On Markov Reward Modelling with FSPNs

Katinka Wolter† and Andrea Zisowsky‡

† *Technische Universität Berlin, Institut für Technische Informatik, Franklinstr. 28/29, 10587 Berlin, Germany, email: katinka@cs.tu-berlin.de*

‡ *Technische Universität Berlin, Fachbereich Mathematik, Strasse des 17. Juni 136, 10623 Berlin, Germany, email: zisowsky@math.tu-berlin.de*

Abstract

In this paper fluid stochastic Petri nets (FSPNs) will be used for modelling reward in a performability model. Two variations of a known performability model are presented in order to demonstrate the ability of FSPNs in modelling accumulated rate reward as well as accumulated impulse reward. In the first model two fluid places are used, one of which represents the profit (reward) obtained by operating the system and the other one the buffer, that is approximated continuously. In the second model only one fluid place is used, representing the costs (negative reward) arising due to repair of system components. The costs increase continuously at deterministic rate while the system is in state of repair (which is a rate reward in the model). Additional costs incur each time the buffer fails (which is an impulse reward in the model). With a numerical solution algorithm the distribution of the reward and its mean are computed. The accuracy of the numerical algorithm is studied by showing for the first model the impact of the choice of the discretization stepsizes on the obtained solution. Different boundary conditions are discussed for the second model.

Key words: Accumulated Rate Reward, Accumulated Impulse Reward, Stochastic Petri Nets, Numerical Methods, Accuracy.

1 Introduction

Fluid Petri net models have gained much attention over the last years and a number of different classes of hybrid Petri nets have been proposed. For the specification and qualitative analysis the autonomous non-timed and timed Continuous Petri nets (ACPNs) [DA93] have been used. Performance and dependability evaluation has been carried out with fluid stochastic Petri nets (FSPNs) [TK93,HKNT98]. The discrete part of an FSPN taken on its own is a generalized stochastic Petri net (GSPN) [ABCD95] and the continuous part alone is equivalent to a fluid queueing system [New71,Kob74]. In FSPNs the continuous model part is represented by fluid places and fluid arcs. The level in the fluid places changes at deterministic rate. The stochastic behaviour of the model is determined by the discrete part and the dependencies between discrete and continuous sub-model. In this paper the formalism of FSPNs is extended in so far as in some discrete states the fluid places are emptied and filled at normally distributed rate [Wol99] and in other discrete states these rates are deterministic. In consequence the underlying differential equations are second order partial differential equations (PDEs) in states with normally distributed in- or outflow and first order PDEs in states with deterministic in- and outflow. Therefore, in the former states the model is called a second order FSPN and in the latter states a first order FSPN.

Already in [Hor96] FSPNs have been used for modelling accumulated rate reward. Reward structures are added to Markov models, giving Markov reward models. The measures computed from a Markov reward model are the accumulated reward and the completion time. Both are random variables following two different, but related unknown probability distributions [TR99]. In this paper only accumulated reward is regarded. Two kinds of reward can be distinguished, rate reward and impulse reward. Rate reward is the reward obtained while the model is in a certain discrete state. Impulse reward is achieved with the completion of an action, that is with the transition from one model state to the next. Both, rate and impulse can be instantaneous reward – which is the value of the reward at some point in time – or accumulated reward – the reward that has accumulated over a period of time. Furthermore, transient and steady state reward are distinguished [SM91].

The mean accumulated reward is computed through integration over the state probabilities of the Markov model multiplied with some reward rate for each state. In states where no reward is obtained the reward rate equals zero. In the computation of the mean impulse reward the state transition rate, that gives the impulse is a factor in the reward rate. The moments of the distribution of the accumulated reward are obtained by solving a differential equation using Laplace transforms. However, obtaining higher moments of the accumulated reward distribution is difficult. It has been shown in [Hor96] that the distribution of accumulated reward is computed by solving a differential equation like those underlying an FSPN.

Accumulated reward can be modelled with an FSPN in a very natural way. A fluid place holds the accumulated reward. In each state where rate reward is accumulated, the fluid place is filled at the reward rate. The main advantage of modelling reward with a fluid model is that the full probability density is obtained automatically and from it the moments can be computed.

In this paper not only rate reward but also impulse reward is modelled with an FSPN, which has not been done before. In the FSPN a special kind of arc, the jump arc is introduced to model the instantaneous increase or decrease of the level in a fluid place. The jump in the fluid level happens with the firing of the transition at the arcs origin, the height of the jump is determined by a probability distribution. The Dirac impulse gives a fixed height for each jump.

Jump arcs are similar to the flush-out arcs in [BGG+99,GSB99]. But flush-out arcs always empty a fluid place, whereas jump arcs remove or deposit a variable amount of fluid and hence allow for more modelling flexibility. In [BGG+99,GSB99] multidimensional fluid models are solved with an algorithm especially implemented for this specific problem. The numerical algorithms used for the computations in this paper are more general in that they can solve any FSPN with one or two fluid places. Numerical solution of second order FSPNs with two fluid places has for the first time been presented in an earlier version of this paper [WZ99].

Two variations of the performability model taken from [Mey82] will be studied. The modelled system is a buffered multi-processor system, where both the buffer and the processors can fail. The first model has two fluid places, one representing the buffer content and the other one holding the accumulated reward. The second model has only one fluid place representing the costs arising from repair of system components (negative reward). As long as repair is going on, inflow to the fluid place takes place. Each failure of the buffer causes additional costs once. This is modelled with a jump arc, along which a fixed amount of fluid is deposited.

An issue that is often raised in combination with numerical solution methods and fluid models is the accuracy of the solution. The accuracy is determined by the appropriateness of the model in representing the system and by the accuracy of the solution methods for the given model. We will only address the latter here. The order of accuracy of the discretization scheme can be determined by Taylor series expansions of the single terms of an equation. What this order means for a given model will be studied by using the first performability model. It will be solved with different discretization stepsizes and the solutions will be compared.

The numerical solution is always carried out on a finite domain. The larger the domain, the more expensive is the solution process with respect to memory as well as run time. The use of appropriate boundary conditions may help enormously in obtaining correct solutions, even on a small domain. We will discuss the issue of boundary conditions and take the second model for illustration, even though this is a topic of current research and no results can be presented yet.

In the next section the second order FSPN formalism is introduced. In section 3, the main part of the paper, the two models will be discussed. For both models the underlying differential equations are given. In the algorithm used for their solution the equations are generated automatically from a (textual) description of the Petri net. The transient evolution of the probability density is given as a result for both examples. Together with the first model accuracy of the solution and with the second model the different boundary conditions will be discussed. With section 4 the paper concludes.

2 The FSPN Formalism

For the definition of second order FSPNs the common notations for GSPNs [ABCD95] and for FSPNs [HKNT98] are taken over and extended as follows.

A second order FSPN is formally defined as an 8-tuple FSPN = $(\mathcal{P}, \mathcal{T}, \mathcal{A}, \mathbf{m}_0, r, b, g, w)$ where the set of places ($\mathcal{P} = \mathcal{P}_c \cup \mathcal{P}_d$) is divided into the *fluid (continuous)* and the *discrete* places. Discrete places (the elements of $\mathcal{P}_d$) are drawn as single-lined circles and hold an integer number of *tokens*, whereas the continuous, or fluid, places (the elements of $\mathcal{P}_d$) are drawn as two concentrical circles and they hold a real-valued amount of *fluid*. The set of transitions $\mathcal{T} = \mathcal{T}_E \cup \mathcal{T}_I$ is composed of the exponentially distributed, and the immediate transitions, respectively. Exponentially distributed transitions are drawn as empty rectangles and the immediate transitions as black bars.

The set of arcs $\mathcal{A} = \mathcal{A}_d \cup \mathcal{A}_c \cup \mathcal{A}_j$ is divided into three subsets, the discrete arcs (the elements of $\mathcal{A}_d$), the continuous, or fluid, arcs ($\mathcal{A}_c$) and the jump arcs ($\mathcal{A}_j$). The discrete and the jumps arcs are drawn as single lined arrows, whereas the fluid double-lined arcs are drawn like pipes. Discrete arcs can be input, output, or inhibitor arcs, where the latter have a little circle at their destination instead of the arrow head, while fluid and jump arcs can be only input or output arcs. The jump arcs and the continuous arcs only connect transitions and fluid places, and vice versa. Immediate transitions can be connected to a fluid place by a jump arc only if the immediate transition is also connected to a discrete place.

The *marking* $\mathbf{m} = (s, \mathbf{z}) \in \mathcal{M}$ consists of a discrete part $s = (\#p_i, i \in \mathcal{P}_d)$, where $\#p_i$ denotes the number of discrete tokens in the i-th discrete place p_i, and a continuous part, a vector representing the fluid level in each fluid place, $\mathbf{z} = (z_k, k \in \mathcal{P}_c) \in \mathcal{Z}$. The initial marking is $\mathbf{m}_0 = (s_0, \mathbf{z}_0)$. The total number of discrete states is S, the set of all discrete states is denoted $\mathcal{S}$.

$\mathrm{r}_{t,p}$, $\mathrm{r}_{p,t} : \mathcal{A}_c \times \mathcal{M} \rightarrow \mathrm{I\!R}^2$ is the flow rate function along the fluid arc connecting the timed transition t and the fluid place p or vice versa. The flow rate $r = (\mu, \sigma^2)$ is a normally distributed random variable specified by expectation μ and variance σ^2. Expectation and variance are fixed and time-independent. Let Z_k be the fluid level in place $k \in \mathcal{P}_c$. The change rate of the fluid level is again a normally distributed random variable with

$$\frac{dZ_k}{dt} = \sum_{t \text{ enabled in } \mathbf{m}} r_{t,p}(\mathbf{m}) - \sum_{t \text{ enabled in } \mathbf{m}} r_{p,t}(\mathbf{m}).$$

Reflecting barriers assure, that the fluid level in each place stays within its range $[z^{\min}, z^{\max}]$.

The jump height function $\mathrm{b}_{t,p}$, $\mathrm{b}_{p,t} : \mathcal{A}_j \times \mathcal{M} \to \mathcal{F}$ is a probability density function associated to a jump arc. The size of a fluid jump is a random variable drawn from that distribution.

$g_t : \mathcal{M} \to \mathbb{B}$ is the guard of transition t that can be a function of the discrete and the continuous state. $\lambda_t : \mathcal{M} \to \mathbb{R} \cup \{\infty\}$ is a function of both the continuous and the discrete marking. Immediate transitions have firing rate $\lambda_t = \infty$.

The weight function $w_t : \mathcal{S} \to \mathbb{R}$ is defined for immediate transitions. The firing probability of each of the enabled immediate transitions in a vanishing state is $w_t(s)/ \sum_{t_i \text{enabled in } \mathbf{s}} w_{t_i}(s)$.

The enabling rules for the transitions are the same as in discrete Petri net models. The fluid and the jump arcs act only on the fluid places and do not influence the enabling conditions of the transitions, other than through a guard function.

2.1 The Model Parameters

The dynamics of the CTMC underlying the discrete model part are defined through the generator matrix $\mathbf{Q}$, whereas the continuous dynamics are defined by the fluid change rate $\frac{dZ_k}{dt}$ that describes the change in the fluid level over time. The change in the fluid level is determined by the difference between the inflow and the outflow, each of which is assumed to be normally distributed or deterministic, if the variance of the normal distribution equals zero. The fluid flow is specified separately in each discrete state of the CTMC. The parameters of the normal distribution are either specified by the user or calculated as the diffusion approximation to the discrete transition at the origin of the inflow arcs and at the destination of the outflow arcs [Wol97].

Let μ_a and σ_a^2 be the expectation and variance of an arbitrary inter arrival time distribution of the discrete input transition to the fluid place P_k. Then the infinitesimal mean and variance of the fluid inflow rate are

$$\mu_\alpha = \frac{1}{\mu_a}, \qquad \sigma_\alpha^2 = \frac{\sigma_a^2}{(\mu_a)^3}, \tag{1}$$

and for the fluid outflow rate μ_δ and σ_δ^2 are analogous. (See [Kle76,New71] for a derivation of this formula).

All the incoming and outgoing streams in one marking are added and for the flow rate $R_k(s, \mathbf{z})$ of the fluid place k in every marking $s \in \mathcal{S}$ the following parameters are valid

$$\mu_{k,s} = \sum_{\substack{\text{inflows}\\ \text{to k in s}}} \mu_\alpha - \sum_{\substack{\text{outflows}\\ \text{of k in s}}} \mu_\delta \quad \forall k \in \mathcal{P}_c, s \in \mathcal{S}, \tag{2}$$

$$\sigma^2_{k,s} = \sum_{\substack{\text{inflows}\\ \text{to k in s}}} \sigma^2_\alpha + \sum_{\substack{\text{outflows}\\ \text{of k in s}}} \sigma^2_\delta \quad \forall k \in \mathcal{P}_c, s \in \mathcal{S}, \tag{3}$$

Note, that this simple linear combination only holds, because all the single fluid streams are normally distributed. No covariances are needed because the PDEs (9) and (13) only hold on the interior of the domain, where inflow and outflow can be assumed to be independent of each other. In the boundary points, where restrictions on the relation of in- and outflow have to be imposed, the PDEs are replaced by the boundary condition (11). The expectations and variances for each fluid place are collected into diagonal matrices (if there is none in a discrete state, the entry is equal to zero):

$$\boldsymbol{M}_k = \mathbf{diag}(\mu_{k,1}, \dots, \mu_{k,S}), \tag{4}$$

$$\mathbf{\Sigma}^2_k = \mathbf{diag}(\sigma^2_{k,1}, \dots, \sigma^2_{k,S}). \tag{5}$$

If a hybrid system is modelled $\mu_\alpha, \sigma^2_\alpha, \mu_\delta$ and σ^2_δ are given by the user and (1) is not needed.

The jump height function $b(.)$ can have an arbitrary probability density. For all discrete states the functions are collected into a diagonal matrix $\mathbf{B} = \mathbf{diag}(b_1(.), \dots, b_S(.))$.

3 The Models

Two models shall be studied in this paper. Both are based on the performability model of a buffered multiprocessor system presented in [Mey82]. The first one uses two fluid places, one for the buffer content and the other for accumulating reward, that flows in as long as servers and buffer are operational. The second model has only one fluid place, that represents the costs arising from repair of processors and buffer. As in reference [Mey82], in the first model processors and buffer may fail but no repair is considered. Opposed to this in the second model both processors and buffer can be repaired.

Reward increases at constant speed in each discrete state. Therefore the inflow rate to the fluid places representing the accumulated reward has zero variance. In the first model the buffer is represented by a fluid place which is emptied and filled at randomly varying rate. So, the first model is a second order FSPN in all states that have an in- or outflow of the buffer, whereas the second model purely is a first order FSPN with fluid jumps.

3.1 Two-dimensional Fluid Rate Reward Model

In this section at first the model will be described in detail. Then the corresponding differential equations and their solutions are explained. At last, in a separate subsection, accuracy of the obtained solution will be studied. In the performability model there are three processors that may fail independently at rate λ_p. The buffer has L stages that can fail independently at rate λ_b. If one of the stages fails, it causes a crash of the complete system. The failure rate of the whole buffer is therefore $L\lambda_b$.

Jobs arrive to the system at rate α and they are served at rate δ per processor. So the buffer is being filled as long as it is operational, even if all servers are down. There is an outflow of the buffer as long as at least one processor is operational. Reward is accumulated at rate r per operational processor and while the buffer is not empty. The reward inflow r_r is a function of the global state of the system and is specified in each discrete state of the model.

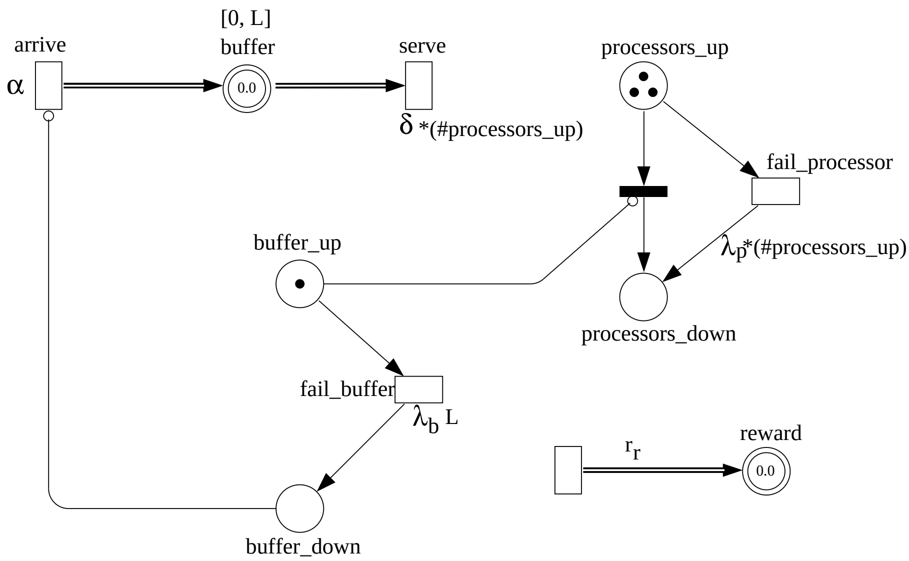

Figure 1. FSPN model of the performability model

In this model the buffer is considered as being filled with a large number of items, that are measured in some unit. So the arrival and service processes can be seen as continuous streams entering and leaving the buffer. The inflow and outflow rates of the fluid place representing the buffer are approximations computed from the corresponding discrete streams.

Figure 1 shows an FSPN model of the system. The buffer is represented by the one fluid place. Its capacity is $L = 4$. When the buffer is down all processors are put into the set of failed processors and the system terminates because none of the transitions is enabled. Complete failure of the system is an absorbing state. The second fluid place represents the accumulated reward. The rate function r_r is defined in such a way, that there is an inflow of 0.1 units per operational processor and per time unit only if the buffer is not empty. Hence the fluid flow rate accumulating reward depends on the content of the buffer.

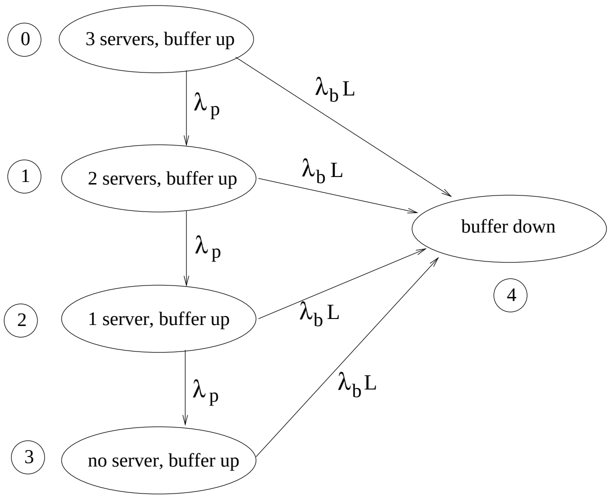

Figure 2. Reachability graph of the FSPN model

The reachability graph is shown in figure 2. Only the discrete states are labelled, the components for the two fluid levels are omitted. The numbers in the small circles denote the numbering of the discrete states as it is used for the parameter vectors and the results. The failure rate of the buffer has been set to $\lambda_b = 10^{-4}$ per hour and the processor failure rate is $\lambda_p = 10^{-3}$ per hour. The mission time is 100 hours.

For the computation of the fluid change rates (1), (2) and (3) have been used. In all states s where the buffer is operational the expectation μ_b of the fluid change rate of the buffer is computed following (2):

$$\mu_b = \mu_\alpha - \#\text{servers\_up} \cdot \mu_\delta$$

and the variance σ_b^2 equals according to (3)

$$\sigma_b^2 = \sigma_\alpha^2 + \#\text{servers\_up} \cdot \sigma_\delta^2.$$

For the exponential distribution (1) evaluates to $\mu_\alpha = \alpha$ and $\sigma_\alpha^2 = \alpha$. The same holds for the exponentially distributed service with rate δ. With $\alpha = 0.75$ and $\delta = 0.5$ the fluid flow parameters evaluate to

$$\boldsymbol{M}_b = \mathbf{diag}(-0.75, -0.25, 0.25, 0.75, 0.0) \tag{6}$$

and

$$\mathbf{\Sigma}_b^2 = \mathbf{diag}(2.25, 1.75, 1.25, 0.75, 0.0). \tag{7}$$

Reward is accumulated at constant rate of 0.1 per operational processor. Hence, the fluid flow parameters for the second fluid place are

$$\boldsymbol{M}_r = \mathbf{diag}(0.3 \cdot r_{rew}, 0.2 \cdot r_{rew}, 0.1 \cdot r_{rew}, 0.0, 0.0) \tag{8}$$

with

$$r_{rew} = \begin{cases} 1.0 & \text{if } z_1 \geq 0.2 \\ 0.0 & \text{else} \end{cases}$$

and $\mathbf{\Sigma}_r^2$ is the matrix with all entries equal to zero.

The reachability graph gives a graphical representation of the stochastic process underlying this model. Augmenting it with two continuous variables, one for each fluid place, gives a description of the full stochastic process which is formally defined as

$$\{(S(t), Z_1(t), Z_2(t)), t \in \mathbb{R}_0^+\},$$

where $S(t)$ is the discrete marking at time t, and $Z_k(t)$ is a random variable, representing the fluid level in fluid place k at time t.

At time t the transient probability of the model being in discrete state i with fluid level in an infinitesimal interval around (z_1, z_2), is called the *fluid density* and is denoted by $\pi_i(t, z_1, z_2) = \frac{\partial}{\partial z_1} \frac{\partial}{\partial z_2} P(S(t) = i, Z_1(t) \leq z_1, Z_2(t) \leq z_2)$. This density is the component-wise solution of the following differential equation in vector form

$$\begin{aligned} \frac{\partial \boldsymbol{\pi}(t, z_1, z_2)}{\partial t} &+ \frac{\partial}{\partial z_1}\Big(\boldsymbol{\pi}(t, z_1, z_2)\mathbf{M}_b(z_1)\Big) + \frac{\partial}{\partial z_2}\Big(\boldsymbol{\pi}(t, z_1, z_2)\mathbf{M}_r(z_2)\Big) \\ &- \frac{1}{2}\frac{\partial^2}{\partial z_1^2}\Big(\boldsymbol{\pi}(t, z_1, z_2)\mathbf{\Sigma}_b^2(z_1)\Big) - \boldsymbol{\pi}(t, z_1, z_2)\mathbf{Q}(z) = 0 \\ &\forall t, \forall z \in \mathring{\Omega} = (z_1^{\min}, z_1^{\max}) \times (z_2^{\min}, z_2^{\max}). \end{aligned} \tag{9}$$

Note that there is only one matrix of variances while there are two matrices of means, one for each continuous direction. Eqn. (9) is a system of diffusion-convection equations, also called Fokker-Planck equations. The system of equations is coupled through the generator matrix $\mathbf{Q}$ of the discrete Markov process. The expectations $\mathbf{M}$ define the drift and the variances $\mathbf{\Sigma}^2$ define the diffusion.

An initial as well as a boundary condition on either end of the domain must be imposed. The initial condition is defined by the model's configuration at time zero, whereas the boundary conditions are determined by the fact that the integral over a density always evaluates to one.

Let $\boldsymbol{\pi}_0$ be the vector of initial discrete state probabilities and let $\mathbf{z}_0$ be the vector of initial fluid levels, then the initial conditions are

$$\boldsymbol{\pi}(0, z_1, z_2) = \boldsymbol{\pi}_0 \cdot \delta(z_1 - z_{1,0})\delta(z_2 - z_{2,0}), \tag{10}$$

where $\delta(z_k - z_{k,0})$ is the Dirac impulse at $z_k = z_{k,0}$.

The boundary condition is equivalent to a reflecting barrier at the origin and at the upper bound [CM65]

$$\sum_{k=1}^{2} \left[\frac{1}{2} \frac{\partial}{\partial z_k} \boldsymbol{\pi}(t, z_1, z_2) \boldsymbol{\Sigma}_k^2(z_1, z_2) - \boldsymbol{\pi}(t, z_1, z_2) \mathbf{M}_k(z_1, z_2) \right]_{z=\{z_1^{\min}, z_1^{\max}\} \times \{z_2^{\min}, z_2^{\max}\}} = 0. \tag{11}$$

Equation (9) subject to (10) and (11) has been discretized and solved numerically [Zis98]. The discretization scheme will be described in some detail in the following subsection when regarding the accuracy of the solution.

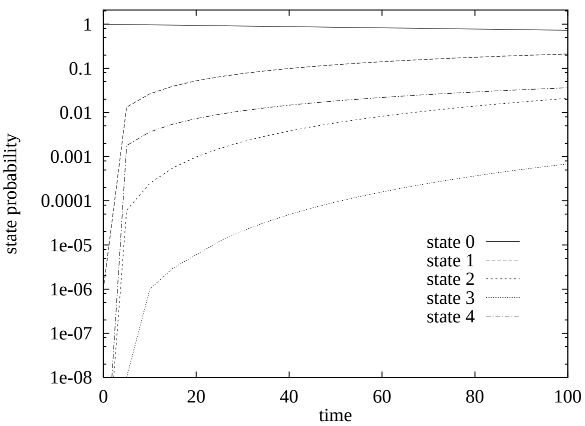

Figure 3. Discrete state probabilities

The result after applying the numerical algorithm is in each discrete state a two-dimensional density, that evolves over time [1]. The sum over all discrete states is a two-dimensional probability density at each time point. The discrete state probabilities are obtained by integrating over both continuous dimensions for one discrete state at each point in time. The state probabilities for all discrete states are shown in logarithmic scale in figure 3.

The left column of Figure 4 shows the probability densities in state zero at selected time points. The probability mass in this state decreases with time.

[1] Strictly speaking this is a density only after normalization. Nonetheless, we will call the fractions of a density also density.

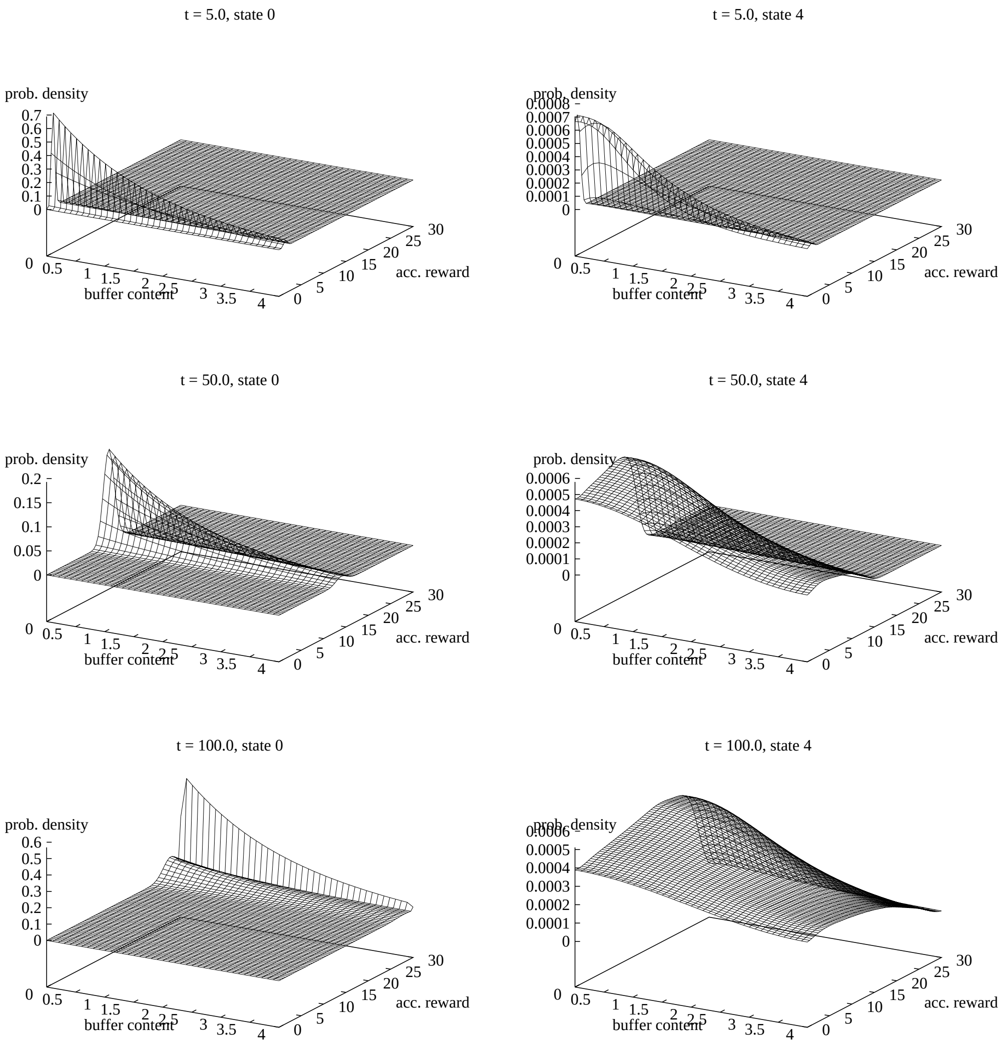

Figure 4. Transient probability density in state zero

Therefore the scaling of the z-axes could not be kept constant. It can be seen, that the accumulated reward increases, while the probability of the buffer being full rather decreases.

In state four, shown in figure 4 in the right column, the buffer is down and less reward is accumulating. One does not see the behaviour the system would show, was it in that state for a long time, because failures can happen at any time with reward having been already accumulated and the buffer being already filled. Therefore even in states where buffer and processors are down one will still have a small probability of an accumulation of reward and of the buffer not being empty.

For comparison reasons the density in state 3 at time $t = 100$ is shown in figure 5. In this state all processors are down but the buffer is still operational and is being filled.

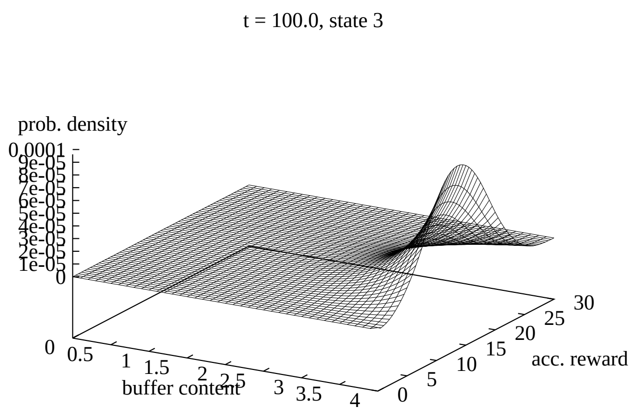

Figure 5. Transient probability density in state three

The mean accumulated reward increases constantly with a gradient of 0.25, whereas the mean amount in the buffer increases slowly and always stays below 1.4. Both can be seen in figure 6.

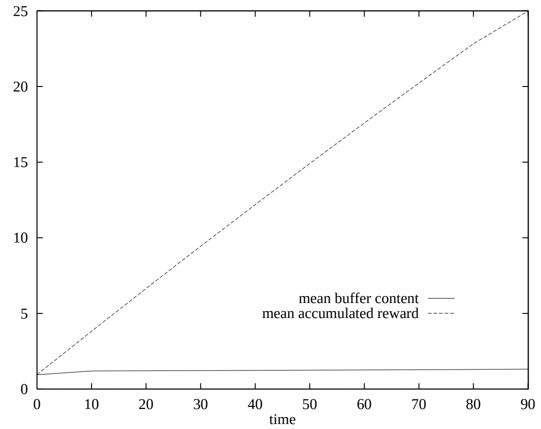

Figure 6. Mean buffer content and mean accumulated reward

The density of the accumulated reward is obtained by integrating at each time point over the buffer content, the second continuous dimension. The resulting function has one dimension less, and its numerical approximation can be plotted over time (see figure 7).

In figure 7 oscillations at early times can be noticed. They disappear with time, which shows the smoothing property of the diffusion-convection equation. It also means, that at early times a finer grid would be better, whereas after 60 time units larger stepsizes would probably do quite as well as the used ones. Here, the benefit of adaptive stepsizes becomes apparent. This is, however left for future work.

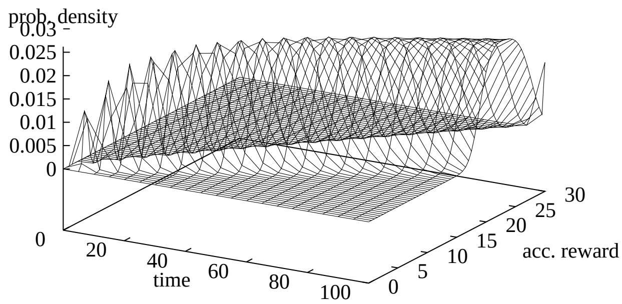

Figure 7. Marginal density for the accumulated reward in state one

New information obtained from this model compared to known solution methods for Markov reward models is the shape of the distribution of accumulated reward. Having an approximation of the full density allows for the computation of quantiles and higher moments of the distribution. The basic measures, such as the state probabilities and the mean accumulated reward have been obtained before.

3.2 Accuracy of the algorithm

At this place we will discuss the accuracy of the applied algorithms. First we briefly describe the discretization scheme and its theoretical properties. It is not possible to make an a-priori estimate on the accuracy for a given model solution. Only the order of accuracy for one discretization step can be given. The theoretical results will be illustrated using the two-dimensional reward model from this section.

Discussing the accuracy of the obtained solution poses at first the question whether the model forms a correct picture of the system to be analyzed. We will not address this issue here, because there is no metric in which to measure modelling quality. Instead we will elaborate on the solution quality for a given model. More specific, the difference between the analytical formula and the computed numerical approximations will be regarded. The accuracy of the numerical solution is influenced first by the quality of the discretization of the differential equation and second by the accuracy of the solver for the system of linear equations.

The discretization of the system of differential equations is carried out on an equidistant grid with stepsize k in time and stepsizes h_1, h_2 in the two space directions.

The system of equations has been discretized using a θ-scheme. The parameter θ weighs two neighbouring time levels. It can take on all values between zero and one. Depending on θ the discretization is explicit ($\theta = 0$), implicit ($\theta = 1$) or the well-known Crank-Nicolson scheme ($\theta = 1/2$).

For the first derivatives in space an upwind scheme is used. Upwind means that forward and backward differences are used, weighed with the upwind parameter ρ and $(1 - \rho)$, respectively. The parameter ρ depends on the sign of the convection parameter μ. This is motivated by the fact that if e.g. $\mu(z_i) > 0$, the mass moves to the right and the backward difference is more apropriate in describing this motion. Hence, a reasonable choice of ρ is the following:

$\mu_{s,j}(z) > 0$:	$0 < \rho_{s,j} < \frac{1}{2}$	backward difference weighs more,
$\mu_{s,j}(z) < 0$:	$\frac{1}{2} < \rho_{s,j} < 1$	forward difference weighs more.

For the second space derivatives the second order difference quotient and for the time derivative the forward difference is used. The coupling term $\boldsymbol{\pi Q}$ is discretized explicitly, using the two previous time levels. Therefore the set up system of difference equations can be decoupled. A separate system of linear difference equations is solved for each discrete state, and the coupling term appears in each one as an inhomogeneity.

The system of linear equations is for two-dimensional models solved with a LU factorization using pivotizing, for one-dimensional models a sparse least square roots method with iteration precision 10^{-8} is used. The solvers for the systems of linear equations are taken from the package *meschach* [SL94]. The accuracy of these solvers can not be measured separately. We can only study their accuracy in combination with the discretization of the differential equation.

Following [Str89], the main criterion for the quality of a discretization scheme is convergence. A discretization scheme is convergent if the numerical solution approaches the analytical one as the stepsizes are decreased.

For a given discretization convergence is in general not easy to show, but equivalently two related properties can be proven, stability and constistency. A discretization scheme is convergent if and only if it is stable and consistent. Stability is the property, which describes the robustness of the algorithm to small changes in the initial conditions or small random fluctuations. The algorithm is consistent with the equation it describes, if the discrete equation approaches the continuous one for decreasing stepsizes.

Both properties can be shown for the differential equations underlying an FSPN by using the methods in [Str89]. Here only the two-dimensional fluid model will be considered.

Taylor series expansions of the single terms are used to determine the order of consistency. In general the order of consistency is $O(k, h_1, h_2)$. For the Crank-Nicolson scheme the order is $O(k^2, h_1, h_2)$, and setting the upwind parameter to $\rho = 0.5$ yields the central difference for the first space derivatives and also an order of two in each space dimension.

With a von Neumann analysis [Str89] we showed, that the scheme is unconditionally stable for $\theta \geq 0.5$. For all other choices of θ the condition

$$(1-2\theta)\left(\sigma_{1,s}^2 \frac{k}{h_1^2} - \frac{k}{h_1}\mu_{1,s}(2\rho_{1,s}-1) + \sigma_{2,s}^2 \frac{k}{h_2^2} - \frac{k}{h_2}\mu_{2,s}(2\rho_{2,s}-1)\right) \leq 1$$

must hold for every state s in each grid point i for the scheme to be stable. Furthermore the following conditions must hold

$$\rho_{i,j} \leq \frac{\sigma_{i,j}^2}{2h_i\mu_{i,j}}, \qquad \text{if } \mu_{i,j} > 0, \tag{12a}$$

$$\rho_{i,j} \geq 1 - \frac{\sigma_{i,j}^2}{2h_i|\mu_{i,j}|}, \qquad \text{if } \mu_{i,j} < 0, \tag{12b}$$

where $i = 1, 2$ is the index for the space direction. This can be assured by a suitable choice of the upwind parameter ρ.

Since the used algorithm for solving (9) could be shown to be stable and consistent with (9) we can conclude, that smaller stepsizes will yield a better approximation. The solution obtained with the smallest stepsize can therefore be taken as a reference solution.

The following table lists the different models that have been solved, where always $\theta = 1$, and ρ was determined according to (12).

Model nr.	h_1	h_2	k	l1-norm ($t = 60.0$)
0	0.01	0.01	0.01	–
1	0.05	0.05	0.01	0.99999992913917101767
1a	0.05	0.1	0.01	0.99999991856169989646
1b	0.05	0.05	0.05	0.99999999507102010909
2	0.1	0.1	0.01	0.99999998987893468083
3	0.2	0.2	0.01	0.99999999550397478121
4	0.5	0.5	0.01	0.99999998341952289760

Table 1
List of different models and their l1-norms

As it can be seen in table 1, the l1-norm[2] of the numerical solution of each model is even at $t = 60$ (which means $6 \cdot 10^3$ time steps) very close to one. This

[2] The norms are computed as $l1 - norm = h_1 \cdot h_2 \cdot \sum_{\text{all grid points}} \pi$ and $l2 - norm = h_1 \cdot h_2 \cdot \sum_{\text{all grid points}} \pi^2$

is achieved through especially taylored *discrete reflecting boundary conditions*. In deriving the boundary conditions the conservation of probability mass is used as a condition that must be fulfilled. This makes the commonly applied normalization step obsolete. Model number zero could not be solved due to memory overflow, but it is used in the discussion.

In the following we will compare the solutions obtained by using the different stepsizes. We will use in each model the sum over all discrete states, instead of the densities in single discrete states and compare the estimates of the absolute error. The solution of model number 1 is used as the reference solution, the solution that is closest to the real one. We compute the difference of each density to the reference solution by taking the difference in each grid point. The obtained data are the curves (model_1a - model_1), (model_1b - model_1), and so on. For each of the difference data we compute the maximum value, the l1- and the l2-norm as a measure for comparison. They are listed in table 2.

Data	maximum absolute error	l1-norm	l2-norm
diff(1a,1)	0.042586117000	0.216551	0.002705
diff(1b,1)	0.013350898600	0.066889	0.000233
diff(2,1)	0.119628770000	0.890237	0.040651
diff(3,1)	0.154807305500	1.251118	0.066130
diff(4,1)	0.192311555200	1.832396	0.102005

Table 2
Maximum absolute error, l1- and l2 norm of the deviation data

At a first glance it seems surprising, that taking five times the stepsize in time (model_1b) gives a solution that differs much less from the reference solution than the solution obtained with twice as large stepsizes in space (model_1a). The reason becomes apparent, if one considers the number of grid points, which was chosen to discretize space and time variables (see table 3). The discretization in time of model_1b is already sufficiently small (1200 grid points), that it causes no large error. Whereas the reduction from 601 to 301 grid points in space leads to an inadequate resolution, especially since there is no smoothing in z_2 direction ($\mathbf{\Sigma}_r^2 = 0$).

It would have been desirable to solve model0, but the discretization in space results in a grid with more than a million gridpoints, so in each time step and for each discrete state there is a system of over one million equations to solve. Here again one sees the need for improvement of the numerical methods. There are discretization methods known that use only the sum of the gridpoints in each direction not their product, the alternating direction implicit (ADI) schemes. Implementation of such a scheme is also left for future work. A smaller timestep increases the runtime, but does not need additional memory.

Besides that, model_2, model_3 and model_4 have all stepsizes with a deviation

Model nr.	z_1	z_2	t
0	401	2701	$6 \cdot 10^3$
1	81	601	$6 \cdot 10^3$
1a	81	301	$6 \cdot 10^3$
1b	81	601	$12 \cdot 10^2$
2	41	301	$6 \cdot 10^3$
3	21	151	$6 \cdot 10^3$
4	9	61	$6 \cdot 10^3$

Table 3
Number of grid points

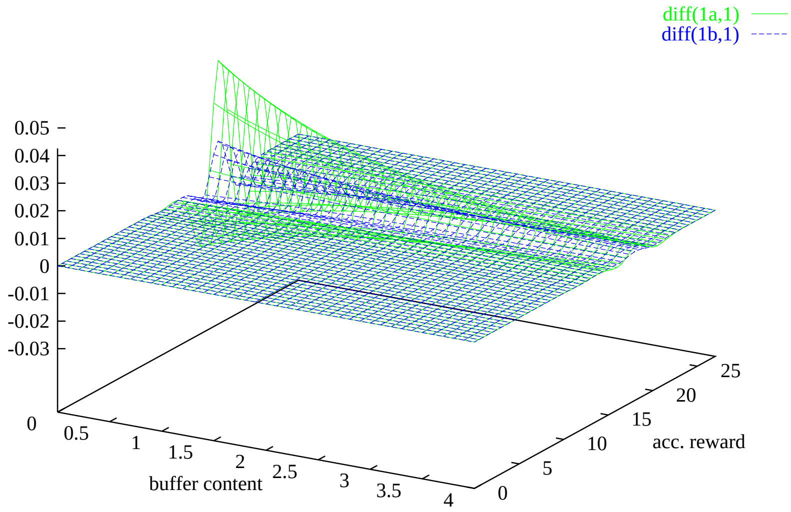

Figure 8. Deviation curves (a)

of one order of magnitude from the reference solution and their maximum error is almost equal, while the norms differ roughly by a factor two. Figures 8 through 11 show the probability density for the whole model, summed over all discrete states, at time $t = 60.0$. Each plot shows two difference curves in the order of the stepsizes. So, figure 8 shows diff(1a,1) and diff(1b,1), which shows that the error diff(1b,1) is substantially smaller. In figure 9 diff(1a,1) and diff(2,1) are compared, showing that the error significantly increases with larger stepsizes, as can also be seen in figures 10 and 11.

It becomes clear, that by choosing too large stepsizes the solution can be up to 0.25 off a good numerical solution. That corresponds to roughly 10% of deviation. Furthermore, the order $O(k, h_1, h_2)$ can not be directly translated

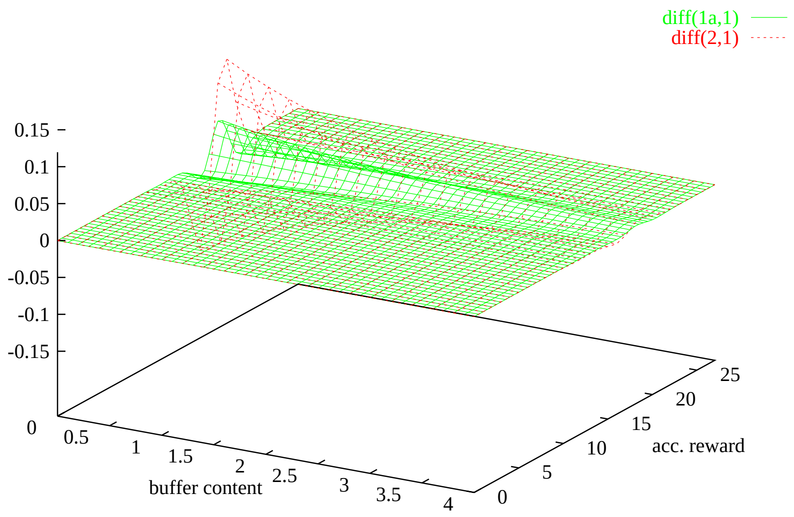

Figure 9. Deviation curves (b)

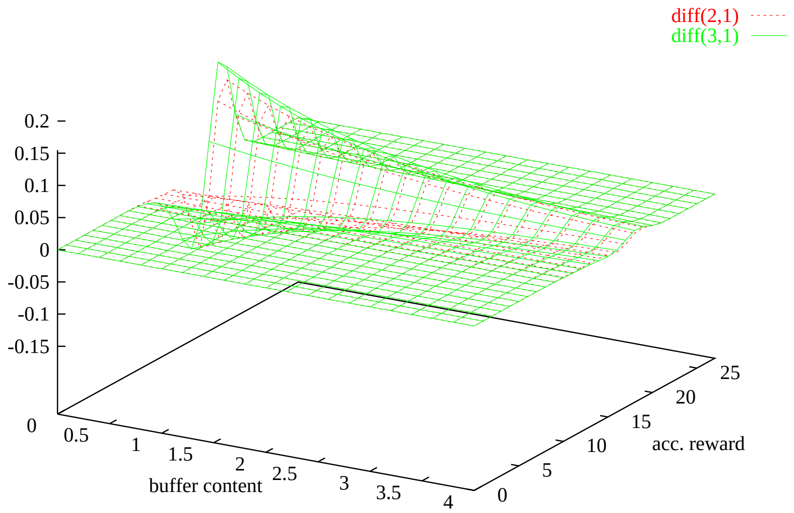

Figure 10. Deviation curves (c)

into a relation of the stepsizes with the error. It can only be said, that the maximum error and its l1-norm is significantly larger than the largest stepsize used in the models involved in the comparison.

Of course, these observations hold only for the given model and will be different in numbers for all other models that will be regarded.

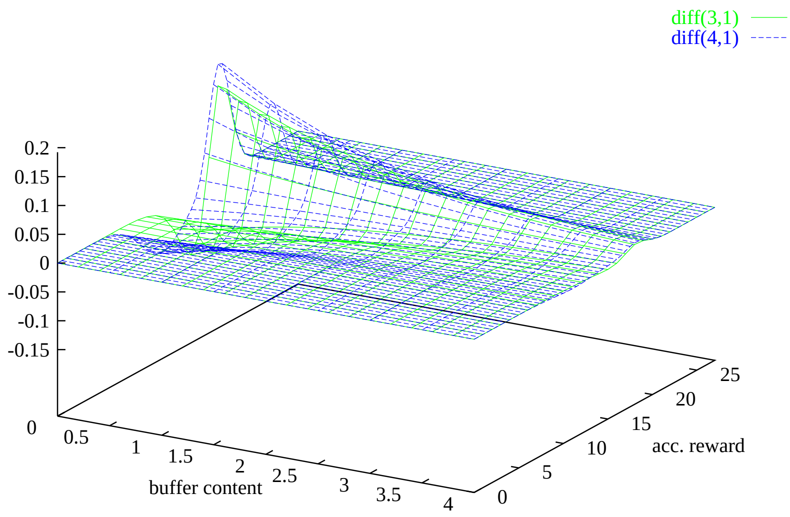

Figure 11. Deviation curves (d)

3.3 Rate and Impulse Reward Model

The second example investigated in this paper is another variation of the performability model. The buffer is modelled with a discrete place so there is only one fluid place. This fluid place represents the costs that arise from repair of the processors as well as the buffer. Each failure of the buffer costs an additional price once, whereas the repair costs per unit of time. The failure rate of the processors is again $\lambda_p = 0.001$ per processor and the repair rate is set to $c_p = 0.25$. The failure rate of the buffer has been increased by two orders of magnitude and is now set to $\lambda_b = 0.04$. This has been done because otherwise the effect of the fluid jumps is not visible in the solutions at all. For the same reason the repair rate of the processors has been chosen rather small. For realistic failure and repair rates there is hardly any accumulation of costs, because the probability of the system being in a failed state is extremely low. Repair of the buffer takes on the average 6 hours ($r_b = 1/6$).

The arrival rate to the buffer is $\alpha = 0.75$ and the service rate is $\delta = 0.5$ per operational processor, as in the first model. The costs that are incurred by the repair of processors and buffer are deterministic at rate 0.1 per failed processor and rate 0.5 for the repair of the buffer and maintenance of the whole system. In consequence, the inflow to the fluid place representing the costs has only a mean, but no variance. Each failure of the buffer costs an additional deterministic amount of 0.5 units.

The buffer has only $L = 2$ stages. The failure rate of the whole buffer is again $L\lambda_b$ and the repair rate of the whole system after a buffer failure is r_b. Upon failure of the buffer data is saved, so that the system can continue processing after repair. With the repair of the buffer the whole system is checked, so that subsequently all processors are up again. It is assumed that there are enough repair facilities to even fix all processors, should they all be down.

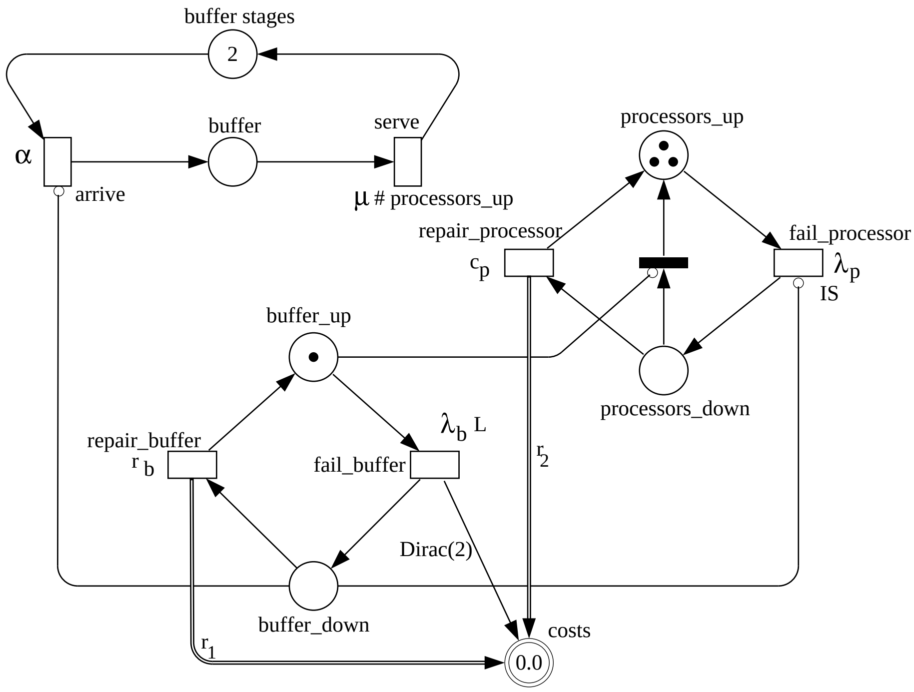

Figure 12. FSPN model of the second performability model

Figure 12 shows an FSPN model of the system. The costs are represented by the fluid place. A jump arc connects transition *fail_buffer* with the place *costs*. When the buffer is down all the failed processors are put back into the set of operational processors. As long as repair of the buffer takes all the processors are in maintenance and repair, and the transition representing the processor failure is disabled as is the arrival of jobs. The main difference between this model and the first one is that the fluid place is not only filled through the fluid arcs but also through a jump arc.

The reachability graph of this model is shown in figure 13. It can be divided in three blocks, each of which represents the system with a different number of tokens in the buffer. Transitions between the blocks are due to arrivals to the buffer and the service of tokens from the buffer. To make the figure a little clearer these transitions are drawn with dashed arcs and have double arrows to represent two arcs. Again the states are only labelled with a description of the discrete state and the continuous variable is omitted. The numbers in the small circles are the numbering used for the results. The model is solved transiently until time 170.0.

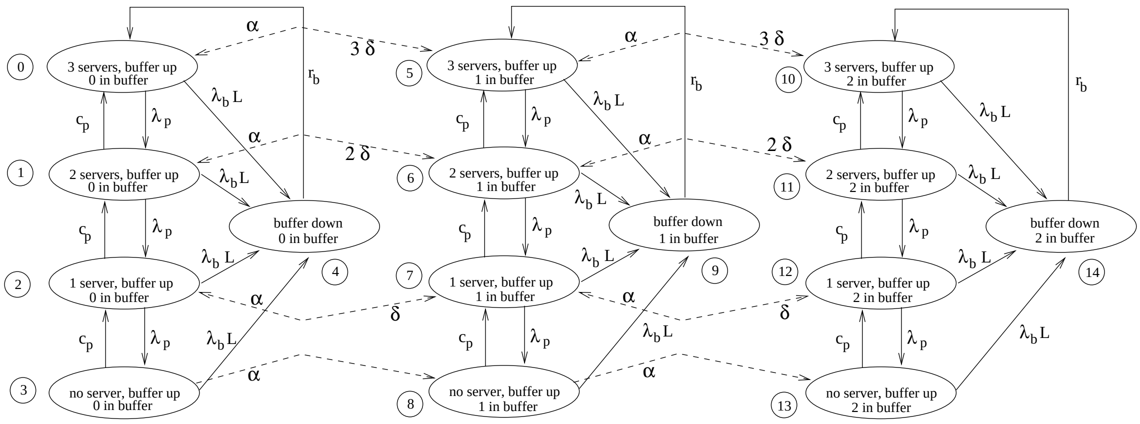

Figure 13. reachability graph of the second performability model

The fluid flow parameters are the same in all three blocks. There is only an inflow to the fluid place, so (2) is given by the Kronecker product $\mathbf{M}_b = \mathbf{diag}(\mu_{r_1} + \mu_{r_2}) = \mathbf{I} \otimes \mathbf{diag}(\boldsymbol{\mu})$ where $\mathbf{I}$ is the (3×3) identity matrix, $\boldsymbol{\mu} = (0.0, 0.1, 0.2, 0.3, 0.5)$ and $\boldsymbol{\Sigma}^2{}_b = \mathbf{0}$. The jump height matrix is $\mathbf{B} = \mathbf{I} \otimes \mathbf{diag}(\mathbf{b})$, where $\mathbf{b} = (det(0.5), det(0.5), det(0.5), det(0.5), 0.0)$ and finally the jump rates are $\boldsymbol{\Lambda} = \mathbf{I} \otimes \mathbf{diag}(\boldsymbol{\lambda})$, where $\boldsymbol{\lambda} = (0.04, 0.04, 0.04, 0.04, 0.0)$. The jump height distribution $det(x)$ denotes the deterministic jump height, implemented by the Dirac impulse at x.

The stochastic process under consideration is a Markov process similar to the one in the first example with only one continuous component. The fluid density in state i is $\pi_i(t, z) = d/dz\, P(S(t) = i, Z(t) \le z)$ and the equation describing the models dynamics is in vector-matrix notation

$$\frac{\partial \boldsymbol{\pi}(t,z)}{\partial t} + \frac{\partial}{\partial z}\Big(\boldsymbol{\pi}(t,z)\mathbf{M}(z)\Big) - \boldsymbol{\pi}(t,z)\Big(\mathbf{Q}(z) - \boldsymbol{\Lambda}(z)\Big) - \int_{z^{\min}}^{z} \boldsymbol{\pi}(t,y)\mathbf{B}(z-y)\, dy\, \boldsymbol{\Lambda}(z) = 0 \qquad \forall t, \forall z \in \mathring{\Omega}. \quad (13)$$

The derivative terms are equivalent to those of the first example reduced to one dimension. Note that there is no second order derivative and no variance. The integral is similar to the integral in the Takács integro-differential equation [Tak55], it is the convolution of the jump height distribution with the fluid density function, multiplied with the rate λ a jump occurs at. These rates have to be subtracted from the diagonal elements of the generator matrix. This means that some mass is removed from the state probability vector and added again through the convolution in a different place on the continuous axis. The convolution gives the density of the sum of two random variables. Here these two random variables are the state vector π and the random variable B of the jump height. For the solution of the differential equation part of (13) the equivalent algorithm to the two-dimensional one is used. The integrals are solved explicitly on the known time step with a simple rectangular summation.

The solutions are plotted for two selected discrete states. The left plot in figure

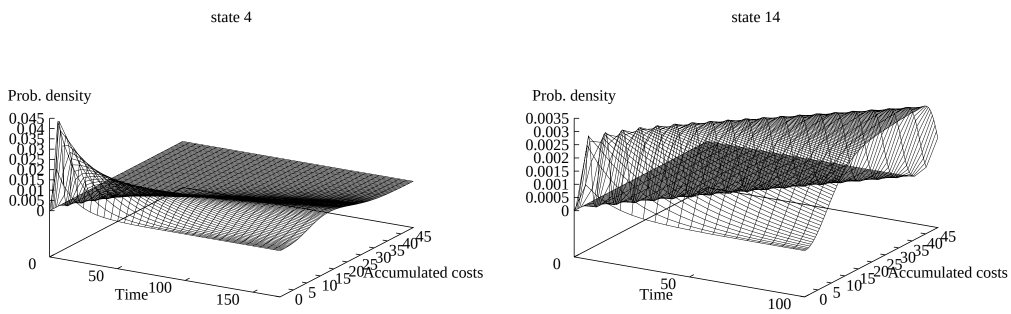

Figure 14. Transient probability density in selected discrete states

14 shows the density in state 4 (buffer down and empty). This is a state where repair at high costs takes place. As opposed to the first model in this model the discrete transition rates are comparable with the fluid flow rates. Therefore the initial peak is transferred partly to the neighbouring states and dominates the curves in those states as well. States 3, 9 and 13 have zero probability still after 170 time units. Only in state 14 (buffer down and holding 2 tokens), shown in the upper right plot of figure 14 the impact of the jumps can be seen clearly. The data for this plot has been slightly manipulated in that all solutions after time 120.0 have been dropped, in order to show the irregular shape of the curve more clearly.

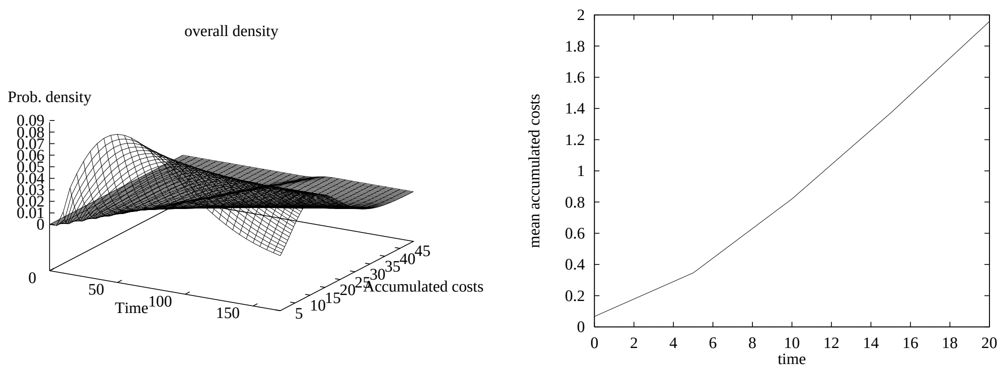

Figure 15. Density of the costs and mean accumulated costs

The left plot in figure 15 shows the fluid density summed over all discrete states. Here all data for the accumulated reward less than 5 units has been dropped. This is done to remove the peak near the origin so that the plot scales differently and the rest of the shape becomes visible. The contribution of state 14 can be faintly seen.

The mean costs are shown in the right plot in figure 15. The curve is plotted only for the first 20 time units. The mean costs increase at slightly rising pace as long as the discrete state probabilities are not yet in steady state.

Looking at the probability density in state 14 in figure 14, one can easily guess the problems concerning this example: Although the modeled accumulated costs z are not necessarily bounded from above, we use reflecting boundary conditions. To obtain reasonable results, we had to choose the domain of computation large enough, such that the accumulated mass at the boundary had no influence on the solution in the selected output range. This procedure causes a lot of unnecessary numerical effort.

To remedy this situation, so called *discrete transparent boundary conditions* can be used, which are capable of modelling solutions on unbounded domains *exactly*. The derivation and implementation of these boundary conditions is currently under investigation. The basic idea is to solve the z-transformed (using a discrete Laplace transformation) difference equation for $z > z^{max}$ and to match it at the boundary $z = z^{max}$ with the numerical solution. If the model parameters are constant for $z > z^{max}$, this can be done analytically, since the transformation with respect to the time variable yields an ordinary second order difference equation with constant coefficients.

These boundary conditions let probability mass go through - therefore the name *transparent*. In consequence, after probability mass has reached the boundary, the l1-norm will no longer evaluate to one. In return, the solution on the regarded domain approximates the differential equation, which is valid without any boundary conditions.

The concrete formulation of these boundary conditions and their application to FSPNs will be presented in a subsequent paper.

4 Conclusions

Two models have been presented in order to demonstrate the modelling power of FSPNs in Markov reward modelling. Both are modifications of a well-known performability model. It has been shown that not only accumulated rate reward can be modelled with an FSPN, but also accumulated impulse reward.

Rate reward is accumulated in an FSPN through the inflow along a fluid arc to a fluid place while a transition is enabled, whereas impulse reward is added upon the firing of a transition as indicated by a jump arc. Jump arcs are labelled with a probability distribution. An amount sampled from that distribution is added to the content of the fluid place at the destination of the jump arc upon the firing of the transition at its origin. The example in this paper uses the Dirac impulse as probability distribution to model deterministic jump height.

The underlying equations are differential equations that also include convolution integrals if jump arcs are present in the model. A numerical discretization method has been used to solve the equations. The numerical algorithm is also able to handle models with two fluid places. Therefore in the first example it was possible to use one fluid place for the accumulated reward and the other one for a continuously approximated buffer. The result after applying the numerical algorithm to the differential equation is a probability density at each time instant. For models with one fluid place this density is supported on one continuous dimension and for models with two fluid places on a plane of two continuous dimensions. The means over the continuous dimensions, the state probabilities and for the two-dimensional models the marginal densities have been regarded.

The accuracy of the discretization scheme has been studied by using the first model. It could be seen that by choosing larger stepsizes in space a i considerable error is caused, while a larger stepsize in time did show hardly any negative effect. The insight we gained from this study is that if a solution process shall be speeded up a larger stepsize in time might be more advisable than taking larger stepsizes in space. Although, this might not hold for all models.

With the second model we discussed the issue of boundary conditions as the commonly used reflecting boundaries limits the continuous variable, which might not be the desired model behaviour. Transparent boundary conditions are able to let probability mass go through and give an approximation to the exact, unbounded differential equation. The boundary conditions are, however, still work in progress.

Acknowledgement

The first author would like to thank Aad van Moorsel for many fruitful discussions.

References

[ABCD95] M. Ajmone Marsan, G. Balbo, G. Conte, and S. Donatelli. *Modelling with Generalized Stochastic Petri Nets.* Series in Parallel Computing. John Wiley & Sons, 1995.

[BGG+99] A. Bobbio, S. Garg, M. Gribaudo, A. Horváth, M. Sereno, and M. Telek. Modeling Software Systems with Rejuvenation, Restoration and Checkpointing through Fluid Stochastic Petri Nets. In *Proc. Eighth International Workshop on Petri Nets and Performance Models - PNPM'99*, pages 82–91, Zaragoza, Spain, 1999.

[CM65] D.R. Cox and H.D. Miller. *The Theory of Stochastic Processes.* Chapman and Hall, 1965.

[DA93] R. David and H. Alla. Autonomous and Timed Continuous Petri nets. In G. Rozenberg, editor, *Advances in Petri Nets 1993*, volume 674 of *Lecture Notes in Computer Science*, pages 71–90. Springer-Verlag, 1993.

[GSB99] M. Gribaudo, M. Sereno, and A. Bobbio. Fluid Stochastic Petri Nets: An Extended Formalism to Include non-Markovian Models. In *Proc. Eighth International Workshop on Petri Nets and Performance Models - PNPM'99*, pages 74–81, Zaragoza, Spain, 1999.

[HKNT98] G. Horton, V. G. Kulkarni, D. M. Nicol, and K. S. Trivedi. Fluid Stochastic Petri Nets: Theory, Applications and Solution. *European Journal of Operations Research*, 105(1):184–201, February 1998.

[Hor96] G. Horton. Computation of the Distribution of Accumulated Reward with Fluid Stochastic Petri nets. In *Proc. 2nd IEEE Int. Computer Performance and Dependability Symposium (IPDS)*, pages 90–95, Urbana-Champaign, Illinois, USA, 4-6 September 1996. IEEE-CS Press.

[Kle76] L. Kleinrock. *Queueing Systems Vol II: Computer Applications.* John Wiley, 1976.

[Kob74] H. Kobayashi. Application of the Diffusion Approximation to Queueing Networks II: Nonequilibrium Distributions and Applications to Computer Modeling. *Journal of the ACM*, 21(3):459–469, July 1974.

[Mey82] J. F. Meyer. Closed-Form Solutions of Performability. *IEEE Transactions on Computers*, C-31(7):648–657, July 1982.

[New71] G. F. Newell. *Applications of Queueing Theory.* Chapman and Hall, Ltd. (London), 1971.

[SL94] D. E. Stewart and Z. Leyk. Meschach: Matrix Computations in C. In *Proceedings of the Centre for Mathematics and its Applications*, volume 32, Australian National University, 1994.

[SM91] W. H. Sanders and J. F. Meyer. A Unified Approach for Specifying Measures of Performance, Dependability, and Performability. In A. Avizienis and J. Laprie, editors, *Dependable Computing for Critical Applications*, volume 4 of *Dependable Computing and Fault-Tolerant Systems*, pages 215–237. Springer, 1991.

[Str89] J. C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations.* Chapmann & Hall, 1989.

[Tak55] L. Takács. Investigation of waiting time problems by reduction to Markov processes. *Acta Math. Acad. Sci. Hungar.*, 6:101–129, 1955.

[TK93] K. S. Trivedi and V. G. Kulkarni. FSPNs: Fluid Stochastic Petri Nets. In *Proc. 14th Int. Conf. on the Application and Theory of Petri Nets*, pages 24–31, Chicago, 1993.

[TR99] M. Telek and S. Rácz. Numerical analysis of large Markov Reward Models. *Performace Evaluation*, 36&37:95–114, August 1999.

[Wol97] K. Wolter. Second Order Fluid Stochastic Petri Nets: an Extension of GSPNs for Approximate and Continuous modelling. In *Proc. 1st World Congress on Systems Simulation (WCSS'97)*, pages 328–332, Singapore, Sept. 1–3 1997.

[Wol99] K. Wolter. Jump Transitions in Second Order FSPNs. In *Proc. 7th Int. Symp. on Modelling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'99)*, pages 156–163, Washington DC, USA, October 1999. IEEE-CS Press.

[WZ99] K. Wolter and A. Zisowsky. On Markov Reward Modelling with Second Order FSPNs. In J. T. Bradley and N. J. Davies, editors, *Proc. 15th annual UK Performance Engineering Workshop*, pages 167–177, Bristol, UK, July 22-23 1999. TR CSTR-99-007.

[Zis98] A. Zisowsky. Entwurf und Implementierung eines Verfahrens für die transiente Analyse fluider stochastischer Petri-Netze. Master's thesis, TU Berlin, August 1998.