Can We Trust PRA ?

Qui a vist Paris et noun Cassis, ren a vist.
If one has seen Paris, but not Cassis, one has seen nothing.
--- an old Provencal expression

W. Epstein(1) and A. Rauzy(2)

(1) ABS Consulting, Koraku Mori, Building, 1-4-14 Koraku Chome, Bunkyo-ku, Tokyo, 112-0004, Japan, sepstein@absconsulting.com

(2) IML/CNRS, 163, Avenue de Luminy, Case 907, Marseille, 13288 Cedex 9, France, arauzy@iml.univ-mrs.fr

Abstract : The Fault Trees/Event Trees method is widely used in industry as the underlying formalism of Probabilistic Risk Assessment. Almost all of the tools available to assess event tree models implement the “classical” assessment technique based on minimal cutsets and the rare event approximation. Binary Decision Diagrams are an alternative approach, but they were up to now limited to medium size models because of the exponential explosion of the memory requirements. We have designed a set of heuristics which make it possible to quantify, by means of BDDs, all of the sequences of a large event tree model coming from the nuclear industry. For one of the first times, it was possible to compare results of the classical approach with those of the BDD approach, i.e. with exact results. This article reports this comparison and shows that the minimal cutsets technique gives wrong results in a significant proportion of cases. Hence, our question in the title of this article.

1 Introduction

Katatsuburi
soro-soro nobore
fuji no yama
oh snail
climb Mount Fuji,
but slowly, slowly

--- *Issa*

The Fault Trees/Event Trees method is widely used in industry. Probabilistic Risk Assessment in the nuclear industry relies worldwide almost exclusively on this technique. Several tools are available to assess event tree models. Almost all of them implement what we call the "classical" approach: first, event tree sequences are transformed into Boolean formulae. Then, after possibly applying some rewriting rules, minimal cutsets of these formulae are determined. Finally, various probabilistic measures are assessed from the cutsets (including probabilities and/or frequencies of sequences, importance factors, sensitivity analyzes, …). This approach is broadly accepted. However, it comes with several approximations:

- In order to assess probabilistic quantities from the cutsets, the rare event approximation is applied. Under certain conditions the min-cut upper bound approximation can be used, but only when the boolean equation does not have negation and all basic event probabilities are quite low, at least smaller than 10^{-2}.
- In order to minimize cutsets, and therefore avoiding combinatorial explosion, probability truncation (hereafter referred to as simply truncation) is applied.
- Finally, in order to handle success branches, various recipes more or less mathematically justified are applied.

Since, up to now, all of the assessment tools rely on the same technology (with some variations indeed), it was not possible to verify whether the above approximations are accurate for large real-life models, especially since to compute error bounds, the exact solution is necessary

In the beginning of the nineties, a new technology was introduced to handle Boolean models: Bryant's Binary Decision Diagrams (BDD for short) [Bry86,Bry92]. One of the major advantages of the BDD technology is that it provides exact values for probabilistic measures [Rau93,DR00]. It does not need any kind of truncation or approximations. BDDs are however highly memory consuming. Very large models, such as event trees of the nuclear industry, were beyond their reach. Nevertheless, the methodology can be improved by means of suitable variable heuristics and formula rewritings.

Recently, we were given a rather large event tree model (coming from the nuclear industry). We designed a strategy, i.e. a sequence of rewritings, that made it possible to handle all of the 181 sequences of the model within reasonable running times and memory consumptions. For one of the first times, it was possible to compare results of the classical approach with those of the BDD approach, i.e. with exact results.

As the epigram to this section intimates, we should not draw definitive conclusions from a single test case. But a single example suffices to ring the alarm bell: the classical approach gives wrong results in a significant proportion of cases. This is true for sequence frequencies and, although to a lesser extent in the problem under study, for component ranking via importance factors.

The remainder of this article is organized as follows. Section 2 is devoted to terminology (Boolean formulae, event trees, …). Sections 3 and 4 present respectively the classical and the BDD approaches. Section 5 gives some insights on the test case we used for this study. Section 6 reports comparative results for the computation of sequence frequencies. Section 7 extends the comparative analysis to importance factors. Section 8 considers briefly the complexity, runtime, and space considerations when trying to solve large problems. Finally, section 9 presents our preliminary conclusions.

2 Terminology

Proper notation, the basting that holds the fabric of mathematics in shape, is both the sign and the cause of clear thinking.

--- to paraphrase Lynn Truss in EATS, SHOOTS & LEAVES

2.1 Boolean Formulae

Throughout this article we consider Boolean formulae. Boolean formulae are built over a denumerable set of variables and the connectives and, or, not, k-out-of-n, and so on. Their semantics is defined, as usual, by means of the truth tables of connectives. We denote by *var(F)* the set of variables that occur in the formula *F*. In the example to be studied, *F* represents a top event and the variables represent component failures, or basic events. We use the arithmetic notation for connectives: *F.G* denotes the formula *"F and G"* and *F+G* denotes the formula *"F or G"*. The formula "*not F*" is denoted either by *-F* or by $\overline{F}$.

A formula is coherent if it does not contain negations. From a strict mathematical viewpoint, this definition is too restrictive, e.g. *--F* is coherent (assuming *F* is). However, it is sufficient for our purpose.

A literal is either a variable or its negation. A product is a conjunct of literals. It is sometimes convenient to see products as sets of literals. A minterm of a formula *F* is a product that contains either positively or negatively each variable of *var(F)*. If *n* variables occur in *F*, 2^n minterms can be built over *var(F)*. In other words, minterms one-to-one correspond with truth assignments of variables of *F*. By abuse of notations, we shall write $\pi(F) = 1$ (resp. *0*) if the truth assignment that corresponds to the minterm π satisfies (resp. falsifies) *F*. We shall say that π belongs to *F* when $\pi(F) = 1$. A formula is always equivalent to the disjunction of its minterms.

Let π be a (positive) product and *F* be a formula. We denote by π_F^c the minterm of *F* built by adding to π all the negative literals built over the variables of *F* that do not occur already in π. For instance, if *var(F)={a,b,c}* and π=*a*, then $\pi_F^c = a.\overline{b}.\overline{c}$. We shall omit the subscript when the formula *F* is clear from the context.

Let π be a positive product and *F* be a formula. π is a cutset of *F* if π_F^c satisfies *F*. A cutset π is minimal if no proper subset of π is a cutset. We shall denote by *MCS[F]* the set of minimal cutsets of *F*. The reader interested by a more thorough treatment of minimal cutsets should refer to [Rau01].

2.2 Event Trees

The Fault Tree/Event Tree method is probably the most widely used for risk assessment, especially in the nuclear industry. We assume the reader is familiar with this method (see [KH96] for a good introduction).

Fig. 1 (left) represents an event tree. As usual, upper branches represent successes of the corresponding safety systems, lower branches represent failures. In the fault tree linking approach (the one we consider here), each sequence is compiled into the conjunct of the top events (for failure branches) or negation of top events (for success branches) encountered along the sequence. The Boolean formulae associated with the sequences of the above event tree are given on the same figure (right), assuming that the failures of each safety system are described by means of a fault tree whose top event has the same name as the system.

It is worth noticing that the above compilation is an approximation. In our example, safety systems *F*, *G* and *H* probably don't work simultaneously, but are rather called in sequence. We shall not consider this issue here. The reader interested by mathematical foundations of event trees should refer to Papazoglou's important article [Pap98].

3 The classical approach to assess event trees

Da Vinci was so steeped in his own tradition that each step he took trancended it.

--- Scott Buchanan, EMBERS of the WORLD

3.1 Principle

By construction, sequences of event trees are mutually exclusive. Therefore, they can be treated separately, at least for what concerns the computation of their probabilities.

The classical approach to assess event trees works as follows.

- First, sequences are compiled as explained above.
- Second, some rewriting is performed on the formula associated with each sequence (e.g. modularization) in order to facilitate their treatment.
- Third, minimal cutsets of each sequence (or group of sequences) are determined. Classical algorithms to compute the minimal cutsets work either top-down (e.g. [FV72, Rau03]) or bottom-up (e.g. [JK98,JHH04]).

- Fourth, probabilities/frequencies of sequences are assessed from the cutsets. More generally, cutsets are used to get various measures of interest such as importance factors of components, sensitivity to variations in basic event probabilities, …

In this process, three kinds of approximations are used:

- Sequences, including success branches, are quantified by means of minimal cutsets (which, by definition, do not embed negations).
- Truncation is applied to limit the process, and therefore reduce the possibility of combinatorial explosion.
- Probabilities are evaluated using one of two first order approximations: the rare event approximation or min-cut upper bound.

In the remainder of this section, we shall discuss the consequences of these three kinds of approximations.

3.2 The rare event approximation

Let us assume, for a while, that minimal cutsets represent exactly the sequence. The rare events approximation is used to assess the probability of the sequence. Namely, for a sequence *S* (or more exactly the Boolean formula *S* that represents the sequence), the probability of *S* is assessed as follows.

$$p(S) \quad \approx \quad \sum_{\pi \in MCS[S]} p(\pi) \tag{1}$$

The rare event approximation is actually the first term of the Sylvester-Poincaré development to compute the probability of a union of events:

$$p(E_1 \cup \ldots \cup E_k) \quad = \quad \sum_{1 \leq i \pi k} p(E_i) - \sum_{1 \leq i < j \leq k} p(E_i \cap E_j) + \ldots + 1^{-k} p(E_1 \cap \ldots \cap E_n) \tag{2}$$

The rare event approximation gives an upper bound of the probability, it is therefore conservative. By computing the second term of the development, one gets a lower bound of the probability (these two values constitute the first pair of so-called Boole-Bonferroni bounds):

$$\sum_{1\le i\pi k} p(E_i) - \sum_{1\le i<j\le k} p(E_i \cap E_j) \quad \le \quad p(E_1 \cup \ldots \cup E_k) \quad \le \quad \sum_{1\le i\pi k} p(E_i) \qquad (3)$$

When the number of cutsets is large, the computation of more terms is intractable. The rare event approximation gives accurate results when the probabilities of basic events are low. In the presence of relatively high probabilities (say >10^{-2}) and/or many minimal cutsets, the approximation is no longer valid. Consider for instance a *3*-out-of-*6* system *S*, with *p(e)=0.1* for each basic event *e*. The exact probability of *S* is *0.01585*. The Boole-Bonferroni bounds given by equation (3) are respectively *0.01009* and *0.02*, a rather rough approximation in both cases. The min-cut upper bound approximation is also no longer valid when relatively high probabilities are present, either from negation or embedding alignment frequencies in the fault trees.

3.3 Truncation in minimal cutsets determination

In general, sequences of large event trees admit huge numbers of minimal cutsets. Therefore, only a subset of the latter's can be considered (the most important ones, in terms of probability, one expects). Algorithms to compute minimal cutsets apply truncation to keep only few thousands cutsets (beyond computations are intractable).

The choice of the right truncation value is a result of trade-offs between accuracy of the computation and resource (time and memory) consumption. Expert knowledge about the expected probability of the sequence plays also an important role in that choice.

It remains that, by applying truncation, one gets an optimistic approximation. Moreover, there is no way to ensure that this approximation is accurate. For instance, if we keep a thousand cutsets of probability 10^{-9} and by the way we ignore a million cutsets of order 10^{-11}, then we underestimate the risk by a factor 10. This problem is largely ignored by most of the practitioners.

3.4 Quantification of success branches

But the main problem in the classical approach stands in the way success branches are (badly or even not at all) taken into account. None of the classical algorithms are actually able to deal with negations, for two main reasons. First, by definition, minimal cutsets do not contain negative literals. Therefore, the functions they encode are coherent. The notion of minimal solutions of general (coherent or non-coherent) functions exists (this is the notion of prime implicants), but that's another (very different) story. Second, truncation and minimality tests and reduction rules used by classical algorithms are not compatible with negations. The interested reader should see [Rau01] for a detailed discussion on that topics, including theoretical computational complexity arguments.

Event trees assessment tools take into account success branches in various ways. Let $S = F_1 \ldots F_m . \overline{G_1} \ldots \overline{G_n}$ be a sequence.

The first approximation consists in ignoring success branches. Pure and simple.

$$p(S) \approx RE_1(S) = \sum_{\pi \in MCS[F_1 \ldots F_m]} p(\pi) \tag{4}$$

The second approximation consists in correcting the approximation (4) by introducing a negative factor. E.g.

$$p(S) \approx RE_2(S) = \sum_{\pi \in MCS[F_1 \ldots F_m]} p(\pi) \times \left[1 - \sum_{\pi \in MCS[G_1 + \ldots + G_m]} p(\pi) \right] \tag{5}$$

The second term of approximation (5) is sometimes replaced by the product of the *[1-$RE_1(G_i)$]*'s, which is certainly not better.

The third and more serious approximation consists in removing (known as delete terming) from *MCS[$F_1 \ldots F_n$]* the cutsets π such that π_S^c satisfies $G_1 + \ldots + G_n$. It can be shown that, provided the *Fi*'s and the *G_j*'s are coherent (in other words, have no negation within the functions), this operation gives actually the minimal cutsets of *S*.

$$p(S) \approx RE_3(S) = \sum_{\pi \in MCS[S]} p(\pi) \tag{6}$$

Some authors propose process success branches as follows. First, negations are pushed down toward variables, using de Morgan's Laws. Second, new variables are introduced to represent negative literals. Third, minimal cutsets of the rewritten formula are computed. Finally, those that contain both a variable and its (encoded) negation are eliminated. This attempt is interesting. However, it cannot work correctly because of truncation. Consider for instance the formula *F=not-a.not-b.(c+d)*. If we apply truncation to eliminate cutsets whose order is greater than 2, then we get no cutset at all for *F*, which is indeed incorrect. Another problem is exemplified by the formula $F = \overline{G}.H$ where $G=(a_1.b_1)+\ldots+(a_n.b_n)$ and *H* is any formula that possibly depends on the a_i's and the b_j's. $not\text{-}G = (not\text{-}a_1 + not\text{-}b_1) \ldots (not\text{-}a_n + not\text{-}b_n)$ has 2^n cutsets, which indeed explodes the solution space, whatever the cutsets of *F* may be. For these reasons, we shall not consider this idea here, which is far too dangerous.

Approximations $RE_1(S)$, $RE_2(S)$ and $RE_3(S)$ may also give incorrect results as exemplified by the following example.

$$\begin{aligned} S &= F.\overline{G} \\ F &= \left[(a_1+b_1).(a_2+b_2).(a_3+b_3)\right] + c_1.c_2.c_3 \\ G &= (a_1+a_2+a_3) + (d_1+d_2+d_3+d_4+d_5) \end{aligned}$$

with $p(a_i)=p(b_j)=p(d_k)=0.1$ and $p(c_l)=0.001$. It is easy to verify that $p(S)=0.00043$ $RE_1(S)=0.0069$, $RE_2(S)=0.0039$, $RE_3(S)=0.001$. While truncation is not a problem for this example, all three approximations, even the best one, overestimate the true sequence probability by more than a factor of 2.

This example is indeed somewhat artificial. However, the same kind of problems do occur in real-life PSA, as we shall see.

4 The BDD approach to assess event trees

If a man does not keep pace with his companions, perhaps it is because he hears a different drummer. Let him step to the music which he hears, however measured or far away.

Henry David Thoreau, WALDEN

Bryant's Binary Decision Diagrams [Bry86], BDD for short, are now a well-known and widely used technique [Bry92]. In this section, we recall briefly the basics of this technique and we discuss its use to assess event trees (J. Andrews initiated this work in [AD00]).

4.1 Binary Decision Diagrams

The Binary Decision Diagram of a formula is a compact encoding of the truth table of this formula. From a BDD, it is possible to perform efficiently all of the probabilistic quantifications (top event probability, importance factors,…). The BDD representation is based on the Shannon decomposition: Let *F* be a Boolean formula that depends on the variable *v*, then

$$F = v.F[v \leftarrow 1] + \overline{v}.F[v \leftarrow 0] \qquad (7)$$

By choosing a total order over the variables and applying recursively the Shannon decomposition, the truth table of any formula can be graphically represented as a binary tree. The nodes are labelled with variables and have two outedges (a *then*-outedge, pointing to the node that encodes *F[v←1]*, and an *else*-outedge, pointing to the node that encodes *F[v←0]*). The leaves are labelled with either 0 or 1. The value of the formula for a given variable assignment is obtained by descending along the corresponding branch of the tree. The Shannon tree for the formula $F = ab + \overline{a}c$ and the lexicographic order is pictured Fig. 2 (dashed lines represent *else*-outedges).

Indeed such a representation is very space consuming. It is however possible to shrink it by means of the following two reduction rules.

- Isomorphic subtrees merging. Since two isomorphic subtrees encode the same formula, at least one is useless.
- Useless nodes deletion. A node with two equal sons is useless since it is equivalent to its son ($F = v.F + \overline{v}.F$).

By applying these two rules as far as possible, one get the BDD associated with the formula. A BDD is therefore a directed acyclic graph. It is unique, up to an isomorphism. This process is illustrated on Fig. 2.

Logical operations (and, or not) can be performed directly on BDD. In this way, the Shannon tree is never built then shrunk. The BDD of a formula is obtained by composing the BDD of its subformulae. An efficient implementation of a BDD package is described in reference [BRB90].

4.2 Application to Fault Trees/Event Trees assessment

Thanks to the Shannon decomposition, the probability of a formula *F* can be computed efficiently from the BDD that encodes *F* (and the probabilities of basic events). The following equality holds.

$$p(v.F_1 + \overline{v}.F_0) \quad = \quad p(v).p(F_1) + [1-p(v)].p(F_0) \qquad (8)$$

It is easy to derive a recursive algorithm from equality (8) [Rau93]. This algorithm is linear in the size of the BDD and gives exact results. It needs no truncations and makes no approximation. Importance factors can also be computed efficiently and exactly from BDD [DR00].
By slightly modifying the semantics of nodes, BDD can also be used to compute and to encode minimal cutsets (see [Rau93, Rau01]). BDDs that encode minimal cutsets are called ZBDD, from the name given by in its Minato's seminal article [Min93]. Truncation can be applied to keep only the most relevant cutsets.

Hence, the BDD approach to assess event trees works as follows.

- First, sequences are compiled as explained above.
- Second, some rewriting is performed on the formula associated with each sequence in order to facilitate their treatment and to select a good variable ordering. We shall discuss this very important issue in the next section.
- Third, the BDD that encode the sequence is computed.
- Fourth, the exact value of the probability (or the frequency) of the sequence is computed from the BDD. More generally, importance factors of components, as well as sensitivity to variations in basic event probabilities are assessed from the BDDs in a exact way.

As a fourth or fifth step and for the sake of the verification of the model, minimal cutsets can be extracted. However, this is not necessary. Moreover, since minimal cutsets are used only for verification purposes, one need only consider very few of them. In fact, for an analyst to consider more than a few hundred cutsets may be cognitively infeasible.

4.3 Pre-processing and variable ordering

It is widely known, since the very first uses of BDDs [Bry86], that the chosen variable ordering has a strong impact on the size of BDDs, and therefore on the efficiency of the whole methodology. The way formulae are written may also influence strongly the sizes of intermediate BDDs [MJF99]. With very large models, such as sequences of nuclear PSA, this issue must be carefully addressed in order to avoid the exponential explosion of the BDD size.

Finding the best ordering (or even a reasonably good one) is a very hard problem (namely, it is NP-complete [BW96]). Two kinds of heuristics are used to determine which variable ordering to apply. Static heuristics are based on topological considerations and select the variable ordering once for all (e.g. [FFK88, MIY90, BRKM91]). Dynamic heuristics change the variable ordering at some points during the computation. They are thus more versatile than the former, but the price to pay is a serious increase of running times. Sifting is the most widely used dynamic heuristics [Rud93]. Event tree sequences involve in general far too many variables to make dynamic reordering feasible.

For the purpose of this study, we designed a rewriting strategy made of five different heuristics applied in a row. The precise description of this strategy will be the subject of a forthcoming article. Basic operations are the following:

- Gate coalescing (at the top of the formula).
- Gate decomposition, according to [MJF96].
- Constants (house events, boundary conditions, alignment impacts, etc.) and subsumed occurrences (e.g. $v+v.F+G = v+G$) propagation.
- Domain specific information (common cause groupings, mutually exclusive computations).
- Gate fan-ins reordering, according to [FFK88, MIY90].
- Depth-First Left Most variable ordering.

This strategy made it possible to handle all the sequences within reasonable times and memory consumptions. Without it, some of the sequences were intractable. The Aralia [Arb00] computation engine, designed by one of the authors, embeds many heuristics and makes it possible to program strategies. The design of good strategies for event tree processing is a major part of the LukeTreeWalker project [ER04].

5 A Case Study

Mais cher Woody, on doit goûter le vin délicatement, si on l'apprécier à sa juste valeur.
But my dear Woody, one must place wine gently in one's mouth, if one wishes to make an informed judgement.
--- CoCo to Woody, at Maison d'H

5.1 The Model

The basis of this study is an actual event tree coming from the nuclear industry. This event tree is made of 181 sequences, with a total of 2259 basic events. Broken down by sequence, the smallest is made of 78 gates and 158 basic events. The largest one is made of 1128 gates and 1745 basic events. The mean numbers of gates and basic events are respectively 857 and 1455 per sequence. Among the 181 sequences, 171 lead to core damage. Table 3 gives more details about the distribution of sequences according to their sizes.
Ten fault trees, representing the top events of the event trees, are used to build the sequences (plus 8 individual events). The smallest of these fault trees is made 74 gates and 155 basic events. The biggest one is made of 561 gates and 946 basic events. The mean numbers of gates and basic events are respectively 277 and 490.

5.2 Efficiency of the BDD approach

For each sequence, we computed (with the strategy discussed in section 4.3) the following data structures and quantities.

- The formula rewritten by the strategy.
- The BDD that encodes this formula.
- The probability of the sequence computed from the BDD.

- The minimal cutsets of the sequence. However, it is not possible to compute all of the minimal cutsets (for most of the sequences there are more than 10^9 of them). The BDD approach can efficiently count the minimal cutsets, and the prime implicants, even though none need be listed.

To limit the number of minimal cutsets generated, we used a probability truncation limit defined thusly:

$$\textit{(Cutset Value)} >= \textit{(BDD Value of the Sequence)} * 10^{-4}$$

So if the probability of a sequence is 10^{-9}, as calculated by BDD, we limited the cutsets to those whose probabilities are greater than 10^{-13}.

It is worth noting that we used BDDs to calculate the exact results as well as to create ZBDDs, a data structure from which we can extract the cutsets [Min93]. The cutsets we obtained are the same as those that would have been obtained with a classical bottom-up or a top-algorithm. However, to extract them from the ZBDD is much faster.

6 Comparison between the classical and the BDD approaches

[T]hese discoveries clearly confute the Ptolemaic system, and they agree admirably with this other position and confirm it.

--- Galileo, in a letter to the Grand Duchess Christina of Lorraine

6.1 Experimental protocol

In this section, we compare the probabilities of sequences we obtained with the BDD approach with those that would have been obtained with a classical approach. The questions we aim to answer are the following.

Question 1: In the classical approach, is it necessary to compute more than one term of the Sylvester-Poincaré development?

Question 2: Is the approximation RE_1 accurate (recall that RE_1 consists in ignoring success branches)?

Question 3: Is the approximation RE_2 accurate (recall that RE_2 consists in correcting RE_1 by multiplying it with the probability of success branches)?

Question 4: Is the approximation RE_3 accurate (recall that RE_3 consists in computing the probability of the sequence from its minimal cutsets and employing "delete term").

In order to answer questions 1 to 4, we computed, for each sequence, the following quantities:

- The exact probability of sequence, computed from the BDD.
- The first term of the Sylvester-Poincaré development (RE_3) computed from the minimal cutsets of the sequence (recall that we keep only cutsets whose contribution is at least 10^{-4} times the probability of the sequence as calculated by BDD).
- The difference between the first and the second terms of the Sylvester-Poincaré development, still computed from the cutsets.
- The exact probability of the conjunction of the failure branches of the sequence computed from the BDD that encodes this conjunction. It is worth noticing that this approximation should be better than RE_1 as we defined it section 2. However, for the sake of the simplicity, we shall denote it RE_1.
- The exact probability of the conjunction of the failure branches times one minus the exact probability of the disjunction of the success branches (both obtained by the BDDs that encode them). For the same reason as previously, this quantity should be a better approximation than RE_2 but we shall denote it RE_2.

Except the first sequence, that contains only success branches and whose probability is 0.999817, probabilities of sequences range rather log-uniformly from $4.99\ 10^{-5}$ to $2.87\ 10^{-18}$. It would be an error to concentrate on most probable sequences for each sequence corresponds to a different situation. The less frequent sequences may also be those with the most severe consequences for the environment. Table 5 gives a distribution of the sequences according to their probabilities (this distribution is a bit arbitrary for the reasons we just gave).

6.2 Analysis of the results

Question 1: the question 1 is easy to answer. the range given by the two first terms of the Sylvester-Poincaré development is narrow for all of the sequences. The relative difference second-term / first-term never exceeds 6%. This means that the rare event approximation is accurate, at least for what concerns the quantity it assesses.

Question 2: To answer this question, we compute the relative difference $[RE_1(S)-p(S)]/p(S)$ which represents the relative error one makes by considering RE_1 (note that RE_1 is always bigger than the exact probability). Table 6 gives the distribution of the sequences according to the values of this ratio.
For only one sixth of the sequences RE_1 is pessimistic by a factor less than 2! For half of the sequences, RE_1 is pessimistic by at least two orders of magnitude! Sequence number 13 has the "gold medal" with a relative error of 6.53 10^{7} for a probability 1.72 10^{-12}.

Question 3: Table 7 gives the distribution of the sequences according to the relative error made by the approximation RE_2.
RE_2 corrects a bit RE_1. However, it gives very pessimistic results for two thirds of the sequences and is still pessimistic by two orders of magnitude for half of the sequences. Sequence number 13 has again the "gold medal" with a relative error of 3.98 10^{6}.

Question 4: The answer to this question is a bit more complex for RE_3 gives sometimes optimistic results. However, the greater the number of minimal cutsets considered, the less the expectation to be optimistic. So, on the one hand, one may argue that we didn't take into account enough cutsets. On the other hand, the truncation has to be put somewhere in order to avoid prohibitively long run times. By setting it to 10^{-4} the probability of the sequence, we adopted a quite conservative attitude. If we had used the min-cut upper bound approximation, the results would have been even more optimistic.

Table 8 gives the distribution of sequences according to the relative error made by RE_3, when RE_3 is optimistic.

RE_3 is thus optimistic in more than a quarter of the sequences. It is optimistic by a factor 2 in at least one sequence whose probability is around 10^{-9} and by a factor 4 for at least one sequence whose probability is around 10^{-12}.

Table 9 gives the distribution of sequences according to the relative error made by RE_3, when RE_3 is pessimistic.
RE_3 is thus pessimistic by a factor *2* or more for 104 sequences among 181 and by a factor 10 or more for one third of the sequences. For instance, it is pessimistic by a factor 14 for the sequence number whose exact probability 7.42 10^{-6}.

In order to confirm these results, we calculated the cutsets whose fractional contributions to the probability of the sequence is greater than 10^{-6} (rather than 10^{-4}). Indeed, running times and numbers of cutsets increase quite a lot (running times are up to 20 minutes and for some of the sequences up to 200,000 cutsets show up). With such a low truncation, RE_3 is pessimistic on all but 7 sequences. On the latter sequences, it is optimistic by at most 7%, which is quite acceptable. Table 10 gives the distribution of sequences according to the relative error made by RE_3, when RE_3 is pessimistic. It is worth noticing that RE_3 is now pessimistic by a factor greater than 10 for more than a third of the sequences and by a factor greater than 80 for 18 of them.

Last, but not least: the sum of the probabilities of core damage sequences computed with the BDD approach is 2.27 10^{-5}. With the classical approach we obtain, for both cutoffs (10^{-4} times the probabilty of the sequences and 10^{-6} times the probability of the sequences), 1.29 10^{-4}. It has been pointed out by M. Barrett that just summing the probabilities obtained for each sequence may be incorrect because some cutsets may be duplicated (or even subsumed). With the BDD approach this problem does not exist since the sequences are exclusive to one another by construction. In order to confirm our observations, we performed the following experiment. For different absolute cut-offs, we computed the cutsets for each core-damage sequence, we collected all of these cutsets together, we removed those duplicated and subsumed, and then we computed the probability of a core damage from the resulting set. Table 11 gives the results obtained for absolute cut-offs of 10^{-10}, 10^{-11}, 10^{-12} and 10^{-13} (i.e. we kept only those cutsets whose probabilities are greater than the above cut-offs).

For all of these cut-offs, the probability of a core damage is 1.21 10^{-4}. Therefore, no matter which way it is applied, the classical approach overestimates by almost a factor 5 the likehood of core damage. Recall that even in the small example at the end of Section 3.2, all approximations overstated results by a factor of 2.

7 Importance Factors

Now I'm gettin' mad,so I'm gonna ask you again: Who's on First?

--- Lou Costello

Probabilities (or frequencies) of sequences are interesting *per se*. It is however also very important to rank basic events according to their contributions to the risk. This is in general done through the assessment of so-called importance factors. In this section, we present results for three importance factors that are widely used in the industry: the Critical Importance Factor (also called the Fussel-Vesely Importance), the Risk Achievement Worth and the Risk Reduction Worth.

7.1 Definitions

The critical importance factor of a basic event *e* (for a sequence *S*), denoted by *CIF(S,e)*, is defined as follows.

$$CIF(S,e) \quad = \quad \frac{p(e)}{p(S)} \times \frac{\partial(p(S))}{\partial(p(e))} \tag{9}$$

The risk achievement worth, denoted by *RAW(S,e)*, is defined as follows.

$$RAW(S,e) \quad = \quad \frac{p(S \mid e)}{p(S)} \tag{10}$$

Finally, the risk reduction worth, denoted by *RRW(S,e)*, is defined as follows.

$$RRW(S,e) \quad = \quad \frac{p(S)}{p(S \mid \overline{e})} \tag{11}$$

Definitions (9) to (11) can be assessed either from cutsets or from BDDs [DR00].

7.2 Results

We present below the results we obtained for sequence number 4. Results for the other sequences are similar. This sequence is made of 1744 basic events and 1128 gates. It has 447 minimal cutsets whose probability is greater that 10^{-4} its probability, namely 2.23632 10^{-9}. The rare event approximation applied on these cutsets gives 1.19841 10^{-9}.

Tables 12, 13 and 14 give respectively the most important 20 basic events according to respectively their CIF, RAW and RRW (computed from the BDD). These tables show that not only the values computed from BDD and cutsets are different, but also the induced ranking is different. Table 11 shows many basic events whose RAW is 1. This just means that these basic events don’t show up in the truncated list of cutsets. It is well known however that the RAW must be considered with care [WW96]. The BDD approach shows just how much care is required to achieve accurate RAW values.

The difference in importance calculated from cutsets and BDDs is extraordinarily important. To undertake risk informed maintenance, risk informed asset management, and outage/shutdown planning, the relative importance of components must be known with a high degree of accuracy to insure the safety of the populace, the safety of the environment, and the proper allocation of funds to achieve the former goals.

Moreover, to calculate the importance of components, one usually combines the importance measures of the basic events which refer to these components (remember that basic events are propositions about states of components, “Pump A fails to run;” , and not things in, and of, themselves, such as the object, Pump A). Importance measures of components cannot be combined linearly from importance measures of cutsets [BA01]. BDD, however, does not suffer from this limitation, and we can combine the importance measures of basic events calculated from the BDD structure to give an exact importance measure of the components.

8 Runtime, Space, and Complexity

Faire de la bonne cuisine demande un certain temps. Si on vous fait attendre, c’est pour mieux vous servir, et vous plaire.

Good cooking takes time. If you are made to wait, it is to better serve you, and to please you.

--- Menu of Restaurant Antoine, New Orleans

This section describes the maximum and average running times for each category of sequences, details of which are in table 4. These results are obtained by accumulating the running times to rewrite the formula, to build the BDD, to compute the probability from the BDD and to build the ZBDD that encodes the minimal cutsets (with a 10^{-4} relative cut-off). They were observed on a laptop computer running Windows 2000 with a processor speed of 1.8 Ghz and with a 1 gigabyte of RAM. Table 4 shows that it takes at most 156.1 seconds to handle a sequence. All but 2 sequences are treated in less than 75 seconds. On average it takes 16.03 seconds to fully quantify a sequence. These running times can surely be improved by using larger hashtables, a faster computer, or by improving the strategy. More importantly, the resulting BDD can be cached for subsequent quantification at a later time with different variable probabilities. It takes at most 4.27 seconds to compute the probability of a sequence from its BDD (and 0.81 seconds on average). The running time to assess a probability from a BDD doesn't depend on the probabilities of basic events.

It takes at most 4,86 millions nodes to build the BDD and the ZBDD that encodes the cutsets. On average these computations require 1,28 millions nodes. It other words, given the size of the hashtables, the most difficult sequence is handled within around 200 megabytes.

The past 20 years have seen the movement of crucial engineering programs from mainframe and mini-computers to personal computers, and dramatic increases in computation speed and memory limits. We all remember the PC-AT in 1986 which increased the clock speed of the original PC from 4.77 Mhz to 6 Mhz, while now off-the-shelf hardware can run at 2.6 Ghz.

While all of this is well and good, it has also given us unrealistic notions of what runtimes and memory requirements should be when solving NP-hard problems, such as the one described in this article. Certainly computing from BDDs will take longer and need more space than simple bottom-up algorithms with truncation. But when solving calculations to help us understand the risk to and safety of populations and environments, the runtime difference between 5 seconds and 50 seconds can rationally be ignored when the extra 45 seconds will produce exactly correct answers.

Another interesting question is how one measures the complexity of an event tree made up of fault trees? Does one count the number of gates and basic events? Does one count the number of levels in the structure? How does one score the combination of operators so as to distinguish the difference in complexity between *F = -a+b+(c*d)* and *G = -(a*(b+c) xor d)*? Does one count the number of independent sub-trees, the number of branches for each gate, the number of negations? This is not simply an academic question. By understanding the complexity of the problem space, a set of heuristics can be chosen quickly, instead of randomly trying one after another.

In computer science, the analogous problem exists in measuring the “size” of a program. There are many putative measures being used: SLOC, McCabe’s Cyclomatic Complexity, NPATH, Halstead Software Science are examples of some standard metrics. The authors are investigating such metrics, which will be detailed in a forthcoming technical report.

9 Conclusion

There is no single development, in either technology or management technique, which by itself promises even one-order of magnitude improvement within a decade in productivity, in reliability, in simplicity.

--- Fredrick P. Brooks, Jr., NO SILVER BULLET

In this article, we have reported the results of a comparative study of two technologies to assess risk models: the classical approach, widely used and trusted, based on minimal cutsets and the BDD approach, improved by means of heuristics, that in making no approximations, gives exact results. The study is based on an actual linked-fault tree model representing an event tree coming from the nuclear industry. We used the Aralia computation engine which implements both approaches as well as many heuristics and formula rewriting strategies.

Indeed, definitive conclusions cannot be drawn from a single example. However, our test case is sufficiently large and representative and the results are sufficiently clear to make the following observations.

- The approximation that consists in taking into account failure branches only should be avoided. Our experiments show that, even corrected by a factor obtained from success branches, this approximation overestimates, very often by orders of magnitude, the probability of the sequence.
- The assessment of the probabilities of the sequences through the minimal cutsets should be considered with care. In a significant number of cases, this approximation gives optimistic results, because of truncations. Such an underestimation of the risk is not acceptable. Moreover, the same truncation that gives an optimistic result in one sub-system may give a very pessimistic result in another sub-system, within the same top event! With a truncation set to 10^{-4} times the probability of the sequence, we observed, among the 181 sequences of our test case, results that are optimistic by a factor 4 together with results that are pessimistic by a factor 96.
- Such variations make the ranking of sequences according to their contributions to the overall risk delicate, if not dubious.
- The classical approach overestimates the likehood of a core damage by almost a factor 5.
- Because of imprecision on the values of probabilities (when computed from the cutsets), the rankings of basic events induced by importance factors should be considered with care. This remark is especially important for the so-called risk achievement worth that can miss important basic events.

The above observations do not mean that existing PRA studies based on event trees must be discarded. Nevertheless, they are a stone in the garden of the classical approach based on minimal cutsets. On the other hand, we don't claim that Binary Decision Diagrams are the universal panacea. This technique still suffers from the exponential explosion of memory requirements. We have shown that heuristics can be designed which improve dramatically its efficiency. They are however hard to tune. More experiments, more efforts are necessary to make our approach able to deal with all the existing models.

One final note: as Fred Brooks' quotation states at the beginning of this section, we should expect no silver bullet to slay the werewolf of complex computations. What is important is that any solution is (1) productive, that it allows us to work orderly and rationally, (2) reliable, that it gives us the correct answers with explicit knowledge of the error bounds, and (3) simple, that it allows us to confirm the problem we are solving and its solution. This study is a step towards this goal.

10 Bibliography

[AD00] J.D. Andrews and S.J. Dunnett, Event Tree Analysis using Binary Decision Diagrams, *IEEE Transactions on Reliability*, Vol 49, No 2, June 2000, pp230-239.

[ARb00] Arboost Technologies. *ARALIA 4.2e Users' Manual*, February, 2002.

[BW96] B. Bollig and I. Wegener. Improving the Variable Ordering of OBDDs is NP-Complete. *IEEE Trans. on Software Engineering*, 45(9):993–1001, Sep. 1996.

[BA01] E. Borgonovo and G.E. Apostolakis, A new importance measure for risk-informed decision making *Reliability Engineering and System Safety*, Volume 72, Issue 2 , pp 193-212, May 2001.

[Bry86] R. Bryant. Graph Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, 35(8):677–691, August 1986.

[BRB90] K. Brace, R. Rudell, and R. Bryant. Efficient Implementation of a BDD Package. In *Proceedings of the 27th ACM/IEEE Design Automation Conference*, pages 40–45. IEEE 0738, 1990.

[Bry92] R. Bryant. Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams. *ACM Computing Surveys*, 24:293–318, September 1992.

[BRKM91] K.M. Butler, D.E. Ross, R. Kapur, and M.R. Mercer. Heuristics to Compute Variable Orderings for Efficient Manipulation of Ordered BDDs. In *Proceedings of the 28th Design Automation Conference, DAC'91*, June 1991.

[DR00] Y. Dutuit and A. Rauzy. Efficient Algorithms to Assess Components and Gates Importances in Fault Tree Analysis. *Reliability Engineering and System Safety*, 72(2):213–222, 2000.

[ER04] S. Epstein and A. Rauzy. LukeTreeWalker: Specifications. Technical Report TR2004-01. ARBoost Technologies.

[FFK88] M. Fujita, H. Fujisawa, and N. Kawato. Evaluation and Improvements of Boolean Comparison Method Based on Binary Decision Diagrams. In *Proceedings of IEEE International Conference on Computer Aided Design, ICCAD'88*, pages 2–5, 1988.

[FV72] J.B. Fussel and W.E. Vesely. A New Methodology for Obtaining Cut Sets for Fault Trees. *Trans. Am. Nucl. Soc.*, 15:262–263, June 1972.

[KH96] H. Kumamoto and E. J. Henley. *Probabilistic Risk Assessment and Management for Engineers and Scientists*. IEEE Press. 1996. ISBN 0-7803-6017-6.

[JK98] W.S. Jung and D.K. Kim. FORTE: a fast new algorithm for risk monitors and PSA. *Proceedings of the Fourth International Conference on Probabilistic Safety Assessment and Management*. New York, USA, p. 1221, 1998.

[JHH04] W. S. Jung, S. H. Han and J. Ha. A fast BDD algorithm for large coherent fault trees analysis. *Reliability Engineering and System Safety*. Vol. 83, pp 369-374, 2004

[MIY90] S. Minato, N. Ishiura, and S. Yajima. Shared Binary Decision Diagrams with Attributed Edges for Efficient Boolean Function Manipulation. In L.J.M Claesen, editor, *Proceedings of the 27th ACM/IEEE Design Automation Conference, DAC'90*, pages 52–57, June 1990.

[Min93] S. Minato. Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems. In *Proceedings of the 30th ACM/IEEE Design Automation Conference, DAC'93*, pages 272–277, 1993.

[MJF99] R. Murgai, J. Jain and M.Fujita, Efficient Scheduling Techniques for ROBDD Construction, *Proceedings of the International Conference on VLSI Design*, pp 394-401, 1999.

[Pap98] I.A. Papazoglou. Mathematical foundations of event trees. *Reliability Engineering and System Safety*, 61:169–183, 1998.

[Rau93] A. Rauzy. New Algorithms for Fault Trees Analysis. *Reliability Engineering & System Safety*, 05(59):203–211, 1993.

[Rau01] A. Rauzy. Mathematical Foundation of Minimal Cutsets. *IEEE Transactions on Reliability*, volume 50, number 4, pages 389-396, 2001.

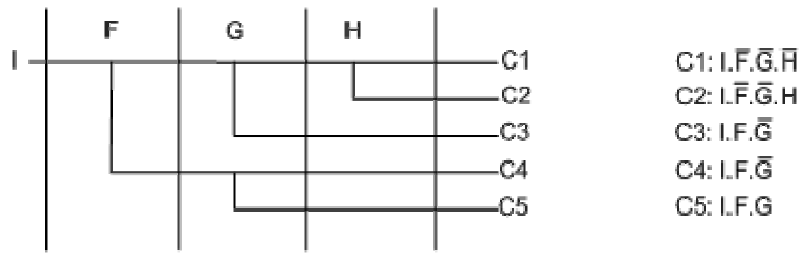
[Rau03] A. Rauzy. Towards an Efficient Implementation of Mocus. *IEEE Transactions on Reliability*, vol. 52:2, pp 175-180, 2003.

[Rud93] R. Rudell. Dynamic Variable Ordering for Ordered Binary Decision Diagrams. In *Proceedings of IEEE International Conference on Computer Aided Design, ICCAD'93*, pages 42–47, November 1993.

[WW96] I.B. Wall and D.H. Worledge. Some perspectives on risk importance measures. In *Proceedings of the international conference on Probabilistic Safety Assessment, PSA'96*, pages 203–207, 1996.

Acknowledgements:

The authors would like to thank K. Kawai, J. Sakaguchi, and K. Nishi from MHI for their help, D. Johnson and D. Wakefield from ABS Consulting, R. Cooke from TU Delft, K. Fleming from Technical Insights Inc., and M. Barrett from Duke Energy for careful reviews of this article and their support, and A. Soma from ABS Consulting Japan for her help.



F
G
H
I
C1
C2
C3
C4
C5
C1: $I.\overline{F}.\overline{G}.\overline{H}$
C2: $I.\overline{F}.\overline{G}.H$
C3: $I.F.\overline{G}$
C4: $I.F.\overline{G}$
C5: $I.F.G$

Fig. 1. An event tree and Boolean formulae associated with its sequences.

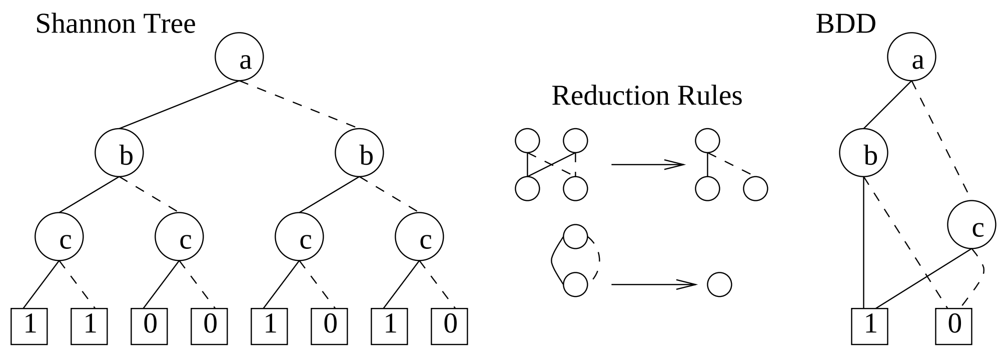


Figure 2. From the Shannon Tree to the BDD.

Number of BE	Number of gates	Number of sequences
from 158 to 161	from 78 to 80	7
256	142	1
769, 770	403,404	3
from 1422 to 1424	from 805 to 837	65
from 1546 to 1558	from 858 to 913	65
1621, 1622	from 1020 to 1052	30
1744, 1745	from 1097 to 1128	10

Table 3. Distribution of sequences by size

Number of BE	Number of sequences	max	mean
from 158 to 161	7	0.12	0.11
256	1	0.17	0.17
769, 770	3	0.95	0.94
from 1422 to 1424	65	24.23	7.40
from 1546 to 1558	65	124.91	19.62
1621, 1622	30	45.55	22.16
1744, 1745	10	156.1	46.47

Table 4. Running times in seconds (distribution by number of BE of sequences)

probability of the sequence	number of sequences
less that 10^{-12}	*100*
from 10^{-9} to 10^{-12}	*51*
from 10^{-5} to 10^{-9}	29

Table 5. Distribution of the sequences according to their probabilities

$RE_1(S)$-$p(S)$/$p(S)$	number of sequences
less than *1*	*28*
between *1* and *10*	*28*
between *10* and *100*	*42*
between *100* and *1000*	*39*
beyond *1000*	*51*

Table 6. Distribution of the sequences according to the relative error made by the approximation *RE_1*

$[RE_2(S)$-$p(S)]/p(S)$	number of sequences
less than *1*	*59*
between *1* and *10*	*38*
between *10* and *100*	*30*
between *100* and *1000*	*31*
beyond *1000*	*33*

Table 7. Distribution of the sequences according to the relative error made by the approximation *RE_2*

$p(S)$-$RE_3(S)/RE_3(S)$	number of sequences
less than *0.10*	*8*
between *0.10* and *0.50*	*19*
between *0.5* and *1*	*9*
between *1 and 3.10*	*15*
total	*53/181*

Table 8. Distribution of the sequences according to the relative error made by the approximation *RE_3* when it is optimistic (with a 10^{-4} relative cut off).

$RE_3(S)$-$p(S)/p(S)$	number of sequences
less than *0.10*	*15*
between *0.10* and *1*	*9*
between *1* and *10*	*44*
between *10 and 96.83*	*60*
total	*128/181*

Table 9. Distribution of the sequences according to the relative error made by the approximation RE_3 when it is pessimistic (with a 10^{-4} relative cut off).

$RE_3(S)$-$p(S)/p(S)$	number of sequences
less than *0.10*	*55*
between *0.10* and *1*	*22*
between *1* and *10*	*29*
between *10 and 78*	*50*
between *78* and *116*	*18*
total	*174/181*

Table 10. Distribution of the sequences according to the relative error made by the approximation RE_3 when it is pessimistic (with a 10^{-6} relative cut off).

cut off	10^{-10}	10^{-11}	10^{-12}	10^{-13}	*BDD (exact value)*
probability	$1.21\ 10^{-4}$	$1.21\ 10^{-4}$	$1.21\ 10^{-4}$	$1.21\ 10^{-4}$	$2.27\ 10^{-5}$
number of cutsets	*978*	*3535*	*13172*	*51493*	
number of BE	*180*	*367*	*600*	*785*	*2259*

Table 11. Probability of a core damage computed with different absolute cut offs.

Rank	CIF(S,e)/BDD	CIF(S,e)/cutsets
1	0.605467	0.81045
2	0.441697	0.615231
3	0.288971	0.172044
4	0.141314	0.135865
5	0.137265	0.223926
6	0.137265	0.223926
7	0.137265	0.223926
8	0.0622317	0.041397
9	0.0622317	0.041397
10	0.0622317	0.041397
11	0.0622284	0.041397
12	0.0622284	0.041397
13	0.0622284	0.041397
14	0.0573555	0.0511157
15	0.0573555	0.0511157
16	0.0522112	0.0355426
17	0.052187	0.0355426
18	0.0429735	0.0302301
19	0.0429735	0.0302301
20	0.0429735	0.0302301

Table 12. The 20 most important basic events according to their CIF for sequence 4 sorted by the second column.

Rank	RAW(S,e)/BDD	RAW(S,e)/cutsets
1	68803.5	92096.8
2	6770.64	1
3	6770.64	1
4	6770.64	1
5	6770.64	1
6	6770.64	1
7	4405.06	1
8	4405.06	1
9	2822.88	1
10	2822.88	1
11	827.259	795.395
12	827.259	288.623
13	827.259	288.625
14	827.259	1
15	827.259	265.348
16	827.259	265.348
17	827.259	265.348
18	762.086	1
19	762.086	1
20	390.821	636.928

Table 13. The 20 most important basic events according to their RAW for sequence 4 sorted by the second column

Rank	RRW(S,e)/BDD	RRW(S,E)/cutsets
1	2.53464	5.27566
2	1.79114	2.59896
3	1.40641	1.20779
4	1.16457	1.15723
5	1.15911	1.28854
6	1.15911	1.28854
7	1.15911	1.28854
8	1.06636	1.04318
9	1.06636	1.04318
10	1.06636	1.04318
11	1.06636	1.04318
12	1.06636	1.04318
13	1.06636	1.04318
14	1.06085	1.05387
15	1.06085	1.05387
16	1.05509	1.03685
17	1.05506	1.03685
18	1.0449	1.03117
19	1.0449	1.03117
20	1.0449	1.03117

Table 14. The 20 most important basic events according to their RRW for sequence 4 sorted by the second column