Acta Simulatio - International Scientific Journal about Simulation
Volume: 4 2018 Issue: 2 Pages: 1-11 ISSN 1339-9640

doi:10.22306/asim.v4i2.42
Received: 12 Mar. 2018
Accepted: 20 May 2018

SIMULATING FSPN MODELS USING PROCESS-BASED DISCRETE-EVENT SIMULATION LANGUAGE

Zoran Kotevski
Faculty of information and communication technologies, "St. Kliment Ohridski" University – Bitola
Partizanska bb, 7000 Bitola, R. Macedonia. E-mail: zoran.kotevski@fikt.edu.mk

Keywords: fluid stochastic Petri nets, process-based discrete-event simulation, simulation of continuous quantities, peer-to-peer, video streaming, performance evaluation
Abstract: Fluid Stochastic Petri Nets (FSPN) is a mathematical and graphical formalism designed for modeling and behavior evaluation of complex stochastic and hybrid systems that concurrently employ discrete and continuous logic. Analytic performance evaluations of FSPN models require a solution to a complex system of partial differential equations whose generation and solution can easily become intractable. This problem occurs because the number of differential equations in the system directly corresponds to the number of discrete states of the FSPN model. For FSPN models that exhibit large state spaces, the only feasible solution method is by the use of simulations. However, for certain FSPN models, the existing FSPN simulation methods and software packages do not provide a feasible solution, which was the main motivation to describe the simulation challenges of certain FSPN models and explore for possible alternatives. In this paper, two approaches for simulation of FSPN models using process-based discrete-event simulation language are presented. The two different approaches are evaluated in the context of simulation speed and accuracy. The results obtained show that continuous quantities in FSPN models can be effectively simulated using discrete events without compromising the accuracy of the simulation outcome.

1 Introduction

Fluid Stochastic Petri Nets (FSPN) [1-7] is a modeling formalism that was introduced more than two decades ago and was built as an extension to the well-established formalism of stochastic Petri nets. The main motivation for their introduction was the requirement to represent certain quantities as a fluid flow, rather than discrete tokens, to approximate token movement. It was a natural requirement because many physical systems explicitly contain fluid like quantities that are controlled by discrete logic. FSPN models are mainly intended for modeling hybrid dynamic systems that possess discrete and continuous components which evolve over time, such as traffic systems and/or computer networks. The downside of the implementation of FSPN models is that analytic evaluations of performance measures require a solution to a complex system of partial differential equations of hyperbolic type. Such solution can easily become intractable, except for small and well-structured FSPN models. Nonetheless, for more complex FSPN, where the state space is quite expanded, numeric solutions are impossible. This is due to the very fact that the number of differential equations in the system directly corresponds to the number of discrete states (state space) of the FSPN model. Considering these limitations to provide analytical behavior evaluations, the simulation approaches emerge as an important and unique alternative that offer rather convenient way of performing the required evaluation tasks.

Since the introduction of the FSPN formalism, only a few efforts describe methods for simulation of FSPN models. The reason for this deficit in flexible simulation methods that can offer solid control over the simulation process is due to the complexity of the mixed (discrete and continuous) state space of the FSPN models. All these complexities for providing solutions to FSPN models are the main reason why FSPN, even though a powerful modeling formalism, is largely avoided, especially in scientific areas that are less supported by mathematical backgrounds. This research addresses the main issues of existing FSPN simulation methods and tools, and proposes two alternative solution approaches. Both approaches involve process-based (PB) discrete-event simulations (DES), but differ in the essence on how the continuous quantities are simulated using discrete logic. As a case study, this research presents simulations and behavior evaluations of an FSPN model of Peer to Peer (P2P) Live Video Streaming (LVS) system that was infeasible to evaluate using the existing simulation techniques and/or software tools.

The main contribution of this research can be summarized as: i) state of the art about existing methods and tools for simulation of FSPN models, ii) proposal of alternative approaches for simulation of FSPN models using a PB DES language, and iii) a case study of the proposed simulation approaches used in behavior evaluations of a P2P LVS system. The proposed simulation approaches exceed the limitations and extend the capabilities of previous FSPN simulation methods, while offering higher flexibility, intuitive simulation definition, shorted simulation programming time and increased possibilities for generating statistical results, all with solid stability of the simulation process. All the observed improvements can offer the FSPN formalism to a broader range of researchers and research areas, since the FSPN

formalism has a great potential for modeling variety of complex systems that are of stochastic and concurrent nature.

The rest of the paper is organized as follows. Section 2 presents the previous related work, the motivation, as well as the proposal of the two approaches for FSPN simulation. Section 3 briefly describes the FSPN model of a P2P LVS system that is used as a case study to present the simulation results of the different approaches. Section 4 gives a technical elaboration of the two different simulation approaches that are implemented using PB DES library. The results of the behavior evaluations, along with the simulation performance results are given in section 5. Section 6 concludes the paper with a summary of contribution.

2 Motivation and related work

The first effort that suggests a method for simulation of FSPN models is proposed by David and Andrew [8]. This effort was based on CSIM toolkit [13] where the simulation is carried out by the interaction of light-weight threads called "processes", much alike the methodology proposed in this research. Intriguingly, since its introduction in 1995, process-based simulation of FSPN did not receive much attention in the following years. The proposal in [8] presents an FSPN simulator capable of generating quite adequate statistical output, presented in Table 1.

Table 1 Statistical Output of FSPN Simulator Developed in [8]

Discrete places	Maximum, average and current number of tokens Average waiting time for a token
Fluid places	Maximum, minimum, average, and current amount of fluid
Transitions	Total, mean and percentage of firing times and the number of firings

The crucial limitation of this simulator is that it was developed before a number of extensions to the standard FSPN formalism were introduced in later years, such as Non-Markovian FSPN or FSPN augmented with flush-out arcs. Besides the satisfying produced output, several statistical values are not provided, such as the deviation around the average values and the total time a certain fluid place is filled to the maximum. In simple words, the main issue as with the other existing simulation tools is that they possess similar problem of fixed range of input and confined output values. These characteristics directly influence the flexibility of the simulation process and the capabilities of the simulation tools.

Ciardo et al. [9] proposed a DES methodology for simulation of FSPN models addressing several challenges for systems with no unstable behavior. Each element of the simulation is defined by the modeler, including the generation of random numbers for transition firings. Mainly, this DES approach requires a solution to a system of ordinary differential equations at each step of the simulation, which appears to be quite costly. Problems in models that may exhibit occurrence of an infinite number of events in a finite time are not addressed since this issue cannot be managed by conventional DES techniques. Quite similar approach is presented in the work of Gribaudo and Sereno [10], with a main difference in the generation of random deviates based on non-homogeneous Poisson processes (NHPP). The last two simulation techniques are implemented in the latest version of the SPNP (Stochastic Petri Net Package) [11] tool that, besides other capabilities, from its version 6 from 1999, can simulate FSPN models as well. However, SPNP tool possesses several limitations, such as (i) the fluid flow rates between two transition firings, or until hitting a bound in a fluid place, are only linear, (ii) the guards and the flows can depend only on the discrete marking, the bounds of fluid places and thresholds in fluid places (and not on the whole state space), (iii) the cumulative measures to be computed cannot involve a fluid place as they are computed by the sum of the measure in the current state multiplied by the time to the next event (a firing of a transition or a fluid event, i.e. a bound is hit in a fluid place).

In 2002, Horton [12] proposed a novel approach for simulation of Stochastic Petri Nets (SPN) using the Proxel-Based (PXB) approach. The reason for this proposal was to leverage the accuracy of the experimental class of simulation techniques, such as DES. The main advantage of the PXB method is that it does not employ random numbers, nor it does set up a system of differential equations. Actually, it dynamically describes and follows the flow of probability among the states of the model in a very intuitive manner. Because this method works with the state-space of the model, as well as all other deterministic approaches, it suffers from state-space explosion. Actual implementations of this approach to simulate FSPN models are not determined.

It is quite interesting to note that in present times there exist a large number of tools for simulating Petri net models in general, such as TimeNET [14], QPME [15] or PIPE2 [16], to name a few. The researchers at The Department of Informatics, Faculty of Mathematics, Informatics and Natural Sciences at the University of Hamburg, have compiled a list of even 85 Petri net simulation tools that are publicly available at [18]. But, two essential reasons prevent the users to frequently employ such tools for FSPN simulation. First, these tools commonly exhibit certain limitations, usually in the range of supported input/output parameter values, that results with declined tool capabilities. Second, none of the listed tools, nor any other formal tool except SPNP, support any means to simulate FSPN models.

The elaborated limitations of the previous simulation methods were an inspiration to explore for an approach for effective simulation of FSPN models, thus the option to consider the usage of PB DES language emerged. Taking into account the previous experiences on the possible ways to simulate hybrid (discrete and continuous) logic, this research hypothesizes that FSPN models can be successfully simulated with the employment of discrete-events paradigm. The outcome of this investigation was that, besides surpassing the outlined limitations of previous developed methods, the PB DES approach offers high flexibility and virtually endless capabilities of the simulation process. All the improvements are supported with the intuitive manner of model definition, increased simulation programming speed and various ways of gathering statistical results. The next section elaborates the FSPN model as a case stydy to evaluated its behavior using the proposed simulation approaches.

3 FSPN model for performance analysis of a P2P LVS systems

The whole modeling is based on the work of Kumar et al. [17], where two important P2P streaming dimensions are defined. The first one is the maximum achievable rate that can be streamed to each individual peer at a given time, presented in eq. (1).

$$r_{MAX} = \min\left\{ r_{SERVER}, \frac{r_{SERVER} + \sum_{i=1}^{n} r_{PEER\_i}}{n} \right\} \quad (1)$$

where: r_{MAX} – maximum achievable streaming rate; r_{SERVER} – upload rate of the server; r_{PEER_i} – upload rate of the i^{th} peer; n – the total number of concurrently participating peers. Clearly, r_{MAX} is a function that depends on r_{SERVER}, r_{PEER_i} and n.

The second important dimension defined is the Universal Streaming (*UniStream*), that refers to a streaming situations when each participating peer receives the video stream with bitrate no less than the video rate, and in [17] it is achieved if and only if $r_{MAX} \geq r_{PLAY}$.

For the purpose of completeness, the FSPN model of P2P LVS system (Figure 1), introduces two additional dimensions: $r_{CONTROL}$ representing the network exchange of control messages among the participating peers and the stream function $\psi()$ which, instead of the maximum, represents the actual streaming rate to any individual peer at a given time. The boundaries of $\psi()$ are given in eq. (2).

$$\psi() = \begin{cases} r_{MAX}, & if \;\; V_{BUF} < V_{BUF\_MAX} \\ r_{PLAY} + r_{CONTROL}, & if \;\; V_{BUF} = V_{BUF\_MAX} \end{cases} \quad (2)$$

where: V_{BUF} is the amount of fluid in the peer's buffer, and V_{BUF_MAX} is the buffer's maximum capacity.

Consequently, since one more stream that drains the peer's buffer is introduced in the modeling framework, the condition for achieving *UniStream* in this FSPN model is given in eq. (3).

$$\psi() \geq r_{PLAY} + r_{CONTROL} \quad (3)$$

The performance of the system can be obtained by the calculation of the Probability for *UniStream* ($P_{UniStream}$).

The FSPN model of a P2P LVS system (Figure 1) accounts for: *network topology*, *peer churn*, *scalability*, *peer upload bandwidth heterogeneity*, *video buffering*, *control traffic overhead*, *admission control for lesser contributing peers* and *sudden disconnection for unidentified reason* given by the flush-out arc. *Asymmetric network settings* are assumed, where peers have infinite download, but limited upload bandwidths, while stream delay, peer selection strategies and chunk size are not taken into account. The considered P2P LVS system adopts mesh network topology, where peers are randomly organized into swarm groups or swarm neighborhoods, and each group member communicates with all his neighbors exchanging video chunks. Peers' upload bandwidth (UB) heterogeneity is implemented by classifying peers into two classes, high contributing peers (HP) and low contributing peers (LP) based on their UB capabilities.

The presented FSPN model comprises two main parts: the discrete part and the continuous (fluid) part of the net. Single line circles represent discrete places that accommodate discrete tokens. The tokens, which represent peers, move via single line arcs to and out of the discrete places. Fluid arcs, through which fluid is pumped, are drawn as double lines to suggest pipes. The fluid is pumped through fluid arcs and is streamed to and out of the unique fluid place P_{BUF}, which represents a single peer's buffer. The rectangles represent timed transitions with exponentially distributed firing times, and the thin short lines are immediate transitions. Peer arrival, in general, is described as a stochastic process with exponentially distributed inter-arrival times, with mean $1/\lambda$, where λ represents the arrival rate. Another assumption is made that, after joining the system, peers' sojourn times (T) are also exponentially distributed. Clearly, since each peer is immediately served after joining the system, we have a queuing network model with an infinite number of servers and exponentially distributed joining and leaving rates. The mean service time T is equal to $1/\mu$, which transferred to FSPN notation leads to the definition of the departure rate as μ multiplied by the number of peers that are concurrently being served. λ represents peers' arrival in general, but the different types of peers do not share the same occurrence probability (p_H and p_L). This occurrence distribution is defined by the immediate transitions T_{A_HP}

SIMULATING FSPN MODELS USING PROCESS-BASED DISCRETE-EVENT SIMULATION LANGUAGE
Zoran Kotevski

and T_{A_LP} and their weight functions p_H and p_L. HP arrive with rate $\lambda_H = p_H * \lambda$, and LP arrive with rate $\lambda_L = p_L * \lambda$, where $p_H + p_L = 1$. In this particular case $p_H = p_L = 0.5$, but, if needed, these occurrence probabilities can be altered. In this manner, the model with peer churn is represented by two independent *M/M/∞* Poisson processes, one for each of the different types of peers. The average number of peers that are concurrently being served defines the size of the system as a whole (S_{SIZE}) and is derived from the queuing theory as in eq. (4):

$$S_{SIZE} = \lambda/\mu \tag{4}$$

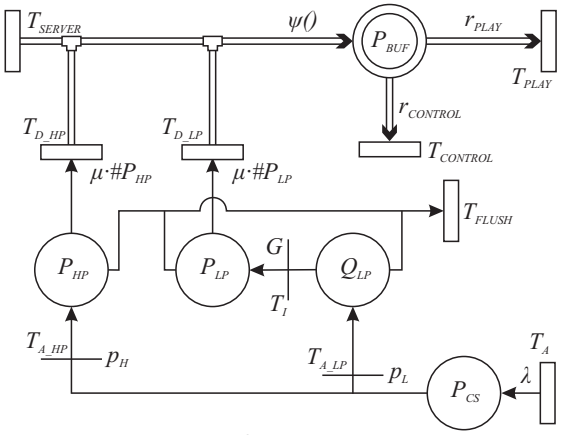

Figure 1 FSPN model of a P2P LVS system with admission control and sudden disconnection

T_A is a timed transition with exponentially distributed firing times that represents peer arrival, and upon firing (with rate λ) puts a token in P_{CS}. P_{CS} (representing the control server) checks the type of the token and immediately forwards it to one of the discrete places P_{HP} or Q_{LP} (P_{LP}). Discrete places P_{HP} and P_{LP} accommodate the different types of peers in the P2P live streaming system model. Q_{LP} on the other hand, represents queuing station for LP, which is connected to P_{LP} with the immediate transition T_I that is guarded by a *Guard function* G.

The Guard function G is a Boolean function whose values are based on a given condition. The expression of a given condition is the argument of the Guard function and serves as enabling condition for the transition T_I. If the argument of G evaluates to true, T_I is enabled. Otherwise, if the argument of G evaluates to false, T_I is disabled. For a model where admission control is not taken into account G is always enabled, but when we want to evaluate the performance of a system that incorporates admission control we should set the argument of the guard function as in eq. (5).

Transitions T_{D_HP} and T_{D_LP} are enabled only when there are tokens in discrete places P_{HP} and P_{LP}. These are marking dependent transitions, which, when enabled, have exponentially distributed firing times with rates $\mu{\cdot}\#P_{HP}$ and $\mu{\cdot}\#P_{LP}$ respectively, where $\#P_{HP}$ and $\#P_{LP}$ represent the number of tokens in each discrete place. Upon firing they take one token out of the discrete place to which they are connected.

$$G\left\{\begin{array}{l}\dfrac{r_{SERVER} + \#P_{HP}\cdot r_{HP} + (\#P_{LP}+1)\cdot r_{LP}}{\#P_{HP} + \#P_{LP} + 1} \ldots \\ \ldots \geq r_{VIDEO} + r_{CONTROL}\end{array}\right\} \tag{5}$$

Concerning the fluid part of the model, video bits are represented as atoms of fluid that travel through the fluid pipes (network infrastructure) with rate dependent on the system's state (marking). Beside the stream function as a derivative of several parameters, three separate fluid flows (streams) that travel through the network with different bitrates are identified. The main video stream represents the video data that is streamed from the source to the peers that is referred to as the *video rate* (r_{VIDEO}). The second stream is the play stream which is the stream at which each peer plays the streamed video data, referred to as the *play rate* (r_{PLAY}), and the third stream is the control traffic overhead, referred to as *control rate* ($r_{CONTROL}$), which describes the exchange of control messages needed for the logical network construction and management. As mentioned earlier, transitions T_{D_HP} and T_{D_LP} are enabled only when there are tokens in the discrete places P_{HP} and P_{LP} respectively and beside the fact that they consume tokens when firing, when enabled, they constantly pump fluid through the fluid arc to the fluid place. Flow rates of $\psi()$ are piecewise constant and depend on the number of tokens in the discrete places and their upload capabilities. Continuous place P_{BUF} represents single peer's buffer, which is constantly filled with rate $\psi()$ and drained with rate ($r_{PLAY} + r_{CONTROL}$). V_{BUF} is the amount of fluid in P_{BUF} and V_{BUF_MAX} is the buffer's maximum capacity. Transition T_{SERVER} represents the functioning of the server, which is always enabled (except when there are no tokens in any of the discrete places) and constantly pumps fluid toward the continuous place P_{BUF} with maximum upload rate of r_{SERVER}. Transition T_{PLAY} represents the video playout, which is also always enabled and constantly drains fluid from the continuous place P_{BUF}, with rate r_{PLAY}. $T_{CONTROL}$, that represents the exchange of control messages among neighboring peers, is the third transition that is always enabled, has the priority over T_{PLAY}, and constantly drains fluid from P_{BUF} with rate $r_{CONTROL}$. For further analysis, the rate of $r_{CONTROL}$ is derived from [19], where it is defined that it *linearly* depends on the number of peers in the neighborhood, and for r_{VIDEO} of 128 kbps, the protocol overhead is about 2% for a group of 64 users, which leads to a bitrate of 2.56 kbps. For the performance analyses it is assumed that peers are organized in neighborhoods with an average size of 50 members where $r_{CONTROL}$ is 2 kbps. Transition T_{FLUSH} connects the flush-out arc that represents sudden unintentional disconnection that can happen due to various reasons such as power drop for example. For the sake of convenience and chart plotting, the average upload

rate of the participating peers as $r_{AVERAGE}$ is also defined, which is given in Eq. (6):

$$r_{AVERAGE} = p_H * r_{HP} + p_L * r_{LP} \quad (6)$$

4 Simulation solution to the FSPN model

4.1 Simulation challenges and issues

Conducting performance evaluations of the behavior of a P2P LVS system using the FSPN model presented in Figure 1, requires a rather complex approach. In the following lines the issues that arise if aforementioned simulation solutions are employed to solve the FSPN model given in Figure 1 are presented.

a) The model exhibits state space explosion. Since the number of peers from a certain class that are present in the system follows a Poisson probability distribution, a good approximation of the maximum number of peers from a single class that could concurrently be present in the system is twice of the average. For a system with an average of only 50 peers for each of the two classes, the state space of the model would count approximately 10 000 possible states. This implies that an analytic solution is clearly impossible.
b) The simulation method proposed in [12] does not appear to be feasible as well, since this method works with the state-space of the model meaning that it suffers from the state-space explosion in the exact same manner as the analytic solution.
c) If we are interested in the total time a certain fluid place is filled to the maximum, we should not adhere to the simulation solution presented in [8] because this method does not provide such performance measure.
d) The approach proposed in [9] requires a solution to a system of ordinary differential equations at each step of the simulation. The presented FSPN model, depending on the presumed average system size, can exhibit state space expansion in the order of millions of states, thus the simulation approach appears quite costly to be employed.
e) In the FSPN model, the stream function $\psi()$ does not depend on the discrete marking only, but on the fluid marking of the fluid place P_c as well. Therefore, the solution of the model and the behavior estimations cannot be performed using the simulation method presented in [10].
f) The tools presented in [18] (except SPNP [9]) do not offer any means to simulate FSPN models.

Considering all these limitations it was only expected to search for an alternative approach for solving complex FSPN models for which analytic solution or the presented simulation methods appeared to be infeasible. For such FSPN models, the usage of PB DES language imposed as a promising alternative. PB DES languages are general simulation technologies, but it appeared that they could feasibly be used for simulation of FSPN models. For this purpose the SimPy simulation library was employed, which is a PB DES package based on standard Python programming language. It is quite simple, but yet extremely powerful DES package that provides the modeler with simulation *processes* that can be used for active model components (such as customers, messages or vehicles) i.e. transitions in the FSPN model, and *resource facilities* (*resources*, *levels* and *stores*), which are used for passive simulation components that form limited capacity congestion points like servers, counters, and tunnels, i.e. discrete places in FSPN notation. SimPy also provides monitor variables that help in gathering statistics, while the random variables are provided by the standard Python random module.

4.2 Simulation of the discrete part of the FSPN model

The simulation of the discrete part of the FSPN using discrete-events simulations comes quite natural. Regarding the representation of the FSPN model in SimPy terminology, all the timed transitions are described as SimPy active components, i.e. processes that act upon predefined and exponentially distributed firing times.

Table 2 Pseudocode of some places and transitions in SimPy terminology

SimPy definition of the places P_{HP} and P_{BUF}

```
define P_HP as Resource:
  resource type = Level
  initialBuffered = 0 (empty)
  enable content monitoring = True

define P_BUF as Resource:
  resource type = Level
  initialBuffered = 0 (empty)
  enable content monitoring = True
```

SimPy definition of transitions s T_{A_HP} and T_{D_HP}

```
define T_A_HP as Process:
 while simulationTime < end:
   put one token in Level P_HP
   wait exp_var_time: λ * p_H
   go to line 3

define T_D_HP as Process:
 while simulationTime < end AND P_HP not empty
   get one token from Level P_HP
   wait exp_var_time: μ * #P_HP
   go to line 3
```

The places (regardless of whether they are discrete or continuous) are described as passive SimPy components i.e. resource facilities of the type Level. Table 2 presents pseudocode of the definition of transitions and places as SimPy elements.

Besides the functional logic, the discrete part of the model is used to count the number of peers present in the queueing station Q_{LP} at the end of each simulation run, thus the average number of peers that are forced to wait is derived as an average among the total number of

simulation runs, while it is also used to calculate their waiting times.

4.3 Simulation of the fluid part of the FSPN model

For the simulation of the continuous part of the model two simulation approaches are presented. It is quite interesting to note that the simulation of continuous quantities using discrete events raised questions of whether the simulation would suffer from certain inaccuracies, thus, two approaches were tested to realize the simulation task. While both simulation approaches employ the same discrete logic explained so far, they essentially differ in the manner of simulation of the fluid part of the FSPN model.

The first approach is realized by the implementation of *Temporal-Discretization* (TD), where the simulation time is divided into short time intervals and in each interval a process performs an action that checks the system state and applies a fluid volume change (FVC) in P_{BUF} according to the current fluid flows ($\psi(), r_{PLAY}, r_{CONTROL}$). The main issue that raises using the TD approach is the accuracy of the results, because the shorter the time step is, the more accurate the results are expected to be.

The second approach performs quite differently and it does not suffer from inaccuracies, compared to the TD approach. It is an *Event-Driven* (ED) approach, which is based on the calculation of FVC in the place P_{BUF} for the timeframe between the last two events, and it is activated every time an event ocurs. It works similarly as the solutions proposed in [9, 10], except that the guard functions can depend on the combined discrete/continuous state of the model and the cumulative measures to be computed can involve a fluid place. While TD simulation approach can be considered as a *discrete* simulation, ED approach is a *continuous* one, or it can at least be defined as a hybrid simulation, in the same manner as FSPN models are hybrid (discrete and continuous) models .

For gathering the results the frequency theory of probability is used, where $P_{UniStream}$ is computed as the amount of time the system spends in *UniStream* mode against the total simulation time.

The *discrete* TD approach is applied by the definition of a so-called tracker process (or processes) that is activated in short time intervals (time step). When activated, the tracker process calculates the stream function $\psi()$ based on the system state and applies the corresponding change to the volume of fluid (V_{BUF}) in the fluid place P_{BUF}. All the possible system markings are categorized in four distinct system state cases, given in Table 3.

The rates at which fluid builds up in the fluid place P_{BUF}, in each of these four cases, can be described with the equations that are given in eq. (7).

Table 3 Categories of possible system states

IF	*THEN*
CASE 1	
$V_{BUF} = V_{BUF\_MAX}$ *AND* $r_{MAX} \geq r_{VIDEO} + r_{CONTROL}$	$\psi() = r_{VIDEO} + r_{CONTROL}$ *AND* $r_{PLAY} = r_{VIDEO}$
CASE 2	
$0 < V_{BUF} \leq V_{BUF\_MAX}$ *AND* $r_{MAX} < r_{VIDEO} + r_{CONTROL}$	$\psi() = r_{MAX}$ *AND* $r_{PLAY} = r_{VIDEO}$
CASE 3	
$0 \leq V_{BUF} < V_{BUF\_MAX}$ *AND* $r_{MAX} \geq r_{VIDEO} + r_{CONTROL}$	$\psi() = r_{MAX}$ *AND* $r_{PLAY} = r_{VIDEO}$
CASE 4	
$V_{BUF} = 0$ *AND* $r_{MAX} < r_{VIDEO} + r_{CONTROL}$	$\psi() = r_{MAX}$ *AND* $r_{PLAY} < r_{VIDEO}$

$$\frac{dV_{BUF}(t)}{dt} = \begin{cases} 0 & CASE\ 1 \\ \psi() - r_{VIDEO} - r_{CONTROL} & CASE\ 2 \\ \psi() - r_{VIDEO} - r_{CONTROL} & CASE\ 3 \\ 0 & CASE\ 4 \end{cases} \quad (7)$$

Since *UnStream* happens in three of the four system state cases, it is much easier to calculate the Degraded Service Probability (DSP), and afterwards calculate $P_{UniStream}$ as in eq. (8).

$$P_{UniStream} = 1 - DSP \quad (8)$$

For this purpose eq. (3) is modified, thus in the presented FSPN model of P2P LVS system the DSP is achieved if and only if eq. (9) is satisfied.

$$\psi() < r_{PLAY} + r_{CONTROL} \quad (9)$$

The *continuous* ED simulation approach uses the system conditions between the last two events, i.e. the last event that activated the ED calculations and it's preceding event. After each event, the changes of the volume of fluid in the fluid place is calculated for the period between those two events only. The maximum amount of FVC that can happen in the continuous place (not taking into account its limited capacity) is given by the following eq. (10).

$$\Delta V_{BUF} = (\psi() - r_{PLAY} - r_{CONTROL})\,\Delta t \quad (10)$$

Where $\Delta t = t_1 - t_0$ is the time that has passed between the two last events, and ΔV_{BUF} is the total amount of change in the volume of fluid in P_{BUF}.

The instantaneous rate of change in the fluid place would be the first derivative of eq. (10), given in eq. (11).

$$\frac{dV_{BUF}}{dt} = \psi() - r_{PLAY} - r_{CONTROL} \tag{11}$$

Similarly as with the TD approach, the DSP is only achieved if eq. (9) is satisfied. In such scenario, where the initial amount of fluid in the continuous place, between two consecutive events, is already known, the exact time at which the fluid place goes empty and the degraded service starts can be calculate using the following eq. (12).

$$\Delta t_{EMPTY} = \frac{V_{BUF\_0}}{\psi() - r_{PLAY} - r_{CONTROL}} \tag{12}$$

where V_{BUF_0} is the amount of fluid in P_{BUF} at the beginning of the last calculation period, i.e. V_{BUF} is the condition of the P_{BUF} at the moment event that happened before the last event, and it is used as a starting input variable for the calculation of the changes that happen in the timeframe between the last two events.

If $\Delta t_{EMPTY} < \Delta t$ than the time that the system spends in degraded service mode, for the period between the last two events is given in eq. (13), but otherwise DSP has not happened yet.

$$DSP_{TIME\_INTERVAL} = \Delta t - \Delta t_{EMPTY} \tag{13}$$

In this manner, the calculations are repeated on every event occurrence, which imposes certain questions about the simulation processing durations, but, since the exact timeframe that the system spends in degraded service mode is calculated without any compromise to the accuracy, the ED simulation approach imposes as a quite valuable alternative.

5 Simulation results and performance

Using the previously elaborated simulation approaches the presented FSPN model is simulated while custom tailoring the required output. Even though the SimPy simulation package offers some built-in functions for gathering statistics, the tremendous control over the simulation process that SimPy offers enables to define a statistical output that satisfies any user requirement. The performed behavior evaluations are based on the following input parameters: $r_{SERVER} = 700\ kbps$, $r_{HP} = 700\ kbps$, $r_{LP} = 100\ kbps$. T_{FLUSH} fires with rate $\lambda_{FLUSH} = 5.5 * 10^{-5}$, i.e. it fires, on average, every 5 hours. In most common cases T_{FLUSH} would fire once or twice during the 10 hours of simulation time. The 10 hours of simulation duration are set for a single simulation run, but for the calculation of a single point on the performance charts an average of 75 simulation runs is obtained. The firing rates of T_{D_HP} and T_{D_LP} are marking dependent, where they are calculated as the departure rate (μ) multiplied by the number of tokens in the corresponding discrete place. It is assumed that the average sojourn time T is 45 minutes, thus $\mu = 3.7 * 10^{-4}\ [peers * sec^{-1}]$.

5.1 Behavior evaluation of the modeled P2P LVS system

In the following figures, several performance charts representing different aspects of the system's behavior are presented, but it must be noted that much more performance measures can be obtained depending on the modeler intentions and the model requirements. Figure 2 presents the probability for *UniStream* for systems of different magnitudes. Surprisingly, from all the presented charts the main conclusion is that both simulation approaches perform nearly identically. The performance of the P2P LVS system rises linearly with the system up-scaling, but only up to a certain point when $r_{VIDEO} \leq r_{AVERAGE}$.

As r_{VIDEO} continues to rise, $P_{UniStream}$ more and more steeply declines with system growth. This means that large systems, i.e. systems with many concurrently connected peers, exhibit better overall performance and provide higher quality of user experience compared to the smaller-sized systems.

Figure 3 and Figure 4 present the average number of low contributing peers that are forced to wait for entrance in the system, as well as their average waiting times, for the four different system sizes. It is quite interesting to note that the waiting times in the different system sizes are quite similar, while the number of waiting peers largely differs among the systems with different system magnitude, which is certainly an expected behavior. Nevertheless, the waiting times are about 14 minutes, while the average number of peers that can be found in the queueing place is about 14 for the smallest system and about 170 for the largest system.

SIMULATING FSPN MODELS USING PROCESS-BASED DISCRETE-EVENT SIMULATION LANGUAGE
Zoran Kotevski

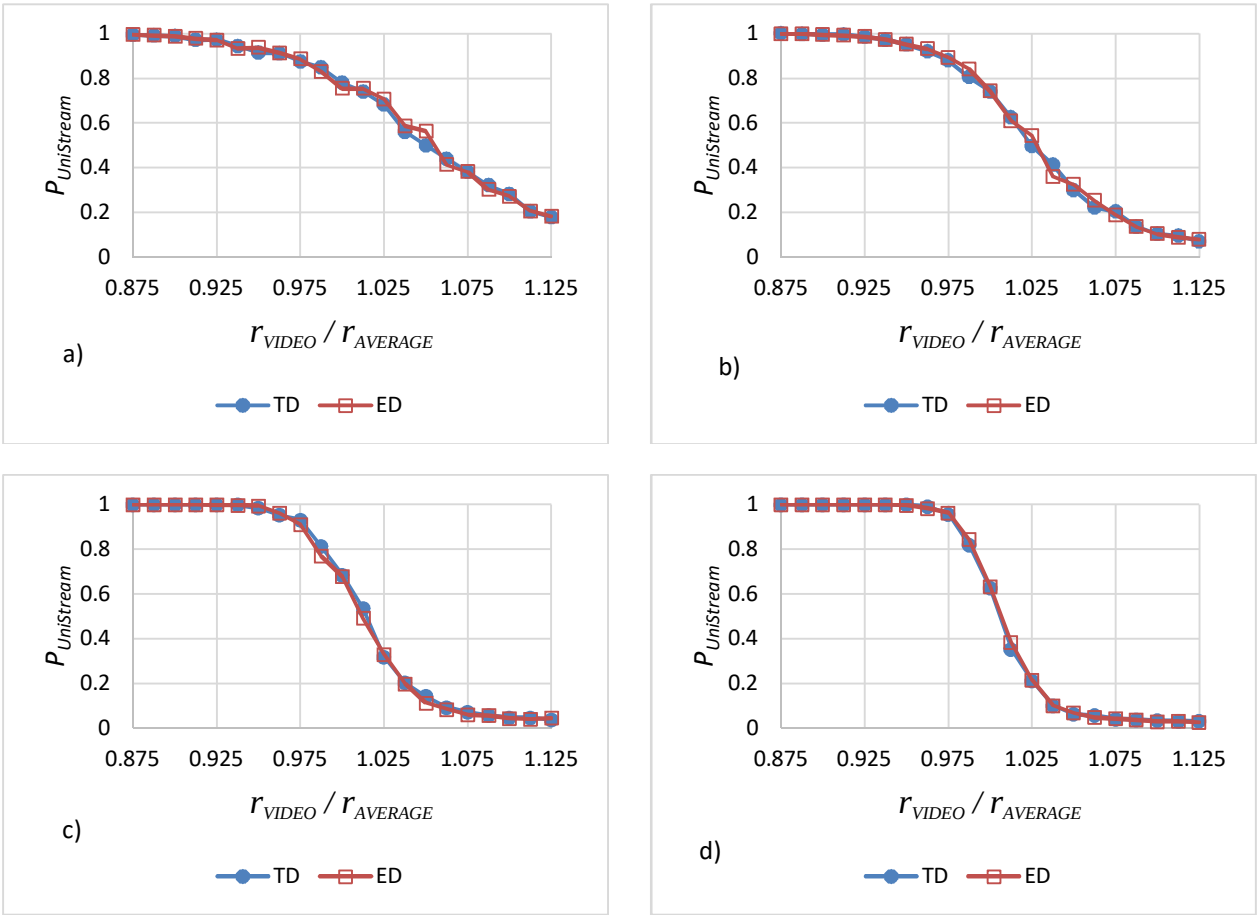

Figure 2 Performance of the P2P LVS system for the different average system magnitudes:
a) average magnitude of 100 peers; b) average magnitude of 200 peers
c) average magnitude of 500 peers; d) average magnitude of 1000 peers

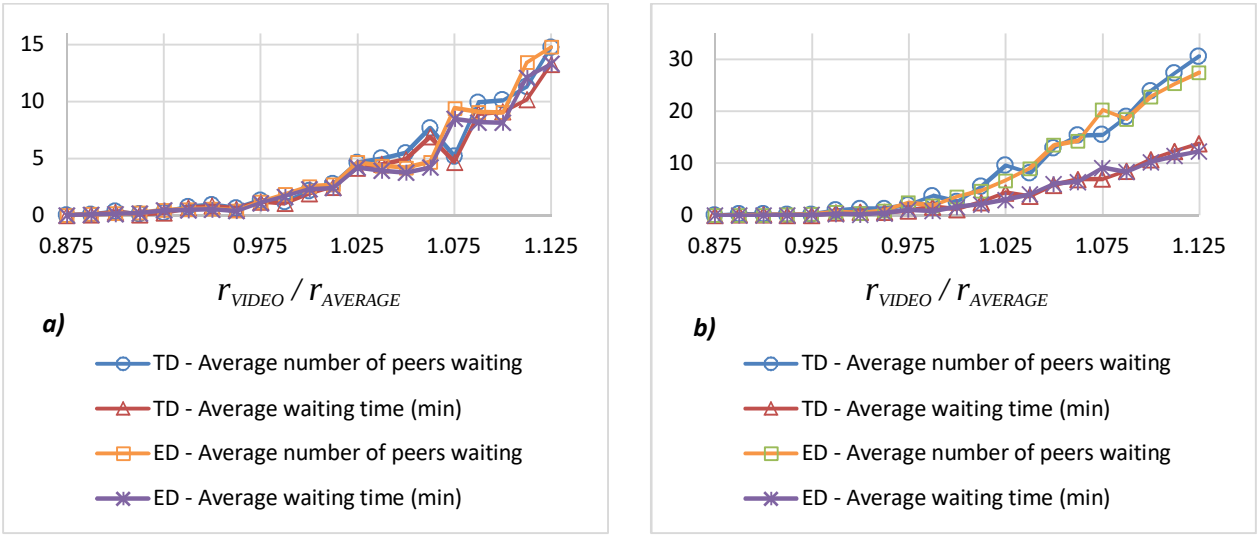

Figure 3 Average number of peers waiting and waiting times for small to medium systems
a) average magnitude of 100 peers; b) average magnitude of 200 peers

SIMULATING FSPN MODELS USING PROCESS-BASED DISCRETE-EVENT SIMULATION LANGUAGE
Zoran Kotevski

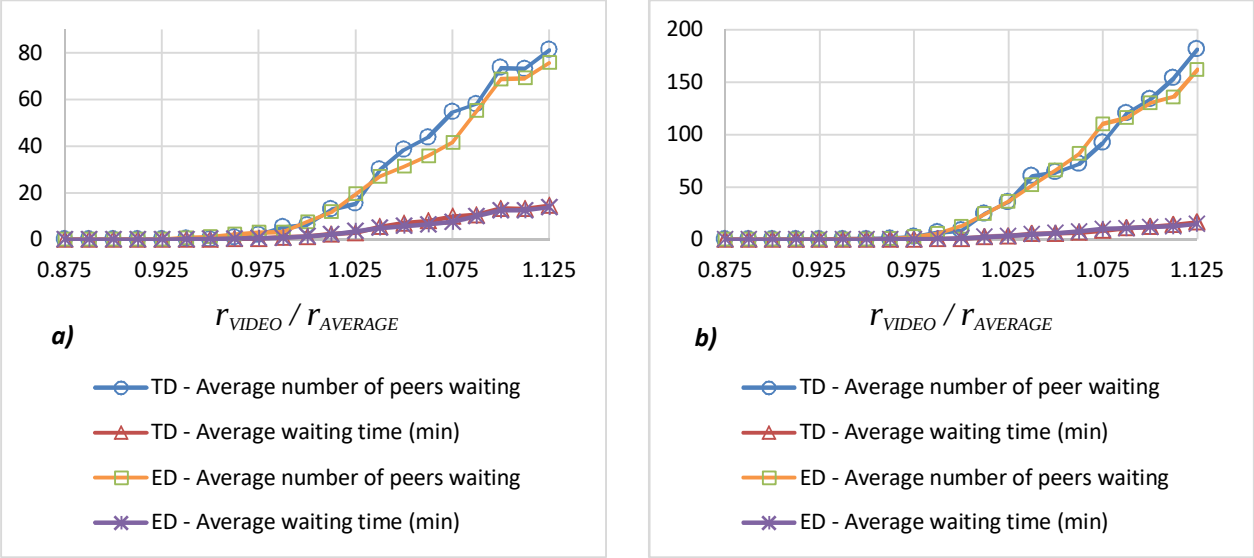

Figure 4 Average number of peers waiting and waiting times for medium to large systems a) average magnitude of 500 peers; b) average magnitude of 1000 peers

5.2 Performance comparison of the different simulation approaches

Simulation performance mainly depends on the approach applied for the simulation of the continuous part of the FSPN model, since the simulation of the discrete part is identical in both simulation approaches. The number of performed FVC calculations at the TD approach depends on the predefined time step, and it is exact. On the other hand, the number of FVC calculations at the ED approach depends on the number of the events occurred at the timed transitions, regardless whether they are arrivals or departures. This fact for the ED approach implies that the number of FVC calculations would be different for each ED simulation run with same input values, but will float around the average. What is more interesting to note is that it is expected that the ED simulation durations would increase with the system up-scaling, since the number of occurring events is much higher for systems of greater magnitudes. In Table 4, the expected mean number of total FVC calculations for both simulation approaches, for a single simulation run, are presented. The data implies that using the continuous (ED) simulation approach is time beneficial only for smaller system magnitudes, while for the larger systems it can be quite processing intensive.

Table 5 presents a comparison of the actual simulation durations of the two approaches and the different system magnitudes. It appears that the expected number of FVC calculations highly influences the performance of the simulations. Of course, the intensity of calculations for the fluid part differs among the different approaches, thus for the system of a magnitude of 500 peers, even though it is expected for the ED approach to be processed more quickly, the simulations last longer than expected. Nevertheless, the main reason to use ED approach was the accuracy of the processing of the continuous quantities, which was used to obtain important conclusions concerning both simulation approaches.

Table 4 Average number of expected FVC calculations per simulation approach for various system magnitudes and time step of 2 sec. for the TD approach

Simulation duration for a single run [sec]	Average time to peer arrival [sec]	Total arrivals on average	Total depatures on average	Total calculations on average (ED)	Total calculations on average (TD)
36000	54	667	617	1283	18000
36000	27	1333	1233	2567	18000
36000	13.5	2667	2467	5133	18000
36000	5.4	6667	6167	12833	18000
36000	2.7	13333	12333	25667	18000

Concerning the TD approach, the main expected issue was how it would deal with the accuracy, because it was expected that is would depend on the time step. As mentioned earlier in the paper, it is expected that the shorter the time step is the more accurate the results would be. The time step for a certain simulation should be much shorter that the average time that passes between two consecutive peer arrivals.

Table 5 Performance of the different simulation approaches and system sizes

Simulation approach	Average system magnitude [peers]	Arrival rate [peers/second]	Average time to peer arrival [seconds]	Time step [seconds]	Simulation duration [seconds]
TD	100	0.037	27	2	1387
ED	100	0.037	27	/	639
TD	200	0.074	13.5	2	2472
ED	200	0.074	13.5	/	1828
TD	500	0.185	5.4	2	6916
ED	500	0.185	5.4	/	8325
TD	1000	0.37	2.7	2	17030
ED	1000	0.37	2.7	/	26701

As given in Table 5, the system with an average magnitude of 100 peers has a solid difference between the average time to peer arrival and the time step for TD simulations, thus high accuracy of the results obtained was expected. But, the TD simulation of the large system with an average of 1000 peers was set with a time step that is almost equal to the average time to peer arrival. Nevertheless, even though the expectations were that is would exhibit certain inaccuracies, the results presented in the previous section show that the accuracy of TD approach is not compromised, at least in these tested situations when the time step is less or almost equal to the average time to peer arrival.

Conclusions

FSPN formalism extends the basic capabilities of general Petri net modeling in a way that it allows fluid like quantities to reside in Petri Net places. Even though this innovation enhances the analytic possibilities of Petri nets, it imposes many new challenges by leveraging the complexity of the solution methods and lowering the number of simulation techniques that can be feasibly utilized. In this paper, the existing techniques that are proposed for simulation of FSPN models are analyzed, and main limitations and issues are elaborated. To overcome the presented challenges, this research proposes two different simulation approaches for simulating FSPN models by the use of PB DES simulation paradigm. The presented approaches appear to be highly flexible providing nearly endless capabilities for control over the simulation process. All the simulation enhancements are augmented with intuitiveness of the modeling process, as well as an increased speed of simulation programming. In addition, simulations using PB DES language offers various means of gathering statistic results.

The integrity of the proposal is supported by a case study of an FSPN model simulation using SimPy library that is used to analyze the performance of a P2P LVS system. The presented simulation results imply that PB DES languages can be feasibly used to simulate complex FSPN models. In this manner two different simulation approaches are explored. The first (TD) approach is a discrete approach where simulation of fluid quantities is performed using temporal discretization, while the other (ED) approach can be considered as a continuous one. Comparison of the results obtained confirms that both approaches offer solid performance and provide nearly identical results. The main difference between the two approaches are the processing durations that depend mainly on the number of the discrete events that occur during the simulation process for the ED approach, and the predefined time step for the TD approach.

Considering the possibilities of the presented simulation approaches, as well as the modeling capabilities of FSPN paradigm, we can expect to bring the FSPN formalism to a wider range of researcher profiles. We believe that the discipline of FSPN modeling and analysis of systems can be introduced in even more research areas of various scientific fields which deal with models that are of stochastic, dynamic and concurrent nature.

References

[1] TRIVEDI, K.S., KULKARNI, V.G.: '*FSPNs: Fluid Stochastic Petri Nets*', Proceedings of the 14th International Conference on Application and Theory of Petri Nets, pp 24-31, 1993.

[2] WOLTER, K., HORTON, G., GERMAN, R.: Non-Markovian Fluid Stochastic Petri Nets. *Technical Report, Technical University of Berlin*, 1996.

[3] WOLTER, K., GERMAN, R.: '*Second Order Non-Markovian Fluid Stochastic Petri Nets*', Proceedings of the Workshop on Performability Modeling of Computer and Communication Systems, 1996.

[4] HORTON, G., KULKARNI, V.G., NICOL, D.M., TRIVEDI K.S.: Fluid Stochastic Petri Nets: Theory, Applications and Solutions Techniques, *EU Journal of Operational Research*, 105(1):184-201, 1998.

[5] GRIBAUDO, M., HORVATH, A.: Fluid Stochastic Petri Nets Augmented with Flush-out Arcs: A Transient Analysis Technique, *IEEE Transactions on Software Engineering*, 28(10):944-955, 2002.

[6] WOLTER, K.: '*Second Order Fluid Stochastic Petri Nets: An Extension of GSPNs for Approximate and Continuous Modelling*', Proceedings of World Congress on System Simulation, pp. 328-332, 1997.

[7] GRIBAUDO, M., SERENO, M., BOBBIO, A.: '*Fluid Stochastic Petri nets: An Extended Formalism to Include Non-Markovian Models*', Proceedings of the 8th International IEEE Workshop on Petri Nets and Performance Models, pp. 74-81, 1999.

[8] DAVID, M.N., ANDREW, S.M.: '*The Fluid Stochastic Petri Net Simulator*', Proceedings of the Sixth International Workshop on Petri Nets and Performance Models (PNPM '95), pp:214-215, 1995.

[9] CIARDO, G., NICOL, D., TRIVEDI, K.S.: Discrete-event Simulation of Fluid Stochastic Petri Nets, *IEEE Transactions of Software Engineering*, 25(2):207-217, 1999.

[10] GRIBAUDO, M., SERENO, M.: '*Simulation of Fluid Stochastic Petri Nets*', 8th IEEE International Symposium on Modeling Analysis and Simulation of Computer and Telecommunication Systems, pp 231-239, 2000.

[11] CIARDO, G., MUPPALA, J., TRIVEDI, T.: '*SPNP: Stochastic Petri Net Package*', Proceedings of the Third IEEE International Workshop on Petri Nets and Performance Models (PNPM89), pp. 142-151, 1989.

[12] HORTON, G.: '*A New Paradigm for the Numerical Simulation of Stochastic Petri Nets With General Firing Times*', Proceedings of the European Simulation Symposium, 2002.

[13] SCLHWETMAN, H.: '*CSIM: A C-based, Process Oriented Simulation Language*', Proceedings of the 1986 Winter Simulation Conference, pages 387-396, 1986.

[14] GERMAN, R., KELLING, C., ZIMMERMANN, A., HOMMEL, G.: TimeNET: a Toolkit for Evaluating Non-Markovian Stochastic Petri Nets, *Performance Evaluation*, *24*(1):69-87, 1995.

[15] KOUNEV, S., DUTZ, C., BUCHMANN, A.: '*QPME – Queueing Petri Net Modeling Environment*', Proceedings of the Third IEEE International Conference on Quantitative Evaluation of Systems, (QEST 2006), pp.115-116, 2006.

[16] DINGLE, N.J., KNOTTENBELT, W.J., SUTO, T.: PIPE2: A Tool for the Performance Evaluation of Generalised Stochastic Petri Nets, *ACM SIGMETRICS Performance Evaluation Review*, 36(4):34-39, 2009.

[17] KUMAR, R., LIU, Y., ROSS, K.: Stochastic Fluid Theory for P2P Streaming Systems. *IEEE INFOCOM*, Anchorage, USA, pp. 919-927, 2007.

[18] Petri Nets Tools Database Quick Overview, Universität Hamburg, Germany, [Online], Available: http://www.informatik.uni-hamburg.de/TGI/PetriNets/tools/quick.html [12 June 2018], 2018.

[19] GUO, H., LO, K.T., QIAN, Y., LI, J.: Peer-to-Peer Live Video Distribution under Heterogeneous Bandwidth Constraints, *IEEE Trans on Parallel and Distributed Systems*, 20(2): 233–245, 2009.

Review process
Single-blind peer review process.