Non deterministic Repairable Fault Trees for computing optimal repair strategy

Marco Beccuti, Daniele Codetta-Raiteri,
Giuliana Franceschinis
Dip. di Informatica, Univ. del Piemonte Orientale
Via Bellini 25/G, Alessandria, Italy
{beccuti, raiteri, giuliana}@mfn.unipmn.it

Serge Haddad
LSV, ENS Cachan, CNRS
61, avenue du Président Wilson
Cachan, France
haddad@lsv.ens-cachan.fr

ABSTRACT

In this paper, the *Non deterministic Repairable Fault Tree* (NdRFT) formalism is proposed: it allows to model failure modes of complex systems as well as their repair processes. The originality of this formalism with respect to other Fault Tree extensions is that it allows to face repair strategies optimization problems: in an NdRFT model, the decision on whether to start or not a given repair action is non deterministic, so that all the possibilities are left open. The formalism is rather powerful allowing to specify which failure events are observable, whether local repair or global repair can be applied, and the resources needed to start a repair action. The optimal repair strategy can then be computed by solving an optimization problem on a *Markov Decision Process* (MDP) derived from the NdRFT. A software framework is proposed in order to perform in automatic way the derivation of an MDP from a NdRFT model, and to deal with the solution of the MDP.

Keywords

Fault Tree, Optimal repair strategy, Markov Decision Process, Markov Decision Petri Net

1. INTRODUCTION

The Fault Trees (FT) [21] are a well-known formalism for the evaluation of dependability of complex systems. They provide an intuitive representation of the system in terms of its faults, modeling how the combinations of failure events relative to the components of the system, can cause the failure of the sub-systems or of the whole system.

Many extensions of this formalism have been proposed in order to enhance the advantages of the FT for the design and the assessment of the systems (e.g. Dynamic FT [16], Parametric FT [5], etc.). Among these extensions, in [12] the Repairable FT (RFT) was presented in order to evaluate the effect of different repair policies on a repairable system.

In this paper, we present a new FT extension, called *Non deterministic Repairable Fault Tree* (NdRFT) which has been designed to define and solve repair strategy optimization problems: in an NdRFT model the possible repair strategies are not predefined; on the contrary, the best strategy, minimizing the failure probability of the global system, is automatically computed. This is done by defining the NdRFT semantics in terms of a *Markov Decision Process* (MDP), a formalism embedding non deterministic and probabilistic behavior [14, 17], and then solving the optimization problem using the methods available for MDPs.

The generation of the MDP is achieved by an intermediate translation of the NdRFT model into a *Markov Decision Petri Net* (MDPN) [3]: this allows to reuse the efficient algorithms devised to derive an MDP from an MDPN. Moreover a direct translation from NdRFT to MDP requires to implement a mechanism to combine the failure/repair events of all components into a single complex transition or action: this is already implemented for MDPN formalism.

The NdRFT formalism allows to express in an elegant way several possible start repair options based on: 1) the concept of “observability” of events (repair actions can only be triggered by observable failures), 2) the notion of local versus global repair action, 3) the notion of repair supervisor component, in case of global repair. Very few restrictions are imposed on the scope of repair actions (so that the repair of each basic component can start based on observations made on different failure events). The NdRFT formalism allows the modeler to express in a familiar language (NdRFT extends FT) the failure mode and the repair options in the system; in this way, he avoids to deal with a larger, unstructured and state-level MDP model that is instead derived from the NdRFT model.

The paper is structured as follows: Sec. 2 presents some related work about FT, tools for FT analysis, and RFT; in Sec. 3 we provide the formal definition of the NdRFT formalism; Sec. 4 explains how to derive from a NdRFT model, the corresponding MDPN; in Sec. 5 we present a software framework for the design and the solution of NdRFT models in order to compute the optimal repair strategy and the corresponding dependability of the system; finally in section 6, we present and analyze some experimentations.

2. RELATED WORK

2.1 Fault Trees

In the FT formalism, nodes can belong to one of these two categories: events and gates. Events concern the failure of components, subsystems or of the whole system. We can consider an event as a Boolean variable: it is initially $false$ and it becomes $true$ when the failure occurs.

An example is shown in Fig. 1.b. The events graphically represented as a rectangle with an attached circle are called *Basic Events* (BEs) and model the failure of the components of the system; such events are stochastic, so their occurrence is ruled by some probability distribution.

The events depicted simply by a rectangle represent the failure of subsystems; we call them *Internal Events* (IEs) and they are the output of a gate node. *Gates* are connected by means of arcs to several input events and to a unique output event; the effect of a gate is the propagation of the failure to its output event if a particular combination of its input events occurs. In the standard version of the FT formalism three types of gate are present and correspond to the AND, OR and *"K out of N"* Boolean functions.

Finally, we have a unique event called *Top Event* (TE), modeling the failure of the whole system. The FT incorporates a Boolean formula expressing the TE truth value as a function of its variables (BEs).

The analysis of an FT model returns several dependability measures such as the system reliability, the system minimal cut-sets, the criticality of each component [21]; in particular, the system *reliability* at time t is the probability that the system has been working in the time interval $(0, t)$. The most efficient way to perform the analysis of an FT, consists of generating the *Binary Decision Diagram* (BDD) [7] representing the same Boolean formula expressed by the FT: efficient algorithms allow to compute on the BDD the measures cited above [18].

2.2 Tools for FT analysis

Several software tools support the FT analysis. Some of them can deal with the repair, but they allow only to model the repair of single components: the repair process is triggered by the component failure and has effect only on the same component. For instance in the tool ASTRA [13] developed by the *European Commission Joint Research Centre* (JRC), one of the parameters of a BE is the time to repair the component whose failure is modeled by the same BE. In other tools, the time to repair a component is a random variable ruled by some distribution such as the negative exponential one. This is the case of the following tools where a repair rate can be associated with a BE: *Stars Studio* developed by JRC [13], *HIMAP* [15] by Iowa State University, *Relex* [25], *FTA-Pro* [24] by Dyadem, *FaultTree+* [26] by Isograph Software, *FTAnalyzer* [27] by Advanced Logistic Development (ALD).

The *SHARPE* tool [20] allows hierarchical modeling: the probability to occur of a BE can be set equal to some measure computed on another kind of model, for instance a *Continuous Time Markov Chain* (CTMC) [20]. In this way, the failure and repair mode of a component may be more complex than a simple transition from the working state to the failure state and vice-versa. In any case, the model representing the failure and repair mode of the component has to be manually drawn by the modeler. Hierarchical modeling is possible by means of the *HIMAP* tool as well [15].

A *Dynamic Fault Trees* (DFT) [1] is a particular extension of FT where dependencies between BEs can be set by means of the *dynamic gates*. The analysis of a DFT model can be performed by conversion into a *Continuous Time Markov Chain* (CTMC) [16] and is supported by the tool *Galileo* [22]. In [6], a DFT model can include the repair of components, and the analysis of the model is faced in by exploiting *Input-Output Interactive Markov Chains*, however each repair action still concerns a single component.

2.3 Repairable Fault Trees

In the literature, the *Repairable Fault Tree* (RFT) formalism [12] is the only extension of FT that allows to model the repair of a subsystem when triggered by a specific failure event. This means that the repair process concerns a set of components instead of a single one. Moreover, in the RFT formalism, the repair action is not simply ruled by a repair rate, but it is influenced by a *repair policy*: defining a repair policy in a RFT model means setting the parameters ruling each aspect of the repair process, such as the mean time to detect the failure, the mean time to repair a single component or a set of components, the number of repair facilities, the order of repair of the components. >From a RFT model we can compute the system *availability* at time t; this means the probability that the system is working at time t.

The RFT differs from the FT, for the introduction of a new primitive called *Repair Box* (RB) [12] allowing the model designer to represent the presence of a repair process involving a certain set of components called *basic coverage set* (Cov_{BE}) of the RB; such action is activated by the occurrence of a specific failure event called *trigger event* and concerning a component or a subsystem. The effect of the RB is setting the value of the BEs in its Cov_{BE} to *true* (working), if their current value is $false$ (failed). Such repair action is performed according to the repair policy associated with the RB node. Actually, the effect of the RB does not influence only the BEs in its basic coverage set, but also all the IEs whose value can be expressed by a Boolean function over a set of BEs including at least one BE in Cov_{BE}. In [12], the computation of the system *availability* from its RFT model, has been faced by converting the RFT model into a *Generalized Stochastic Petri Net* (GSPN).

In the RFT formalism, the repair policy (or strategy) is defined by the modeler and is associated with the RB primitive; therefore the only way for the modeler to identify the best policy, consists of analyzing the system according to several repair policies by constructing several RFT models, and by comparing the system availability values returned by the RFT models analysis. So, the RFT formalism does not allow to automatically determine the best repair policy.

The possibility to determine the optimal repair policy given all the repair possibilities, is an issue concerning several fields of engineering. So far, this problem has been usually faced in the literature in analytical ways, typically in form of operative research problems [8, 23, 19]. The NdRFT formalism presented in this paper, is an attempt to deal with the problem of optimal repair strategy, by building a graph based model having an intuitive notation and allowing to model several repair options together with the failure combinations in the system.

3. NON DETERMINISTIC RFT

3.1 NdRFT syntax

In this section the formal definition of the NdRFT is provided and commented through an example:

Definition 1 (Non deterministic Repairable FT). *An NdRFT is a five-tuple:*

$$\mathcal{S} = \langle \mathcal{E}, \mathcal{G}, \mathcal{A}, \mathcal{R}, res_0 \rangle$$

where:
$\mathcal{E}$ is the set of events.
$\mathcal{G}$ is the set of gates; $\mathcal{E} \cap \mathcal{G} = \emptyset$. A gate g has a type[1] *denoted $g.type \in \{$*`and`*,* `or`*$\}$.*
$\mathcal{A}$ is the set of arcs, a subset of $\mathcal{E} \times \mathcal{G} \cup \mathcal{G} \times \mathcal{E}$. For x belonging to $\mathcal{E} \cup \mathcal{G}$, we denote $x^\bullet \equiv \{y \mid (x,y) \in \mathcal{A}\}$ and ${}^\bullet x \equiv \{y \mid (y,x) \in \mathcal{A}\}$. $\mathcal{A}$ satisfies:

1. $\forall g \in \mathcal{G}, |g^\bullet| = 1$ and $\forall e \in \mathcal{E}, |{}^\bullet e| \leq 1$

2. There is exactly one event, denoted $\top$ and called Top Event, s.t. $\top^\bullet = \emptyset$; all other events satisfy $|e^\bullet| \geq 1$

3. The set of events can be partitioned into basic *events $\underline{\mathcal{E}} \equiv \{e \mid {}^\bullet e = \emptyset\}$ and* internal *events $\overline{\mathcal{E}} \equiv \{e \mid {}^\bullet e \neq \emptyset\}$*

4. The (directed) graph induced by $\mathcal{A}$ is acyclic.

$\mathcal{R}$ is a finite set of repair resource types; $res_0 \in Bag(\mathcal{R})$ is the multiset of available resources, where $Bag(\mathcal{R})$ is a generalization of a set, so that it can contain several occurrences of the same element.
Each event is associated with a set of attributes, related to its failure probability and to the definition of the applicable repair actions. Any event e is either observable *($e.obs =$* `true`*) or* non observable *($e.obs =$* `false`*); only observable events can trigger a repair action.*
Moreover, each BE e has the following additional attributes:

1. a fault probability denoted $e.fprob$ ranging over $[0,1]$;

2. a repair attribute denoted $e.rep \in \{$`true`*,* `false`*$\}$ indicating if the event is repairable or not; if $e.rep =$* `true`*, e has also a repair probability denoted $e.rprob \in [0,1]$ and a bag of resources denoted $e.res \in Bag(\mathcal{R})$.*

Finally, each internal observable event e has the following additional attributes:

1. a set of BEs that should be repaired in case of e failure, denoted $e.torep$ such that $e' \in e.torep \Rightarrow e'.rep = true$, moreover there is a path from e to e' according to $\mathcal{A}$;

2. a repair strategy denoted $e.str \in \{$`global`*,* `local`*$\}$. When the strategy associated with an event is global , it also has a repair probability denoted $e.rprob \in [0,1]$ and a bag of resources denoted $e.res \in Bag(\mathcal{R})$.*

Let us comment the above definition by means of the example of Fig. 1 (whose meaning will be explained in Sec. 6): in the picture the events are depicted in a different way according to their *obs* and *rep* attribute values. Down arrows, labeled with a number, next to BEs indicate their failure probabilities; up arrows, labeled with a number, next to repairable BEs or to internal events with global strategy, indicate the repair probability. Basic events $A3$ and $P3$ in the example are not repairable. In the NdRFT formalism, the assumption of discrete time holds: the time to fail (repair) a component is ruled by the geometric distribution having as parameter the failure (repair) probability (see section 3.3).

The failure of an observable and repairable BE e (e.g. $A1$) can immediately trigger a repair action of the component, while the repair of a non observable (but repairable) event e (e.g. $A2$) can only be triggered by an observable internal event connected to e (for $A2$ it can be $U2$ or TE). Intuitively, observability is related to the possibility of detecting a failure. In the example of Fig. 1 we have only one type of resource and each repair action requires only one resource (observe that any local repair action, including the one triggered by the TE, requires one resource for *each* BE to be repaired).

The event attribute *repair strategy* defines the granularity of the repair process triggered by the occurrence of the (internal) event e: if the repair strategy is *global* (as for $U2$ in the example), *all* the repairable basic components in *e.torep* ($A2$ and $P2$ in the example) are repaired simultaneously and brought back to the working state when the global repair process terminates. This means that a global repair process is a unique repair process (e.g. representing the substitution of a down server with a new server: all components are substituted at once); while a global repair action is ongoing, the basic components in *e.torep* cannot be simultaneously involved in any other repair action (global or local). If instead the repair policy is *local* (as for TE in the example), for each repairable BE component in *e.torep*, it is possible to decide to repair or not such component; moreover the single components repair may not start simultaneously (e.g. because there are not enough resources). A BE can appear in the *torep* set of several internal events; for example $A2$ and $P2$ are in the *torep* set of both $U2$ and TE: when a failure has occurred for only one among the two BEs, the local strategy could be more appropriate, but it can be activated only if TE has occurred already. Otherwise, if both $A2$ and $P2$ failure has occurred, the global repair of $U2$ may be more convenient. Observe that given the example NdRFT structure, $U2$ can immediately witness the failure of one or both events $A2$ and $P2$, and trigger the substitution of both.

3.2 MDP semantics of NdRFT

MDP definition. A (discrete time and finite) MDP is a dynamic system where the transitions between states follow a two-step process. First, one non deterministically selects an action inside the subset of enabled actions. Then one samples the new state with respect to a probability distribution depending on the current state and the selected action. The non deterministic step represents a decision taken by a controller in order to manage the system, or a behavior triggered by the environment that the system cannot control. Our approach is based on the former interpretation. The probabilistic step takes into account that the effect of an action statistically depends on non modeled (or unknown) parameters.

In order to formally define the objective to optimize, one associates a reward with any state and selected action (the reward can also be interpreted as a cost). The following definition formalizes these concepts.

Definition 2 (Markov Decision Process, MDP:). *An MDP $\mathcal{M}$ is a four-tuple $\mathcal{M} = \langle S, A, p, r \rangle$ where:*

1. S is a finite set of states,

2. A is a finite set of actions defined as $\bigcup_{s \in S} A_s$ where A_s is the set of enabled actions in state s,

3. $\forall s \in S, \forall a \in A_s, p(\cdot|s,a)$ is a (transition) probability distribution over S such that $p(s'|s,a)$ is the probability to reach s' from s by triggering action a,

4. $\forall s \in S, \forall a \in A_s, r(s,a) \in \mathbb{R}$ is the reward associated with state s and action a.

Once an action choice is fixed, the MDP behaves like a

[1]Since the proposed optimization method is based on the state space, other gate types could easily be considered, including dynamic ones: in this paper only and/or gates are considered for the sake of space.

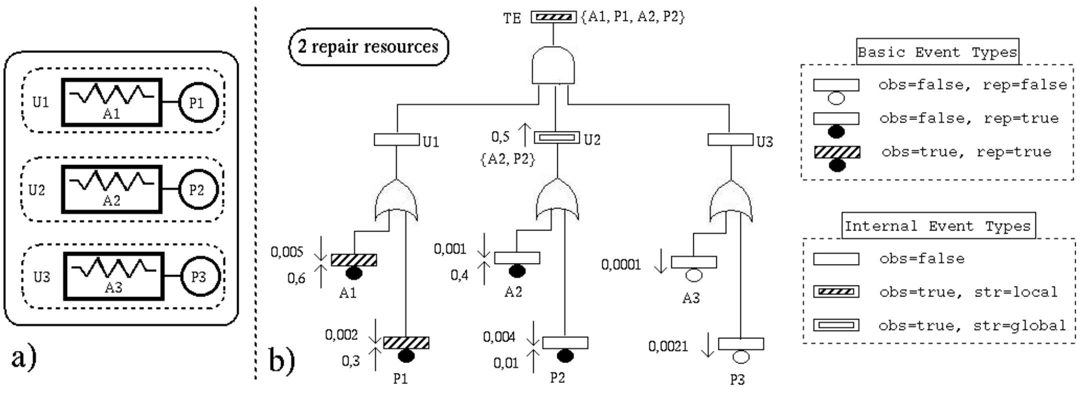

Figure 1: a) The block scheme of the AHRS. b) The NdRFT model of the AHRS.

Markov chain and different global measures on the random path can be defined as for example the (discounted) sum of rewards or the average of the rewards. The goal of the analysis is computing the optimal value of the measure, and when possible, computing the associated strategy. In finite MDPs, efficient solution techniques have been developed to this purpose [17] and different tools are based on this theory (see for instance the experiment section).

NdRFT semantics. The semantics of an FT is simply a Boolean formula expressing the TE truth value as a function of the BEs truth value; the possible (minimal) failure configuration leading to the TE and their occurrence probability (at time t) can be efficiently computed using a BDD [18] representation of the FT, without need to develop its dynamic failure behavior. NdRFT semantics instead (as well as RFT one) requires to explicitly expand and analyze the dynamic behavior of the model since the introduction of the repair processes adds the possibility for events to switch between the up and down state several times within a given observation period.

In this paragraph, we will define precisely the dynamic behavior of an NdRFT, which can be described by an MDP. Let us first define the MDP states:

Definition 3 (MDP$_{NdRFT}$ state). *A state ρ of the MDP corresponding to a given NdRFT is a tuple:*

$$\rho = \langle \{st_e\}_{e \in \mathcal{E}}, \{sup_e\}_{e \in \mathcal{E}} \rangle$$

where $st_e = \{Up, Down, LocRep, GlobRep\}$ represents the state of the component/subsystem whose failure corresponds to the event e, with $e = false \Leftrightarrow st_e = Up$ otherwise $e = true \Leftrightarrow st_e = Down \vee st_e = LocRep \vee st_e = GlobRep$. Only BEs can be in repair state, and for these events sup_e represents the supervisor of the repair process: this is the basic e itself if the repair action is local, while in case of global repair the supervisor is the internal event that triggered it. Observe that $st_e \in \{Up, Down\} \Leftrightarrow sup_e = NULL$. The state of the internal events in ρ can only be in $\{Up, Down\}$ and is derived from the state of the BEs and the FT structure. The initial state ρ_0 is: $\rho_0 : \forall e \in \mathcal{E}, st_e^0 = Up \wedge sup_e = NULL$.

For each state ρ, reachable from the initial state ρ_0 (following the transition rules described next),the multiset of busy resources is defined as: $res_\rho = \sum_{e \in sup(\mathcal{E})} e.res$.

The dynamic of the MDP corresponding to an NdRFT, is defined in terms of two steps: a non deterministic one (selecting the subset of repair actions that should start) and a probabilistic one (probabilistically choosing the newly occurred failure events and which ongoing repair actions have completed). Of course at each time the following condition must be verified: $res_\rho \subseteq res_0$, i.e. $\forall r \in \mathcal{R}, res_\rho(r) \leq res_0(r)$, where $res_i(r)$ is the multiplicity of r in res_i.

The state change at each step is described hereafter:

Non Deterministic step: MDP actions. This step comprises a (possibly empty) set of repair start decisions for BEs. The repair must be triggered by (basic or internal) observable events that are in state $Down$. For each repair start decision, the supervisor of the involved event must be specified: it is the event itself in case of local repair, while it is the internal trigger event for global repair.

The conditions for the two types of state change are:

$st_e : Down \rightarrow LocRep$: (1) $e.obs = true \;\wedge\; e.rep = true$ or (2) $\exists e' : e \in e'.torep, st_{e'} = Down, e'.obs = true, e'.str = local$; in both cases $sup_e = e$.

$st_e : Down \rightarrow GlobRep$: $\exists e' : st_{e'} = Down, e'.obs = true, e'.str = global$; in this case the state change must happen simultaneously for every $e \in e'.torep$ and $sup_e = e'$.

Of course the set of chosen repair actions leading to a new state ρ must be consistent with the requirement $res_\rho \subseteq res_0$.

The possible actions A_ρ available in state ρ of the MDP are thus all the legal repair start decision sets satisfying the conditions and the resource constraints described above.

Probabilistic step. In this step the possible state changes for each BE e are:

$st_e : Up \rightarrow Down$: with probability $e.fprob$ (or remain in the Up state with probability $1 - e.fprob$);

$st_e : LocRep \rightarrow Up$: with probability $1 - e.rprob$ (or remain in the $LocRep$ state with probability $e.rprob$);

$st_e : GlobRep \rightarrow Up$: with probability $1 - sup(e).rprob$ (or remain in the $LocRep$ state with probability $sup(e).rprob$): this state change must happen simultaneously for all $e'' \in sup(e).torep$ (it is a single probabilistic choice with synchronous effect on all events involved in the repair action). For any event e that returns to the Up state, sup_e is reset to $NULL$. Hence the probabilistic choice that follows a given non deterministic action in the MDP, corresponds to a probabilistic step as described above: the probability of each step is obtained as the product of the probabilities of each single event state transition.

As already remarked, the states of the internal events are derived from those of the BEs using the FT structure.

This completes the definition of the MDP underlying a given NdRFT. The optimization problem has the following goal: minimizing the probability (at time t or in steady state) of being in a state where the TE failure has occurred.

In practice, the computation of the optimal strategy requires three steps: (1) generation of the MDP from the NdRFT, (2) analysis of the MDP, (3) presentation of the results in a form that is understandable for the designer.

These steps can be automatized. The first step can be implemented in two ways: defining an algorithm that generates the set of reachable states, the corresponding non deterministic actions and consequent probabilistic state change, or translating the NdRFT in an intermediate model for which the above tasks have already been defined and implemented. In this paper we propose to use the second approach and provide an algorithm for translating an NdRFT into a *Markov Decision Petri Net* (MDPN) [3]. From the MDPN model an MDP can be automatically derived. This allows to reuse the efficient algorithms devised to derive an MDP from an MDPN Observe that a direct translation from NdRFT to MDP requires to implement a mechanism to combine the failure/repair events of all components into a single complex transition or action, that it is already implemented for MDPN formalism.

3.3 Discussion

The NdRFT model is a discrete time one. This can be justified by the fact that faults in plants are often detected at the time a sampling is performed through some sensor: sampling is usually done periodically according to a synchronous schema. Due to the discrete time assumption, the specification of the failure and repair process of each (basic) *repairable* component x is given by probability $P_{Failure}(x)$ and $P_{Repair}(x)$. $P_{Failure}(x)$ (resp. $P_{Repair}(x)$) represents the probability that a failure (resp. the end of the repair) occurs at any (discrete) time step provided the corresponding component is up (resp. is down and under repair). As a consequence, the time to failure of a component, and its repair time have geometric distribution:
$P(TtF_e = k) = (1 - P_{Failure}(e))^{k-1} P_{Failure}(e)$
$P(repTime_e = k) = (1 - P_{Repair}(e))^{k-1} P_{repair}(e)$

In NdRFT the repair policy is not completely specified (instead, this is the case for RFT): the choice to repair or not a repairable components fault is non deterministic. This leads to an MDP semantics: as a consequence, we can compute the optimal repair strategy minimizing the failure probability of the global system. Observe that even without taking into account the cost of repair, finding the optimal strategy is not trivial, since we account for limited repair resources (each repair action can be associated with a multiset of required resources to complete it).

In the NdRFT we can model processes where the components or the subsystems under repair return available as soon as possible (maybe in a degraded state) without waiting the repair of all its down BE components. Moreover, the notion of observability allows to specify when a fault can be detected, and hence when the corresponding repair activity can start (this generalizes the notion of *trigger event*).

Finally repair actions may involve common components: this choice increases the flexibility in the choice among the possible repair strategies that may be pursued, still allowing a simple and clean semantics based on the notions of observability and of global vs. local repair strategy.

4. TRANSLATING NDRFT INTO MDPN

In this section we are going to describe how to obtain from an NdRFT model the corresponding MDPN model. An informal introduction to the MDPN formalism is provided first, then the pattern-based translation algorithm is presented. The proof of the translation correctness is provided in [4].

The generation of the MDP from the MDPN model can be performed as described in [3]. The MDP obtained is solved in order to find the optimal repair strategy (at finite horizon t or in steady state, as appropriate) and the corresponding Top Event failure probability.

A brief introduction to MDPNs. MDPNs were first introduced in [3] as high level models to specify the behavior of an MDP. The main features of the high level formalism are the possibility to specify the general behavior as a composition of the behavior of several components (some of which are controllable); moreover each MDP non deterministic or probabilistic transition can be composed by a set of non deterministic or probabilistic steps, each one involving a subset of components.

An MDPN model is composed of two parts, both specified using the PN formalism: the PN^{nd} subnet and the PN^{pr} subnet (describing the non deterministic (ND) and probabilistic (PR) behavior respectively); the two subnets share the set of places, while having disjoint transition sets. In both subnets the transitions are partitioned into “run” and “stop” subsets, and each transition has an associated set of components involved in its firing. Transitions in PN^{pr} have a “weight” attribute, used to compute the probability of each firing sequence. Run transition firings represent intermediate steps in a ND/PR transition at the MDP level, while Stop transitions represent the final step in a ND/PR transition, for all components involved in it. An MDPN model behavior alternates between ND transition sequences and PR transition sequences, initially starting from a ND state. The PR sequences are determined according to the PN^{pr} structure, and include exactly one stop transition for each component; the ND sequences are determined by the PN^{nd} structure, and include exactly one stop transition for each controllable component plus a stop “global” transition. The generation of the MDP corresponding to a given MDPN has been described in [3]: it consists of (1) a composition step, merging the two subnets in a single net, (2) the generation of the reachability graph RG of the composed net, (3) two reduction steps transforming each PR and ND sequence in the RG into a single MDP transition.

In the next subsections a pattern based approach to generate a MDPN mimicking the dynamic behavior of an NdRFT is presented. We introduce the set of repairable basic components: $\underline{\mathcal{E}}_R = \{e \in \underline{\mathcal{E}} | e.repair = true\}$, and the set $Comp^{pr}$ of components of the MDPN and the subset $Comp^{nd}$ of controllable components: $Comp^{pr} = \underline{\mathcal{E}}$, $Comp^{nd} = \underline{\mathcal{E}}_R$.

The PN^{pr} and the PN^{nd} are obtained directly from the NdRFT model using a pattern-based approach. We illustrate the method informally describing the basic patterns, and how to instantiate and compose them.

4.1 Generating the PR subnet

Fig. 2 shows how each BE can be translated in a PN^{pr}

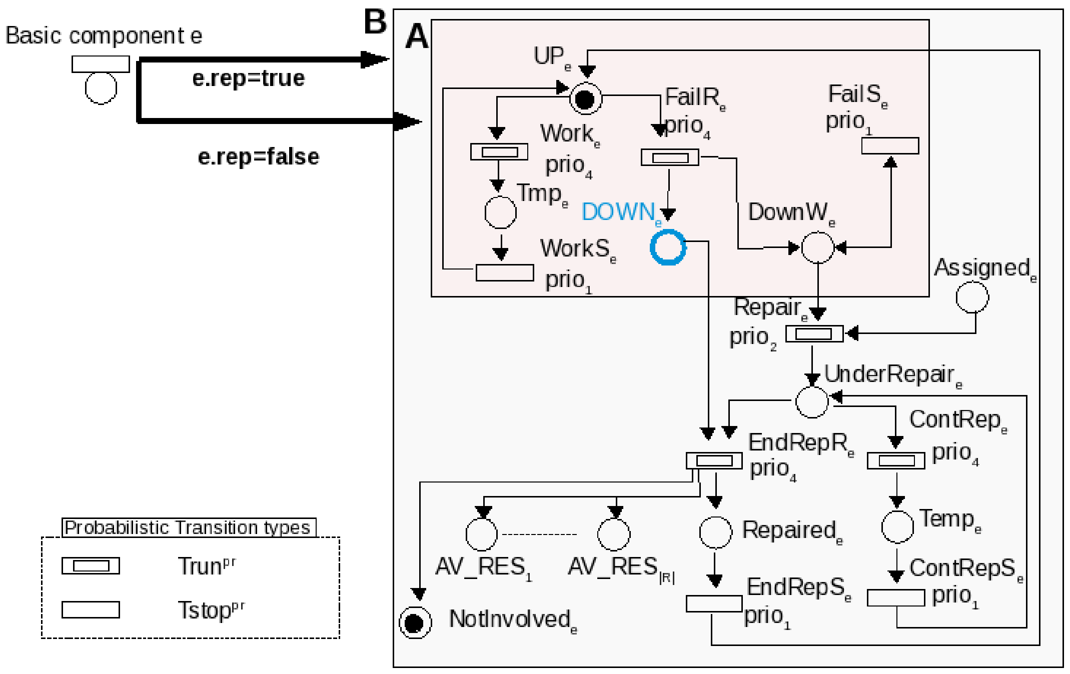

Figure 2: Conversion of the NdRFT BEs into submodels of PN^{pr} of an MDPN.

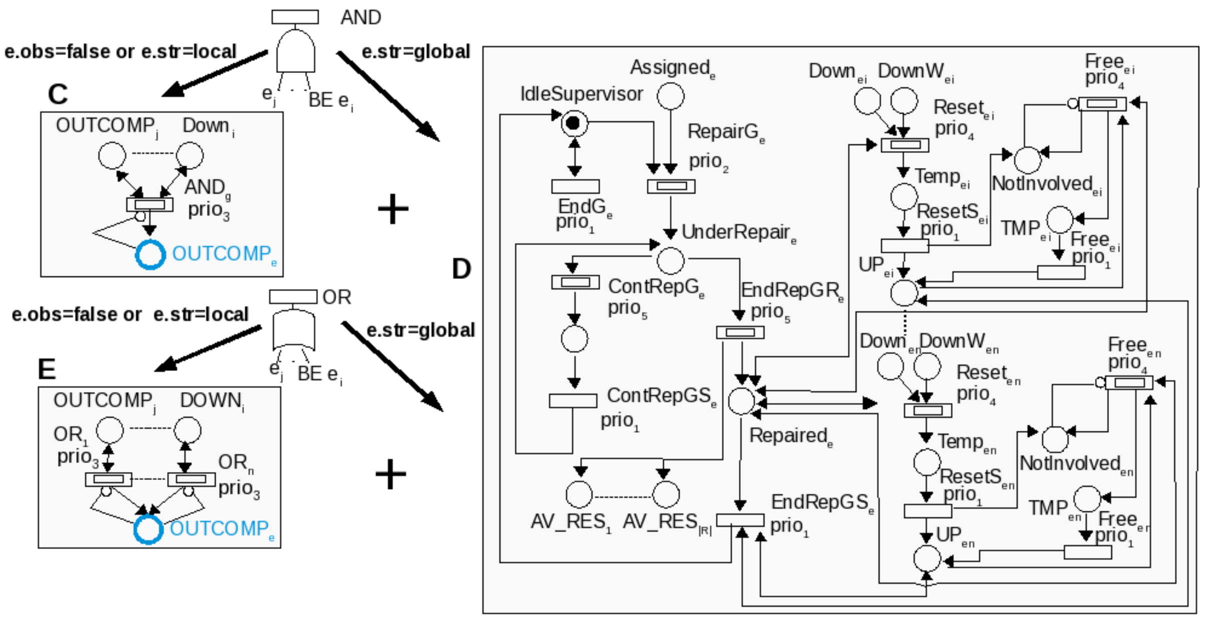

Figure 3: Conversion of the NdRFT AND/OR gate plus its output event into submodels of PN^{pr} of an MDPN.

submodel according to their *rep* attribute: each non repairable event is translated into subnet **A** while each repairable event is translated into subnet **B**. It is easy to recognize the places that model the state of each (basic) event e labeled UP_e, $DOWN_e$ and $UnderRepair_e$. Run and Stop transitions have different icons, so that they can be easily distinguished. At each probabilistic step an Up component can either remain Up (stop transition $Work_e$) or go $Down$ (sequence $FailR_e$, $FailS_e$). A $Down$ component can either remain $Down$ (stop transition $FailS_e$) or start its repair (run transition $Repair$, either followed by the sequence $ContRep_e$ and $ContRepS_e$, meaning that the repair has not completed in the current time unit, or by the sequence $EndRepR_e$, $EndRepS_e$ if the repair completes). Place $Assign_e$ is set by the PN^{nd} when a decision to repair e is taken. Places AV_RES_i represent the resources, and they become available as the repair ends. The $rprob$ and $fprob$ attributes associated with the events are used to properly weight the transitions representing failure and end/continuation of repair actions.

The conversion rule for an AND gate g and its output event e is shown in Fig. 3. We emphasize that the two PN models in Fig. 3 are templates that must be instantiated according to the set of input events of each gate. Subnets **C** and **E** simply model the propagation of the faults from the input events of the gate to its output event. Internal events that are not observable or have local repair strategy are translated into these simple subnets. Those with a global repair strategy have an additional subnet **D** (common to both gate types) shown on the right: this subnet represents the corresponding global repair process (see details in [4]).

The algorithm visits all the events in the NdRFT and generates for each of them an appropriate PN submodel (the selection of the appropriate PN submodel follows the indications depicted in the template figures). Finally all submodels are composed by merging the places with equal label, leading to the whole probabilistic subnet of the MDPN.

4.2 Generation of the ND subnet

The corresponding PN^{nd} is built instead from the template subnets depicted in Fig. 4 and 5. The basic idea is that the PN^{nd} submodel must decide whether a repair action must be started for each down BE. Firing of stop transition $NoAssign_e$ means that a non repair decision has been taken for event e, while firing of stop transition $Assign_e$ corresponds to the opposite decision: observe that the second decision can be taken only if the needed resources are available (input places AV_RES_i and the event is not involved in some global repair (input place $NotInvolved_e$). Start of local repair actions triggered by observable BEs are represented by subnet G, Start of local repair actions triggered by observable internal events are represented by subnet L, start of global repair actions are represented by subnet I. Subnet H instead is needed for technical reasons: it is used to "refresh" the state of the internal events which must be recomputed at the end of each probabilistic step (after all fail/repair steps have been taken for BEs). Again the final PN^{nd} submodel is obtained by properly composing the subnets generated for each event in the NdRFT.

Finally in order to analyze the MDPN model, one has to define its reward functions. They are defined as follows: $rs(TE) = -1$ otherwise 0; $\forall t \in T^{nd}, rt(t) = 0$; $r_g = sum(rs, rt)$.

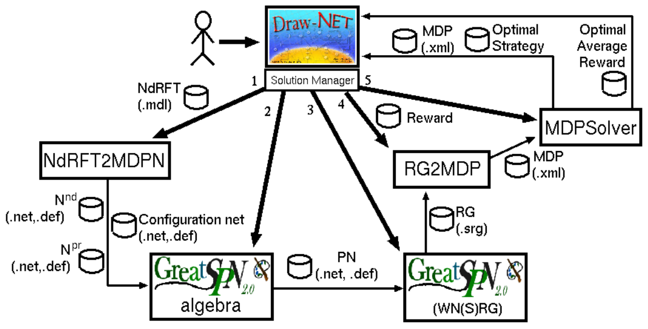

Figure 7: Framework architecture.

This means that a negative reward (corresponding to a penalty) is associated to each state where the TE has failed. All other states and all actions have reward of 0. This means that every time unit spent in a state with a TE failure gives us a penalty of -1. The optimization problem hence consists in finding the strategy that maximizes the reward (i.e. that minimizes the penalty).

More complex reward structures can be naturally devised to take into account the cost of repair actions, as well as the penalties due to the fact that the system is in a degraded state (the system is up but some subsystem is down, e.g. corresponding to a system with degraded performance).

5. FRAMEWORK ARCHITECTURE

The architecture of our framework for the NdRFT design and solution, is depicted in Fig. 7 and extends the one presented in [2], by introducing the new module *NdRFT2MDPN* able to convert an NdRFT model into MPDN, according to the conversion rules defined in Sec. 4. The solution process of an NdRFT model comprises five steps:

1. The NdRFT model drawn by the user by means of *Draw-Net* [11], is stored in a XML file (.mdl) and becomes the input of *NdRFT2MDPN*. The resulting MDPN model consists of two separate *Petri Nets* (PN): the probabilistic PN (N^{pr}) and the non deterministic PN (N^{nd}); each of these nets is stored in a couple of files (.net, .def) according to the *GreatSPN* [10] file format.
2. The N^{pr} and the N^{nd} models are composed by place merging; this is done by means of the *algebra* tool [10]. The result of this step is a Petri Net (PN).
3. The PN is the input of *WN(S)RG* generating the *Reachability Graph* (RG) [9]. The resulting graph is stored in a specific file (.srg).
4. From the graph obtained in step 3, an MDP is derived by means of the *RG2MDP* converter.
5. The obtained MDP is stored in an XML file which is in turn processed by the *MDPSolver* producing the *optimal repair strategy*. According to such strategy the system unavailability can be computed. Both results can be visualized by *Draw-Net*.

6. EXPERIMENT RESULTS

The example we report is inspired to the *Active Heat Rejection System* (AHRS) presented in [1]. The block scheme of our version of the AHRS's architecture is depicted in Fig. 1.a: the system is composed by three redundant ther-

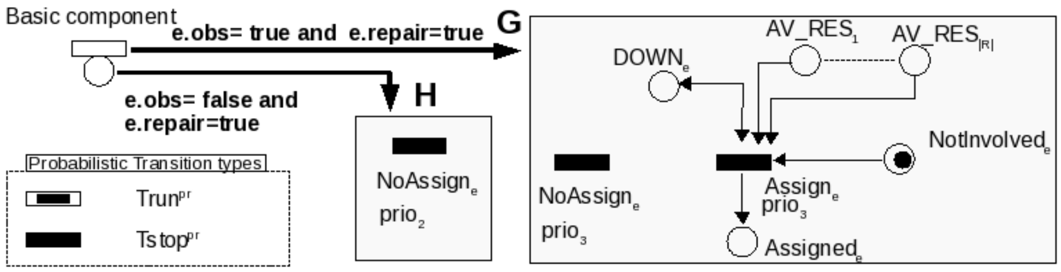

Figure 4: Conversion of the NdRFT BEs into submodels of PN^{nd} of an MDPN.

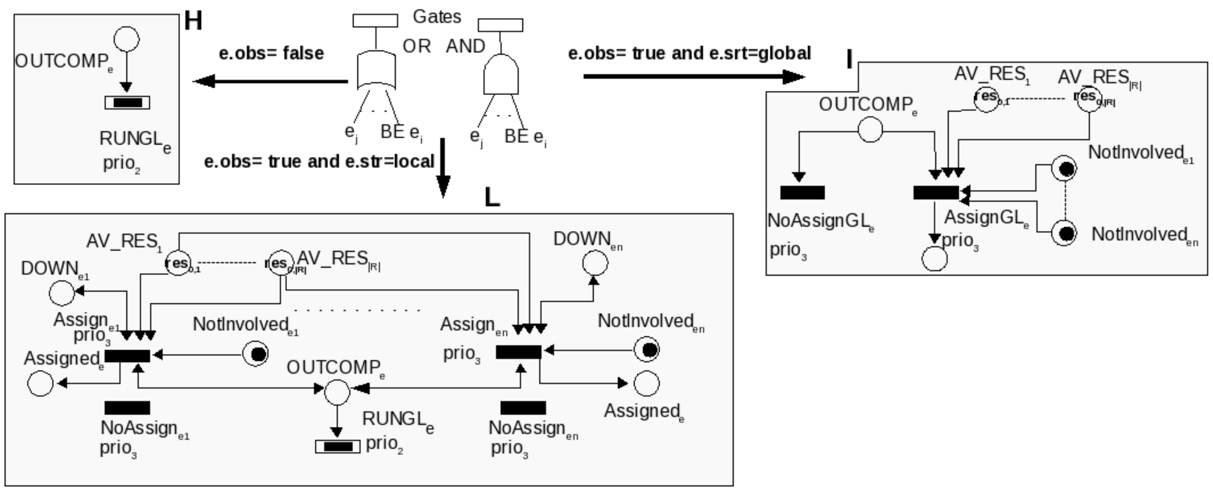

Figure 5: Conversion of the NdRFT gate into submodels of PN^{nd} of an MDPN.

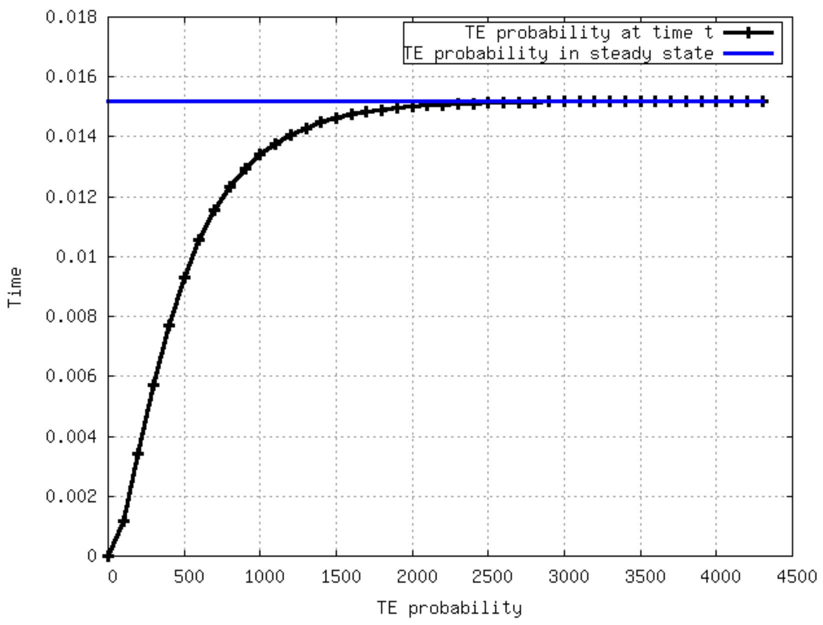

Figure 6: TE probability at time t increasing the t values ($0 \leq t \leq 4500$)

mal rejection units $U1$, $U2$ and $U3$. $U1$ is composed by the heat source $A1$ and the power source $P1$. Similarly, $U2$ is composed by $A2$ and $P2$, while $U3$ by $A3$ and $P3$.

Fig. 1.b shows the NdRFT model for the AHRS system; the failure probability ($\downarrow$) and the repair probability ($\uparrow$) of each basic component are shown in the same figure. The unit $U1$ fails if its heat source $A1$ is failed or if its power source $P1$ is failed. Similarly, the failure of $U2$ and $U3$ is due to the failure of their respective heat source or power source. The failure of the whole system (TE) occurs if all the thermal rejection units are failed.

The NdRFT model in Fig. 1.b shows that in our version of the AHRS, several components are repairable ($A1$, $P1$, $A2$, $P2$), whereas their failure can be observable or not. Two repair processes can be activated: 1) a global repair process in case of failure of $U2$ and involving the components $A2$ and $P2$; 2) a local repair process in case of the system failure (TE) and involving the components $A1$, $P1$, $A2$ and $P2$. In case of global repair, one repair resource is used to repair the subsystem; in case of local repair instead, one resource has to be dedicated to the repair of each component of the system. We suppose that in our case study, two repair resources are available (Fig. 1). One resource is used for the global repair of $U2$: while such repair process is running, the local repair of the system (TE) may start but in this case, it can exploit only one resource because the other one is already is used in the global repair of $U2$. So only one component ($A1$ or $P1$) could be locally repaired during the global repair of $U2$. If instead the local repair of the system starts while the global repair of $U2$ is not running, then the local repair can exploit both resources and two components among $A1$, $P1$, $A2$, $P2$ can be repaired at the same time. In this case, during the local repair of the system, the global repair of $U2$ can not run since all the resources are already in use.

The RG of the MDPN model obtained by the NdRFT in Fig. 1.b has 11.515 states; while the underlying MDP has 389 states. This difference in terms of number of states between the RG of the MDPN[2] and the obtained MDP is due to the fact that the MDPN formalism gives a macroscopic view of probabilistic and non deterministic behaviors of the system. In other words, at MDPN level, complex non deterministic and probabilistic behaviors are expressed as a composition of simpler non deterministic or probabilistic steps, that will be reduced to a single step in the final MDP.

Since the non repairable components ($A3$ and $P3$) cannot induce directly the failure of the global system, we can compute the average reward and the optimal strategy of the underlying MDP at infinite horizon. Observe that defining the optimal strategy for this model is not trivial: for instance when all the basic events are down then the optimal strategy suggests us to repair $P1$ with a local repair action, while $A2, P2$ with a global repair action. This is justified by the fact that the global repair action of $A2, P2$ needed only one resource. Instead when $A1, P1, P2$ and $P3$ are down, it suggests to repair $P1$ and $P2$ with a local repair action. The choice to repair locally $P2$ is justified by the fact that in this case the probability to repair the component $(1 - P2.rprob)$ in one time unit is greater than that associated with the global repair action $(1 - U2.rprob)$.

Moreover we have computed the TE probability in steady state, solving the DTMC obtained from the underlying MDP fixing the action to take in every state according to the performed optimal strategy. In particular we have obtained that the TE probability of this model in steady state is 0.0151943. We have also studied the TE probability at time t, so that we have observed that this probability converges to the steady state probability. TE probability at time 4300 is equal to TE probability in steady state as shown in Fig. 6.

[2]We recall that the RG of the MDPN model is used in the reduction step for obtaining the MDP as described in [3]

Table 1: Experiments increasing the example size

	RG		RRG	MDP	
Com.	St.	Time	St.	St.	Time
1,1,1	11.515	1s	5.262	389	0
2,1,1	50.844	7s	21.094	937	6s
2,2,1	921.354	167s	401.350	7.754.	1.630s
2,2,2	16.841.490	≈23h	6.048.310	32.558	≈4h
Using priorities among the transitions					
1,1,1	3.189	0	1.572	389	0
2,1,1	35.555	8s	11.581	937	4s
2,2,1	453.257	145s	147.716	7.754	1.614s
2,2,2	2.919.999	≈2h	1.048.310	32.558	7.006s

Finally the Tab. 1 shows some experiments performed increasing the dimension of our example. Practically we have replicated the subtrees of the NdRFT model in Fig. 1.b. For instance, in Tab. 1, $2, 2, 2$ means that we have duplicated the subtrees rooted in $U1$, $U2$, $U3$ respectively, while $1, 1, 2$ means that we have duplicated only the subtree of $U3$.

The computation has been performed with an INTEL Centrino DUO 2.7 of 2Gb memory capacity. In particular the first column shows the model complexity, the second and the third one the RG number of states and its computation time, the fourth the RRG number of states, and the last two columns the MDP number of states and its generation and solution time.

These results show that state space grows very fast (the state space explosion problem), so that the model becomes quickly intractable. A further reduction of the number of states for this model can be achieved associating different priorities with the system transitions such that the number of possible interleavings of the non deterministic/probabilistic actions in each path are reduced (see the results in second part of Tab. 1). It is important to observe that a different priority level can be set up only among independent actions; in fact if the actions are not independent then all the priority can constrain the set of policies to be considered (and may exclude the optimal one).

Another possible way to mitigate the state space explosion problem consists in translating the NdRFT model into a *Markov Decision Well-formed Net model* (MDWN) [3]. From a MDWN, a reduced MDP can be obtained, and provides the optimal strategy equivalent to the one given by the not reduced MDP.

7. CONCLUSION AND FUTURE WORK

We have defined a new FT extension called NdRFT that allows to model failure modes of complex systems as well as their repair processes. The originality of this formalism with respect to other proposals is that it allows to manage repair strategies optimization problems. This is done by defining the NdRFT semantics in terms of an MDP and then solving the optimization problem using the techniques available for MDPs. The generation of the MDP is achieved by an intermediate translation of the NdRFT model into an MDPN, so that we can reuse the efficient algorithms devised to derive an MDP from an MDPN. We have also highlighted that NdRFT allows to express in an elegant way several possible repair start options based on the following concepts: observability of events, the notion of local versus global repair action, the notion of repair supervisor component in case of global repair.

A possible future work is extending the NdRFT, so that the modeler can directly define more complex reward functions, for instance considering the cost of repair actions, or the penalties due to the fact that the system is in a degraded state (the system is up, but some subsystem is down, e.g. corresponding to a system with degraded performance).

Another future work in order to mitigate the well-known state space explosion problem in the final MDP, consists in translating the NdRFT model into a MDWN instead of an MDPN. In fact for MDWN, an efficient analysis technique taking advantage from the intrinsic symmetries of the system, is developed: from an MDWN it is possible to derive a *Symbolic Reachability Graph* [9], and from it a reduced MDP can be obtained. This allows a computational cost reduction, but the optimal strategy computed on the reduced MDP is equivalent to the one computed on the ordinary MDP. This possibility is useful when the system is characterized by symmetries and redundancies in its structure.

Observe that the NdRFT formalism could be extended by considering *dynamic gates* [16], which allow to express functional and temporal dependencies among component failures, as well as repair resources preemption.

8. ACKNOWLEDGMENTS

The activity of M. Beccuti, D. Codetta-Raiteri and G. Franceschinis has been partially supported the EU-Project CRUTIAL IST-2004-27513.

The activity of S. Haddad has been partially supported by ANR project Checkbound ANR-06-SETI-002.

The activity of all the authors has been partially supported by the bilateral project Galileo.

9. REFERENCES

[1] T. Assaf and J. B. Dugan. Diagnostic Expert Systems from Dynamic Fault Trees. In *Annual Reliability and Maintainability Symposium 2004 Proceedings*, pages 444–450, Los Angeles, CA USA, January 2004.

[2] M. Beccuti, D. Codetta-Raiteri, G. Franceschinis, and S. Haddad. A framework to design and solve Markov Decision Well-formed Net models. In *Int. Conf. on Quantitative Evaluation of Systems*, pages 165–166, Edinburgh, Scotland, UK, September 2007.

[3] M. Beccuti, G. Franceschinis, and S. Haddad. Markov Decision Petri Net and Markov Decision Well-Formed Net Formalisms. *Lecture Notes in Computer Science*, 4546:43–62, 2007.

[4] M. Beccuti, G. Franceschinis, and S. Haddad. Non deterministic Repairable Fault Trees for computing optimal repair strategy. Technical Report TR-INF-2008-07-05, Dip. di Informatica, Univ. del Piemonte Orientale, 2008.

[5] A. Bobbio, G. Franceschinis, R. Gaeta, and G. Portinale. Parametric fault tree for the dependability analysis of redundant systems and its high-level Petri net semantics. *IEEE Transactions on Software Engineering*, 29(3):270–287, March 2003.

[6] H. Boudali, P. Crouzen, and M. Stoelinga. Dynamic Fault Tree Analysis Using Input/Output Interactive Markov Chains. In *Int. Conf. on Dependable Systems and Networks*, pages 708–717, June 2007.

[7] R. E. Bryant. Symbolic Boolean manipulation with Ordered Binary Decision Diagrams. *ACM Computing Surveys*, 24:293–318, 1992.

[8] I. Castroa and E. Sanjuan. An optimal repair policy for systems with a limited number of repairs. *European Journal of Operational Research*, 187(1):84–97, May 2008.

[9] G. Chiola, C. Dutheillet, G. Franceschinis, and S. Haddad. Stochastic well-formed coloured nets for symmetric modelling applications. *IEEE Transactions on Computers*, 42(11):1343–1360, nov 1993.

[10] G. Chiola, G. Franceschinis, R. Gaeta, and M. Ribaudo. GreatSPN 1.7: Graphical Editor and Analyzer for Timed and Stochastic Petri Nets. *Performance Evaluation, special issue on Performance Modeling Tools*, 24(1-2):47–68, November 1995.

[11] D. Codetta-Raiteri, G. Franceschinis, and M. Gribaudo. Defining formalisms and models in the Draw-Net Modelling System. In *Int. Workshop on Modelling of Objects, Components and Agents*, pages 123–144, Turku, Finland, June 2006.

[12] D. Codetta-Raiteri, G. Franceschinis, M. Iacono, and V. Vittorini. Repairable Fault Tree for the automatic evaluation of repair policies. In *Int. Conf. on Dependable Systems and Networks*, pages 659–668, Florence, Italy, June 2004.

[13] S. Contini. *ASTRA - Advanced Software Tool for Reliability Analysis, Theoretical Manual*. European Commission Joint Research Centre (JRC), Ispra, Italy, http://www.jrc.ec.europa.eu, 1999. EUR 18727en.

[14] H. Howard. *Dynamic Programming and Markov Process*. MIT Press, 1960.

[15] G. Krishnamurthi, A. Gupta, and A. K. Somani. HIMAP: Architecture, Features, and Hierarchical Model Specification Techniques. *Lecture Notes in Computer Science*, 1469:348–351, 1998.

[16] R. Manian, D. Coppit, K. Sullivan, and J. Dugan. Bridging the Gap Between Systems and Dynamic Fault Tree Models. In *Annual Reliability and Maintainability Symposium*, pages 105–111, 1999.

[17] M. Puterman. *Markov decision processes : Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994.

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

[19] R. Righter. Optimal maintenance and operation of a system with backup components. *Probability in the Engineering and Informational Sciences*, 16(3):339–349, 2002.

[20] R. A. Sahner, K. S. Trivedi, and A. Puliafito. *Performance and Reliability Analysis of Computer Systems; An Example-based Approach Using the SHARPE Software Package*. Kluwer Academic, 1996.

[21] W. Schneeweiss. *The Fault Tree Method*. LiLoLe Verlag, 1999.

[22] K. J. Sullivan, J. B. Dugan, and D. Coppit. The Galileo fault tree analysis tool. In *Annual Int. Symposium on Fault-Tolerant Computing*, pages 232–235, Madison, Wisconsin, USA, June 1999.

[23] G. J. Wang and Y. L. Zhang. Optimal periodic preventive repair and replacement policy assuming geometric process repair. *IEEE Transactions on Reliability*, 55(1):118–122, March 2006.

[24] FTA-Pro tool's web page. http://www.dyadem.com/products/ftapro/index.php.

[25] Relex tools' web page. http://www.relex.com/products/ftaeta.asp.

[26] Web page of the FaultTree+ tool by Isograph Software. http://www.isograph.com/faulttree.htm.

[27] Web page of the FTAnalyzer tool by Advanced Logistic Development (ALD). http://www.ald.co.il/products/FTAnalyzer.html.