Hybrid approach for the assessment of PSA models by means of Binary Decision Diagrams

Cristina Ibáñez-Llano*
Instituto de Investigación Tecnológica (IIT). Escuela Técnica Superior de Ingeniería ICAI. Universidad Pontificia Comillas. C/Santa Cruz de Marcenado 26, 28015 Madrid, SPAIN
Tel: +34 91 542-2800, Fax: +34 91 542-3176
cristina.ibanez@iit.upcomillas.es

Antoine Rauzy
Dassault Systèmes. 10 rue Marcel Dassault CS 40501, 78946 VELIZY VILLACOUBLAY CEDEX, FRANCE
Antoine.RAUZY@3ds.com

Enrique Meléndez
Consejo de Seguridad Nuclear (CSN). C/ Justo *Dorado 11. 28040 Madrid, SPAIN*
ema@csn.es

Francisco Nieto
Instituto de Investigación Tecnológica (IIT). Escuela Técnica Superior de Ingeniería ICAI. Universidad Pontificia Comillas. C/Santa Cruz de Marcenado 26, 28015 Madrid, SPAIN
nieto@iit.upcomillas.es

(*) Corresponding author

Abstract: Binary Decision Diagrams are a well-known alternative to the minimal cutsets approach to assess Reliability Boolean models. They have been applied successfully to improve Fault Trees models assessment. However its application to solve large models, and in particular the Event Trees coming from the PSA studies of the nuclear industry remains to the date out of reach of an exact evaluation. For many real PSA models it may be not possible to compute the BDD within reasonable amount of time and memory without considering truncation or simplification of the model.

This paper presents a new approach to estimate the exact probabilistic quantification results (probability/frequency) based on combining the calculation of the MCS and the truncation limits, with the BDD approach, in order to have a better control on the reduction of the model and to properly account for the success branches. The added value of this methodology is that it is possible to ensure a real confidence interval of the exact value and therefore an explicit

knowledge of the error bound. Moreover, it can be used to measure the acceptability of the results obtained with the traditional techniques. The new method was applied to a real life PSA study and results obtained confirm the applicability of the methodology and open a new viewpoint for further developments.

Keywords: Probabilistic Safety Assessment, Binary Decision Diagrams, Event Trees, hybrid approach

1. Introduction

Probabilistic Safety Assessment is a well-established technique for integrating various Reliability models and to numerically quantify to the frequency of damage in nuclear facilities. Its use is now widespread in nuclear regulation as it complements traditional deterministic analysis, providing a comprehensive and structured approach to identifying undesired accident scenarios, computing its likelihood in terms of occurrence frequency, and assessing the consequences and mitigation strategies. In terms of the mathematical tools used, PSA studies rely on the Fault Tree/Event Tree (FT/ET) methodology to obtain the response model of the facility.

The majority of the computational tools used to assess the FT/ET models have implemented what is called the “classical” approach, namely the kinetic tree theory [1]. This approach, broadly used and accepted, is based on the computation of the minimal cutsets (MCS for short) by means of Boolean reduction and of the use of probabilistic (frequency) cutoffs, owing to the complexity of the models. Truncation cutoffs on probability (or frequency) and also on the order of the MCS have to be applied to avoid MCS explosion. To avoid computational complexity, success (i.e. negated) logic is avoided in FT/ET evaluation.

Bryant’s Binary Decision Diagrams (BDD) [2, 3] are a well-known alternative to the minimal cutsets approach to assess Boolean models. BDDs have the remarkable property of having complexity that is not related to the number of cutsets of the encoded Boolean formula. Conversely to the classical methodology, the BDD approach involves no approximation in the quantification of the model and is able to handle correctly negative logic (success branches) at low additional complexity cost. However BDDs are also subject to combinatorial explosion as the final size of the BDD is very sensitive to the variable ordering needed to convert the model into it.

After more than two decades of application, the BDD methodology has been applied successfully to improve fault tree assessment and its introduction in the field has permitted renewing its algorithmic framework. In the last years several works have as well undertaken its application to Event Tree Assessment [4-6]. However attempts to apply it to very large models, such as the ones coming from the PSA studies of the nuclear industry, which includes several thousand of basic events and logic gates, remains to the date out of reach of a full automatic treatment. Although some attempts have been successful [4] , for such large models it might be not possible to compute the BDD within reasonable amount of time and computer memory without considering truncation or simplification of the model. Consequently it is necessary to explore new approaches to the problem. A potential solution is to develop a hybrid approach that combines the calculation of the MCS with the BDD approach, which allows obtaining a better and more controllable bound approximation of the model. The motivation and the basis of this new approach is the principal contribution of the work presented in this paper.

The remainder of this paper is organized as follows. Section 2 is devoted to introduce some basic terminology, to describe the particularities of the PSA models and to introduce the case study. Section 3 reviews the existing approaches for the FT/ET assessment, namely the classical approach and the BDD approach. Section 4 is specifically focused in the problem of model simplification that is performed with the classical approach. Section 5 presents the mathematical foundation of the hybrid approach. Finally, the experimental results and the conclusions are provided in sections 6 and 7 respectively.

2. Description of PSA models

This section is devoted to introduce the basic terminology and concepts needed to describe the Boolean models used in PSA studies and to present the case study.

2.1. Terminology

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of Boolean variables. We will briefly review some basic definitions concerning Boolean algebra.

A Boolean formula, *F,* denoted here by upper case letters, is a term inductively constructed over the two Boolean constants, 0 and 1, a denumerable set of variables *X*, and the usual logic connectives: the disjunction, equivalent to the OR operator and denoted by + or $\vee$, the conjunction equivalent to the AND operator and denoted by $\cdot$ or $\wedge$ and the negation or NOT

operator, represented by the arithmetic symbol – or ¬. A literal is either a variable *v* or its negation $\overline{v}$. A product π is a set of literals that does not contain both a literal and its opposite. Typically, it is assimilated with the conjunction of its elements.

A miniterm on a set of variables *X* is a product that contains a literal built over each variable of *X*. For *n* variables, there exist 2^n miniterms that can be constructed on *X*.

An assignment σ of a set of variables *X* is a mapping from *X* to $\{0,1\}$ that assign a value to each variable of *X* (true=1/false=0). There is a one to one correspondence between miniterms over a finite set of variables *X* and assignments. An assignment (equivalently a miniterm) satisfies a formula *F* if $\sigma(F)=1$. In that case we say that σ belongs to *F*, i.e, $\sigma \in F$, and that σ is a solution of *F*.

There exist a natural order over literals: $\overline{v} < v$. This order can be extender to miniterms: $\pi \leq \rho$ iff for each variable of *X*, $\pi(v) \leq \rho(v)$. A formula *F* is monotone if for any pair of miniterms σ, ρ that satisfy *F* such that $\sigma \leq \rho$ then $\rho \in F$ implies that $\pi \in F$. The monotonic Boolean functions are precisely those which can be defined only with AND, OR and K/N operators and do not contain negations.

A product σ that satisfies a function *F* is also called an implicant of *F*. An implicant of *F* is prime if no proper subset of it is an implicant of *F*. In the general case, if the function in not monotone, prime implicants may contain negated variables. Any formula is equivalent to the disjunction of its prime implicants, or equivalently to a set of miniterms that satisfy it, leading to a representation in terms of a disjunction of conjunctions also called sum of products.

For any two formulae *F* and *G*, we say that *F* implies *G* if for any assignment satisfying *F* $\sigma \in F$, then it satisfies *G* as well. This is denoted by $F \models G$.

We denote by $F_{v \leftarrow e}$ the function in which the value of *v* is substituted by the constant $e \in \{0,1\}$. $F_{v \leftarrow 1}$ and $F_{v \leftarrow 0}$ are the positive and negative cofactors of *F* w.r.t. the variable *v*.

2.2. Boolean models

Boolean models are commonly used in risk analysis of industrial facilities to develop a representation of the overall system in terms of logic diagrams. In the case of PSA studies, the technique used for the schematic representation of the facility are the combination of Fault Trees and Event Trees.

Fault Trees are deductive models used to identify the causes of failures of a system in terms of its subsystems and basic component failures. The basic events represent component failures and unavailabilities or human errors, to which a probability distribution is associated (i.e. events for which data are available). From a mathematical point of view a Fault Tree is a Boolean formula. Variables correspond to basic events of the tree, internal tree nodes or gates correspond to formula connectives, and the final equation of the formula represents the top event of the tree.

Fault trees are classified according to their logic into coherent and non-coherent categories. In a coherent fault tree, each component in the system is relevant, and the structure function is monotonically increasing. A fault tree that contains only AND gates, OR gates, and/or independent events is always coherent. Whenever a NOT logic gate is introduced or directly implied into a fault tree, the latter is likely to become non-coherent. In non-coherent fault trees, the working or success states of components as well as their failures (negative and positive events respectively) contribute to the failure of the system. If the NOT logic can be eliminated from the fault tree, the fault tree is coherent. If the NOT logic cannot be eliminated from the fault tree, the fault tree is non-coherent. For a more precise definition of coherency based on the structure function of the fault tree, see [7].

Traditional solution of coherent fault trees involves the determination of the so-called minimal cut sets (MCS). They represent the minimal combinations of component failures leading to a failure. For coherent fault trees, this definition matches the formal notion of prime implicant and so the function can be expressed as a disjunction of all its MCS. However, this is not the case for non-coherent fault trees because these no longer have the monotone properties. For this later case, the notion of MCS should be replaced by the notion of prime implicants. The mathematical details of these concepts are expounded in [8].

Event Trees constitute an inductive technique used to examine all possible responses to a potential hazardous initiating event (called initiator). It works forward in time considering all possible subsequent events until the consequences are known – either the system reaches a stable state or some level of failure or damage occurs. Branch points on the tree structure represent the success or failure of systems and operator actions designed to respond in order to mitigate the initiating event. In its graphical representation, upper branches represent successes of the corresponding safety system or event, while lower branches represent its failure. Note that the existence of the success branches makes the Event Trees intrinsically non-coherent.

Concerning the integration of the fault trees and event trees models, in the nuclear PSA studies, there has been traditionally two different modelling approaches: the fault trees linking approach and the event tree linking approach [9]. They both utilize a combination of fault trees and event trees to represent the model, although they differ in the degree of emphasis in the use of the models, and in the manner in which the system fault tree logic model are combined to represent the entire accident sequences. A large majority of PSA models worldwide uses the linked fault tree approach as the preferred modelling approach. Throughout this paper we will consider this approach.

In the linked fault tree approach, the failure of each mitigating system is described by means of a fault tree whose top event is the failure of the system. Each path of the tree, starting at the initiating event and ending into a final consequence defines a sequence of the event tree. Of interest on the safety analyses are the sequences leading to a damage state of the system. Each individual accident sequence is obtained by first logically linking the FT logic models for the top events along the sequence path of interest. Those that are successful along the sequence path will appear negated at the top event. Because the modelling approach uses relatively small Event Trees to represent the accident scenarios, the fault trees models are developed to represent the same system under different initiating conditions. In addition, they are characterized for including common support systems. This causes that the Fault Trees have repeated subsystems thus sharing basic events, which becomes of crucial importance in its analysis.

The sequences of the Event Tree are by construction mutually exclusive, so they can be analysed separately. The Boolean equation associated to the sequence corresponds to the Boolean product of each of the fault trees (or basic events) corresponding to each safety system defining a branching point. We will denote by S a sequence in which there are n failed and m success systems. All F_i and G_j are coherent: $S = F_1 \cdot ... \cdot F_n \cdot \overline{G}_1 \cdot ... \cdot \overline{G}_m$.

Due to the strong dependencies among the fault trees, this equation is a non decomposable function so the sequence has to be treated entirely, as will be explained in the next sections. Moreover these logical connections between the fault tree models increase the degree of redundancy of the model and therefore its complexity. Additionally, reflecting the ET formulation, this equation is non-coherent despite of being constructed from a set of monotone formulae, as a consequence of the negations appearing at the top event of the success branches.

The use of NOT gates to represent the successful events also increases the complexity of the problem because the algorithms needed to analyse the system are more complicated.
Summarizing, PSA models are made of large, complex, non decomposable and non coherent models. The concurrence of all these features makes the model especially difficult to deal with.

2.3. A case study

The case study selected comes from a "medium-sized" Event Tree belonging to a real world PSA study corresponding to a Spanish Nuclear Power Plant as shown in Figure 1. It depicts the evolution of a large loss of coolant accident in a BWR plant. Following the initiator (I), several systems are needed to mitigate the accident. Firstly, the reactor trip function (E1) shuts down the reactor. Its failure leads to a different event tree (not addressed here). After the successful reactor trip, injection in the vessel is needed to cool the reactor core. Failure sequences include the failure of both the High Pressure Core Spray System, HPCS (F1), and the low-pressure injection systems (Low Pressure Core Injection, LPCI, and Low Pressure Core Spray, LPCS) (F2). Success of any of those still needs long-term cooling, RHR (F3), or containment spray, CS (F4), to maintain core conditions. If they are not effective, then containment venting, CV (F5), or suppression pool makeup, SP (F6), are needed to cope with the accident. Failure of the containment venting needs the recovery of RHR (E2) and inventory control with systems outside the containment, CI, to avoid damage (E3). Failure of containment venting also needs inventory control with systems outside the containment to avoid damage. Capital letters F corresponds to Fault Trees, while capital letters E correspond to basic events
It has a total number of 1649 basic events and 9 functional events defining the branching points from which 6 of them are defined by means of Fault Trees besides an extra fault tree containing the conditions in which operation is prohibited by the plant's Technical Specifications which has to be deleted from the final solution (F0).
Table 1 presents some statistics regarding the number of basic events (#BE) and number of MCS (#MCS) of the Fault Trees, to reflect an idea of the size of the models. Although the level of the model complexity can be measured by its size, it is important to consider other aspects such as the dependencies among the systems and components, which influence as well the difficulty to solve it. As it happens with most of the PSA studies, strong dependencies exist between the Fault Trees of this Event Tree. Table 2 shows the matrix of shared variables for each pair of Fault Trees. Instead, the diagonal displays the total number of variables of the

model. A group of 4 fault trees (F2, F3, F4 and F6) have strong dependencies – the percentage of common variables with respect to the total size of the domain ranges from 68 to 88%. The same applies for the number of common gates.

The model has a total number of 19 sequences, although only 6 of them are of interest in terms of the safety analysis as they lead to a core damage state for the system. The largest sequence has a total number of 1395 basic events ant the smallest 1115. In subsequent sections we will discuss the results for the sequences analysis in more detail.

3. Current approaches for the PSA model assessment

In this section we review the two different existing approaches to asses Boolean models in the context of PSA studies, namely the classical approach based on the computation of MCS and the most recent based on Binary Decision Diagrams.

3.1. The classical PSA methodology

The methodology used to quantify Event Trees has almost not changed since the conception of the technique back in the 1970s when it was successfully used in the WASH 1400 study. In fact, the majority of the commercial tools used for reliability analyses implement this classical approach, often referred as the MCS approach, since the methodology relies on the computation of the minimal cutsets of the Boolean equation which defines the model.

As it was already mentioned, sequences of the Event Tree are by construction mutually exclusive, so they can be analysed separately to obtain the frequency of each scenario. However, analysing each sequence turns out to be more difficult. In the usual case, the Fault Trees defining the safety systems corresponding to the branch events of the Event Trees are not disjoint. Thus, quantifying the probability of each sequence is more complex than just finding the product of the frequency of the initiating event with the probabilities of passing along each branching point. On the contrary, when dealing with dependencies between the top events it is necessary to compile the Boolean equation of the whole sequence, which is obtained by combining the fault tree models appearing along the sequence path under a higher AND gate, leading to a master FT. It is important to recall that, since the sequence record may contain success branches, the resulting master FT corresponding to the sequence will be, in general, non coherent.

Owing to both the complexity of the calculation and the large size of the resulting equations for the real models, several approximations have to be considered when applying this methodology, namely,

– the truncation in the MCS computation,
– the approximation in the probability calculation and,
– the treatment of the success branches and the negations.

The remainder of the section is devoted to explain briefly these issues.

3.1.1. Truncation of MCS

It is well known that the number of MCS of a Boolean equation grows exponentially with respect to the number of variables. In the case of the Boolean equation extracted for PSA studies this number is by far too big to be computationally treated. Therefore, it is necessary to eliminate those minimal cut sets that make a negligible contribution to risk estimates and to keep only a subset of the whole set. A common practice is to truncate the MCSs. For this purpose, a truncation threshold is established to eliminate negligible parts of the equation during the development of the equations. Classical algorithms to compute minimal cutsets (which are based on working top-down [10, 11] or bottom-up [12] or both of them [13]) apply cutoffs on probability (or frequency) and order of the MCS to avoid combinatorial explosion in the number of MCS. In general, only a few thousand cutsets are kept. The choice of the right truncation value is a result of trade-off between accuracy and computational time efforts and memory space requirements. Current truncation values as suggested in the security procedures guides of the regulatory bodies range from 10^{-8}/year to 10^{-12}/year with respect to the core damage. However, since this truncation limit is subjectively or heuristically selected by the analyst, the top event probability will be underestimated due to the truncation, since a large number of MCS are discarded. The extent of the deviation might not be known if the exact value is not available. Thus, truncation is a factor contributing to the loss of accuracy in the quantification. This issue has always been of great concern in the nuclear related PSA studies. Consequently several methods have been proposed to estimate the truncation error and to determine the proper truncation limits to improve the efficiency and accuracy of the MCS-based approach [14, 15]. Still, these works have been proposed mainly for coherent models, so it remains necessary to consider the problem in the context of the event tree models and for the treatment of success branches.

3.1.2. Approximation of probability

Given a finite set of pairwise not disjoints events, the probability of the union of all the events is given by the exclusion-inclusion equation also known as the Sylvester-Poincaré development as expressed in (1)

$$p(E_1 \cup E_2 ... \cup E_k) = \sum_{1 \le i \le k} p(E_i) - \sum_{1 \le i < j \le k} p(E_i \cap E_j) + ... + 1^{-k} p(E_1 \cap ... \cap E_k) \quad (1)$$

When applied to a sufficiently large amount of MCS, which are non-disjoint in general, this equation quickly becomes intractable due to the exponential blow up of terms. Therefore, the probability of the union of the set of relevant MCS is obtained by approximating the above equation up to a given order. If only the first order terms are used, the approximation obtained is the so called Rare Event Approximation (REA), as it ignores the possibility that two or more rare events with low probabilities can occur simultaneously (Equation 2). The use of this approximation is fully justified since the number of basic events with high probability is limited, but becomes more questionable when events with high failure probability are considered.

$$p(E_1 \cup E_2 ... \cup E_k) \approx p_{REA} = \sum_{1 \le i \le k} p(E_i) \quad (2)$$

Another commonly used quantification approach which requires the same amount of computational resources as the previous one is the so called Min Cut Upper Bound (MCUB) approximation which is expressed in equation 3.

$$p(E_1 \cup E_2 ... \cup E_k) \approx p_{MCUB} = 1 - \prod_{i=1}^{k} (1 - p(E_i)) \quad (3)$$

In any case, both approximations are upper bounds of the exact results and therefore give conservative results.

3.1.3. Treatment of success branches

Another critical problem with the classical approach stands in the way success branches are taken into account for the quantification. None of the classical algorithms are able to deal with negations because, by definition, MCS does not contain negative literals.

The rigorous way to treat these negations, which appear uniquely at the top gates of success branches, is to treat them as logical complements in generating minimal cutsets. However the use of negations dramatically slows the Boolean reduction process so it might be impractical in terms of computing time to generate minimal cutsets that account for the negations even with the fastest cutest generation algorithms. In practice current tools offer different workarounds to

take success branches into account [4]. The most simple and trivial consists in ignoring success branches, treating the sequence as if it was coherent.

$$p(S) \approx \sum_{\pi \in MCS[F_1,...F_n]} p(\pi) \qquad (4)$$

Another approximation consists in correcting the previous one by introducing a factor computing the probability of success:

$$p(S) \approx \left[1 - \sum_{\pi \in MCS[G_1+...+G_m]} p(\pi) \right] \left(\sum_{\pi \in MCS[F_1,...F_n]} p(\pi) \right) \qquad (5)$$

A third and better approximation, used in most tools, is the delete-term approximation which consists in removing from the set of minimal cutest of the coherent sequence $MCS[F_1,...,F_n]$ the cutsets contained in any cutset of the negated top events $G_1 + ... + G_m$. It can be shown that, provided that all F and G are coherent, this operation leads to the minimal cutsets of the sequence $MCS[S]$:

$$p(S) \approx \sum_{\pi \in MCS[S]} p(\pi) \quad (6)$$

However, truncation on the computations of the MCS of both the coherent and non coherent sections of the sequence might invalid this last result.

Finally, it has to be highlighted that the effect of this set of approximations goes in different directions: while the MCS truncation derives in a optimistic approximation in what concerns to the coherent part of the model, the approximation on probability and the effect of the MCS truncation on the negated terms of the sequence lead to a over estimation of the exact value. Therefore, even if the final result might be close to the exact one, there is no warranty that it remains above or below the exact result as we are left with unknown error bounds in both directions. Moreover, over-conservative results may lead to wrong interpretations of risk importance measures as they are usually calculated from the resultant truncated MCSs.

3.2. BDD methodology

Binary Decision Diagrams are a compact data structure to encode and manipulate Boolean functions. Originally developed by Bryant [2, 3], they provide a compact way of representing the logic underneath a Boolean function. This section will briefly describe the basic concepts related with this technique and discuss its application to the assessments of FT and ET models.

3.2.1. Definition and properties

Formally, a BDD is defined as a directed acyclic graph (DAG) derived from the binary decision tree induced by the Shannon decomposition of the structure function. Let f be a Boolean function, and x a variable included in its domain. Then, the Shannon decomposition of the function, which is expressed through the *ite (if-then-else)* connective, allows dividing the function into two disjoints components both of which do not depend in x:

$$f = ite(x, f\big|_{x=1}, f\big|_{x=0}) = (x \wedge f\big|_{x=1}) \vee (\neg x \wedge f\big|_{x=0}) \qquad (7)$$

By choosing a total order of all the variables and applying Shannon decomposition recursively, the truth table of the function can be represented by a binary tree. This Shannon tree is, in general, not compact, and contains redundant nodes and isomorphic sub-graphs. It is, however, possible to apply reductions rules:

- Isomorphic sub-trees merging: identical formulae are eliminated as they encode the same formula.
- Useless node deletion: a node with two equal sons is equivalent to itself ($f=ite(x,f,f)$).

By applying these two rules exhaustively one get the BDD associated with the formula and guarantees the two most important properties of the BDD: the final representation is unique up to isomorphism, and is the most compact representation of the truth table of the function. A BDD is composed of terminal and non terminal nodes connected by branches. Each internal or non terminal node of the DAG represents a variable of the function, and has two outgoing labelled branches. They indicate either occurrence or non-occurrence of the variable: left or then branch, labelled with 1, and represented by a solid line, points to the son node encoding $f(x=1)$, and right or else branch, labelled with 0, and represented by a dotted line, points to the son node encoding $f(x=0)$. All paths through the BDD start at the root vertex and terminate at one of the two terminal nodes, labelled 0 and 1, which represent the constant functions. Paths leading to terminal node 1 give the assignments that satisfy the Boolean function. If the BDD is not minimal not all of those solutions will be minimal. Figure 2 shows the Shannon tree and the derived BDD of the function $f = x_1 \cdot x_2 + x_3$.

BDD construction is never done by reducing the previously built Shannon tree. Instead, the construction is done in a compositional way, by composing smaller BDD corresponding to its sub-formulae

The Shannon decomposition defines a recursive procedure to the construct the BDD from a given Boolean expression, starting from the root node, and descending in the decision tree.

However, in practice BDD construction is never done by first building the Shannon tree and then applying the reduction rules. Instead, the construction of the BDD is done using a compositional approach by decomposing the Boolean expression into smaller sub-expressions, the BDDs of which are calculated and then combined with the appropriate connectives to create more complex BDDs. In this way, new nodes are incorporated to the graph in a bottom-up way. In the case of a fault tree model, individual BDDs are assembled sequentially from the bottom gate to the top gate of the tree. The use of a table of nodes (called '*unique table'*), guarantees the application of reduction rules while constructing the BDD. This way, each node is constructed only once. For more details about those algorithms, the reader should refer to [16-18]. For details about issues related with an efficient implementation of a BDD package, see [19, 20].

3.2.2. Application to FT/ET assessment: improvements & drawbacks

Fault Tree analysis is a two-fold problem: it involves both quantitative and qualitative aspects. The application of the BDD technique on the Reliability Engineering field has made it possible to improve the efficiency and the accuracy of both sides of analysis, allowing exact probabilistic quantification, compact encoding of the MCS, and correct handling of negative logic and non-coherent systems. In particular, one of its mayor advantages is that the probability of the top event can be obtained directly from the BDD, as a sum of probabilities of the disjoint paths through the BDD and without the need to evaluate the MCS as intermediate results. As a result of the Shannon decomposition, which divides the system logic into two disjoints parts at each node, the following equation holds:

$$p(f) = p(x) \cdot p(f\big|_{x=1}) + (1 - p(x)) \cdot p(f\big|_{x=0}) \qquad (8)$$

This equality induces a recursive algorithm which is linear in the size of the BDD and gives exact values [17]. In fact, this method is extremely efficient because only one pass through the BDD structure is required to calculate all measures including importance factors [21].

Despite all its good properties and the improvements that the BDD method offers, there are some computational difficulties when working with BDD which are of special importance when dealing with large models such as the ones coming from the nuclear PSA studies. The methodology requires to build the BDD to be able to analyze the system it encodes (recall that in our case this stands for the equation of the whole sequence). To start the BDD codification an initial ordering of the variables on which the function depends has to be fixed. It is well

known that the final size of the BDD and therefore the efficiency of the method depends heavily of the chosen variable ordering, as pointed out by Bryant since the very beginning. As a result, the process, which might be very time and memory consuming, is, in addition, dramatically affected by the ordering selected. Choosing a bad initial ordering may lead to a blow up of the memory consumption and therefore to the impossibility to produce any BDD at all for large models.

Finding the optimal ordering is a NP-hard (Non-deterministic Polynomial-time hard) problem [22] owing to its combinatorial nature, so it is computationally intractable [23]. In order to find reasonable good orderings, research efforts have been aimed to design suitable heuristic methods, including good pre-processing strategies. Over the last 20 years many techniques have been proposed and a large number of articles have been published on those subjects. For a good review of the topic the following articles offers a more detail explanation and subsequent references [24-27]. In addition it has to be mentioned that there is no universal ordering scheme that can be successfully used to produce a minimal BDD for any fault tree model, which means that a specific strategy might work perfectly for particular model and be unsuccessful with other.

3.2.3. BDD approach in PSA models

Nowadays the BDD methodology can be considered a mature technology in the Reliability Engineering field. Up to now, most of the work has been applied to assess successfully small and medium size fault tree models, typically with up to several hundreds of basic events. This consolidation is demonstrated by the development of several software tools based on the BDD technology such as ARALIA, initially developed at Bordeaux University, ASTRA, developed by the EU-Joint Research Centre for security related applications, or FTRex, developed by the Korea Atomic Energy Research Institute.

Over the last years several works have been dedicated to extend the BDD technique to the Event Tree assessment for both coherent and non-coherent systems in various fields of application [4-6, 27-29]. However, when considering PSA models coming from the nuclear field, this approach turns out to be more difficult due to the size and the complexity of the models. As was previously mentioned, current PSA models consist of a large number of basic events and gates and are characterized for having many of the events repeated within the tree or sequence structure. In most of the cases, for such large models, it becomes impossible to fully convert the model to a BDD due to the exponential blow up of the memory requirements.

Specific models have been assessed only by using different techniques developed to solve this particular models: in [4] a set of dedicate strategies based on pre-processing methods and different static heuristics allowed handling of the model within reasonable times and memory consumptions; in [5] it is proposed to combine several static and dynamic ordering techniques, some of which are applied locally to the common variables of the model, although the computational cost is extremely high. Despite these works, there is no guarantee of being successful in general.

Although it is commonly accepted that current PSA tools need to be improved with more accurate and efficient algorithms and programming techniques in order to overcome some of its deficiencies (as was reviewed in section 3.1), doubtlessly current PSA models are too big and complex to be tackled by the BDD technology uniquely. Yet, application of BDD needs further improvements to adequate the technology for the correct treatment of such type of models. New approaches need to be propounded so as to consolidate the BDD technology as a feasible solution to improve PSA assessment tools in the nuclear field.

4. Model pruning in the MCS approach

Probabilistic analyses are performed with codes that produce results which have not been contrasted with alternative methodologies to validate them. The basis of the MCS approach is to reduce the model to a manageable size by discarding the less significant failure modes and to produce only the most relevant failure paths. This approximation is justified by the fact that these cutsets capture, in general, the most relevant parts of the model in terms of the contribution to the top event probability. Even so, it is important to draw attention to the manner in which the model is approximated with this type of approach. To illustrate this problematical issue, this section is devoted to present the results of the MCS approach obtained by means of the classical algorithms and to compare them with the exact results (when possible) in terms of the final probabilities as well as in the reduction performed in the model.

First of all, several results are presented concerning the quantification of the isolated fault tree models of the case study as a first step to study the practicability of this approach. The MCS were obtained by means of classical algorithms with three different absolute cutoffs values (10^{-10}, 10^{-11}& 10^{-12}) [30]. Table 3 shows statistics regarding the size of the models such as the number of basic events and minimal cutsets for each example for the full model and with the different cutoffs. It can be seen that the bigger the model is, the larger the reduction is in terms

of the number of basic events considered as relevant in the model. For these models, it was possible to compute the BDD of the full fault trees, so it was possible to use the exact results to compare with the MCS-based results. Columns 5 and 6 of Table 3 shows the top event probabilities computed with the classical tools using the MCS approach and the first order approximation (i.e. the rare event approximation) and the probabilities computed with the BDD approach respectively. Regarding the comparison between both approaches, differences can be seen to be very small. Additionally, it can be seen that the final results with the MCS approach are conservative only thanks to the rare event approximation. This can be assured because the Fault Tree models are coherent functions. However, it could happen that results were not conservative if the cutoffs are not sufficiently low as it happens with F6.

Only the sequences leading to core damage were quantified with the classical tools. Again, three different absolute cutoff values were employed (10^{-12}, 10^{-15} & 10^{-18}), both considering the success branches and not ("coherent" sequences). Results are given in Table 4. For each one of the cases, and for each cutoff value, the number of relevant MCS of the model, the number of basic events which appears on the relevant set of MCS, and the top probability obtained from the set of MCS, are presented. As with the previous results the BDD of the full model was undertaken to compute the exact values (final column of Table 4). Because it was only required to have the exact values mainly for comparison purposes, the cases were executed using dynamic variables reordering which turned out to be extremely time consuming. All cases were run in a computer with a Pentium Core Duo CPU at 3.16 GHz and 3.85GB RAM. For the cases that were finally solved it was required up to 120 hours of execution time. Moreover, even with this type of optimization techniques, for some of the complete sequences it was impossible to construct the BDD due to memory blow-up (memory resources were over 2 gigabytes for sequences 13 and 16).

Several remarks can be done from the analysis of this table. First of all, it can be seen that the differences between both approaches are more significant than the ones obtained for the analysis of each of the Fault Trees, due to the fact that models are bigger and more complex and, above all, because they are non coherent. As was already stated these results confirm that, when models include negations, ignoring success branches leads to even more over-conservative results. Secondly, the number of MCS and basic events that are actually used for the MCS-based calculations is extraordinarily small, although, with appropriate cut-off values the results given tend to be reasonably good.

Other experiments [4] reveal that this behavior is not particular of this model, and that, for most of the cases, only a modest fraction of the solutions of the model, and thus a small number of basic events, is actually used for the computations with the classical algorithms. Hence the MCS approach can be considered as a pruning process of the model. Because it tries to retain the most relevant solutions of the model in terms of its contributions to the final probability, it is normally considered an efficient way of pruning the model. However, there are some problems related with the pruning of the model performed by the MCS approach related mainly with the truncation process and also with the treatment of the success branches. First, small changes in the cutoff value often leads to the elimination of many MCS as their probabilities are usually clustered around a small subset of values. As a consequence the MCS computations process is too drastic and non smooth. Secondly, it is not model related in the sense that there is no control over the direct effect of the pruning process on the model during the MCS computation. Thirdly, as it was previously mentioned, success branches are badly taken into account because the truncation process is as well applied into the negative fault tree models before introducing them into the model computation.

Although the unique means of overcome the truncation problems is to obtain the BDD of the full model, from a practical point of view and due to the sizes of the industrial PSA models, we cannot avoid the model reduction. Still it is necessary to improve this pruning process to be able to accurately control the final quantification results. Therefore, suitable pruning processes are required.

5. Hybrid approach for ET assessment based on syntactical reduction

As a result of what has been concluded in the previous section, the key issue is that the reduction process should be more understandable and better controlled than it is when the model is manipulated through the one based on the MCS expansion and truncation. Moreover, it is required to be compatible with the treatment of negative logical models. Several works have proposed hybrid methods to apply the reduction in the progress of the BDD construction using as well truncation limits, leading to a truncated BDD [8, 12].

Instead, we propose that the reduction process should be applied directly to the model. This would allow obtaining a derived sub-model which could be quantified by means of the BDD technology, to take benefit of its good properties, that is, the exact assessment and, above all, the correct treatment of negations. In previous studies [30] the authors experienced that in order

to have an efficient transformation of the derived sub-model to a BDD encoding, it was essential to keep the topological information of the model, as the computation of the BDD from the MCS representation of the model was much more expensive. Therefore it is necessary to obtain a domain reduction by means of syntactical transformations which preserve the model topology. These transformations should always provide a domain reduction but may reduce or expand the model solution space depending on the purpose of the transformation. When achieving a reduction of the solution space, it could be used as a lower approximation of the model. On the other hand, when an expansion is performed the resulting derived model will be an upper approximation of the original one. Hence, it is needed to establish the mathematical basis of this approach as well as the practical implications, all of which is discussed in the following sections.

5.1. Syntactical model transformation

This section is devoted to state the mathematical grounds of the proposed syntactical transformations and thus some notation has to be introduced. Let F a monotone Boolean function. Let F^L be a derived sub-model of F obtained by eliminating part of its solutions, thus, constricting the solution space. The simplest process will be to fix the value of a set of variables to a constant value, $F^L = F_{V\leftarrow 0}$. *A priori* the set V can be any subset of the domain of F; for the case of a fault tree model it will be obtained from the analysis of the fault tree using the classical MCS approach as will be detailed later, hence the name of hybrid approach. Then the following implications holds: $F^L \models F$, that is, that all satisfying assignment of F^L satisfy as well F, and conversely $\overline{F} \models \overline{F^L}$. We call F^L the lower approximation or reduction of F. On the contrary, we denote by F^U another derived model of F obtained by expanding the solution space, so that $F \models F^U$, that is, where all the solutions of F^U are solutions of F. Just as previously, it holds that $\overline{F^U} \models \overline{F}$. For instance, this expansion can be achieved by performing the following transformation: $F^U = F_{V\leftarrow 1}$. Notice that although the domain is reduced, the solution space is augmented.

Now consider F and G monotone Boolean functions. Let H be defined as: $H = \overline{F} \cdot G$ and define the following transformations of H: $H^L = \overline{F^U} \cdot G^L$, and $H^U = \overline{F^L} \cdot G^U$.

Each term of these transformations of H has the following properties.

- If $F^U \dashv F$ and $G^L \models G$, then $\overline{F^U} \cdot G^L \models \overline{F} \cdot G$, so $H^L \subseteq H$, i.e. $H^L \models H$.
- Conversely, if $F^L \models F$ and $G^U \dashv G$, then $\overline{F^L} \cdot G^U \dashv \overline{F} \cdot G$, so $H \subseteq H^U$, i.e. $H \models H^U$.

The conclusion says that the set of minimal solutions of function *H* is bounded by those of H^L and H^U, which in practice means that, in the quantification, the exact result lies within an interval that can be readily obtained.

Now, consider the reduction transformation introduced at the beginning based on the constant substitution $F^L = F_{V \leftarrow 0}$. This transformation is equivalent to the MCS truncation process, if we set *V* to be the set of variables appearing in the discarded MCS. Considering what was presented in section 3.1, we recall that the MCS approach perform such a reduction process in all of the coherent functions of the sequence equation, to proceed afterwards to apply the delete term. Therefore, it can be assimilated with this third transformation of the model: $H^A = \overline{F^L} \cdot G^L$. Although H^A is neither a superset nor a subset of *H* in terms of the solution space, it can be considered a practical approximation of *H* in the context of the PSA models. Then, the transformations of *H* satisfy the following relations:

$$\begin{array}{l} H^L = \overline{F^U} \cdot G^L \\ \quad \| \\ H = \overline{F} \cdot G \quad \approx \quad H^A = \overline{F^L} \cdot G^L \\ \quad \| \\ H^U = \overline{F^L} \cdot G^U \end{array} \qquad (9)$$

The previous results can be generalized for the case of a model defined by a product of several coherent functions F_i and G_j. For the general case, some of these functions could appear negated, as it happens with the accident sequences of the PSA models.

Thus, consider again *S* to be a general event tree sequence: $S = F_1 \cdot \ldots \cdot F_n \cdot \overline{G}_1 \cdot \ldots \cdot \overline{G}_m$. The practical application of these mathematical developments allows obtaining a bounding interval for the quantification results in terms of an upper and a lower approximation of the Boolean equation of S, and the approximation S^A. Note that these developments are compatible with and formalize the current delete-term operation. In order to obtain an upper approximation of *S* a syntactical transformation is needed such that positive functions F_i are upper approximated while the negated functions G_j are lower approximated so that $\overline{G}_j$ is also upper approximated. Conversely, to obtain a lower approximation of S the required syntactical transformations

should perform the opposite operations. In that case we would obtain the following upper and lower transformations of S : $S^U = [F_1]^U \cdot ... \cdot [F_n]^U \cdot \left[\overline{G}_1\right]^U \cdot ... \cdot \left[\overline{G}_m\right]^U = F_1^U \cdot ... \cdot F_n^U \cdot \overline{G_1^L} \cdot ... \cdot \overline{G_m^L}$ and $S^L = [F_1]^L \cdot ... \cdot [F_n]^L \cdot \left[\overline{G}_1\right]^L \cdot ... \cdot \left[\overline{G}_m\right]^L = F_1^L \cdot ... \cdot F_n^L \cdot \overline{G_1^U} \cdot ... \cdot \overline{G_m^U}$. Additionally, a third approximate estimation can be computed applying the lower transformation for all the models as previously: $S^A = F_1^L \cdot ... \cdot F_n^L \cdot \overline{G_1^L} \cdot ... \cdot \overline{G_m^L}$.

Finally an important comment is in order. This approach allows obtaining both an upper and a lower approximation of the model on a purely syntactical basis. Thus, both lower and upper bounds of the exact results, as well as approximate results can be computed. Not only this approach offers a formal process for the reduction of the model, it also has the added value of providing bounds in both directions. If appropriate transformation criteria are used, the results obtained can offer real knowledge of the final probabilities of the model and measure the error introduced by the MCS approach.

Appendix A lists the mathematical results in a formalized way.

5.2. Transformation criteria

The idea behind the transformations of the model previously presented is to eliminate variables of the model so that the domain is reduced and the derived sub-model is more easily converted to its BDD. It has been shown that these approximations work correctly as long as the solution space of the derived model is either a subset or a superset of the original one. Therefore, a transformation is valid if it both fulfills this condition and it is based on syntactically manipulating the model. Although any criterion supporting these two properties is acceptable from a theoretical point of view, it is necessary to consider as well the practical implications to obtain suitable transformation criteria. Considering both the theoretical and the practical conditions this section offers different criteria for the syntactical transformation. Advantages and practical difficulties are as well mentioned.

1. Constant propagation

As it was mentioned previously, the simplest method to reduce the domain and simplify the model is to set some of its variables to a constant value, $F_{CP} = F_{V \leftarrow x}$. Let denote MCS_k the minimal cutsets obtained with a truncation value k and consider V to be the set of variables that do not appear in these MCS_k, which correspond to the discarded variables of the model. Then,

setting the variables of V to 0 is equivalent to the truncation process of the MCS but with the advantage that, although the less significant MCS have been discarded from the derived model, it is still defined in terms of logical gates and therefore the topological information is preserved. So, in order to obtain a lower approximation of the model (to reduce the solution space), variables of V have to be set to 0, $F_{CP}^{L} = F_{V \leftarrow 0}$. On the other hand, to obtain the upper approximation (which expand the solution space) variables of V have to be set to 1, $F_{CP}^{U} = F_{V \leftarrow 1}$. This latter approximation has, however, an inherent difficulty which cannot be easily overcome in practice and render it not very suitable for its practical implementation. Recall that the models are made of several coherent functions. As each function is monotonous, all variables appear in its positive form. Thus, there exists a big asymmetry of the logic between setting its components to failure or to success. While setting variables to 0 only eliminates certain failure scenarios, the operation of setting variables to 1 may lead to fail the whole system. It is sufficient that any combination of the set V that has to be substituted to 1 constitute a solution of the model. If this happens, the upper approximation of the model corresponds to the full solution space, which is a too coarse grain approximation. Moreover, even if the set V is filtered to avoid this problem, still the final probability of the system is dramatically sensitive to this type of substitution, because these models are designed to be highly reliable. Therefore it is necessary to refine this criterion to control the impact of the transformation into the probability, especially for the upper approximation.

2. Merging

It is clear that, in order to avoid the previous problems, it becomes necessary to avoid substituting the discarded variables to 1, but still it is needed to eliminate them from the model. Instead of substituting them to a constant value, a finer approach will be to apply a merging principle to collapse these variables to a smaller number of them.

Different variants of the merging principle have been investigated and tested, although not all of them have given good results in practice. The idea of the general merging principle is to successively merge pairs of variables d_1 and d_2 from the set V of discarded variables in a new fictitious event, h, and to substitute the original pair by this new event in the model. This elementary process, which is repeated for any two pairs of discarded variables, is illustrated in Figure 3. It looks similar to the factorization stage of the Faunet reduction explained in [31], where pairs of events that always occur together in the same gate type are combined to form a

single event. However, in the merging principle suggested here, pairs do not appear together explicitly in the model, but are considered in order to manipulate and simplify it. If the new event is defined as the union of both variables, that is, as $h_U = d_1 \vee d_2$, then the transformed model will be an upper approximation of the original one. Instead, if the new event is defined as the intersection of both variables, $h_L = d_1 \wedge d_2$, the transformed model will be a lower approximation. Because the pair of events merged does not show up any more in the model but under the new event h (i.e. h_L or h_U), once the substitution is performed, the event h can be considered as a module, and therefore can be treated as a basic event, by assigning to it the corresponding probability: $p(h_U) = p(d_1) + p(d_2) - p(d_1) \cdot p(d_2)$ and $p(h_L) = p(d_1) \cdot p(d_2)$. This general principle can be applied to successive pairs of variables of the set *V*. The new variables created by this merging principle can be considered as non-important variables of the model, so that they can also be included in the set *V* and merged. It is important to notice that, because the principle is applied in a pure syntactical base, it is guaranteed that they are upper and lower approximations respectively.

The general principle can be applied in different ways. The most basic approach would be to apply it until all the variables have been globally merged into a single variable which collects all the information of the whole set of unnecessary variables. This can be called the pure global merging principle. However, it might be not necessary to merge all of them, or to merge all to a single variable. Instead, the principle could be applied to several subgroups of variables, obtaining a merging variable for each subgroup. Thus, different variants or refinements could be created.

The pure global merging principle gives results similar to those of the constant propagation for the lower transformation of the model, so both of them can be used. Instead, in the case of the upper approximation, the global merging principle offers very bad results as the final results is dramatically sensible to this transformation, so it has to be refined. The reason is that the new "modules" created to merge all the variables and to substitute them is the result of an OR gate, the probability of which is obtained by adding the probabilities of the variables that are merged (assuming that the cross terms are discarded). Since there are big differences between the probabilities of all the variables of the model (they might be several orders of magnitude of difference), the substitution of events with low probability by the new event propagates to a high overestimation of the result so that very high bounds are obtained. Just as it happened with the upper approximation of the constant propagation criterion $F_{CP}^{U} = F_{V \leftarrow 1}$, relatively small

modifications of the events that imply to decrease the reliability of some particular subsystems are propagated in the model with a great impact in the final probability.

A refined approach to overcome this problem is to cluster the variables by its probability and restrict the merging principle to each cluster, where only variables with similar probabilities are merged. The larger the number of created cluster is, the better is the final result although there is a smaller reduction of the domain, and in addition, several new variables are added to the model. The application of this refinement decreases the value of the upper bound with respect to the pure global merging principle. However, as several redundant variables corresponding to the representatives of each cluster are created, the complexity of the model increases in a way such that it becomes much more difficult to obtain the BDD. Hence, it is necessary to obtain a tradeoff between the decrease in the number of variables and the increase of the redundancy which ultimately affects the complexity of the model.

In order to obtain a more robust principle for the upper transformation of the model a third refinement has been devised, where the merging principle is applied locally at each gate of the model and only for the OR gates (as the new variables are defined to be this type of gates). So, instead of defining the groups according to the global clustering, they are defined as being part of the same gate, so that the topological relationships can be taken into account. For each different gate a new variable is created to substitute all of the discarded variables of the gate. Gates that are shared between different models are approximated by the same new variable, so that it is only created once and dependency relationships are maintained. Although the decrease in the number of variables is lower than with previous alternatives, this approach offers the best results in practice because the degree of redundancy added to the model is also lower, and therefore the tradeoff between simplification and redundancy is more balanced.

As it has already been stated, the upper and lower approximations cannot be treated with the same criteria owing to the big asymmetry that exist when the space solution of this type of models is modified. In general, all the criteria used to obtain the lower approximation of these models, which are coherent, are less problematic than the ones required for obtaining the upper approximations. In addition, the number of negated models appearing in the sequences is usually smaller than the positives, so that the upper approximation of the whole sequence, which requires obtaining the upper transformations of the positive functions, becomes more complicated. For these reasons, the best approach is to combine the different criteria so that the final approximation is accurate enough. Thus, for each approximation of the sequence, the

criterion to be used to transform each coherent model depends on whether it is a failure branch or not. For the upper bound, failure branches require an upper transformation while success branches require a lower transformation, and conversely for the lower bound. The most efficient method to obtain the lower transformation of each function is to apply the constant propagation principle, as it reduces both the domain and the complexity of the model and has almost no impact in the final probability (as showed in section 4). On the contrary, for the upper transformation, the most effective method in practice is the merging principle restricted to each OR gate, because it is the one that better controls the impact of the approximation on the final probability.

Additionally, the fact that the functions have common variables has to be taken into account explicitly, since the transformation procedures are applied in each of the coherent functions of the model, some of which appear in its positive form and others in its negative form. In the case of the negated coherent functions (a process that indeed transforms them into non-coherent functions) care must be taken in applying different criteria to the same variable if it appears in both the coherent and non-coherent functions. In order to keep the consistency of the whole model, it is necessary to first manipulate the models of different sign in order to make them disjoint. Recall that the reduction procedure is the same for all the functions having the same sign. Thus, it is only required to duplicate the common gates of the functions of one sign, for example, the ones that appear negated.

6. Experimental results

This section presents the numerical results obtained by applying this new hybrid approach to quantify linked fault tree models coming from PSA studies. This analysis considers four from the six accident sequences of the case study previously presented in sections 2.3. The other two sequences are not considered for being excessively trivial. The equations of these sequences are the following:

$$S_5 = \overline{F_0} \cdot \overline{E_1} \cdot \overline{F_1} \cdot F_3 \cdot F_4 \cdot \overline{F_5} \cdot F_6 \cdot E_3$$

$$S_8 = \overline{F_0} \cdot \overline{E_1} \cdot \overline{F_1} \cdot F_3 \cdot F_4 \cdot F_5 \cdot E_2 \cdot E_3$$

$$S_{13} = \overline{F_0} \cdot \overline{E_1} \cdot F_1 \cdot \overline{F_2} \cdot F_3 \cdot F_4 \cdot \overline{F_5} \cdot F_6 \cdot E_3$$

$$S_{16} = \overline{F_0} \cdot \overline{E_1} \cdot F_1 \cdot \overline{F_2} \cdot F_3 \cdot F_4 \cdot F_5 \cdot E_2 \cdot E_3$$

The three different approximations (S^L, S^U, S^A) derived from the application of theorems 1 and 2 for the case of a sequence S (equation (9)) and presented in the previous section haven been computed for each sequence and with a range of different cutoff values. In all the cases, the BDD have been obtained as follows. A cutoff value is selected to obtain the set of variables to be discarded in each of the Fault Tree models. These variables are the ones that do not show up in the MCS computed with this cutoff value. The Fault Trees are transformed using the criteria already explained depending on both the sign of the Fault Tree and the approximation chosen for the sequence. Each BDD is computed after the transformation of each Fault Tree model. Finally, all of them are composed by applying an incremental procedure as described in [27]. As a side comment, the first model F_0 is not reduced in any case because its main function is to delete solutions that are prohibited by the technical conditions. Also, components defined as basic events (E_i) are only considered at the end to multiply the final result by its probability.

Concerning the variable ordering issue, after transforming the Fault Tree models, but before computing its BDD, a total ordering of the variables that still belong to any of the models is obtained. To do this, a random rewriting of the model and the basic depth-first ordering scheme are applied. The procedure to rewrite the tree randomly is as follows. First, a unique index is associated randomly with each gate and basic event in the tree. Secondly, these indices are randomly sorted. Finally, the inputs of each gate are sorted according to these new indices. As it was mentioned previously, the basic depth-first traversal of the tree is one of the most popular heuristics and gives good results in many cases for it preserves the locality of the variables. However, they are very sensitive to the way the Boolean model is written, so it is necessary to take into account the aforementioned rewriting methods as part of the orderings [32]. In our experiment, a seed is generated to perform first a random rewriting to determine the order in which the Fault Trees are placed to be composed. In a second step, each Fault Tree model is rewritten as explained (rearranging the inputs of each gate).

As it was previously mentioned the advantage of performing the model transformation prior to the BDD conversion is that the topological information can be fully exploited, both for the ordering and for the BDD computation. Additionally, as they are combined employing the incremental approach, the transformations can be applied to each Fault Trees individually. This allows having better control on the reduction performed on each model, hence providing more flexibility to the process.

In this case the same cutoff was used for all the Fault Trees of the sequence for simplicity. Further developments may consider selecting different cutoff values for each Fault Tree to adjust individually the accuracy of its transformation before their combination in the sequence. Besides, the range of cutoff values that has been tested depends on the model characteristics and its complexity, and has been obtained experimentally. Starting from a reasonable value, it has been decreased until it was not possible to obtain the BDD. This point is reached sooner with the upper approximation as it is the one that eliminate fewer variables so the domain reduction is minor.

The aim of this experiment is to analyze the quantification results and to assess the precision of the bounds obtained, in order to determine if this approach provides an adequate interval which contains the exact value without computing it explicitly. In addition, the purpose of the experiment is to compare the results with the ones obtained with the MCS approach.

Results shown in this section include the final sequence frequency and the number of variables for each test. The latter is a good indicator of the level of reduction achieved and the percentage of the model that is considered. Although all the criteria that have been explained in the previous section were tested, only the results of the most effective method in practice are shown. In addition, running times are not given because the aim of this experiment is to analyze the quantification results. Also, it has to be mentioned that due to the lack of robustness against rewritings and the sensitivity to the initial writing of the model, the authors experienced that it was fundamental to obtain a good rewriting of the model to start with the process, that is, a good initial seed. Consequently, it is fundamental to perform a preprocessing step that experimentally finds appropriate seeds for each particular model before initiating the bounds computation.

All the experiments were run in an Intel Core Duo processor at 3.16 GHz and with 3.23 Gb of RAM. The results of the four sequences are shown in Tables 5-8, and are represented graphically in Figures 4-7 respectively, together with previous results of Table 4 (MCS approach and exact results). Results of this table (if they are known and non-zero) are drawn using horizontal lines. Additionally, a grey band has been drawn to represent the interval where the exact result must be, bounded by the more precise bounds of S^L and S^U. Omitted values of the tables were not computed because the simplification was too large to be worth it (the uppermost rows of S^L/S^A) or because they were too complex and run out of memory (the final rows of S^L/S^U). In the case of sequence 8, because the results of the three approximations are

very close, its graphical representation has been restricted in the Y axes to a narrower band in order to appreciate them properly. This excludes the results of the MCS approach.

In all the sequences except in sequence 13 it was possible to obtain a relatively small interval containing the exact result by applying the hybrid approach, thanks to the convergence of both approximations. For sequence 13 it was not possible to compute a better upper bound with any of the criteria studied. The difficulty may lay on the fact that this is the sequence with the smaller frequency, so it is more difficult to be quantified. In fact, in this case, it was possible to quantify it with the MCS approach only with the lower cutoff selected for our experiment (10^{-18}), which is certainly lower than the values used in practice.

On the contrary, for the remainder sequences the results offered by the hybrid approach demonstrate that the MCS approach is over estimating the real value. Moreover, when the cutoff is decreased, this over estimations becomes even bigger, which means that the MCS results does not converge to the exact value even though the cutoff is set to consider more MCS. This might be happening due to the effect introduced by the approximation in the probability (like the rare event approximation expressed in equation (2)) together with the incorrect treatment of the negated headers.

Among all the sequences the one that is more interesting is sequence 16. In this case, not only the exact result was bounded very efficiently, but it is shown that the MCS with a cutoff of 10^{-15} is underestimating the real value. In that case, the calculation of the lower bound can be used in addition to establish if the cutoff used in the classical approach is low enough to ensure that is, at least, above the exact result.

Finally it has to be remarked that in all the cases the results of S^L are very similar to S^A, although the advantage of the later is that it can be computed with lower cutoff values, because it allows eliminating more variables of the model. Thus, even if the approximation S^A cannot be considered from a strict mathematical point of view as a lower bound of the exact results, results confirm that it is a very good approximation from a practical point of view taking into account the special characteristics of these models. While the calculation of S^L and S^U offers a mechanism to compute the real interval where the exact result must be which is theoretically justified, the computation of S^A can be used to complement this analysis as a more practical estimator.

7. Discussion and conclusions

Binary Decision Diagrams have proved to be of great interest from both a practical and theoretical point of view. In the reliability engineering framework it can be considered a mature technology. However, large models in general, and specifically some of the ones coming from the PSA studies of the nuclear industry, are still out of reach of an exact evaluation. The models are too large and complex to be fully mastered with any of the current existing approaches and tools. It can be argued that these models are too large and complex to be fully analyzed and that approximate results are good enough in practical applications. However, it is important to highlight that these results have not been yet contrasted to verify its validity. On the other hand, the use of PSA studies has started to become more widespread and models have become more detailed and complex. The extensive use of these models for current and future applications requires a better understanding of the limitations of its techniques and demand methods to estimate more precisely the error associated with the results obtained, especially when taking into account that the amount of the truncation error is more sensitive to the size and the complexity of the model, which is above all due to the existence both of strong dependencies among the systems and of success branches (negative logic).

This article presents a new approach to estimate the exact quantification results based in combining the information provided by the classical MCS approach using truncation limits with the BDD approach. Thereby, a better control on the reduction of the model and a proper account for the success branches can be achieved. The core of the methodology is to transform syntactically the model to reduce its complexity. An important remark is that once the model is transformed, the topological information is still available, which benefits the BDD codification. Another key issue is that the process is applied to each component of the master FT defining the sequence equation. This allows, on the one hand, applying the reduction more smoothly as each of the fault tree models are more manageable, and, on the other, treating the positive and negative elements separately. In fact, an advantage of this approach is that the sets V can *a priori* be any set of variables. Moreover, if the MCS are used to select the unimportant variables, a different cutoff value for each fault tree could be used as it was previously mentioned. This gives large flexibility to the analyst concerning the reduction process because of the possibility to examine each fault tree model and to analyze whether there are some variables or subsystems that are to be kept on the model for any physical reason.

Current methods to analyze FT/ET models consist in considering a subset of all the solutions, namely, in evaluating a derived model which is a lower approximation, and estimating the truncation error. In contrast with this perspective, the added value of this new approach is that it allows transforming the model not only to obtain a lower approximation, but also an upper approximation consisting in a derived model which contains all the solutions of the original model. This allows ensuring a real confidence interval of the exact value, if the cutoff has been appropriately adjusted, because both approaches converge to the exact value. Subsequently, an explicit knowledge of the error bound is obtained.

The methodology presented here has been mathematically founded. In addition, the results confirm the applicability of the methodology to estimate the exact result. Besides this estimation, the results that are obtained can be used to measure the acceptability of the estimation obtained with the MCS approach. The results presented in this article for a real case study show that the curve obtained from the MCS approach does not necessary converge to the exact value when the cutoff is set to be inferior. This means that lowering the cutoff might lead to more overconservative results, while if it is set not low enough, the results could be under the exact result. With the approach exposed in this work, a formal method is offered to ensure this. First, the lower bound of the model allows validating if the cutoff selected in the current established analysis is low enough to guarantee that the result is not underestimated. In that case, it should be lowered until the results are at least above the lower bound of the model. On the other hand, the upper bound can be used to establish the effect of the overestimation on account of the approximation on the probability and the truncation or the suppression of the success branches. Although less relevant than the former, such an underestimation of the risk should not be acceptable either, because there is still an uncertainty of the error committed.

The methodology presented in this article allows the computation of three different approximations of the model. Regarding the two approximations which are demonstrated to be real bounds of model, S^L and S^U, the computation of the upper bound has proven to be more difficult to obtain. Sequences are mostly constructed with positive fault trees corresponding to the failure of the safety systems, which have to be upper approximated. In that case, and due to the special nature of these models, this is a more difficult task because the models are dramatically sensitive to a modification which implies making them less reliable. On the other hand, concerning the third approximation S^A, even though it is not properly a bound of the model, the results have confirmed that it is a very good approximation in practice for this type

of models. First, its behavior has turned out to be very similar to the lower bound S^L but with the advantage that, as the domain is more reduced, it can be computed with lower cutoff values, allowing a more precise estimation of the model. Secondly, it can be assimilated with the process followed by the MCS approach when the delete term is applied to manage the success branches. However, the advantage of this approach lies in that it does not suffer from the problematic of the approximation of the probability. Also, because BDDs are more efficient to encode the model, a more significant portion of it can be considered.

Future work should be directed to apply this methodology to bigger benchmarks to study the limitations of the approach, especially when considering the upper bound. In such cases there are still some experiments to be carried out regarding several aspects. One issue that has been already mentioned is the fine tune of the cutoff value for each fault tree model separately to optimize the reduction. A major issue where there is still room for improvement is the investigation of other transformation criteria that might be required to manage bigger models.

Acknowledgements

This research has been supported by the Spanish Safety Council (CSN).

Appendix A

Consider F and G monotone Boolean functions, and let H be defined as: $H = \overline{F} \cdot G$. The following syntactical transformations of H: $H^L = \overline{F^U} \cdot G^L$, and $H^U = \overline{F^L} \cdot G^U$ verifies the following properties:

Theorem 1: The space solution of function H is bounded by H^L and H^U so: $H^L \models H \models H^U$.

The proof follows for the transformation properties of each term of H. If $F^U =\mid F$ and $G^L \models G$, then $\overline{F^U} \cdot G^L \models \overline{F} \cdot G$, so $H^L \subseteq H$, i.e. $H^L \models H$. Conversely, If $F^L \models F$ and $G^U =\mid G$, then $\overline{F^L} \cdot G^U =\mid \overline{F} \cdot G$, so $H \subseteq H^U$, i.e. $H \models H^U$.

Theorem 2: More generally, the transformations of H satisfy the following relations:

$$
\begin{array}{l}
H^L = \overline{F^U} \cdot G^L \\
\quad \| \\
H = \overline{F} \cdot G \quad \approx \quad H^A = \overline{F^L} \cdot G^L \\
\quad \| \\
H^U = \overline{F^L} \cdot G^U
\end{array} \tag{9}
$$

References

[1] Vesely WE A time dependant methodology for Fault Tree evaluation. Nuclear Engineering Design 1970;13:337-360.

[2] Bryant RE. Graph-Based algorithms for Boolean function manipulation. IEEE Transactions on Computers 1986;C-35:667-691.

[3] Bryant, RE. Symbolic Manipulation with Ordered Binary Decision Diagrams. ACM Computing Surveys 1992; 24: 293-318.

[4] Epstein S, Rauzy A. Can we trust PRA? Reliability Engineering & System Safety 2005; 88: 195-205.

[5] Nusbaumer OPM. Analytical Solutions of Linked Fault Tree Probabilistic Assessment using Binary Decision Diagrams with Emphasis on Nuclear Safety Applications. PhD Thesis. Swiss Federal Institute of Technology. 2007.

[6] Andrews JD, Dunnett SJ. Event-Tree Analysis Using Binary Decision Diagrams. IEEE Transactions on Reliability 2000; 49:230-238.

[7] Beeson S, Andrews JD. Calculating the Failure Intensity of a Non-coherent Fault Tree using the BDD technique. Quality and Reliability Engineering International. 2004; 20: 225-235.

[8] Rauzy A. Mathematical Foundations of Minimal Cutsets. IEEE Transactions on Reliability 2001; 50: 389-396.

[9] Nuclear Regulatory Commission (NRC). A Guide to the Performance of Probabilistic Risk Assessments for Nuclear Power Plants (NUREG/CR-2300);1983.

[10] Fussel JB, Vesely WE. A new methodology for obtaining cut sets for Fault Trees Transactions of American Nuclear Society 1972; 15: 262-263.

[11] Rauzy A. Toward an Efficient Implementation of the MOCUS Algorithm. IEEE Transactions on Reliability 2003; 52: 175-180.

[12] Jung WS, Han SH, Ha JJ. A fast BDD algorithm for large coherent fault tree analysis. Reliability Engineering & System Safety 2004; 83:369-374.

[13] Contini S. A new hybrid method for fault tree analysis. Reliability Engineering and System Safety 1995; 49: 13-21.

[14] Cepin M. Analysis of truncation limit in probabilistic safety assessment. Reliability Engineering and System Safety 2005; 87: 395-403.

[15] Jung WS, Yang JE, and Ha JJ. Development of measures to estimate truncation error in fault tree analysis. Reliability Engineering and System Safety 2005; 90: 30-36.

[16] Rauzy A. New algorithms for Fault Trees Analyses. Reliability Engineering & System Safety 1993; 40:203-211.

[17] Rauzy A. A brief introduction to Binary Decision Diagrams. European Journal of Automation (Journal Européen des sytemes Automatises) 1996; 30:1033-1050.
[18] Rauzy A. BDD for Reliability Studies. In: Misra KB, editor. Handbook of Performability Engineering: Elsevier; 2008, p. 381-396.
[19] Brace KS, Rudell RL, Bryant RE. Efficient implementation of a BDD package. Proceedings of ACM/IEEE Design Automation Conference (DAC'90); 1990; Orlando, Florida, USA.
[20] Rudell RL. Dynamic Variable Ordering for Ordered Binary Decision. IEEE/ACM International Conference on Computer-Aided Design. ICCAD'93; 1993; Santa Clara, California, USA.
[21] Sinnamon RM, Andrews JD. Improved Accuracy in Quantitative Fault Tree Analysis. Quality and Reliability Engineering International 1997; 13: 285-292.
[22] Bollig B, Wegener I. Improving the variable ordering of OBDD is NP-Complete. IEEE Transactions on Software Engineering 1996; 45: 993-1001.
[23] Friedman SJ,and Supowit KJ. Finding the optimal variable ordering for Binary Decision Diagrams. IEEE Transactions on Computers 1990; C-39: 710-713.
[24] Bouissou M, Bruyère F, Rauzy A. BDD Based Fault Tree processing: a comparison of variable ordering heuristics. Proceedings of European Safety & Reliability Association. ESREL'97, Lisbon, Portugual, 1997.
[25] Sinnamon RM, Andrews JD. Improved Efficiency in Qualitative Fault Tree Analysis. Quality and Reliability Engineering International 1997; 13: 293-298.
[26] Bartlett LM. Variable Ordering Heuristics for Binary Decision Diagrams. PhD Thesis. University of Loughborough, 2000.
[27] Ibañez-Llano C, Meléndez E, Nieto F. Variable Ordering Schemes to apply to the Binary Decision Diagram methodology for Event Tree Sequences Assessment. Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability 2008; 222:7-16.
[28] Cojazzi, GGM, Contini S, Renda G. On the need of exact probabilistic quantification in FT/ET analysis. Proceedings of European Safety & Reliability Association. ESREL'05, Tri City, Poland, 2005.
[29] Contini S, Cojazzi G.G.M. and Renda G. On the use of non-coherent fault trees in safety and security studies. Reliability Engineering and System Safety 2008; 93: 1886– 1895.
[30] Ibañez-Llano C, Rauzy A, Meléndez E, Nieto F. Minimal cutsets-based reduction approach for the use of binary decision diagrams on probabilistic safety assessment fault tree models. Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability 2009; 223(4): 301-311.
[31] Reay KA, Andrews JD. A Fault Tree Analysis strategy using Binary Decision Diagrams. Reliability Engineering & System Safety 2002; 78: 45-56.
[32] Rauzy A. Some Disturbing Facts about Depth First Left Most Variable Ordering Heuristics for Binary Decision Diagrams. Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability 2008; 222: 463-720.

List of Figures

Figure 1. The Event Tree of the case study with 19 sequences

Figure 2. From the Shannon tree to the Binary Decision Diagram

Figure 3. Application of the general merging principle for a pair of variables d_1 and d_2
Figure 4. Probability of sequence 5 with MCS approach vs. Hybrid approach
Figure 5. Probability of sequence 8 (detail of hybrid approach results)
Figure 6. Probability of sequence 13 with MCS approach vs. Hybrid approach
Figure 7. Probability of sequence 16 with MCS approach vs. Hybrid approach

List of Tables

Table 1. Statistics of the fault trees of the case study
Table 2. Matrix of the number of shared basic events between the fault trees of the case study
Table 3. Assessment of the Fault Tree models with MCS vs. BDD approaches
Table 4. Assessment of some of the Event Tree sequences with the MCS approach
Table 5. Probabilities of the three approximations for sequence 5 with different cutoffs
Table 6. Probabilities of the three approximations for sequence 8 with different cutoffs
Table 7. Probabilities of the three approximations for sequence 13 with different cutoffs
Table 8. Probabilities of the three approximations for sequence 16 with different cutoffs