Discrete-event simulation of fluid stochastic Petri nets*

Gianfranco Ciardo[1] ciardo@cs.wm.edu
David Nicol[2] nicol@cs.dartmouth.edu
Kishor S. Trivedi[3] kst@egr.duke.edu

[1] Dept. of Computer Science, College of William and Mary, Williamsburg, VA 23187
[2] Dept. of Computer Science, Dartmouth College, Hanover, NH 03755
[3] CACC, Dept. of Electrical and Computer Eng., Duke University, Durham, NC 27708

Abstract

The purpose of this paper is to describe a method for the simulation of the recently introduced fluid stochastic Petri nets. Since such nets result in rather complex system of partial differential equations, numerical solution becomes a formidable task. Because of a mixed (discrete and continuous) state space, simulative solution also poses some interesting challenges, which are addressed in the paper.

Keywords: Discrete-event simulation, continuous system simulation, stochastic Petri nets.

1 Introduction

Discrete-event dynamic systems are commonplace, and discrete-state models are normally used to study their behavior. Ordinary and stochastic Petri nets, for example, provide a convenient and concise method of describing these systems [4, 8, 11, 22, 25]. However, the underlying state space of these models tends to be extremely large in practical modeling applications, often forcing us to seek approximate solution methods. An example is the fluid flow approximations in performance analysis of queueing systems [5, 15, 21], where a large number of discrete entities is modeled as a single continuous variable.

On the other hand, hybrid systems, that is, systems having both discrete and continuous components that evolve over time, have received increasing attention in the last few years, due to the ubiquitous trend of employing digital controllers in traditionally analog environments such as power generators, chemical plants, or water distribution systems.

Thus, it is natural to extend Petri nets so that they can have a hybrid state-space that enables the study of both otherwise discrete systems through fluid approximations, and of truly hybrid systems. In addition, the behavior of such models can be deterministic or stochastic. Various formalisms falling in this category have appeared. Timed continuous Petri nets, whose places are marked in a continuous way, and, more recently, hybrid Petri nets, which also contain ordinary places containing an integral number of tokens, have been introduced by David [14, 13]. Fluid stochastic Petri nets (FSPNs) have been introduced by Trivedi and Kulkarni [24], and considerably enhanced in [18]; these included immediate and exponentially distributed firing times for the transitions, as in the Generalized Stochastic Petri Nets GSPNs [3].

In FSPNs, as in hybrid Petri nets, the places of the Petri net are partitioned into two classes, one containing a nonnegative (integer) number of tokens, just as in ordinary Petri nets, the other containing a nonnegative (real) level of fluid. However, initial definitions of FSPNs posed severe restrictions on the semantic behavior of these nets, to make the models analytically tractable. The numerical solution algorithms proposed in [24, 18] are applicable only when the interactions between the discrete and continuous portion of the net satisfy fairly strong assumptions.

We observe that, even before performing a numerical study of the performance or reliability of a system, answers to various "logical" questions are often required, such as "is the system bounded?", or "does it have a home state?". Unfortunately, the decidability of one of the most natural and important analysis questions, "can a particular state be reached starting from a given initial state?" depends on the type of model. For example, reachability is decidable for ordinary Petri nets [6], but not for Turing-equivalent formalisms, such as Petri nets with transition priori-

*This research was partially supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-19480. Additionally, D. Nicol was supported in part by the National Science Foundation under grants CCR-9201195 and NCR-9527163, and K. Trivedi was supported in part by the National Science Foundation under grant NSF-EEC-94-18765.

ties or inhibitor arcs [1, 17]. For hybrid models, hence for our FSPNs, it has been shown that reachability is decidable only in the one- and two-dimensional case [7], even under the assumptions that the number of possible discrete states is finite and that the equations describing the evolution of the continuous components have piecewise-constant derivatives.

In this paper, thus, we take a radically different approach. By not seeking a numerical solution of the models, but, rather, accepting to employ a distribution-driven simulation, we are free from many of the previous restrictions. We can then define a very general FSPN formalism, and proceed to investigate efficient simulation algorithms, according to the characteristics of the model under study. Admittedly, this prevents us from performing any general type of reachability study but such shortcoming is shared by all simulation-based approaches. On the other hand, avoiding the generation of the (discrete projection of) the state space altogether is also an advantage, because this is a very memory-intensive step.

The extensions to the FSPN formalism we propose include:

- fluid impulses associated with both immediate and timed transition firings,
- guards for both immediate and timed transitions, dependent on both fluid levels and on the discrete marking.

These extensions are quite natural and useful, given the type of systems we intend to address. Fluid impulses are the continuous analogue of ordinary token movements for ordinary Petri nets, while complete dependency of any behavior (including the guards of immediate transitions) on the entire state of the system (including the current fluid levels) is certainly a desirable orthogonality goal. Indeed, one could argue that these are not really "extensions", but rather that the initial definitions of FSPNs were "restrictions" motivated by the desire of allowing a numerical solution.

Using simulation as a solution method frees us from these restrictions. It should be noted that our extended FSPN formalism is not any more difficult to simulate than the restricted type initially proposed in [18, 24]. However, we do not mean to suggest that its simulation is straightforward. In fact, several innovations are needed, because the simulation of the resulting hybrid models is greatly complicated by the complex dependencies that the behavior of the net can have on the discrete, and even more on the continuous, evolution of the state.

In this regard, the contributions of this paper include:

- Generation of random deviates based on a non-homogeneous Poisson process, using the "thinning" technique [20].
- Interleaving of ODE solution for fluid places with simulation of discrete events in the FSPN.
- Definition of restrictions under which one can more efficiently integrate the change of fluid levels using built-in closed-form results, such as decoupled behavior and special classes of functions for the fluid rates.

It should also be noted that simulation of hybrid models suffers from the same problems affecting discrete-event simulation. If we are interested in a rare event, long run times will be required to obtain useful confidence intervals, regardless of whether the event is defined as a condition on the discrete marking or on the continuous fluid levels. If we are interested in long-term behavior, deciding when the transient effect of the particular initial state chosen becomes statistically negligible is a difficult problem. Hybrid models further complicate these issues. A logical condition affecting the behavior of the model might be formally defined as "places a and b have the same fluid level", but, in a practical implementation using floating point representation, such a test is seldom appropriate; a better approach is to define some relative range within which we can consider two levels to be equal. A related issue is that of regenerative simulation for hybrid models. While a discrete model can have specific regeneration points (discrete markings), requiring the same of a hybrid model might be excessive, since the continuous levels of various places might never return to exactly the same point at the same time; of course, the floating point roundoff errors only make things worse. In this paper, we only discuss how to perform a transient simulation, that is, up to a given finite time τ; as in the simulation of discrete-state models, one can hope to approximate stationary behavior by using a large τ (in comparison to the timing of the net's transitions), but we do not claim that this is indeed the case in general.

Before concluding this introduction, we stress that methods proposing a mixed discrete-continuous approach to simulation are widely used in industrial applications. Indeed, many of the major simulation tools (including SIMAN, ProModel, and Arena) support the development of general mixed models. There is also a literature on optimizations for managing the

execution of such models (e.g., [19]). Work in this area either attempts to optimize *general*, models, at the price of ignoring optimizations that are formalism specific, or develop optimizations for a certain formalism. Our work is of the latter type, by wedding of the SPN paradigm—which is essentially stochastic and essentially discrete—with continuous components. Our contribution lies in exploring the interaction of this specialized SPN paradigm with continuous simulation. Not only is this combination of characteristics unusual in the more general mixed simulation context, it is particularly important to develop these notions in the context of the SPN/GSPN community. Ours is a step towards identifying a modeling framework in which analytic, simulated, discrete, and continuous solutions of submodels might be seamlessly joined.

After introducing the FSPN model in the next section, we describe the method of simulation for the most general case in Section 3. Section 4 considers various subclasses of FSPNs that can be studied using faster simulation algorithms. Examples are provided in Section 5. Section 6 concludes the presentation.

2 Fluid stochastic Petri nets

In the following, we denote sets by upper case calligraphic letters. In particular, $\mathcal{N}$, $\mathcal{R}$, and $\mathcal{R}_0$ indicate the natural, real, and nonnegative numbers, respectively.

For simplicity, we only address exponentially distributed firing times, but we discuss the profound implications of allowing the distributional parameters (the firing rates) to depend on the fluid levels. Generally distributed firing times are clearly useful, and, in connection with discrete-event simulation, do not add much complexity to the solution if their distributional parameters are independent of the fluid levels. Indeed, our prototype SPNP implementation [12] includes several other distributions: constant, uniform, geometric, Weibull, truncated normal, and lognormal. However, the definition of the interruption policies (what happens to the remaining firing times of transitions when one of them fires) requires complex descriptions in full generality [10, 23]; this is not the case with the exponential distribution, due to its memoryless property.

A fluid stochastic Petri net (FSPN) is a tuple $\left(\mathcal{P}^D, \mathcal{P}^C, \mathcal{T}^T, \mathcal{T}^I, a, f, g, \lambda, w, b, \mathbf{m}^0, \mathbf{x}^0\right)$, where:

- $\mathcal{P}^D = \{p_1, \ldots, p_{|\mathcal{P}^D|}\}$ and $\mathcal{P}^C = \{q_1, \ldots, q_{|\mathcal{P}^C|}\}$ are two disjoint and finite sets of places. Let $\mathcal{P} = \mathcal{P}^D \cup \mathcal{P}^C$. A (discrete) place $p \in \mathcal{P}^D$ is drawn with a single circle and can contain a number of tokens $\mathbf{m}_p \in \mathcal{N}$. A (continuous) place $q \in \mathcal{P}^C$ is drawn with two concentric circles and can contain a level of fluid $\mathbf{x}_q \in \mathcal{R}_0$. The marking, or state, of the FSPN is given by a pair of vectors describing the contents of each place, $(\mathbf{m}, \mathbf{x}) \in \hat{\mathcal{S}} = \mathcal{N}^{|\mathcal{P}^D|} \times \mathcal{R}_0{}^{|\mathcal{P}^C|}$. We call $\hat{\mathcal{S}}$ the "potential state space", as opposed to the "actual state space" $\mathcal{S} \subseteq \hat{\mathcal{S}}$, the set of markings actually reachable during the evolution of the FSPN. The marking $(\mathbf{m}, \mathbf{x})$ evolves in time, which we indicate by τ, so, formally, we can think of it as a stochastic process $\{(\mathbf{m}(\tau), \mathbf{x}(\tau)), \tau \geq 0\}$.

- $\mathcal{T}^T = \{t_1, \ldots, t_{|\mathcal{T}^T|}\}$ and $\mathcal{T}^I = \{u_1, \ldots, u_{|\mathcal{T}^I|}\}$ are two disjoint and finite sets of transitions. Let $\mathcal{T} = \mathcal{T}^T \cup \mathcal{T}^I$. A (timed) transition $t \in \mathcal{T}^T$ is drawn as a rectangle and has an exponentially distributed firing time. An (immediate) transition $u \in \mathcal{T}^I$ is drawn as a thin bar and has a constant zero firing time.

- $a : \left((\mathcal{P}^D \times \mathcal{T}) \cup (\mathcal{T} \times \mathcal{P}^D)\right) \times \hat{\mathcal{S}} \to \mathcal{N}$ and $a : \left((\mathcal{P}^C \times \mathcal{T}) \cup (\mathcal{T} \times \mathcal{P}^C)\right) \times \hat{\mathcal{S}} \to \mathcal{R}_0$ describe the marking-dependent cardinality (for discrete places) or the fluid impulse (for continuous places) of the input and output arcs connecting transitions and places. We use the same symbol for both, and we draw them as thin arcs with an arrowhead on their destination, since the type of place eliminates any possibility of confusion. Graphically, the arcs is drawn only if the function is not identically equal zero; the function describing a is written on the arc, with a missing inscription indicating the constant value 1.

- $f : \left((\mathcal{P}^C \times \mathcal{T}^T) \cup (\mathcal{T}^T \times \mathcal{P}^C)\right) \times \hat{\mathcal{S}} \to \mathcal{R}_0$ describes the marking-dependent fluid rate of the input and output arcs connecting timed transitions and continuous places. These fluid arcs are drawn with a thick line, and an arrowhead on their destination. Also in this case the arc is omitted if function is identically equal zero and the function is written on the arc, with a default value 1.

- $g : \mathcal{T} \times \hat{\mathcal{S}} \to \{0, 1\}$ describes the marking-dependent guard of each transition.

- $\lambda : \mathcal{T}^T \times \hat{\mathcal{S}} \to \mathcal{R}_0$ and $w : \mathcal{T}^I \times \hat{\mathcal{S}} \to \mathcal{R}_0$ describe the marking-dependent firing rates (for timed transitions) and weights (for immediate transitions).

- $b : \mathcal{P}^C \times \mathcal{N}^{|\mathcal{P}^D|} \to \mathcal{R}_0 \cup \{\infty\}$ describe the fluid bounds on each continuous place. This bound has no effect when it is set to infinity. Note that

b depends only on the discrete part of the state space, $\mathcal{N}^{|\mathcal{P}^D|}$, not on $\hat{\mathcal{S}}$, to avoid the possibility of circular definitions.

- $(\mathbf{m}^0, \mathbf{x}^0) \in \hat{\mathcal{S}}$ is the initial marking. We require that, for any continuous place $q \in \mathcal{P}^C$, $\mathbf{x}_q \leq b_q(\mathbf{m}^0)$. Graphically, the initial marking is represented by writing the value of $\mathbf{m}_p^0$, or $\mathbf{x}_q^0$, inside the corresponding place. A missing value indicates zero. For discrete places, it is also common to draw $\mathbf{m}_p^0$ tokens inside the place, if this number is small.

The meaning of these quantities is given by the enabling and firing rules. We say that a transition $t \in \mathcal{T}$ has concession in marking $(\mathbf{m}, \mathbf{x})$ iff

$$\forall p \in \mathcal{P}^D,\ a_{p,t}(\mathbf{m}, \mathbf{x}) \leq \mathbf{m}_p \quad \text{and} \quad g_t(\mathbf{m}, \mathbf{x}) = 1.$$

If any immediate transition has concession in $(\mathbf{m}, \mathbf{x})$, it is said to be enabled and the marking is said to be vanishing. Otherwise, the marking is said to be tangible and any timed transition with concession is enabled in it. In other words, a timed transition is not enabled in a vanishing marking even if it has concession.

Some definitions of SPNs allow one to disable a transition t with concession in a marking by specifying a zero rate or weight for it, or by introducing inhibitor arcs, drawn with a circle instead of an arrowhead. Since we can represent these behaviors by an appropriate definition of the input arc cardinalities or the guards, we assume, without loss of generality, that rates and weights are positive for an enabled transition. Inhibitor arcs can then be considered merely as a shorthand[1].

Let $\mathcal{E}(\mathbf{m}, \mathbf{x})$ denote the set of enabled transitions in marking $(\mathbf{m}, \mathbf{x})$. Enabled transitions may change the marking in two ways. First, a transition $t \in \mathcal{T}$ enabled in marking $(\mathbf{m}, \mathbf{x})$ yields a (possibly) new marking $(\mathbf{m}', \mathbf{x}')$, when it fires. We then write $(\mathbf{m}, \mathbf{x}) \xrightarrow{t} (\mathbf{m}', \mathbf{x}')$, where

$$\begin{aligned} \forall p \in \mathcal{P}^D, \qquad & \mathbf{m}'_p = \mathbf{m}_p + a_{t,p}(\mathbf{m}, \mathbf{x}) - a_{p,t}(\mathbf{m}, \mathbf{x}) \\ \forall q \in \mathcal{P}^C, \qquad & \mathbf{x}'_q = \min\{b_q(\mathbf{m}'), \max\{0, \mathbf{x}_q \\ & \qquad + \ a_{t,q}(\mathbf{m}, \mathbf{x}) - a_{q,t}(\mathbf{m}, \mathbf{x})\}\} \end{aligned}$$

(the min and max operator are used to ensure that the new fluid level $\mathbf{x}'_q$ is between zero and the bound for place q evaluated in the new discrete marking). Second, if marking $(\mathbf{m}, \mathbf{x})$ is tangible, fluid flows continuously through the arcs f of enabled timed transitions connected to continuous places. The potential rate of change of fluid level for the continuous place $q \in \mathcal{P}^C$ in tangible marking $(\mathbf{m}, \mathbf{x})$ is

$$\delta_q^{pot}(\mathbf{m}, \mathbf{x}) = \sum_{t \in \mathcal{E}(\mathbf{m}, \mathbf{x})} f_{t,q}(\mathbf{m}, \mathbf{x}) - f_{q,t}(\mathbf{m}, \mathbf{x}).$$

However, the fluid level can never become negative or exceed the bound $b_q(\mathbf{m})$, so the (actual) rate of change over time, τ, while in marking $(\mathbf{m}, \mathbf{x})$, is

$$\delta_q(\mathbf{m}, \mathbf{x}) = \frac{\mathrm{d}\mathbf{x}_q}{\mathrm{d}\tau} =$$

$$\begin{cases} 0 & \text{if } (\mathbf{x}_q = 0 \text{ and } \delta_q^{pot}(\mathbf{m}, \mathbf{x}) \leq 0) \text{ or} \\ & (\mathbf{x}_q = b_q(\mathbf{m}) \text{ and } \delta_q^{pot}(\mathbf{m}, \mathbf{x}) \geq 0) \\ \delta_q^{pot}(\mathbf{m}, \mathbf{x}) & \text{otherwise} \end{cases} . \tag{1}$$

The stochastic evolution of the FSPN in a tangible marking is governed by a race [2]: the timed transition t with the shortest firing time is the one chosen to fire next, unless some fluid levels reach particular values and cause t to become disabled prior to its firing. In a vanishing marking, instead, the weights are used to decide which transition should fire: an immediate transition u enabled in marking $(\mathbf{m}, \mathbf{x})$ fires with probability

$$\frac{w_u(\mathbf{m}, \mathbf{x})}{\sum_{u' \in \mathcal{E}(\mathbf{m}, \mathbf{x})} w_{u'}(\mathbf{m}, \mathbf{x})}. \tag{2}$$

3 General case

The FSPN definition we just gave is very powerful, but it allows one to describe models whose solution can be quite difficult, even with discrete-event simulation.

3.1 Unstable behavior

It is unfortunately possible to define FSPNs that have an "unstable" behavior, that is, the simulation would have to process an infinite number of discrete events in a finite amount of time. It should be noted that these unstable behaviors were already possible in the original definitions of FSPNs, and that they presented the same difficulties and had to be treated as errors in the same way.

[1] If, in $(\mathbf{m}, \mathbf{x})$, an inhibitor arc from $p \in \mathcal{P}^D$ ($q \in \mathcal{P}^C$) to $t \in \mathcal{T}$ has cardinality $c \in \mathcal{N}$ ($c \in \mathcal{R}^0$), t is disabled if $c \geq \mathbf{m}_p$ ($c \geq \mathbf{x}_q$). The same behavior can be modeled by ensuring that the guard g_t evaluates to 0 in $(\mathbf{m}, \mathbf{x})$.

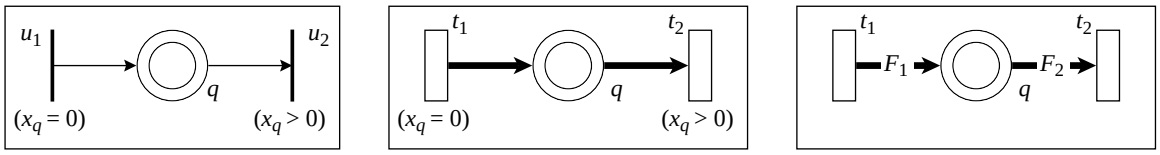

Figure 1: FSPNs exhibiting unstable behaviors.

3.1.1 Infinite number of discrete events

Consider the FSPNs in Fig. 1 on the left. Immediate transitions u_1 and u_2 alternatively put and remove a unit impulse instantaneously. Thus, without advancing time, an infinite number of events can occur.

This is analogous to the so-called "vanishing-loop" in GSPNs [9] and, in particular, to the "absorbing" type, which, once encountered, will keep reoccurring with probability one. With few exceptions [16], such a behavior has been considered a modeling error in the literature on discrete-state models, hence we do the same for FSPNs.

Another well-known type of instability involves actual time advances between change of markings, but increasingly faster, so that an infinite number of events can occurs in a finite (although nonzero) time. One can just think of the classical example of nonregular continuous-time Markov chain, where the rate of going from state i to state $i+1$ is 2^i. This is always considered an error.

3.1.2 Infinite number of infinitesimal events

The instability of the model in the middle of Fig. 1 is instead exclusive to models with a state having a continuous component, such as our FSPNs. When $\mathbf{x}_q^0 = 0$, timed transition t_1 is enabled and timed transition t_2 is disabled. However, as soon as the fluid arc starts adding fluid to q, the situation is reversed, t_1 becomes disabled, while t_2 becomes enabled and starts emptying q. It could be argued that, in such a situation, q will always be empty, but any model where an infinite number of events occurs in a finite time (e.g., transitions t_1 and t_2 become enabled and disabled an infinite number of times) cannot be managed by conventional discrete-event simulation techniques. Hence, we will consider such behavior illegal.

The model on the right, with constant fluid rates F_1 and F_2, could also be considered unstable if $F_2 > F_1$. Both t_1 and t_2 are always enabled, hence there is a continuous flow into q at rate F_1 due to t_1. However, the outgoing flow due to t_2 cannot be F_2. Our definition simply states that δ_q is identically equal 0 in this case, implying that the outgoing flow is effectively reduced to be F_1, instead of F_2. In other words, the arc from q to t_2 can be thought to have effect only a fraction F_1/F_2 of the time. Since this type of behavior can be easily managed by examining all the flows incident to a continuous place, we do not regard it as an error.

We stress that detection of instability might be accomplished "on-the-fly" while running the simulation, provided we keep a stack of markings visited without advancing the simulation clock. However, this is possible only when the instability causes the FSPN to return to the same marking with probability one. If, on the other hand, the markings visited in zero time are nonrepeating, the stack will simply grow without limit until the simulation program runs out of memory. Again, this is not a new problem: one can simply think of a GSPN with an immediate transition t that, once enabled, keeps adding a token at a time to a place p that does not affect the enabling of t.

Unfortunately, these situations cannot be detected in general even for GSPNs, since this formalism is Turing-equivalent, so we have no hope to devise and algorithm to detect them in general for FSPNs either. Hence, in practice, we can implement checks to discover only the simple cases of instability, but we must assume that no other instability exist in the model.

3.2 Stable behavior

We now describe how a model with no unstable behaviors can be studied. Assume that we have just entered tangible marking $(\mathbf{m}, \mathbf{x})$. If there is any enabled transition, each continuous component $\mathbf{x}_q$ might vary in a very general way over time. In a marking $(\mathbf{m}, \mathbf{x}(0))$, we can apply Eq. 1 to each $q \in \mathcal{P}^C$ and obtain the system of ordinary differential equations

$$\forall q \in \mathcal{P}^C,\ \frac{\mathrm{d}\mathbf{x}_q(\tau)}{\mathrm{d}\tau} = \sum_{t \in \mathcal{E}(\mathbf{m}, \mathbf{x}(0))} f_{t,q}(\mathbf{m}, \mathbf{x}(\tau)) - f_{q,t}(\mathbf{m}, \mathbf{x}(\tau))$$

$$\text{with given initial condition } \mathbf{x}(0), \qquad (3)$$

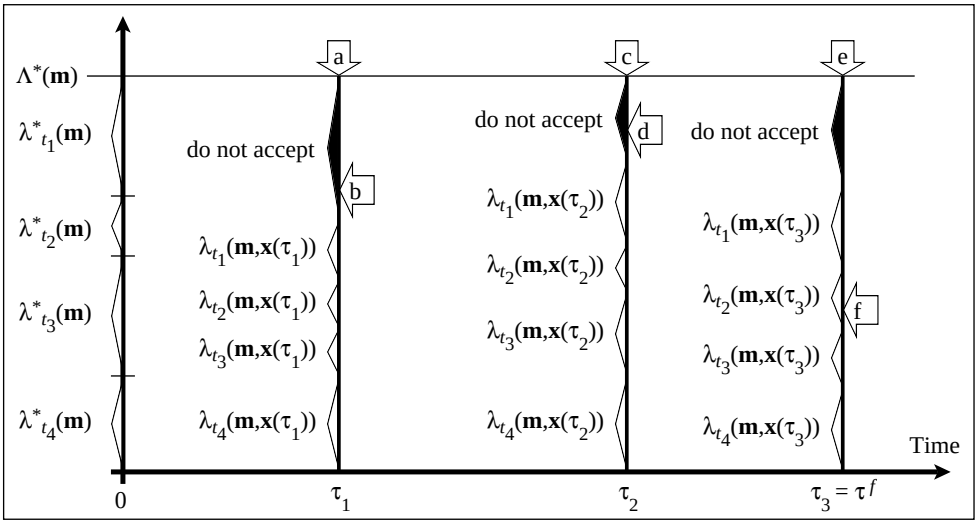

Figure 2: Sampling the NHPP process underlying a FSPN.

which is valid in any interval $[0, \tau)$ during which the set of enabled transitions does not change, that is, $\mathcal{E}(\mathbf{m}, \mathbf{x}(\tau)) = \mathcal{E}(\mathbf{m}, \mathbf{x}(0))$. We can then consider two cases, based on whether the set $\mathcal{E}(\mathbf{m}, \mathbf{x}(\tau))$ is independent or not on the continuous component $\mathbf{x}$.

3.2.1 Enabling independent of the fluid levels

In the simpler case, the cardinality of the arcs connected to discrete places and the guards do not depend on $\mathbf{x}$. Even so, the firing times behave as a nonhomogeneous Poisson process (NHPP) whose rate depends on the continuous marking, and so some care is required in sampling the firing instants. We assume that the firing rate of each timed transition t can be bounded from above by $\lambda_t^*(\mathbf{m})$, given our knowledge of its dependence on the fluid marking. That is, when the discrete marking is $\mathbf{m}$, the rate of t satisfies $\lambda_t(\mathbf{m}, \mathbf{x}) \leq \lambda_t^*(\mathbf{m})$, for any value of $\mathbf{x}$ that might be reached in conjunction with $\mathbf{m}$. We can therefore sample from the NHPP using the technique of "thinning" [20], where we sample "potential firing instants" in accordance with a homogeneous Poisson arrival process with rate

$$\Lambda^*(\mathbf{m}) = \sum_{t \in \mathcal{E}(\mathbf{m}, \mathbf{x})} \lambda_t^*(\mathbf{m}).$$

From this process, we can define a sequence of increasing time instants $(\tau_1, \tau_2, \ldots)$. Starting from $i = 1$, we "accept" τ_i, that is, we declare that a firing occurred at time τ_i, with probability $\Lambda(\mathbf{m}, \mathbf{x}(\tau_i))/\Lambda^*(\mathbf{m})$, where

$$\Lambda(\mathbf{m}, \mathbf{x}(\tau_i)) = \sum_{t \in \mathcal{E}(\mathbf{m}, \mathbf{x})} \lambda_t(\mathbf{m}, \mathbf{x}(\tau_i)).$$

In other words, we use the actual firing rates at time τ_i as a weight, to determine whether the event corresponds to a true firing or not. This requires us to solve for the value of $\mathbf{x}(\tau_1)$, by integrating the system of ordinary differential equations (3). If τ_1 is accepted, we stop. Otherwise, we integrate until τ_2, compute $\mathbf{x}(\tau_2)$, and decide whether to accept τ_2 or not, and so on. Eventually, this process stops at some step i, giving us a sampling $\tau^f = \tau_i$ of the actual firing time.

For example, Fig. 2 illustrates the case where four transitions are enabled in $(\mathbf{m}, \mathbf{x})$: t_1, t_2, t_3, and t_4. The sequence of labeled arrows shows the random deviates that must be generated, in order. First, we generate τ_1 (a) according to the distribution Expo($\Lambda^*(\mathbf{m})$). Then we generate a random deviate (b) $\sim$Unif$(0, \Lambda^*(\mathbf{m}))$. In the figure, this happens to fall in the interval corresponding to the "do not accept" case. Thus, we need to generate another potential firing time (c) by sampling the distribution Expo($\Lambda^*(\mathbf{m})$) again and summing the sampled value to τ_1, obtaining τ_2. We also need another random deviate (d) $\sim$Unif$(0, \Lambda^*(\mathbf{m}))$, which also, in the figure, happens to cause a rejection. Finally, we generate a third potential firing time and we add it to τ_2, resulting in τ_3 (e). When we sample (f) $\sim$Unif$(0, \Lambda^*(\mathbf{m}))$ again, we finally obtain a value falling in the interval

corresponding to t_2, hence we schedule the firing of t_2 at time τ_3.

It is then apparent that the expected number or random deviates that need to be generated to decide which transition to fire is larger when the bounds $\lambda_t(\mathbf{m},\mathbf{x})$ for the enabled transitions are less tight, since this increase the likelihood of rejections when performing the thinning. On the other hand, if the rates of the enabled transitions in marking $\mathbf{m}$ are a function of $\mathbf{x}$, but their total, $\sum_{t\in\mathcal{E}(\mathbf{m},\mathbf{x})}\lambda_t(\mathbf{m},\mathbf{x})$ is a known constant independent of $\mathbf{x}$, $\Lambda^*(\mathbf{m})$ can be set to this value, and no rejection will occur. Then, only two deviates are needed: the first one to decide the value of τ_1 and the second one to decide which transition to fire among the enabled ones.

3.2.2 Enabling dependent on the fluid levels

If, instead, the set of enabled transitions can change as $\mathbf{x}$ evolves, we also need to consider an "enabling event" at the time τ^e when the first change in $\mathcal{E}(\mathbf{m},\mathbf{x})$ occurs. The method to compute τ^e depends on the nature of the dependencies.

In principle, we should know the value of $\mathbf{x}(\tau)$ over the entire horizon $\tau\in[0,\tau^f]$. We can still use integration but, in full generality, we have to check whether the set $\mathcal{E}(\mathbf{m},\mathbf{x})$ has changed, at each integration step. These additional checks can be quite expensive, since they potentially imply reevaluating many marking-dependent functions.

If we find no value $\tau^e\in[0,\tau^f]$ for which the set of enabled transitions changes, the next event to schedule is the firing at time τ^f. Otherwise, we must schedule an "enabling event" at time τ^e, the time of the first change in $\mathcal{E}(\mathbf{m},\mathbf{x})$. Of course, in this case, we can stop the integration at time τ^e, without having to reach time τ^f.

We stress that, regardless of whether the enabled set can change or not, the generation of next firing times is considerably simplified if the firing rates of the enabled timed transitions are not dependent on fluid levels, since the machinery of NHPP-based generation of random deviates can be avoided.

3.2.3 Processing immediate firings

The processing of the scheduled event causes a change of marking, from $(\mathbf{m},\mathbf{x})$ to $(\mathbf{m}',\mathbf{x}')$, where $\mathbf{m}'=\mathbf{m}$ if the event was of the enabling type. Then, in marking $(\mathbf{m}',\mathbf{x}')$, a finite sequence of immediate firings might take place, just as in ordinary, non-fluid, SPNs, until the next tangible marking $(\mathbf{m}'',\mathbf{x}'')$ is reached. Thanks to the memoryless property of the exponential distribution, the evolution of the FSPN from this point on is analogous to the evolution from the initial marking, that is, we do not need to be concerned about the "remaining firing times" of transitions that were already enabled prior to reaching this marking.

4 FSPNs with efficient solution

The general behavior just described requires us to solve the system of ordinary differential equations (3) at each step of the simulation. This computation can be quite costly. We now examine various subclasses of FSPNs which, due to their restricted marking-dependent behavior, have simpler solution algorithms.

4.1 Uncoupled behavior

A restriction on the type of dependency allows us to uncouple the system, resulting in a set of ordinary differential equations that can be solved independently. This requires that the fluid rates incident on q, hence $\delta_q(\mathbf{m},\mathbf{x})$, depend only on $(\mathbf{m},\mathbf{x}_q)$, not on the fluid levels in the other continuous places:

$$\forall(\mathbf{m},\mathbf{x}),(\mathbf{m},\mathbf{x}')\in\hat{\mathcal{S}},\ \mathbf{x}_q=\mathbf{x}'_q\Rightarrow\delta_q(\mathbf{m},\mathbf{x})=\delta_q(\mathbf{m},\mathbf{x}').$$

As in the general case, we can still distinguish whether the set of enabled transitions can be affected by $\mathbf{x}$ or not, and the NHPP random variate generation needs to be used only if their firing rates depend on $\mathbf{x}$.

4.2 Predefined classes of behaviors

For particular cases of uncoupled dependencies, we can even have a built-in closed form solution, which will avoid the need for numerical integration altogether.

4.2.1 Linear fluid change rate

One such case is when, in a given marking $(\mathbf{m},\mathbf{x})$,

$$\frac{\mathrm{d}\mathbf{x}_q(\tau)}{\mathrm{d}\tau}=A(\mathbf{m})\cdot\mathbf{x}_q(\tau)+B(\mathbf{m}),\qquad A(\mathbf{m})\neq 0$$

that is, the fluid change rate for a continuous place is a linear function of the fluid level in the place itself. In this case, the solution is

$$\mathbf{x}_q(\tau)=-\frac{B(\mathbf{m})}{A(\mathbf{m})}+\left(\mathbf{x}_q(0)+\frac{B(\mathbf{m})}{A(\mathbf{m})}\right)e^{A(\mathbf{m})\tau},$$

assuming that $\mathbf{x}_q$ remains between 0 and $b_q(\mathbf{m})$ during $[0,\tau]$. This answers the question of how much the fluid

level in a place will change during the firing time τ of a timed transition. Inversely, the time τ_q when place q reaches a certain fluid level threshold L_q is given by

$$\tau_q = \frac{\ln\left(\dfrac{L_q + \dfrac{B(\mathbf{m})}{A(\mathbf{m})}}{\mathbf{x}_q(0) + \dfrac{B(\mathbf{m})}{A(\mathbf{m})}}\right)}{A(\mathbf{m})}, \qquad (4)$$

if this quantity is positive (if it is negative, we can simply define $\tau_q = \infty$, that is, the threshold L_q cannot be reached in this marking).

If the set of enabled transitions can only change when some place q reaches a threshold level L_q, then we can simply define the time τ^e of the next enabling event as

$$\tau^e = \min_{q \in \mathcal{P}^C} \{\tau_q\}.$$

4.2.2 Constant fluid change rate

When $A(\mathbf{m}) = 0$, that is, when the fluid change rate is a constant, the solution is much simpler,

$$\frac{\mathrm{d}\mathbf{x}_q(\tau)}{\mathrm{d}\tau} = B(\mathbf{m}) \qquad \Rightarrow \qquad \mathbf{x}_q(\tau) = \mathbf{x}_q(0) + B(\mathbf{m})\tau,$$

again assuming that $\mathbf{x}_q$ remains between 0 and $b_q(\mathbf{m})$ during $[0, \tau]$. The time τ_q when place q reaches the threshold L_q is then

$$\tau_q = \frac{L_q - \mathbf{x}_q(0)}{B(\mathbf{m})}, \qquad (5)$$

if this quantity is positive, infinity otherwise.

4.3 Discretized dependency on x

Complete dependency on the marking $(\mathbf{m}, \mathbf{x})$ is desirable in principle, but the solution complexity it entails can be large, and its full power in a model. A simpler type of dependency is obtained by enforcing a discretization on the behavior of the FSPN with respect to the continuous component $\mathbf{x}$. This can be accomplished using a finite set $\mathcal{L} = \{\mathbf{l}_1, \ldots, \mathbf{l}_{|\mathcal{L}|}\}$ of boolean threshold-type functions of the marking, where

$$\mathbf{l}_k(\mathbf{m}, \mathbf{x}) \stackrel{\text{def}}{=} (\mathbf{l}_k^*(\mathbf{m}, \mathbf{x}) \leq \beta(\mathbf{m})), \qquad \text{with} \quad \beta(\mathbf{m}) \in \mathcal{R}$$

and $\mathbf{l}_k^*(\mathbf{m}, \mathbf{x})$ is a real function of the marking whose form depends on the nature of the functions δ_q for the places involved in its definition. We consider two cases leading to efficiency improvements for the simulation:

- For any fluid place $q \in \mathcal{P}^C$ with a linear fluid change rate (Sec. 4.2.1), we can define functions of the form

  $$\mathbf{l}_k^*(\mathbf{m}, \mathbf{x}) = \alpha_{k,q}(\mathbf{m})\mathbf{x}_q,$$

  where the coefficient $\alpha_{k,q}$ is a real function of the discrete portion of the marking,

  $$\alpha_{k,q} : \mathcal{N}^{|\mathcal{P}^D|} \to \mathcal{R}.$$

- For any set $\mathcal{Q} \subseteq \mathcal{P}^C$ of fluid places with constant fluid change rates (Sec. 4.2.2), we can define functions of the form

  $$\mathbf{l}_k^*(\mathbf{m}, \mathbf{x}) = \sum_{q \in \mathcal{Q}} \alpha_{k,q}(\mathbf{m})\mathbf{x}_q,$$

  where the coefficients $\alpha_{k,q}$ are, as above, real functions of $\mathbf{m}$ alone.

Given a marking $(\mathbf{m}, \mathbf{x})$, we can define the "discretized" marking $(\mathbf{m}, \mathbf{l})$ obtained from $(\mathbf{m}, \mathbf{x})$ through $\mathcal{L}$. If we force a (for discrete places only), g, and λ to be defined on the discretized marking $(\mathbf{m}, \mathbf{l})$, rather than on the original mixed marking $(\mathbf{m}, \mathbf{x})$, then the logical and timing behavior of the FSPN can change only when the value of one or more of the functions $\mathbf{l}_k$ changes its truth value, that is, when the corresponding function $\mathbf{l}_k^*$ crosses the threshold value $\beta_k(\mathbf{m})$, or when a firing occurs.

Given the way each function $\mathbf{l}_k^*$ is defined, however, it is easy to compute the time τ_k at which this can happen. For a function $\mathbf{l}_k$ of the first type, we can determine τ_k by using the right-hand-side of equation (4), with L_q set to $\beta_k/\alpha_{k,q}$. For a function $\mathbf{l}_k$ of the second type, we can determine τ_k as follows:

1. compute the overall change rate in the value of $\mathbf{l}_k^*(\mathbf{m}, \mathbf{x})$, $\Delta_k(\mathbf{m}) = \sum_{q \in \mathcal{Q}} a_{k,q}\delta_q(\mathbf{m}, \mathbf{x})$, a quantity that does not depend on $\mathbf{x}$ given our assumption of constant fluid change rate;

2. if $\Delta_k(\mathbf{m})$ is zero, $\mathbf{l}_k^*(\mathbf{m}, \mathbf{x})$ is constant and will not hit the threshold, set $\tau_k = \infty$;

3. otherwise, compute the distance d_k from β_k to $\mathbf{l}_k^*(\mathbf{m}, \mathbf{x})$,

   $$d_k = \beta_k - \sum_{q \in \mathcal{Q}} a_{k,q}\mathbf{x}_q;$$

4. if $\Delta_k(\mathbf{m})$ and d_k have the same sign, set

   $$\tau_k = d_k/\Delta_k(\mathbf{m}),$$

   otherwise set $\tau_k = \infty$.

Then, the time of the next event to be managed by the simulator is simply the minimum between the first hitting of a threshold, $\min\{\tau_k : 1 \leq k \leq |\mathcal{L}|\}$, and the minimum firing time. However, the firing times are now guaranteed to be constant within a given discretized marking $(\mathbf{m}, \mathbf{l})$. Hence, the entire simulation can proceed as in a traditional discrete-event simulation, with the exception that the types of events that need to be scheduled in the event queue are either transition firings or the hitting of a threshold.

Fortunately, there is no need to place the same restriction on the fluid impulses (a for continuous places) or the weights w, since the simulation always evaluates the value of impulses and weights only at a specific and known instant in time, and the identity of the arcs and transitions for which they need to be evaluated is known as well. Applying the restriction to these quantities as well would prevent us from modeling useful behaviors, such as emptying a continuous place, or choosing between two immediate transitions with probability proportional to the level in two continuous places, but would not simplify the simulation algorithms.

We stress that this restricted type of discretization still allows to model many interesting behaviors, such as disabling a transition t when a fluid level $\mathbf{x}_q$ reaches a given level β, provided place q has either linear or constant fluid change rate. With places having constant fluid change rates, even more general tests can be performed, such as testing for the condition $\mathbf{x}_{q_1} + \mathbf{x}_{q_2} \leq \mathbf{x}_{q_3}$, which corresponds to the boolean threshold function $\mathbf{l}_k(\mathbf{m}, \mathbf{x}) = (\mathbf{x}_{q_1} + \mathbf{x}_{q_2} - \mathbf{x}_{q_3} \leq 0)$. Nonlinear conditions such as $\mathbf{x}_{q_1}\mathbf{x}_{q_2} \leq \mathbf{x}_{q_3}$, however, cannot be captured by the proposed discretization.

We conclude this section on discretized behavior by observing that an alternative could be to define *integer*, instead of *boolean*, threshold functions, for example as

$$\mathbf{l}_k(\mathbf{m}, \mathbf{x}) \stackrel{\text{def}}{=} \lfloor \mathbf{l}_k^*(\mathbf{m}, \mathbf{x}) \rfloor .$$

This would be more general, since the component $\mathbf{l}$ of the discretized marking can now in principle describe infinite sets, instead of just $2^{|\mathcal{L}|}$ tuples. However, a simulator using this convention would have to schedule events each time a function $\mathbf{l}_k$ crosses an integer boundary, even if many of these events might not really affect the net's behavior.

5 Examples

We illustrate the power of the proposed FSPN formalism with a few examples.

5.1 A queue with impatient customers and breakdowns

Consider a queue with a server subject to breakdowns and repairs. The customers arrive with a constant rate, and queue in an unbounded waiting room. They are served in first-come-first-serve order, but, once their service starts, they can become impatient and leave before completion (see Fig. 3). Unlike other system with impatient customers, the amount of time a customer has been in the queue before his service begins does not affect his decision to leave. The arcs from *Serving* to *Busy* and from *Waiting* to *Idle* are used to count time into the two places, hence they have fluid rate one. The arcs from *Busy* and *Idle* to *Serving* (or *Leave*) have impulse $\mathbf{x}_{Busy}$ and $\mathbf{x}_{Idle}$ defined on them, respectively. Hence, they are "flushing" arcs, they have the effect of emptying the two places immediately after the firing of *Serving* (or *Leave*).

The guard of immediate transition *Leave* specifies when the customer at the head of the queue decides to leave. Various policies can be easily modeled.

1. The total amount of time from the moment service began exceeds a certain threshold *MAX*. Then, we could define the guard g_{Leave} to be the boolean expression $(\mathbf{x}_{Busy} + \mathbf{x}_{Idle} = MAX)$.

   This policy is representative of situations where, once the server begins operating on a customer, the operation must complete within a certain time, for example to avoid spoilage.

2. The total amount of time a customer has not received any service from the moment service began exceeds a certain threshold *MAX*. Then, $g_{Leave} = (\mathbf{x}_{Idle} = MAX)$.

   This could represent a similar situation, where spoilage occurs only when the customer is not being served.

3. A customer has waited for an uninterrupted period of time *MAX* without receiving any service. Then, $g_{Leave} = (\mathbf{x}_{Idle} = MAX)$, after adding an impulse arc $a_{Idle,Repair}(\mathbf{m}, \mathbf{x}) = \mathbf{x}_{Idle}$, so that place *Idle* becomes empty after each repair.

   This could represent a situation, where, in addition to occurring only when the customer is not being served, any spoilage immediately disappears as soon as service resumes.

4. A customer has spent more time waiting for the server to be operational than receiving service, from the moment service began. Then, $g_{Leave} = (\mathbf{x}_{Idle} > \mathbf{x}_{Busy})$.

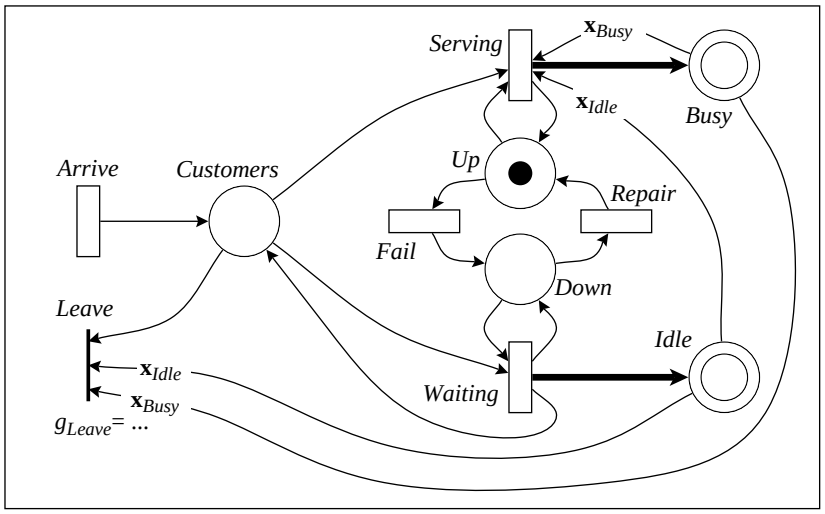

Figure 3: The FSPN of a queue with impatient customers and breakdowns.

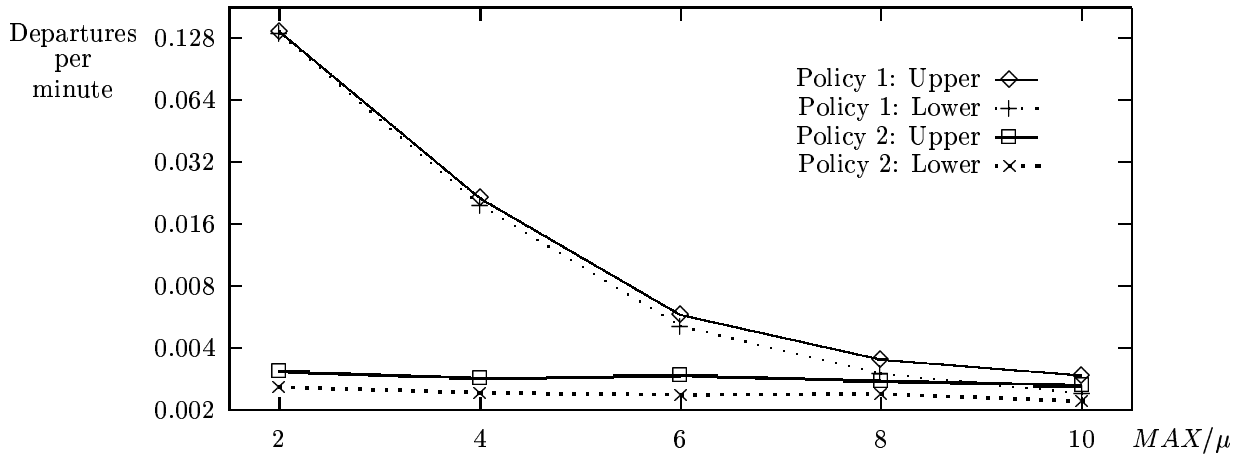

Figure 4: Lower and upper limits of the 95% confidence interval for the expected frequency of impatient departures.

A measure of interest is the fraction of jobs that decides to leave,

$$\frac{\text{number of firings of } \textit{Leave} \text{ up to time } \tau}{\text{number of firings of } \textit{Arrive} \text{ up to time } \tau}.$$

Fig. 4 compares the throughput of transition *Leave* according to the first two policies, averaged over the interval [0...10,000], as a function of MAX normalized by the average service time $1/\mu$. Note that a logarithmic scale is used on the y-axis. The following parameters are assumed:

- Arrival rate = 1.0/min.
- Service rate $\mu = 5.0$/min.
- Failure rate = 1/64min, Repair rate = 1/8min.
- $MAX = 2/\mu, 4/\mu, \ldots, 10/\mu$ (i.e., the customer becomes impatient after 2, 4, ..., 10 times the average service time).
- Initial state as shown in Fig. 3.

The results where obtained from our prototype FSPN simulator implemented in SPNP[2] [12].

For each choice of MAX, twenty independent replications were performed. The total runtime to compute

[2] At the moment, only a subset of the classes of FSPNs satisfying the restrictions of Sect. 4.2 and 4.3 have been implemented in SPNP.

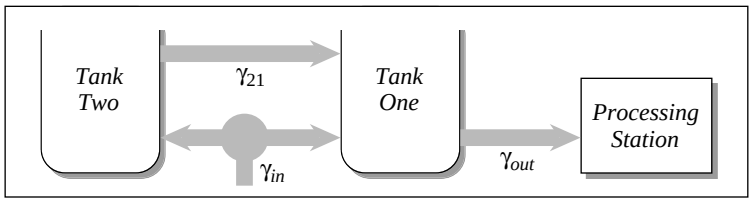

Figure 5: A dual-tank processing facility.

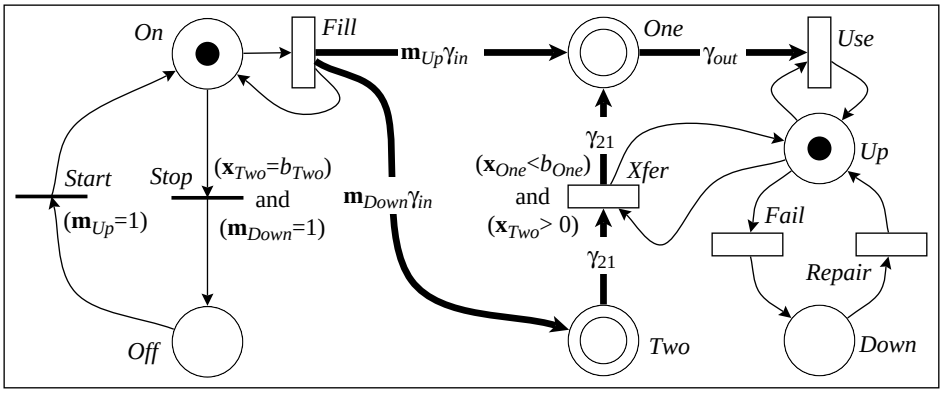

Figure 6: The FSPN of the dual-tank processing facility.

all the data shown in Fig. 4 was approximately seven minutes using an ordinary Unix workstation.

5.2 A dual-tank processing facility

Consider a processing plant where, during normal operation, liquid enters a main tank, *One*, from an external source with rate γ_{in}, and is used by a processing station, with a (potential) rate $\gamma_{out} > \gamma_{in}$ (see Fig. 5).

However, the processing station is subject to breakdowns, during which it cannot process the liquid. Interrupting the flow from the external source of liquid into the main tank is an expensive operation, hence, a second additional tank, Two, is present. When the processing station is down, the liquid is automatically routed to tank Two, which has a maximum capacity b_{Two}. Only when the second tank is full, the flow from the external source is shut down. After a repair, the processing can resume and the liquid is routed again from the external source, which is restarted if it had been shut down, into tank *One*. In addition, any liquid in tank Two is pumped into tank *One*, with rate γ_{21}. If $\gamma_{in} + \gamma_{21} > \gamma_{out}$, the level in tank *One* will increase while the processing station is working to catch up after a repair. Since tank *One* has a maximum capacity, b_{One}, the flow from tank Two to tank *One*, rather than the flow from the external source, is slowed down when this limit is reached. The guard $g_{Xfer} = (\mathbf{x}_{One} < b_{One})$ in the FSPN of Fig. 6 enforces this behavior.

The main reason for having two tanks, instead of a single large one, is efficiency. As the liquid needs to be maintained at a given temperature, tank *One* is constantly heated, while tank Two is heated only when it contains liquid, because of a breakdown. Indeed, the two measures we could be interested in computing are:

$$\frac{\text{number of firing of } Stop \text{ up to time } \tau}{\tau},$$

the frequency at which the external source needs to go through a start-stop cycle, and

probability that tank Two is not empty at time τ.

Fig. 7 reports one of these measures, the expected frequency of stops, computed over the interval [0...100,000], as a function of γ_{in}. The following parameters are assumed:

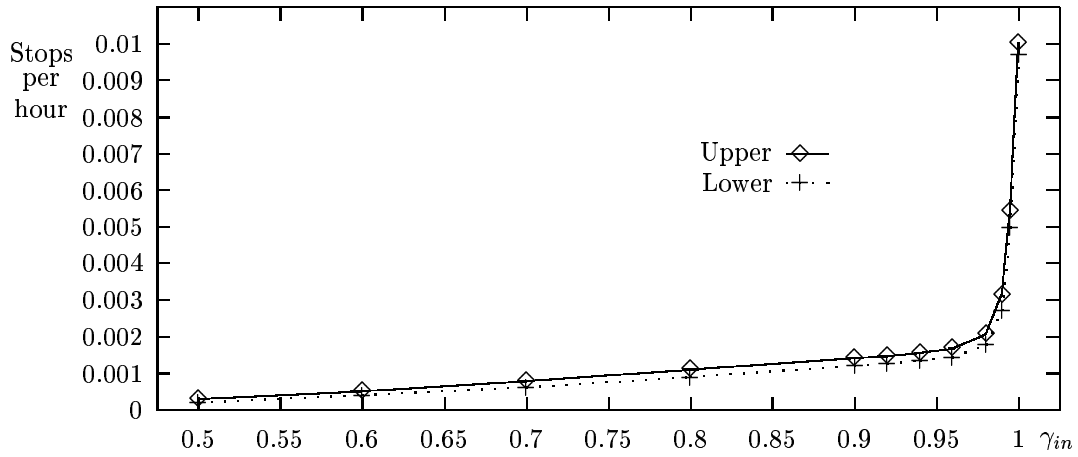

Figure 7: Lower and upper limits of the 95% confidence interval for the expected frequency of stops.

- $b_{One} = 1$, $b_{Two} = 2$.
- $\gamma_{in} = 0.5, \ldots, 1.0$, $\gamma_{21} = 0.1$, $\gamma_{out} = 1.0$.
- Failure rate = 1/100hr, Repair rate = 1/hr.
- Initial state: $\mathbf{m}_{On} = 1$, $\mathbf{m}_{Up} = 1$, $\mathbf{x}_{One} = 1.0$.

For each choice of γ_{in}, ten independent replications were performed. The total runtime to compute all the data shown in Fig. 7 was approximately two minutes using an ordinary Unix workstation.

6 Conclusion

In this paper we extended the modeling power of recently introduced fluid stochastic Petri nets. Since equations characterizing the evolution of such FSPNs constitute a coupled system of partial differential equations, their generation and solution can become intractable but for small or very well-structured FSPNs. Hence, discrete-event simulation becomes an important alternative avenue for the solution of FSPNs. However, due to the mixed nature of the state space, with discrete and continuous components and arbitrary interactions between them, simulation also poses several challenges that we address. When we can characterize the type of interactions as belonging to one of the several restricted classes of models we define, a better suited, and faster, simulation algorithm can be employed for the solution.

References

[1] T. Agerwala. A complete model for representing the coordination of asynchronous processes. Hopkins Computer Research Report 32, Johns Hopkins University, Baltimore, Maryland, July 1974.

[2] M. Ajmone Marsan, G. Balbo, A. Bobbio, G. Chiola, G. Conte, and A. Cumani. The effect of execution policies on the semantics and analyis of Stochastic Petri Nets. *IEEE Trans. Softw. Eng.*, 15(7):832–846, July 1989.

[3] M. Ajmone Marsan, G. Balbo, and G. Conte. A class of Generalized Stochastic Petri Nets for the performance evaluation of multiprocessor systems. *ACM Trans. Comp. Syst.*, 2(2):93–122, May 1984.

[4] M. Ajmone Marsan, G. Balbo, and G. Conte. *Performance models of multiprocessor systems.* The MIT Press, Cambridge, MA, 1986.

[5] D. Anick, D. Mitra, and M. Sondhi. Stochastic theory of data-handling systems. *Bell Syst. Techn. J.*, 61(8):1871–1894, Oct. 1982.

[6] T. Araki and T. Kasami. Some decision problems related to the reachability problem for Petri nets. *Theoretical Computer Science*, 3:85–104, 1977.

[7] E. Asarin, O. Maler, and A. Pnueli. Reachability analysis of dynamical systems having piecewise-constant derivatives. *Theoretical Computer Science*, 138:35–65, 1995.

[8] C. G. Cassandras. *Discrete Event Systems: Modeling and Performance Analysis.* Aksen Associates, 1993.

[9] G. Ciardo. *Analysis of large stochastic Petri net models.* PhD thesis, Duke University, Durham, NC, 1989.

[10] G. Ciardo. Discrete-time Markovian stochastic Petri nets. In W. J. Stewart, editor, *Computations with Markov Chains*, pages 339–358. Kluwer, Boston, MA, 1995.

[11] G. Ciardo, J. K. Muppala, and K. S. Trivedi. Analyzing concurrent and fault-tolerant software using stochastic Petri nets. *J. Par. and Distr. Comp.*, 15(3):255–269, July 1992.

[12] G. Ciardo, K. S. Trivedi, and J. K. Muppala. SPNP: Stochastic Petri net package. In *Proc. 3rd Int. Workshop on Petri Nets and Performance Models (PNPM'89)*, pages 142–151, Kyoto, Japan, Dec. 1989. IEEE Comp. Soc. Press.

[13] R. David. Modeling of hybrid systems using continuous and hybrid Petri nets. In *Proc. 7th Int. Workshop on Petri Nets and Performance Models (PNPM'97)*, pages 47–58, St. Malo, France, June 1997. IEEE Comp. Soc. Press.

[14] R. David and H. Alla. Continuous Petri nets. In *Proc. 8th European Workshop on Application and Theory of Petri Nets*, pages 275–294, Zaragoza, Spain, 1987.

[15] A. I. Elwalid and D. Mitra. Statistical multiplexing with loss priorities in rate-based congestion control of high-speed networks. *IEEE Trans. Comm.*, 42(11):2989–3002, Nov. 1994.

[16] W. K. Grassmann and Y. Wang. Immediate events in Markov chains. In W. J. Stewart, editor, *Computations with Markov Chains*, pages 163–176. Kluwer, Boston, MA, 1995.

[17] M. Hack. Decidability questions for Petri nets. Technical Report 161, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA, June 1976.

[18] G. Horton, V. Kulkarni, D. Nicol, and K. Trivedi. Fluid stochastic Petri nets: Theory, application, and solution. *Europ. J. of Oper. Res.*, 105:184–201, 1998.

[19] J. F. Klingener. Programming combined discrete-continuous models for performance. In *Proc. 1996 Winter Simulation Conf.*, pages 833–839, 1996.

[20] P. A. W. Lewis and G. S. Shedler. Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly*, 26:403–414, 1979.

[21] D. Mitra. Stochastic theory of fluid models of multiple failure-susceptible producers and consumers coupled by a buffer. *Adv. Appl. Prob.*, 20:646–676, 1988.

[22] T. Robertazzi. *Computer Networks and Systems: Queueing Theory and Performance Evaluation.* Springer-Verlag, 1990.

[23] M. Telek, A. Bobbio, and A. Puliafito. Steady state solution of MRSPN with mixed preemption policies. In *Proc. IEEE International Computer Performance and Dependability Symposium (IPDS'96)*, pages 106–115, Urbana-Champaign, IL, USA, Sept. 1996. IEEE Comp. Soc. Press.

[24] K. S. Trivedi and V. G. Kulkarni. FSPNs: fluid stochastic Petri nets. In *Proc. 14th Int. Conf. on Applications and Theory of Petri Nets*, pages 24–31, Chicago, IL, June 1993.

[25] N. Viswanadham and Y. Narahari. *Performance Modeling of Automated Manufacturing Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1992.