Generalized Boolean logic Driven Markov Processes: a powerful modeling framework for Model-Based Safety Analysis of dynamic repairable and reconfigurable systems

Pierre-Yves Piriou, Jean-Marc Faure, Jean-Jacques Lesage

To cite this version:

HAL Id: hal-01393700
https://hal.archives-ouvertes.fr/hal-01393700
Preprint submitted on 8 Nov 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Generalized Boolean logic Driven Markov Processes: a powerful modeling framework for Model-Based Safety Analysis of dynamic repairable and reconfigurable systems

Pierre-Yves Piriou\textsuperscript{a}, Jean-Marc Faure\textsuperscript{b}, Jean-Jacques Lesage\textsuperscript{c}

\textsuperscript{a}Electricité de France, R\&D, 78400 Chatou, France
\textsuperscript{b}LURPA, ENS Cachan, Univ. Paris Sud, Sapmeca, Univ. Paris-Saclay, 94235 Cachan, France
\textsuperscript{c}LURPA, ENS Cachan, Univ. Paris Sud, Univ. Paris-Saclay, 94235 Cachan, France

Abstract

This paper presents a modeling framework that permits to describe in an integrated manner the structure of the critical system to analyze, by using an enriched fault tree, the dysfunctional behavior of its components, by means of Markov processes, and the reconfiguration strategies that have been planned to ensure safety and availability, with Moore machines. This framework has been developed from BDMP (Boolean logic Driven Markov Processes), a previous framework for dynamic repairable systems. First, the contribution is motivated by pinpointing the limitations of BDMP to model complex reconfiguration strategies and the failures of the control of these strategies. The syntax and semantics of GBDMP (Generalized Boolean logic Driven Markov Processes) are then formally defined; in particular, an algorithm to analyze the dynamic behavior of a GBDMP model is developed. The modeling capabilities of this framework are illustrated on three representative examples. Last, qualitative and quantitative analysis of GBDMP models highlight the benefits of the approach.

Keywords: Model Based Safety Analysis, Generalized Boolean logic Driven Markov Processes, Dynamic and repairable system, Reconfiguration strategies, Moore machine

1. Introduction

Safety analysis of a critical system requires first that the structure of this system has been previously modeled. Qualitative and quantitative analysis results depend indeed not only on the features of the system components but also on their organization (serial or parallel configuration, \(k\text{-out-of-}n\) redundancies). System structure is classically modeled by a tree with logical gates in fault tree analysis, a popular and widespread safety assessment technique in industry. A weakness of this approach has been identified since more than twenty years, however. Only combinations of faults are considered whereas in some cases the failure of the system depends on fault sequences. This explains why several proposals of dynamic ([1]), or temporal ([2]), fault trees that permit to obtain these sequences have been published; formalization of the dynamic gates that are included in these trees by means of Petri nets ([3]), Markov chains ([4]), algebraic approaches ([5], [6] [7], [8], [9] and [10]), has been also presented.

All these extensions of the original fault-tree method have assumed that the components of the system under analysis are not repairable, which is not the case for every critical system and in particular for systems whose duration of the mission is over several years, like power plants and power distribution networks. New modeling frameworks (e.g. [11], [12], [13] and [14]) have then been developed. These formalisms allow to model explicitly, in addition to the structure of the system, the dysfunctional behavior of its components by using for instance Markov processes or transition systems.

Nevertheless, despite the benefit of these worthwhile contributions for a more accurate safety analysis, an issue remains. Redundancies management requires to define reconfiguration strategies, e.g. to describe how the service is transferred from a main component which has failed to one or several spare components and how the operation of the main component is resumed once it has...
been repaired. Reconfiguration strategies can be complex when multi-state components are considered and deserve to be explicitly and formally described. Moreover, they are performed by human operators or, more and more frequently, automatic systems. Whatever the nature of this reconfiguration controller, it may fail and this failure can impact safety ([15]). Hence, a reconfiguration strategy may fail either because the coverage of the fault(s) that trigger(s) this reconfiguration is not perfect or because its control fails. Numerous worthwhile results ([16] and [17] for instance) have been previously obtained to deal with the first issue. The aim of this paper is to tackle out the second issue.

Therefore we propose a novel modeling framework that supports Model Based Safety Analysis (MBSA) of dynamic repairable and reconfigurable systems. It permits to describe at once the structure of the critical system with a causal tree, the dysfunctional behavior of its components by means of switched Markov processes, and the reconfiguration strategies with Moore machines.

It has been termed Generalized Boolean Logic Driven Markov Processes (GBDMP) because it generalizes the BDMP frame defined in ([11]). A draft version of this framework has been sketched in ([18]); only modeling of reconfiguration strategies in a non-formal manner was considered in this reference. The current paper presents a widely extended - modeling of the structure of the system and of the dysfunctional behavior of components is now also considered - and far more formalized version.

The outline of the paper is the following. Section 2 starts with a reminder on BDMP; the limitations of this framework for reconfiguration modeling are then shown in this section. The syntax and semantics of GBDMP are detailed respectively in the third and the fourth section; the evolution rules of a GBDMP model are stated and an algorithm to animate such a model according to these rules is proposed too. This theoretical contribution is illustrated in the fifth section with three simple but representative examples whereas section 6 focuses on qualitative and quantitative analysis of GBDMP models. Finally, concluding remarks and perspectives are drawn up in section 7.

2. Modeling with BDMP

The BDMP framework has been introduced ([11]) for safety analysis of systems whose components are repairable. To meet this objective, the structure is modeled by a fault tree that includes not only logical gates but also triggers; the role of a trigger is to require or not some nodes of the tree. Moreover, the leaves of the tree are no more basic events which can be represented by Boolean variables but a description of the failure/repair behavior of components in the form of Triggered Markov Processes (TMP). The formal definition of BDMP is reminded and exemplified below; discussion of the example permits to pinpoint the limitations of BDMP for reconfiguration modeling.

2.1. Formal definition

Definition 1. Formally, a BDMP is a 4-tuple \( \langle F, te, T, (P) \rangle \) where:

- \( F \) is a multi-top fault tree, i.e. a 3-tuple \( \langle N, E, \kappa \rangle \) where:
  - \( N = G \cup L \) is a set of nodes, which is partitioned in two disjoint sets: \( G \) (set of gates) and \( L \) (set of leaves);
  - \( E \subseteq G \times N \) is a set of oriented edges, such that \( < N, E > \) is a directed acyclic graph;
  - \( \kappa \in G \rightarrow \mathbb{N}^* \) is a function that determines the gates kind. Let \( g \) be a gate which has \( n \) sons: if \( \kappa(g) = n \) then \( g \) is an AND gate, if \( \kappa(g) = 1 \) then \( g \) is an OR gate, and more generally, if \( \kappa(g) = k \) then \( g \) is a \( k/n \) gate;

- \( te \in G \) is the top event of \( F \);

- \( T \subseteq (N \backslash \{ te \}) \times (N \backslash \{ te \}) \) is a set of triggers;

- \( P \) is a set of Triggered Markov Processes (TMP) associated to the leaves. A TMP is a 5-tuple \( \langle Z_0, Z_1, X_F, f_{0 \rightarrow 1}, f_{1 \rightarrow 0} \rangle \) where:
  - \( Z_0 \) and \( Z_1 \) are two homogeneous continuous Markov chains. We denote by \( X_0 \) and \( X_1 \) their respective state spaces;
  - \( X_F \subseteq X_0 \cup X_1 \) is the subset of failure states;
  - \( f_{0 \rightarrow 1} \in X_0 \times X_1 \rightarrow [0, 1] \) is the probabilistic transfer function between \( X_0 \) and \( X_1 \);
  - \( f_{1 \rightarrow 0} \in X_1 \times X_0 \rightarrow [0, 1] \) is the probabilistic transfer function between \( X_1 \) and \( X_0 \).

2.2. Example of BDMP

A BDMP model is depicted in Figure 1. The set of gates in the fault tree is \( G = \{ G_1, G_2, G_3 \} \) with \( \kappa(G_1) = 2, \kappa(G_2) = \kappa(G_3) = 1 \). The set of leaves is \( L = \{ C_1, C_2, C_3 \} \). One trigger is introduced (dashed arrow from \( G_2 \) to \( G_3 \)); this trigger means that when the output of \( G_2 \) is True (resp. False) the part of the system related to \( G_3 \) (\( C_2 \) or \( C_3 \)) is required (resp. not required).
The TMP associated to every leaf\(^1\) comprises four states: \(S\) (Standby), \(F_1\) (Faulty during standby), \(W\) (Working) and \(F_2\) (Faulty during working). The solid arrows represent transitions of continuous Markov chains (the label of the transition is a failure or repair rate in this case), whereas the dashed arrows represent operation mode changes, from standby to working and vice versa (the label of the transition is then the probability of firing when the change is required). Thus, this TMP is composed of two Markov chains which describe the behavior of a working and standby component and are connected by two transfer functions which model the actions of the trigger.

To sum up, a BDMP model is a fault tree whose leaves are TMP. The state of each node \(n\) (leaf or gate) is characterized by two Boolean variables that represent its activation status \(M_n\) and its failure status \(F_n\). The activation statuses are controlled by the triggers; when the origin of a trigger is faulty (respectively not faulty), the destination is required (respectively not required).

Hence, a node is activated (\(M_n\) becomes True) if and only if it is required and at least one of its fathers in the tree is activated, assuming that the top event is always active. The failure status of a gate is computed from the failure statuses of its sons like in classical fault tree analysis.

\(^1\)It is possible to associate different types of TMP, with always two Markov chains, to the leaves. The transition from \(S\) to \(F_1\) is removed if it is assumed that the component cannot fail in standby mode, for instance. Furthermore, failure on-demand can be easily modeled by replacing the transition from \(S\) to \(W\) by two transitions: one from \(S\) to \(F_1\) with a probability \(p\) (failure probability) and the other one from \(S\) to \(W\) with a probability \(1 - p\). Nevertheless, only one type will be considered here for brevity reasons.

2.3. Reconfiguration modeling

The concept of trigger that is introduced by the BDMP framework is a first attempt to model reconfiguration. Despite its novelty and interest when repairable systems are considered, this modeling primitive presents three limitations:

- First, only one reconfiguration strategy is considered: the destination of the trigger is activated as soon as the origin of the trigger fails and is deactivated as soon as the origin is repaired. This strategy is not the only one which is used in practice, however. When standby redundancy is implemented with two identical components, with the same failure rate, for instance, it is frequent to activate the origin, once repaired, only when the destination has failed to balance the working durations of the two components, and decrease the risk of failure on demand, if it exists.

- Second, the models of components (leaves of the fault tree) include only two operation modes: working and standby. Nonetheless, real components of critical systems may have more than two modes, for instance a standby mode, a normal mode and an overspeed mode, the latter one being a solution to perform the service during a limited time when the component is the only faultless one that remains.

- Last, possible failure of the trigger is not considered. It is assumed indeed that, when the origin of a trigger fails, the trigger always sends to its destination a request to move to the working operation mode. This is unfortunately not always true in practice, and especially when the trigger is implemented by an automatic system that comprises electronic boards, relays, etc. which may fail.

To overcome these limitations (restricted number of reconfiguration strategies, of operation modes, failure of the control of the reconfiguration not considered) a novel framework is defined in the next section.

3. Generalized Boolean logic Driven Markov Processes (GBDMP)

GBDMP have been defined from BDMP by replacing first the concept of trigger by that of switch whose behavior is described by a Moore machine; complex
reconfiguration strategies can then be modeled. Moreover, TMP are replaced by SMP (Switched Markov Processes) to model components with more than two operation modes. Last, control of the reconfiguration strategy is explicitly modeled and connected to switches; hence, the impact of failures of this control can be considered.

The syntax of these models is first detailed in what follows; properties that must be satisfied by well-formed GBDMP are stated too.

3.1. Overall description

Definition 2. A Generalized Boolean logic Driven Markov Processes is a 6-tuple \( < V, E, \kappa, \nu, \text{str}, \text{smp} > \) where:\n
\[ V = N \cup S = G \cup L \cup S \text{ is a set of vertices partitioned into the nodes (i.e. the gates and the leaves) and the switches.} \]

\[ E = E_F \cup E_S \text{ is a set of oriented edges, such that } E_F \subseteq G \times N \text{ and } E_S \subseteq (N \times S) \cup (S \times N); \]\n
\[ \kappa : G \rightarrow \mathbb{N}^+ \text{ is a function that determines the gates kind (just as with BDMP);} \]

\[ \nu : E \rightarrow \mathbb{N} \text{ is a function that associates an integer label to each edge;} \]

\[ \text{str} : S \rightarrow M \text{ is a function that associates a Moore machine (a strategy) to each switch;} \]

\[ \text{smp} : C \rightarrow \mathbb{P} \text{ is a function that associates a SMP to each component;} \]

A simple GBDMP is shown at Figure 2. The graphical representation of leaves and gates of the fault tree is the same as for BDMP. A dashed rectangle represents a switch \( (S = \{ S1 \}) \) and the solid (resp. dashed) arrows the edges of \( E_F \) (resp. \( E_S \)), which connect respectively the gates to the nodes (leaves or gates) and the switches to the nodes or the nodes to the switches; the label of an edge is the value of the function \( \nu \) for this edge. The behavior of the leaves \( C1, C2, C3 \) and \( C4 \) is depicted at part b) of Figure 2 and that of \( S1 \) at part c). Compared to Figure 1, this GBDMP includes a new component \( (C4) \) that is in charge of reconfiguration.

Two directed graphs can be defined in the structure of a GBDMP model:

- \( G_F = < N, E_F > \) is the graph classically called the Fault Tree. The label of an edge of this graph corresponds to an operation mode of the destination node. At figure 2 a), for instance, the labels of the two edges from \( G3 \) to \( C2 \) and \( C3 \) mean that these leaves must be respectively in their second and first operation mode when this gate is required. In a similar way, \( C2 \) must be in its first (resp. second) operation mode when \( G2 \) (resp. \( G3 \)) is required.

- \( G_S = < V, E_S > \) is the graph where switches are connected to nodes. The labels of edges of this graph correspond merely to numbers of the inputs and outputs of switches.

For every vertex of these two graphs, it is possible to define:

- the sets of its downstream and upstream vertices:
  \[ \forall n \in N, \Gamma^-_F(n) = \{ g \in G | (g, n) \in E_F \} \]
  \[ \forall g \in G, \Gamma^-_S(g) = \{ n \in N | (g, n) \in E_S \} \]
  \[ \forall s \in S, \Gamma^-_S(s) = \{ n \in N | (s, n) \in E_S \} \]

- its indegree and outdegree (\( G \) can be replaced by \( G_F \) or \( G_S \) herebelow):
  \[ \forall v \in V, d^-_F(v) = \text{Card}(\Gamma^-_F(v)) \]
  \[ \forall v \in V, d^-_S(v) = \text{Card}(\Gamma^-_S(v)) \]

\(^2M \text{ and } \mathbb{P} \text{ designate respectively the set of Moore machines and the set of SMP.} \]
These notations will be used in the remainder of this section, in particular to state the consistency properties of well-formed GBDMP.

3.2. Leaf behavior modeling

The behavior of a leaf is modeled by a k-SMP which is composed of k Markov chains. Each Markov chain corresponds to an operation mode and comprises faultless and faulty states; the transitions between these states are stochastic because they model mainly failures and repairs. It must be noted that the Markov chains that compose a k-SMP are not necessarily homogeneous; different distributions (e.g., exponential, log-normal, Weibull) can be associated to transitions. However, quantitative analysis of the constructed model requires the tool which will be selected for this analysis is able to deal with the distributions introduced in the model. In the examples of this paper, only exponential distributions (then constant failure and repair rates) will be considered because this distribution is the most common one.

In the example of Figure 2 b), the 3-SMP associated to the leaves C1, C2 and C3 comprises three Markov chains to represent a component with two working modes and one standby mode; in this model, it is assumed that no failure occurs in the standby mode and that the failure rate in the second working mode is greater than the corresponding rate in the first working mode. The set of states $X^P$ of the k-SMP $P$ is the union of the sets of states of the chains; similarly, the set of states $X^F$ of the k-SMP is the union of the sets of faulty states of the chains. $k(k-1)$ probabilistic transfer functions between the chains of a k-SMP must be defined.

The value of the transfer function between two states of two different chains is equal to 1 (deterministic transfer) if no failure on-demand is considered (case of Figure 2 b) when the operation mode is changed and belongs to [0, 1] otherwise.

Definition 3. A k-mode Switched Markov Process (k-SMP) is defined as a 3-tuple

$P = \langle (Z^P_i)_{0 \leq i < k}, X^P, (f^P_{i,j})_{0 \leq i < j \leq k} \rangle$ where:

- $(Z^P_i)_{0 \leq i < k}$ is a family of Markov chains i.e. $\forall i \in [0, k-1]$, $Z^P_i$ is a 3-tuple $< X^P_i, A^P_i, p_0^P_i >$ where:
  - $X^P_i$ is a finite set of states;
  - $A^P_i : (X^P_i)^2 \rightarrow \mathbb{R}^+$ is the matrix of transition rates;
  - $p_0^P_i : X^P_i \rightarrow [0, 1]$ is the initial probability distribution ($\sum_{x \in X^P_i} p_0^P_i(x) = 1$);

$X^P = \bigcup_{i=0}^{k-1} X^P_i$ denotes the set of all states of the SMP.

- $X^P \subseteq X^P$ is the subset of failure states;

- $(f^P_{i,j})_{0 \leq i < j \leq k} \subseteq [0, 1]$ is a family of probabilistic transfer functions, i.e. 

$\forall (i, j) \in [0, k-1]^2 \forall x \in X^P_i, \sum_{y \in X^P_j} f^P_{i,j}(x, y) = 1$.

When a k-SMP is associated to a leaf, it is said that the dimension of this leaf is equal to k. The activation status of a leaf whose dimension is greater than 2 cannot be represented by a Boolean variable, as this was the case with BDMP$^3$, but by an integer. Calculus of the value of this integer variable will be dealt with in the next subsection.

3.3. Node status variables

For each node (leaf or gate) $n \in N$ of the fault tree, three status variables must be defined:

- $F_n$: a Boolean variable ($F_n \in \{\text{False}, \text{True}\}$) that represents the failure status of the node ($F_n = \text{True} \Leftrightarrow n$ is faulty);

- $R_n$: a binary variable ($R_n \in \{0, 1\}$) that represents the requirement status of the node ($R_n = 1 \Leftrightarrow n$ is required to perform the service);

- $M_n$: a positive integer variable ($M_n \in \mathbb{N}$) that represents the activation status of the node ($M_n = k \Leftrightarrow n$ is in the operation mode number $k$).

The failure statuses are determined as follows:

- For a leaf $l \in L$, $F_l$ is True when the active state of the SMP associated to this leaf (denoted $X_l$) is a faulty state.

$$X_l \in 3^{\text{SMP}(l)} \Rightarrow F_l = \text{True} \quad (1)$$

- For a gate $g \in G$, $F_g$ is True when the number of its sons that are either faulty or non-required is greater than $\kappa(g)$.

$$\text{Card}(\{n \in G^+_g | (F_n \vee \neg R_n) \geq \kappa(g)\}) \Rightarrow F_g = \text{True} \quad (2)$$

In the example of Figure 2, the failure statuses of the leaf C1 and the gates G1 and G2, for instance, are respectively obtained as follows:

$^3$A TMP can be seen as a 2-SMP.
\[ X_{C1} \in \{ F_0, F_1, F_2 \} \Rightarrow F_{C1} = True \]
\[ Card(n \in [G2, G3] | F_n \lor \neg R_n) \geq 2 \Rightarrow F_{G1} = True \]
\[ Card(n \in [C1, C2] | F_n \lor \neg R_n) \geq 1 \Rightarrow F_{G2} = True \]

When a node is not connected to any switch output, it is always required \((R_n = 1)\). Else, its requirement status is obtained from the Moore machine associated to its upstream switch (in \(G_S\)) as explained in the next section.

The activation status of a node \(n\) is computed with Eq. (3):

\[
\begin{cases} 
  \text{if } \Gamma_{gr}(n) \neq \varnothing & M_n = R_n \cdot \max_{g \in \Gamma_{gr}(n)} \{ M_g, v((g, n)) \} \\
  \text{else} & M_n = R_n 
\end{cases}
\]

where \(M_g\) is the activation status of an upstream gate \(g\) and \(v((g, n))\) is the label of the edge between \(g\) and \(n\) in \(G_F\).

For the example of Figure 2, for instance: \(M_{C2} = R_{C2}.\max(M_{G2}, M_{G3})\) (the activation status of \(C2\) is equal to 1 when \(G2\) is activated and 2 when \(G3\) is activated.)

Each possible value of the activation status of a leaf refers to a Markov chain of the associated SMP. For each leaf \(l \in L\), while \(M_l = i\) \((i \in \mathbb{N})\) the active Markov chain of \(smp(l)\) has to be the chain number \(i\) \((M_l = i \Rightarrow X_l \in X_{smp(l)}^\text{active})\).

### 3.4. Switch behavior modeling

The role of a switch is to set/reset the requirement statuses of the nodes that are connected to its outputs according to the values of its inputs and the reconfiguration strategy which is described by the associated Moore machine.

A Moore machine [19] is an automaton with inputs and outputs which is defined as follows:

**Definition 4.** A Moore Machine is defined as a 6-tuple \(M = \langle Q_M, \Sigma_I^M, \Sigma_O^M, \text{trans}^M, \text{out}^M \rangle\) where:

- \(Q_M\) is a finite set of states;
- \(Q_I^M\) is the initial state;
- \(\Sigma_I^M\) is the input alphabet;
- \(\Sigma_O^M\) is the output alphabet;
- \(\text{trans}^M : Q_M \times \Sigma_I^M \rightarrow Q_M\) is the transition function;
- \(\text{out}^M : Q_M \rightarrow \Sigma_O^M\) is the output function.

In the graphical representation of this automaton (Figure 2 c), the labels of the transitions are elements of the input alphabet and the elements of the output alphabet are associated to the states.

It is then possible to represent any reconfiguration strategy with a Moore machine by defining the input/output alphabets of this machine as follows, assuming that the elements of the input (output) alphabets are ordered according to the labels of the edges of \(G_S\) that are incoming (outgoing) to (from) the switch to which this machine is associated.

- An element of the input alphabet of a Moore machine represents a combination of states of the nodes which are connected to the inputs of the switch whose behavior is described by this machine. In most cases, it is sufficient to know the failure status \(F_n\) of a node to characterize its state and select the appropriate reconfiguration strategy. More details on the state of the node are needed sometimes, however; in these cases, the state of the node will be characterized by the active state \(X_i\) of the associated SMP. Hence, when the switch which is associated to the Moore machine owns \(i\) inputs, an element of the input alphabet will be a vector with \(i\) components that are either failure statuses or SMP states. For the Moore machine \(M1\) at Figure 2 for instance, the elements of the input alphabet are built from the possible states \((W, F)\) of the SMP associated to \(C4\) (first input) and the failure status \((True, False)\) of \(G2\) (second input).

- An element of the output alphabet of a Moore machine represents a combination of requirement statuses of the nodes which are connected to the outputs of the switch whose behavior is described by this machine. For the same example, the elements of the output alphabet are built from the possible requirement statuses \((0, 1)\) of \(G2\) and \(G3\).

Globally, this machine describes a reconfiguration strategy where \(G2\) must be required and \(G3\) must not when \(G2\) is faultless (state \(q_0\) of the Moore machine) and vice versa when \(G2\) is faulty (state \(q_1\) of the Moore machine). This strategy may fail in case of failure of \(C4\). No state change is possible indeed in this case, even if necessary.

The formula that describes how the requirement status of a node \(n\) is updated can now be given:

\[
\begin{cases} 
  \text{if } \exists s \in S((s, n) \in E_S) & R_n = (\text{out}_{\text{str}}(U_s))_{b(s, n)} \\
  \text{else} & R_n = 1 
\end{cases}
\]
where $U_s$ denotes the active state of the Moore machine associated to the switch $s$, and $\text{out}(s,t)$ is the output function of this Moore machine (cf. Definition 4), thus $\text{out}(s,t)(U_s)$ is the element number $\nu((s, n))$ of the output of the Moore machine $\text{str}(s)$ when its active state is $U_s$. For the example of Figure 2, for instance: $R_{G3} = \sigma_1$ with $(\sigma_0, \sigma_1) = \text{out}(1)(U_{S1})$

Last, it can be noted that the behavior of a BDMP trigger can be modeled (Figure 3) by a Moore machine with only one input (the failure status of the origin of the trigger) and one output (the required position of the destination of the trigger). Only one strategy is possible however: the destination is required whenever the origin is faulty and not required otherwise. The control of the reconfiguration is obviously out of the scope of this modeling, as pointed out in subsection 2.3.

![Figure 3: Moore machine that models the strategy of a BDMP trigger](image)

### 3.5. Consistency properties

A GB-DMP model is obtained by integrating a representation of the structure of the system by using a fault tree $G_F$ and a graph $G_S$ that describes the inputs and outputs of switches, switched Markov processes to describe the dysfunctional behavior of the leaves of the fault tree and Moore machines to describe the functional behavior of the switches. To ensure consistency of this model, five properties that must be satisfied by any GB-DMP have been defined.

**Property 1.** The number of sons of a gate must be compatible with its type:

$$\forall g \in G, \kappa(g) \leq d_G^*(g)$$

This property means that a $k$-out-of-$n$ gate must have at least $k$ sons.

**Property 2.** A node (gate or leaf of the fault tree) cannot be connected to several outputs of switches:

$$\forall n \in N, d_{G_F}^*(n) \leq 1$$

This property avoids conflicts between reconfiguration orders.

**Properties 3 and 4** focus on switches and their associated Moore machines. The ranges of the input and output numbers of a switch $s$ will be respectively noted $I^{s}(s)$ and $\Gamma^{s}(s)$:

$$\Gamma^{s}(s) = \begin{cases} [0, d_{G_F}^*(s) - 1] & \text{if } \nu((n, s)) \text{ is a real issue because they are static, i.e. they do not depend on the current state of the model.} 

A GB-DMP model that satisfies these properties is called well-formed. Its evolutions in response to sequences of events can be analyzed once the semantics of GB-DMP has been formally defined. This is the objective of the next section.

### 4. GB-DMP semantics

The global state of a GB-DMP at a given date is completely defined by the set of the state variables of every leaf ($X_i \in X_{\text{mp}}(t), 1 \leq L$) and the set of the state variables of every switch ($U_s \in Q_{\text{str}}(s), \forall s \in S$). Hence, the dynamic behavior of a GB-DMP can be represented by a state model whose states are global states of the GB-DMP and transitions are determined as explained below.
4.1. Spontaneous and provoked events

The evolutions of a GBDMP model are driven by two types of events:

- **spontaneous events**: A spontaneous event is an uncontrollable event; its occurrence date is a random variable. Failure events (except failure on-demand), repair events, phase change events are examples of spontaneous events. They correspond to the solid arrows in the SMP representation (cf. Figure 2 b).

- **provoked events**: A provoked event is the consequence of a spontaneous event. As the reactions of the GBDMP are assumed instantaneous, the date of such an event is the same as that of its cause. Operation mode changes, e.g. from standby to working, and failure on-demand are examples of provoked events. When several provoked events are concurrent for a leaf after the occurrence of a given spontaneous event, the probabilities of those events are given by the transfer function of the corresponding SMP. These events correspond to the dashed arrows in the SMP representation (cf. Figure 2 b).

### 4.2. GBDMP evolution rules

The initial state of a GBDMP model is obtained as follows:

1. The active state of every Moore machine is its initial state: \( \forall s \in S, U_s = q_{str(s)} \).
2. The requirement and activation statuses of every node are computed respectively according to Eqs. (4) and (3).
3. The initial state of every leaf can then be determined using the initial probability distribution \( p_{M_0}^{\text{mp}(0)} \) of the corresponding SMP.
4. The failure status of every node is computed according to Eqs. (1) and (2).

The state of a GBMP is said stable if and only if the activation and failure status of every leaf complies with the state of the associated SMP. The stability condition of a state is formally given at Eq. (5).

\[
\forall l \in L, \left\{ X_l \in X_{M_l}^{\text{mp}(l)} \land (F_l \land X_l \in X_F^{\text{mp}(l)}) \lor (\neg F_l \land X_l \notin X_F^{\text{mp}(l)}) \right\}
\]

A stable state can change only when a spontaneous event occurs. The state of a leaf is then changed and the new stable state of the GBDMP is determined by:

1. Updating every other variable ( statuses of nodes and active state of Moore machines).
2. If the new state is not stable, provoked events occur to set every SMP in the correct mode. If one of these events is a failure on-demand, steps 1 and 2 must be repeated until the reached state is stable.

It must be noted that the loop introduced above (repetition of the steps 1 and 2) is not infinite because at worst it will finish when every component will be faulty. Computation of the new stable state, which is a fixed point research characterized by the stability condition, always converges.

In response to spontaneous events, a GBDMP model evolves from stable state to stable state by crossing unstable states. This is illustrated at Figure 4, for the example of Figure 2, where solid and dashed rectangles represent respectively stable and unstable states. It is assumed that the probability of the initial state of the SMP associated to every leaf is equal to 1 to define the initial state of the GBDMP. From this state, the evolution starts when the leaf \( C1 \) fails what causes the evolution of \( S \) from \( q_0 \) to \( q_1 \). This evolution implies that \( C1, C2 \) and \( C3 \) have to be switched respectively into mode 0, 2 and 1, what explains the following occurrences of the three provoked events. The final state is stable according to Eq. (5).

### 4.3. Simulation of a GBDMP

Once the evolution rules defined, an algorithm to obtain the evolutions of a GBDMP in response to a sequence of spontaneous events has been developed (Algorithm 1). It is assumed that simultaneous occurrences of spontaneous events are not possible. Hence, as an evolution of the GBDMP between two successive stable states is instantaneous (instantaneous reaction of the GBDMP), the GBDMP is always in a stable state when a spontaneous event occurs.

Dependency analysis of the variables which characterize a GBDMP state ( statuses and state variables of SMP and Moore machines) must be performed before
computing their new values because these variables are highly interdependent, as illustrated at figure 5.

![Dependency graph for the variables of Figure 2](image)

Figure 5: Dependency graph for the variables of Figure 2

It must be underlined that the updating of variables is possible if and only if the GBDMP is well-formed and in particular satisfies property 5. When this is the case, it is performed in Algorithm 1 by ranking the vertices (nodes and switches) \( V \) of the considered GBDMP according to their relative positions in \( \mathcal{G}_F \) and \( \mathcal{G}_S \).

A prototype tool named SAGE (Safety Analysis in a GBDMP Environment) has been developed to implement this algorithm. This tool includes also edition and simulation functions and has been used to build and analyze the three examples of the next section.

5. Examples

The aim of this section is to show the modeling capabilities of the GBDMP framework on the basis of three simple but representative examples. Several reconfiguration strategies and failure of the control of these strategies are addressed in the first example. The second example focuses on components with more than two operation modes and the third one on a simple phased-mission system. For each example, the corresponding

**Algorithm 1** Discrete Event Simulation of a GBDMP model

**Require:**
- \( \bullet < V, E, k, \nu, \text{str}, smp > \) a well-formed GBDMP model (cf. Definitions 2, 3, 4 and Rules 1, 2, 3, 4 and 5).
- \( \bullet \sigma = [e_1, ..., e_k] \) a sequence of spontaneous events.

**Ensure:** A possible evolution of the GBDMP model.

1: \{\} Initialization:
2: \( \text{lev}_{\max} := \max_{v \in V}(\text{Level}(v)) \)
3: \( \forall n \in N : F_n := \text{False} \)
4: \( \forall s \in S : U_s := d_0 \)
5: \( \text{lev} := \text{lev}_{\max} \)
6: while \( \text{lev} \geq 0 \) do
7: \( \forall n \in N | \text{Level}(n) = \text{lev} \) to initialize \( R_n \)
8: \( \text{lev} := \text{lev} - 1 \)
9: end while
10: while \( \text{lev} \leq \text{lev}_{\max} \) do
11: \( \forall n \in N | \text{Level}(n) = \text{lev} \) to initialize \( M_n \)
12: \( \text{lev} := \text{lev} + 1 \)
13: end while
14: \( \forall l \in L : \) to initialize \( X_l \) using \( p^{\text{imp}}_{M_l} \)
15: \{\} Main loop:
16: \( i := 0 \)
17: while \( i \leq k \) do
18: if \( i \neq 0 \) then
19: occurrence of \( e_i \) \{modification of the state variable for the related leaf.
20: end if
21: isStable := \text{False}
22: while isStable = \text{False} do
23: while \( \text{lev} \geq 0 \) do
24: \( \forall n \in N | \text{Level}(n) = \text{lev} \) to update \( F_n \)
25: \( \forall s \in S | \text{Level}(s) = \text{lev} \) to update \( U_s \)
26: \( \forall n \in N | \text{Level}(n) = \text{lev} \) to update \( R_n \)
27: \( \text{lev} := \text{lev} - 1 \)
28: end while
29: while \( \text{lev} \leq \text{lev}_{\max} \) do
30: \( \forall n \in N | \text{Level}(n) = \text{lev} \) to update \( M_n \)
31: \( \text{lev} := \text{lev} + 1 \)
32: end while
33: \( \forall l \in L \) \( X_l \not\in X^{\text{imp}}_{M_l} \) to update \( X_l \) \{occurrence of provoked events
34: if \( \forall l \in L | (F_l \land X_l \not\in X^{\text{imp}}_{F_l}) \) then
35: isStable := \text{True}
36: end if
37: end while
38: \( i := i + 1 \)
39: end while
GBDMP model is detailed and the evolution of this model in response to a sequence of failure and repair events is analyzed; for simplicity reasons, it will be assumed in these analyses that the probability of the initial state of the SMP associated to every leaf is equal to 1 and that every transition between the Markov chains of this SMP is deterministic.

5.1. Two different reconfiguration strategies implemented on control devices that may fail

5.1.1. Example description

This example (Figure 6) comprises two groups of redundant components \((C1a, C1b, C1c)\) and \((C2a, C2b)\); the nature of these components does not matter. Every component can be in active mode or standby mode and may fail and be repaired in both modes.

The strategies selected for the two groups are different, however:

- The first group performs correctly its service when at least two components among the three ones are faultless; by default, \(C1a\) and \(C1b\) are active and \(C1c\) in standby. When one of the active components fails, it is replaced by the standby component if it is faultless. The operation of the failed component is resumed only when it is repaired and one of the currently active components fails. This type of resumption of operation for a repaired component will be termed resuming at the latest.

- The second group performs correctly its service when at least one component among the two ones is faultless; \(C2a\) must be active whenever it is faultless. Hence, when this component is repaired after it has failed, it must immediately be set in its active mode. This type of resumption will be termed resuming at the earliest.

The two strategies will be modeled by different Moore machines that will be described in what follows.

Furthermore, the control devices \(D1\) and \(D2\) own two failure modes:

- frozen (the output of the device is stuck in its current position and the combination of active components cannot be modified); the failure and repair rates are respectively \(\lambda_f\) and \(\mu_f\).

- bad contact (the output of the device is in open circuit and no combination of active components can be selected); the failure and repair rates are respectively \(\lambda_{bc}\) and \(\mu_{bc}\).

5.1.2. Modeling

The GBDMP representation of the structure of the example is given at figure 7. In addition to the classical fault tree, this well-formed model includes two switches and two leaves that correspond to the devices where the control of the reconfiguration is implemented \((D1\) and \(D2)\).

![Figure 7: Model of the structure of the example of Figure 6](image)

The SMP associated to the leaves \(C1a\) to \(C2b\) is a classical 2-SMP and that associated to \(D1\) and \(D2\) is a 1-SMP that is shown at figure 8.

![Figure 8: Switched Markov Process De](image)

A first benefit of the GBDMP framework must be clearly highlighted at this point. Using SMP to describe the behavior of the leaves permits to consider several
failure modes, even for the control of the reconfiguration.

Last, the Moore machines M2 and M3 that describe respectively the behavior of the GBDMP switches S1 and S2 are given at figure 9. These machines model the two reconfiguration strategies previously described.

The elements of the input alphabet of M2 (M3) are combinations of the active state of the SMP of D1 (D2) and the failure statuses of C1a, C1b and C1c (C2a, C2b); the elements of the output alphabet are combinations of the requirement statuses of C1a, C1b and C1c (C2a, C2b). The character _ means that any value is possible.

For M2 for instance, (2, _ _ _ _) means that D1 is faulty, the failure mode being frozen, and C1a, C1b and C1c can be faulty or not; the operation mode of C1a, C1b and C1c cannot be modified in this case, whatever it should be. (0, T, F, F) means that D1 is faultless, C1a faulty, C1b and C1c faultless; C1a is no more required and must be replaced by C1c (transition from q0 to q2).

Figure 9: Moore machine M2 (on the top) and M3 (at the bottom)

5.1.3. Simulation

An example of evolutions in response to a sequence of spontaneous events \( f_{C1c} \rightarrow f_{C1a} \rightarrow r_{C1c} \rightarrow r_{C1a} \), where \( f_{C1c} (f_{C1a}) \) represents the failure of C1c (C1a) and \( r_{C1c} (r_{C1a}) \) the repairs of C1c (C1a) is presented at Table 1. The states of C1a, C1b, C1c and the failure status of G1 are given in the rows of this table. These results are consistent with the strategy selected for this group of components: the service is not provided once two components have failed and C1a remains in standby mode once repaired (resuming at the latest strategy).

<table>
<thead>
<tr>
<th>sequence</th>
<th>0 ( \rightarrow ) f_{C1a}</th>
<th>1 ( \rightarrow ) f_{C1a}^{frozen}</th>
<th>2 ( \rightarrow ) r_{C1a}</th>
<th>3 ( \rightarrow ) r_{C1a}</th>
</tr>
</thead>
<tbody>
<tr>
<td>X_{C1a}</td>
<td>W</td>
<td>W</td>
<td>F_2</td>
<td>F_1</td>
</tr>
<tr>
<td>X_{C1b}</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
</tr>
<tr>
<td>X_{C1c}</td>
<td>S</td>
<td>F_1</td>
<td>F_1</td>
<td>W</td>
</tr>
<tr>
<td>F_{G1}</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
</tr>
</tbody>
</table>

Table 1: Example of evolution for the first group (S: Standby, F1: Faulty and standby, W: Working and F2: Faulty and working)

The results of a similar analysis with the tool SAGE for the second group of components is given at Table 2. The sequence is \( f_{C2a} \rightarrow f_{C2a}^{frozen} \rightarrow r_{C2a} \rightarrow f_{C2a} \), where \( f_{C2a} (f_{C2b}) \) represents the failure of C2a (C2b), \( r_{C2a} \) the repairs of C2a, and \( f_{C2b} \) the failure of D2 in the frozen mode. The states of C2a, C2b, D2 and the failure status of G2 are given in the rows of this table.

This analysis highlights strongly the interest of the GBDMP framework, where failures of the control are considered, for MBSA. When C2b fails indeed, the service is no more provided (G2 becomes faulty) while C2a has been previously repaired because D2 is faulty, in a frozen failure mode; the control of the reconfiguration is lost. This significant result could not be obtained with other frameworks that do not consider control devices failures.

Table 2: Example of evolution for the second group (S: Standby, F1: Faulty and standby, W: Working and F2: Faulty and working)

<table>
<thead>
<tr>
<th>sequence</th>
<th>0 ( \rightarrow ) f_{C2a}</th>
</tr>
</thead>
<tbody>
<tr>
<td>X_{C2a}</td>
<td>W</td>
</tr>
<tr>
<td>X_{C2b}</td>
<td>S</td>
</tr>
<tr>
<td>X_{D2}</td>
<td>0</td>
</tr>
<tr>
<td>F_{G1}</td>
<td>False</td>
</tr>
</tbody>
</table>

5.2. Multi-state pumps

5.2.1. Example description

Some industrial plants comprise pumps which own 3 operation modes:

- **Off**. The pump is inactive (in standby mode). It cannot fail but can be repaired with a repair rate \( \mu \).
- **On**. The pump is in its normal operation mode. It can fail with a failure rate \( \lambda \) and be repaired with a repair rate \( \mu \).
- **Over**. The pump is in an overspeed operation mode. It can fail with a greater failure rate \( 4\lambda \) and be repaired with the same repair rate \( \mu \).
An example of use of two such pumps is shown at figure 10. The service is correctly performed when either both pumps are working in normal operation mode or one pump is in Over mode; the latter solution is selected when one pump has failed.

5.2.2. Modeling

The GB-DMP of this example is given at Figure 11. In the structure view (Figure 11 a), the edges which connect both leaves to gates G2 and G3 have different labels because G2 fails when at least one pump fails in normal operation mode, while G3 fails when both pumps have failed in the Over mode indeed. With other words, when G2 is required and not G3, P1 and P2 are in mode On, whereas when G3 is required (whatever the value of the requirement status of G2) they are in mode Over. Indeed, according to Eq. (3):

- $R_{G2} = 1 \land R_{G3} = 0 \Rightarrow M_{G2} = 1 \land M_{G3} = 0 \Rightarrow M_{P1} = M_{P2} = \max(1, M_{G2}, 2, M_{G3}) = 1$
- $R_{G2} = 1 \land R_{G3} = 1 \Rightarrow M_{G2} = 1 \land M_{G3} = 1 \Rightarrow M_{P1} = M_{P2} = \max(1, M_{G2}, 2, M_{G3}) = 2$

In the 3-SMP $Pu$ associated to every leaf (Figure 11 b), the three chains 0, 1 and 2 correspond to respectively the operation modes Off, On and Over. The only element of the input (output) alphabet of the Moore machine (Figure 11 c) is the failure status of G2 (requirement status of G3); therefore, the role of the switch is to require G3 when G2 has failed.

5.2.3. Simulation

Table 3 shows the results of Algorithm 1 for the sequence $f_{P1} \rightarrow f_{P2} \rightarrow r_{P1} \rightarrow r_{P2}$, with the same notations than for the first example. These results correspond to the expected behavior; when one pump fails, the operation mode of the remaining faultless pump is switched to the Over mode and when both pumps are faultless their operation mode is On. This example shows that multi-state components with more than two operation modes can easily be considered into a GB-DMP model.

5.3. A simple phased mission system

5.3.1. Example description

A simple plant where two liquids are poured in a tank then mixed is sketched at Figure 12. In the first phase, the valve V1 is open and the valve V2 closed to pour the first liquid; when the phase change event $\delta_1$ occurs, the valve V1 is closed and the valve V2 opened to pour the second liquid and so on. Both valves may fail stuck-open or stuck-closed. Every failure is revealed only when the operation mode of the valve must be changed for a phase change and may be considered as a failure on-demand. It will be assumed that the failure (repair) rate $\lambda(\mu)$ is the same for the two types of failure.

This plant may be seen as a very simple phased-mission system. Despite its structure remains unchanged, the dysfunctional behavior and success criterion of its components change from one phase to the other one indeed. When V2 is stuck-open for instance, this valve is faulty during the first phase and faultless in the second one.
5.3.2. Modeling

The dysfunctional behavior of a valve is modeled by the 2-mode SMP $Va$ (Figure 13 b). The two Markov chains 0 and 1 represent respectively the dysfunctional behavior when the valve is expected closed and open. When the active state of this SMP for $V2$ is $C$ for instance, it can evolve to $O$ for a phase change (phase 1 to phase 2), provided that $V2$ be faultless, or to $S\ C$ if $V2$ fails during phase 1. This failure will be detected only at the phase change (transition from $S\ C$ to $SC_1$).

In the structure view (Figure 13 a), the edges which connect both leaves $V1$ and $V2$ to gates $G2$ and $G3$ have different labels because when $G2$ ($G3$) is active, the first (second) phase is performed; hence $V1$ ($V2$) is expected to be open and $V2$ ($V1$) closed.

5.3.3. Simulation

Table 4 shows the results of algorithm 1 for sequence $stack_{V1} \rightarrow \delta_{12} \rightarrow r_{V1}$ where $stack_{V1}$ represents a stuck-open failure of $V1$ during the first phase (transition from $O$ to $SO$ in the SMP of $V1$) and $r_{V1}$ the repairs of this failure (transition from $SO$ to $C$ in the SMP of $V1$). The results correspond to what was forecast, e.g. $G1$ becomes faultless only when $V1$ has been repaired even if the state of $V2$ has correctly changed after the occurrence of $\delta_{12}$.

Table 4: Behavior of the model for a scenario that involves $V1$ and the phase selector

<table>
<thead>
<tr>
<th>sequence</th>
<th>$0$</th>
<th>$stack_{V1}$</th>
<th>$\delta_{12}$</th>
<th>$r_{V1}$</th>
<th>$3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_{V1}$</td>
<td>$O$</td>
<td>$SO$</td>
<td>$SO_0$</td>
<td>$C$</td>
<td></td>
</tr>
<tr>
<td>$X_{V2}$</td>
<td>$C$</td>
<td>$C$</td>
<td>$O$</td>
<td>$O$</td>
<td></td>
</tr>
<tr>
<td>$X_{Phase\ select}$</td>
<td>$\varphi_1$</td>
<td>$\varphi_1$</td>
<td>$\varphi_2$</td>
<td>$\varphi_2$</td>
<td></td>
</tr>
<tr>
<td>$F_{G1}$</td>
<td>False</td>
<td>False</td>
<td>True</td>
<td>False</td>
<td></td>
</tr>
</tbody>
</table>

This example has showed that components with several failure models can be modeled in the GBDMP framework and that mission-phased systems can be considered too, what is not surprising because phase change is a particular reconfiguration mechanism.

6. Qualitative and quantitative analysis

This section aims to show how the choice of a reconfiguration strategy impacts the results of qualitative and quantitative analysis. To meet this objective, only one basic example, a classical standby redundancy system with two components $A$ and $B$, will be focused on. It will be assumed that every component may fail on demand; hence, its dysfunctional behavior is depicted by the SMP of Figure 15. This model is easily obtained from that of Figure 1 b) by adding a transition from the state $S$ (Standby) to the state $F2$ (Faulty during working); $\gamma$ is the failure on demand rate.

Four GBDMP models of the considered system are proposed at Figure 14:

1. The reconfiguration strategy of the first model is identical to that of a BDMP trigger and no failure of this strategy is considered. Hence, this model behaves strictly as a BDMP.

---

4In SMP $Ph$: $\varphi_i$ means phase $i$ is the active phase.
In SMP $Va$: $C$ = Closed; $S\ C$ = unattended Stuck-Closed; $SC_0$ = Stuck-Closed, expected closed; $SC_1$ = Stuck-Closed, expected open; $O$ = Open; $SO$ = unattended Stuck-Open; $SO_0$ = Stuck-Open, expected closed; $SO_1$ = Stuck-Open, expected open.
Failure of A $\Rightarrow$ activation of B
Repair of A $\Rightarrow$ deactivation of B

If PLC is not faulty:
Failure of A $\Rightarrow$ activation of B
Repair of A $\Rightarrow$ deactivation of B

If PLC is not faulty:
Failure of A $\Rightarrow$ deactivation of A, activation of B
Repair of A $\Rightarrow$ activation of A, deactivation of B

If PLC is not faulty:
Active comp. is faulty and inactive comp. is not $\Rightarrow$ deactivation of active comp., activation of inactive comp.

2. Failure of the reconfiguration strategy is integrated in the second model. It is assumed that reconfiguration is controlled by a PLC (Programmable Logic Controller) that may fail.

3. Deactivation/activation of component A after failure/repair is explicitly modeled in the third model while this component remained always active in the previous model.

4. Last, the fourth model uses the reconfiguration strategy resuming at the latest that has been defined at subsection 5.1 (the Moore machine M9 is the adaptation for two components of M2, depicted at Figure 9) while the strategy resuming at the earliest was selected for the third model.

These four reconfiguration strategies are formally described by the Moore machines associated to the switches M6 to M9.

6.1. Qualitative analysis

For dynamic systems, this analysis delivers the set of Minimal Cut Sequences (MCS), minimal set of minimal-length sequences of events that lead the system from its initial state to a failure state [20]. The minimality criterion is defined from a specific partial order relation between the cut sequences. This relation is based on a sequence inclusion relation (all events of the shortest sequence appear in the same order in the larger one) and an inclusion relation on the sets of faulty components at the end of the sequences. A detailed presentation of these relations can be found in [21]; computation of the set of MCS from a GBDMP model relies on a breadth-first exploration of the GBDMP state space and is also described in this reference. The results of this computation for the four models of Figure 14 are given at Table 5.

First, this table shows clearly that increasing the accuracy of modeling tends to enlarge the set of MCS. This

In this table, $f_A$, $f_B$ and $f_{PLC}$ means respectively failure of A, B and PLC (either in active mode or in standby mode), and $f'_B$ means on demand failure of B.
Table 5: Minimal Cut Sequences for the four models

<table>
<thead>
<tr>
<th>MCS</th>
<th>concerned model</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_A f_B$</td>
<td>1,2,3,4</td>
</tr>
<tr>
<td>$f_A f^1_B$</td>
<td></td>
</tr>
<tr>
<td>$f_B f_A$</td>
<td></td>
</tr>
<tr>
<td>$f_{PLC} f_A$</td>
<td>2,3,4</td>
</tr>
<tr>
<td>$f_{PLC} f_{PLC} f_B$</td>
<td>3,4</td>
</tr>
<tr>
<td>$f_A f_{PLC} f_{PLC} f_B$</td>
<td>4</td>
</tr>
</tbody>
</table>

is not really surprising but motivates an accurate modeling of reconfiguration strategies to forecast relevant set of MCS. The first three MCS are obtained with the four models and easy to interpret: the system fails totally when both components A and B have failed (B may have failed in active or standby mode or on demand). The fourth MCS is also easy understandable: the system fails when A fails after the PLC has failed because the service cannot be then transferred to B. The fifth and sixth MCS require a deeper reasoning because they are longer and include a repair event. For the fifth MCS, the system fails when B fails (last event of the sequence) while A has been previously repaired (third event of the sequence) because the PLC is faulty (second event) and therefore is not able to resume the service from B to A. A similar reasoning can be made for the sixth sequence. Comparison of the models 3 and 4 on the basis of this only analysis leads to favor the strategy resuming at the earliest (strategy selected for the model 3) because the sixth MCS is not possible with this strategy (the service is immediately switched to A once repaired). Nevertheless, this partial conclusion must be smoothed by the results of quantitative analysis of these models.

6.2. Quantitative analysis

This analysis will focus on the unavailability of the four models. Several contributions for scalable quantitative analysis techniques have been already published (see [22] and [23]). The curves of Figure 16 have been obtained by using the method described in [23] with the following numerical values: $\lambda = 10^{-3}h^{-1}$; $\lambda_S = 5.10^{-4}h^{-1}$; $\mu = 10^{-1}h^{-1}$ and $\gamma = 0.2$ (arbitrary values selected to accentuate the differences). In this approach, which was developed to increase scalability of quantitative analysis, a reduced-size Markov chain which includes only the most likely states is built from a high-level model (such as a GBDMP). Construction of this chain relies on [24] and a state relevance factor which represents the likelihood of a state and is computed from the transition rates of the SMP.

The unavailability of model 1 is the lowest one simply because the failure of the component that controls the reconfiguration (PLC) is not taken into account in this model. This model is too optimistic. The second model is less unavailable than the third and fourth ones because A is always active in this model and hence cannot fail on demand. Comparison of the models 3 and 4 leads this time to favor the strategy resuming at the latest (strategy selected for the model 4) because it minimizes the reconfiguration occurrences and consequently the risk of failure on demand. Hence, selecting a resuming strategy requires an expert decision on the bases of both qualitative and quantitative analyses.

7. Conclusions and Perspectives

This paper has presented the syntax and semantics of the GBDMP framework that has been developed to allow modeling explicitly and accurately reconfiguration strategies and considering the failures of the control of the reconfiguration. In our opinion, the main novelty of this framework is modeling of the reconfiguration strategies by Moore machines whose inputs depend on failure states of components of the process and the control. To ensure consistency of a GBDMP model that integrates a model of the structure of the system, in the form of fault tree enriched with switches, models of components, in the form of SMP, and switches whose behavior is described by Moore machines, five properties that must be satisfied by a well-formed model have been stated. An algorithm to analyze the evolutions of a GBDMP model in response to a sequence of spontaneous events has been developed and implemented in a prototype tool, too.
The treatment of representative examples has shown the benefits of this framework: different reconfiguration strategies can be precisely considered, the impact of failures of the control can be studied, components with several operation and failure modes can be introduced. Hence, the initial objective of this work has been met. Moreover, it has been pinpointed that extension to modeling and analysis of phased-mission systems is possible because phase change is a particular reconfiguration mechanism.

Nevertheless, construction of a GBDMP model is a difficult task that requires a lot of expertise. To overcome this issue, two solutions are possible. The first one consists in building libraries of SMP and Moore machines for typical components and reconfiguration strategies to allow modular construction by instantiation and assembly. The second one is a posteriori formal verification of dynamic properties of GBDMP models; model-checking of GBDMP models using the tool NuSMV is an on-going work in our laboratory.

8. Acknowledgment

This work has been funded by the French Investment of Future Program: Generic Components of Embedded Software as part of the CONNEXION project.

9. References