An Ordering Heuristic To Develop The Binary Decision Diagram Based on Structural Importance

L.M.Bartlett and J.D.Andrews
Loughborough University
Loughborough
Leicestershire
LE11 3TU

Abstract

Fault tree analysis is often used to assess risks within industrial systems. The technique is commonly used although there are associated limitations in terms of accuracy and efficiency when dealing with large fault tree structures. The most recent approach to aid the analysis of the fault tree diagram is the Binary Decision Diagram (BDD) methodology. To utilise the technique the fault tree structure needs to be converted into the BDD format. Converting the fault tree requires the basic events of the tree to be placed in an ordering. The ordering of the basic events is critical to the resulting size of the BDD, and ultimately affects the performance and benefits of this technique. A number of heuristic approaches have been developed to produce an optimal ordering permutation for a specific tree. These heuristic approaches do not always yield a minimal BDD structure for all trees. This paper looks at a heuristic that is based on the structural importance measure of each basic event. Comparing the resulting size of the BDD with the smallest generated from a set of six alternative ordering heuristics, this new structural heuristic produced a BDD of smaller or equal dimension on 77 percent of trials.

Keywords: Fault Tree Analysis; Binary Decision Diagrams; Ordering Heuristics; Structural Importance.

1. Introduction

There are a number of assessment methods that can be adopted to qualify and quantify the risk of hazardous incidents within an industrial system. The most common approach is that of fault tree analysis, which is based on Kinetic Tree Theory[1]. Despite its widespread use the fault tree methodology has limitations, especially when analysing large tree structures. To overcome these limitations, over the last five years a completely new approach has been generated. This new approach is called the Binary Decision Diagram (BDD) methodology[2-7]. Utilising this new technique involves converting the fault tree into an alternative representation. If this transformation is possible then the analysis procedure is qualitatively more efficient and quantitatively more accurate. However, these benefits can only be used if the BDD can be generated, and one which is minimal in size. To make the conversion the basic events of the fault tree need to be taken in a specified order. This ordering is crucial to the size of the resulting BDD, where a good ordering can result in a very efficient analysis and a poor ordering can lead to problems.

Within the literature there are a number of possible ordering heuristics to convert the fault tree structure[8-12]. There are two types of ordering heuristic reported namely neighbourhood and weighting methods. Neighbourhood methods produce an ordering by performing a systematic traversal of the tree adding the basic events to the ordered list as they are encountered. Weighting methods usually make two passes of the tree. On the first pass a weight is given to each basic event and gate within the tree, and the second pass then orders the basic events depending on the weights allocated in the first pass. Using both these types of heuristic the resulting size of the BDD is variable, where sometimes a minimal BDD will be produced for a tree and the same heuristic applied to a different tree will result in a BDD of incredible size. On reviewing the heuristics currently in the literature certain problem areas were evident. To try and overcome these limitations the structural importance measure was investigated.

2. Binary Decision Diagrams

A BDD is a directed acyclic graph, as shown in figure 1. All paths through the BDD start at the root vertex and terminate in one of two states, either a 1 state which corresponds to a system failure, or a 0 state which corresponds to a system success. A BDD is composed of terminal and non-terminal vertices, which are connected by branches. Non-terminal vertices correspond to the basic events of the fault tree.

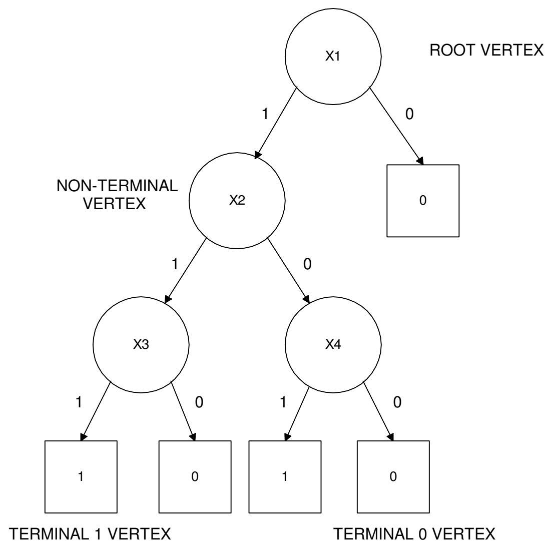


Figure 1: A Binary Decision Diagram

All the left branches leaving a vertex are the 1 branches (component failure occurs) and all the right branches the 0 branches (component functional). Every path starts from the root vertex, and proceeds down through the diagram to the terminal vertices. Only the vertices that lie on a 1 branch on the way to a terminal 1 vertex are included in the path. All the paths terminating in a 1 state give the cut sets of the fault tree. For example, the cut sets of figure 1 are:

1) *X1X2X3* 2) *X1X4*.

3. Variable Ordering Problem

In constructing the BDD, the ordering of the basic events is crucial to the size of the resulting diagram. Using an inefficient ordering scheme will produce a non-minimal BDD structure. Alternative ordering schemes will produce BDD's of different sizes, the smaller the BDD the more optimal the diagram. To illustrate this fact, consider the simple fault tree shown in figure 2. The tree has four basic events, where *X2* is repeated.

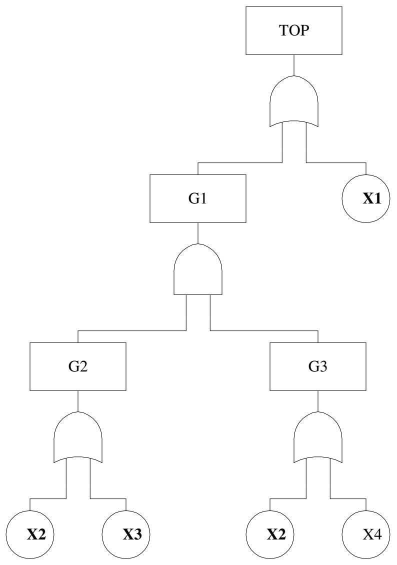
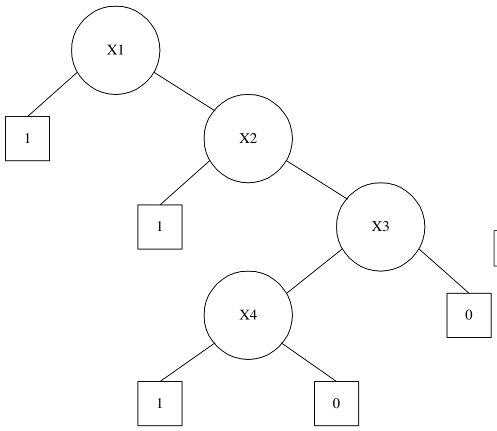


Figure 2: A Simple Fault Tree

If the basic event ordering permutation of *X1<X2<X3<X4* is taken, the resulting BDD is shown in figure 3. This structure consists of only four nodes, it is a minimal structure and hence produces only minimal cut sets.



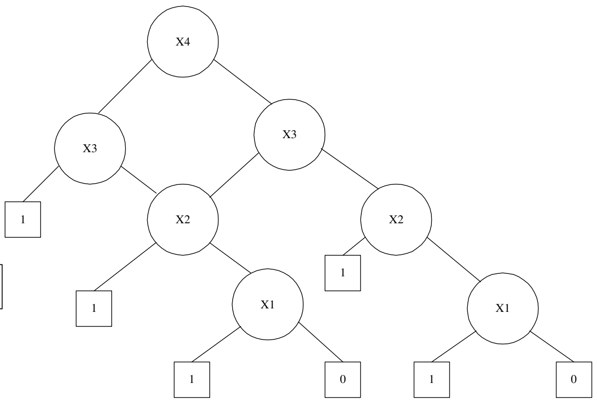


Figure 3: Ordering *X1<X2<X3<X4*

Figure 4: Ordering *X4<X3<X2<X1*

However, if the alternative ordering permutation of *X4<X3<X2<X1* is taken the resulting BDD (shown in figure 4) consists of seven nodes, it is non-minimal and yields non-minimal cut sets. To analyse the second BDD would involve a minimisation procedure to find the minimal cut sets and the quantification would be less efficient. For larger fault tree structures the efficiency of the resulting BDD is more critical, and in the worst case of using a poor ordering permutation, the diagram may be unsolvable.

4. Drawbacks of Current Heuristics

On reviewing the heuristics currently in the literature certain problem areas were evident. One was that many of the heuristics were affected by how the fault tree was drawn, therefore for the same logic expression a number of different BDDs could result depending on how the fault tree was represented. Also many of the heuristics have a structured pattern. That is, the ordering permutation is generated by going from the top of the tree to the bottom, and it does not allow for components to be selected from different branches of the tree and lie next to each other in the ordering list. Another problem is how to deal with events that occur more than once. For heuristics that incorporate weights, different sized BDDs may result when the order in which components with the same weighting are separated. For example, if a top-down approach of ordering is applied to the tree and the equally weighted component highest up the tree is taken first this may result in a different ordering than when considering ordering the weighted component that occurs most often first.

From this the properties required in a good ordering heuristic seem to be:

- The contribution of an event to the system failure mode must be reflected in the ordering produced.
- The ordering must be robust i.e. the ordering must be dependent upon the logic function represented by the fault tree and not influenced by the way the fault tree has been drawn.
- To uniquely map the fault tree onto a single event ordering.

Considering these points the structural importance measure was investigated. This heuristic satisfies two out of the three points above. It does represent the contribution each component makes to the occurrence of the top event, and it is also unaffected by the way the tree is written or drawn. However, the ordering produced is not unique because some components will have the same contribution and the means of breaking these ties will affect the ordering.

5. Structural Importance Measures

5.1. Definition of Importance Measures

A very useful piece of information, which can be derived from a system reliability assessment, is the *importance measure* for each component or minimal cut set. For each component it's importance signifies the role that it plays in either causing or contributing to the occurrence of the top event. This role is given a rank in terms of a numerical value.

Importance measures can be categorised in two ways: (1) deterministic; and (2) probabilistic. As the ordering issue does not depend on the probabilistic failure characteristics of the components just it's position in the tree, the deterministic structural importance measure is analysed and discussed as a potential ordering mechanism.

5.2. Deterministic Measures of Importance

Deterministic measures assess the importance of a component to the system operation without considering the component's probability of failure. One such measure is the *structural measure of importance, SMI*, which is defined for a component, *i*, as

$$SMI_i = \frac{number\ of\ critical\ system\ states\ for\ component\ i}{total\ number\ of\ states\ for\ the\ (n-1)\ remaining\ components}$$

A critical state for component *i* is a state for the remaining (*n* -1) components such that a failure of component *i* causes the system to go from a working state to a failed state.

5.3. Calculation Method For Deterministic Structural Importance Measure

To illustrate this structural importance measure, consider the fault tree drawn in figure 5. The logic expression for the top event is: $TOP = A + B.D + B.C$

Therefore, the top event will occur (or top event failure) if *A* occurs, or both *B* and *C* occur, or *B* and *D* both occur. To generate a variable ordering the structural importance measures of each component need to be calculated. The procedure to carry out this process can be simplified as follows:

For each component:

1. Find the possible states for the remaining components.
2. Test whether each of the remaining states are critical for the chosen component.

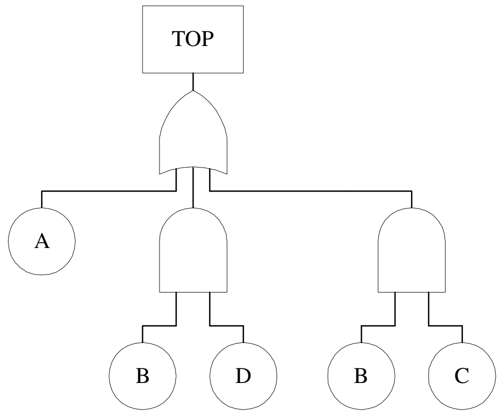


Figure 5: Simple Fault Tree Structure

Taking component *A* from figure 5 as an example, there are eight states for the remaining components these are:

1. *B* working, *C* working and *D* working $(\overline{B}\overline{C}\overline{D})$
2. *B* failed, *C* working and *D* working $(B\overline{C}\overline{D})$
3. *B* working, *C* failed and *D* working $(\overline{B}C\overline{D})$
4. *B* and *C* failed, and *D* working $(BC\overline{D})$
5. *B* working, *C* working and *D* failed $(\overline{B}\overline{C}D)$
6. *B* and *D* failed, and *C* working $(B\overline{C}D)$
7. *B* working, and *C* and *D* failed $(\overline{B}CD)$
8. *B*, *C* and *D* failed (BCD)

To explain the theory behind a critical state, each of the eight states for the remaining components needs to be examined. If component *A* is working and given the states of the remaining components the system works (top event non-occurrence) then this reflects the possibility of a critical state. The determining factor is whether failure of component *A* causes the system to fail. If it does then this is referred to as a critical state for component *A*.

Given one state for the remaining components, namely *B*, *C* and *D* all working $(\overline{B}\overline{C}\overline{D})$, with component *A* working $(\overline{A})$ the system would be working. If component *A* then failed the system would fail, and this can be defined as a critical state. Therefore, the structural importance measure for component *A* (SMI_A) is calculated as shown in table 1.

States for other components	Critical State for *A*
$\overline{B}\overline{C}\overline{D}$	Yes
$B\overline{C}\overline{D}$	Yes
$\overline{B}C\overline{D}$	Yes
$BC\overline{D}$	No
$\overline{B}\overline{C}D$	Yes
$B\overline{C}D$	No
$\overline{B}CD$	Yes
BCD	No

(NB. the $\overline{}$ means component works)

Table 1: Calculation of The Structural Importance Measure For Component *A*

There are eight states for the remaining components, of which five of these are critical for component *A*, hence $SMI_A = 5/8$.

The same process is repeated for all of the other components, in this example variable *B*, *C* and *D*. Hence, the structural importance measure of component *B* is 3/8, and the structural importance of both component *C* and *D* is 1/8.

On gaining each of the importance measures, the remaining factor is to order the components in descending order depending on the values calculated.

5.4. Problems Implementing Methodology

To calculate the structural importance measure for all the components within a fault tree as illustrated above even for small trees is relatively time consuming. This is ever more prevalent with large fault tree structures where the number of possible combinations of the working and failed states of the components is exponentially increasing with respect to the number of components.

Programming the method utilised in section 5.3 would not be very efficient, as the process would require the following:

1. The logic expression for top event.
2. For each component, the program would need to repeatedly substitute in values for the remaining components in the system, for every state combination.

Whilst simple in concept it is computationally time consuming to perform the procedure for each variable.

5.5. Using Birnbaums's Structural Importance Measure

An alternative method to calculate the structural importance measure involves the probabilistic importance measure of Birnbaum, namely *Birnbaums Measure of Criticality.* Lambert[13] stated that this probabilistic measure could be used to evaluate the structural importance measure. However, this still requires the system probability function $Q(\underline{q})$ or an approximation of it.

Birnbaums measure of criticality ($G_i(\underline{q})$) is defined as:

$$G_i(\underline{q}) = Q(1_i, \underline{q}) - Q(0_i, \underline{q})$$

where $Q(\underline{q})$ is the probability that the system fails, and

$$Q(1_i, \underline{q}) = (q_1, q_2, \ldots\ldots, q_{i-1}, 1, q_{i+1}, \ldots\ldots\ldots q_n)$$

and $$Q(0_i, \underline{q}) = (q_1, q_2, \ldots\ldots, q_{i-1}, 0, q_{i+1}, \ldots\ldots\ldots q_n).$$

From Lamberts[13] paper it states that if the probability of failure of component i, $q_i(t)$, is set equal to ½ for all $i \neq j$, then the number of states in which component i is critical, denoted by B_i, is defined as:

$$B_i = \{\ Q(1_i, \underline{1/2}) - Q(0_i, \underline{1/2})\ \}$$

Implementing this numerically for the tree in figure 5, the top event probability expression is given as:

$$Q(\underline{q}) = q_A + q_B q_C + q_B q_D - q_A q_B q_C - q_A q_B q_D - q_B q_C q_D + q_A q_B q_C q_D$$

Calculating the structural importance measure for A;

$$Q(1_A, \underline{q}) = 1$$
$$Q(0_A, \underline{q}) = q_B q_C + q_B q_D - q_B q_C q_D$$

Therefore, $B_A(\underline{q}) = 1 - (q_B q_C + q_B q_D - q_B q_C q_D)$, and $B_A(\underline{1/2}) = 1 - (3/8) = 5/8.$

The same principle is applied to B, C and D, resulting in $B_B(\underline{1/2}) = 3/8,\ B_C(\underline{1/2}) = B_D(\underline{1/2})$ $1/8$.

6. Results Using BDD For Calculation Procedure

To test the potential of the component structural importance method to the problem of generating a variable ordering heuristic to yield an optimal BDD the Birnbaum structural importance measure was used. Software was available to produce the Birnbaum Measure of Criticality from the BDD. As initially it is the validity of the measure that is being established and not the efficiency of the technique then a BDD, which has been constructed using a different variable ordering, has been used to gain the importance measures.

To establish the influence of the new ordering permutation a comparison was made with the best of the six previously identified alternative heuristics[14]. It is the number of nodes of the BDD structure that are used in the comparison process. The reason for this was that the smaller the initial BDD, from which the quantification process can be carried out, the more efficient the quantification process, and also to determine the minimal cut sets less work is involved in minimisation.

In table 2 three groupings relating to the number of nodes in the BDD have been identified for the comparison. These are less than the previous best, equal to the previous best, and greater than the previous best. For this method to be successful then the majority or all of the trees when converted using this new ordering need to result in a BDD of the same or smaller dimension than the previous best of the set of six alternatives. Two hundred and twenty five fault trees were compared and the results are shown in table 2.

Nodes in comparison (to best)	No. of trees	% of trees	Total =/<
=	77	34.2	
<	96	42.7	76.9 %
>	52	23.1	

Table 2: Results of Comparison of Structural Importance Ordering and Previous Best Ordering on BDD Size

From these results, it is concluded that approximately 77 % of all the trees tested within the data set, using the structural importance ordering yields a BDD of equal or smaller dimension than the previous best scheme ordering. Of the remaining 23 percent of trees, which had BDDs of greater size than the previous best, the BDDs were larger by varying degrees of magnitude. Some BDDS were only a couple of nodes larger whereas others were a couple hundred nodes larger.

From the set of six orderings the distribution of 'best heuristics' is illustrated in table 3. The total number of trees that each scheme predicts the best result for is greater than 225 in total as some heuristics produce a BDD of equal size to another. The best heuristic out of the six for overall performance is the modified top-down, left-right approach, although the winning margin is very small. This new structural importance ordering heuristic clearly outperforms all of these six heuristics individually and more research is needed to unravel it's full potential and try to establish an efficient method of calculation.

Ordering Heuristic	Number of Instances Minimal BDD Produced	Percentage of Times Best BDD Produced
Top-down, left-right	87	15.4
Modified Top-down, left-right	169	29.8
Depth-First	120	21.2
Modified Depth-First	117	20.6
Priority Depth-First	36	6.3
Modified Priority Depth-First	38	6.7

Table 3: Performance of Six Different Ordering Heuristics in Producing BDDs

8. Approximated Structural Importance Method

Having established the value of the structural importance approach to variable ordering an efficient means of calculating this ranking is required. To find an alternative method of calculating the mathematical structural importance measure without generating the BDD first an approximation technique was used. The principle of the Birnbaum structural importance measure was applied directly to the tree. Using this technique, the selected component assumes the failure probability of 1 and 0 on two consecutive runs, the rest of the components are given failure probabilities equal to ½ and the probability of occurrence of the top event is evaluated by working up through the tree structure. The BDD variable ordering is generated depending on the basic event that generates the largest probability value contribution for the top event. The difference with this and the exact version of Birnbaum's structural measure is the terms in the unavailability expression of the top event. When this approximation method is applied to a tree the logical redundancies have not been reduced by Boolean algebra and so the cut sets may not be minimal. For this reason the structural importance values and resulting weights are not exactly the same but it may still offer a relatively good ordering heuristic.

To illustrate the application of this proposed Birnbaum method, consider the tree whose data file representing the fault tree structure would be written in the following format:

```
TOP      OR    2   1   Gate1   Gate2   A
Gate1    AND   0   2   B       C
Gate2    AND   0   2   B       D
```

The code written to establish the structural importance value for each component within the data file follows these steps:

1. Make a list of all the components within the tree structure, easiest method using a top-down, left-right approach.
2. Repeat step 2 twice, first setting the selected component failure probability to 1 and the second time with the selected component failure probability set to 0.
    a. Start at the top of the data file.
    b. Repeat the following steps:
        - Work through the data file and find gates with only basic events, substitute in value for selected component and ½ for the remaining component failure probabilities.

- Calculate intermediate values of importance – if the gate is an AND gate multiply values of inputs, if an OR gate then use $1-\prod_{i=1}^{n}(1-q_i)$.
- Substitute in data file value for gate just calculated.
- Continue through the data file, if at bottom, start process again at top searching for gates whose inputs all have a calculated value.
- Calculate new intermediate results.
- Continue until the Top Event gate has been given a value.

c. Record value.

3. Subtract the value gained from the second run from the first. The result is the approximation for the structural importance value of the component.

Therefore, the structural measure for each component in the data file is:

- Measure for $A = 1 - 7/16 = 9/16$
- Measure for $B = 7/8 - 1/2 = 3/8$
- Measure for $C = 13/16 - 5/8 = 3/16$
- Measure for $D = 13/16 - 5/8 = 3/16$

From these values and using the top-down, left-right method of ordering for matched components, the ordering would be:

$$A < B < D < C$$

This is the same as that generated with the mathematical structural importance measure.

The approach was tested on a number of trees with non-repeated events, some produced the same order others did not. As no approach so far has given exact results as compared to the mathematical structural importance measure, this approximated measure was evaluated further.

Computer code was produced to generate the variable ordering list for two hundred and twenty five fault trees (used for other mathematical method) using the Birnbaum measure applied directly to the tree. The number of trees for which the BDD produced was equal to, greater than, or less than the size of the BDD derived from the best scheme of the six alternatives is given in table 4.

Outcome	Percentage
< 77	
= 74	67.1
> 74	32.9

Table 4: Results Using Approximation Method

From table 4 it can be seen that for 67.1 % of the test set of trees the BDD produced was of equal or smaller dimension than the BDD produced using the best scheme option from a set of six. This result is the highest percentage of equal or smaller BDDs of the three approximation methods tested. In comparison to the mathematical structural importance measure, whose percentage for this category of BDDs was 77.3%, this approximated figure is nearing the same accuracy.

9. Summary of Structural Measure Findings

- Static variable orderings can be created in two ways:
  - A structured traversal of the tree (preserving neighbourhoods)
  - A method allocating weights to events (not necessarily preserving neighbourhoods)
- The result of both ordering methods are dependent on the way the tree has been drawn, not the logic function it represents.
- A deterministic importance measure has been applied to generate an ordering of the variables of a fault tree, the *structural measure*, which is dependent upon the logic function and not the drawing of the fault tree.
- To assess the effectiveness of the structural measure BDDs using this ordering have been compared to BDDs generated by using other variable ordering heuristics applying a structured traversal. The results proved to be consistently good.
- The structural importance approach has proven to produce a BDD of equal or smaller dimension than the previous best result from an ordering selected from six structured traversal alternatives on 77 % of occasions.
- Using the two pass approach, such that the structural importance is generated using the system probability function from the BDD generated using the best ordering over a selection of trees – modified top-down, left-right traversal. A second BDD is then produced using the importance ordering.
- To improve upon the efficiency of the two pass method, an approach to find an alternative structural importance measure was to use the Birnbaum measure applied directly to the fault tree. Results using this technique produced equal or lesser sized BDDs on 67 % of occasions.
- It is felt that this approximated structural importance ordering heuristic is the best method to date in trying to achieve a minimal or near minimal BDD structure for any given fault tree.
- Future work in this area needs to focus on improving the simplified versions of the structural importance measure and finding an alternative method that approaches the 77 percent performance of the mathematical measure.

References

[1] W. E. Vesely. A Time Dependent Methodology for Fault Tree Evaluation. Nuclear Design and Engineering, vol. 13, 1970, p337-360.

[2] S. B. Akers. Binary Decision Diagrams. IEEE Transactions on Computers, vol. C-27, 1978, p509-516.

[3] A. Rauzy. A Brief Introduction to Binary Decision Diagrams. European Journal of Automation, vol. 30, No. 8, 1996.

[4] A. Rauzy. New algorithms for fault tree analysis. Reliability Engineering and System Safety, vol. 40, 1993, p203-211.

[5] R. M. Sinnamon and J. D. Andrews. Quantitative Fault Tree Analysis Using Binary Decision Diagrams. European Journal of Automation, vol. 30, No. 8, 1996.

[6] R. E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. IEEE Transactions on Computers, vol. C-35, No. 8, 1986.

[7] R. M. Sinnamon and J. D. Andrews. Improved Efficiency in Qualitative Fault Tree Analysis. Advances in Reliability Technology Symposium, Manchester, 1996.

[8] R. M. Sinnamon and J. D. Andrews. Fault Tree Analysis and Binary Decision Diagrams. Proceedings of the Reliability and Maintainability Symposium, Las Vegas, January 1996.

[9] S. Minato, N. Ishiura and S. Yajima. On variable ordering of binary decision diagrams for the application of the multi-level logic synthesis. Proceedings of the European Conference on Design Automation, 1991, p50-54.

[10] M. Bouissou, F. Bruyere and A. Rauzy. BDD Based Fault-Tree Processing: A Comparison of Variable Ordering Heuristics. Proceedings of ESREL '97 Conference, August 1997.

[11] M. Bouissou. An Ordering Heuristic for Building Binary Decision Diagrams from Fault Trees. Proceedings of Reliability and Maintainability Symposium, Las Vegas, 1996, p208-214.

[12] M. Fujita, H. Fujisawa, N. Kawato. Evaluation and Improvements of Boolean Comparison Method Based on Binary Decision Diagrams. IEEE Transactions on Computer Aided Design (Conference), Nov 1988, p2-5.

[13] H. E. Lambert. Measures of Importance of Events and Cut Sets in Fault Trees. Reliability and Fault Tree Analysis: Theoretical and Applied Aspects of System Reliability and Safety Assessment, SIAM, Philadelphia, 1975, p77-100.

[14] J. D. Andrews and L. M. Bartlett. Efficient Basic Event Orderings For Binary Decision Diagrams. Proceedings of the Annual Reliability and Maintainability Symposium, 1998, p61-68.