Progression of the Binary Decision Diagram Conversion Methods

Lisa Bartlett, Ph.D.; Systems Engineering Department; Loughborough University; Loughborough; UK

Keywords: Fault Tree Analysis, Binary Decision Diagrams, Ordering Heuristics

Abstract

The Binary Decision Diagram method has been formulated over the last decade. It is the latest methodology developed to overcome the deficiencies of the common fault tree analysis technique. Its advantages include increased efficiency in determining the qualitative characteristics of a failure mode represented using a fault tree, and improved accuracy when calculating the corresponding quantitative performance measures. The disadvantage of the approach however, is that the conversion from the fault tree can not be guaranteed to be optimal, reducing the advantages of using the method. Until this is rectified the inclusion of this methodology within commercial packages is not possible.

The conversion process involves selecting the basic events from the fault tree and generating an ordering from which the Binary Decision Diagram is constructed. It is this ordering which is crucial. A number of conversion alternatives exist in the research literature, from heuristic approaches examining the positioning of the basic events in the fault tree, to the latest developments which look at selection mechanisms using pattern recognition approaches. This paper reviews the conversion approaches available, the advantages and disadvantages of each, and the latest methodologies being investigated.

Introduction

The Binary Decision Diagram (or BDD) is a means to analyse failure modes represented by a fault tree (refs. 1-5). The method permits exact quantification of system performance measures, as well as an efficient derivation of the minimal cut sets. This development has taken place over the last ten years. Although the mathematical foundations of the approach are well grounded, its lack of use in commercial packages is due to problems in converting the fault tree to the alternative BDD representation.

The conversion of the fault tree requires its basic events to be placed in an order. This ordering determines how the BDD is constructed. Different orderings for the same fault tree structure will produce BDD's of different sizes. Ideally a minimal BDD is required, one that is smallest in size, which will allow the minimal cut sets to be derived directly, without any minimization procedures. This structure will permit an efficient quantification process requiring fewer steps. Determining what this ordering should be has been the focus of much research.

Several research papers (refs. 6-11) have been published which investigate different ordering strategies. Initial orderings were proposed which traverse the tree in a structured method (refs. 6-10), hence basic events situated next to each other in the fault tree are likely to be ordered next to each other. These heuristics were found to be effective for specific fault tree structures, but a general heuristic that produces a minimal BDD for all fault trees has yet to be found. Lack of this efficient ordering for any tree structure has resulted in the inability to integrate the methodology successfully into a commercial fault tree package.

To overcome this conversion problem, advances have been made in three main directions. Firstly alternative heuristics have been proposed which concentrate on the more influential basic events, by ranking them in terms of their contribution to the occurrence of the failure mode (ref. 11). The second area of research focuses on selection mechanisms (refs. 12-14), which choose an ordering from a set of alternatives depending on the fault tree under analysis. These methods have shown potential to find a minimal BDD for all trees, although further refinement is required. The final avenue is to use a progressive ordering technique, where rather than having a set order before the conversion process, this order is changeable during the process. Each of these techniques are discussed within the paper and conclusions drawn concerning each.

Binary Decision Diagrams

BDD Representation: The BDD considers each basic event in turn, as specified by the ordering generated from the fault tree. Each basic event forms a vertex or node within the diagram (represented by a circle in figure 1). From each vertex two branches emanate, one relating to the functioning of the basic event (branch labelled 0) and one referring to the failure of the basic event (branch labelled 1). The diagram terminates in one of two states (or terminal vertices represented by the square boxes), where a terminal 1 vertex refers to system failure and a terminal 0 vertex system functioning.

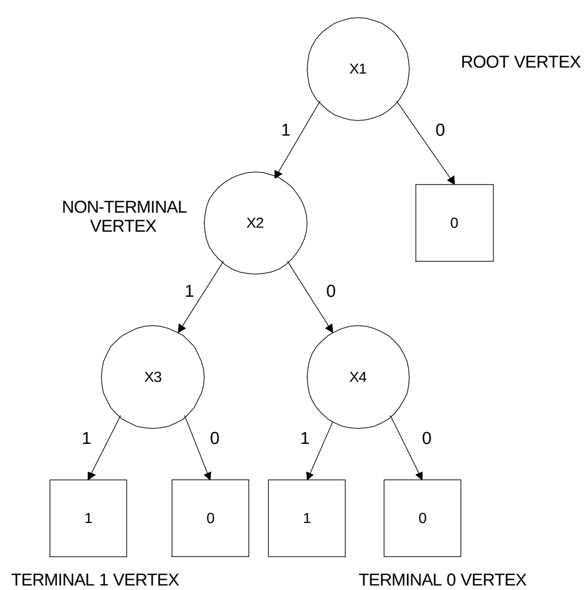


Figure 1 - A Binary Decision Diagram

To establish the cut sets of the original fault tree, the paths resulting in system failure are examined. All basic events that lie on a 1 branch on the way to a terminal 1 vertex are included in the cut set. Therefore, the cut sets from figure 1 are {X1,X2,X3} and {X1,X4}. The method to convert a fault tree to its equivalent BDD is described in many publications and the reader is referred to two for details (refs. 2,6).

The Conversion 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 for qualitative and quantitative analysis. To illustrate this fact, consider the simple fault tree shown in figure 2. The tree has four basic events, where X2 is repeated.

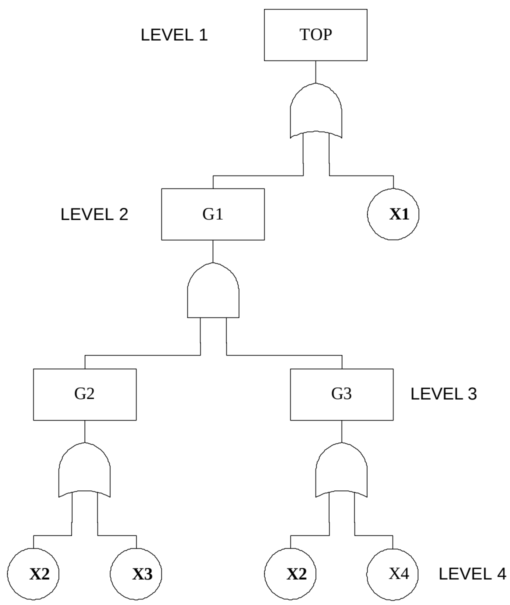
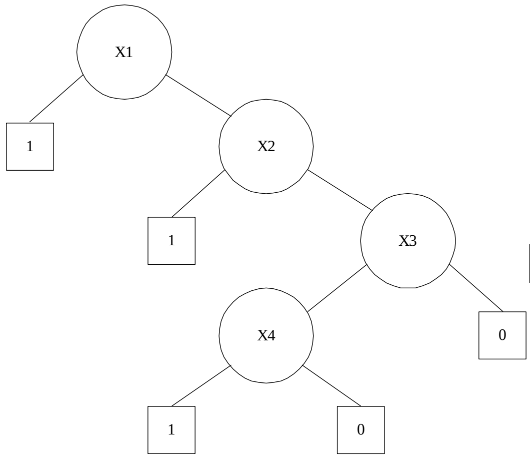


Figure 2 - A Simple Fault Tree

If the basic event ordering permutation of X1<X2<X3<X4 is taken, where X1 is considered first, then X2 and so on, the resulting BDD is shown in figure 3. All left branches represent basic event failure and right branches basic event functioning. This structure consists of only four nodes (or vertices), it is a minimal structure and hence produces only minimal cut sets, these being {X1}, {X2}, and {X3,X4}. 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. The cut sets are {X3,X4}, {X4,X2}, {X4,X1}, {X3,X2}, {X3, X1}, {X2}, and {X1}. A minimization procedure is required to derive the minimal cut sets for this BDD. 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 production of the diagram may not be possible within the computers memory capacity.



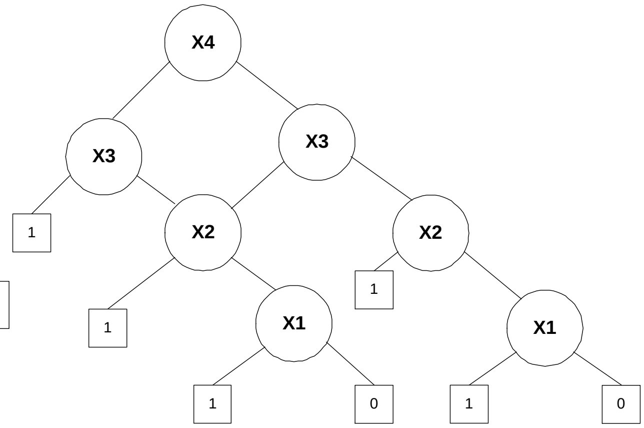


Figure 3 - Ordering X1<X2<X3<X4

Figure 4 - Ordering X4<X3<X2<X1

The objective is to produce an ordering scheme that achieves the ‘best’ BDD, where ‘best’ implies the smallest BDD, ideally a minimal structure. The remaining sections of this paper will discuss the commonly used ordering approaches and the latest developments.

Heuristic Approaches

The approaches that are grouped in this category all have a structured methodology, yet are based on no mathematical foundations. The most common heuristic for ordering is produced by listing the basic events in a top-down, left–right manner from the original fault tree. To illustrate consider the fault tree in figure 2. The fault tree structure has four distinct levels separated by gates. The process of ordering begins at the top most level, and continues downward until the last level is reached. At each level, the ordering commences in a left to right path and basic events that are encountered are added to the ordering list. If a basic event is encountered which is already positioned higher up the tree and has therefore already been incorporated in the list, then it is ignored. The ordering using this method for the fault tree is therefore X1<X2<X3<X4. The BDD in this case is minimal with four nodes and yields the three minimal cut sets directly. However, BDD's produced using this simple ordering of the variables are frequently inefficient since they produce a large number of non-minimal cut sets. The non-minimum BDD must undergo the minimising procedure to obtain the minimal cut sets, which can cause an undesirable increase in computer time.

An alternative approach is referred to as the Depth-first method, whereby the top-down principle is applied to subtrees of the whole tree separately. Each top event input forms a subtree. In figure 2 the subtrees are {Gate1} and {X1}. Using this heuristic for the tree represented in figure 2 would give the ordering X2<X3<X4<X1. The BDD using this ordering is non-minimal with five nodes, and producing four cut sets. It must be noted that if the fault tree shown in figure 2 was drawn differently this methodology, like the top-down, left-right approach, would result in a different ordering.

Research by Sinnamon and Andrews (ref. 6) investigated the effects that different ordering schemes produced on the resulting BDD. Six different heuristics were compared. The results showed that there were vast differences in the number of computations required to construct the BDD. Hence, great savings can be made in terms of computation time and memory requirements when an efficient ordering of the basic events can be established. The research highlighted that each tree has an individual variable ordering that will optimise its size and that a general ordering scheme that will be 'best' for all trees doesn't seem apparent. Research by Bouissou (ref. 8) showed similar findings, along with evidence that these heuristic methods are subject to a lack of robustness in terms of ordering differences due to re-drawing of the fault tree.

The advantage of all of these methods is that they are very simplistic and easy to implement. In considering the literature however certain problem areas are evident. One is that many of the heuristics are affected by how the fault tree is drawn, therefore for the same failure mode 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 basic events to be selected from different branches of the tree and lie next to each other in the ordering list.

Ranking Methods

Reasons for Implementation: Considering the inadequacies of the heuristic approaches, the properties required in a good ordering method 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.

Using these properties progression of the conversion methodology focused on ranking methods. The structural importance measure (ref. 11) was implemented which 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 basic events will have the same contribution and the means of breaking these ties will affect the ordering.

Definition of Structural Importance: 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 its 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.

A deterministic measure has been applied to the conversion problem. These 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.

Application to Conversion Problem: To calculate the number of critical states for every component is computationally very time consuming, therefore an alternative method to calculate the structural importance measure is used. This involves the probabilistic importance measure of Birnbaum, namely *Birnbaums Measure of Criticality.* Lambert (ref. 12) stated that this probabilistic measure could be used to evaluate the structural importance measure. Using Lamberts methodology it states that if the probability of failure of component *i*, $q_i(t)$, is set equal to ½ for all i ¹ j, then the number of states in which component *i* is critical, denoted by B_i, is given in equation 1.

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

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

$$Q(1_i, \underline{1/2}) = (q_1=1/2, q_2=1/2, \ldots\ldots, q_{i-}=1/2_1, 1, q_{i+1}=1/2, \ldots\ldots\ldots q_n=1/2)$$

and $$Q(0_i, \underline{1/2}) = (q_1=1/2, q_2=1/2, \ldots\ldots, q_{i-}=1/2_1, 0, q_{i+1}=1/2, \ldots\ldots\ldots q_n=1/2)$$

The numerical value representing the importance of each component is then used to rank each in an ordered list. The technique was tested on two hundred and twenty five fault trees. It was concluded that approximately 77 % of all the trees tested yielded a BDD of equal or smaller dimension than the previous best scheme ordering. This previous best scheme ordering was the smallest BDD generated using six alternative methods. Of the remaining 23 percent of trees, the BDDs were larger by varying degrees of magnitude.

The advantage of this ranking method is that it indicates that a small BDD is achievable for a large number of trees. However, the current method of generation requires the calculation of the probability expression for the failure mode, whose simplest derivation is by using the BDD. Therefore, this method currently requires two BDD’s, one for ranking and one for analysis. For this method to be used, not only does the performance still need to be improved but the method to obtain the ranking needs extensive consideration.

Selection Procedures

Overview: From reviewing the ordering approaches, it was clear, that although one specific method could not guarantee a minimal BDD for any one fault tree, there was the capability to produce a minimal BDD for most trees with the methods available, the difficulty is finding the correct one. The next area of research has been on trying to derive a selection procedure, whereby an ordering method could be chosen depending on the particular fault tree under analysis.

The basis of the selection mechanism is a rule based pattern recognition approach. There are several different types of pattern recognition approach, for example, classifier systems, neural networks, Bayesian methods and Fuzzy Logic. In the literature, the classifier system (ref. 13) has been used in conjunction with a genetic algorithm, and also two different forms of neural network (ref. 14) have been applied for this selection purpose. With the classifier system, the aim is to try and generate, initially randomly and then refine through training, a set of rules. Thus, when the classifier system is given a specific fault tree structure it can select, depending on the rules, the best of a set of alternative ordering schemes which will yield the most minimal BDD structure.

A neural network is a method of identifying patterns. It can be regarded as a particular choice for a set of functions that map a set of input variables to a set of output variables. Thus, this technique when applied to the variable ordering problem has inputs which represent the fault tree and outputs that represent the different ordering heuristics. There are a number of different types of neural network, namely single layer networks, multi-layer networks and radial basis functions. The latter two networks have been used in the ordering research (ref. 14).

The work done using classifiers has indicated that the method could be trained to predict the best of alternative ordering schemes to use on a fault tree structure to yield an efficient BDD representation. Although current limitations are due to the need for more example fault tree structures from which the rules can be learned and also the factors used for rule generation need enhancement to model the complexity of the problem. More promise has been shown with the neural network approaches as a selection mechanism. This is explained in a more detail in the following section.

Neural Networks: The multi-layer network or perceptron model has several distinct layers. The network has an input layer, an output layer and a number of hidden layers each with a specified number of nodes. The general architecture of the network is shown in figure 5. Each of the layers are connected by weights (W_{ij}), which determine the predictive ability of the network.

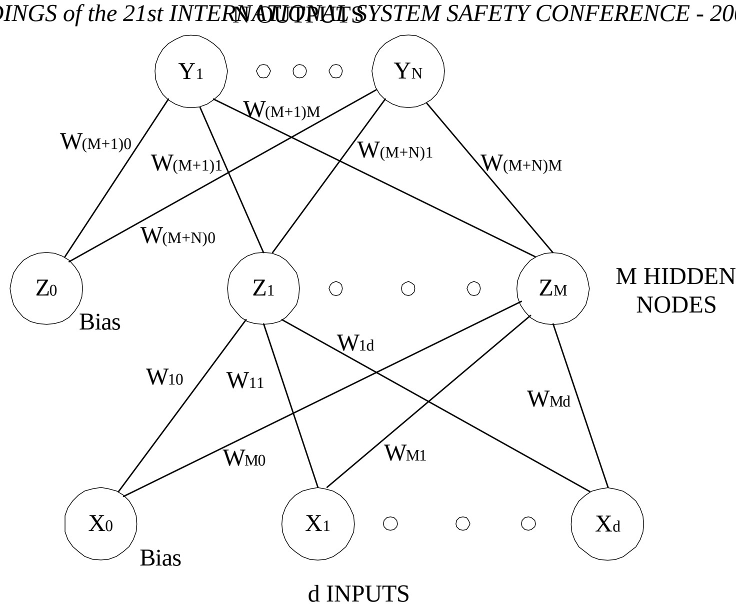


Figure 5: Diagram Representing a Multi-layer Perceptron

There are two modes of operation: a training phase to determine the optimum weights of the network and a predictive phase to generate the desired outputs for a previously unseen input. During the training phase an error back-propagation algorithm is used. The algorithmic process consists of two possible passes through the different layers of the network: a forward pass and a backward pass. The forward calculation pass uses fixed weights to derive the outputs for given inputs. The backward pass, on the other hand, allows the weights to be adjusted. This action permits the actual response of the network to move closer to the desired response. When the error has been reduced sufficiently it is these weights that are used as fixed values in the predictive phase. How well the network has been trained and models the problem will be reflected in the prediction of new input data. If the network has been trained well it will generalise well to new data and a correct response should be predicted.

Results using this research have shown potential for providing a means to obtain a minimal BDD with a 70% success rate, using a test set of twenty trees, with the remaining 30% producing a near minimal BDD. The same predictive potential has been found using the radial basis function neural network. Like multi-layer perceptrons the network has a number of input nodes, representing characteristics of the fault tree structure, and a number of output nodes, related to possible ordering heuristics. The functioning of the radial basic function network differs but the aim is the same. The same predictive capability was found in the research. The results also showed that for the 30% of incorrectly predicted trees, the answer produced was the second or third best alternative. Hence near minimal BDD structures were produced.

The advantages of these network approaches are that the benefits of a range of ordering methods can be used. This relieves the pressure for one heuristic alone to contain the capabilities to convert any tree. The disadvantage is that the network selection mechanism doesn't currently have the necessary predictive capability to be used in commercial packages. Methods to improve the selection mechanism have been addressed by using a Jacobian method to establish the sensitivity of the ordering heuristics to the fault tree structure (ref. 15). The aim is to alter the characteristics used for pattern recognition within the network to enhance the performance of the selection mechanism. This is an area of ongoing research.

Ordering at Each Node – Progressive Ordering

The latest progression of the research in this area has focussed on generating an ordering at each phase of conversion, i.e. after each node in the BDD, rather than one ordering for the whole process. Rauzy et al., (ref. 16) implemented this progressive approach where the aim is to reduce the size of each branch of the BDD. This is achieved by taking the basic events which form the smallest combination with other basic events in the logic expression. Repeated events are also given priority. For example, if the top event failure mode (TOP) is given by the expression TOP = ab + bcd + dei + acei, the first node in the BDD would be 'b', as basic event 'b' is contained in the smallest product combination and also is repeated within other products. The next basic events to be used are determined by considering the resulting logic expressions along the 1 and 0 branches of node 'b'. Conclusions made were that the method was easy to apply and gave good results. Whether the BDD was minimal was not clear, however BDDs were produced for some trees where alternative methods failed. In addition, an advantage this method has is that different orderings can be used along different branches.

The disadvantage of this approach is that it works from a logic expression incorporating the minimal cut sets of the fault tree, which is one of the desired end products of the BDD analysis. Therefore the methodology of allowing different orderings along different branches of the BDD shows potential, yet its application needs improvement. The fundamentals of the approach combined with an acceptable means of application are an area of research currently under investigation.

Conclusions

The development of the BDD conversion problem has progressed in three main areas, single heuristics, selection procedures and progressive ordering. There are a large number of single heuristics, with only a few mentioned in this paper. Most are easy to use with simple rules for forming the ordered list of basic events from the fault tree. Despite this derivation simplicity a minimal BDD for all trees is not guaranteed. More promise has been shown with the structural importance ranking method, yet difficulties lie in finding a simple method to obtain the numerical rankings from which the list is formulated.

The avenues which show the most potential to enable the use of this technique in commercial packages are the selection mechanism and progressive ordering methods. Ongoing research into the inputs of the selection mechanism seeks to enhance the current predictive potential. The latest innovative idea of progressive ordering is in its infancy and further improvements are expected. The research is heading in directions where the evolution of a possible solution is emerging and will allow this new methodology to become part of everyday commercial packages. This will inevitably enable the analysis of any fault tree to benefit from its advantages.

References

1. S.B.Akers, "Binary Decision Diagrams". IEEE Transactions on Computers, vol. C-27, 1978, p509-516.
2. A.Rauzy, "New Algorithms for Fault Tree Analysis". Reliability Engineering and System Safety, vol. 40, 1993, p203-211.
3. R.M.Sinnamon and J.D.Andrews, "Quantitative Fault Tree Analysis Using Binary Decision Diagrams". European Journal of Automation, vol. 30, No. 8, 1996.
4. A.Rauzy, "A Brief Introduction to Binary Decision Diagrams". European Journal of Automation, vol. 30, No. 8, 1996.
5. 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.
6. R.M.Sinnamon and J.D.Andrews, "New approaches to Evaluating Fault Trees". Proceedings of Esrel'95 Conference, June, 1995, p241-254.
7. 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.
8. 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.
9. M. Bouissou. An Ordering Heuristic for Building Binary Decision Diagrams from Fault Trees. Proceedings of Reliability and Maintainability Symposium, Las Vegas, 1996, p208-214.
10. 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.
11. L.M.Bartlett and J.D.Andrews. An Ordering Heuristic to Develop the Binary Decision Diagram Based on Structural Importance. Reliability Engineering and System Safety, 2001, vol. 72, p31 – 38.
12. 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.
13. 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.
14. L.M.Bartlett and J.D.Andrews. Choosing a Heuristic for the Fault Tree to Binary Decision Diagram Conversion using Neural Networks. IEEE Transactions on Reliability, 2002, vol. 51 (3), p344-349.
15. L.M.Bartlett. Important Fault Tree Characteristics for Efficient BDD Construction. Proceedings of the 15th Advances in Reliability Techniques Symposium, April 2003, p275-290.
16. A.Rauzy. A new methodology to handle Boolean models with loops. IEEE Transactions on Reliability, 2003 (to appear).

Biography

Lisa M Bartlett; Department of Systems Engineering; Loughborough University; Loughborough; LE11 3TU; U.K. E-mail: L.M.Bartlett@lboro.ac.uk.

Dr. Lisa Bartlett is a lecturer in the Department of Systems Engineering at Loughborough University. She is one of four academic members of staff within the Reliability Research Group of the department. She gained her PhD in Fault Tree Analysis methods in 2000 from Loughborough University. Her PhD research focused on the Binary Decision Diagram approach, and her current research interests continue in this area.