2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks

Efficient transient analysis of a class of compositional Fluid Stochastic Petri Nets

Peter Buchholz
Informatik IV
TU Dortmund
D–44221 Dortmund, Germany
Email: peter.buchholz@cs.tu-dortmund.de

Tuğrul Dayar
Department of Computer Engineering
Bilkent University
TR-06800 Ankara, Turkey
Email: tugrul@cs.bilkent.edu.tr

***Abstract*—Fluid Stochastic Petri Nets (FSPNs) which have discrete and continuous places are an established model class to describe and analyze several dependability problems for computer systems, software architectures or critical infrastructures. Unfortunately, their analysis is faced with the curse of dimensionality resulting in very large systems of differential equations for a sufficiently accurate analysis. This contribution introduces a class of FSPNs with a compositional structure and shows how the underlying stochastic process can be described by a set of coupled partial differential equations. Using semi discretization, a set of linear ordinary differential equations is generated which can be described by a (hierarchical) sum of Kronecker products. Based on this compact representation of the transition matrix, a numerical solution approach is applied which also represents transient solution vectors in compact form using the recently developed concept of a Hierarchical Tucker Decomposition. The applicability of the approach is presented in a case study analyzing a degrading software system with rejuvenation, restart, and replication.**

***Index Terms*—Performance and Dependability Analysis, Hybrid Stochastic Models, Fluid Stochastic Petri Nets, Numerical Algorithms, Data Structures**

I. Introduction

Stochastic models including discrete and continuous state variables are an important modeling framework for performance and dependability analysis beyond pure discrete state discrete event systems. Continuous variables are used to abstract from a large number of discrete entities [2] or to describe some continuous physical quantity such as the filling of a battery or a tank [13], [24]. *Fluid Stochastic Petri Nets* (FSPNs) which have discrete and continuous places are a model class that combines the advantages of Petri Nets with the possibility to specify hybrid stochastic systems. Several classes of *hybrid* Petri Nets have been defined, which differ in various details. We consider here FSPNs [18] in an extended version with flush-out arcs and marking dependent rates [16].

Although FSPNs are known for some time and have even been extended to allow a compositional description of systems [1], [15], the analysis is often restricted to relatively simple models with constant flow rates. One has two general possibilities to analyze an FSPN, hybrid stochastic simulation or the numerical solution of the set of partial differential equations (PDEs) described by the FSPN. As for discrete state models, numerical analysis has the advantage of providing more accurate results, in particular, if the evolution of the system behavior over some time interval has to be analyzed, as it is necessary for many problems. However, numerical analysis is faced with the problem of *state space explosion* which limits the applicability of the approach to relatively small models. Stochastic simulation, in principle can be applied to arbitrary models, but it should be remarked that simulation of FSPNs has to combine the analysis of the continuous evolution of the fluid part, which has to be analyzed by some form of discretization, and marking dependent firing rates of transitions, resulting in a non-homogeneous behavior according to the discrete part. This implies that available simulators cannot handle general FSPNs and simulation can be time consuming [8]. To the best of our knowledge, only prototype simulators are available for FSPNs which allow only the analysis of restricted classes of models. Apart from the specific problems of simulating FSPNs, the determination of the transient behavior of stochastic models over an interval of time often results in large confidence intervals because estimators are dependent over a time series.

We consider in this paper the numerical analysis of FSPNs and try to alleviate the problem of state space explosion. Two steps are introduced for this purpose. First, we introduce a class of compositional FSPNs which allow one to specify models by the composition of small components and use the component structure to define a multi-dimensional state space. For analysis, the fluid part has to be discretized which often results in a large discrete state space with very regular structure. The transition matrix of the discretized system of equations is presented by a block structured matrix where each submatrix results from the Kronecker product of small component matrices. The composition and the resulting matrix structure is more general than the one proposed in [15]. This first step results in a very compact representation of the transition matrix which can be used in iterative numerical techniques as done for discrete state Markov models for some time [9]. However, this step is usually not sufficient to analyze very large models because solution vectors still have lengths equal to the size of the state space. For analyzing very large models, a second step is necessary to represent solution vectors in a more compact way which usually implies the introduction of an approximation. For this purpose, recently developed data

2158-3927/18/$31.00 Â©2018 IEEE
DOI 10.1109/DSN.2018.00039

structures for higher dimensional tensors can be used [17]. We use a *Hierarchical Tucker Decomposition* (HTD) which has already been applied for the analysis of PDEs [20] and for the stationary analysis of Markov chains [5]. Here we extend the approaches to the transient analysis of FSPNs.

The new contributions of the paper are a compositional class of FSPNs, the compact representation of the transition matrix after spatial discretization, the integration of the compact matrix structure and the HTD data structure for vectors in numerical solvers for transient analysis, a full C-implementation of the approach to obtain numerical results, and the application of the method to an example from software dependability. The presented solution methods can also be applied to discrete state Markov models, but they are especially well suited to analyze hybrid systems due to the regular structure of the discrete state space after spatial discretization.

In the next section, we first present the class of FSPNs. In Section III, the stochastic process resulting from spatial discretization is introduced. Then, in Section IV, transient numerical analysis and the HTD data structure for vectors are introduced. Section V includes the application example. The paper ends with the conclusions and a short outlook on possible extensions.

II. Compositional Fluid Stochastic Petri Nets

We consider Fluid Stochastic Petri Nets (FSPNs) similar to the classes that have been defined in [8], [15], [16], [18].

Definition 1: A Fluid Stochastic Petri Net (FSPN) is an 8-tuple denoted by $(\mathcal{P}, \mathcal{T}, M_0, \mathcal{A}, B, W, F, R)$, where

- $\mathcal{P}$ is the set of places which is subdivided into the set of discrete places $\mathcal{P}_d$ and the set of continuous places $\mathcal{P}_c$;
- $\mathcal{T}$ is the set of transitions partitioned into a set of stochastically timed transitions $\mathcal{T}_e$ and a set of immediate transitions $\mathcal{T}_i$,
- $M_0 = (\boldsymbol{m}_0, \boldsymbol{x}_0)$ is the initial marking, where $\boldsymbol{m}_0 \in \mathbb{N}^{|P_d|}$ is a row vector containing the number of tokens in each discrete place and $\boldsymbol{x}_0 \in \mathbb{R}_{\geq 0}^{|P_c|}$ is a row vector which contains for each continuous place the level of fluid at the place; $\mathcal{M}_d$ is the set of all reachable discrete markings, $\mathcal{M}_c$ is the set of all reachable continuous markings, and $\mathcal{M}$ ($\subseteq \mathcal{M}_d \times \mathcal{M}_c$) is the set of all reachable markings;
- $\mathcal{A}$ is the set of arcs which is subdivided into the set of discrete arcs $\mathcal{A}_d : ((\mathcal{P}_d \times \mathcal{T}) \cup (\mathcal{T} \times \mathcal{P}_d)) \rightarrow \mathbb{N}$ (where $\mathcal{A}_d$ defines the multiplicity of the arc), the set of continuous arcs $\mathcal{A}_c : (\mathcal{P}_c \times \mathcal{T}_e) \cup (\mathcal{T}_e \times \mathcal{P}_c) \rightarrow \{0, 1\}$, and the set of flush-out arcs $\mathcal{A}_f : (\mathcal{P} \times \mathcal{T}) \rightarrow \{0, 1\}$;
- The function $B : \mathcal{P}_c \rightarrow \mathbb{R}_{>0}$ defines the capacities of continuous places;
- The weight function $W : \mathcal{T}_i \rightarrow \mathbb{R}_{>0}$ for immediate transitions has the usual meaning;
- The firing rate function $F : \mathcal{T}_e \times \mathcal{M} \rightarrow \mathbb{R}_{\geq 0}$ defines the transition rates of timed transitions in each marking;
- The flow rate function $R : \mathcal{A}_c \times \mathcal{M} \rightarrow \mathbb{R}_{\geq 0}$ defines the marking dependent rate of fluid flow across continuous arcs.

For transition t, we define the sets of discrete input and output places $\bullet t = \{p \in \mathcal{P}_d \mid \mathcal{A}_d(p,t) > 0\}$ and $t\bullet = \{p \in \mathcal{P}_d \mid \mathcal{A}_d(t,p) > 0\}$, the sets of continuous input and output places $\circ t = \{p \in \mathcal{P}_c \mid \mathcal{A}_c(p,t) = 1\}$ and $t\circ = \{p \in \mathcal{P}_c \mid \mathcal{A}_c(t,p) = 1\}$, and the set of places connected by a flush-out arc $\diamond t = \{p \in \mathcal{P} \mid \mathcal{A}_f(p,t) = 1\}$. Note that in our setting, flushing out of places by immediate transitions is also possible. The input and output transitions of places (discrete or continuous) $\bullet p$ and $p\bullet$ are defined similarly. Furthermore, $p\diamond$ for $p \in \mathcal{P}$ is the set of transitions connected via a flush-out arc. The notation can be naturally extended to sets of places or transitions.

The dynamic behavior of FSPNs will only be briefly introduced here; details can be found in the above mentioned papers (e.g., in [16], [18]). Observe that the enabling of transitions only depends on the discrete marking. That is, transition $t \in \mathcal{T}$ is enabled in marking $M = (\boldsymbol{m}, \boldsymbol{x}) \in \mathcal{M}$ if for all $p \in \mathcal{P}_d$, we have $\mathcal{A}_d(p,t) \leq \boldsymbol{m}_p$.

We let $\mathcal{E}(\boldsymbol{m})$ denote the set of all enabled transitions in discrete marking $\boldsymbol{m} \in \mathcal{M}_d$. If immediate transitions are enabled, they have priority over timed transitions, i.e., timed transitions fire if $\mathcal{E}(\boldsymbol{m}) \cap \mathcal{T}_i = \emptyset$. If transition $t \in \mathcal{T}$ fires in marking $M \in \mathcal{M}$, then the new marking $M' = (\boldsymbol{m'}, \boldsymbol{x'})$ is given by $\boldsymbol{m'}_p = \boldsymbol{m}_p - \mathcal{A}_d(p,t) + \mathcal{A}_d(t,p)$ and $\boldsymbol{x'} = \hat{\boldsymbol{x}}$, where $\hat{\boldsymbol{x}}_p = 0$ if $p \in \diamond t$ and $\hat{\boldsymbol{x}}_p = \boldsymbol{x}_p$ if $p \notin \diamond t$. The firing of a transition flushes out all the places that are connected with flush-out arcs to t. We denote this as $\boldsymbol{m} \stackrel{t}{\rightarrow} \boldsymbol{m'}$. Firing times of timed transitions are exponentially distributed with rates $F(t, M)$. Marking dependent transition rates are very powerful, if they are applied in full generality, because they may be used to set the rate of an enabled transition to zero and can thus simulate inhibitor arcs.

For continuous place $p_c \in \mathcal{P}_c$ and marking $M \in \mathcal{M}$, the potential change rate of the fluid level for all timed transitions $t \in \mathcal{E}(M)$ is given by

$$\hat{r}_c(M) = \sum_{t \in \mathcal{E}(M) \cap \bullet p_c} R((t, p_c), M) - \sum_{t \in \mathcal{E}(M) \cap p_c \bullet} R((p_c, t), M).$$

The potential flow does not consider the bounds of fluid places which are 0 and $B(p_c)$. Now, let $M(\tau) \in \mathcal{M}$ be the marking at time $\tau \geq 0$. The actual change of the fluid in continuous place $p_c \in \mathcal{P}_c$ is then given by

$$r_c(M(\tau)) = \partial \boldsymbol{x}_c(\tau)/\partial\tau = \begin{cases} \hat{r}_c(M(\tau)) & \text{if } (0 < \boldsymbol{x}_c(\tau) < B(p_c) \wedge \\ & \quad (\hat{r}_c(M(\tau^-))\hat{r}_c(M(\tau^+)) \geq 0)) \ \vee \\ & \quad (\boldsymbol{x}_c(\tau) = 0 \wedge \hat{r}_c(M(\tau) > 0)) \vee \\ & \quad (\boldsymbol{x}_c(\tau) = B(p_c) \wedge \hat{r}_c(M(\tau) < 0)) \\ 0 & \text{otherwise.} \end{cases}$$

Example. As a running example, we choose a simple FSPN which has been introduced in [16]. The net is shown in Figure 1; for an interpretation of its behavior we refer to the original paper. The net consists of two discrete and two continuous places. We assume that transitions $t_1, \ldots, t_4$ have marking independent firing rates $\lambda_1, \ldots, \lambda_4$. Transition t_1 upon firing flushes out both continuous places, whereas t_2

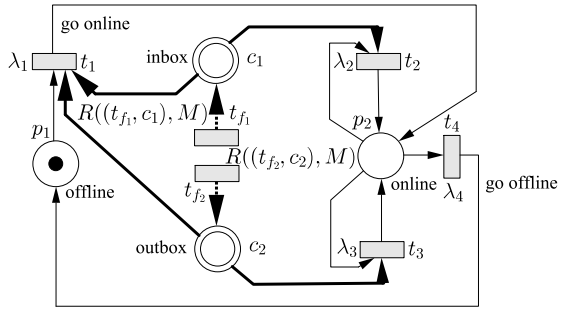

Fig. 1. Simple FSPN for a person using a mailer.

and t_3 upon firing flush out continuous places c_1 and c_2, respectively. We assume that the fluid upper bounds of the continuous places c_1 and c_2 are normalized to 1. For the transitions t_{f_i}, we assume a firing rate of 1, which is only relevant as a scaling factor since the transitions are always enabled and the firing does not modify the discrete marking. Transitions t_{f_i} fill the adjacent continuous place c_i with fluid. We consider different functions, $R((t_{f_i}, c_i), M)$, namely (for $i = 1, 2$)

1) $\frac{dx^{(i)}}{dt} = \alpha_i$,
2) $\frac{dx^{(i)}}{dt} = 2\alpha_i(1 - x^{(i)})$,
3) $\frac{dx^{(i)}}{dt} = 2\alpha_i(1 - x^{(i)})(1.8x^{(j)} + 0.1)$
   where $j \neq i,\ j = 1, 2$.

■

We consider a compositional approach to describe and analyze FSPNs. Compositional analysis of discrete Petri nets without continuous places is known for several decades. A well known class are superposed SPNs and GSPNs [11], [12]. For FSPNs, compositional approaches have been proposed in [1], [15]. Although these papers are the first to introduce ideas to represent the transition rate matrix of the discretized stochastic process in a compact form using Kronecker products, they do not exploit the resulting structure for a numerical analysis.

A superposed FSPN (SFSPN) is an FSPN in which the set of places is partitioned into disjoint subsets $\mathcal{P}^{(j)}$ ($j = 1, \ldots, J$). Then every marking $M \in \mathcal{M}$ can be represented as $(M^{(1)}, \ldots, M^{(J)})$, where $M^{(j)}$ considers only the places in $\mathcal{P}^{(j)}$. This partition naturally defines sets of transitions for subnets. Transition t belongs to subnet j if $t \in {}^\bullet\mathcal{P}^{(j)} \cup \mathcal{P}^{(j)} {}^\bullet \cup \mathcal{P}^{(j)}{}^\diamond$ and it is local to subnet j if additionally $t \notin ({}^\bullet\mathcal{P}^{(i)} \cup \mathcal{P}^{(i)} {}^\bullet \cup \mathcal{P}^{(i)}{}^\diamond)$ for all $i \neq j$ and the rate of the transition depends only on the marking of places from $\mathcal{P}^{(j)}$. Now, let $\mathcal{T}^{(j)}$ be the set of transitions belonging to subnet j. Then $(\mathcal{P}^{(j)}, \mathcal{T}^{(j)}, M_0^{(j)}, \mathcal{A}^{(j)}, B^{(j)}, W^{(j)}, F^{(j)}, R^{(j)})$ defines a subnet, where the functions are restricted to the subsets $\mathcal{P}^{(j)}$ and $\mathcal{T}^{(j)}$. A subnet only acts independently of its environment if all transitions are local. In this case, the subnet does not interact with its environment and can obviously be analyzed independently.

Example. (continued) We use the following partition of the places $\mathcal{P}^{(1)} = \{p_1, p_2\}$, $\mathcal{P}^{(2)} = \{c_1\}$, and $\mathcal{P}^{(3)} = \{c_2\}$ resulting in $\mathcal{T}^{(1)} = \{t_1, t_2, t_3, t_4\}$, $\mathcal{T}^{(2)} = \{t_1, t_2, t_{f_1}\}$, and $\mathcal{T}^{(3)} = \{t_1, t_3, t_{f_2}\}$. Thus, we have one subnet with two discrete places and two subnets with one continuous place each. ■

Although we consider dependencies among subnets, we need some restrictions for an efficient analysis exploiting the decompositional structure. Our restrictions are less strict than those in [15], but they obviously put some limitations on the nets that can be analyzed with the proposed approach. In the following, we assume that:

1) The set of discrete markings, $\mathcal{M}_d$, is finite. For the analysis that follows, we assume that vanishing markings associated with immediate transitions are omitted and $\mathcal{M}_d$ contains only tangible markings. This requires the effect of immediate transitions to be included in the timed transitions between tangible markings which implies that all immediate transitions are local or the subnet can be transformed to a subnet with only local immediate transitions by an equivalence relation.
2) Each continuous place $p_c \in \mathcal{P}_c$ has a finite bound $B(p_c) < \infty$.
3) For each continuous place $p_c \in \mathcal{P}_c$, fixed discrete marking $\boldsymbol{m} \in \mathcal{M}_d$, and all $\boldsymbol{x} \in \mathcal{M}_c$, one of the following three mutually disjoint conditions holds:
   a) $r_c(\boldsymbol{m}, \boldsymbol{x}) > 0$ for $\boldsymbol{x}_c < B(p_c)$ and $r_c(\boldsymbol{m}, \boldsymbol{x}) = 0$ for $\boldsymbol{x}_c = B(p_c)$, or
   b) $r_c(\boldsymbol{m}, \boldsymbol{x}) < 0$ for $\boldsymbol{x}_c > 0$ and $r_c(\boldsymbol{m}, \boldsymbol{x}) = 0$ for $\boldsymbol{x}_c = 0$, or
   c) $r_c(\boldsymbol{m}, \boldsymbol{x}) = 0$.
4) Each subnet contains at most one continuous place such that the local state can be expressed as $(\boldsymbol{m}^{(j)}, x^{(j)})$, where $x^{(j)}$ is the filling of the local continuous place and $x^{(j)}$ is omitted if subnet j contains no continuous place. If the subnet contains no discrete place, $\boldsymbol{m}^{(j)}$ is omitted.
5) The function $F(t, M)$ is decomposable; that is,

$$F(t, M) = \lambda_t \prod_{j=1}^{J} g^{(j)}(t, M^{(j)}),$$

   where $\lambda_t \in \mathbb{R}_{>0}$ is constant and $g^{(j)}(t, (\boldsymbol{m}^{(j)}, x^{(j)}))$ are nonnegative functions that are continuous in $x^{(j)}$.
6) Like the transition rate function, flow rates for places $p_c \in \mathcal{P}_c$ and transitions $t \in {}^\bullet p_c$ can be decomposed as

$$R((t, p_c), M) = \lambda_t \prod_{j=1}^{J} f_{in}^{(j)}((t, p_c), M^{(j)}),$$

   where $f_{in}^{(j)}((t, p_c), (\boldsymbol{m}^{(j)}, x^{(j)}))$ are continuous functions in $x^{(j)}$ and $f_{in}^{(j)}((t, p_c), (\boldsymbol{m}^{(j)}, x^{(j)})) > 0$ if $p_c \notin \mathcal{P}^{(j)}$, and $f_{in}^{(j)}((t, p_c), (\boldsymbol{m}^{(j)}, x^{(j)})) \geq 0$ if $p_c \in \mathcal{P}^{(j)}$

for all $x^{(j)} \in (0, B(p_c))$. Similarly, flow rates for places $p_c \in \mathcal{P}_c$ and transitions $t \in p_c\bullet$ can be decomposed as

$$R((p_c,t),M) = \lambda_t \prod_{j=1}^{J} f_{out}^{(j)}((p_c,t),M^{(j)}),$$

where $f_{out}^{(j)}((p_c,t),(\boldsymbol{m}^{(j)},x^{(j)}))$ are continuous functions in $x^{(j)}$ and $f_{out}^{(j)}((p_c,t),(\boldsymbol{m}^{(j)},x^{(j)})) > 0$ if $p_c \notin \mathcal{P}^{(j)}$, and $f_{out}^{(j)}((p_c,t),(\boldsymbol{m}^{(j)},x^{(j)})) \geq 0$ if $p_c \in \mathcal{P}^{(j)}$ for all $x^{(j)} \in (0, B(p_c))$.

The first two assumptions and the fifth assumption also exist in [15]. The third assumption guarantees that no intermediate boundaries for the fluid flow exist. This means that the flow changes its direction or stops at the natural boundaries 0 and $B(p_c)$ or when the discrete marking changes. This implies $\hat{r}_c((\boldsymbol{m},\boldsymbol{x})(\tau^-))\hat{r}_c((\boldsymbol{m},\boldsymbol{x})(\tau^+)) > 0$ or $= 0$ for all $\boldsymbol{x}_c(\tau) \in (0, B(p_c))$. The last three assumptions are related to the superposed FSPN to enable an efficient transient analysis.

Example. (continued) For the FSPN with the three subnets, from our 5th assumption we can write

$$F(t_k,M) = \lambda_{t_k} g^{(1)}(t_k,M^{(1)}) g^{(2)}(t_k,M^{(2)}) g^{(3)}(t_k,M^{(3)}),$$

where

$$\lambda_{t_k} = \lambda_k \quad \text{and} \quad g^{(j)}(t_k,M^{(j)}) = 1 \\ \text{for } j = 1,2,3 \quad \text{and} \quad k = 1,2,3,4.$$

Regarding assumption 6, since we only have incoming arcs to continuous places in this example, for $i = 1, 2$ we can write

$$R((t_{f_i},c_i),M) = \lambda_{t_{f_i}} f_{in}^{(1)}((t_{f_i},c_i),M^{(1)}) \\ f_{in}^{(2)}((t_{f_i},c_i),M^{(2)}) f_{in}^{(3)}((t_{f_i},c_i),M^{(3)}),$$

where for the three different flow rate functions we have

1) $\lambda_{t_{f_i}} = \alpha_i$ and $f_{in}^{(j)}((t_{f_i},c_i),M^{(j)}) = 1$ for $j = 1,2,3$ and $i = 1,2$;
2) $\lambda_{t_{f_i}} = 2\alpha_i$, $f_{in}^{(1)}((t_{f_i},c_i),M^{(1)}) = 1$ for $i = 1,2$, $f_{in}^{(2)}((t_{f_1},c_1),M^{(2)}) = 1 - x_1$, $f_{in}^{(3)}((t_{f_2},c_2),M^{(3)}) = 1 - x_2$, $f_{in}^{(3)}((t_{f_2},c_2),M^{(3)}) = f_{in}^{(2)}((t_{f_1},c_1),M^{(2)}) = 1$;
3) $\lambda_{t_{f_i}} = 2\alpha_i$, $f_{in}^{(1)}((t_{f_i},c_i),M^{(1)}) = 1$ for $i = 1,2$, $f_{in}^{(2)}((t_{f_1},c_1),M^{(2)}) = 1 - x_1$, $f_{in}^{(3)}((t_{f_2},c_2),M^{(3)}) = 1 - x_2$, $f_{in}^{(3)}((t_{f_1},c_1),M^{(3)}) = 1.8x_2 + 0.1$, $f_{in}^{(2)}((t_{f_2},c_2),M^{(2)}) = 1.8x_1 + 0.1$.

■

III. Stochastic Behavior of the Net

To derive the stochastic behavior, we first define the matrices of the complete net following [16], [18]. Let

$$q_{k,l}(t,\boldsymbol{x}) = \begin{cases} F(t,(\boldsymbol{m}_k,\boldsymbol{x})) & \text{if } t \in \mathcal{E}(\boldsymbol{m}_k) \wedge \boldsymbol{m}_k \xrightarrow{t} \boldsymbol{m}_l \\ 0 & \text{otherwise} \end{cases}$$

be the transition rate between the discrete markings $\boldsymbol{m}_k$ and $\boldsymbol{m}_l$ due to transition $t \in \mathcal{T}_e$ when the fluid level is $\boldsymbol{x}$. These rates are the entries of an $(|\mathcal{M}_d| \times |\mathcal{M}_d|)$ matrix $\boldsymbol{Q}(t,\boldsymbol{x})$. For $\mathcal{U} \subset \mathcal{T}_e$, we write $\boldsymbol{Q}(\mathcal{U},\boldsymbol{x}) = \sum_{t\in\mathcal{U}} \boldsymbol{Q}(t,\boldsymbol{x})$, and for $\mathcal{U} = \mathcal{T}_e$, we write $\boldsymbol{Q}(\boldsymbol{x})$ rather than $\boldsymbol{Q}(\mathcal{T}_e,\boldsymbol{x})$. To capture diagonal elements, we define the diagonal matrix $\boldsymbol{D}$ with diagonal elements

$$\boldsymbol{D}_{k,k}(\boldsymbol{x}) = \sum_{l=1}^{|\mathcal{M}_d|} \boldsymbol{Q}_{k,l}(\boldsymbol{x}).$$

Furthermore, we define the matrix of fluid flows for continuous place $p_c \in \mathcal{P}_c$ as $\boldsymbol{R}_c(\boldsymbol{x}) = \mathrm{diag}(r_c(\boldsymbol{m}_k,\boldsymbol{x}))$ and the row vector $\boldsymbol{\pi}(\tau,\boldsymbol{x}) = (\pi_k(\tau,\boldsymbol{x}))$ for $\boldsymbol{m}_k \in \mathcal{M}_d$, $\tau \geq 0$. The vector $\boldsymbol{\pi}(\tau,\boldsymbol{x})$ contains the probability densities of finding the FSPN in discrete states $\boldsymbol{m}_k \in \mathcal{M}_d$ at time τ when the fluid level is $\boldsymbol{x}$. Initially $\boldsymbol{\pi}_k(0,\boldsymbol{x}) = 1$ for $\boldsymbol{x} = \boldsymbol{x}_0$, $k = 0$, and it is 0 otherwise. Without flush-out arcs, the evaluation of $\boldsymbol{\pi}(\tau,\boldsymbol{x})$ follows the PDEs

$$\frac{\partial \boldsymbol{\pi}(\tau,\boldsymbol{x})}{\partial \tau} + \sum_{p_c\in\mathcal{P}_c} \frac{\partial\left(\boldsymbol{\pi}(\tau,\boldsymbol{x})\boldsymbol{R}_c(\boldsymbol{x})\right)}{\partial \boldsymbol{x}_c} = \boldsymbol{\pi}(\tau,\boldsymbol{x})\left(\boldsymbol{Q}(\boldsymbol{x}) - \boldsymbol{D}(\boldsymbol{x})\right). \tag{1}$$

With flush-out arcs the PDEs become more complex, because several cases with empty and non-empty continuous places have to be considered. We use in the following a slightly different notation to introduce the PDEs than in the original papers [15], [16], which is in our opinion more intuitive. Below it will be shown that with the introduced compositional structure, flush-out arcs can be handled much easier.

With flush-out arcs, we have to consider transitions from $\mathcal{P}_c\diamond$, i.e., all transitions connected with flush-out arcs to continuous places. Let $\overline{\mathcal{P}_c\diamond} = \mathcal{T} \setminus \mathcal{P}_c\diamond$, and let $\mathcal{F}(t) = \{p \mid p \in \mathcal{P}_c,\ p \in \diamond t\}$ be the set of continuous places which are flushed out by firing transition t. We assume that the places in $\mathcal{F}(t)$ have the indices i_1^t through $i_{|\mathcal{F}(t)|}^t$. The vector $\boldsymbol{x}(\mathcal{F}(t)) = (\boldsymbol{x}_c)_{p_c\in\mathcal{F}(t)}$ is equal to the vector $\boldsymbol{x}$ restricted to the places in $\mathcal{F}(t)$. With flush-out arcs, (1) becomes

$$\frac{\partial \boldsymbol{\pi}(\tau,\boldsymbol{x})}{\partial \tau} + \sum_{p_c\in\mathcal{P}_c} \frac{\partial\left(\boldsymbol{\pi}(\tau,\boldsymbol{x})\boldsymbol{R}_c(\boldsymbol{x})\right)}{\partial \boldsymbol{x}_c} = \\ \boldsymbol{\pi}(\tau,\boldsymbol{x})\left(\boldsymbol{Q}(\overline{\mathcal{P}_c\diamond},\boldsymbol{x}) - \boldsymbol{D}(\boldsymbol{x})\right) + \sum_{t\in\mathcal{P}_c\diamond} \delta\left(\boldsymbol{x}(\mathcal{F}(t)) = \boldsymbol{0}\right) \\ \int_{y_1^t=0}^{\infty} \cdots \int_{y_{|\mathcal{F}(t)|}^t=0}^{\infty} \boldsymbol{\pi}(\tau,\boldsymbol{x} + \sum_{j=1}^{|\mathcal{F}(t)|} \mathbb{1}_{i_j^t} y_{i_j^t}) \\ \boldsymbol{Q}(t,\boldsymbol{x} + \sum_{j=1}^{|\mathcal{F}(t)|} \mathbb{1}_{i_j^t} y_{i_j^t}) dy_{i_1^t} \ldots dy_{i_{|\mathcal{F}(t)|}^t}, \tag{2}$$

where $\mathbb{1}_i$ is a row vector with 1 in position i and 0 elsewhere, and the indicator function $\delta\left(\boldsymbol{x}(\mathcal{F}(t)) = \boldsymbol{0}\right) = 1$ if all entries of $\boldsymbol{x}$ that belong to places from $\mathcal{F}(t)$ (i.e., continuous places that are flushed out by t) are zero, otherwise the function returns 0 which implies that the multiple integral is not evaluated. Observe that (2) reduces to (1) when there are no flush-out arcs connected to continuous places, and hence, $\mathcal{P}_c\diamond = \emptyset$ and $\mathcal{F}(t) = \emptyset$. No additional boundary equations are required because they are included in the definition of flow rates.

For numerical analysis, we apply a semi discretization using a finite volume technique. We discretize the spatial dimension

c (i.e., filling of fluid place p_c) in n_c intervals each of length $\delta_c = B(p_c)/n_c$. Letting $C = |\mathcal{P}_c|$, the resulting discrete state space can include up to $|\mathcal{M}_d| \prod_{c=1}^{C} n_c$ states. States are described by row vectors $(k_0, \boldsymbol{k})$ of length $C+1$ such that $\boldsymbol{k} = (k_1, \ldots, k_C)$ with $k_c = 0, \ldots, n_c - 1$ being the discretized continuous state and k_0 being the discrete state. In this discretization, the flow is defined to be constant over interval k_c of length δ_c and the discretized continuous state k_c can only change to state $k_c - 1$ or $k_c + 1$ in one transition.

Transition rates between discrete states due to firing transition t are given by

$$q_{(k_0,\boldsymbol{k}),(l_0,\boldsymbol{l})}(t) = \frac{1}{\prod_{c=1}^{C} \delta_c} \cdot \begin{cases} \int_{\kappa_1=k_1\delta_1}^{(k_1+1)\delta_1} \cdots \int_{\kappa_C=k_C\delta_C}^{(k_C+1)\delta_C} q_{k_0,l_0}(t,(\kappa_1,\ldots,\kappa_C))d\kappa_1 \ldots d\kappa_C \\ \quad \text{if}(\boldsymbol{l} = \boldsymbol{k} \wedge t \notin \mathcal{P}_c\diamond) \vee \\ \quad (\boldsymbol{l} = (\sum_{p_i \in \mathcal{P}_c \setminus \diamond t} \mathbb{1}_i) \odot \boldsymbol{k} \wedge t \in \mathcal{P}_c\diamond), \\ \mathbf{0} \quad \text{otherwise}, \end{cases} \tag{3}$$

where $\odot$ is the elementwise product of vectors. For the fluid flow of continuous place $p_c \in \mathcal{P}_c$ in discrete state $(\boldsymbol{m}, \boldsymbol{k})$, we define

$$f_c(\boldsymbol{m}, \boldsymbol{k}) = \frac{1}{\prod_{c=1}^{C} \delta_c} \int_{\xi_1=k_1\delta_1}^{(k_1+1)\delta_1} \cdots \int_{\xi_C=k_C\delta_C}^{(k_C+1)\delta_C} r_c(\boldsymbol{m}, \xi_1, \ldots, \xi_C)d\xi_1 \ldots d\xi_C. \tag{4}$$

Transitions resulting from a change of fluid level may change state $(\boldsymbol{m}_k, \boldsymbol{k})$ to $(\boldsymbol{m}_k, \boldsymbol{k} \pm \mathbb{1}_c)$ for some $p_c \in \mathcal{P}_c$ as long as the transition is allowed, i.e., resulting state belongs to the reachable state space. Thus, we define

$$q^c_{(k_0,\boldsymbol{k}),(k_0,\boldsymbol{k}+\mathbb{1}_c)} = \begin{cases} f_c(\boldsymbol{m}_{k_0}, \boldsymbol{k}) & \text{if } f_c(\boldsymbol{m}_{k_0}, \boldsymbol{k}) > 0 \wedge \boldsymbol{k}_c < n_c - 1, \\ 0 & \text{otherwise}, \end{cases}$$

$$q^c_{(k_0,\boldsymbol{k}),(k_0,\boldsymbol{k}-\mathbb{1}_c)} = \begin{cases} -f_c(\boldsymbol{m}_{k_0}, \boldsymbol{k}) & \text{if } f_c(\boldsymbol{m}_{k_0}, \boldsymbol{k}) < 0 \wedge \boldsymbol{k}_c > 0, \\ 0 & \text{otherwise}, \end{cases} \tag{5}$$

so that only the rates defined in (5) can be nonzero; all remaining rates $q^c_{(k_0,\boldsymbol{k}),(l_0,\boldsymbol{l})}$ are zero. Transition rates between states are then given by

$$q_{(k_0,\boldsymbol{k}),(l_0,\boldsymbol{l})} = q^c_{(k_0,\boldsymbol{k}),(l_0,\boldsymbol{l})} + \sum_{t \in \mathcal{T}} q_{(k_0,\boldsymbol{k}),(l_0,\boldsymbol{l})}(t). \tag{6}$$

Let $\mathcal{S}$ be the reachable state space of the discretized system; states in $\mathcal{S}$ can be ordered lexicographically, resulting in a consecutive renumbering. Then the rates computed in (6) can be collected in an $(|\mathcal{S}| \times |\mathcal{S}|)$ matrix $\boldsymbol{Q}$ with diagonal elements $\boldsymbol{Q}_{k,k} = -\sum_{l \neq k} \boldsymbol{Q}_{k,l}$, where $|\mathcal{S}| \leq |\mathcal{M}_d| \prod_{c=1}^{C} n_c$. The solution of the ordinary differential equations (ODEs)

$$\frac{d\boldsymbol{\pi}(\tau)}{d\tau} = \boldsymbol{\pi}(\tau)\boldsymbol{Q} \tag{7}$$

for time τ with initial condition $\boldsymbol{\pi}(0)$ resulting from the discretized initial state of the FSPN, is then the probability density of the FSPN at time τ. From $\boldsymbol{\pi}(\tau)$ further transient results can be derived.

In principle, an FSPN can be analyzed by constructing matrix $\boldsymbol{Q}$ and then solving the resulting ODE in (7). However, two quite general problems occur implying that this approach can realistically only be used for relatively simple and small nets with one or two continuous places:

1) The evaluation of the integrals in (3) and (4) can be cumbersome for more complex functions.
2) The curse of dimensionality implies that the reachable state space grows rapidly for an increasing number of continuous places and a finer discretization; a fine discretization is often necessary to obtain a sufficient accuracy.

For the first problem, we restrict the class of nets as done above. Especially the decomposability of the flow and rate functions allow us to separate the different dimensions for integration. To cope with very large state spaces, we present first an approach to represent matrix $\boldsymbol{Q}$ in a hierarchical form where the different submatrices can be represented as the sum of Kronecker products of small matrices. Then in the following section, we introduce an approach to avoid additionally the complete enumeration of the state space in vector $\boldsymbol{\pi}(\tau)$ by using implicit approximate representations that have been recently developed in the field of statistical physics and for the solution of higher dimensional PDEs [17].

By exploiting the compositional structure of SFSPNs, matrix $\boldsymbol{Q}$ can be represented in compact form. Let $\mathcal{S}^{(j)}$ be the projection of the global state space $\mathcal{S}$ to the places from $\mathcal{P}^{(j)}$. States of subnet j are represented by $(\boldsymbol{m}^{(j)}, x^{(j)})$ where $\boldsymbol{m}^{(j)}$ includes the marking of the discrete places from $\mathcal{P}^{(j)}$ and $x^{(j)}$ is the marking of the continuous place. For subnets without continuous or discrete places, the corresponding variables are omitted. If $\mathcal{P}^{(j)}$ contains a continuous place $p_c \in \mathcal{P}_c$, the fluid level is discretized as for the FSPN. Let $\delta^{(j)} = B(p_c)/n^{(j)}$ for $p_c \in \mathcal{P}^{(j)}$ be the discretization factor. After discretization, we obtain $(\boldsymbol{m}^{(j)}, k^{(j)})$ with $k^{(j)} = 0, \ldots, n^{(j)} - 1$. The dynamic behavior of the subnet is described by $|\mathcal{S}^{(j)}| \times |\mathcal{S}^{(j)}|$ matrices $\boldsymbol{A}^{(j)}_t$. For $t \in \mathcal{T}^{(j)}$ and $p_c \in \mathcal{P}^{(j)}_c$, we obtain

$$a^{(j)}_t((k_0,k_1),(l_0,l_1)) = \frac{1}{\delta^{(j)}} \cdot \begin{cases} \int_{\xi=k_1\delta^{(j)}}^{(k_1+1)\delta^{(j)}} g^{(j)}(t,(\boldsymbol{m}^{(j)}_{k_0},\xi))d\xi \\ \quad \text{if } \boldsymbol{m}^{(j)}_{k_0} \xrightarrow{t} \boldsymbol{m}^{(j)}_{l_0} \wedge ((k_1 = l_1 \wedge t \notin p_c\diamond) \vee \\ \qquad\qquad (l_1 = 0 \wedge t \in p_c\diamond)) \\ \int_{\xi=k_1\delta^{(j)}}^{(k_1+1)\delta^{(j)}} \left| f^{(j)}((t,c),(\boldsymbol{m}^{(j)}_{k_0},\xi)) \right| d\xi \\ \quad \text{if } l_0 = k_0 \wedge ((k_1 > 0 \wedge l_1 = k_1 - 1 \wedge \\ \qquad\qquad f^{(j)}((t,c),(\boldsymbol{m}^{(j)}_{k_0},x^{(j)}) < 0) \vee \\ \qquad\qquad (k_1 < n^{(j)} - 1 \wedge l_1 = k_1 + 1 \wedge \\ \qquad\qquad f^{(j)}((t,c),(\boldsymbol{m}^{(j)}_{k_0},x^{(j)}) > 0) \\ 0 \quad \text{otherwise}. \end{cases} \tag{8}$$

The values $a_t^{(j)}((k_0,k_1),(l_0,l_1))$ form matrix $\boldsymbol{A}_t^{(j)}$. For $t \notin \mathcal{T}^{(j)}$, we have $\boldsymbol{A}_t^{(j)} = \boldsymbol{I}$. Observe that (8) is much simpler than (3)-(6) which have to be evaluated for general FSPNs to obtain matrix $\boldsymbol{Q}$.

Example.(continued) Let us provide the entries of matrices $\boldsymbol{A}_t^{(j)}$ for $t \in \mathcal{T}$ in our running example, noting that $\boldsymbol{A}_t^{(1)} \in \mathbb{R}_{\geq 0}^{2\times 2}$, $\boldsymbol{A}_t^{(2)} \in \mathbb{R}_{\geq 0}^{n_2\times n_2}$, and $\boldsymbol{A}_t^{(3)} \in \mathbb{R}_{\geq 0}^{n_3\times n_3}$. The entries due to transitions $t_1,\ldots,t_4$ are given by

$$a_{t_1}^{(1)}((1,0),(0,1)) = g^{(1)}(t_1,(1,0)) = 1,$$
$$a_{t_1}^{(2)}(k_1,0) = \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} g^{(2)}(t_1,\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_2-1,$$
$$a_{t_1}^{(3)}(k_1,0) = \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} g^{(3)}(t_1,\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_3-1,$$

$$a_{t_2}^{(1)}((0,1),(0,1)) = g^{(1)}(t_2,(0,1)) = 1,$$
$$a_{t_2}^{(2)}(k_1,0) = \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} g^{(2)}(t_2,\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_2-1,$$
$$a_{t_2}^{(3)}(k_1,k_1) = \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} g^{(3)}(t_2,\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_3-1,$$

$$a_{t_3}^{(1)}((0,1),(0,1)) = g^{(1)}(t_3,(0,1)) = 1,$$
$$a_{t_3}^{(2)}(k_1,k_1) = \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} g^{(2)}(t_3,\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_2-1,$$
$$a_{t_3}^{(3)}(k_1,0) = \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} g^{(3)}(t_3,\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_3-1,$$

$$a_{t_4}^{(1)}((0,1),(1,0)) = g^{(1)}(t_4,(0,1)) = 1,$$
$$a_{t_4}^{(2)}(k_1,k_1) = \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} g^{(2)}(t_4,\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_2-1,$$
$$a_{t_4}^{(3)}(k_1,k_1) = \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} g^{(3)}(t_4,\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_3-1,$$

In matrix notation, we have

$$\boldsymbol{A}_{t_1}^{(1)} = \mathbb{1}_1^T\mathbb{1}_0, \quad \boldsymbol{A}_{t_1}^{(2)} = \mathbb{I}^T\mathbb{1}_0, \quad \boldsymbol{A}_{t_1}^{(3)} = \mathbb{I}^T\mathbb{1}_0,$$
$$\boldsymbol{A}_{t_2}^{(1)} = \mathbb{1}_0^T\mathbb{1}_0, \quad \boldsymbol{A}_{t_2}^{(2)} = \mathbb{I}^T\mathbb{1}_0, \quad \boldsymbol{A}_{t_2}^{(3)} = I,$$
$$\boldsymbol{A}_{t_3}^{(1)} = \mathbb{1}_0^T\mathbb{1}_0, \quad \boldsymbol{A}_{t_3}^{(2)} = I, \quad \boldsymbol{A}_{t_3}^{(3)} = \mathbb{I}^T\mathbb{1}_0,$$
$$\boldsymbol{A}_{t_4}^{(1)} = \mathbb{1}_0^T\mathbb{1}_1, \quad \boldsymbol{A}_{t_4}^{(2)} = I, \quad \boldsymbol{A}_{t_4}^{(3)} = I.$$

The entries due to transitions t_{f_1} and t_{f_2} are given by

$$a_{t_{f_i}}^{(1)}((p_1,p_2),(p_1,p_2)) = f_{in}^{(1)}(t_{f_i},(p_1,p_2)) = 1 \quad \text{for } i = 1,2,$$

and for the three different flow rate functions

1)
$$a_{t_{f_1}}^{(2)}(k_1,k_1+1) = \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} f_{in}^{(2)}(t_{f_1},\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_2-2,$$
$$a_{t_{f_1}}^{(3)}(k_1,k_1) = \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} f_{in}^{(3)}(t_{f_1},\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_3-1,$$
$$a_{t_{f_2}}^{(2)}(k_1,k_1) = \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} f_{in}^{(2)}(t_{f_2},\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_2-1,$$
$$a_{t_{f_2}}^{(3)}(k_1,k_1+1) = \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} f_{in}^{(3)}(t_{f_2},\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_3-2,$$

2)
$$\begin{aligned} a_{t_{f_1}}^{(2)}(k_1,k_1+1) &= \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} f_{in}^{(2)}(t_{f_1},\xi)d\xi \\ &= \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} (1-\xi)d\xi \\ &= 1 - k_1\delta^{(2)} - \delta^{(2)}/2 \quad \text{for } k_1 = 0,\ldots,n_2-2, \\ a_{t_{f_1}}^{(3)}(k_1,k_1) &= \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} f_{in}^{(3)}(t_{f_1},\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_3-1, \\ a_{t_{f_2}}^{(2)}(k_1,k_1) &= \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} f_{in}^{(2)}(t_{f_2},\xi)d\xi = 1 \quad \text{for } k_1 = 0,\ldots,n_2-1, \\ a_{t_{f_2}}^{(3)}(k_1,k_1+1) &= \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} f_{in}^{(3)}(t_{f_2},\xi)d\xi \\ &= \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} (1-\xi)d\xi \\ &= 1 - k_1\delta^{(3)} - \delta^{(3)}/2 \quad \text{for } k_1 = 0,\ldots,n_3-2, \end{aligned}$$

3)
$$\begin{aligned} a_{t_{f_1}}^{(2)}(k_1,k_1+1) &= \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} f_{in}^{(2)}(t_{f_1},\xi)d\xi \\ &= \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} (1-\xi)d\xi \\ &= 1 - k_1\delta^{(2)} - \delta^{(2)}/2 \quad \text{for } k_1 = 0,\ldots,n_2-2, \\ a_{t_{f_1}}^{(3)}(k_1,k_1) &= \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} f_{in}^{(3)}(t_{f_1},\xi)d\xi \\ &= \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} (1.8\xi + 0.1)d\xi \\ &= 1.8(2k_1+1)\delta^{(3)}/2 + 0.1 \quad \text{for } k_1 = 0,\ldots,n_3-1, \end{aligned}$$

$$
\begin{aligned}
a^{(2)}_{t_{f_2}}(k_1,k_1) &= \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} f^{(2)}_{in}(t_{f_2},\xi)d\xi \\
&= \tfrac{1}{\delta^{(2)}} \int_{k_1\delta^{(2)}}^{(k_1+1)\delta^{(2)}} (1.8\xi+0.1)d\xi \\
&= 1.8(2k_1+1)\delta^{(2)}/2 + 0.1 \\
&\quad \text{for } k_1 = 0,\ldots,n_2-1, \\
a^{(3)}_{t_{f_2}}(k_1,k_1+1) &= \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} f^{(3)}_{in}(t_{f_2},\xi)d\xi \\
&= \tfrac{1}{\delta^{(3)}} \int_{k_1\delta^{(3)}}^{(k_1+1)\delta^{(3)}} (1-\xi)d\xi \\
&= 1 - k_1\delta^{(3)} - \delta^{(3)}/2 \\
&\quad \text{for } k_1 = 0,\ldots,n_3-2.
\end{aligned}
$$

In matrix notation, we have

$$\boldsymbol{A}^{(1)}_{t_{f_i}} = I \quad \text{for } i = 1,2,$$

and for the three different flow rate functions we obtain for function 1:

$$
\begin{aligned}
\boldsymbol{A}^{(2)}_{t_{f_1}} &= \text{supdiag}(\mathbb{1}^T), & \boldsymbol{A}^{(3)}_{t_{f_1}} &= I, \\
\boldsymbol{A}^{(2)}_{t_{f_2}} &= I, & \boldsymbol{A}^{(3)}_{t_{f_2}} &= \text{supdiag}(\mathbb{1}^T),
\end{aligned}
$$

for function 2:

$$
\begin{aligned}
&\boldsymbol{A}^{(2)}_{t_{f_1}} = \text{supdiag}((1-\tfrac{\delta^{(2)}}{2},\ldots,1-(n_2-2)\delta^{(2)}-\tfrac{\delta^{(2)}}{2})^T), \\
&\boldsymbol{A}^{(3)}_{t_{f_1}} = I, \quad \boldsymbol{A}^{(2)}_{t_{f_2}} = I, \\
&\boldsymbol{A}^{(3)}_{t_{f_2}} = \text{supdiag}((1-\tfrac{\delta^{(3)}}{2},\ldots,1-(n_3-2)\delta^{(3)}-\tfrac{\delta^{(3)}}{2})^T),
\end{aligned}
$$

for function 3:

$$
\begin{aligned}
&\boldsymbol{A}^{(2)}_{t_{f_1}} = \text{supdiag}((1-\tfrac{\delta^{(2)}}{2},\ldots,1-(n_2-2)\delta^{(2)}-\tfrac{\delta^{(2)}}{2})^T), \\
&\boldsymbol{A}^{(3)}_{t_{f_1}} = \text{supdiag}((\tfrac{1.8\delta^{(3)}}{2}+0.1,\ldots,\tfrac{1.8(2n_3-1)\delta^{(3)}}{2}+0.1)^T), \\
&\boldsymbol{A}^{(2)}_{t_{f_2}} = \text{supdiag}((\tfrac{1.8\delta^{(2)}}{2}+0.1,\ldots,\tfrac{1.8(2n_2-1)\delta^{(2)}}{2}+0.1)^T), \\
&\boldsymbol{A}^{(3)}_{t_{f_2}} = \text{supdiag}((1-\tfrac{\delta^{(3)}}{2},\ldots,1-(n_3-2)\delta^{(3)}-\tfrac{\delta^{(3)}}{2})^T),
\end{aligned}
$$

where supdiag$(\cdot)$ denotes a matrix with nonzero entries in its superdiagonal given by its vector argument. ■

We now introduce how $\boldsymbol{Q}$ can be expressed in terms of the matrices $\boldsymbol{A}^{(j)}_t$. It is well known that the relation $\mathcal{S} \subseteq \times_{j=1}^J \mathcal{S}^{(j)}$ holds. Unfortunately, the potential state space $\times_{j=1}^J \mathcal{S}^{(j)}$ is often a superset of the reachable state space $\mathcal{S}$ [4]. In this case, an additional hierarchy is introduced to compose subsets of the local states. Different approaches to generate such a hierarchy are nowadays well known [3], [9], [10], [22] and will not be repeated here.

We briefly introduce the hierarchical structure. Each subnet state space is decomposed into disjoint subsets $\mathcal{S}^{(j)}[k]$ for $k = 0,\ldots,m^{(j)}-1$. The macro state space is then defined as $\mathcal{MS} \subset \times_{j=1}^J \{0,\ldots,m^{(j)}-1\}$ such that

$$\mathcal{S} = \bigcup_{\boldsymbol{k}\in\mathcal{MS}} \mathcal{S}[\boldsymbol{k}] = \bigcup_{(k_1,\ldots,k_J)\in\mathcal{MS}} \times_{j=1}^J \mathcal{S}^{(j)}[k_j]. \quad (9)$$

According to the decomposition of the state space, each matrix $\boldsymbol{A}^{(j)}_t$ can be decomposed into $(m^{(j)})^2$ submatrices $\boldsymbol{A}^{(j)}[k,l]$ for $k,l = 0,\ldots,m^{(j)}-1$. Furthermore, we define diagonal matrices $\boldsymbol{D}^{(j)}_t = \text{diag}\left(\boldsymbol{A}^{(j)}_t \mathbb{1}^T\right)$. Matrix $\boldsymbol{Q}$ can then be represented in a block structured form with $|\mathcal{MS}|^2$ blocks $\boldsymbol{Q}[\boldsymbol{k},\boldsymbol{l}]$ $(\boldsymbol{k},\boldsymbol{l} \in \mathcal{MS})$ such that

$$
\boldsymbol{Q}[\boldsymbol{k},\boldsymbol{l}] = \begin{cases} \sum_{t\in\mathcal{T}} \lambda_t \bigotimes_{j=1}^J \boldsymbol{A}^{(j)}_t[k_j,l_j] & \text{for } \boldsymbol{k} \neq \boldsymbol{l}, \\ \sum_{t\in\mathcal{T}} \lambda_t \bigotimes_{j=1}^J \left(\boldsymbol{A}^{(j)}_t[k_j,k_j] - \boldsymbol{D}^{(j)}_t[k_j,k_j]\right) & \text{for } \boldsymbol{k} = \boldsymbol{l}. \end{cases} \quad (10)
$$

This representation is compact because a matrix of dimension $|\mathcal{S}|$ which is often in $O(\prod_{j=1}^J |\mathcal{S}^{(j)}|)$ is represented by small matrices of order $|\mathcal{S}^{(j)}|$. The representation is particularly well suited for iterative numerical methods since the vector matrix products with matrix $\boldsymbol{Q}$ can be realized efficiently [9], [25].

For transient analysis, which is considered here, the uniformization method is often applied, resulting in

$$\boldsymbol{\pi}(\tau) = \boldsymbol{\pi}(0) \sum_{k=0}^{\infty} Poiss(q_{\max}\tau,k) \left(\boldsymbol{I} + \frac{1}{q_{\max}}\boldsymbol{Q}\right)^k, \quad (11)$$

where $Poiss$ are the Poisson probabilities and $q_{\max} \geq \max\left(|\boldsymbol{Q}_{s,s}|\right)$. Other transient solvers like Runge-Kutta Formulae (RKF) or Backward Differentiation Formulae (BDF) can be realized similarly. For details we refer to [25].

Although matrix $\boldsymbol{Q}$ is represented in compact form helping us to circumvent the curse of dimensionality to a large extent, numerical analysis as in (11) requires vectors of length $|\mathcal{S}|$, which still limits the applicability of semi discretization. In the following section, we present a more compact approach to represent vectors which requires, in contrast to the exact compact representation of matrix $\boldsymbol{Q}$, the introduction of an approximation.

Example.(continued) For the simple example, a hierarchical structure is not necessary because the reachable state space is identical to the potential state space consisting of the cross product of the state spaces of the three components. Thus the discretized system has $2n_2n_3$ states and the transition matrix can be represented by sums of Kronecker products of three matrices with the dimensions 2×2, $n_2 \times n_2$ and $n_3 \times n_3$, respectively. ■

IV. Transient Numerical Analysis

The problem of combinatorially growing state spaces for high dimensional problems is, of course, not restricted to FSPNs. It is known in many areas where PDEs are used, examples are biological systems or statistical physics. In these areas compact representations to approximate higher dimensional tensors in a more efficient way have been developed [17], [23]. These representations have recently also been applied to the stationary analysis of Markov models [5], [19]. We extend the approaches here to the transient analysis of SFSPNs.

In the following, we consider the representation of a vector $\boldsymbol{x}$ including one entry for each state from $\mathcal{S}[\boldsymbol{k}]$. This vector

is multiplied with submatrices $\boldsymbol{Q}[\boldsymbol{k},\boldsymbol{l}]$ in numerical analysis algorithms. To represent all states, $|\mathcal{MS}|$ vectors are necessary.

The simplest form of a compact representation for $\boldsymbol{x}$ using data structures with $O(\max_{j\in\{1,\ldots,J\}}|\mathcal{S}^{(j)}[k_j]|)$ entries is the following representation as a Kronecker product [17], [25]

$$\boldsymbol{x} \approx \bigotimes_{j=1}^{J} \boldsymbol{x}^{(j)}, \tag{12}$$

where vectors $\boldsymbol{x}^{(j)}$ are of length $|\mathcal{S}^{(j)}[k_j]|$. Since matrix $\boldsymbol{Q}[\boldsymbol{k},\boldsymbol{l}] = \sum_t \otimes_{j=1}^{J} \boldsymbol{C}_t^{(j)}$, the multiplication can be realized as follows.

$$\left(\bigotimes_{j=1}^{J} \boldsymbol{x}^{(j)}\right)\left(\sum_t \bigotimes_{j=1}^{J} \boldsymbol{C}_t^{(j)}\right) = \sum_t \bigotimes_{j=1}^{J} \boldsymbol{x}^{(j)}\boldsymbol{C}_t^{(j)} \tag{13}$$

Consequently, the result is a sum of Kronecker products of vectors, one for each term in the sum. This implies that for an exact computation, in each iteration of the transient solution algorithm the number of Kronecker products is increased by a factor of $O(|\mathcal{T}|)$ which means that after some iterations the representation is no longer compact. A straightforward idea is to approximate a sum of Kronecker products with a single Kronecker product as done in [6]. However, although this approach resulted in some cases in good results, the approximation error is uncontrolled and the representation is fixed. More appropriate is a flexible representation which allows one to control the approximation error.

Such a representation is available with the HTD format [17], [21] which will be briefly introduced here. The idea is to represent vector $\boldsymbol{x}$ as a binary tree, as shown for the case $J = 4$ in Fig. 2. Matrix $\boldsymbol{U}^{(j)}$ is a $|\mathcal{S}^{(j)}[k_j]| \times r^{(j)}$ matrix, the intermediate matrices $\boldsymbol{B}^{(ij)}$ are of dimensions $r^{(i)}r^{(j)} \times r^{(ij)}$ for $(i,j) \in \{(1,2),(3,4)\}$ and the root matrix $\boldsymbol{B}^{(1234)}$ is a $r^{(12)}r^{(34)} \times 1$ matrix (a vector). This representation can be easily extended to more than 4 subnets. If the number of subnets is not a power of 2, still one node per subnet has to be at the bottom level.

Vector $\boldsymbol{x}$ with the HTD representation from Fig. 2 can then be represented as

$$\boldsymbol{x}^T = \left(\boldsymbol{U}^{(1)} \otimes \boldsymbol{U}^{(2)} \otimes \boldsymbol{U}^{(3)} \otimes \boldsymbol{U}^{(4)}\right)\left(\boldsymbol{B}^{(12)} \otimes \boldsymbol{B}^{(34)}\right)\boldsymbol{B}^{(1234)}.$$

Memory requirements for such a representation with J subnets are bounded by $Jn_{\max}r_{\max} + (J-2)r_{\max}^3 + r_{\max}^2$ floating point numbers, where $n_{\max} = \max\left(|\mathcal{S}^{(j)}[k_j]|\right)$ and $r_{\max} = \max(r^{(i)})$ for some node i in the tree. Values $r^{(i)}$ denote ranks, and as long as the ranks are not too large, the representation remains compact.

The interesting aspect is that we can compute the product of a Kronecker product of matrices with a vector in HTD form in a very efficient way, mainly by computing products $\left(\boldsymbol{U}^{(j)}\right)^T \boldsymbol{C}^{(j)}$. For details we refer to the corresponding algorithm in [21]. If two vectors in HTD form are added, then the rank of the resulting representation equals the sum of the ranks of the two representations that are added. Thus, the same problem as for simple Kronecker products comes up. However, for the HTD representation a well defined approximation algorithm exists. This algorithm computes for each matrix in the tree a singular value decomposition which allows one to truncate the ranks according to a well defined error bound (for details see [5], [21]). The local truncation operation in each node results in an optimal local approximation of the exact representation in terms of the 2-norm [14].

Algorithms to add vectors in HTD format and to perform the truncation can be found in [21]. By setting a truncation bound, a trade off between accuracy and memory requirements is achieved. For the transient analysis of FSPNs, three approximations are relevant; namely the discretization interval length of the spatial dimensions, the discretization interval for the time step, and the truncation bounds. If a differential equation solver like RKF or BDF is used, the time step is set adaptively (for details see [25]). For uniformization, Poisson probabilities are truncated instead which can be done a priori (for details see [25]). The other values are set manually.

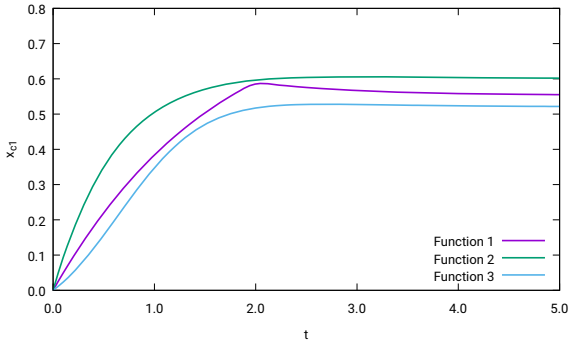

Fig. 3. Filling of place c_1 with the different flow functions.

Example.(continued) The simple example is analyzed for $n_1 = n_2 = 1000$ which implies that the resulting discrete model has 2 million states. This value is sufficient since a further increase of n_i changes the results only marginally. A coarser discretization using $n_1 = n_2 = 100$ is not sufficient in particular for the third flow function which introduces a dependency between the filling of both fluid places. The relative difference in the time dependent filling of place c_1 (and also c_2) between the discretization with $n_i = 100$ and $n_i = 1000$ goes up to 22%. This shows that a fine discretization is important but results in huge matrices and vectors. Fig. 3 shows the time dependent filling of place c_1 under the different flow functions. The results have been computed with uniformization using the (exact) Kronecker representation of the matrices, with a flat vector and a truncation bound of 10^{-6} for the Poisson probabilities.

We analyze the small example with two variants of uniformization and two versions of RKF. For the first version of uniformization (*unif*) we set the truncation bound for the Poisson probabilities to 10^{-3} and the truncation bound of the HTD structure for the vector to 10^{-6}. In the second version (*unif2*) we choose truncation error 10^{-3} for the Poisson probabilities and truncation bound 10^{-7} for the HTD structure.

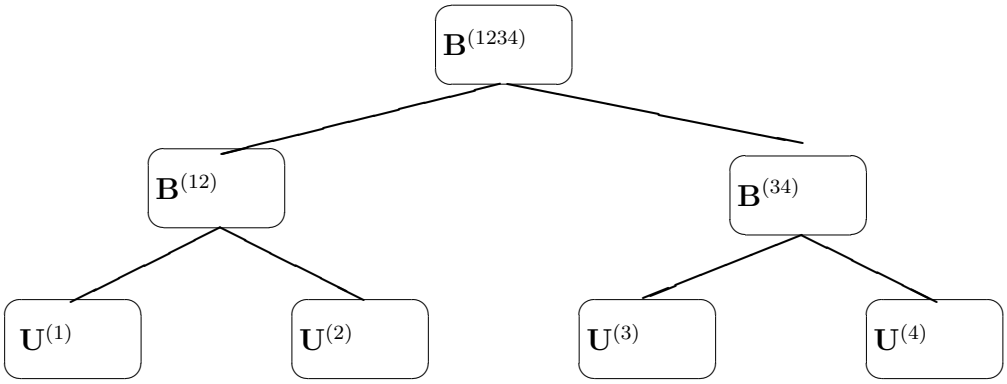

Fig. 2. HTD representation

	Func. 1		Func. 2		Func. 3	
Method	mean	max	mean	max	mean	max
unif	2.9%	9.9%	3.2%	9.7%	3.0%	9.4%
unif2	0.1%	0.5%	0.1%	0.3%	0.2%	0.6%
rk	2.8%	9.4%	1.0%	5.0%	1.8%	4.8%
rk2	0.2%	0.8%	0.0%	0.0%	0.0%	0.1%

TABLE I
MEAN AND MAXIMAL RELATIVE ERRORS OF THE METHODS WITH HTD VECTOR REPRESENTATION.

For RKF we use the explicit variant RKDP4(5) and choose the local error bound and HTD truncation error equal to 10^{-4} for the first variant (*RK*) and local error and HTD truncation error 10^{-5} for the second variant (*RK2*). As a rule of thumb one should choose similar values for the local error bound and the truncation error in RKF, whereas in uniformization, where a global truncation error is defined for the Poisson probabilities, the truncation bound for the HTD structure should be smaller than the truncation bound for the Poisson probabilities divided by $\alpha \in [0.01, 0.1]$ times the number of iterations necessary to reach the Poisson truncation bound, which can be computed a priori from the Poisson probabilities. Recall that the truncation bound for the HTD structure translates to a truncation error measured in the 2-norm of the vector approximation.

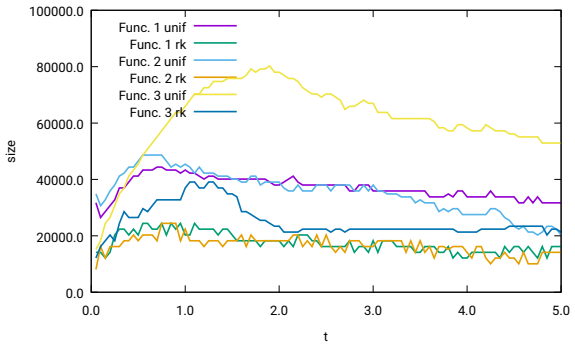

Fig. 4. Memory requirements for the methods *unif* and *rk*.

With the four different methods we analyze the three versions of the example for the interval $[0, 5]$ and compute the filling of the fluid places at 100 equidistant time points. Tab. I includes the mean and maximal difference of the relative error between the approximate solution with the HTD vector representation and the exact solution with the flat vector. It can be seen that for the variants *unif* and *rk* the relative errors are all below 10%, whereas the other two variants yield very good approximations. All results are much better than the results one would obtain from a coarse discretization with only 100 intervals. Memory requirements for the HTD vector representation are shown in Fig. 4 for *unif* and *rk* and in Fig. 5 for *unif2* and *rk2*. The first two solution approaches with the larger truncation bounds for the HTD structure have very low memory requirements. RKF requires only about 20000 floating point values to represent the vector with 2 million entries for the first two flow functions. The third flow function is more complex such that the vector representation needs slightly more memory. Uniformization with smaller HTD truncation bound requires slightly more memory. In general the memory requirements are in the range of the requirement of a flat vector representation for $n_i = 100$. The methods *unif2* and *rk2* need slightly more memory but it can be seen that for this small example memory requirements are reduced by more than an order of a magnitude even for these methods that compute results with an accuracy that is sufficient for most applications. ■

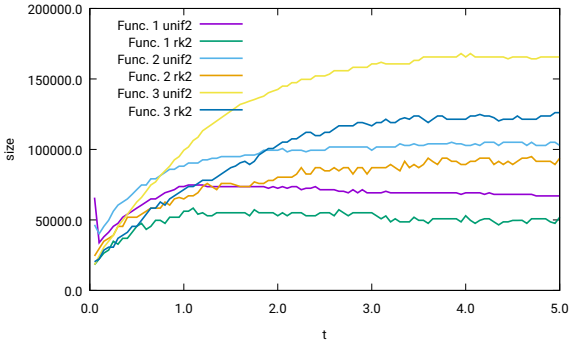

Fig. 5. Memory requirements for the methods *unif2* and *rk2*.

V. NUMERICAL RESULTS

We also consider a relatively more complicated model from [1]. It is that of the degrading software system with rejuvenation, restart, and replication illustrated in Fig. 6. In

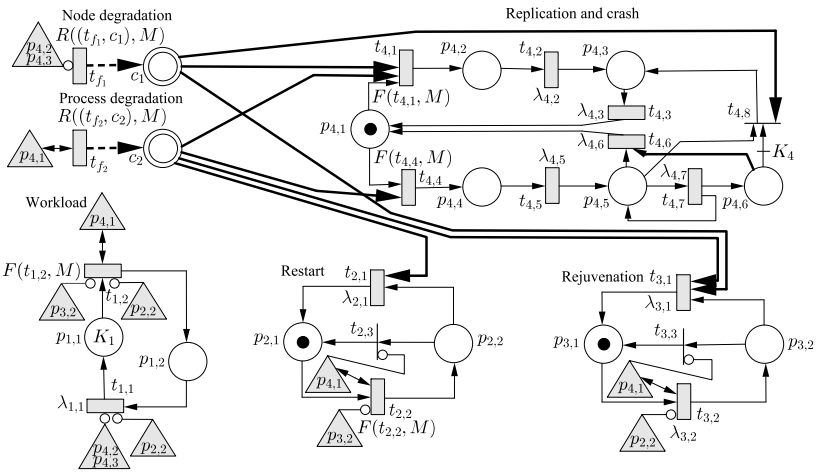

Fig. 6. FSPN model of a degrading software system.

addition to the features of our running example, this model has immediate transitions, and an arc multiplicity of $K_4 > 0$. The semantics of the triangles which are connected to transitions is as follows. If a triangle is connected to a bidirectional arc having two arrows, then the transition is enabled if each place in the triangle contains at least one token. If it is connected by an inhibitor arc (that is, an arc with a small circle), then the transition is enabled if none of the places in the triangle contains a token. Firing of a transition does not modify the marking of the places in connected triangles. The FPSN has twelve discrete and two continuous places. The set of immediate transitions is given by $\mathcal{T}_i = \{t_{2,3}, t_{3,3}, t_{4,8}\}$. Upon firing, transitions $t_{3,1}$ and $t_{4,1}$ flush out both continuous places, transitions $t_{2,1}$, $t_{4,4}$, and $t_{4,8}$ flush out continuous place c_2, and transition $t_{4,6}$ flushes out the discrete place $p_{4,6}$.

We assume that the fluid upper bounds of the continuous places c_1 and c_2 are normalized to 1. As in our running example, we assume a firing rate of 1 for the transitions t_{f_1} and t_{f_2}. However, contrary to the running example, these transitions can be disabled due to the inhibitor arcs, yet the firing still does not modify the discrete marking. Transitions $t_{1,2}$, $t_{2,2}$, $t_{4,1}$, and $t_{4,4}$ have marking dependent firing rates

$$\begin{aligned} F(t_{1,2}, M) &= 1/(m_{p_{1,1}}(x_1+1)(x_2+1)), \\ F(t_{2,2}, M) &= \begin{cases} 100 & \text{if } x_2 > 0.75 \\ 0.0001 & \text{otherwise} \end{cases}, \\ F(t_{4,1}, M) &= m_{p_{1,1}}(8x_1+1)/2400, \\ F(t_{4,4}, M) &= m_{p_{1,1}}(8x_2+1)/1920. \end{aligned}$$

The remaining firing rates given by

$$\lambda_{1,1} = \lambda_{3,1} = 0.2, \quad \lambda_{2,1} = \lambda_{4,2} = 1, \quad \lambda_{3,2} = 1/360,$$

$$\lambda_{4,3} = 0.1, \quad \lambda_{4,5} = 2, \quad \lambda_{4,6} = 0.32, \quad \lambda_{4,7} = 0.08$$

are constant. Transitions t_{f_1} and t_{f_2} fill the adjacent continuous places c_1 and c_2 respectively with flow rate functions

$$\begin{aligned} R((t_{f_1}, c_1), M) &= m_{p_{1,1}}/128 \quad \text{and} \\ R((t_{f_2}, c_1), M) &= m_{p_{1,1}}(x_1+1)/96. \end{aligned}$$

We use the following partition of the places

$$\begin{aligned} &\mathcal{P}^{(1)} = \{p_{1,1}, p_{2,1}, p_{2,2}, p_{3,1}, p_{3,2}, p_{4,1}, \ldots, p_{4,6}\}, \\ &\mathcal{P}^{(2)} = \{c_1\}, \quad \mathcal{P}^{(3)} = \{c_2\} \end{aligned}$$

resulting in

$$\begin{aligned} &\mathcal{T}^{(1)} = \mathcal{T}, \quad \mathcal{T}^{(2)} = \{t_{3,1}, t_{4,1}, t_{4,8}, t_{f_1}\}, \\ &\mathcal{T}^{(3)} = \{t_{2,1}, t_{3,1}, t_{4,1}, t_{4,4}, t_{f_2}\}. \end{aligned}$$

Thus, we have one subnet with twelve discrete places and two subnets with one continuous place each.

From our 5th assumption, for transitions

$$t_{1,1}, t_{2,1}, t_{3,1}, t_{3,2}, t_{4,2}, t_{4,3}, t_{4,5}, t_{4,6}, t_{4,7}$$

with constant firing rates, we have

$$F(t_k, M) = \lambda_{t_k} g^{(1)}(t_k, M^{(1)}) g^{(2)}(t_k, M^{(2)}) g^{(3)}(t_k, M^{(3)}),$$

where

$$\lambda_{t_k} = \lambda_k \quad \text{and} \quad g^{(j)}(t_k, M^{(j)}) = 1 \quad \text{for } j = 1, 2, 3.$$

For the marking dependent firing rates, we have

$$\begin{aligned} \lambda_{1,2} &= 1, \\ g^{(1)}(t_{1,2}, M^{(1)}) &= 1/m_{p_{1,1}}, \\ g^{(2)}(t_{1,2}, M^{(2)}) &= 1/(x_1+0.5), \\ g^{(3)}(t_{1,2}, M^{(3)}) &= 1/(x_2+0.5), \\ \lambda_{2,2} &= 1, \\ g^{(1)}(t_{2,2}, M^{(1)}) &= 1, \\ g^{(2)}(t_{2,2}, M^{(2)}) &= 1, \\ g^{(3)}(t_{2,2}, M^{(3)}) &= \begin{cases} 100 & \text{if } x_2 > 0.75 \\ 0.0001 & \text{otherwise} \end{cases}, \\ \lambda_{4,1} &= 1/2400, \\ g^{(1)}(t_{4,1}, M^{(1)}) &= m_{p_{1,1}}, \\ g^{(2)}(t_{4,1}, M^{(2)}) &= 8x_1+1, \end{aligned}$$

$$
\begin{aligned}
g^{(3)}(t_{4,1}, M^{(3)}) &= 1, \\
\lambda_{4,4} &= 1/1920, \\
g^{(1)}(t_{4,4}, M^{(1)}) &= m_{p_{1,1}}, \\
g^{(2)}(t_{4,4}, M^{(2)}) &= 1, \\
g^{(3)}(t_{4,4}, M^{(3)}) &= 8x_2 + 1.
\end{aligned}
$$

Regarding assumption 6, for $i = 1, 2$ we can write

$$
\begin{aligned}
R((t_{f_i}, c_i), M) &= \lambda_{t_{f_i}} f_{in}^{(1)}((t_{f_i}, c_i), M^{(1)}) \\
&\quad f_{in}^{(2)}((t_{f_i}, c_i), M^{(2)}) f_{in}^{(3)}((t_{f_i}, c_i), M^{(3)}),
\end{aligned}
$$

where

$$
\begin{aligned}
\lambda_{t_{f_1}} &= 1/128, \\
f_{in}^{(1)}((t_{f_1}, c_1), M^{(1)}) &= m_{p_{1,1}}, \quad \text{and} \\
f_{in}^{(j)}((t_{f_1}, c_1), M^{(j)}) &= 1 \quad \text{for } j = 2, 3, \\
\lambda_{t_{f_2}} &= 1/96, \\
f_{in}^{(1)}((t_{f_2}, c_2), M^{(1)}) &= m_{p_{1,1}}, \\
f_{in}^{(2)}((t_{f_2}, c_2), M^{(2)}) &= x_1 + 1, \quad \text{and} \\
f_{in}^{(3)}((t_{f_2}, c_2), M^{(3)}) &= 1.
\end{aligned}
$$

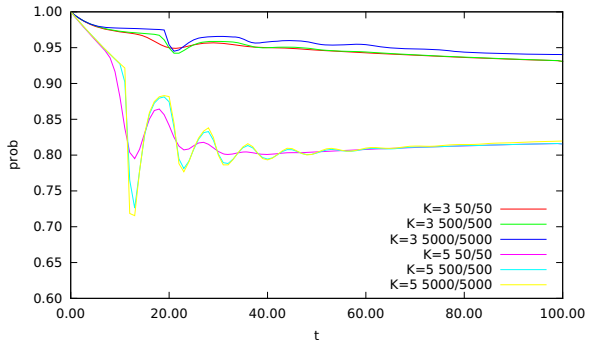

Fig. 7. Probability of a normal operation of the system.

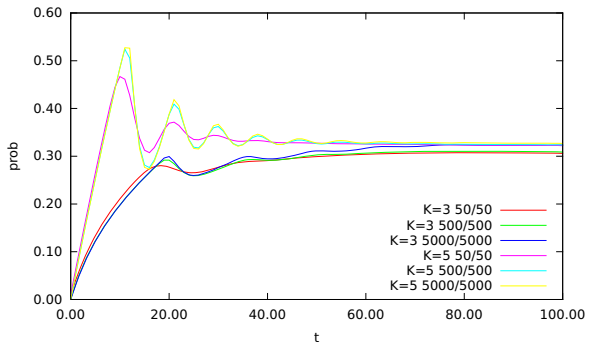

Fig. 8. Process degradation.

Although the example has again two fluid places, it differs significantly from the simple running example. Due to the flush-out arcs and state dependent flow and firing rates, the reachable states space is a proper subset of the potential state space and a hierarchical structure has to be built. We analyze two configurations with $K_1 = K_4 = 3$ and $K_1 = K_4 = 5$. In both cases, 7 macro states are necessary to represent the reachable state space in compositional form. For the discretization of the fluid places we choose $n_i \in \{50, 500, 5000\}$ $(i = 1, 2)$. The example is analyzed in the interval $[0, 100]$ starting with the discrete marking shown in Fig. 6 and empty fluid places. For the analysis uniformization is applied, Poisson probabilities are truncated with bound 0.001. Experiments are performed on a PC with $16GB$ main memory, Intel i7 3.6Ghz CPU with 8 cores. However, the algorithms use only one core currently.

	Configuration			
Measure	50/50	500/500	5000/500	5000/5000
states	30,606	3,006,006	30,060,006	300,060,006
non-zeros	184,250	18,267,500	182,715,500	1,825,175,000
nz struct	756	6,606	33,606	65,106
nz htd 0.1	1849	35,963	216,287	868,097
nz htd 0.01	3,816	60,389	349,552	980,453
time flat	0.00	0.47	–	–
time struct	0.01	0.35	4.10	45.63
time htd 0.1	0.01	0.16	0.93	11.53
time htd 0.01	0.02	0.37	2.20	24.62

TABLE II
RESULTS FOR THE EXAMPLE WITH $K_1 = K_4 = 3$.

	Configuration			
Measure	50/50	500/500	5000/500	5000/5000
states	46,510	4,515,010	45,150,010	450,150,010
non-zeros	297,940	29,304,490	293,1124,90	2,925,544,990
nz struct	866	6,716	33,716	65,216
nz htd 0.1	5,122	45,166	225,064	908,898
nz htd 0.01	9,011	70,125	388,080	1,046,009
time flat	0.00		–	–
time struct	0.01	0.66	7.25	128.5
time htd 0.1	0.03	0.23	0.85	11.01
time htd 0.01	0.05	0.55	2.63	16.10

TABLE III
RESULTS FOR THE EXAMPLE WITH $K_1 = K_4 = 5$.

Some results for the two configurations and different number of discretization intervals are shown in Figs. 7 and 8. In the first figure the mean probability of a normal operation is shown. It can be seen that the curves have some peaks in the beginning which can only be analyzed with a fine discretization. Results for the process degradation, which corresponds to the fluid level in place c_2, are similar. Tables II and III summarize additional facts about the solution effort. The second line of each table contains the number of discretization intervals for places c_1 and c_2, respectively. We analyze four different configurations. The line starting with *states* contains the number of states in the discrete model. The line starting with *non-zeros* contains the number of non-zero elements in matrix $\boldsymbol{Q}$, which corresponds to the number of transitions plus the number of diagonal elements. *nz struct* equals the number of nonzero elements in the matrices of the subnets which can be used to build the complete matrix using (10). The following two lines, starting with *nz htd 0.1* and *nz htd 0.01* show the number of floating point numbers which are stored in the HTD structure when the example is analyzed with uniformization and a maximal error in the transient results, which is less than 10% or 1%. This requires truncation bounds for the HTD

structures between 10^{-6} and $5 \cdot 10^{-8}$. Truncation bounds for the largest configurations are only estimates because we were not able to compute the exact results in an acceptable time. The last four lines of the tables include times to perform one iteration, which means to compute one vector matrix product, with different matrix and vector representations. Time is measured in seconds of CPU time. For the final solution, between 10000 and 94100 iterations are necessary. Unfortunately, the system becomes more stiff if the number of intervals for discretization is increased and a stiff system requires more iterations. This means that for the largest models the largest number of iterations has to be performed to reach the required accuracy.

It can be noticed that, as for the simple running example, the HTD structure remains very compact. This has the effect that vector matrix products can be computed faster than with a flat vector and even with a sparse matrix implementation. However, this can only be observed for large state spaces. For small matrices the overhead of the HTD computations dominates. Further results, including results about the performance of different transient measures will be made available in a separate report.

VI. Conclusion

The paper presents a compositional class of FSPNs and a new numerical approach for their transient analysis. The approach is tailored towards the analysis of very large state spaces. It uses a compact, but still exact, representation of the transition matrix and a compact, but approximate, representation of the solution vector. In this way, state spaces of size that does not allow the representation of each state in computer memory can be analyzed numerically with an often acceptable accuracy.

The presented model can be extended in various directions. It is possible to consider second order fluid models in a similar way [7], [26]. Furthermore, unbounded fluid places can be integrated by using an exponential transformation. To improve the solution time, the inherent parallelism in the different steps of the approach needs to be exploited.

References

[1] Andrea Bobbio, Sachin Garg, Marco Gribaudo, András Horváth, Matteo Sereno, and Miklós Telek. Compositional fluid stochastic Petri net model for operational software system performance. In *IEEE International Conference on Software Reliability Engineering Workshops, ISSRE Workshops 2008, Seattle, WA, USA, November 11-14, 2008*, pages 1–6. IEEE, 2008.

[2] Luca Bortolussi, Jane Hillston, Diego Latella, and Mieke Massink. Continuous approximation of collective system behaviour: A tutorial. *Perform. Eval.*, 70(5):317–349, 2013.

[3] Peter Buchholz. Hierarchical structuring of superposed GSPNs. *IEEE Trans. Softw. Eng.*, 25:166–181, 1999.

[4] Peter Buchholz, Gianfranco Ciardo, Susanna Donatelli, and Peter Kemper. Complexity of memory-efficient Kronecker operations with applications to the solution of Markov models. *INFORMS J. Comput.*, 12:203–222, 2000.

[5] Peter Buchholz, Tugrul Dayar, Jan Kriege, and M. Can Orhan. On compact solution vectors in Kronecker-based Markovian analysis. *Perform. Eval.*, 115:132–149, 2017.

[6] Peter Buchholz and William H. Sanders. Approximate computation of transient results for large Markov chains. In *1st International Conference on Quantitative Evaluation of Systems (QEST 2004), 27-30 September 2004, Enschede, The Netherlands*, pages 126–135. IEEE Computer Society, 2004.

[7] Dongyan Chen, Yiguang Hong, and Kishor S. Trivedi. Second-order stochastic fluid models with fluid-dependent flow rates. *Perform. Eval.*, 49(1/4):341–358, 2002.

[8] Gianfranco Ciardo, David M. Nicol, and Kishor S. Trivedi. Discrete-event simulation of fluid stochastic Petri nets. *IEEE Trans. Software Eng.*, 25(2):207–217, 1999.

[9] Tugrul Dayar. *Analyzing Markov Chains using Kronecker Products: Theory and Applications*. Springer, New York, 2012.

[10] Tugrul Dayar and M. Can Orhan. Cartesian product partitioning of multi-dimensional reachable state spaces. *Probab. Eng. Inf. Sci.*, 30:413–430, 2016.

[11] Susanna Donatelli. Superposed stochastic automata: A class of stochastic Petri nets with parallel solution and distributed state space. *Perform. Eval.*, 18(1):21–36, 1993.

[12] Susanna Donatelli. Superposed generalized stochastic Petri nets: Definition and efficient solution. In Robert Valette, editor, *Application and Theory of Petri Nets 1994, 15th International Conference, Zaragoza, Spain, June 20-24, 1994, Proceedings*, volume 815 of *Lecture Notes in Computer Science*, pages 258–277. Springer, 1994.

[13] Hamed Ghasemieh, Anne Remke, and Boudewijn R. Haverkort. Survivability analysis of a sewage treatment facility using hybrid petri nets. *Perform. Eval.*, 97:36–56, 2016.

[14] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. John Hopkins Studies in the Mathematical Sciences. The Johns Hopkins University Press, 3 edition, 1996.

[15] Marco Gribaudo and András Horváth. Fluid stochastic Petri nets augmented with flush-out arcs: A transient analysis technique. *IEEE Trans. Software Eng.*, 28(10):944–955, 2002.

[16] Marco Gribaudo, Matteo Sereno, András Horváth, and Andrea Bobbio. Fluid stochastic Petri nets augmented with flush-out arcs: Modelling and analysis. *Discrete Event Dynamic Systems*, 11(1-2):97–117, 2001.

[17] Wolfgang Hackbusch. *Tensor Spaces and Numerical Tensor Calculus*. Springer, Heidelberg, 2012.

[18] Graham Horton, Vidyadhar G. Kulkarni, David M. Nicol, and Kishor S. Trivedi. Fluid stochastic Petri nets: Theory, applications, and solution techniques. *European Journal of Operational Research*, 105(1):184–201, 1998.

[19] Daniel Kressner and Francisco Macedo. Low-rank tensor methods for communicating Markov processes. In G. Norman and W. Sanders, editors, *Proceedings of the 11th International Conference on Quantitative Evaluation of Systems*, volume 8657 of *Lecture Notes in Computer Science*, pages 25–40. Springer, Heidelberg, 2014.

[20] Daniel Kressner and Christine Tobler. Preconditioned low-rank methods for high-dimensional elliptic PDE eigenvalue problems. *Comput. Meth. in Appl. Math.*, 11(3):363–381, 2011.

[21] Daniel Kressner and Christine Tobler. Algorithm 941: htucker - A matlab toolbox for tensors in hierarchical tucker format. *ACM Trans. Math. Softw.*, 40(3):22:1–22:22, 2014.

[22] Kristóf Marussy, Attila Klenik, Vince Molnár, András Vörös, István Majzik, and Miklós Telek. Efficient decomposition algorithm for stationary analysis of complex stochastic Petri net models. In Fabrice Kordon and Daniel Moldt, editors, *Application and Theory of Petri Nets and Concurrency - 37th International Conference, PETRI NETS 2016, Toruń, Poland, June 19-24, 2016. Proceedings*, volume 9698 of *Lecture Notes in Computer Science*, pages 281–300. Springer, 2016.

[23] Ivan V. Oseledets and Eugene E. Tyrtyshnikov. Breaking the curse of dimensionality, or how to use SVD in many dimensions. *SIAM J. Scientific Computing*, 31(5):3744–3759, 2009.

[24] Carina Pilch and Anne Remke. Statistical model checking for hybrid petri nets with multiple general transitions. In *47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2017, Denver, CO, USA, June 26-29, 2017*, pages 475–486. IEEE Computer Society, 2017.

[25] William J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, Princeton, NJ, 1994.

[26] Kathinka Wolter. *Performance and dependability modeling with second-order fluid stochastic Petri Nets*. PhD thesis, TU Berlin, 1997.