Probabilistic Model Checking of non-Markovian Models with Concurrent Generally Distributed Timers

András Horváth, Marco Paolieri, Lorenzo Ridi, Enrico Vicario
Università di Torino - horvath@di.unito.it
Università di Firenze - {marco.paolieri, lorenzo.ridi, enrico.vicario}@unifi.it

***Abstract*—In the analysis of stochastic concurrent timed models, probabilistic model checking combines qualitative identification of feasible behaviors with quantitative evaluation of their probability. If the stochastic process underlying the model is a Continuous Time Markov Chain (CTMC), the problem can be solved by leveraging on the memoryless property of exponential distributions. However, when multiple generally distributed timers can be concurrently enabled, the underlying process may become a Generalized Semi Markov Process (GSMP) for which simulation is often advocated as the only viable approach to evaluation.**

The method of stochastic state classes provides a means for the analysis of models belonging to this class, that relies on the derivation of multivariate joint distributions of times to fire supported over Difference Bounds Matrix (DBM) zones. Transient stochastic state classes extend the approach with an additional age clock associating each state with the distribution of the time at which it can be reached.

We show how transient stochastic state classes can be used to perform bounded probabilistic model checking also for models with underlying GSMPs, and we characterize the conditions for termination of the resulting algorithm, both in exact and approximate evaluation. We also show how the number of classes enumerated to complete the analysis can be largely reduced through a look-ahead in the non-deterministic state class graph of reachable DBM zones. As notable traits, the proposed technique accepts efficient implementation based on DBM zones without requiring the split of domains in regions, and it expresses the bound in terms of a bilateral constraint on the elapsed time without requiring assumptions on the discrete number of executed transitions. Experimental results based on a preliminary implementation in the Oris tool are reported.

***Index Terms*—Generalized Semi-Markov Process, Non-Markovian Stochastic Petri net, probabilistic model checking, stochastic state class, DBM zones.**

I. Introduction

Model checking is an automated way to verify whether a model satisfies formally specified properties which can combine temporal logical operators, time constraints, and quantifiers over the set of feasible behaviors. This has been developed and applied mainly in correctness verification of sequencing and timeliness requirements of concurrent and timed systems [19]. More recently, the approach has been proposed also in a quantitative perspective that aims not only at verifying the feasibility of the specified behavior, but also at evaluating its probability. This can serve to evaluate dependability and performance indices that require quantitative evaluation of steady-state and transient rewards along a restricted set of feasible behaviors. To this end, a measure of probability is associated with non-deterministic choices so that the behavior of the model is characterized by an underlying stochastic process [17].

Various formulations have been proposed, first addressing models with an underlying Discrete Time Markov Chain [38], [26] and then more complex stochastic processes [2], [31], [5]. In particular, continuous time probabilistic model checking was efficiently implemented for Continuous Time Markov Chains (CTMC) [5], [30], [18]. For this class of processes, both steady state and transient probabilities can be derived through efficient numerical techniques. Moreover, since all timers are distributed according to an exponential (memoryless) distribution, the logical location of the model comprises a sufficient description of its state, facilitating the integration of algorithmic state space exploration and quantitative probabilistic evaluation. However, the Markov assumption also restricts the expressive power of the model, ruling out generally-distributed timers and thus preventing representation of behaviors that accumulate memory with the passage of time. Continuous phase-type approximants [34] can partially smooth the problem, but they encounter a critical trade-off between accuracy and complexity, and they are in any case unable to represent timers with finite support distributions.

Probabilistic model checking over continuous-time non-Markovian models was addressed in [8] for one-clock Timed Automata. The one-clock assumption guarantees that memory is reset at every restart so that the underlying process is a semi-Markov Process [22], as occurring in non-Markovian Stochastic Petri Nets under the so-called enabling-restriction [17], [16]. When multiple generally distributed (non-deterministic) timers are allowed to overlap their activity cycles [17], [41], the underlying process does not satisfy the enabling restriction and it may belong to the class of Generalized Semi-Markov Processes (GSMP) [24], for which simulation is often advocated as the only viable approach to evaluation.

An approach for the analysis of models with multiple generally distributed (non-deterministic) timers possibly supported over finite domains was proposed in [12], [15] by integrating concepts of qualitative verification and quantitative evaluation. The approach relies on symbolic derivation of a probability

density function supported over classes of equivalent states encoded through Difference Bounds Matrix zones. This enables exact steady state analysis, provided that activity cycles of generally distributed transitions cannot infinitely overlap [15], [28], which in turn can be efficiently detected on the non-deterministic state class graph of reachable zones usually employed in qualitative verification [39]. Transient analysis is obtained through the addition of an age clock that associates each class with the distribution of the absolute time at which it can be reached [27], largely relaxing the conditions that guarantee termination of the analysis within a given time bound. The calculus involved in the derivation of probability density functions over DBM zones was recently proposed also for Duration Probabilistic Automata, which compose a set of acyclic Semi-Markov Processes under control of a non-deterministic scheduler [32]. Symbolic derivation of a closed-form for a density function over equivalence classes was independently proposed also in [1], with a calculus similar to that of [12] but relying on a partition of the support of times to fire based on regions [3] rather than DBM zones. In that paper, the derivation is applied to probabilistic model checking over a subclass of GSMPs, providing a way to decide a real-time until operator through the introduction of a bound on the maximum number of steps taken by the process.

In this paper, we propose an analytic approach for the evaluation of a probabilistic bounded until operator for models with multiple generally distributed timers with possibly overlapping activity cycles. The proposed technique permits the expression of the bound as a bilateral constraint on the elapsed continuous time rather than on the discrete number of executed transitions, and it accepts efficient implementation based on DBM zones without requiring a region-based partition. This also enables a look-ahead strategy that leverages on the non-deterministic state class graph of reachable zones to reduce the number of stochastic classes traversed during the analysis.

The rest of the paper is organized as follows. In Sects. II-III, we recall the definition of stochastic Time Petri Nets and the concepts of their analysis through transient stochastic classes. In Sect. IV we show how transient stochastic state classes can be used to evaluate the probability of satisfaction of a time-bounded until operator, and we characterize the conditions for termination both in exact and approximate evaluation. In Sect. V, we show how the number of classes enumerated to complete the analysis can be reduced by looking ahead reachable DBM zones without evaluating their probabilities. Finally, in Sect. VI-VII, we report experimental results based on a preliminary implementation in the Oris tool, and we draw conclusions.

II. Stochastic Time Petri Nets

We formulate our technique with reference to a class of non-Markovian Petri Nets which we call stochastic Time Petri Nets (sTPN) [15], [12]. The name is intended to suggest that sTPNs can be regarded as a stochastic extension of an underlying non-deterministic Time Petri Net, which in fact turns out to determine the conditions for the termination of the analysis.

$$sTPN = \langle P, T, A^-, A^+, A^\bullet, m_0, EFT, LFT, \mathcal{F}, \mathcal{C} \rangle$$

As in Time Petri Nets [33], [7], [39]:

- P is a set of places;
- T is a set of transitions disjoint from P;
- $A^- \subseteq P \times T$, $A^+ \subseteq T \times P$ and $A^\bullet \subseteq P \times T$ are pre-conditions, post-conditions, and inhibitor arcs, respectively;
- $m_0 : P \to \mathbb{N}$ is the initial marking;
- $EFT : T \to \mathbb{R}_0^+$ and $LFT : T \to \mathbb{R}_0^+ \cup \{+\infty\}$ associate each transition with an Earliest and a Latest Firing Time, with $EFT(t) \leq LFT(t)$.

In addition:

- $\mathcal{C}$ associates each transition with a real-valued weight;
- $\mathcal{F}$ associates each transition $t \in T$ with a cumulative probability distribution F_t supported on the firing interval $[EFT(t), LFT(t)]$.

The state of an sTPN is a pair $s = \langle m, \tau \rangle$, where $m : P \to \mathbb{N}$ is a *marking* and $\tau : T \to \mathbb{R}_0^+$ associates each transition with a *time-to-fire*. A transition t_0 is *enabled* if each of its input places contains at least one token and none of its inhibiting places contain any token, and it is *firable* if it is enabled and its time-to-fire $\tau(t_0)$ is not higher than that of any other enabled transition. When multiple transitions are firable, the choice is resolved through a random switch determined by the weight $\mathcal{C}$:

$$Prob\{t_0 \text{ is selected}\} = \frac{\mathcal{C}(t_0)}{\sum_{t_i \in T^f(s)} \mathcal{C}(t_i)}$$

with $T^f(s)$ denoting the set of transitions firable in state s.

When a transition t_0 fires, the state $s = \langle m, \tau \rangle$ is replaced by $s' = \langle m', \tau' \rangle$, which we write as $s \xrightarrow{t_0} s'$. Marking m' is derived as usual in Petri Nets:

$$\begin{array}{ll} m_{tmp}(p) = m(p) - 1 & \forall p.\ \langle p, t_0 \rangle \in A^- \\ m'(p) = m_{tmp}(p) + 1 & \forall p.\ \langle t_0, p \rangle \in A^+ \end{array} \tag{1}$$

Transitions that are enabled both by the intermediate marking m_{tmp} and by m' are said to be *persistent*, while those that are enabled by m' but not by m_{tmp} are said to be *newly enabled*. If t_0 is still enabled after its own firing, it is always regarded as newly enabled [7], [39]. For any transition t_i that is persistent after the firing of t_0, the time-to-fire is reduced by the time elapsed in the previous state:

$$\tau'(t_i) = \tau(t_i) - \tau(t_0) \tag{2}$$

Whereas, the time-to-fire of each newly enabled transition t_a is a random value sampled according to the probability distribution F_{t_a}:

$$\begin{array}{c} EFT(t_a) \leq \tau'(t_a) \leq LFT(t_a) \\ Prob\{\tau'(t_a) \leq x\} = F_{t_a}(x) \end{array} \tag{3}$$

Without loss of generality, we rule out deterministic transitions with $EFT(t) = LFT(t) > 0$, which could be encompassed in the treatment by resorting to the partitioned form of DBM zones described in [15]. We maintain in the treatment transitions with $EFT(t) = LFT(t) = 0$ which have relevance for the issue of termination. We call them immediate, and call timed any transition for which $LFT(t) > 0$. Though not

strictly necessary, we also assume that F_t can be expressed as the integral function of a probability density function f_t:

$$F_t(x) = \int_0^x f_t(y)dy \tag{4}$$

If t is an immediate transition, $f_t(y) = \delta(y)$ denotes the Dirac impulse function.

In [25] a more general class of Stochastic Petri Nets is considered, which is there shown to be able to represent any process in the class of GSMPs [24]. With respect to that formulation, stochastic Time Petri Nets do not encompass randomization of state updates and timers evolving with state-dependent rates. In particular, the latter limitation prevents sTPNs from representing GSMPs that can be generated by a Preemptive Resume policy [9], [13].

A. Underlying stochastic process

The evolution of the marking of an sTPN identifies an underlying continuous time stochastic process [17] that may accumulate memory over time due to the presence of generally distributed (GEN) transitions. This may result in different classes of processes depending on the conditions of persistence of GEN transitions.

If the model never permits a GEN transition be persistent at a change in the enabling status of other GEN transitions, then it satisfies the so-called "enabling restriction" and falls in a subclass of Markov Regenerative Processes (MRGP) [29]. In this case, activity cycles of GEN transitions never overlap, and analysis can be carried out through numerical integration of Generalized Markov Renewal Equations with local and global kernels computed on CTMCs subordinated to the activity cycles of GEN transitions [16], [10].

When multiple GEN transitions can be concurrently enabled, their activity cycles can overlap. The underlying process may still fall in the class of MRGP or become a Generalized Semi Markov Process depending on whether the model guarantees or not that a regeneration point is eventually reached with probability 1. In both cases, the only analytical approaches for the evaluation of the process are the method of supplementary variables [20], [23], [37] or the method of stochastic state classes [15], [27].

In principle, the overlap of GEN activity periods can occur in any model with at least two GEN transitions. However, the really most complex and general case occurs when two GEN transitions persist at the firing of a third GEN transition. In this case, the times to fire of persistent GEN transitions become mutually dependent variables and their joint support may become a Difference Bounds Matrix (DBM) zone [21], [39], [6]. This means that the times to fire values are distributed over a domain represented by a set of linear inequalities of the form

$$D = \begin{cases} \tau_i - \tau_j \leq b_{ij} \\ \tau_* = 0 \\ \forall i \neq j \in [0, N-1] \cup \{*, age\} \end{cases} \tag{5}$$

with $b_{ij} \in \mathbb{R} \cup \{+\infty\}$. A non-empty DBM zone has a unique *normal form* of representation where b_{ij} coincides with the maximum value that can be attained by the difference $\tau_i - \tau_j$; this form is univocally identified by the condition:

$$b_{ij} \leq b_{ih} + b_{hj} \qquad \forall i, j, h \in [0, N-1] \cup \{*, age\} \text{ with } i \neq j \neq h \neq i \tag{6}$$

and it can be derived in polynomial time as the solution of an all-shortest-path problem [39].

III. The method of transient stochastic state classes

Stochastic state classes extend Difference Bounds Matrix zones used in qualitative symbolic analysis of the state space of timed models [7], [39], [21] with a density-function that provides a measure for the probability of individual states. In so doing, the graph of non-deterministic state classes based on DBM zones is expanded into a stochastic-class-graph that makes explicit the transition probabilities among classes and the multivariate distribution of times-to-fire of transitions enabled in each class [12], [15]. Transient stochastic classes [27] extend the concept with a supplementary *age* clock that encodes the opposite of the time elapsed since the beginning of the execution under analysis. The age clock is initially set equal to zero and it is always decreased at every transition firing.

A. Transient Stochastic State Classes

Definition 3.1: We call *aged-state* a state of the sTPN supplemented with an *age* clock evaluating the time elapsed since the beginning of the execution:

$$\text{aged-state} = \langle m, \tau_{age}, \underline{\tau} \rangle$$

where: $m \in \mathbb{N}^{|P|}$ is a marking; $T(m)$ is the set of transitions enabled by m and $|T(m)|$ its cardinality; $\underline{\tau} \in \mathbb{R}_0^{+|T(m)|}$ is a valuation for the vector of times-to-fire of enabled transitions; and $\tau_{age} \in \mathbb{R}_0^-$ is a value for the age clock encoding the opposite of the elapsed time.

We call *transient stochastic class* a continuous set of aged-states sharing a common marking and equipped with a multivariate *aged-state density function*.

Definition 3.2: A *transient stochastic class* is a triple

$$\langle m, D, f_{\langle \tau_{age}, \underline{\tau} \rangle} \rangle$$

where:

- m is a marking;
- $\langle \tau_{age}, \underline{\tau} \rangle$ is a random variable called *clock vector* composed of the scalar variable τ_{age} associated with the age clock, and by the vector $\underline{\tau} \stackrel{def}{=} \langle \tau_0, \tau_1, ..., \tau_{N-1} \rangle$ of the times-to-fire of transitions enabled by m, with τ_i denoting the time-to-fire of transition t_i;
- $f_{\langle \tau_{age}, \underline{\tau} \rangle}$ is the probability density function of $\langle \tau_{age}, \underline{\tau} \rangle$;
- D is the support of $f_{\langle \tau_{age}, \underline{\tau} \rangle}$.

Transient stochastic classes are associated with a succession relation extending in a probabilistic perspective the usual reachability relation among state classes [35].

Definition 3.3: We say that $\Sigma' = \langle m', D', f' \rangle$ is the successor of $\Sigma = \langle m, D, f \rangle$ through $t_0 \in T$, and we write

$\Sigma \stackrel{t_0}{\Longrightarrow} \Sigma'$, if, given that the marking of the model is m and its clock vector is a random variable distributed according to f over D, then:

- t_0 has a non-null probability to be the first transition to fire;
- under the assumption that t_0 is the first transition to fire, its firing yields the marking m' with a clock vector distributed over D' according to f'.

Enumeration of the relation $\stackrel{t}{\Rightarrow}$ requires a calculus for the identification of outgoing events and the derivation of successors.

Transition $t_0 \in T$ is an outgoing event from $\Sigma = \langle m, D, f \rangle$ iff t_0 is enabled by m and there is a non-null probability μ_0 that its time-to-fire is not higher than that of any other enabled transition:

$$\mu_0 = \int_{D \bigcap \{\langle x_{age}, \underline{x} \rangle | x_0 \leq x_n, \forall t_n \in T(m)\}} f(x_{age}, \underline{x}) dx_{age} d\underline{x} > 0 \tag{7}$$

If t_0 is a timed transition, then there is a null probability that the time-to-fire of t_0 is equal to that of any other enabled transition, and μ_0 is thus the probability that t_0 is the outgoing event from Σ. Whereas, if t_0 is an immediate transition, then the probability of t_0 is determined by the weights $\mathcal{C}$:

$$\mu_0 = \frac{\mathcal{C}(t_0)}{\sum_{t_i \in T^I(m)} \mathcal{C}(t_i)} \tag{8}$$

where $T^I(m)$ is the set of immediate transitions enabled by m.

Marking m' is derived in standard manner by moving tokens as usual in Petri Nets and D' is given as the successor of D as usual in symbolic state space analysis based on DBM zones [39]. Besides, derivation of the state density function f' extends [12], [15] so as to account for the fact that τ_{age} is never restarted and that it does not restrict the range of times-to-fire under which a transition may fire. The steps of the derivation are reported in [27].

IV. Using Transient Stochastic State Classes in Probabilistic Model Checking

A. Problem formulation

We are given a bounded time interval $[\alpha, \beta]$ with $0 \leq \alpha \leq \beta \in \mathbb{R}$, an initial transient class $\Sigma_0 = \langle M_0, D_0, f_0 \rangle$ where the age is concentrated on 0 (i.e. $f_0(x_{age}, x_0, ..., x_{N-1}) = \delta(x_{age}) \cdot g_0(x_0, ..., x_{N-1})$), and two state formulas ϕ_1 and ϕ_2 that classify the logical locations of the model through a Boolean combination of conditions on the marking (i.e., $\phi ::= d | \phi_1 \wedge \phi_2 | \neg\phi$ with $d : \mathbb{N}^{|P|} \rightarrow \{true, false\}$).

We address the problems of exact-, approximate- and threshold-probabilistic-model-checking [8]. In the *exact* formulation, we are interested in evaluating $P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta)$, which will denote the probability that, starting from a state sampled in Σ_0 according to f_0, the model reaches a location that satisfies ϕ_2 at a time in the interval $[\alpha, \beta]$ after visiting only states that satisfy ϕ_1:

$$P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta) \stackrel{def}{=} Pr\{\Sigma_0 \models \phi_1 Unt_{[\alpha,\beta]} \phi_2\} \tag{9}$$

In the *approximate* formulation, the problem is generalized accepting a safe accuracy in the evaluation. Specifically, given a tolerance $\epsilon \in \mathbb{R}_0^+$, we want to find an upper and a lower estimate on $P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta)$ that are closer than ϵ:

$$Pr^- \leq P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta) \leq Pr^+$$
$$Pr^+ - Pr^- \leq \epsilon$$

Finally, in the *threshold* formulation, the evaluation is cast into a decision form that amounts to checking whether the probability $P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta)$ is higher than a given threshold Δ.

All three problems can be cast in the formulation of [5], and all of them can be managed with non-substantial variations in the enumerative structure of the algorithm that we propose. To make notation easier, without loss of generality, we focus on the approximate model checking problem. Our formulation extends that of [1] as the set of selected behaviors can be bounded in terms of the time elapsed without making assumptions on the number of steps that the model is allowed to take before reaching the satisfaction of ϕ_2. A bound on the number of steps can easily be encompassed also in our formulation, but we avoid to describe it here, as a bound on the elapsed time is in general more significant in the modeling perspective. A less substantial extension regards the fact that we consider a bilateral time bound $[\alpha, \beta]$ rather than an upper bound $[0, \beta]$. Finally, as in [5], note that when $\alpha = \beta$ and $\phi_1 = true$ the problem amounts to the (possibly approximate) evaluation of the transient probability that the model satisfies a given condition ϕ_2 at a given time $\alpha = \beta$.

B. Enumeration of transient stochastic classes

The calculus of successor classes enjoys some properties that take a major relevance for the practical viability of the approach.

If the vector of times-to-fire $\underline{\tau}$ in the initial class Σ_0 is supported over a DBM zone D, then all its successors still have domains in DBM form, with the age ranging over negative values. This yields a compact partition of the state space in the sense that any two aged-states reached through the same firing sequence (i.e. through the same order of firings) will be enclosed in the same transient stochastic class, which would not hold for a partition based on regions.

According to [14], if density functions of transitions in the sTPN model are continuous functions, then the density functions of the reached transient stochastic classes are continuous piecewise functions with analytic representation over a finite partition of the support made of DBM subdomains. As reported in [14], this partition in subzones comprises the dominating factor of complexity in the practical implementation of the theory, as it yields an exponential explosion not only in the number of classes but also in their internal representation. The usage of DBM zones largely limits this explosion with respect to simple regions [1] and also opens the way to techniques that leverage on the continuity of the state density function to identify a global approximant over the entire DBM domain.

If transition density functions are monovariate expolynomial functions (i.e. $f_t(y) = \sum_{k=1}^{K} c_k y^{a_k} e^{-\lambda_k y}$ with $a_k \in \mathbb{N}$,

$c_k \in \mathbb{R}$, and $\lambda_k \in \mathbb{R}_{\geq 0}$), then density functions of reachable transient classes accept a closed form, which is a multivariate piecewise expolynomial function [17], [1], [14], [36]. This class of functions is closed with respect to all the operations required in the calculus of successors, which can be performed in a symbolic closed form efficiently implemented in the Oris tool and the Sirio Java API [11], [15].

Given an initial transient stochastic class Σ_0, the transitive closure of the succession relation $\stackrel{t_0}{\Longrightarrow}$ identifies a *transient stochastic tree*:

Definition 4.1: A *transient stochastic tree* is a tuple $\langle n_0, N, \Sigma, \Sigma_0, E, \mu \rangle$, where: N is a countable set of nodes; $n_0 \in N$ is the root node; Σ associates each node $n \in N$ with a transient stochastic class $\Sigma(n)$, with $\Sigma(n_0) = \Sigma_0$; $E \subseteq N \times T \times N$ is a set of edges such that $\langle n, t, n' \rangle \in E$ iff $\Sigma(n) \stackrel{t}{\Rightarrow} \Sigma(n')$. The edges of the tree can be labeled with a real function $\mu : E \to [0,1]$ returning the value of the probability of the transition $\Sigma(n) \stackrel{t}{\Rightarrow} \Sigma(n)$ derived according to Eqs.(7-8).

Nodes can be labeled with a real function $\eta : N \to [0,1]$ that returns 1 for the root node n_0, and for any other node n returns the product of the probabilities associated with the edges traversed in the path from the root to n, which we denote by $\rho(n)$.

According to Def. 3.3, for any given node n, the value returned by $\eta(n)$ is the probability that the model runs along the transition sequence $\rho(n)$. Moreover, if $\Sigma(n) = \langle m, D, f \rangle$, then f is the probability density function of the aged-states collected in $\Sigma(n)$ conditioned to the execution sequence $\rho(n)$. Thus, for any node n with $\Sigma(n) = \langle m, D, f \rangle$ and for any two values $\alpha \in \mathbb{R}_0^+$ and $\beta \in \mathbb{R}_0^+ \bigcup \{\infty\}$ with $\alpha \leq \beta$, the probability $P(n, \alpha, \beta)$ that the model completes the sequence that reaches the transient class $\Sigma(n)$ within a time falling in the interval $[\alpha, \beta]$ is equal to

$$P(n, \alpha, \beta) = \eta(n) \cdot \pi(\Sigma(n), \alpha, \beta) \tag{10}$$

where

$$\pi(\Sigma(n), \alpha, \beta) = \int_{D \bigcap \alpha \leq -x_{age} \leq \beta} f(x_{age}, \underline{x}) dx_{age} d\underline{x} \tag{11}$$

represents the probability that the class $\Sigma(n)$ is entered in the interval $[\alpha, \beta]$ given that the model performs the transition sequence $\rho(n)$.

C. Decision algorithm

Algorithm-A reported in Fig. 1 evaluates the probability $P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta)$ following the usual pattern of forward reachability analysis, which in this case relies on two invariants: a set Γ contains all the transient stochastic classes that have been reached in the analysis but not yet processed, each associated with the probability $\eta(\Sigma) \in (0,1]$ to be reached from the initial class; two probability values P_{OK} and P_{KO} provide lower bounds on the probability that the until formula is satisfied and on the probability that it is not satisfied, so as to define a safe estimate on $P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta)$:

$$P_{OK} \leq P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta) \leq 1 - P_{KO}$$

While Γ is not empty and $P_{OK} + P_{KO} \leq 1 - \epsilon$, the algorithm repeatedly selects and removes a class $\Sigma = \langle m, D, f \rangle$ from the frontier Γ (line 6); according to whether the marking m satisfies the conditions ϕ_1 and ϕ_2, it evaluates the probability measure of the subsets of Σ that satisfy the until formula and of the subsets that certainly do not satisfy it, assigning them to P_{OK} and P_{KO}, respectively; finally, if ϕ_1 holds, it computes the successor classes of the subset of Σ that cannot be assigned either to P_{OK} or to P_{KO}, and adds them to Γ.

```
1   η(Σ0) = 1
2   Γ = {Σ0}
3   P_OK = 0
4   P_KO = 0
5   while Γ ≠ ∅ and P_OK + P_KO < 1 − ε
6       select and remove a class Σ = ⟨m, D, f⟩ from Γ
7       if m ⊨ ¬φ1 ∧ ¬φ2
8           P_KO = P_KO + η(Σ)
9       elseif m ⊨ ¬φ1 ∧ φ2
10          P_KO = P_KO + η(Σ_[0,α]) + η(Σ_[β,+∞])
11          P_OK = P_OK + η(Σ_[α,β])
12      elseif m ⊨ φ1 ∧ ¬φ2
13          P_KO = P_KO + η(Σ_[β,+∞])
14          Γ = Γ ∪ SUCCESSORS(Σ_[0,β])
15      elseif m ⊨ φ1 ∧ φ2
16          P_KO = P_KO + η(Σ_[β,+∞])
17          P_OK = P_OK + η(Σ_[α,β])
18          Γ = Γ ∪ SUCCESSORS(Σ_[0,α])
```

Fig. 1. Algorithm-A: evaluates $P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta)$ with approximation $\epsilon \geq 0$. At lines 14 and 18 the steps that add elements to the list of pending classes take relevance for the issue of termination.

Given a transient state class $\Sigma = \langle m, D, f \rangle$, the subset $\Sigma_{[\alpha,\beta]} = \langle m, D_{[\alpha,\beta]}, f_{[\alpha,\beta]} \rangle$ is computed as

$$D_{[\alpha,\beta]} = D \cap \{\alpha \leq -\tau_{age} \leq \beta\}$$
$$f_{[\alpha,\beta]}(x_{age}, \underline{x}) = \frac{f(x_{age}, \underline{x})}{\pi(\Sigma, \alpha, \beta)}$$
$$\eta(\Sigma_{[\alpha,\beta]}) = \eta(\Sigma) \cdot \pi(\Sigma, \alpha, \beta).$$

Successors of $\Sigma_{[\alpha,\beta]}$, indicated as $\text{SUCCESSORS}(\Sigma_{[\alpha,\beta]})$, are derived following the usual enumeration of transient stochastic state classes. Each successor Σ' is associated with a probability mass $\eta(\Sigma') = \eta(\Sigma_{[\alpha,\beta]}) \cdot \mu_0$, where μ_0 is the probability of the outgoing event from $\Sigma_{[\alpha,\beta]}$ to Σ', according to Eqs.(7-8).

D. Termination

Termination of Algorithm-A jointly involves the finiteness of reachable markings and the absence of Zeno behaviors that can engage infinite events without letting time diverge beyond α or β. We focus here on the latter aspect, under the assumption that the number of markings that can be reached within the bound β is finite, which better compares with models with a finite number of logical locations. Various results could be generalized to the case of infinite markings under the assumption that the model does not include inhibitor arcs, which in turn guarantees that the set of enabled immediate transitions is monotonic with respect to the marking.

It is intuitive that different conditions hold whether $\epsilon = 0$ or $\epsilon > 0$, i.e. whether the evaluation aims at an exact or

an approximate value of the probability $P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta)$, as the latter case rules out unfair null-probability infinite behaviors [4].

In the case $\epsilon = 0$, for every feasible behavior ρ accepted by the model, the tree includes a path with edges labeled by the same transitions of ρ. Algorithm-A thus turns out to be non-terminating iff it reaches a class that includes at least one state allowing a behavior that can reach in zero-time the same marking without visiting any intermediate marking with terminating conditions of state formulas. The concept is formalized in the following Theorem.

Theorem 4.1: With $\epsilon = 0$, Algorithm-A does not terminate if and only if there exists a node n_a that enables a sequence $\rho = t_1 \rightarrow t_2 \rightarrow ... \rightarrow t_N$ reaching a node n_b such that all the following conditions hold:

(i) n_a and n_b have the same marking;
(ii) $\Sigma(n_a)$ includes aged-states that satisfy $-\tau_{age} < \alpha$ and all the markings visited from n_a to n_b satisfy ϕ_1, or $\Sigma(n_a)$ includes aged-states that satisfy $\alpha \leq -\tau_{age} \leq \beta$ and all the markings visited from n_a to n_b satisfy $\phi_1 \wedge \neg\phi_2$;
(iii) $EFT(t_n) = 0$ for every transition t_n in the sequence ρ.

Proof - if: Let n_a and n_b be two nodes with the same marking, and let $EFT(t_n) = 0$ for every transition t_n along the sequence $\rho = t_1 \rightarrow t_2 \rightarrow ... \rightarrow t_N$ from n_a to n_b. Assume that none of the classes visited by ρ is terminal, either because it satisfies $\phi_1 \wedge \neg\phi_2$ and includes states with $-\tau_{age} \leq \beta$ or because it satisfies $\phi_1 \wedge \phi_2$ and includes states with $-\tau_{age} < \alpha$. Let s_a be a state in $\Sigma(n_a)$ with the minimum accepted value of $-\tau_{age}$, and let s'_a be the state that gives a null time-to-fire to each enabled transition t_n that will persist until the firing along ρ, while agreeing with s_a on every other clock (including the age). The state s'_a can be proven to be in Σ_a as every transition t_n that will fire along ρ was newly enabled (either in Σ_a or in any ancestor) with a firing domain initially set to the interval $[0, LFT(t_n)]$. Consider now the infinite behavior of the model that starts from state s'_a, always gives a null time-to-fire to every transition included in ρ, and resolves every choice among transitions with equal time-to-fire according to the order of ρ. This behavior indefinitely repeats ρ without ever advancing the age or reaching a terminal state, and thus yields an infinite sequence of nodes in the transient stochastic tree enumerated by Algorithm-A.

Proof - only if: Let ρ^* be an infinite sequence of transient classes in the tree. Since all non-terminal classes in the tree include only non conclusive states, all classes visited by ρ^* satisfy ϕ_1 and none of them include states satisfying ϕ_2 with $-\tau_{age} \in [\alpha, \beta]$. Since the model accepts a finite number of markings, a marking m_* is visited infinitely often, and there exists an infinite number of (possibly equal) subsequences of ρ^* that start and terminate on marking m^*. Moreover, since all classes visited by ρ^* include states with $-\tau_{age} \leq \beta$, there exists an infinite number of subsequences of ρ^* that traverse only transitions t_n with $EFT(t_n) = 0$. ■

Theorem 4.1 provides a test that can be embedded in Algorithm-A so as to make it terminating, either because the analysis of the transient tree is successfully completed or because it is shown that an exact evaluation with $\epsilon = 0$ cannot be performed within a finite number of classes.

Whether the objective of evaluation is relaxed by allowing some error threshold $\epsilon > 0$, the hypothesis of Theorem 4.1 is sufficient to guarantee termination, but it is not necessary as behaviors with probability lower than the approximation threshold ϵ are possible. Using a more relaxed (yet not necessary) condition, termination can be obtained by an extension of Algorithm-A ruling out cyclic executions that are necessarily immediate. This relies on the concept of *vanishing* and *time-blocking* classes.

Definition 4.2: We call *vanishing* a transient class $\Sigma = \langle m, D, f \rangle$ where there is at least one enabled transition t_n that is bound to fire immediately ($\exists t_n \in T(m)$ such that $\forall \underline{\tau} \in D,\ \tau_n = 0$). Conversely, we call *tangible* a transient class that is not vanishing.

Since the model does not include transitions with $EFT = LFT > 0$, it can be easily proven that a class is vanishing if and only if its marking enables at least one immediate transition, and that all non-null probability outgoing events of a vanishing class are immediate transitions. Also note that according to Eq.(8), in a vanishing class, discrete transition probabilities of outgoing events are determined by the marking, independently from the state density distribution and its support. Finally, note that the firing of an immediate transition does not advance the age: if $\Sigma \stackrel{t_0}{\Rightarrow} \Sigma'$ and t_0 is immediate, then Σ' contains an aged-state s' iff Σ contains an aged-state s which agrees with s' on the age and on the time-to-fire of every transition that is persistent at the firing of t_0.

Definition 4.3: We call *time-blocking* a transient class that is vanishing and whose descendants are all vanishing.

From the definition follows that, with probability 1, time does not advance any further as soon as a time-blocking class is reached. According to this, if Algorithm-A reaches a time-blocking class it will never terminate. The problem can be fixed by modifying Algorithm-A so as to detect time-blocking classes and avoid their explicit enumeration, performing the transient analysis of a subordinated DTMC. To this end, Algorithm-B reported in Fig. 2 provides a terminating procedure to decide whether a vanishing class is time-blocking.

The partial correctness of Algorithm-B is trivial if $\overline{\Sigma}$ is decided not to be time-blocking (line 5) as the algorithm identifies in a constructive manner a non-vanishing class reached from $\overline{\Sigma}$. Whereas, in case $\overline{\Sigma}$ is decided to be time-blocking (line 9), it is sufficient to note that when Ω is empty,

```
IS-TIME-BLOCKING(Σ̄)
1  Ω = {Σ̄}
2  while Ω ≠ ∅
3       select and remove a class Σ from Ω
4       if ∃Σ' ∈ SUCCESSORS(Σ) : Σ' is tangible
5            return FALSE
6       else Ω = Ω ∪ {Σ' ∈ SUCCESSORS(Σ) : the path
7                 from Σ̄ to Σ doesn't include any class
8                 with a marking equal to that of Σ'}
9  return TRUE
```

Fig. 2. Algorithm-B: decides whether $\overline{\Sigma}$ is a time-blocking class.

all possible markings that can be reached from $\overline{\Sigma}$ have already been identified and they all turned out to be vanishing. Besides, termination is guaranteed within a constant number of steps dependent on the finite number of reachable markings and transitions in the model. Note that when $\overline{\Sigma}$ turns out to be a time-blocking class, the subsequent behavior of the model is characterized as a Discrete Time Markov Chain (DTMC), whose states are the markings identified by Algorithm-B and whose transition probabilities are completely determined by these markings according to Eq.(8).

Algorithm-B can be integrated into Algorithm-A so as to obtain Algorithm-C (which we do note write explicitly) that avoids the expansion of time-blocking classes: at lines 14 and 18, each successor Σ' is added to Γ only if it is tangible or if it is vanishing but Algorithm-B guarantees that it is not a time-blocking class. If a successor Σ' turns out to be time-blocking, then its probability can be allocated to P_{OK} and P_{KO} through transient analysis of the DTMC starting from Σ'. By construction, Σ' does not contain any state with age higher than β (i.e. $\forall(m, \tau_{age}, \underline{\tau}) \in \Sigma, \ -\tau_{age} \leq \beta$). Besides, since time does not advance after a time-blocking class has been entered, the probability measured over aged-states of Σ' that satisfy $\tau_{age} < \alpha$ can be assigned to P_{KO} as no descendant of any of these states can ever reach the age α. Whereas, the probability of the states that satisfy $\alpha \leq -\tau_{age} \leq \beta$ is assigned either to P_{OK} or P_{KO} depending on the satisfaction of ϕ_1 and ϕ_2 in the markings of vanishing classes enumerated during the time-block detection algorithm: if none of the classes enumerated in the time-block detection algorithm satisfy the condition ϕ_2, then the entire measure of the subset of Σ' that satisfies $\alpha \leq -\tau_{age} \leq \beta$ is assigned to P_{KO}; if all classes enumerated in the time-block detection algorithm satisfy the condition ϕ_2, then the entire measure of the subset of Σ that satisfies $\alpha \leq -\tau_{age} \leq \beta$ can be assigned to P_{OK}; in the remaining case, the assignment to P_{OK} and P_{KO} is carried out through a first passage analysis of the DTMC starting from Σ'.

When time-blocking classes have been removed from the enumeration, Algorithm-C always terminates either because it reaches a time block, or because the time bound β is exceeded with probability $1 - \epsilon$:

Theorem 4.2: $\forall \epsilon > 0$, Algorithm-C always terminates.

Proof: Ab absurdo, if the algorithm does not terminate, $\forall N \in \mathbb{N}$ the union of transient classes reached after N steps still contains non-conclusive states (i.e. states reached traversing ϕ_1-states without exceeding β or satisfying ϕ_2 after α).

According to this, there exists an infinite path ρ^* in the transient tree that does not exceed the time limit of β.

Since the starting node of ρ^* can not be time-blocking, with probability 1, ρ^* eventually traverses some transition t with $LFT(t)$. This result is easily lifted to show that ρ^* traverses infinite transitions with $LFT > 0$.

Since the model includes a finite number of transitions, there is some transition t_* with $LFT(t_*) > 0$ that is fired infinitely often along ρ.

The time elapsed between two subsequent firings of t_* is not lower than the time from newly enabling to firing of t_*, which by construction is a random variable X_* distributed over $[EFT(t_*), LFT(t_*)]$ according to F_{t_*}.

The overall duration of ρ^* is thus lower bounded by an infinite sum of i.i.d. variables distributed as X_*. This sum diverges beyond any finite limit (and in particular beyond β with probability 1. ■

V. State space reduction through look-ahead of reachable non-deterministic zones

The model checking technique of Algorithm-A enables a parsimonious approach to the state-space enumeration of sTPN models; in fact, the enumeration is restricted only to classes giving a non-null contribution to P_{OK} or P_{KO}. However, a major complexity is still determined by the piecewise partitioning of state density functions that arises during successive evaluations of transient state classes. To mitigate this problem, the approximation technique of [14] can be used to keep partitioning limited and reduce complexity at the price of obtaining an approximate solution. An alternative approach, that in many cases is able to strongly accelerate convergence of Algorithm-A, is based on a non-deterministic analysis of the model, in order to anticipate detection of transient classes contributing to P_{OK} and P_{KO} without the need to compute the corresponding state density functions.

In more detail, this look-ahead technique is based on the generation of a *transient non-deterministic tree* obtained through the transitive closure of a non-deterministic succession relation among *transient non-deterministic state classes*.

All these concepts can be regarded as non-deterministic reductions of those of *transient stochastic class* (Def. 3.2), stochastic succession relation (Def. 3.3) and *transient stochastic tree* (Def. 4.1). In particular, a *transient non-deterministic state class* is a pair $\langle m, D\rangle$, where m is a marking and D is the DBM domain encoding admissible values for the non-deterministic variable $\langle \tau_{age}, \underline{\tau}\rangle$ made of the age clock τ_{age} and the vector $\underline{\tau}$ of times-to-fire of transitions enabled by m. The succession relation among these classes is obtained through a straightforward transient extension of that introduced in [39] for Time Petri Net models. Finally, a *transient non-deterministic tree* is a tuple $\langle n_0, N, S, S_0, E\rangle$ where n_0, N and E are defined as in Def. 4.1, while S associates each node n with a transient non-deterministic class $S(n)$ with $S(n_0) = S_0$.

According to these definitions, a transient non-deterministic tree $T = \langle n_0, N, S, S_0, E\rangle$ is *isomorphic* to the corresponding transient stochastic tree $T' = \langle n'_0, N', \Sigma, \Sigma_0, E', \mu\rangle$, i.e. a bijection f can be defined between the vertex sets of T' and T so that $\Sigma(i)$ is a successor of $\Sigma(j)$ in T' if and only if $S(h) = f(\Sigma(i))$ is a successor of $S(k) = f(\Sigma(j))$ in T.

Moreover, corresponding classes share the same marking m and the same domain of timers D, i.e. if $\Sigma(i) = \langle m, D, f_{\langle \tau_{age}, \underline{\tau}\rangle}\rangle$ then $f(\Sigma(i)) = S(j) = \langle m, D\rangle$. This implies that any formula ϕ that is verified by the marking of $\Sigma(i)$ is also verified by that of $f(\Sigma(i)) = S(j)$, and that any assertion regarding the domain of timers D is valid on both $\Sigma(i)$ and $S(j)$.

From the perspective of the model checking problem, this remark enables a classification of each node in the transient

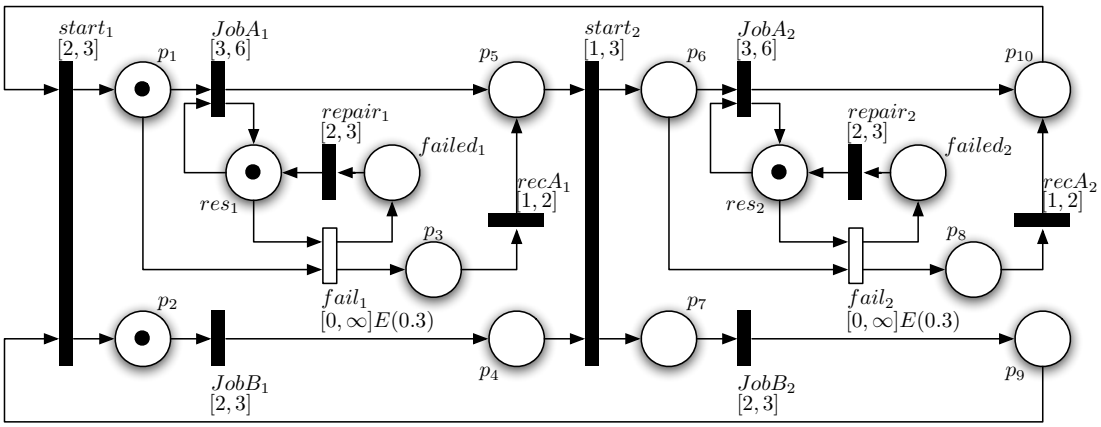


Fig. 3. sTPN model of two production cells. Transitions are labeled with their firing interval; E(0.3) denotes an exponential transition with rate 0.3, while all other transitions are uniformly distributed over their interval.

non-deterministic tree as an "OK"/"KO"/"unknown" node, based on the satisfaction of ϕ_1 and ϕ_2 and on the possible range of the age timer τ_{age} with respect to the analysis time interval $[\alpha, \beta]$. Specifically, an "OK" node is a node that only contributes to P_{OK} and whose descendants are all labeled as "OK"; a "KO" node is a node that only contributes to P_{KO} and whose descendants are all labeled as "KO"; all other nodes are classified as "unknown".

Once all nodes have been classified, the resulting labeled transient non-deterministic tree can be used to avoid the evaluation of successors in Algorithm-A (lines 14 and 18) when a class Σ is encountered such that the corresponding non-deterministic class $f(\Sigma)$ is classified as "OK" or "KO". In such cases, the whole probability mass $\eta(\Sigma)$ is assigned to P_{OK} or P_{KO}, respectively, and successors of Σ are not evaluated because their contribution to P_{OK} or P_{KO} is already included in $\eta(\Sigma)$.

Hence, enumeration of the underlying non-deterministic model (requiring a negligible computational effort with respect to Algorithm-A) is exploited to dynamically prune the transient stochastic tree, resulting in a significant performance boost of the model checking algorithm.

VI. Computational Experience

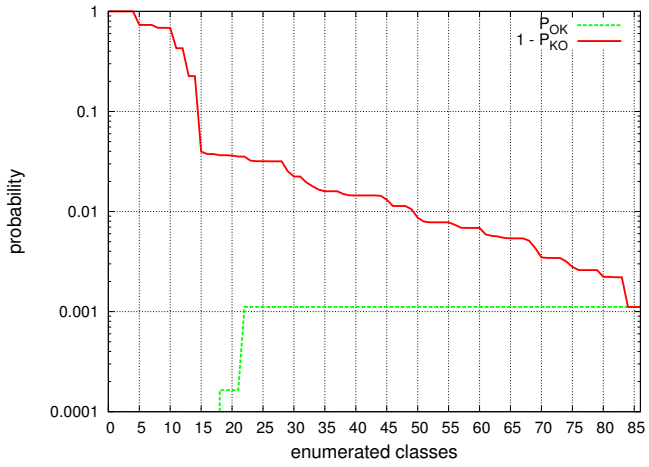
We illustrate the results obtained through a preliminary implementation in the Oris tool. We refer to the model in Fig. 3, which was introduced in [15] in the context of steady state analysis. The model represents two production cells that repeatedly pass control to each other through transitions $start_1$ and $start_2$. Each cell carries out two parallel activities with uniform duration, named $JobA$ and $JobB$. $JobA$ requires a resource called res which may fail according to an exponential distribution with rate 0.3. If failure occurs, $JobA$ is replaced by a recovery activity $recA$, and a repair action is started. Both recovery and repair activities take uniformly distributed durations.

We evaluate here the probability to reach at a time in the interval $[5, 7]$ a state in which both resources are simultaneously failed, without visiting any state subsequent to the success in the execution of $JobA$ in the first cell. To this end, we assume $\alpha = 5$ and $\beta = 7$, with state formulas $\phi_2 = failed1 \wedge failed2$ and $\phi_1 = \neg(p5 \wedge res1) = \neg(p5) \vee \neg(res1)$. Note that ϕ_1 is the complement of the logical condition reached after a successful execution of $JobA_1$, which could better fit in an action-based formulation of the logic and the decision algorithm.

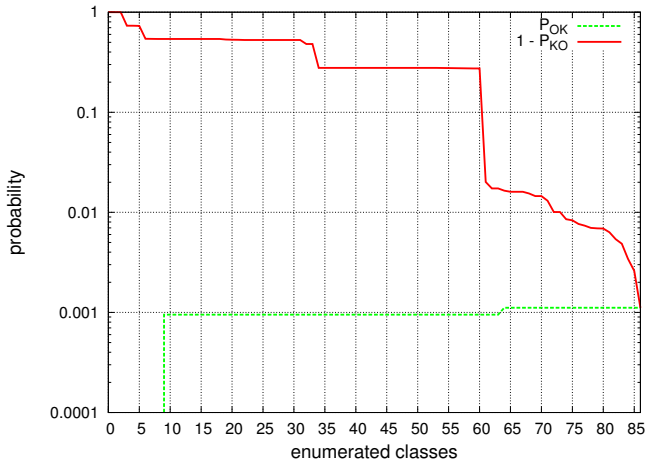
Fig. 4 plots the estimates $1 - P_{KO}$ and P_{OK} as a function of the number of enumerated transient classes. In particular, Fig. 4(a), 4(b) and 4(c) plot the results obtained enumerating classes with a FIFO (breadth-first), LIFO (depth-first) and η-priority policy, respectively. The latter implements the queue Γ as a priority queue, ordering classes Σ within Γ according to their probability mass $\eta(\Sigma)$. Intuitively, the policy of selection of the next class to expand in the enumeration process (line 6 of Algorithm-A) gives space to convergence optimizations through appropriate heuristics. In this case, for instance, Breadth-First initially accelerates the decrease of $1-P_{KO}$ as it first explores classes with higher probability η but containing states with age under the lower bound α; Priority Policy further boosts this process, as it explicitly prioritize classes with higher η.

The adoption of the look-ahead technique of Sect. V determines a strong reduction in the complexity of the model checking algorithm. To illustrate this result, we repeat the experiment on the model of Fig. 3 evaluating the probability of state formulae ϕ_1 and ϕ_2 within the interval $[5, 7]$. Results are shown in Fig. 5: the number of enumerated transient stochastic classes falls from 87 in the standard approach to 26 in the look-ahead approach.

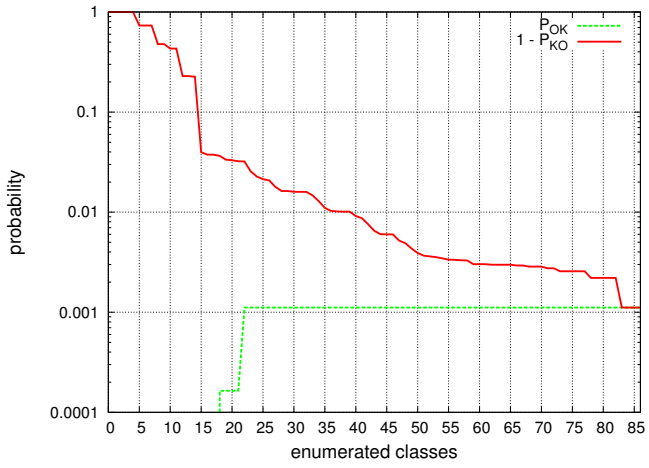
Enlarging the analysis interval to $[5, 18]$ so as to include an entire activity cycle of both production cells (i.e. to cover at least an entire time period until the first firing of $start_1$), the number of enumerated non-deterministic classes increases to 5240, a number that would make the standard approach unviable. However, tree pruning based on the look-ahead technique reduces the number of enumerated classes to 26. It is worth noting that this is exactly the same result obtained for the $[5, 7]$ interval; the fact is due to the timing structure of the model, causing this particular $\phi_1 Unt_{[\alpha,\beta]} \phi_2$ formula to be verifiable only within a narrow time window (that excludes the interval between 7 and 18). The look-ahead technique enables



(a)



(b)



(c)

Fig. 4. Logarithmic plots of $1 - P_{KO}$ (continuous line) and P_{OK} (dotted line) converging to $P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta)$, as a function of the number of classes enumerated with FIFO (a), LIFO (b) and η-priority policy (c).

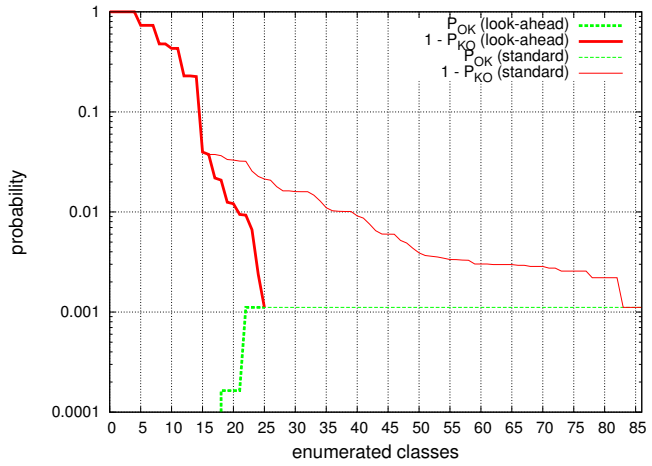


Fig. 5. Logarithmic plots of $1 - P_{KO}$ (continuous lines) and P_{OK} (dotted lines) converging to $P(\Sigma_0, \phi_1, \phi_2, \alpha, \beta)$, as a function of the number of classes enumerated with the standard approach (thin lines) and the look-ahead approach (thick lines). In both cases, nodes are extracted from queue Γ following a η-priority policy.

early detection of such conditions, so as to avoid the much more complex (and, in this case, useless) enumeration of the transient stochastic tree.

VII. Conclusions and Future Directions

The method of stochastic state classes provides an analytical approach to the evaluation of non-Markovian models with multiple generally distributed transitions possibly supported over bounded domains. To this end, the method embeds a discrete time chain in the underlying stochastic process by sampling the marking and the distribution of times to fire of generally distributed transitions after each firing. In so doing, the analysis results in the traversal of a graph of so-called state classes, each encoding a multivariate probability density function supported over a DBM zone. While the method was initially targeted to steady state analysis [12], [15], the introduction of an age clock that supplements process samples enabled its extension to the evaluation of transient behavior [27].

We showed here how transient stochastic state classes can be effectively cast into a probabilistic time-bounded model checking algorithm, enabling exact or approximate analysis for a large class of processes beyond the limits of the enabling restriction and encompassing GSMPs where active timers are constrained to evolve with unitary speed.

This achieves a twofold advancement with respect to the literature of probabilistic model checking of non-Markovian models [1]. On the one hand, the proposed technique supports the evaluation of a probabilistic until operator with a bound expressed on the elapsed continuous time rather than on the discrete number of occurred transitions. This better fits the needs of the application in the context of real time systems and relaxes the assumptions required to guarantee termination in exact or approximated analysis.

On the other hand, traversal of reachable transient stochastic state classes results in a coarse partition of the state space

based on zones rather than regions. This largely reduces the complexity of the analysis, which here involves not only the enumeration of supports but also the symbolic evaluation of probability density functions. Moreover, this enables an effective look-ahead strategy that leverages on the non-deterministic state class graph of reachable zones to prune classes for which probability density functions don't need to be computed.

Results achieved so far permit now to target the integration with generalized regenerative theory developed in [27] and the experimentation of the impact of different heuristics driving the order of expansion in the traversal of the tree of transient stochastic state classes.

REFERENCES

[1] R. Alur and M. Bernadsky. Bounded model checking for gsmp models of stochastic real-time systems. In *9th Intl.Conf.on Hybrid Systems: Computation and Control*, volume 39, pages 19–33. Springer-Verlag, 2006.

[2] R. Alur, C. Courcoubetis, and D. L. Dill. Model-checking for probabilistic real-time systems. In *In Automata, Languages and Programming: Proceedings of the 18th ICALP, Lecture Notes in Computer Science 510*, pages 115–126. Springer, 1991.

[3] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[4] C. Baier, N. Bertrand, P. Bouyer, T. Brihaye, and M. Groesser. Almost-sure model checking of infinite paths in one-clock timed automata. In *LICS '08: Proceedings of the 2008 Annual Symposioum on Logic in Computer Science*, Washington, DC, USA, 2008.

[5] C. Baier, B. Haverkort, H. Hermanns, and J.-P. Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE Trans. Softw. Eng.*, 29(6):524–541, 2003.

[6] J. Bengtsson and W. Yi. Timed automata: Semantics, algorithms and tools. In *Concurrency and Petri Nets*. LNCS 3098, 2004.

[7] B. Berthomieu and M. Diaz. Modeling and verification of time dependent systems using Time Petri Nets. *IEEE Trans. on Soft. Eng.*, 17(3), March 1991.

[8] N. Bertrand, P. Bouyer, T. Brihaye, and N. Markey. Quantitative model-checking of one-clock timed automata under probabilistic semantics. In *QEST '08: Proceedings of the 2008 Fifth International Conference on Quantitative Evaluation of Systems*, pages 55–64, Washington, DC, USA, 2008.

[9] A. Bobbio, A. Puliafito, and M. Tekel. A modeling framework to implement preemption policies in non-Markovian SPNs. *IEEE Transactions on Software Engineering*, 26(1):36–54, 2000.

[10] A. Bobbio and M. Telek. Markov Regenerative SPN with non-overlapping activity cycles. *Int. Computer Performance and Dependability Symp. - IPDS95*, pages 124–133, 1995.

[11] G. Bucci, L. Carnevali, L. Ridi, and E. Vicario. Oris: a tool for modeling, verification and evaluation of real-time systems. *accepted for publication in STTT, Int.Journal on Software Tools for Technology Transfer*, 2010.

[12] G. Bucci, R.Piovosi, L. Sassoli, and E. Vicario. Introducing probability within state class analysis of dense time dependent systems. *Proc. of the 2^{nd} Int. Conf. on the Quant. Evaluation of Sys.(QEST)*, September 2005.

[13] L. Carnevali, J. Giuntini, and E. Vicario. A symbolic approach to quantitative analysis of preemptive real-time systems with non-markovian temporal parameters. In *Proceedings of the 6th Int. Conf. on Performance Evaluation Methodologies and Tools*, 2011.

[14] L. Carnevali, L. Grassi, and E. Vicario. State-density functions over DBM domains in the analysis of non-Markovian models. *IEEE Trans. on Soft. Eng.*, 35:178 – 194, March-April 2009.

[15] L. Carnevali, L. Sassoli, and E. Vicario. Using stochastic state classes in quantitative evaluation of dense-time reactive systems. *IEEE Trans. on Soft. Eng.*, Nov. 2009.

[16] H. Choi, V. G. Kulkarni, and K. Trivedi. Markov Regenerative Stochastic Petri Nets. *Perf. Eval.*, 20:337–357, 1994.

[17] G. Ciardo, R. German, and C. Lindemann. A characterization of the stochastic process underlying a Stochastic Petri Net. *IEEE Trans. Softw. Eng.*, 20(7):506–515, 1994.

[18] G. Ciardo and A. S. Miner. Smart: The stochastic model checking analyzer for reliability and timing. In *1st International Conference on Quantitative Evaluation of Systems (QEST 2004), 27-30 September 2004, Enschede, The Netherlands*, pages 338–339, 2004.

[19] E.M. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.

[20] D.R. Cox. The analysis of non-markovian stochastic processes by the inclusion of supplementary variables. *Proceedings of the Cambridge Phylosophical Society*, 51:433–440, 1955.

[21] D. Dill. Timing assumptions and verification of finite-state concurrent systems. *Proc.Workshop on Computer Aided Verification Methods for Finite State Systems*, 1989.

[22] E.Cinlar. *Introduction to Stochastic Processes*. Prentice Hall, Englewood Cliffs, 1975.

[23] R. German and C. Lindemann. Analysis of Stochastic Petri Nets by the method of supplementary variables. In *Performance Evaluation*, volume 20, pages 317–335, 1994.

[24] P. W. Glynn. A GSMP formalism for discrete-event systems. *Proceedings of the IEEE*, 77:14–23, 1989.

[25] P. J. Haas and G. S. Shedler. Stochastic Petri net representation of discrete event simulations. *IEEE Transactions on SW Engineering*, 15(4):381–393, 1989.

[26] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6:102–111, 1994.

[27] A. Horváth, L.Ridi, and E.Vicario. Transient analysis of non-Markovian processes with overlapping activity cycles using transient stochastic classes. In *7th International Conference on the Quantitative Evaluation of SysTems (QEST10)*, 2010.

[28] A. Horváth and E. Vicario. Aggregated stochastic state classes in quantitative evaluation of non-Markovian stochastic Petri nets. In *Proc. of 6th International Conference on the Quantitative Evaluation of Systems (QEST)*, Budapest, Hungary, Sept 2009.

[29] V. G. Kulkarni. *Modeling and analysis of stochastic systems*. Chapman and Hall, 1995.

[30] M. Kwiatkowska, G. Norman, and D. Parker. Probabilistic symbolic model checking with PRISM: A hybrid approach. *International Journal on Software Tools for Technology Transfer (STTT)*, 6(2):128–142, 2004.

[31] M. Kwiatkowska, G. Norman, R. Segala, and J. Sproston. Verifying quantitative properties of Continuous Probabilistic Timed Automata. *Proc. CONCUR'00*, 1877:123–137, 2000.

[32] O. Maler, K.G. Larsen, and B.H. Krogh. On Zone-Based Analysis of Duration Probabilistic Automata. In *INFINITY, ser. EPTCS*, volume 39, pages 33–46, 2010.

[33] P. Merlin and D.J. Farber. Recoverability of communication protocols. *IEEE Trans.on Communications*, 24(9), 1976.

[34] M. F. Neuts. *Matrix geometric solutions in stochastic models*. Johns Hopkins University Press, 1981.

[35] W. Penczek and A. Polrola. Specification and model checking of temporal properties in Time Petri Nets and Timed Automata. *Proceedings of the 25^{th} Int. Conf on Application and Theory of Petri Nets, ICATPN2004*, June 2004.

[36] R.A. Sahner and K.S. Trivedi. Reliability modeling using SHARPE. *Reliability, IEEE Transactions on*, 36(2):186–193, 2009.

[37] M. Telek and A. Horvath. Supplementary variable approach applied to the transient analysis of age-MRSPNs. In *Proc. Int. Performance and Dependability Symp.*, pages 44–51, 1998.

[38] M. Y. Vardi. Automatic verification of probabilistic concurrent finite state programs. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 327–338, Washington, DC, USA, 1985.

[39] E. Vicario. Static analysis and dynamic steering of time dependent systems using Time Petri Nets. *IEEE Trans. on Soft. Eng.*, August 2001.

[40] Enrico Vicario. Engineering the usability of a visual formalism for real-time temporal logic. *J. Vis. Lang. Comput.*, 12(6):573–599, 2001.

[41] H. Younes and R. G. Simmons. Solving generalized semi-Markov decision processes using continuous phase-type distributions. In *AAAI'04: Proceedings of the 19th national conference on Artifical intelligence*, pages 742–747. AAAI Press / The MIT Press, 2004.