Computing and Informatics, Vol. 31, 2012, 1001–1016, V 2012-Dec-3

DEPTH-FIRST EVENT ORDERING IN BDD-BASED FAULT TREE ANALYSIS

Yuchang Mo, Farong Zhong, Huawen Liu

Department of Computer Science
Zhejiang Normal University
Jinhua 321004, China
e-mail: myc@zjnu.cn

Quansheng Yang

School of Computer Science and Engineering
Southeast University
Nanjing 210089, China
e-mail: yagnqs@seu.edu.cn

Gang Cui

School of Computer Science and Technology
Harbin institute of technology
Harbin 150001, China
e-mail: cg@hit.edu.cn

Communicated by Vladimír Kvasnička

Abstract. In BDD-based fault tree analysis, the size of BDD encoding fault trees heavily depends on the chosen ordering. From a theoretical point of view, finding the best ordering is an intractable task. So, heuristics are used to get good orderings. The most simple, and often one of the best heuristics is depth first left most (DFLM) heuristic. Although having been used widely, the performance of DFLM heuristic is still only vaguely understood, and not much formal work has been done. This paper starts from two different research objects: fault tree without repeated events (NRFT) and fault tree with repeated events (RFT). For NRFT, the BDD

generated according to DFLM ordering is proved to be the smallest BDD with the size equal to the total number of events. For RFT, a randomized algorithm is firstly proposed to create reliable benchmarks including large number of random fault trees with different specificities. Then, these benchmarks are used to perform two types of experiments to study the performance of DFLM heuristic. For RFT with small number of repeated events, it is found that the sizes of the BDD built over DFLM orderings are only slightly larger than the sizes of the RFT with different specificities. However, with the increase of the number of repeated events, we encounter the size explosion problem, and the change of repeated event distribution patterns will have a significant impact on the sizes of the BDD built over DFLM orderings. We also find that the number of repeated events is the more important measure than some other specificities (shape, logical type of top gate and OR/AND gate distribution) to estimate the level of the difficulty in BDD-based fault tree analysis.

Keywords: Fault tree analysis; benchmark; event ordering; binary decision diagram.

Mathematics Subject Classification 2010: 68M15

1 INTRODUCTION

Fault tree analysis (FTA) is an important technique for reliability and safety analysis. Bell telephone laboratories developed the concept in 1962 for the U.S. Air Force. It was later adopted and extensively applied by the Boeing Company. FTA is now widely used in the electronics, nuclear and aerospace industries.

Binary decision diagrams (BDD) are the state-of-the-art data structure to handle Boolean functions. Since their introduction in the reliability field they have proved to be in many cases a very powerful tool [1]. They made possible the assessment of complex fault-trees both qualitatively (computation of minimal cutsets) and quantitatively (exact calculation of the top event). Tools such as Aralia can in many cases give more accurate results than conventional tools, while running 1 000 times faster [2].

In BDD-based fault tree analysis, a fault tree can be encoded by a BDD. This encoding requires to select a total order over the events. For a given ordering, the representation is unique, up to an isomorphism. The smaller the BDD, the more efficient subsequent operations using it (probability assessment, computation of minimal cutsets) are likely to be. So it is of a great interest to get BDD as small as possible.

The size of BDD encoding fault trees heavily depends on the chosen ordering. Actually, there is often a variation of several orders of magnitude between the sizes of two BDD built over different orderings. From a theoretical point of view, finding the best ordering is an intractable task [3]. So, heuristics are used to get good orderings [4, 5].

Many ordering heuristics have been proposed in the literature. The most simple, and often one of the best heuristics, consist in numbering basic events by means of a depth first left most (DFLM for short) traversal of the tree [6]. DFLM heuristic is "good" because it preserves, at least to a certain extent, the locality of the events. For instance, it numbers basic of events of modules consecutively. Based on DFLM heuristic, different variants have been proposed, such as those consisting in a rearrangement of fanions of gates according to the values of their weights [4], or those consisting in a rearrangement of fanions of events according to the numbers of repetitions [5].

Although having been used widely, the performance of DFLM heuristic is still only vaguely understood, and not much formal work has been done. The research question in this paper is: what's the relation between DFLM's performance and different specificities in the trees (such as the number of repeated events and their distribution patterns, tree shape and size, logical type of top gate and OR/AND gate distribution. ...)?

We start from two different research objects: fault tree without repeated events (NRFT) and fault tree with repeated events (RFT). For NRFT, the BDD generated according to DFLM ordering is proved to be the smallest BDD with the size equal to the total number of events. For RFT, a randomized algorithm is firstly proposed to create reliable benchmarks including large number of random fault trees with different specificities. Then, these benchmarks are used to perform two types of experiments to study the performance of DFLM heuristic. We provide experimental results we got, and give some interesting findings for our research questions.

2 SMALLEST BDD GENERATION FOR NRFT

For NRFT, the BDD generated according to DFLM ordering is proved to be the smallest BDD.

Theorem 1. For NRFT, DFLM ordering is optimal and the time complexity of the smallest BDD generation is $O(n)$.

Proof. The following notations are used:

- X the set including all events contained by a tree
- n the number of events included by a tree, i.e., $n = |X|$
- f the Boolean function equivalent to a tree.

Firstly, we try to prove that the size of the smallest BDD encoding a NRFT is not less than n.

For the sake of contradiction, suppose that the smallest BDD encoding a NRFT is less than n, and event $x_i \in X$ does not appear in this BDD. Thus, for any assignment σ of $X - \{x_i\}$, i.e., a mapping from $X - \{x_i\}$ into $\{0, 1\}$, the following formula (1) holds:

$$f_{x_i=1}(\sigma) = f_{x_i=0}(\sigma) = 1 \vee f_{x_i=1}(\sigma) = f_{x_i=0}(\sigma) = 0. \quad (1)$$

However, a special assignment σ^* can be established as follows:

1. If x_i is connected to an OR gate, for any sibling subtree (connected to the same gate with x_i) t_i and any event x_k included by t_i, $\sigma^*(x_k) = 0$;
2. If x_i is connected to an AND gate, for any sibling subtree (connected to the same gate with x_i) t_i and any event x_k included by t_i, $\sigma^*(x_k) = 1$;
3. Simplify the tree using following rules (2), and goto 1. until there is only one event x_i left.

$$f \vee 1 = 1, f \vee 0 = f, f \wedge 1 = f, f \wedge 0 = 0 \tag{2}$$

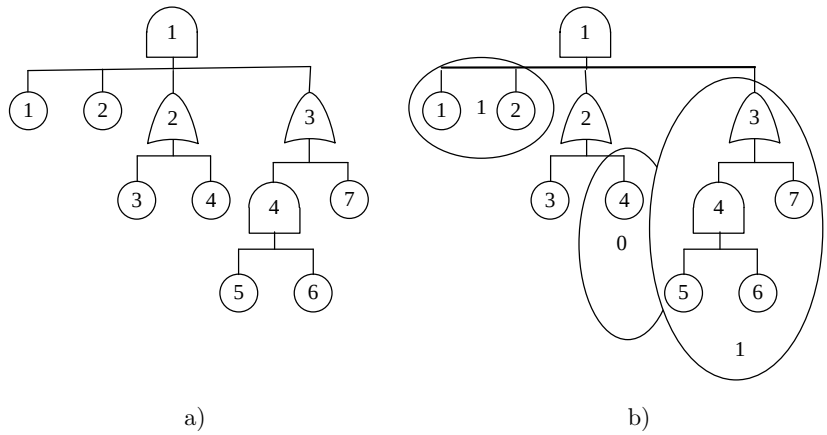


Fig. 1. An illustration of the special assignment σ^*

As an illustration, for the NRFT shown in Figure 1 a), suppose that event x_3 is not included by the smallest BDD. The derivation of the corresponding special assignment σ^* of the events $\{x_1, x_2, x_4, x_5, x_6, x_7\}$ is shown in Figure 1 b) and Table 1.

x	x_1	x_2	x_4	x_5	x_6	x_7
$\sigma^*(x)$	1	1	0	1	1	1

Table 1. An illustration of the special assignment σ^*

It is obvious that for this assignment σ^* the following formula holds:

$$f_{x_i=1}(\sigma^*) = 1 \wedge f_{x_i=0}(\sigma^*) = 0 \tag{3}$$

Therefore, there will not be an equivalent BDD with a size less than n.

Secondly, we try to prove that the size of the equivalent BDD generated using DFLM ordering is equal to n.

According to DFLM ordering, the BDD can be generated in a bottom-up manner. During the generation process, we can find that:

- For each bottom gate (all its fanions are events), the size of the BDD encoding this gate is the number of the fanins (events).
- For each intermediate gate (at least one of its fanions is gate), the size of the BDD encoding this gate is the sum of the sizes of all the BDD encoding its sub-trees.

As an illustration, for the NRFT shown in Figure 1 a), Table 2 gives the results in the bottom-up generation process.

Gate	DFLM ordering	BDD size
G4	$x_5 < x_6$	2
G3	$x_5 < x_6 < x_7$	2+1=3
G2	$x_3 < x_4$	2
G1	$x_1 < x_2 < x_3 < x_4 < x_5 < x_6 < x_7$	$2 + 3 + 2 = 7$

Table 2. BDD generation in a bottom-up manner for NRFT in Figure 1 a)

Thus, the size of the equivalent BDD according to DFLM ordering is n, and the BDD generation takes linear time.

Finally, as a synthesis of the above results, Theorem 1 can be derived. □

3 RFT BENCHMARK

3.1 Requirements

A benchmark is a set of fault trees used to collect experimental data during the performance evaluation process. For example:

1. A benchmark including 13 real life coherent fault trees is used to compare 6 interesting heuristics using Aralia [4].
2. A benchmark including 255 fault trees is used to develop a NN (Neural networks)-based heuristic selection method for the fault tree to BDD conversion process [5].
3. A benchmark including 44 fault trees is used to develop a new ordering methodology which seeks to select events during the conversion process from a fault tree, allowing different potential ordering permutations on each path of the diagram [7].
4. A self-developed benchmark is used to show that the proposed BDD-based algorithm is more efficient than traditional SDP (sum of disjoint products)-based algorithm for reliability analysis of phased-mission systems [8].

However, in the real sense very few benchmarks proposed in the literature are reliable ones due to the reasons outlined as follows.

Requirement 1. For a reliable benchmark, it should include a large number of trees with different structural characteristics, such as tree shape and size, logical type of top gate and OR/AND gate distribution.

The benchmark used in [4] is made of fault trees from different origins; but there is not much description of their structural characteristics and the total number of selected fault trees is only 13.

The benchmark used in [8] has:

1. 7 groups of components;
2. each group has n components;
3. 5 system configurations;
4. 4 mission configurations.

For this benchmark the number of component groups is unchangeable, the number of phases or phase configuration or mission configuration has a limited degree of freedom, and only the component number included by a group can be selected randomly. Due to the small sample size problem, the conclusions or results might become ungeneralizable to fault trees with different characteristics, as shown in [9].

Requirement 2. In a reliable benchmark, the selected fault trees should have different repeated events characteristics, such as the number of repeated events and their distribution patterns.

Given the fact that the repeated events have a great impact on the performance of different FTA techniques, the design and application of a benchmark should pay more attention to repeated events. The benchmarks in the literature have very little description of the diversity of their repeated event distribution patterns [4, 5, 6, 7, 8], or have only small number of repeated events [5, 7] (not larger than 20 repeated events). Thus, the analysts and researchers might feel confused when they try to apply the conclusions in real-world practices or do further research to improve the results [10, 11].

In this paper, we prepare to use a randomized algorithm to create a benchmark consisting of large number of random fault trees with different specificities, and what's more important this kind of diversity is controllable by several identified parameters to help us take on fine-grained performance analysis.

3.2 Design

A randomized FTG algorithm is described here to create a reliable benchmark. The main points are to randomly generate fault tree skeletons based on the given parameters, and then add repeated events to the skeletons randomly.

The FTG algorithm as shown in Table 3 generates a fault tree in a top-down manner under the control of five parameters: pg_i, a_i, b_i, *toplogic*, and *pre*.

The key points of FTG are discussed as follows:

Key point #1: Determine a gate's logic

A simplified fault tree has an alternating sequence of AND and OR gates. Thus, a gate's logic property is closely related to the top gate's logic (denoted with logic parameter toplogic) and its layer's index.

If toplogic is OR, gates in "odd" layers are all OR gates, and gates in "even" layers are all AND gates. If toplogic is AND, gates in "odd" layers are all AND gates, and gates in "even" layers are all OR gates.

Key point #2: Determine a gate's fanions number

The fanions numbers of the gates staying in $\#i$ layer follow a uniform distribution on the interval $[a_i, b_i]$, $1 \leq i <$ the total number of layers.

Key point #3: Determine a node type

Denote the percentage of gates in the set of all nodes in $\#i$ layer with pg_i $(0 \leq pg_i \leq 1)$. A node type can be decided as follows:

1. Generate a random number x from the uniform distribution on the interval $[0, 1]$;
2. If $x < pg_i$, then the node type is gate; else the node type is event.

Denote the number of all nodes in $\#i$ layer with n_i. We can derive the following equation easily:

$$n_{i+1} = n_i \times pg_i \times (a_i + b_i)/2. \tag{4}$$

Thus, to make all layers to have similar sizes,

$$pg_i \times (a_i + b_i)/2 \sim 1. \tag{5}$$

If $pg_i \times (a_i + b_i)/2 > 1$ holds, n_i increases as i increases; and if $pg_i \times (a_i + b_i)/2 < 1$ holds, n_i decreases as i increases.

Normally, a fault tree has much more gates than events in its top part in order to include enough events, i.e., $1 \geq pg_i \gg 2/(a_i + b_i)$ holds in the top part. Whereas it has much more events than gates in the bottom part in order to have a wind-up process, i.e., $2/(a_i + b_i) \gg pg_i \geq 0$ holds in the bottom part.

Key point #4: Add repeated events to the fault tree skeleton

Different events have different Id values. Denote the percentage of repeated events in the set of all events with *pre*. An event type, repeated or not, can be decided as follows:

1. Generate a random number x from the uniform distribution on the interval $[0, 1]$;
2. If $x < pre$ then the event is repeated one; else it is not a repeated event.

Figure 2 is a fault tree example generated by the FTG algorithm. Note that repeated events are labeled with $*$.

0	FTG($toplogic, pg_i, a_i, b_i, pre$) =
1	Generate root gate root, determine its fanins number and logic,
2	Push ($root$),
3	If stack is empty; Return root.
4	Pop (g),
5	Determine g's fanins number and logic,
6	For each g's fanin, do
7	Generate node x, determine its type (gate or event),
8	If x is a gate; Push(x);
9	Goto 3
10	Id = 1;
11	Calculate the total number of events TN
12	For each event, do
13	Determine its type,
14	If it is a repeated event; $event.Id \leq Random(1, TN * pre)$
15	If it is not a repeated event; $event.Id \leq Id{+}{+}$;
16	Return root;

Table 3. FTG algorithm

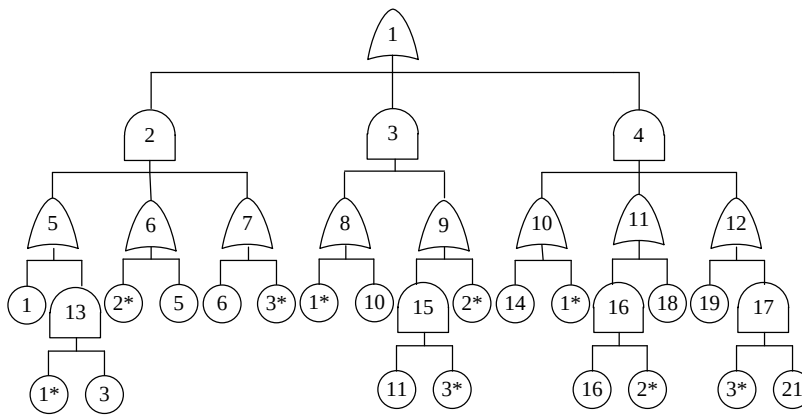


Fig. 2. A fault tree example generated by the FTG algorithm

4 PERFORMANCE ANALYSIS WITH RFT

4.1 Type 1 Experiment (RFT with Small Number of Repeated Events)

To find the relation between DFLM's performance and different specificities in the trees (such as the number of repeated events and their distribution pattern, tree shape and size, logical type of top gate and OR/AND gate distribution. ...), the experiment protocol is described as follows:

1. Initialize the parameters;
2. Run FTG 30 times to create a benchmark including 30 random fault trees;
3. Generate an equivalent BDD using DFLM ordering for each tree in the benchmark, and collect the BDD size information.
4. Analyze the experimental data to get maximum, minimal, mean, deviation, and coefficient of variation (CV) of the obtained 30 different BDD size data using the following Equation (6).

$$\begin{aligned} Mean &= \left(\sum_{i=1}^{30} bddsize_i\right)/30 \\ Dev &= \left[\sum_{i=1}^{30}(bddsize_i - Mean)^2\right]/30 \\ CV &= \sqrt{Dev}/Mean \end{aligned}$$

Table 4 is obtained by performing the above experiment 5 times with the parameters in (7).

$$toplogic = AND\ pg_i = \begin{cases} 1 & i \le 2 \\ 0.3 & 3 \le i \le 7 \\ 0 & 8 \le i \end{cases} \quad a_i/b_i = \begin{cases} 5 & i = 2 \\ 3/5 & 3 \le i \end{cases} \quad pre = 0.05 \qquad (6)$$

Table 5 is obtained by performing the above experiment 5 times with the parameters in (8).

$$toplogic = OR\ pg_i = \begin{cases} 1 & i \le 2 \\ 0.3 & 3 \le i \le 7 \\ 0 & 8 \le i \end{cases} \quad a_i/b_i = \begin{cases} 5 & i = 2 \\ 2/6 & 3 \le i \end{cases} \quad pre = 0.05 \qquad (7)$$

Table 6 is obtained by performing the above experiment 5 times with the parameters in (9).

$$toplogic = OR\ pg_i = \begin{cases} 1 & i \le 2 \\ 0.5 & 3 \le i \le 9 \\ 0 & 10 \le i \end{cases} \quad a_i/b_i = \begin{cases} 5 & i = 2 \\ 2/6 & 3 \le i \end{cases} \quad pre = 0.05 \qquad (8)$$

Check the values of CV of BDD sizes in the above tables. It can be concluded that:

Conclusion 1. For the RFT with small number of repeated events (not more than 5 %), the change of their distribution patterns only causes a slight variation of the sizes of the BDD built over DFLM orderings, i.e., no CV is larger than 0.1.

Check the values of Mean of BDD sizes in the above tables, and compare them with the values of the number of events. It can be concluded that:

Conclusion 2. For the RFT with small number of repeated events (not more than 5 %), there is no size explosion problem, and the sizes of the BDD built over DFLM orderings are never two times larger than the total number of events.

The above two conclusions hold for each experiment on the RFT with different specificities, such as shape, size, logical type of top gate and OR/AND gate distribution.

Thus, for the BDD-based analysis of RFT with small number of repeated events, the required time and memory can be evaluated accurately, and DFLM heuristic is enough to process very large trees with the available computational resources of standard personal computers.

No.	Size of tree	No. of events	No. of gates	No. repeated events	Max of BDD sizes	Min of BDD sizes	Mean of BDD sizes	CV of BDD sizes
#1	99	67	32	5	115	91	102	0.0594219
#2	156	105	51	5	188	158	171	0.0545697
#3	180	121	59	5	210	150	178	0.0868246
#4	201	135	66	5	245	181	215	0.0880845
#5	216	145	71	5	269	222	247	0.0541576

Table 4. Performance results of DFLM heuristic with parameters in (7)

No.	Size of tree	No. of events	No. of gates	No. repeated events	Max of BDD sizes	Min of BDD sizes	Mean of BDD sizes	CV of BDD sizes
#1	144	97	47	5	169	143	154	0.040857
#2	189	127	62	5	233	174	213	0.0605949
#3	228	153	75	5	282	247	263	0.0329972
#4	240	161	79	5	296	249	269	0.0455843
#5	279	187	92	5	350	276	311	0.0592997

Table 5. Performance results of DFLM heuristic with parameters in (8)

4.2 Type 2 Experiment (RFT with Large Number of Repeated Events)

During the above experiments, we find that, for RFT with small number of repeated events, the sizes of the BDDs built over DFLM orderings are never two times larger than the total number of events, and the impact of changing the characteristics (such as repeated event distribution pattern, tree shape and size, logical type of top gate and OR/AND gate distribution) on performance is not significant.

No.	Size of tree	No. of events	No. of gates	No. repeated events	Max of BDD sizes	Min of BDD sizes	Mean of BDD sizes	CV of BDD sizes
#1	1 023	683	340	5	1 307	920	1 127	0.0809006
#2	1 029	687	342	5	1 332	1 161	1 251	0.0360974
#3	1 050	701	349	5	1 340	1 126	1 236	0.0480758
#4	1 089	727	362	5	1 420	1 196	1 285	0.0492721
#5	1 077	719	358	5	1 342	1 121	1 242	0.0460433

Table 6. Performance results of DFLM heuristic with parameters in (9)

Now, let us turn the research object to RFT with large number of repeated events.

Table 7 is obtained by performing the experiment 5 times with the parameters in (10).

$$toplogic = AND\ pg_i = \begin{cases} 1 & i \le 2 \\ 0.3 & 3 \le i \le 5 \\ 0 & 6 \le i \end{cases} \quad a_i/b_i = \begin{cases} 5 & i = 2 \\ 3/5 & 3 \le i \end{cases} \quad pre = 0.2 \tag{9}$$

Table 8 is obtained by performing the experiment 5 times with the parameters in (11).

$$toplogic = OR\ pg_i = \begin{cases} 1 & i \le 2 \\ 0.3 & 3 \le i \le 5 \\ 0 & 6 \le i \end{cases} \quad a_i/b_i = \begin{cases} 6 & i = 2 \\ 2/4 & 3 \le i \end{cases} \quad pre = 0.2 \tag{10}$$

Table 9 is obtained by performing the experiment 5 times with the parameters in (12).

$$toplogic = AND\ pg_i = \begin{cases} 1 & i \le 2 \\ 0.3 & 3 \le i \le 7 \\ 0 & 8 \le i \end{cases} \quad a_i/b_i = \begin{cases} 5 & i = 2 \\ 3/5 & 3 \le i \end{cases} \quad pre = 0.2 \tag{11}$$

Check the values of mean of BDD sizes in the above tables. As we expect, it is found that:

Conclusion 3. Although for the RFT with small number of repeated events there is only a slight increase in BDD size with the increase of tree size, we encounter the size explosion problem when both the tree size and the number of repeated events increase.

Note that, with the increase of the total number of events from 100 to 220 and the number of repeated events from 20 to 50, the mean of the sizes of the BDD built over DFLM orderings dramatically increase from 1 000 to 70 000. If the tree size and the number of repeated events continue to increase, the available computational

resources of standard personal computers will be exhausted quickly. What's more important is that, in order to prove that the new proposal (such as new heuristics, algorithms or tools) is really better than the available counterparties to cope with the size explosion problem, fault trees included by the used benchmark should have more events than 200 and more repeated events than 50. This requirement is often overlooked by the related research work [4, 5, 6, 7].

No.	Size of tree	No. of events	No. of gates	Interval of No. of repeated events	Max of BDD sizes	Min of BDD sizes	Mean of BDD sizes	CV of BDD sizes
#1	141	95	46	[19, 28]	1 606	588	842	0.270096
#2	165	111	54	[25, 33]	4 270	1 349	2 378	0.314447
#3	201	135	66	[27, 37]	1 3517	2 669	5 134	0.44241
#4	207	139	68	[26, 36]	8 606	1 656	4 940	0.324718
#5	231	155	76	[26, 35]	10 196	3 198	6 356	0.284917

Table 7. Performance results of DFLM heuristic with parameters in (10)

No.	Size of tree	No. of events	No. of gates	Interval of No. of repeated events	Max of BDD sizes	Min of BDD sizes	Mean of BDD sizes	CV of BDD sizes
#1	157	105	52	[23, 28]	2 972	883	1 790	0.285413
#2	166	112	54	[24, 32]	4 746	1 609	3 260	0.266595
#3	217	149	68	[22, 38]	17 315	2 077	7 239	0.570824
#4	242	156	75	[23, 39]	27 441	2 164	8 854	0.594620
#5	260	173	87	[24, 35]	32 219	1 961	9 441	0.787493

Table 8. Performance results of DFLM heuristic with parameters in (11)

No.	Size of tree	No. of events	No. of gates	Interval of No. of repeated events	Max of BDD sizes	Min of BDD sizes	Mean of BDD sizes	CV of BDD sizes
#1	228	153	75	[30, 40]	38 322	8 693	22 361	0.385287
#2	249	167	82	[38, 44]	37 254	12 216	2 2884	0.272282
#3	282	189	93	[36, 49]	160 388	29 355	77 639	0.440476
#4	300	201	99	[33, 44]	159 696	10 816	51 316	0.598973
#5	330	221	109	[37, 49]	254 860	16 151	72 923	0.680787

Table 9. Performance results of DFLM heuristic with parameters in (12)

Check the values of CV of BDD sizes in the above tables. It can be concluded that:

Conclusion 4. For the RFT with many repeated events (about 20 percent of the total number of events), the change of their distribution patterns will have a significant impact on the sizes of the BDD built over DFLM orderings.

Note that the CVs in this type of experiments are 10 times larger than those in Tables 4–6, and, for the 30 different BDD size data, the maximum can frequently be 10 times larger than the minimum.

Compare the data in Table 7 and those in Table 8, it can be concluded that:

Conclusion 5. For the RFT with many repeated events (about 20 percent of the total number of events), the values of CVs or BDD sizes are very close for the RFT having similar tree sizes and numbers of repeated events but different shape, logical type of top gate and OR/AND gate distribution. That is to say, the change of those latter tree specificities will not have a significant impact on the sizes of the BDD built over DFLM orderings, and the number of repeated events is the more important measure to estimate the level of the difficulty in BDD-based fault tree analysis.

5 CONCLUSION AND FURTHER RESEARCH

In BDD-based fault tree analysis, the size of BDD encoding fault trees heavily depends on the chosen ordering. The smaller the BDD, the more efficient subsequent operations using it (probability assessment, computation of minimal cutsets) are likely to be; so it is of a great interest to get BDD as small as possible. From a theoretical point of view, finding the best ordering is an intractable task. So, heuristics are used to get good orderings.

Many ordering heuristics have been proposed in the literature. The most simple, and often used one of the best heuristics is DFLM heuristic. The research question in this paper is: what is the relation between DFLM's performance and different specificities in the trees (such as the number of repeated events and their distribution patterns, tree shape and size, logical type of top gate and OR/AND gate distribution. . . .)?

The contributions of this paper are listed as follows:

1. For NRFT, the BDD generated according to DFLM ordering is proved to be the smallest BDD with the size equal to the total number of events.
2. For RFT with small number of repeated events, there is no size explosion problem. The sizes of the BDD built over DFLM orderings are only slightly larger than the total number of events. The available computational resources of standard personal computers is enough to process these RFT with different specificities, such as repeated event distribution pattern, tree size and shape, logical type of top gate and OR/AND gate distribution.
3. With the increase of the number of repeated events, we encounter the size explosion problem, and the change of repeated event distribution patterns will have a significant impact on the sizes of the BDD built over DFLM orderings. We also find that the number of repeated events is the more important measure than some other specificities (shape, logical type of top gate and OR/AND

gate distribution) to estimate the level of the difficulty in BDD-based fault tree analysis.

The most interesting further work, in our minds, is as follows:

1. To compare the performance of different variants proposed based on DFLM heuristic, such as those consisting in a rearrangement of fanins of gates according to the values of their weights, or those consisting in a rearrangement of fanions of events according to the values of their repetitions.
2. To extend the performance analysis practice to multistate systems and multiphase systems, where fault trees have dependent events and mutually exclusive events in addition to repeated events.

Acknowledgement

The research work of this paper was funded by the National Natural Science Foundation of China (No. 60903011, 61272130, 61100119); Natural Science Foundation of Zhejiang Province (No. Y1100689); Natural Science Foundation of Jiangsu Province (No. BK2009267); Project of Science and Technology Department of Zhejiang Province (No. 2010C31122); National Research Foundation for the Doctoral Program of Higher Education of China (No. 20090092120030); the China Scholarship Council under Grant No. 2011833135.

REFERENCES

[1] Rauzy, A.: New Algorithms for Fault Tree Analysis. Reliability Engineering and System Safety, Vol. 40, 1993, pp. 203–211.

[2] ARALIA Group: Computation of Prime Implicants of a Fault Tree within ARALIA. In: Proc. Of the ESREL '95 Conf., Bournemouth, UK.

[3] Bollig, B.—Wegener, I.: Improving the Variable Ordering of OBDDs is NP-Complete. IEEE Transactions on Computers, Vol. 45, 1996, No. 9, pp. 993–1002.

[4] Bouissou, M.—Bruyere, F.—Rauzy, A.: BDD-Based Fault Tree Processing: A Comparison of Variable Ordering Heuristics. In: Proceedings of ESREL 97, 1997, pp. 2045–2052.

[5] Bartlett, L. M.—Andrews, J. D.: Selecting an Ordering Heuristic for the Fault Tree to Binary Decision Diagram Conversion Process Using Neural Networks. IEEE Transactions on Reliability, Vol. 51, 2002, No. 3, pp. 344–349.

[6] Gauthier, J.—Leduc, X.—Rauzy, A.: Assessment of Large Automatically Generated Fault Trees by Means of Binary Decision Diagrams. Journal of Risk and Reliability, Vol. 221, 2007, No. 2, pp. 95–105.

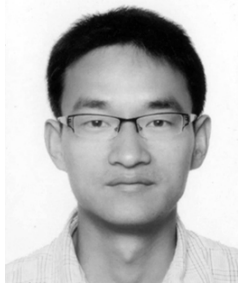
[7] Bartlett, L. M.—Du, S.: New Progressive Variable Ordering for Binary Decision Diagram Analysis of Fault Trees. Quality and Reliability Engineering International, Vol. 21, 2005, No. 4, pp. 413–425.

[8] Zang, X.—Sun, H.—Trivedi, K. S.: A BDD-Based Algorithm for Reliability Evaluation of Phased Mission Systems. IEEE Transactions on Reliability, Vol. 48, 1999, No. 1, pp. 50–60.

[9] Mo, Y. C.: New Insights into the BDD-Based Reliability Analysis of Phased-Mission Systems. IEEE Transactions on Reliability, Vol. 58, 2009, No. 4, pp. 667–678.

[10] Dutuit, Y.—Rauzy, A.: A Linear Time Algorithm to Find Modules of Fault Trees. IEEE Transactions on Reliability, Vol. 45, 1996, No. 3, pp. 422–425.

[11] Reay, K. A.—Andrews, J. D.: A Fault Tree Analysis Strategy Using Binary Decision Diagrams. Reliability Engineering and System Safety, Vol. 78, 2002, No. 1, pp. 45–56.

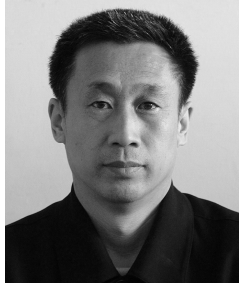


Yuchang Mo is an Associate Professor in the College of Mathematics, Physics and Information Engineering Director of the Dependable Computing Lab and at the Zhejiang Normal University. He received his B. Sc., M. Sc. and Ph. D. degrees in computer science from Harbin Institute of Technology, Harbin, China, 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.

Farong Zhong is currently a Professor at the Department of Computer Science in Zhejiang Normal University. He received his B. Sc. degree in computer science from Shandong University in 1986, M. Sc. and Ph. D. degrees in computer science from Shanghai Jiaotong University in 1994 and 2005 respectively. His research interests focus on process calculus and web services.

Huawen Liu works in Department of Computer Science at Zhejiang Normal University, P.R. China, as a Lecturer. He received his B. Sc. degree in computer science from Jiangxi Normal University in 1999, and M. Sc. and Ph. D. in computer science from Jilin University, P.R. China, in 2007 and 2009, respectively.

Quansheng Yang is currently an Associate Professor with the Department of Computer Engineering, Southeast University (SEU), Nanjing, China and the Director of the Computer Architecture Lab in the School of Computer Science and Engineering at SEU. He received his B. E. (1991) degree in computer science from Huazhong Normal University, Wu Han, China and M. Sc. (1994) degrees in computer science from Southeast University, Nanjing, China. His research interests include computer architecture, advanced CPU architecture, embedded system and reconfigurable computing.



Gang Cui is currently a Professor at the Department of Computer Science in Harbin Institute of Technology, Harbin, China. He serves as a senier member and a committee member of Fault-Tolerance Computing Committee under China Computer Federation (CCF). His research interests involve computer architecture, space computing, high-dependability computing, fault-tolerance computing, etc. He had presided over and taken part in many significant projects assigned by the "863", central ministries, or local government. Till now, he had acquired one first-rate award, two second-rate awards, and three third-rate awards from the relevant central ministries. He also wrote more than 200 papers compiled a teaching book and obtained 60 patents for inventions.