an extraction method and a parallel-execution method for such parallelizable operations. This is the parallel execution method for an operation sequence (or a set of operations). To extract more parallelism, we introduce a dynamic expansion method of a logic operation. The dynamic expansion is a method to obtain sub-operations from ZBDD operation formulae. These sub-operations are executed in parallel and the results of these sub-operations are merged to obtain the result of the original operation.

The parallel execution method was to construct ZBDD from the fault tree layer by layer from bottom to top. We also take the fault tree in fig.1 for example. The whole tree could be divided into four layers which were {D, E, F, G}, {G4, A, B, C, G5}, {G2, G3} and {G1} shown in fig.3. So the construction could be done by four steps: in step 1, basic events D, E, F and G were transformed to ZBDD in parallel, which were done as operation 1, 2, 8 and 9 in table 1; in step 2, G4, A, B, C and G5 were processed to construct ZBDD in parallel, which were done as operation 3, 4, 5, 7 and 10 in table 1; in step 3, G2 and G3 were processed to construct ZBDD in parallel, which were done as operation 6 and 11; in the last step, G1 was processed to construct the whole ZBDD as shown in fig.2, which was corresponding to the last operation in table 1.

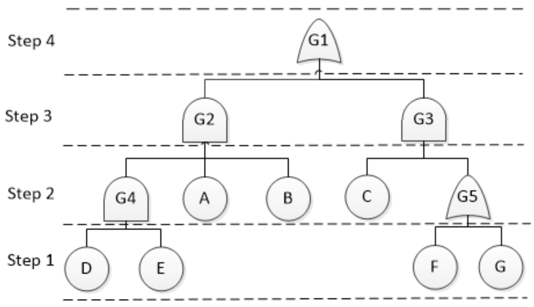

Fig. 3 Steps of the parallel algorithm

In each operation from fault tree to ZBDD, some sub-operations could be done in parallel too. In the four formulas (1)~(4), for example, in the first one, the sub-operation like X_1Y_1, X_1Y_2, X_2Y_1 and X_2Y_2 could be done in parallel.

If suppose each operation lasted t_i' (i = 1, 2…, 12) seconds, so $t_i' \leq t_i$ because of the parallel of the sub-operations of each operation. Then, step 1 would last $max\{t_1',t_2',t_8',t_9'\}$, while step 2 lasting $max\{t_3',t_4',t_5',t_7',t_{10}'\}$, step 3 lasting $max\{t_6',t_{11}'\}$, step 4 lasting t_{12}'. So the whole process of ZBDD construction needed t' seconds which can be obtained by the following formulae.

$$t'=max\{t_1',t_2',t_8',t_9'\}+max\{t_3',t_4',t_5',t_7',t_{10}'\}+max\{t_6',t_{11}'\}+t_{12}' \quad (6)$$

$$t' \leq max\{t_1,t_2,t_8,t_9\}+max\{t_3,t_4,t_5,t_7,t_{10}\}+max\{t_6,t_{11}\}+t_{12}$$
$$<\{t_1+t_2+t_8+t_9\}+\{t_3+t_4+t_5+t_7+t_{10}\}+\{t_6+t_{11}\}+ t_{12}=t \quad (7)$$

That is, $t' < t$ was established.

4. SUMMARY

A parallel manipulation method for ZBDD was presented in the paper, and theoretical analysis demonstrated the higher efficiency of such method. Our parallel algorithm would be implemented in C++ on a shared memory multi-processor system, and its efficiency would be demonstrated quantificationally by performing benchmark tests in future work.

Acknowledgments

This work was supported by the Strategic Priority Research Program of Chinese Academy of Sciences (No. XDA03040000), the National Natural Science Foundation of China (No. 91026004), the Informatizational Special Projects of Chinese Academy of Sciences (No. XXH12504-1-09), and the Foundation of President of Hefei Institutes of Physical Science (No. YZJJ201327).

References

[1] C. Lee, "Representation of switching circuits by binary-decision programs," BELL SYSTEM TECHNICAL JOURNAL, **38**, 985-999 (1959).

[2] S. B. Akers, "Binary decision diagrams," IEEE Transactions on Computers, **c-27**, 509-516 (1978).

[3] A. Rauzy, "New Algorithms for Fault-Trees Analysis," Reliability Engineering & System Safety, **40**, 203-211 (1993).

[4] S. Minato, "Zero-suppressed BDDs for set manipulation in combinatorial problems," in DAC '93: 30th ACM/IEEE-CS Design Automation Conference, Dallas, TX, 272-277(1993).

[5] J. Woo Sik, H. Sang Hoon, and H. Jaejoo, "A fast BDD algorithm for large coherent fault trees analysis," Reliability Engineering & System Safety, **83**, 369-374 (2004).

[6] J. Woo Sik, "ZBDD algorithm features for an efficient Probabilistic Safety Assessment," Nuclear Engineering and Design, **239**, 2085-2092 (2009).

[7] P. Liu, Y. C. Wu, Y. Z. Li, et al., "An ordering scheme of the basic events based on zero-suppressed binary decision diagrams for the large-scale fault tree analysis," Chinese Journal of Nuclear Science and Engineering, **27**, 282-288, (2007).

[8] C. Ibanez-Liano, A. Rauzy, E. Melendez, et al., “Variable ordering techniques for the application of Binary Decision Diagrams on PSA linked Fault Tree models”, Safety, Reliability and Risk Analysis: Theory, Methods and Applications, **1-4**, 2051-2059 (2009).

[9] Y. Z. Li, J. Wang, L. Q. Hu, et al., "A Variable Ordering Heuristic based on Zero-suppressed Binary Decision Diagrams," in Proceedings of International Conference on Nuclear Engineering, Xi'an, Shanxi, China, 691-695 (2010).

[10] J. Wang, F. Wang, J. Q. Wang, et al., " A variable ordering heuristic for risk monitors based on zero-suppressed binary decision diagram," Chinese Journal of Nuclear Science and Engineering, **30**, 360-364 (2010).

[11] W. S. Jung, S. H. Han, and J. E. Yang, "FAST BDD TRUNCATION METHOD FOR EFFICIENT TOP EVENT PROBABILITY CALCULATION," Nuclear Engineering and Technology, **40**, 571-580 (2008).

[12] S. Contini and V. Matuzas, "Analysis of large fault trees based on functional decomposition," Reliability Engineering & System Safety, **96**, 383-390 (2011).

[13] Y. C. Wu, P. Liu, L. Hu, et al., “Development of an integrated probabilistic safety

assessment program," Chinese Journal of Nuclear Science and Engineering, **27(3)**, 270-276 (2007).

[14] M. Nie, Y. C. Wu, X. J. Deng, et al., "Comparative analysis between RiskA and Risk Spectrum in fault tree calculation," Chinese Journal of Nuclear Science and Engineering, **26(4)**, 358-362 (2006).

[15] Y. C. Wu, Y. Z. Li, L. Q. Hu, et al., "Development of the Third Qinshan Nuclear Power Plant Risk Monitor," in Proceedings of International Conference on Nuclear Engineering, Xi'an, Shanxi, China, 697-701 (2010).

[16] F. WANG, Y. LI, J. Q. WANG, et al., "ARCHITECTURE AND DESIGN OF THIRD QINSHAN NUCLEAR POWER PLANT RISK MONITOR," in International Conference on the Physics of Reactors 2012, PHYSOR 2012: Advances in Reactor Physics, Knoxville, Tennessee, USA, 4591-4598(2012).

[17] J. Q. WANG, Y. Z. LI, F. WANG, et al., "DEVELOPMENT AND VALIDATION OF INSTANTANEOUS RISK MODEL IN NUCLEAR POWER PLANT'S RISK MONITOR," in International Conference on the Physics of Reactors 2012, PHYSOR 2012: Advances in Reactor Physics, Knoxville, Tennessee, USA, 4599-4606(2012).

[18] J. W. Xu, J. Wang, S. Q. Chen, et al., "Web-based Fault Tree Collaborative Modeling in RiskA," in ANS PSA 2013 International Topical Meeting on Probabilistic Safety Assessment and Analysis, Columbia, SC(2013).

[19] F. Wang, L. Q. Hu, J. Wang, et al., "A Nuclear Power Plant Risk Monitor Based On Cloud Computing," in 2013 International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering, Emeishan, Sichuan, China, 194-197(2013).

[20] J. Wang, L. Q. Hu, S. Q. Chen, et al., "Verification Of RiskA Calculation Engine Based On Open-PSA Platform," in 2013 International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering, Emeishan, Sichuan, China, 32-35(2013).

[21] J. Q. Wang, F. Wang, J. Wang, et al., "Application of calculation engine of RiskA to nuclear power plant's probabilistic safety assessment," Chinese Journal of Nuclear Science and Engineering, **31(1)**, 75-79 (2011).