HAL
open science

Using Coloured Petri Nets for integrated reliability and safety evaluations

Bruno Pinna, Génia Babykina, Nicolae Brinzei, Jean-François Pétin

To cite this version:

Bruno Pinna, Génia Babykina, Nicolae Brinzei, Jean-François Pétin. Using Coloured Petri Nets for integrated reliability and safety evaluations. 4th IFAC Workshop on Dependable Control of Discrete Systems, DCDS'13, Sep 2013, York, United Kingdom. pp.19-24, 10.3182/20130904-3-UK-4041.00016 . hal-00872417

HAL Id: hal-00872417

https://hal.science/hal-00872417v1

Submitted on 12 Oct 2013

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.

Using Coloured Petri Nets for integrated reliability and safety evaluations

B. Pinna * **G. Babykina** ** **N. Brînzei** ** **J-F. Pétin** **

Université de Lorraine, Centre de Recherche en Automatique de Nancy, CNRS UMR 7039, 2 avenue de la Forêt de Haye, Vandœuvre-lès-Nancy, 54516 FR

* *bru.pinna1@studenti.unica.it*
** {*genia.babykina,nicolae.brinzei,jean-francois.petin*}*@univ-lorraine.fr*

Abstract: Integrated Deterministic and Probabilistic Dependability Analysis (IDPDA) is respectively required for safety properties verification and reliability & availability assessment of critical systems. This paper presents an approach towards IDPDA using Coloured Petri Nets (CPN). Contributions are related to: (a) hierarchical modelling guidelines that cover deterministic and probabilistic features of a physical system under control, (b) coupling Monte-Carlo simulation with CPN model checking that requires a previous determinisation of the CPN stochastic model. Our approach is illustrated using a toy case study.

Keywords: reliability and availability probabilistic assessment, safety deterministic verification, Coloured Petri Nets, stochastic modelling, determinisation.

1. INTRODUCTION

Critical systems are subject to high dependability constraints, including safety, reliability and availability. Safety properties are generally analysed or proven using deterministic techniques such as event-driven simulation or formal verification. The underlying models are state-based models which may be temporised; verification techniques such as model-checking (Clarke et al., 1999) are based, in this case, on the exploration of the state space.
Reliability and availability are usually evaluated through probabilistic indicators, such as the probability of undesirable event occurrence, with the help of Monte-Carlo simulation or analytical resolution. The underlying models are often probabilistic state-based models (stochastic Petri nets, stochastic automata,...) whose transitions can be timed and associated with a discrete or continuous probability distribution.
Integrated **D**eterministic and **P**robabilistic **D**ependability **A**nalysis (IDPDA) combines within a same study formal verification of deterministic behaviour with regard to safety properties and stochastic quantitative assessment of the system reliability and availability. This kind of mixed deterministic and probabilistic analysis is particularly relevant in the context of dynamic reliability where the structure function evolves over the time due to the impact of the physical parameters and device ageing on the dysfunctional behaviour and control architecture reconfiguration. This paper presents a Coloured Petri Nets (CPN) based approach towards IDPDA and focuses on redundant and reconfigurable system where control applies reconfiguration policies to compensate system failures.

⋆ B. Pinna has been funded by the EC Erasmus programme in which Université de Lorraine and Universita degli studi di Cagliari are involved.

Second section introduces the requirements for a formal modelling that supports IDPDA. Third section presents a generic modelling approach based on CPN including hierarchy, stochastic and temporised features to capture functional and dysfunctional features of both control and physical systems. Fourth section shows how obtained CPN can be determinised to enable verification of the safety properties, while it can be used, as is, for analysis of the availability and reliability indicators. Fifth section illustrates the proposed approach on a toy case study. Conclusion and prospects are discussed in the last section.

2. IDPDA MODELLING REQUIREMENTS

IDPDA is a current industrial and scientific challenge for dependability community as shown by recently organised workshops (Adolfsson et al., 2012). As far as modelling is concerned, following main requirements are highlighted:

- the formalism has to provide high level modelling mechanisms such as hierarchy or modularity to capture the complex behaviour of a critical system that is often composed of several components (control devices, process devices, communication devices, ...),
- the formalism has to support Monte-Carlo simulation since the stochastic behaviour is, most of the time, not limited to exponential distributions and consequently cannot be analytically solved as a Markovian process,
- the model must be determinisable in order to produce a finite state space that can be explored to prove some deterministic safety properties.

Among several candidates that more or less cope with those requirements (stochastic automata or Petri Nets, Boolean Driven Markov Process which combines fault trees and automata), the Coloured Petri Nets (CPN), defined by Jensen (Jensen, 1997; Jensen and Kristensen,

2009), has been chosen for modelling the system behaviour including its physical and control parts. CPN is a discrete-event modelling language combining the capabilities of Petri nets with a high-level programming language. Its main differences with ordinary PN are:

- the *colours* that identify and characterize a token with different data types (e.g. Boolean, integer, string, or more complex data structure),
- the *hierarchical* concept that promotes the modelling of a complex CPN by combining several small CPNs; it overcomes the lack of compositionality, that is one of the main critiques raised against Petri net models,
- CPN may be extended to time concept, which is very useful to investigate the dynamic behaviour and to assess dependability indicators such as reliability or MTTF (Mean Time to First Failure).

The proposed approach follows three steps: (a) modelling the system (physical and control parts) using Colored Petri Nets including stochastic aspects, (b) Monte-Carlo simulation to provide probabiblistic indicators, (c) determinization of the model to support formal verification of safety properties.

3. PROPOSED MODELLING APPROACH

Because the CPN and its concepts of "colour", of "hierarchy" and "time" are well known in the in the community of *discrete event systems*, this paper does not present their formal definitions, for which we send the reader to Jensen (1997) or Jensen and Kristensen (2009). Nevertheless, the following sections introduce the way of modelling a system with CPN keeping in mind our objective to provide both probabilistic assessment of the reliability indicators and verification of deterministic safety properties.

3.1 Hierarchical modelling

Individual CPN models can be hierarchically related to each other in a formal way, *i.e.* with a well-defined semantics and formal analysis capabilities. CPN model hierarchy is realized through *substitution transitions*. The idea is to associate a transition (and its surrounding arcs) to a more complex CPN (a module), which gives a more precise and detailed description of the activity represented by the substitution transition. The places connected to a substitution transition, called *socket places*, have clearly defined corresponding places, called *port places*, in the related CPN module. They can transmit a given marking from a high level (level of substitution transition) to a low level (level of module) and vice versa. The number of levels in a hierarchical CPN is not limited, because a CPN module corresponding to a substitution transition can also contain other substitution transitions that are related to lower-level CPN modules.
CPN concept of hierarchy allows us to propose a complex system *modular modelling* based on generic modules that can be instantiated as often as needed. First level of hierarchy separates the physical and the control models of the system. This top level (Fig. 1) contains two substitution transitions (transitions with double-line borders), one of these representing the physical part and the other representing the control part. These transitions exchange information by means of an intermediate place that indicates the status of the system. Each of substitution transitions is assigned a CPN module (called *TPA* for the physical system in Fig. 1 and *Specification* for control part). Physical system model embeds timed stochastic transitions capturing the random failure or reparation events while the control model is, by definition, only deterministic.

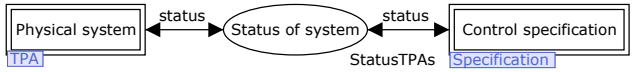

Fig. 1. Top module of the CPN model of system.

The hierarchy can be further developed at different levels, according to the system structure. For example, the physical part can be decomposed into sub-systems such as functional units, devices, components, *etc.* For the control part, sub-modules associated with control functions can be refined. For each level, one or several CPN modules can be instantiated on the basis of generic models.

3.2 Stochastic modelling

To assess dependability indicators, the stochastic events, such as failures and repairs which occur in the physical part of the system, must be taken into account. They are modelled by *stochastic transitions*, which fire after a random enabling time. At the same time, deterministic reaction of the control part must be modelled using *immediate transitions* that occur instantaneously in time.
These requirements are covered by a particular class of Generalized Stochastic Petri Nets (GSPN) defined in (Ajmone Marsan et al., 1984). Immediate transitions have priority over the stochastic transitions: if in a given marking, immediate and stochastic transitions are simultaneously enabled, the immediate transitions are fired firstly in zero time once they are enabled. In contrast, when a token enables several stochastic transitions (competition between transitions T_1 and T_2 in Fig. 2), it is assigned to the transition for which the realisation of a random time variable (characterized by a transition rate λ) is smaller. This transition wins the token but is only fired after the token sojourns in the input place according to random time variable.

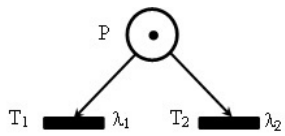

Fig. 2. Concurrence of GSPN stochastic transitions.

In a timed CPN (Jensen, 1997; Jensen and Kristensen, 2009), the time is given by a global clock. In addition to their colour, the tokens in a marking contain a time value, also called a *time stamp*. When a transition is enabled, it is fired and changes the time stamps of tokens which are deposited in its output places. In these places, the tokens remain *frozen* and can not be used to enable other transitions until the current model time (as given by the global clock) is smaller than their time stamps. As soon as the time stamp of the tokens is less than or equal to the current time model, these tokens can enable other transitions which are instantly fired. In other words, the time stamp describes the *earliest* model time from which a

token can be used. Consequently, this behaviour matches the formalised theoretical behaviour of P-timed Petri net operating at its maximum speed.
Regarding the immediate transitions, their behaviour is the same in the GSPN and CPN models, but, unfortunately, the behaviour of the stochastic transitions is quite different in timed CPN model compared to GSPN formal model. If the Petri net of Fig. 2 is a timed CPN, as soon as the token arrives to place P, it is assigned to one of the two transitions, T_1 or T_2, and this transition immediately fires. The token sojourns zero time in place P and its time stamp is modified according to the rate of the corresponding transition and it sojourns in the output place as long as required by its time stamp.
To solve the problem of stochastic transition competition and to force a sojourn time in the input place of a stochastic transition, the idea is to deal with the competition stochastic choice before enabling T_2 and T_3 transitions. This anticipated process is done by modelling the GSPN of Fig. 2 by the CPN of Fig. 3:

- transitions "$Fault1$" and "$Fault2$" correspond to the transitions T_1 and T_2,
- place "$Working$" corresponds to place P,
- additional transition *Starting* takes a random value from distributions associated to T_1 and T_2 and characterised by λ_1 and λ_2 rates; the smallest of these random values is the expected sojourn time in place P. It is allocated to time stamp of the token deposited in place "$Working$" ; this process is coded in ML language (Harper, 1998) and requires the definition of a more complex colour containing the values of the two random variables in addition to token information.
- as soon as global clock reaches the token time stamp, only the transition corresponding to the smallest value of random time variables (as defined in transition "$Starting$") must be fired; this can be done using guard given to arc from place "$Working$" to transitions "$Fault1$" and "$Fault2$".

The proposed model (Fig. 3) ensures a behaviour that is compliant with formal GSPN with *enabled memory* policy for firing transitions. The rationale is generic and can be applied each time a competition between stochastic transitions occurs.

4. TOWARDS IDPDA

4.1 Probabilistic assessment

Dependability indicators assessment requires modelling of dysfunctional behaviour of the system through stochastic events such as failures or reparations. If all the stochastic events are described by exponential distribution functions, a stochastic Petri net is isomorphic to continuous time Markov chain (Molloy, 1981) and in this case an analytical expression of dependability indicators can be obtained. An analytical expression of dependability indicators may also be obtained, if the exponential assumption is slightly released by assuming that at most one transition, among all enabled transitions for each marking, is characterized by any general distribution function while all other enabled transitions are exponential (in this case, the underlying stochastic process is a semi-Markov, semi-regenerative

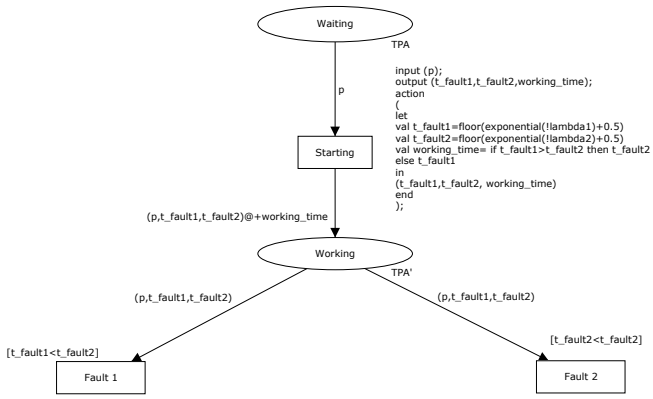

Fig. 3. GSPN Modelling behaviour with a timed CPN.

process or other extensions of Markovian process). In this paper, we want to drop these assumptions by enabling any kind of distribution for all the transitions. Consequently, the underlying process of such a CPN is a non-markovian process and the only way to assess the dependability indicators is the Monte Carlo Simulation way.

In Monte Carlo simulation, it is necessary to define the dependability indicators in terms of performances of Petri nets. In the dependability assessment, two classes of indicators can be defined:

- *probabilistic indicators*, such as reliability, availability or maintenability
- *mean time indicators*, such as MTTF, MTTR, ...

The *probabilistic indicators* are measured by a *probability*, and they can be determined based on marking invariants (a marking invariant is a subset of Petri net places where the number of tokens is constant). A token, that represents a component, a sub-system or a system, evolves in places which describe its state (waiting, working, failure, ...) and these places make a marking invariant. The probability to be estimated, is given by the ratio between the average marking of the place(s) that describes the state(s) characterizing the searched indicator and the sum of the average marking of all places belonging to the invariant, i.e. the number of tokens contained by the places subset:

$$P(state_I) = \frac{M^*(state_I)}{\sum\limits_{P_i \in P_{subset_I}} M^*(P_i)}$$

where $state_I$ is the state that characterises the probabilistic indicator I, $M^*(state_I)$ is its average marking and P$subset_I$ is the places subset of invariant. For example, for one system (number of tokens is equal to 1), its unavailability can be estimated by the following equation:

$$\bar{A} = P\left(state_{\bar{A}}\right) = \frac{M^*(state_{\bar{A}})}{\sum\limits_{P_i \in P_{subset_{\bar{A}}}} M^*(P_i)} = M^*(state_{\bar{A}})$$

where $state_{\bar{A}}$ represent all the down states of the system. Consequently, such probabilistic indicators can be estimated by average marking of the corresponding place(s).
The *mean time indicators* are measured by the average value of the sojourn time in the place(s) characterizing the searched indicator.

$$MTI = \sum_{P_i \in P_I} D^*(P_i) \qquad (1)$$

where MTI is the Mean Time of Indicator I, P_i is a state characterizing the indicator I, P_I the subset of all these places and D*(P_i) is the average value of sojourn time in the place P_i given by Little's formula:

$$D^*(P_i) = \frac{M^*(P_i)}{\sum\limits_{T_i \in {}^\circ P_i} w(T_j, P_i) F^*(T_j)} \quad (2)$$

The denominator of Eq. (2) gives the sum of the product of average frequency F of input transition T_j of place P_i and of the weight of the input arc from T_j to P_i, $w(T_j, P_i)$. The sum is given for all transitions T_j belonging to subset of input transitions of P_i, noted ${}^\circ P_i$. For example, the MTTF of system can be estimated by the following equation:

$$MTTF = \sum_{P_i \in P_{operate}} D^*(P_i)$$

where $P_{operate}$ is the subset of operating places.

4.2 Deterministic verification

Safety verification employs deterministic analysis of the system behaviour in order to:

- prove that the system behaves in accordance with a given specification, i.e. to prove a safety property meaning that something bad never happens. This kind of verification is often used to check the control behaviour correctness with regard to its specification.
- identify critical sequences and their length leading from a given state to an undesired state; this information is useful for dependability improvement.

All these deterministic analysis are based on the exploration of the system *state space*, and in our case, on the CPN marking graph. As the CPN system model contains stochastic timed transitions, the first step toward verification is the determinisation of the CPN model.

Basic ideas for determinisation are the following :

- first thing to be done is obviously to cancel the timed and stochastic features of the transitions,
- this cancellation could give rise to behaviour that was impossible when taking into account stochastic timed transitions. That is for example the case when an immediate transition was concurrent with a timed one in the stochastic timed model. After determinisation, both transitions become immediate and are in competition leading to more marking than possible with timed net.

Let us have a short example to highlight the problem. In the net of Fig. 4, the places P1 and P2 represent process dynamics while t1 and t2 are stochastic timed transitions that represent respectively failure and reparation events. The places P3 and P4 represent control behaviour while t3 and t4 are immediate transitions that represent the control reaction respectively to a failure and reparation events. In the marking $M_c = (0, 1, 1, 0, 1, 0)$, $t2$ and $t3$ are concurrent but $t3$ has priority since it is immediate. Those two transitions remain concurrent in the untimed CPN but no priority will be allocated; the marking M_c has consequently two successors by firing transitions $t2$ or $t3$ and a repetitive sequence ($t1t2$) is possible; this behaviour was not allowed in the stochastic timed Petri Nets (SPN).

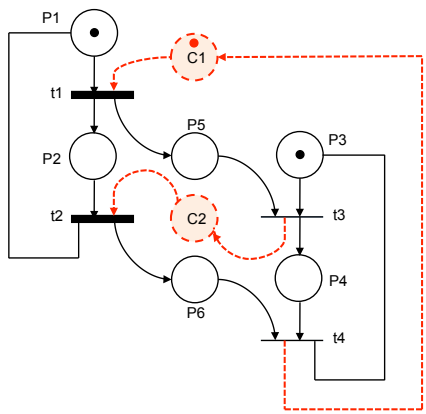

Fig. 4. CPN example for the determinisation problem

To solve the problem, the idea consists in introducing control places that limit the marking graph to the only behaviour that was enabled in the stochastic timed model, and dynamically disable some transition firings. Supervisory control theory and more specifically a synthesis approach which are based on marking invariants can then be used (Giua et al., 1992; Iordache and Antsaklis, 2006). In our case, the expected control places should enforce a set of linear constraints of place markings that maintain the reachable markings in the only behaviour that is enabled by the stochastic CPN. The set of these linear constraints may be expressed by the following equation:

$$Lm \leq k$$

where m is the marking vector of stochastic Petri net, L is a matrix of coefficients and k is a vector of constants. In the net of the Fig. 4, the making constraints are:

(a) $m(P1) + m(P2) + m(P4) + m(P5) \leq 2$
(b) $m(P1) + m(P2) + m(P3) + m(P6) \leq 2$

(a) means that in the marking $(0, 1, 1, 0, 1, 0)$, $t3$ must be fired before $t2$ while (b) means that in the marking $(1, 0, 0, 1, 0, 1)$, $t4$ must be fired before $t1$. These inequalities can be represented by:

$$L = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \text{ and } k = \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$

Applying (Giua et al., 1992), the incidence matrix W_c of the supervisor that enforces the previous constraints and its initial marking are given by:

$$W_c = -LW$$

$$m_{c0} = k - Lm_0$$

where W is the incidence matrix and m_0 is the initial marking of the original SPN. The deterministic untimed net has the following incidence matrix and initial marking:

$$W_{\det PN} = \left[W^T, W_c^T \right]^T$$

$$m_{\det PN0} = \left[m_0^T, m_{c0}^T \right]^T$$

For Fig. 4 net, this algorithm leads to add two places $C1$ and $C2$ with an initial making $m(C1) = 1$ and $m(C2) = 0$.

This operation provides a marking graph that could be analysed using model checking techniques (Clarke et al., 1999). Rationale (see Fig. 5) is based on known transformation of CPN on one side and temporal logic formulae on the other side into Buchi automata. The product of the two

resulting Buchi automata provides the property status: empty product means that the property holds, in other case, the product automata traces give counter-examples.

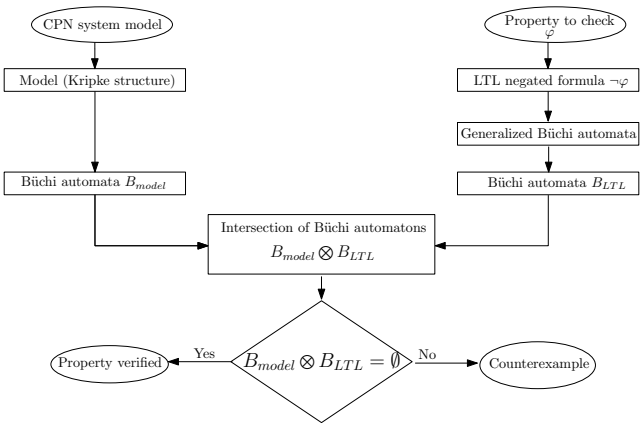

Fig. 5. CPN Model-checking rationale.

5. CASE STUDY

To illustrate the proposed approach, a toy example is extracted from a more real case study proposed by EDF for Approdyn project (Aubry et al., 2012). This example is composed by two parallel feed-water turbo-pumps (TPA). Each pump is composed of two subsystems: a turbine part (noted T) and an out-of-turbine part (noted Out-of-T or OT). If one of those subsystem fails, the corresponding feed-water pump fails. The reliability block diagram of this system is given in Fig. 6.

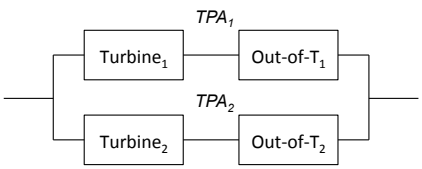

Fig. 6. Reliability block diagram of TPAs system.

The data characterizing the failure and reparation process of each component are presented in Table 1. The failure phenomena are characterised by the exponential law, with the following cumulative distribution function:

$$F(t) = 1 - e^{-\lambda_c t},$$

with the rate parameter $\lambda_c = 1/MTTF_c$ (c=T or Out-of-T component). For the reparation times an Erlang law is considered. Its cumulative distribution function is:

$$F(t) = 1 - \sum_{k=0}^{n-1} \frac{1}{k!} e^{-\lambda_c t} (\lambda_c t)^k$$

where: $\lambda_c = 1/MTTR_c$ (c=T or Out-of-T component) is the rate parameter and n=2 is the order parameter.
The specification of the control part of this system is the following: if both pumps are in ON state, the system is working at the nominal parameters. If one of the components of a pump fails, the other component of the same pump is stopped and a reparation order is given. The system works in a degraded operating mode. When the repair is finished, the system restarts immediately the repaired pump. When both pumps are in failure state, the entire system is failed.

Table 1. TPAs MTTF and MTTR *(in hour)*.

TPA	$MTTF_T$	$MTTF_{OT}$	$MTTR_T$	$MTTR_{OT}$
TPA_1	6780	6854	4	48
TPA_2	2260	6.8×10^6	48	288

The CPN model of this system (physical part and control part) is developed under CPNTools free software. The system model has been developed with two levels of hierarchy (one for the components themselves and another for coupling process and control sub-models). It has 15 places, 13 normal transitions and 2 substitution transitions (specification part is presented in Fig. 7). Four colour set (a single colour for distinguishing *pump*1 and *pump*2, three composed colours to associate to each pump respectively its state, its stochastic duration, and its failure occurrence time) set are used for modelling and several functions are developed in ML language (e.g. calculus of stochastic durations, guards evaluation, and monitor functions).

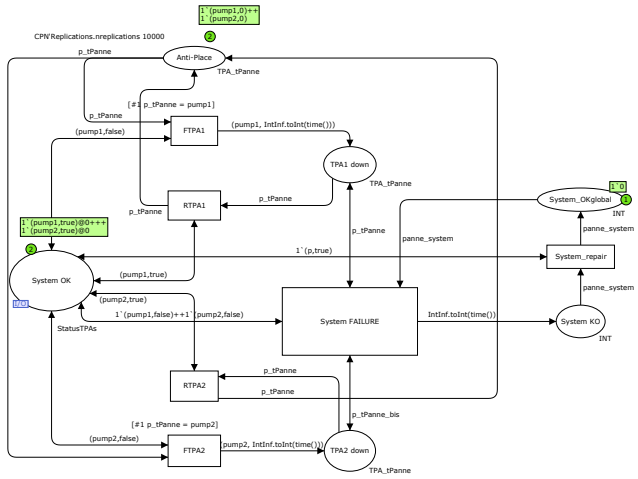

Fig. 7. Model of control part specification.

5.1 Probabilistic assessment

In order to asses the dependability indicators several monitor functions are defined in CPNTools. They extract and collect data from CPN during Monte Carlo simulation. Each dependability indicator is calculated by a monitor function that provides several measures of interest (e.g. the average value, the confidence intervals, the min and max value, *etc.*). The following monitors are defined:

- marking size monitors estimate the unavailability of each pump (a subsystem) and of whole system. The corresponding availability is given by: A = 1 - $\bar{A}$.
- monitors assess the sojourn time in different places to estimate the MTTF, MTTR and MTBF values of each pump and of the whole system. Several CPN places (subset P_I in Eq. (1)) represent the states that must be taken into account for the assessment of one of these indicators. A monitor must be defined for each of these places and the value of searched indicator is the sum of these sojourn times (Eq. (1)).

Some relevant statistical results after 10000 replications are illustrated in Table 2 where 95%$(1/2)CI$ represents the half-length of a 95% Confidence Interval.

Table 2. System performance results

Indicators	Type	TPA1	TPA2	System
MTTFF	Average	3111.7	2234.9	6340.5
(in hour)	95%(1/2)CI	61.75	43.9	375
MTBF	Average	3144.2	2259.2	6312.8
(in hour)	95%(1/2)CI	62.4	44.35	373.35
MTTR	Average	26.4	48	16.4
(in hour)	95%(1/2)CI	0.5239	0.9428	0.9734
Unavailability	Average	0.00694	0.01901	0.00014
(%)	95%(1/2)CI	0.000113	0.000196	0.000012

5.2 Deterministic verification

The TPA model is determinised according to the method given in section 4.2. The resulting marking graph, whose size is 100 arcs and 72 nodes, is then analysed. CPN Tools provides a marking graph analyser (*state space generator*) to perform classical verification (reachability, liveness, ...) and embeds several plug-ins that enable verification with regard to properties expressed in temporal logic formulae: ASAP tool (ASCoVeCo State Space Analysis Platform) presented in Kristensen and Westergaard (2007) based on LTL formulae or ASK-CTL tool based on CTL formulae. The following properties has been checked:

- the control must immediately restart the TPA_1 after it has been repaired ; this illustrates assessment of a controller safety property,
- from any state where TPA_1 is not started, it is always possible to restart it; this illustrates a controller liveness property,
- What is the minimum path from a broken TPA_1 to a state where the entire system is in failure ? This property refers to analysis of critical sequences.

The first two properties can be expressed by LTL or CTL formulae and are verified using ASAP (with a limitation for the counter-example generation) and ASK-CTL:

- TPA_1 repaired event implies that TPA_1 is restarted in the next step that can formalised as: $AG(TPA_1 repaired \Rightarrow AX\ TPA_1 restarted)$,
- for all the path from a selected node where TPA_1 is not started, TPA_1 is restarted should be TRUE in a finite number of steps that can be formalised as : $AG(TPA_1 notstarted \Rightarrow EF\ TPA_1 restarted)$

The third property is analysed using exploration tool that follows three steps:

- finding all the states where TPA_1 is down (noted $init_state$)
- finding all the states where the entire system is down (noted $final_state$)
- finding paths from $init_state$ to $final_state$

These steps can be coded into ML formulae :

- for the first query (second query is similar) by:

```
PredAllArcs(
 fn a => case ArcToBE a of
 Bind.Specification 'FTPA1 (1, {}) => true
 | _ => false)
```

- for the third query by:

```
Reachable'(init_node,final_node)
```

Six nodes where TPA_1 is down $[9, 30, 26, 28, 25, 11]$ and a unique node where the entire system is down [34] are found. The shortest trace from down TPA_1 to node [34] is obtained from node [9] and have a length of 3.

6. CONCLUSION

In this paper, an approach based on Coloured Petri Nets (CPN) has been proposed to cover an integrated probabilistic and deterministic dependability analysis. A first contribution relies on a modelling method that captures the behaviour of a formal Generalised Stochastic Petri Nets in the form of a CPN model. Second contribution provides a determinisation algorithm of stochastic timed Petri Nets to generate a finite marking graph. Based on these two contributions, deterministic safety verification using model-checking techniques and probabilistic reliability & availability assessment using Monte-Carlo simulations can be jointly performed. A scaling up towards industrial-sized systems is currently to be done within the framework of CONNEXION project (French governmental project that brings together the main actors of the French Nuclear Power Plant).

REFERENCES

Adolfsson, Y., Holmberg, J.E., Karanta, I., and Kudinov, P. (eds.) (2012). *Proceedings of the Integrated Deterministic-Probabilistic Safety Analysis Workshop*. Stockholm, Sweden.

Ajmone Marsan, M., Conte, G., and Balbo, G. (1984). A class of generalized stochastic petri nets for the performance evaluation of multiprocessor systems. *ACM Trans. Comput. Syst.*, 2(2), 93–122.

Aubry, J.F., Babykina, G., Brinzei, N., Barros, A., Bérenguer, C., Grall, A., Langeron, Y., Deleuze, G., et al. (2012). The approdyn project: dynamic reliability approaches to modeling critical systems. *Supervision and Safety of Complex Systems*, 181–222.

Clarke, Jr., E.M., Grumberg, O., and Peled, D.A. (1999). *Model checking*. MIT Press, Cambridge, MA, USA.

Giua, A., DiCesare, F., and Silva, M. (1992). Generalized mutual exclusion constraint on nets with uncontrollable transitions. In *Proceedings of the International Conference on Systems, Man, and Cybernetics*, 974–979. Chicago, IL, USA.

Harper, R. (1998). Programming in standard ml.

Iordache, M.V. and Antsaklis, P.J. (2006). *Supervisory Control of Concurrent Systems: A Petri Net Structural Approach*. Birkhauser.

Jensen, K. and Kristensen, L. (2009). *Coloured Petri nets: modeling and validation of concurrent systems*. Springer-Verlag New York Inc.

Jensen, K. (1997). *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use (Volume 1)*, volume 1. Springer Verlag.

Kristensen, L. and Westergaard, M. (2007). The ascoveco state space analysis platform: Next generation tool support for state space analysis. In *Eighth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, 1.

Molloy, M. (1981). *On the Integration of Delay and Throughput Measures in Distributed Processing Models*. Ph.D. dissertation. Univ. of California, Los Angeles.