Systematic extraction of Minimal Cut Sequences from a BDMP model

Pierre-Yves Chaux, Jean-Marc Roussel, Jean-Jacques Lesage, Gilles Deleuze, Marc Bouissou

Systematic extraction of Minimal Cut Sequences from a BDMP model

Pierre-Yves Chaux[a,b*], Jean-Marc Roussel[a], Jean-Jacques Lesage[a], Gilles Deleuze[b], Marc Bouissou[b,c]
[a]LURPA - 61, av. du président Wilson 94235 Cachan cedex, France
[b]EDF R&D - 1, av. du Général de Gaulle 92140 Clamart, France
[c]Ecole Centrale Paris - Grande Voie des Vignes 92295 Chatenay-Malabry Cedex, France

Abstract: *The Boolean logic Driven Markov Processes (BDMPs) have been developed by EDF to conduct predictive risk modeling and assessment for critical systems. A BDMP is a dynamic model based on fault trees dedicated to repairable systems. The qualitative reliability analyses performed on this kind of model consist in computing the minimal scenarios that lead to the system failure which are known as Minimal Cut Sequences (MCS). This paper defines the concept of Minimal Cut sequences for BDMPs and gives a way to compute them using an equivalent finite automaton developed in our previous works.*

Keywords: *BDMP, Dynamic Fault Tree, Minimal Cut Sequence, Repairable System, Finite Automata*

1. INTRODUCTION

For many years, predictive risk modeling and assessment has been a challenge for companies which develop and exploit highly critical systems. Qualitative studies, as part of predictive risk assessment, are a way to qualify the system failure by finding specific failure scenarios or sets of components failures. This study is aiming at extending the current qualitative studies to systems composed of repairable components and thus describe the sequences of events (composed of both failure and repair events) that lead to the system failure.
Fault Trees based models are among the most popular models for risk analysis. For Dynamic Fault Trees (DFT) for example [1], different qualitative analyses techniques have already been proposed. In [2] Minimal Cut Sequences (MCS) are obtained by giving an order to the failure events in the Minimal Cut Sets, which are computed by using an "equivalent static fault tree" [3]. The limits of this approximation have been shown in [4]. In this work, MCS are defined as the minimal elements of the Cut sequences (CS) set (the combinations of failure events that are leading the system from its initial state to its failure [5]). A more rigorous approach is proposed in [6]. In this work, it is shown how any DFT can be formally translated by using a specific algebra. A canonical representation of the structure function of any DFD is also proposed. Afterwards, the set of all the MCS can be extracted from this canonical form, but the algorithmic complexity is large. All these works make the hypothesis that systems are non-reparable (only one occurrence of each components failure is taken into account). In our opinion, this hypothesis is too strong to cope with today industrial needs. It is the reason why EDF proposed Boolean logic driven Markov Processes (BDMP) models [7] for modeling repairable systems.
In a previous work [8] we proposed an algorithm for the systematic generation of an automaton, equivalent to a given BDMP, representing all the possible scenarios implicitly described in the BDMP model. Since this automaton gives a representation of all failure scenarios, including the MCS, we propose in this paper a method for extracting cut sequences and minimal cut sequences from this automaton. To describe and illustrate this extraction process, a middle sized example has been chosen; it is described in the next section of the paper.

2. SAFETY ANALYSIS OF REPAIRABLE SYSTEMS BY BDMP

2.1 A study case: coolant feeding system

In order to illustrate our approach, the system shown in Figure 1 has been chosen. Its main function is to feed another system with a cooling fluid from a source using two groups of pumps; one of these groups is powered by a heavily redundant power supply. This system is considered as repairable since all its components are repairable. The system fails when no fluid can be provided to the other system.
Six subsystems can be identified:

- The (A) subsystem, composed of an electric transformer, is used to provide low voltage electricity; if it fails, a second transformer is available thanks to a passive redundancy.
- The (B) subsystem is a distribution board powering a second distribution board using one of the two transformers.

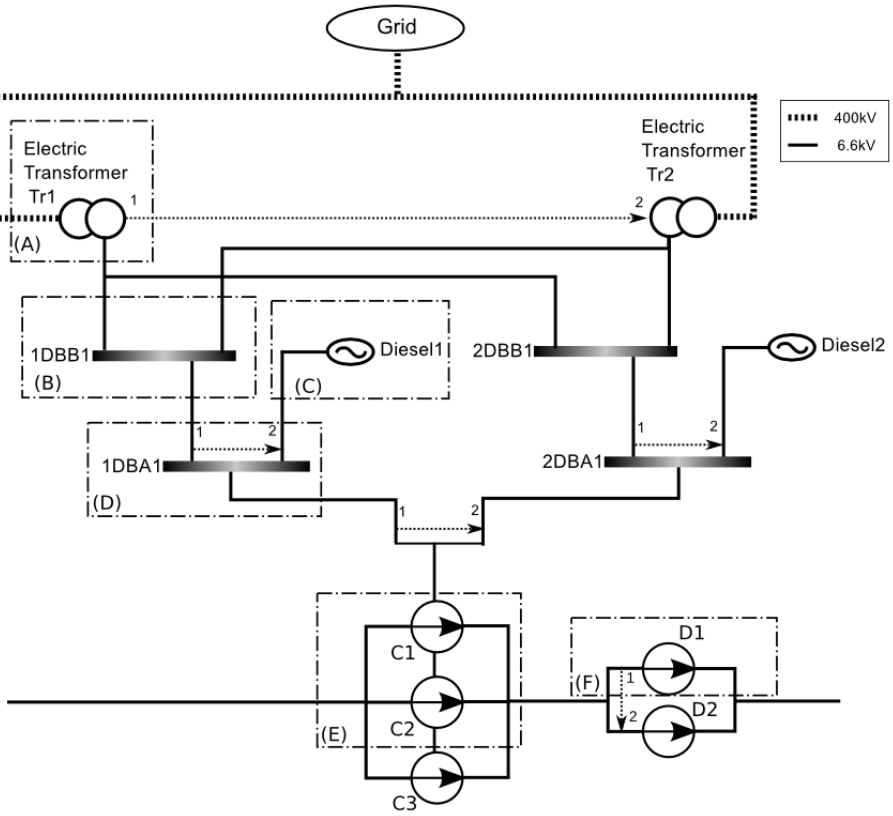

Figure 1: Technical implementation of the coolant feeding system

- The (D) subsystem is the distribution board powering the (E) subsystem, using one of the two possible powering sources (B or C).
- The (E) subsystem is a group of pumps powered by (D), only two pumps are used during operation, if one pump fail the third one is activated.
- The (F) subsystem is a group of pumps which pressurize the fluid; it is composed of one primary pump and one backup pump used for a stand-by redundancy

The subsystem composed of subsystems (B, C, and D) has a backup in order to provide a passive redundancy. Even though most components can only fail when they are active, both diesel generators and all pumps can fail when they are dormant. When a main subsystem fails, the redundant one is set in an active mode; as soon as the main one is repaired, the redundant one is put back in a dormant state.

2.2 BDMP model of the case study

The BDMP have been developed by EDF R&D [7] to model complex failure mechanisms while staying close to the graphical approach of Static Fault Trees. Replacing the Boolean basic components by dynamic ones, it grants the model a dynamic modeling perspective of the system. The main features of this formalism are the ability to model repairable basic components and stand-by redundancies between components, between subsystems and between subsystems and components.

A BDMP can be seen as a 4-tuple [8] where:

- At the top of the BDMP, the *top event* model the system failure;
- At the bottom of the BDMP, the *leaves* model basic components. Each component is modeled by a single leaf. If the component can only fail while in an active mode then it is modeled by a “F leaf”. If it can fail both in dormant and active mode, it is modeled by a “SF leaf”. Repair events of the components are also included in the model of the leaf;
- A coherent static *fault tree structure*, only composed by OR and AND gates, is modeling the combinatorial set of basic components failures that make the system fail;
- *Triggers*, denoted by a dotted arrow which are modeling the stand-by redundancies; if the subsystem (or component) at the origin of the trigger is faulty then the subsystem (or component) at its destination is put in an active mode. (This is a simplified description: refer to [7] for the full

specification of the effect of triggers). Used with the fault tree structure it allows the computation of the mode (dormant or active) of each leaf depending on the subsystem failures.

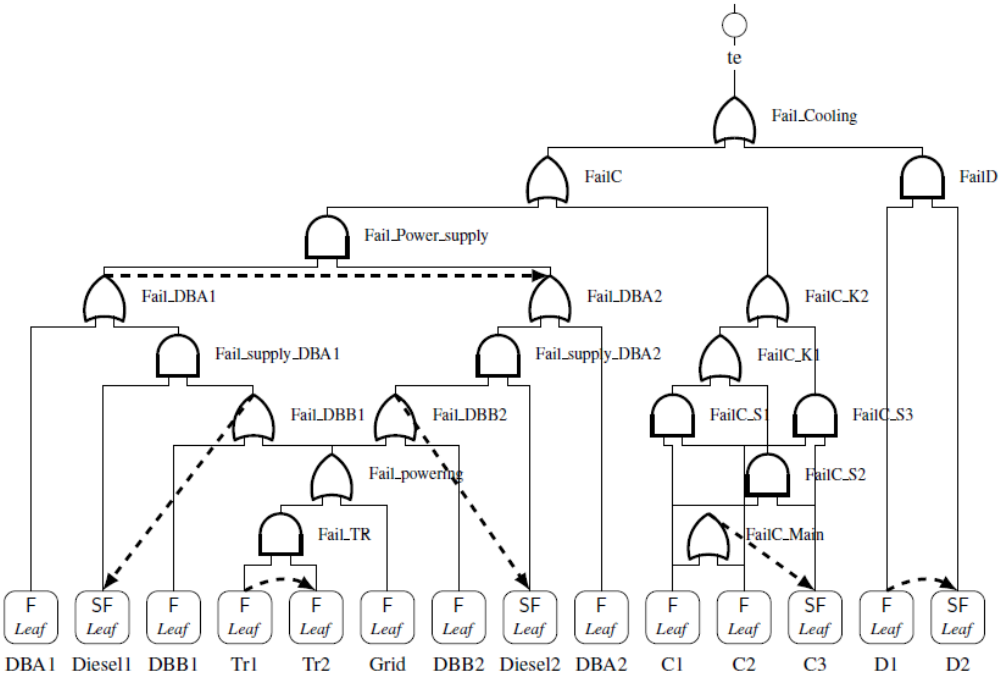

Figure 2 : BDMP model of the coolant feeding system

A BDMP model of the coolant feeding system is presented in Figure 2. The leaves DBA1 to DBA2 and the fault tree and trigger structures above them are modeling the failure of the subsystem composed of the electric supply of the pump group (C1,C2,C3); above the pumps (C1,C2, C3), the tree structure models the "2 out of 3" mechanism, using C3 as a backup.

Since the BDMP is a user friendly tool to model a system, its graphical form allows to implicitly describe a huge set of scenarios that can't be perceived directly from the BDMP; those scenarios have to be extracted and explicitly represented. That is the goal of the finite automaton described below.

2.3 Generation of the equivalent finite automaton (FA)

As the main objective of our work is to obtain the minimal length sequences that lead to the system failure, the first step is to be able to extract the scenarios from the BDMP. Two types of scenarios are of interest:

- All the possible scenarios of failure and repair of all the basic components; they will be represented by the sequences of the *dysfunctional language*,
- The scenarios that lead to the global failure; they will be represented by the sequences of the failure language. Failure sequences are the sequences of failure and repair events which end in a marked state (i.e. a state in which the top event is realized) of the finite automaton equivalent to the BDMP.

Both languages are regular, then it is possible to have an explicit representation using finite automata. The algorithms provided in [8] allow a systematic generation of the equivalent automaton of a BDMP. In the obtained automaton, each state represents a situation of the BDMP (one of the combinations of failure/functioning of each component). If this combination leads to the failure of the whole system then a marked state is used.

In short, the automaton equivalent to a BDMP is the 5-tuple $A = (\Sigma, Q, q_i, QM, \delta =< q_v, u, q_w >)$ where:

- Σ is the entry alphabet containing all failure events ($f_{c,m}$ where c is the component and m its mode – active/dormant) and repair events (r_c where c is the component).
- Q is the set of states, each one is representing a situation of the BDMP.
- q_i is the initial state, in this state all component are functioning.
- QM is the set of marked states, they represent the states where the system is faulty.

- $\delta = < q_v, u, q_w >$ is the transition function modeling the evolution between states when failure or repair events occur.

In the case of the coolant feeding system, the automaton equivalent to the BDMP of Figure 2 (automatically generated) is composed of 16384 states (exactly 2^{14} for this BDMP using 14 leaves) whose 13300 are marked and represent the global failure, and 220416 transitions which represent all the possible evolutions between states.

3. SYSTEMATIC EXTRACTION OF SEQUENCES FROM THE FINITE AUTOMATON

3.1 Dysfunctional and failure sequences

Since a BDMP models a system with repairable components, the FA which is an explicit representation of all the scenarios implicitly described by a BDMP is describing an infinite set of scenarios (or sequences) of length one to infinity. Finite sets of sequences of finite length, which are of specific interest for reliability or availability purposes, must now be extracted from the FA. For that, adapted procedures have been developed.

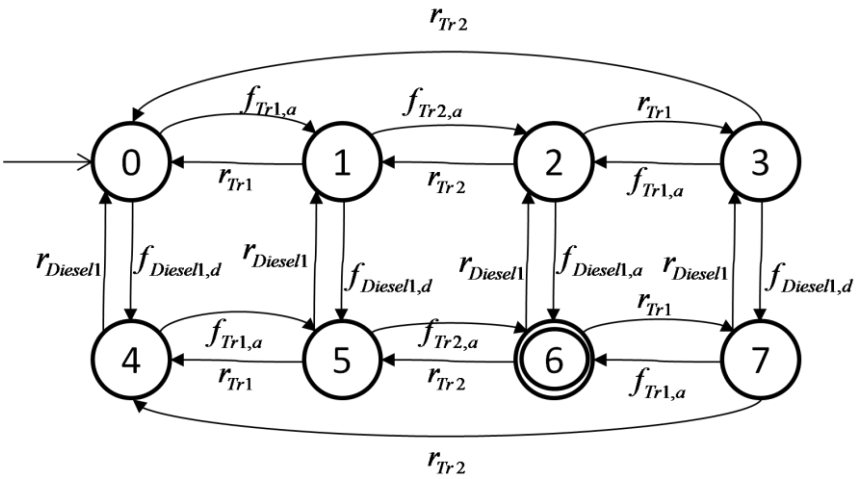

Figure 3: Part of the 'equivalent' automaton of the BDMP of Fig. 2

In order to illustrate the extraction process, a small part of the equivalent automaton of the coolant feeding system is used. This automaton, shown in Figure 3, represents all the possible scenarios of the BDMP if no component can fail, except the two electric transformers (Tr1,Tr2) and the diesel generator of the main subsystem (Diesel1). Taking into account these assumptions, the global system is faulty when Tr1,Tr2 and Diesel1 are faulty.
Different types of sequences can be extracted from this automaton:

- $f_{Tr1,a}$ and $f_{Tr1,a}\, f_{Tr2,a}$ are examples of *dysfunctional sequences*, only composed of failure events that don't lead to the system failure;
- $f_{Tr1,a}\, f_{Tr2,a}\, f_{Diesel1,a}$ is an example of a dysfunctional sequence which is a *failure sequence*, only composed of failure events it leads to the system failure;
- $f_{Tr1,a}\, f_{Tr2,a}\, f_{Diesel1,a} r_{Tr1}$ is an example of a dysfunctional sequence composed by both repair and failure events, it goes through system failure but does not *end* in system failure;
- $f_{Tr1,a}\, f_{Tr2,a}\, f_{Diesel1,a} r_{Tr1}\, f_{Tr1,a}$ is an example of a dysfunctional and a failure sequence, composed by both repair and failure events, it goes through a system failure and *end* in system failure.

For this reduced example only composed of 8 states, 22734 dysfunctional sequences of length 1 to 10 can be extracted; each of them represents a possible scenario of the BDMP. Among these 22734 dysfunctional sequences, 1630 are failure sequences, representing all the scenarios that lead to the system failure.

3.2 Cut Sequences

The goal of this study is to perform a qualitative reliability analysis of the system, to do so the scenarios that lead the system from its initial state to the global failure must be found. Since the goal of such qualitative analysis is only to look for the sequences that lead to the system failure and not to care about what might

happen after the failure, only the sequences that end in the *first failure* of the system will be extracted. If the goal was to perform an analysis of the availability of the system, this limitation must not be taken into account and all scenarios should be kept.

Finally, in the case of repairable systems we define the Cut Sequences (CS) as the sequences that belong to the failure language (a subset of the dysfunctional language) and that end in a marked state of the FA (a state of the system failure) but without passing through a marked state.

For the example of Figure 3, $f_{Tr1,a} f_{Tr2,a} f_{Diesel1,a} r_{Tr1} f_{Tr1,a}$ is a sequence that goes through the system failure (modeled by the state 6) and is ending in the system failure. It is not a Cut Sequence since the system failed but has been repaired. On the other hand, $f_{Tr1,a} f_{Tr2,a} f_{Diesel1,a}$ is a Cut Sequence because only the last reached state is representing the global failure.

In order to only extract CS from the FA in which the system failure is represented by the marked states, the extraction of the sequences need to comply with two rules:

- The sequences must reach a marked state.
- Once a marked state is reached by a sequence, the exploration is stopped and no longer sequences are generated extending the first one.

This process can be illustrated on the example of Figure 3: all CS must reach state 6 and once they have reached it, no longer CS can be obtained by extension. As a result, from the 1630 failure sequences of length 1 to 10, only 543 are Cut Sequences.

4. COMPUTING THE MINIMAL CUT SEQUENCES

4.1 Minimality criterion for minimal cut sequences

Once the Cut Sequences set is extracted, Minimal Cut Sequences (MCS) need to be defined and computed from the whole CS set. The goal is to keep from the CS set only the sequences of minimal length that do not contain any event that is not strictly necessary to lead to the system failure. For example, if we consider the two sequences: $f_{Tr1,a} f_{Tr2,a} f_{Diesel1,a}$ and $f_{Tr1,a} r_{Tr1} f_{Tr2,a} f_{Tr1,a} f_{Diesel1,a}$, both are Cut Sequences but the second one can't be minimal: it contains two events (r_{Tr1}, $f_{Tr1,a}$) that are not necessary to have the global failure. We can be sure of this because $f_{Tr1,a} f_{Tr2,a} f_{Diesel1,a}$ is included into $f_{Tr1,a} r_{Tr1} f_{Tr2,a} f_{Tr1,a} f_{Diesel1,a}$.

It is the reason why the chosen criterion for separating MCS from CS is the non-inclusion of any Cut Sequence into any Minimal cut sequences. A sequence to be is said included in a longer one if all events of this sequence are in the longer one in the same order.

To extract the MCS from the CS set, all CS will be compared with the smaller ones previously found. If a smaller sequence is included then the considered CS is not a MCS. If none is included then the considered CS is a MCS. On the reduced automaton of Figure 3, minimal cut sequences can be found:

- $f_{Tr1,a} f_{Tr2,a} f_{Diesel1,a}$ is one of the smallest Cut Sequences, since the minimum length of a sequence starting from the initial state and ending at state 6 is 3. This sequence is a MCS
- $f_{Tr1,a} f_{Tr2,a} r_{Tr1} f_{Tr1,a} f_{Diesel1,a}$ is a Cut Sequence in which the Cut Sequence $f_{Tr1,a} f_{Tr1,a} f_{Diesel1,a}$ is included then it is not a Minimal Cut Sequence
- $f_{Tr1,a} f_{Tr2,a} r_{Tr1} f_{Diesel1,d} f_{Tr1,a}$ is a Cut Sequence in which the Cut Sequence $f_{Tr1,a} f_{Tr2,a} f_{Diesel1,a}$ is not included ($f_{Diesel1,a}$ is not present in the longer sequence); since no smaller Cut Sequence is included, this sequence composed of repair and failure events is a Minimal Cut Sequence. This sequence is describing scenarios where after both transformers failed, only one is repaired and a fault occur on the diesel which is in a dormant mode since one of the transformers is operating. When the operating transformer fails again, the diesel cannot be used to supply the power, the system is then faulty. This kind of complex scenarios can't be found if the system is considered as non-repairable.

In order to limit the number of Cut sequences that will be compared with other ones, the global process can be decomposed into two sub-processes:

- One off-the-flight process which is comparing the Cut-Sequences from an extracted set.
- One on-the-flight process that will prevent the generation of non-minimal Cut-Sequences.

4.2 On-the-flight minimization process

The main goal of the on-the-flight minimization process is to prevent the extraction of non-minimal Cut Sequences. From a Minimal Cut Sequence, non-minimal Cut sequences can be generated. To illustrate this phenomenon, the Minimal Cut Sequence $f_{Tr1,a} f_{Tr2,a} f_{Diesel1,a}$, extracted from the FA of Figure 3 is used. This sequence is going through states 0, 1, 2 and 6. Using this MCS some non-minimal ones can be constructed. For example after the first event $f_{Tr1,a}$ the automaton is in the state 1, from this state there are many loops going back in that same state ($f_{Diesel1,d} r_{Diesel1}$, $f_{Diesel1,d} r_{Tr1} f_{Tr1,a} r_{Diesel1}$ …). A loop in the automaton is defined as a way to go from one state to the same state by a path of at least one event. All those sub-sequences can be included in $f_{Tr1,a} f_{Tr2,a} f_{Diesel1,a}$ after $f_{Tr1,a}$ to generate non-minimal Cut Sequences ($f_{Tr1,a} f_{Diesel1,d} r_{Diesel1} f_{Tr2,a} f_{Diesel1,a}$ and $f_{Tr1,a} f_{Diesel1,d} r_{Tr1} f_{Tr1,a} r_{Diesel1} f_{Tr2,a} f_{Diesel1,a}$). All these non-minimal cut sequences are generated by going through loops of the automaton.

Then, a way to prevent the generation of Non-Minimal Cut Sequence during extraction is not to go through loops of the automaton. This is done by keeping track of all reached state by a sequence during the extraction process. All sequences trying to go through one of the already reached states in this sequence is not generated.

4.2 Systematic extraction of Minimal Cut Sequences

In order to extract only the Minimal Sequences from an automaton the algorithm 1 is provided. This algorithm can be separated into five parts:

- In lines 1-3, the algorithm is initialized using the sequence empty of symbols and the current state of the automaton is set as the initial one.
- In lines 4-12 is the generation process that explores the automaton using the last state reached by a previous sequence and the possible evolutions from the transitions function. During this part of the algorithm, all dysfunctional sequences are kept for the extraction process. The line 5 describes the condition that restrict the generation to Cut Sequences according to subsection 3.2. Line 7 is the on-the-flight part of minimization; it looks whether the next state reached in the generated sequence is not one of the states already reached in the considered sequence. If the next state is already known the sequence won't be generated.
- In lines 12-13, from all generated sequences, only those which are failure sequences are kept. After line 14, only the Cut Sequences which don't go through loops of the automaton are kept in the S set.
- In lines 15-25 the off-the-flight part of the minimization is done by looking for a smaller Cut Sequence in each remaining Cut Sequences from the S set. If one is found, then the non-minimal Cut sequence in removed from the S set. This set that now only contains Minimal Cut Sequences is finally returned.

This algorithm can be used directly after the algorithms present in [8]. That way all the Minimal Cut Sequence can be systematically generated from a BDMP. On the reduced example of Figure 3, only four MCS are returned by the algorithm:

- $f_{Tr1,a} f_{Tr2,a} f_{Diesel1,a}$
- $f_{Tr1,a} f_{Diesel1,d} f_{Tr2,a}$
- $f_{Diesel1,d} f_{Tr1,a} f_{Tr2,a}$
- $f_{Tr1,a} f_{Tr2,a} r_{Tr1} f_{Diesel1,d} f_{Tr1,a}$

ALGORYTHM 1: Generating non looped failure sequences stopping at the first system failure

Requires: $A = (\Sigma, Q, q_i, QM, \delta =< q_v, u, q_w >)$ Finite automaton

1: $S \leftarrow \varepsilon$ *#Sequences set.*

2: $LS_{\varepsilon} = q_i$ *#Last state reached by a sequence.*

3: $KS_{\varepsilon} = \{q_i\}$ *#Set of all reached states by a sequence.*

4: **FOR EACH** $\sigma_i \in S$:

5: **IF** $LS_{\sigma_i} \notin QM$: *#Stopping at first global failure.*

6: **FOR EACH** $\delta_j \in \delta, \delta_j =< LS_{\sigma i}, u, q_n >$:

7: **IF** $q_n \notin KS_{\sigma i}$: *#Not going to an already known state in this sequence.*

8: $\sigma_n = \sigma_i u$

9: $S \leftarrow \sigma_n$

10: $LS_{\sigma n} = q_n$

11: $KS_{\sigma n} = KS_{\sigma i} \cup \{q_n\}$

12: **FOR EACH** $\sigma_i \in S$: *#Returning only faulty sequences.*

13: **IF** $LS_{\sigma_i} \notin QM$:

14: $S = S - \{\sigma_i\}$

15: **FOR EACH** $\sigma_i \in S$: *#Computing Minimal Cut Sequences set.*

16: **FOR EACH** $\sigma_j \in S, |\sigma_j| \leq |\sigma_i|$:

17: l=0

18: Index=[]

19: **FOR** $k \in [1, |\sigma_i|]$: *#going through current sequence.*

20: **IF** $\sigma_i^k == \sigma_j^l$:

21: l=l+1

22: Index[l]=k

23: **IF** $l == |\sigma_j|$: *#All symbols were found*

24: **IF FOR ALL** $(Index[k], Index[k+1]), Index[k] \leq Index[k+1]$: *#in the same order,*

25: $S = S - \{\sigma_i\}$ *#then the sequence is removed.*

26: **return** S

5. RESULTS FOR THE COOLANT FEEDING SYSTEM

The studied case of Figure 1, modeled by the BDMP given in Figure 2, is represented by a 16384 states equivalent automaton. Algorithm 1 was used to extract the Minimal Cut Sequences of length 0 to 6 from this automaton. As shown in Table 1, 23797948 dysfunctional sequences have been found; 1276750 are failure sequences. Only 129 sequences from the 336365 Cut Sequences are *Minimal* Cut Sequences.

Table 1: Number of sequences (per increasing order of length) for the studied case

Length	# Dysfunctional sequences	# Failure sequences	# Cut Sequences	# non-looped Cut Sequences	# Minimal Cut Seq.	Computing Time
0	1	0	0	0	0	<0.01s
1	11	0	0	0	0	0.01s
2	124	9	9	9	9	0.09s
3	1437	255	172	172	19	0.95s
4	17086	4897	2402	2226	39	8.43s
5	207697	79594	28420	23458	17	78.77s
6	2571592	1191995	305362	215451	45	748.49s
Sum	2797948	1276750	336365	241316	129	

The 9 Minimal Cut Sequences of length 2 are describing the failures of the two groups of pumps since in each of those groups, two failures are sufficient to make the system fail. From length 3, all the Minimal Cut Sequences are describing the minimal scenarios which make the heavily redundant power supply of the system fail. At length 3 and 4 the MCS are only composed of failure events. From length 5 the MCS may contain repair events. Examples of those sequences are shown in Table 2.

Table 2: Examples of Minimal Cut Sequences of length from 2 to 6

Length 2	Length 3	Length 4
f-DBA1-a,f-DBA2-a	f-Diesel1-d,f-Diesel2-d,f-Grid-a	f-Diesel1-d,f-Diesel2-d,f-DBB1-a,f-DBB2-a
f-C3-d,f-C2-a	f-Diesel1-d,f-DBB1-a,f-DBA2-a	f-Diesel1-d,f-Diesel2-d,f-Tr1-a, f-Tr2-a
f-C3-d, f-C1-a	f-Diesel1-d,f-Grid-a,f-Diesel2-a	f-Diesel1-d,f-DBB1-a,f-Diesel2-d,f-DBB2-a
f-C2-a, f-C3-a	f-Diesel1-d,f-Grid-a,f-DBA2-a	f-Diesel1-d, f-DBB1-a, f-DBB2-a, f-Diesel2-a
f-C2-a,f-C1-a	f-Diesel2-d,f-Diesel1-d,f-Grid-a	f-Diesel1-d, f-Tr1-a, f-Diesel2-d, f-Tr2-a
f-C1-a,f-C3-a	f-Diesel2-d,f-DBA1-a,f-DBB2-a	f-Diesel1-d, f-Tr1-a, f-Tr2-a, f-Diesel2-a
f-C1-a,f-C2-a	f-Diesel2-d,f-DBA1-a,f-Grid-a	f-Diesel1-d, f-Tr1-a, f-Tr2-a, f-DBA2-a
f-D2-d,f-D1-a	f-Diesel2-d,f-Grid-a,f-Diesel1-a	f-Diesel2-d, f-Diesel1-d, f-DBB1-a, f-DBB2-a
f-D1-a, f-D2-a	…	…

Length 5	Length 6
f-Tr1-a,f-DBB1-a,f-Diesel1-a,f-Diesel2-d,f-Tr2-a	f-Tr1-a,f-DBA1-a,f-Tr2-a,r-Tr1,f-Diesel2-d,f-Tr1-a
f-Tr1-a,f-DBB1-a,f-Diesel1-a,f-Tr2-a,f-Diesel2-a	f-Tr1-a,f-Tr2-a,f-Diesel1-a,r-Tr1,f-Diesel2-d,f-Tr1-a
f-Tr1-a,f-DBB1-a,f-Diesel2-d,f-Diesel1-a,f-Tr2-a	f-DBA1-a,f-DBB2-a,f-DBB1-a,r-DBA1,f-Diesel2-d,f-Diesel1-a
f-DBA1-a,f-DBB2-a,r-DBA1,f-Diesel2-d,f-DBA1-a	f-DBA1-a,f-DBB2-a,r-DBA1,f-Diesel2-d,f-Diesel1-d,f-DBB1-a
…	…

6. CONCLUSION

In this paper a definition and a method to extract Minimal Cut Sequences from a BDMP model have been given. Using our previous work, the MCS are systematically generated from the finite automaton which is equivalent to a BDMP. This work allows performing qualitative analyses of repairable systems modeled by a BDMP. The computed MCS contain both repair and failure events, describing much more complex scenarios than the MCS obtained from models reserved for non-repairable systems. Our current work has several objectives. First of all, we are developing a mathematical framework that will allow the formal definition of Minimal Cut Sequences for any reliability model whose failure (and repair) scenarios can be represented by a finite automaton. We think that in that way, a unified definition can be found, independently from the model used. Another goal of current research is to optimize the algorithms, by using for example a hashing function in the inclusion test in the minimization process.

References

[1] Dugan, JB. Bavuso, SJ and Boyd, MA. Dynamic fault-tree models for fault-tolerant computer systems. IEEE Transactions on reliability. 41(3), pp. 363-377, 1992

[2] Tang, Z. and Dugan, JB. Minimal cut set/Sequence generation for dynamic fault trees. Reliability and Maintainability, 2004 Annual Symposium-RAMS, pp. 207-213. 2004

[3] Rauzy, A. and Dutuit, Y. Exact and truncated computations of prime implicants of coherent and non-coherent fault trees within Aralia. Reliability Engineering and System Safety. 58(2), pp. 127-144. 1997

[4] Merle, G. Roussel, J.-M. Lesage ,J.-J. and Vayatis, N. Analytical Calculation of Failure Probabilities in Dynamic Fault Trees including Spare Gates, Proceedings of ESREL 2010, pp. 794-801, 2010

[5] Liu, D. and Xing, W. and Zhang, C. and Li, R. and Li, H. Cut sequence set generation for fault tree analysis. Embedded Software and Systems, pp. 592-603. 2007

[6] Merle, G. Roussel, J.-M.and Lesage ,J.-J. Algebraic Determination of the Structure Function of Dynamic Fault Trees. Reliability Engineering and System Safety. 96(2), pp. 267-277. 2011

[7] Bouissou, M. and Bon, J.L. A new formalism that combines advantages of fault-trees and Markov models: Boolean logic Driven Markov Processes. Reliability Engineering and System Safety. 82(2), pp. 149-163. 2003

[8] Chaux, P.-Y. Roussel, J.-M. Lesage, J.-J. Deleuze, G. and Bouissou M. Qualitative analysis of a BDMP by Finite Automaton. Proceedings of ESREL 2011, pp. 2050-2057. 2011