Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Measurement and Information Systems

Configurable Stochastic Analysis Framework for Asynchronous Systems

Scientific Students' Associations Report

Authors:

Attila Klenik
Kristóf Marussy

Supervisors:

dr. Miklós Telek
Vince Molnár
András Vörös

2015.

Contents

Összefoglaló A kritikus rendszerek – biztonságkritikus, elosztott és felhőalkalmazások – helyességének biztosításához szükséges a funkcionális és nemfunkcionális követelmények matematikai igényességű ellenőrzése. Számos, szolgáltatásbiztonsággal és teljesítményvizsgálattal kapcsolatos tipikus kérdés általában sztochasztikus analízis segítségével válaszolható meg.

A kritikus rendszerek elosztott és aszinkron tulajdonságai az *állapottér robbanás* jelenségéhez vezetnek. Emiatt méretük és komplexitásuk gyakran megakadályozza a sikeres sztochasztikus analízist, melynek számításigénye nagyban függ a lehetséges viselkedések számától. A modellek komponenseinek jellegzetes időbeli viselkedése a számításigény további jelentős növekedését okozhatja.

A szolgáltatásbiztonsági és teljesítményjellemzők kiszámítása markovi modellek állandósult állapotbeli és tranziens megoldását igényli. Számos eljárás ismert ezen problémák kezelésére, melyek eltérő reprezentációkat és numerikus algoritmusokat alkalmaznak; ám a modellek változatos tulajdonságai miatt nem választható ki olyan eljárás, mely minden esetben hatékony lenne.

A markovi analízishez szükséges a modell lehetséges viselkedéseinek, azaz állapotterének felderítése, illetve tárolása, mely szimbolikus módszerekkel hatékonyan végezhető el. Ezzen szemben a sztochasztikus algoritmusokban használt vektor- és indexműveletek szimbolikus megvalósítása nehézkes. Munkánk célja egy olyan, integrált keretrendszer fejlesztése, mely lehetővé teszi a komplex sztochasztikus rendszerek kezelését a szimbolikus módszerek, hatékony mátrix reprezentációk és numerikus algoritmusok előnyeinek ötvözésével.

Egy teljesen szimbolikus algoritmust javasolunk a sztochasztikus viselkedéseket leíró mátrix-dekompozíciók előállítására a szimbolikus formában adott állapottérből kiindulva. Ez az eljárás lehetővé teszi a temporális logikai kifejezéseken alapuló szimbolikus technikák használatát.

A keretrendszerben megvalósítottuk a konfigurálható sztochasztikus analízist: megközelítésünk lehetővé teszi a különböző mátrix reprezentációk és numerikus algoritmusok kombinált használatát. Az implementált algoritmusokkal állandósult állapotbeli költség- és érzékenység analízis, tranziens költséganalízis és első hiba várható bekövetkezési idő analízis végezhető el sztochasztikus Petri-háló (SPN) markovi költségmodelleken. Az elkészített eszközt integráltuk a PetriDotNet modellező szoftverrel. Módszerünk gyakorlati alkalmazhatóságát szintetikus és ipari modelleken végzett mérésekkel igazoljuk.

Abstract Ensuring the correctness of critical systems – such as safety-critical, distributed and cloud applications – requires the rigorous analysis of the functional and extra-functional properties of the system. A large class of typical quantitative questions regarding dependability and performability are usually addressed by stochastic analysis.

Recent critical systems are often distributed/asynchronous, leading to the well-known phenomenon of *state space explosion*. The size and complexity of such systems often prevents the success of the analysis due to the high sensitivity to the number of possible behaviors. In addition, temporal characteristics of the components can easily lead to huge computational overhead.

Calculation of dependability and performability measures can be reduced to steady-state and transient solutions of Markovian models. Various approaches are known in the literature for these problems differing in the representation of the stochastic behavior of the models or in the applied numerical algorithms. The efficiency of these approaches are influenced by various characteristics of the models, therefore no single best approach is known.

The prerequisite of Markovian analysis is the exploration of the state space, i.e. the possible behaviors of the system. Symbolic approaches provide an efficient state space exploration and storage technique, however their application to support the vector operations and index manipulations extensively used by stochastic algorithms is cumbersome. The goal of our work is to introduce a framework that facilitates the analysis of complex, stochastic systems by combining together the advantages of symbolic algorithms, compact matrix representations and various numerical algorithms.

We propose a fully symbolic method to explore and describe the stochastic behaviors. A new algorithm is introduced to transform the symbolic state space representation into a decomposed linear algebraic representation. This approach allows leveraging existing symbolic techniques, such as the specification of properties with *Computational Tree Logic* (CTL) expressions.

The framework provides configurable stochastic analysis: an approach is introduced to combine the different matrix representations with numerical solution algorithms. Various algorithms are implemented for steady-state reward and sensitivity analysis, transient reward analysis and mean-time-to-first-failure analysis of stochastic models in the *Stochastic Petri Net* (SPN) Markov reward model formalism. The analysis tool is integrated into the PetriDotNet modeling application. Benchmarks and industrial case studies are used to evaluate the applicability of our approach.

Chapter 1

Introduction

Árvíztűrő tükörfúrógép

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Chapter 2

Background

2.1 Petri nets

Petri nets are a widely used graphical and mathematical modeling tool for systems which are concurrent, asynchronous, distributed, parallel or nondeterministic.

Definition 2.1 A *Petri net* is a 5-tuple $PN = (P, T, F, W, M_0)$, where

- $P = \{p_0, p_1, \dots, p_{n-1}\}$ is a finite set of places;
- $T = \{t_0, t_1, \dots, t_{m-1}\}$ is a finite set of transitions;
- $F \subseteq (P \times T) \cup (P \times T)$ is a set of arcs, also called the flow relation;
- $W : F \to \mathbb{N}^+$ is an arc weight function;
- $M_0 : P \to \mathbb{N}$ is the initial marking;
- $P \cap T = \emptyset$ and $P \cup T \neq \emptyset$ [7].

Arcs from P to T are called *input arcs*. The input places of a transition t are denoted by ${}^\bullet t = \{p : (p, t) \in F\}$. In contrast, arcs of the form (t, p) are called *output arcs* and the output places of t are denoted by $t^\bullet = \{p : (t, p) \in F\}$.

A *marking* $M : P \to \mathbb{N}$ assigns a number of *tokens* to each place. The transition t is *enabled* in the marking M_1 (written as $M_1 [t\rangle$) when $M(p) \geq W(p, t)$ for all $p \in {}^\bullet t$.

Petri nets are graphically represented as edge weighted directed bipartite graphs. Places are drawn as circles, while transitions are drawn as rules or rectangles. Edge weights of 1 are ususally omitted from presentation. Dots on places correspond to tokens in the current marking.

If $M_1 [t\rangle$ the transition t can be *fired* to get a new marking M_2 (written as $M_1 [t\rangle M_2$) by decreasing the token counts for each place $p \in {}^\bullet t$ by $W(p, t)$ and increasing the token counts for each place $p \in t^\bullet$ by $W(t, p)$. Note that in general, ${}^\bullet t$ and $t^\bullet$ need not be disjoint. Thus, the firing rule can be written as

$$M_2(p) = M_1(p) - W(p, t) + W(t, p), \tag{2.1}$$

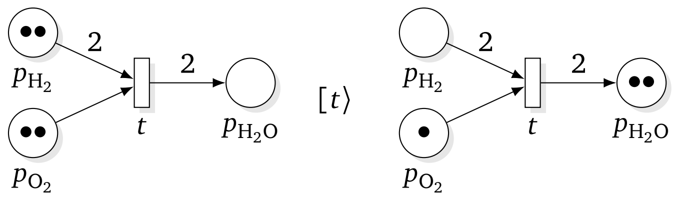


Figure 2.1 A Petri net model of the reaction of hydrogen and oxygen.

where we take $W(x, y) = 0$ if $(x, y) \notin F$ for brevity.

A marking M' is *reachable* from the marking M (written as $M \rightsquigarrow M'$) if there exists a sequence of markings and transitions for some finite k such that

$$M_1 \, [t_{i_1}\rangle \, M_2 \, [t_{i_2}\rangle \, M_3 \, [t_{i_3}\rangle \cdots [t_{i_{k-1}}\rangle \, M_{k-1} \, [t_{i_k}\rangle \, M_k,$$

where $M_1 = M$ and $M_k = M'$. A marking M is in the reachable *state space* of the net if $M_0 \rightsquigarrow M$. The set of all markings reachable from M_0 is denoted by

$$RS = \{M : M_0 \rightsquigarrow M\}.$$

Definition 2.2 The Petri net PN is *k-bounded* if $M(p) \leq k$ for all $M \in RS$ and $p \in P$. PN is *bounded* if it is k-bounded for some (finite) k.

The reachable state space RS is finite precisely when the Peti net is bounded.

Example 2.1 The Petri net in Figure 2.1 models the chemical reaction

$$2\,H_2 + O_2 \to 2\,H_2O.$$

In the initial marking (left) there are two hydrogen and two oxygen molecules, represented by token on the places p_{H_2} and p_{O_2}, therefore the transition t is enabled. Firing t yields the marking on the right where the two tokens on p_{H_2O} are the reaction products. Now t is no longer enabled.

Running example 2.2 In Figure 2.2 we introduce the *SharedResource* model which will serve as a running example throughout this report.

The model consists of a single shared resource S and two consumers. Each consumer can be in one of the C_i (calculating locally), W_i (waiting for resource) and S_i (shared working) states. The transitions r_i (request resource), a_i (acquire resource) and d_i (done) correspond to behaviors of the consumers. The net is 1-bounded, therefore it has finite RS.

The Petri net model allows the verification of safety properties, e.g. we can show that there is mutual exclusion – $M(S_1) + M(S_2) \leq 1$ for all reachable markings –

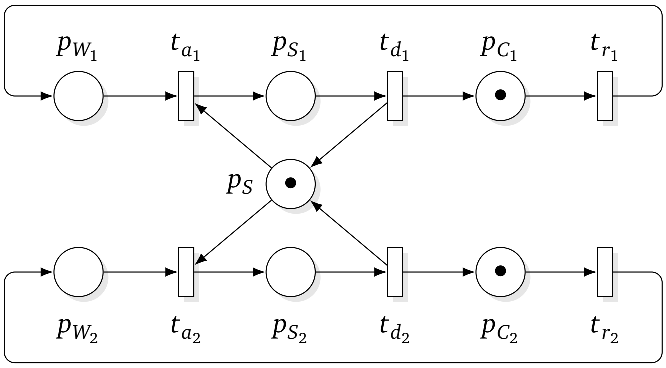


Figure 2.2 The *SharedResource* Petri net model.

or that deadlocks cannot occur. In contrast, we cannot compute dependability or performability measures (e.g. the utilization of the shared resource or number of calculations completed per unit time) because the model does not describe the temporal behavior of the system.

2.1.1 Petri nets extended with inhibitor arcs

One of the most frequently used extensions of Petri nets is the addition of inhibitor arcs, which modifies the rule for transition enablement. This modification gives Petri nets expressive power equivalent to Turing machines [4].

Definition 2.3 A *Petri net with inhibitor arcs* is a 3-tuple $PN_I = (PN, I, W_I)$, where

- $PN = (P, T, F, W, M_0)$ is a Petri net;
- $I \subseteq P \times T$ is the set of inhibitor arcs;
- $W_I : I \to \mathbb{N}^+$ is the inhibitor arc weight function.

Let $^\circ t = \{p : (p, t) \in I\}$ denote the set of inhibitor places of the transition t. The enablement rule for Petri nets with inhibitor arcs can be formalized as

$$M\,[t\rangle \iff M(p) \geq W(p,t) \text{ for all } p \in {}^\bullet t \text{ and } M(p) < W_I(p,t) \text{ for all } p \in {}^\circ t.$$

The firing rule (2.1) remains unchanged.

2.2 Continous-time Markov chains

Continous-time Markov chains are mathematical tools for describing the behavior of systems in countinous time where the random behavior of the system only depends on its current state.

Definition 2.4 A *Continous-time Markov Chain* (CTMC) $X(t) \in S, t \geq 0$ over a finite or countable infinite state space $S = \{0, 1, \ldots, n-1\}$ is a continous-time random process with the *Markovian* or memoryless property

$$\begin{aligned}\mathbb{P}(X(t_k) = x_k \mid X(t_{k-1}) = x_{k-1}, X(t_{k-2}) = x_{k-2}, \ldots, X(t_0) = x_0) \\ = \mathbb{P}(X(t_k) = x_k \mid X(t_{k-1}) = x_{k-1}),\end{aligned}$$

where $t_0 \leq t_1 \leq \cdots \leq t_k$. A CTMC is said to be *time-homogenous* if it also satisfies

$$\mathbb{P}(X(t_k) = x_k \mid X(t_{k-1}) = x_{k-1}) = \mathbb{P}(X(t_k - t_{k-1}) = x_k \mid X(0) = x_{k-1}).$$

In this report we will restrict our attention to time-homogenous CTMCs over finite state spaces. The state probabilities of these stochastic processes at time t form a finite-dimensional vector $\boldsymbol{\pi}(t) \in \mathbb{R}$,

$$\pi(t)[x] = \mathbb{P}(X(t) = x)$$

that satisfies the differential equation

$$\frac{\mathrm{d}\boldsymbol{\pi}(t)}{\mathrm{d}t} = \boldsymbol{\pi}(t) Q \tag{2.2}$$

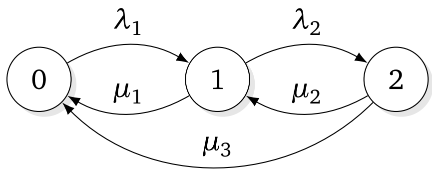
for some square matrix Q. The matrix Q is called the *infinitesimal generator matrix* of the CTMC and can be interpreted as follows:

- The diagonal elements $q[x, x] < 0$ describe the holding times of the CTMC. If $X(t) = x$, the *holding time* $h_x = \inf\{h > 0 : X(t + h) \neq x\}$ spent in state x is exponentially distributed with rate $\lambda_x = -q[x, x]$. If $q[x, x] = 0$, the no transitions are possible form state x and it is said to be *absorbing*.
- The off-diagonal elements $q[x, y] \geq 0$ describe the state transitions. In state x the CTMC will jump to state y at the next state transition with probability $-q[x, y]/q[x, x]$. Equivalently, there is expontentially distributed countdown in the state x for each $y : q[x, y] > 0$ with *transition rate* $\lambda_{xy} = q[x, y]$. The first countdown to finish will trigger a state change to the corresponding state y. Thus, the CTMC is a Kripke structure with exponentially distributed timed transitions.
- Elements in each row of Q sum to 0, hence it satisfies $Q\mathbf{1}^{\mathrm{T}} = \mathbf{0}^{\mathrm{T}}$.

For more algebraic properties of infinitesimal generator matrices, we refer to Plemmons and Berman [8] and Stewart [9].

A state y is said to be *reachable* from the state x ($x \rightsquigarrow y$) if there exists a sequence of states

$$x = z_1, z_2, z_3, \ldots, z_{k-1}, z_k = y$$



$$Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ \begin{pmatrix} -\lambda_1 & \lambda_1 & 0 \\ \mu_1 & -\lambda_2-\mu_1 & \lambda_2 \\ \mu_3 & \mu_2 & -\mu_2-\mu_3 \end{pmatrix} \end{array}$$

Figure 2.3 Example CTMC with 3 states and its generator matrix.

such that $q[z_i, z_{i+1}] > 0$ for all $i = 1, 2, \ldots, k-1$. If y is reachable from x for all $x, y \in S$ y, the Markov chain is said to be *irreducible*. Equivalenty, Q is the infinitesimal generator matrix of an irreducible CTMC if there is no permutation matrix M such that

$$M^{-1}QM = \begin{pmatrix} Q_1 & 0 \\ 0 & Q_2 \end{pmatrix}$$

for some square matrices Q_1, Q_2.

The *steady-state probability distribution* $\boldsymbol{\pi} = \lim_{t\to\infty} \boldsymbol{\pi}(t)$ exists and is independent from the *initial distribution* $\boldsymbol{\pi}(0) = \boldsymbol{\pi}_0$ if and only if the finite CTMC is irreducible. The steady-state distribution is a stationary solution of eq. (2.2), therefore it satisfies the linear equation

$$\frac{\mathrm{d}\boldsymbol{\pi}}{\mathrm{d}t} = \boldsymbol{\pi} Q = \mathbf{0}. \tag{2.3}$$

Example 2.3 Figure 2.3 shows a CTMC with 3 states. The transitions from state 0 to 1 and from 1 to 2 are associated with exponentially distributed countdowns with rates λ_1 and λ_2 respectively, while transitions in the reverse direction have rates μ_1 and μ_2. The transition form state 2 to 0 is also possible with rate μ_3.

The rows (corresponding to source states) and columns (destination states) of the infinitesimal generator matrix Q are labeled with the state numbers. The diagonal element $q[1,1]$ is $-\lambda_2 - \mu_1$, hence the holding time in state 1 is exponentially distributed with rate $\lambda_2 + \mu_1$. The transition to 0 is taken with probability $-q[1,0]/q[1,1] = \mu_1/(\lambda_2 + \mu_1)$, while the transition to 2 is taken with probability $\lambda_2/(\lambda_2 + \mu_1)$.

The CTMC is irreducible, because every state is reachable from every other state. Therefore, there is a unique steady-state distribution $\boldsymbol{\pi}$ independent from the initial distribution $\boldsymbol{\pi}_0$.

2.2.1 Markov reward models

Continous-time Markov chains may be employed in the estimation of performance measures of models by defining *rewards* that associate *reward rates* with the states of a CTMC. The momentary reward rate random variable $R(t)$ can describe performance measures defined at a single point of time, such as resource utilization or probability of failure, while the *accumulated reward* random variable $Y(t)$ may correspond to performance measures associated intervals of time, such as total downtime.

Definition 2.5 A *Continous-time Markov Reward Process* over a finite state space $S = \{0, 2, \dots, n-1\}$ is a pair $(X(t), \mathbf{r})$, where $X(t)$ is a CTMC over S and $\mathbf{r} \in \mathbb{R}^n$ is a *reward rate vector*.

The element $r[x]$ of reward vector is a momentary reward rates in state x, therefore the reward rate random variable can be written as $R(t) = r[X(t)]$. Accumulated rewards reward until time t is calculated by integration as

$$Y(t) = \int_0^t R(\tau)\,\mathrm{d}\tau.$$

TODO Valamit írni és hivatkozni arról, hogy az $R(t)$ és $Y(t)$ eloszlásait nehéz meghatározni, mi csak a várható értékekkel foglalkozunk + behivatkozni valamit a várható értékhez.

Given the initial probability distribution vector $\boldsymbol{\pi}(0) = \boldsymbol{\pi}_0$ the expected value of the reward rate at time t can be calculated as

$$\mathbb{E}\,R(t) = \sum_{i=0}^{n-1} \pi(t)[i] r[i] = \boldsymbol{\pi}(t)\,\mathbf{r}^\mathrm{T}, \tag{2.4}$$

which requires the solution of the initial value problem

$$\frac{\mathrm{d}\boldsymbol{\pi}(t)}{\mathrm{d}t} = \boldsymbol{\pi}(t)\,Q, \quad \boldsymbol{\pi}(0) = \boldsymbol{\pi}_0$$

to from the inner product $\mathbb{E}[R(t)] = \boldsymbol{\pi}(t)\,\mathbf{r}^\mathrm{T}$. To obtain the expected steady-state reward rate (if it exists) the linear equation (2.3) should be solved instead for the steady-state probability vector $\boldsymbol{\pi}$.

The expected value of the accumulated reward is

$$\begin{aligned} \mathbb{E}\,Y(t) &= \mathbb{E}\left[\int_0^t R(\tau)\,\mathrm{d}\tau\right] = \int_0^t \mathbb{E}[R(\tau)]\,\mathrm{d}\tau \\ &= \int_0^t \sum_{i=0}^{n-1} \pi(\tau)[i] r[i]\,\mathrm{d}\tau = \sum_{i=0}^{n-1} \int_0^t \pi(\tau)[i]\,\mathrm{d}\tau\, r[i] \end{aligned}$$

$$= \int_0^t \boldsymbol{\pi}(t)\,\mathrm{d}\tau\,\mathbf{r}^{\mathrm{T}} = \mathbf{L}(t)\,\mathbf{r}^{\mathrm{T}},$$

where $\mathbf{L}(t) = \int_0^t \boldsymbol{\pi}(t)\,\mathrm{d}\tau$ is the accumulated probability vector, which is the solution of the initial value problem

$$\frac{\mathrm{d}\mathbf{L}(t)}{\mathrm{d}t} = \boldsymbol{\pi}(t), \quad \frac{\mathrm{d}\boldsymbol{\pi}(t)}{\mathrm{d}t} = \boldsymbol{\pi}(t)\,Q, \quad \mathbf{L}(0) = \mathbf{0}, \quad \boldsymbol{\pi}(0) = \boldsymbol{\pi}_0.$$

Example 2.4 Let c_0, c_1 and c_2 denote operating costs per unit time associated with the states of the CTMC in Figure 2.3. Consider the Markov reward process $(X(t), \mathbf{r})$ with reward rate vector

$$\mathbf{r} = \begin{pmatrix} c_0 & c_1 & c_2 \end{pmatrix}.$$

The random variable $R(t)$ describes the momentary operating cost, while $Y(t)$ is the total operating expenditure until time t. The steady-state expectation of R is the average maintenance cost per unit time of the long-running system.

2.2.2 Sensitivty

Consider a reward process $(X(t), \mathbf{r})$ where both the infinitesimal generator matrix $Q(\boldsymbol{\theta})$ and the reward rate vector $\mathbf{r}(\boldsymbol{\theta})$ may depend on some *parameters* $\boldsymbol{\theta} \in \mathbb{R}^m$. The *sensitivity* analysis of the rewards $R(t)$ may reveal performance or reliability bottlenecks of the modeled system and aid designers in achieving desired performance measures.

Definition 2.6 The *sensitivity* of the expected reward rate $\mathbb{E}R(t)$ to the parameter $\theta[i]$ is the partial derivative

$$\frac{\partial\, \mathbb{E}R(t)}{\partial\, \theta[i]}.$$

The model reacts to the change of parameters with high absolute sensitivity more prominently, therefore they can be promising avenues of system optimization.

To calculate the sensivity of $\mathbb{E}R(t)$, the partial derivative of both sides of eq. (2.4) is taken, yielding

$$\frac{\partial\, \mathbb{E}R(t)}{\partial\, \theta[i]} = \frac{\partial\, \boldsymbol{\pi}(t)}{\partial\, \theta[i]}\mathbf{r}^{\mathrm{T}} + \boldsymbol{\pi}(t)\left(\frac{\partial\, \mathbf{r}}{\partial\, \theta[i]}\right)^{\mathrm{T}} = \mathbf{s}_i(t)\,\mathbf{r}^{\mathrm{T}} + \boldsymbol{\pi}(t)\left(\frac{\partial\, \mathbf{r}}{\partial\, \theta[i]}\right)^{\mathrm{T}},$$

where $\mathbf{s}_i$ is the sensitivity of $\boldsymbol{\pi}$ to the parameter $\theta[i]$.

In transient analysis, the sensitivity vector $\mathbf{s}_i$ is the solution of the initial value problem

$$\frac{\mathrm{d}\mathbf{s}_i(t)}{\mathrm{d}t} = \mathbf{s}_i(t)Q + \pi(t)V_i, \quad \frac{\mathrm{d}\boldsymbol{\pi}(t)}{\mathrm{d}t} = \boldsymbol{\pi}_i(t)Q, \quad \mathbf{s}_i(0) = \mathbf{0}, \quad \boldsymbol{\pi}(0) = \boldsymbol{\pi}_0,$$

where $V_i = \partial Q(\theta)/\partial \theta[i]$ is the partial derivative of the generator matrix. A similar initial value problem can be derived for the sensitivity of $\mathbf{L}(t)$ [1].

To obtain the sensitivity $\mathbf{s}_i$ of the steady-state probability vector $\boldsymbol{\pi}$, the system of linear equations

$$\mathbf{s}_i Q = -\boldsymbol{\pi} V_i, \quad \mathbf{s}_i \mathbf{1}^T = 0$$

is solved.

Another type of sensitivty analysis considers *unstructured* small perturbations of the infinitesimal generator matrix Q instead of dependecies on parameters. This latter, unstructured analysis may be used to study the numerical stability and conditioning of the solutions of the Markov chain.

2.2.3 Time to first failure

Let $D \subsetneq S$ be a set of *failure states* of the CTMC $X(t)$ and $U = S \setminus D$ be a set of operating states. We will assume without loss of generality that $U = \{0, 1, \dots, n_U - 1\}$ and $D = \{n_U, n_U + 1, \dots, n - 1\}$.

The matrix

$$Q_{UD} = \begin{pmatrix} Q_{UU} & \mathbf{q}_{UD}^{\mathrm{T}} \\ \mathbf{0} & 0 \end{pmatrix}$$

is the generator matrix of a CTMC $X_{UD}(t)$ in which all the failures states D were merged into a single state n_U and all outgoing transitions from D were removed. The matrix Q_{UU} is the $n_U \times n_U$ upper left submatrix of Q, while the vector $\mathbf{q}_{UD} \in \mathbb{R}^{n_U}$ is defined as

$$q_{UD}[x] = \sum_{y \in D} q[x, y].$$

If the initial distribution $\boldsymbol{\pi}_0$ is 0 for all failure states (i.e. $\pi_0[x] = 0$ for all $x \in D$), the *Time to First Failure*

$$TFF = \inf\{t \geq 0 : X(t) \in D\} = \inf\{t \geq 0 : X_{UD}(t) = n_U\}$$

has *phase-type distribution* with parameters $(Q_{UU}, \mathbf{q}_{UD}, \boldsymbol{\pi}_U)$, where $\boldsymbol{\pi}_U$ is the vector containing the first n_U elements of $\boldsymbol{\pi}_0$. In particular, the *Mean Time to First Failure* is

$$MTFF = \mathbb{E}[TFF] = -\pi_D Q_{UU}^{-1} \mathbf{1}^{\mathrm{T}}.$$

The probability of a y-mode failure ($y \in D$) is

$$\mathbb{P}(X(TFF + 0) = y) = -\pi_D Q_{UU}^{-1} \mathbf{q}_{Uy}^{\mathrm{T}},$$

where $\mathbf{q}_{Uy} \in \mathbb{R}^{n_U}$, $q_{Uy}[x] = Q[x, y]$ is the vector of transition rates from operation state to the failure state y. **TODO Hivatkozas a PH-eloszlashoz es a szamitasokhoz**.

2.3 Stochastic Petri nets

While reward processes based continous-time Markov chains allow the study of dependability or reliability measurements, the explicit specification of stochastic processes and rewards is often cumbersome. More expressive formalisms include queing networks, stochastic process algebras, Stochastic Automata Networks, Stochastic Activity Networks (SAN) and Stochastic Petri Nets (SPN). **TODO Kanonikus hivatkozasokat keresni**

Stochastic Petri Nets extend Petri nets by assigning random exponentially distributed random delays to transitions [6]. After the delay associated with an enabled transition is elapsed the transition fires *atomically* are transitions delays are reset.

Definition 2.7 A Stochastic Petri Net is a pair $SPN = (PN, \Lambda)$, where PN is a Petri net (P, T, F, W, M_0) and $\Lambda : P \to \mathbb{R}^+$ is a transition rate function.

Likewise, a stochastic Petri net with inhibitor arcs is a pair $SPN_I = (PN_I, \Lambda)$, where PN_I is a Petri net with inhibitor arcs.

A finite CTMC can be associated with a bounded stochastic Petri net (with inhibitor arcs) as follows:

1. The reachable state space of the Petri net is explored. We associate a consecutive natural numbers with the states such that the state space is
$$RS = \{M_0, M_1, M_2, \dots, M_{n-1}\},$$
where M_0 is the initial marking. From now on, we will use markings $M_x \in RS$ and natural numbers $x \in \{0, 1, \dots, n-1\}$ to refer to states of the model interchangably.

2. We define a CTMC $X(t)$ over the finite state space
$$S = \{0, 1, 2, \dots, n-1\}.$$
The initial distribution vector will be set to
$$\boldsymbol{\pi}(0) = \boldsymbol{\pi}_0 = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \end{pmatrix}$$
in the analysis steps ($\pi_0[x] = \delta_{0x}$).

3. The generator matrix $Q \in \mathbb{R}^{n \times n}$ encodes the possible state transitions of the Petri net and the associated transition rate $\Lambda(\cdot)$ as
$$q_O[x, y] = \sum_{\substack{t \in T \\ M_x[t\rangle M_y}} \Lambda(t) \quad \text{if } x \neq y,$$
$$q_O[x, x] = 0,$$

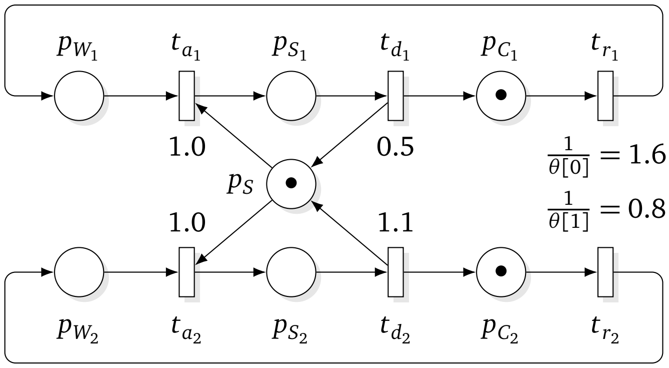


Figure 2.4 Example stochastic Petri net for the *SharedResource* model.

$RS =$

	S	C_1	W_1	S_1	C_2	W_2	S_2	
M_0	1	1	0	0	1	0	0	initial
M_1	1	0	1	0	1	0	0	client 1 waiting
M_2	1	1	0	0	0	1	0	client 2 waiting
M_3	1	0	1	0	0	1	0	1 waiting, 2 waiting
M_4	0	0	0	1	1	0	0	client 1 shared working
M_5	0	0	0	1	0	1	0	1 shared working, 2 waiting
M_6	0	1	0	0	0	0	1	client 2 shared working
M_7	0	0	1	0	0	0	1	1 waiting, 2 shared working

Table 2.1 Reachable state space of the *SharedResource* model.

$$Q = Q_O - \operatorname{diag}\{Q_O \mathbf{1}^\mathrm{T}\},$$

where the summation is done over all transition from the marking M_x to M_y, while Q_O and $Q_D = -\operatorname{diag}\{Q_O \mathbf{1}^\mathrm{T}\}$ are the off-diagonal and diagonal parts of Q, respectively.

Running example 2.5 Figure 2.4 shows SPN *SharedResouce* model, which is the Petri net from Figure 2.2 on page 5 extended with exponential transition rates.

The transitions a_1, d_1, a_2 and d_2 have rates 1.0, 0.5, 1.0 and 1.1, respectively. The parameter vector $\boldsymbol{\theta} = (0.625, 1.25) \in \mathbb{R}^2$ is introduced such that the transitions r_1 and r_2 have rates $1/\theta[0]$ and $1/\theta[1]$.

The reachable state space (Table 2.1) contains 8 markings which are mapped to the integers $S = \{0, 1, \ldots, 7\}$. The state space graph along with the transition rates of

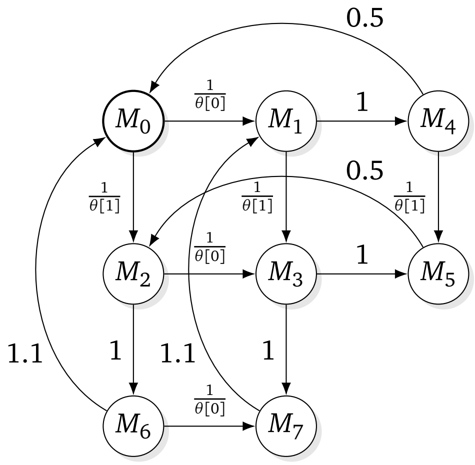


Figure 2.5 The CTMC associated with the *SharedResource* SPN model.

the CTMC is shown in Figure 2.5. The generator matrix is

$$Q = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{cccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{pmatrix} * & \frac{1}{\theta[0]} & \frac{1}{\theta[1]} & 0 & 0 & 0 & 0 & 0 \\ 0 & * & 0 & \frac{1}{\theta[1]} & 1 & 0 & 0 & 0 \\ 0 & 0 & * & \frac{1}{\theta[0]} & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & * & 0 & 1 & 0 & 1 \\ 0.5 & 0 & 0 & 0 & * & \frac{1}{\theta[1]} & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & * & 0 & 0 \\ 1.1 & 0 & 0 & 0 & 0 & 0 & * & \frac{1}{\theta[0]} \\ 0 & 1.1 & 0 & 0 & 0 & 0 & 0 & * \end{pmatrix} \end{array},$$

where in each row the diagonal element is the negative of the sum of the other elemens so that $Q\mathbf{1}^{\mathrm{T}} = \mathbf{0}^{\mathrm{T}}$. The CTMC is irreducible, therefore it has a well-defined steady-state distribution.

TODO GSPN-ekre hivatkozni, de csak roviden – vagy csak future workben?

2.3.1 Stochastic reward nets

Definition 2.8 A *Stochastic Reward Net* is a triple $SRN = (SPN, rr, ir)$, where SPN is a stochastic Petri net or a stochastic Petri net, $rr : \mathbb{N}^P \to \mathbb{R}$ is a *rate reward function* and $ir : T \times \mathbb{N}^P \to \mathbb{R}$ is an *impulse reward* function. A stochastic Reward net with inhibitor arcs is a triple $SRN_I = (SPN_I, rr, ir)$, where SPN_I is a stochastic Petri net with inhibitor arcs.

The rate reward $rr(M)$ is the reward gained per unit time in marking M, while $ir(t, M)$ is the reward gained when the transition t fires in marking M. The instantaneous reward rate and accumulated reward at time t is denoted by $R(t)$ and $Y(t)$, respectively.

If $ir(t, M) \equiv 0$, the SRN is equivalent to the Markov reward process $(X(t), \mathbf{r})$, where $X(t)$ is the CTMC associated with the stochastic Petri net and

$$\mathbf{r} \in \mathbb{R}^n, \quad r[x] = rr(M_x).$$

If there are impulse rewards, exact calculation of the expected reward rate $\mathbb{E} R(t)$ and expected accumulated reward $\mathbb{E} Y(t)$ can be performed on reward process $(X, \mathbf{r})$,

$$r[x] = rr(M_x) + \sum_{t \in T, M_x[t\rangle} \Lambda(t)\, ir(t, M_x),$$

where the summation is taken over all enabled transitions. In general, the distributions of $R(t)$ and $Y(t)$ cannot be derived by this method. **TODO Impulzus rewardok szamolasara cikkre hivatkozni?**

Running example 2.6 The SRN model

$$rr_1(M) = M(P_{S_1}) + M(P_{S_2}), \quad ir_1(t, M) = 0$$

describes the utilization of the shared resouce in the *SharedResouce* SPN (Figure 2.4 on page 12). $R_1(t) = 1$ if the resource is allocated, hence $\mathbb{E} R_1(t)$ is the probability that the resource is in use at time t, while $Y(t)$ is the total usage time until t.

Another reward structure

$$rr_2(M) = 0, \quad ir_2(t, M) = \begin{cases} 1 & \text{if } t \in \{t_{r_1}, t_{r_2}\}, \\ 0 & \text{otherwise} \end{cases}$$

counts the completed calculations, which are modeled by tokens leaving the places C_1 and C_2. The exprected steady-state reward rate $\lim_{t\to\infty} \mathbb{E} R(t)$ equals the number of calculations per unit time in a long-running system, while $Y(t)$ is the number of calculations performed until time t.

The reward vectors associated with these SRNs are

$$\begin{array}{rcccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \mathbf{r}_1 = \big(& 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \big), \\ \mathbf{r}_2 = \big(& \frac{1}{\theta[0]} + \frac{1}{\theta[1]} & \frac{1}{\theta[1]} & \frac{1}{\theta[0]} & 0 & \frac{1}{\theta[1]} & 0 & \frac{1}{\theta[0]} & 0 \big). \end{array}$$

2.3.2 Superposed stochastic Petri nets

Definition 2.9 A *Superposed Stochastic Petri Net* (SSPN) is a pair $SSPN = (SPN, \mathscr{P})$, where $\mathscr{P} = \{P^{(0)}, P^{(1)}, \ldots, P^{(J-1)}\}$ is partition of the set of places $P = P^{(0)} \cup P^{(1)} \cup \cdots \cup P^{(J-1)}$ [5]. Superposed stochastic Petri nets with inhibitor arcs $SSPN_I = (SPN_I, \mathscr{P})$ are defined analogously.

The jth *local net* $LN^{(j)} = ((P^{(j)}, T^{(j)} = T_L^{(j)} \cup T_S^{(j)}, F^{(j)}, W^{(j)}, M_0^{(j)}), \Lambda^{(j)})$ can be constructed as follows:

- $P^{(j)}$ is the corresponding set from the partition of the original net.
- $T^{(j)}$ contains the local transition $T_L^{(j)}$ and synchronizing transitions $T_S^{(j)}$. A transition is *local* to $LN^{(j)}$ if it only affects places in $P^{(j)}$, that is,

$$T_L^{(j)} = \{t \in T : {}^\bullet t \cup t^\bullet \subseteq P^{(j)}\}. \tag{2.5}$$

  No transition may be local two more than one local net.

  A transition *synchronizes* with $LN^{(j)}$ if it affects some places in $P^{(j)}$ but it is not local to $LN^{(j)}$,

$$T_S^{(j)} = \{t \in T : ({}^\bullet t \cup t^\bullet) \cap P^{(j)} \neq \emptyset\} \setminus T_L^{(j)}. \tag{2.6}$$

- The relation $F^{(j)}$ and the functions $W^{(j)}$, $M_0^{(j)}$, $\Lambda^{(j)}$ are the appropriate restrictions of the original structures, $F^{(j)} = F \cap ((P^{(j)} \times T^{(j)}) \cup (T^{(j)} \times J^{(j)}))$, $W^{(j)} = W|_{F^{(j)}}$, $M_0^{(j)} = M_0|_{P^{(j)}}$, $\Lambda^{(j)} = M_0|_{T^{(j)}}$.

If there are inhibitor arcs in $SSPN_I$, inhibitor arcs must be considered when local net $LN_I^{(j)}$ is constructed. The set ${}^\bullet t \cup t^\bullet$ is replaced with ${}^\bullet t \cup t^\bullet \cup {}^\circ t$ in eqs. (2.5) and (2.6) so that the enablement of local transitions only depends on the marking of places in $P^{(j)}$ and only places in $P^{(j)}$ may be affected upon firing. In addition, the inhibitor arc relation and weight function are restricted as $I^{(j)} = I \cap (P^{(j)} \cap T^{(j)})$, $W_I^{(j)} = W_I|_{I^{(j)}}$.

Running example 2.7 Figure 2.6 shows a possible partitioning of the *Shared-Resource* SPN into a SSPN. The components $P^{(0)}$ and $P^{(1)}$ are the model the two consumers, while $P^{(2)}$ contains the unallocated resource S.

The transitions r_1 and r_2 are local to $LN^{(0)}$ and $LN^{(1)}$, respectively, while a_1, d_1, a_2 and d_2 synchronize with $LN^{(2)}$ and the local net associated with their consumers.

The *local reachable state space* $RS^{(j)}$ of $LN^{(j)}$ is the the set of markings beloning to the state space RS of the original net restricted to the places $P^{(j)}$ (duplicates removed),

$$RS^{(j)} = \{M^{(j)} : M \in RS, M^{(j)} = M|_{P^{(j)}}\}.$$

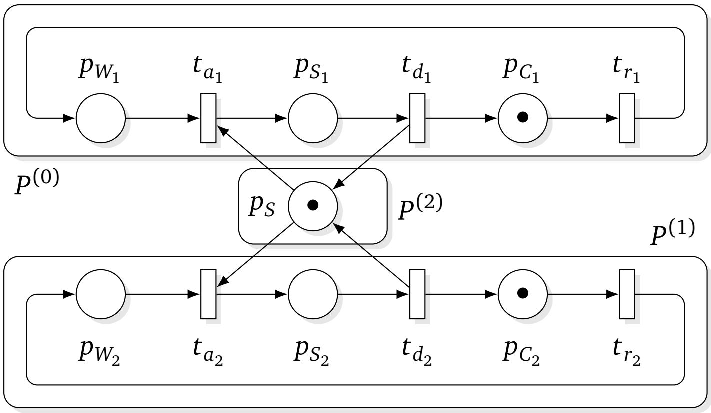


Figure 2.6 A partitioning of the *SharedResource* Petri net.

This is a *subset* of the reachable state space of $LN^{(j)}$, in particular, $LN^{(j)}$ is always finite if RS is finite, even if $LN^{(j)}$ is not bounded. Analysis techniques for generating local state spaces include *partial P-invariants* [3] and explicit projection of global reachable markings [2].

The *potential state space PS* of an SSPN is the Descares product of the local reachable state spaces of its components

$$PS = RS^{(0)} \times RS^{(1)} \times \cdots \times RS^{(J-1)},$$

which is a (possibly not proper) superset of the global reachable state space RS.

We will associate the natural numbers $S^{(j)} = \{0, 1, \dots, n_j - 1\}$ with the local reachable markings $RS^{(j)} = \{M_0, M_1, \dots, M_{n_j - 1}\}$ to aid the construction of Markov chains and use them interchangably. The notation

$$M = \mathbf{x} = (x^{(0)}, x^{(1)}, \dots, x^{(J-1)})$$

refers to the marking

$$M(p) = M^{(j)}_{x^{(j)}}(p), \quad \text{if } p \in P^{(j)},$$

which is the union of the local markings $M^{(0)}_{x^{(0)}}, M^{(1)}_{x^{(1)}}, \dots, M^{(J-1)}_{x^{(J-1)}}$.

Running example 2.8 The local reachable markings of the *SharedResource* SSPN are enumerated in Table 2.2.

The transitions d_1 and d_2 are always enabled in $LN^{(2)}$ because all their input places are located in other components, thus $LN^{(2)}$ is an unbounded Petri net. Despite this, $RS^{(2)}$ is finite, because it only contain the local markings which are reachable in the original net.

$$RS^{(0)} = \left\{ \begin{array}{c|ccc} & C_1 & W_1 & S_1 \\ \hline M_0^{(0)} & 1 & 0 & 0 \\ M_1^{(0)} & 0 & 1 & 0 \\ M_2^{(0)} & 0 & 0 & 1 \end{array} \right\},$$

$$RS^{(1)} = \left\{ \begin{array}{c|ccc} & C_2 & W_2 & S_2 \\ \hline M_0^{(1)} & 1 & 0 & 0 \\ M_1^{(1)} & 0 & 1 & 0 \\ M_2^{(1)} & 0 & 0 & 1 \end{array} \right\}, \quad RS^{(2)} = \left\{ \begin{array}{c|c} & S \\ \hline M_0^{(2)} & 1 \\ M_1^{(2)} & 1 \end{array} \right\}$$

Table 2.2 Local reachable markings of the *SharedResouce* SSPN from Figure 2.6.

The potential state space PS contains $3 \cdot 3 \cdot 2 = 18$ potential markings, although only 8 are reachable (Table 2.1 on page 12). For example, the marking $(2, 2, 0)$ is not reachable, as it would violate mutual exclusion.

2.4 Kronecker algebra

Definition 2.10 The *Kronecker product* of matrices $A \in \mathbb{R}^{n_1 \times m_1}$ and $B \in \mathbb{R}^{n_2 \times m_2}$ is the matrix $C = A \otimes B \in \mathbb{R}^{n_1 n_2 \times m_1 m_3}$, where

$$c[i_1 n_1 + i_2, j_1 m_1 + j_2] = a[i_1, j_1] b[i_2, j_2].$$

Some properties of the Kroncker product are

1. Associativity:
$$A \otimes (B \otimes C) = (A \otimes B) \otimes C,$$
which makes J-way Kronecker products $A^{(0)} \otimes A^{(1)} \otimes \cdots \otimes A^{(J-1)}$ well-defined.

2. Distributivity over matrix addition:
$$(A + B) \otimes (C + D) = A \otimes C + B \otimes C + A \otimes D + B \otimes D,$$

3. Compatibility with ordinary matrix multiplication:
$$(AB) \otimes (CD) = (A \otimes C)(B \otimes D),$$

in particular,

$$A \otimes B = (A \otimes I_2)(I_1 \otimes B)$$

for appropriately-sized identity matrices I_1 and I_2.

We will occasionally employ multi-index notation to refer to elements of Kronecker product matrices. For example, we will write

$$b[\mathbf{x},\mathbf{y}] = b[(x^{(0)}, x^{(1)}, \ldots, x^{(J-1)}), (y^{(0)}, y^{(1)}, \ldots, y^{(J-1)})] = \\ a^{(0)}[x^{(0)}, y^{(0)}]a^{(1)}[x^{(1)}, y^{(1)}] \cdots a^{(J-1)}[x^{(J-1)}, y^{(J-1)}],$$

where $\mathbf{x} = (x^{(0)}, x^{(1)}, \ldots, x^{(J-1)})$, $\mathbf{y} = (y^{(0)}, y^{(1)}, \ldots, y^{(J-1)})$ and B is the J-way Kronecker product $A^{(0)} \otimes A^{(1)} \otimes \cdots \otimes A^{(J-1)}$.

Definition 2.11 The *Kronecker sum* of matrices $A \in \mathbb{R}^{n_1 \times m_1}$ and $B \in \mathbb{R}^{n_2 \times m_2}$ is the matrix $C = A \oplus B \in \mathbb{R}^{n_1 n_2 \times m_1 m_3}$, where

$$C = A \otimes I_2 + I_1 \otimes B,$$

where $I_1 \in \mathbb{R}^{n_1 \times m_1}$ and $I_2 \in \mathbb{R}^{n_2 \times m_2}$ are identity matrices.

Example 2.9 Consider the matrices

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \qquad B = \begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}.$$

Their Kronecker product is

$$A \otimes B = \begin{pmatrix} 1\cdot 0 & 1\cdot 1 & 2\cdot 0 & 2\cdot 1 \\ 1\cdot 2 & 1\cdot 0 & 2\cdot 2 & 2\cdot 0 \\ 3\cdot 0 & 3\cdot 1 & 4\cdot 0 & 4\cdot 1 \\ 3\cdot 2 & 3\cdot 0 & 4\cdot 2 & 4\cdot 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 2 \\ 2 & 0 & 4 & 0 \\ 0 & 3 & 0 & 4 \\ 6 & 0 & 8 & 0 \end{pmatrix},$$

while their Kronecker sum is

$$A \oplus B = \begin{pmatrix} 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 \\ 3 & 0 & 4 & 0 \\ 0 & 3 & 0 & 4 \end{pmatrix} + \begin{pmatrix} 0 & 1 & 0 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 2 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 2 & 0 \\ 2 & 1 & 0 & 2 \\ 3 & 0 & 4 & 1 \\ 0 & 3 & 2 & 4 \end{pmatrix}.$$

Chapter 3

Stochastic analysis

3.1 Steady-state analysis

3.2 Transient analysis

3.2.1 Transient probability calculation

3.2.2 Accumulated probability calculation

3.3 Rewards and sensitivity

3.3.1 Stochastic reward nets

3.3.2 Sensitivity of rewards

Chapter 4

Efficient generation and storage of continous-time Markov chains

4.1 State-space exploration

4.1.1 Explicit state-space exploration

4.1.2 Symbolic methods

Multivalued decision diagrams

Edge-labeled decision diagrams

4.2 Storage of generator matrices

4.2.1 Explicit matrix storage

Dense matrices

Sparse matrices

Column major versus row major storage

4.2.2 Kronecker decomposition

4.2.3 Block Kronecker decomposition

4.3 Matrix composition

4.3.1 Generating sparse matrices from symbolic state spaces

4.3.2 Explicit block Kronecker decomposition

4.3.3 Symbolic block Kronecker decomposition

Chapter 5

Algorithms for stochastic analysis

5.1 Steady-state analysis

5.1.1 Explicit solution by LU decomposition

5.1.2 Stationary iterative methods

Power iteration

Jacobi iteration and Jacobi over-relaxation

Gauss–Seidel iteration and successive over-relaxation

5.1.3 Krylov subspace methods

Biconjugate gradient stabilized (BiCGSTAB)

5.2 Transient analysis

5.2.1 Uniformization

Calculation of uniformization weights

- Weights for transient probability with *trimming*

- Weights for accumulated probability

Steady-state detection

5.3 Processing results

5.3.1 Calculation of rewards

Symbolic storage of reward functions

5.3.2 Calculation of sensitivity

Sensitivity of state probabilities

Sensitivity of rewards

Chapter 6

Configurable stochastic analysis

6.1 Matrix storage and algorithm selection in practice

6.2 Implementation of configurable workflows

Chapter 7

Evaluation

7.1 Benchmark models

7.1.1 Synthetic models

Resource sharing

Kanban

Dining philosophers

7.1.2 Case studies

Performability of clouds

7.2 Baselines

7.2.1 PRISM

7.2.2 SMART

7.3 Results

Chapter 8

Conclusion

8.1 Future work

References

[1] James T. Blake, Andrew L. Reibman, and Kishor S. Trivedi. "Sensitivity Analysis of Reliability and Performability Measures for Multiprocessor Systems". In: *SIGMETRICS*. 1988, pp. 177–186. DOI: 10.1145/55595.55616.

[2] Peter Buchholz. "Hierarchical Structuring of Superposed GSPNs". In: *IEEE Trans. Software Eng.* 25.2 (1999), pp. 166–181. DOI: 10.1109/32.761443.

[3] Peter Buchholz and Peter Kemper. "On generating a hierarchy for GSPN analysis". In: *SIGMETRICS Performance Evaluation Review* 26.2 (1998), pp. 5–14. DOI: 10.1145/288197.288202.

[4] Piotr Chrzastowski-Wachtel. "Testing Undecidability of the Reachability in Petri Nets with the Help of 10th Hilbert Problem". In: *Application and Theory of Petri Nets 1999, 20th International Conference, ICATPN '99, Williamsburg, Virginia, USA, June 21-25, 1999, Proceedings*. Vol. 1639. Lecture Notes in Computer Science. Springer, 1999, pp. 268–281. DOI: 10.1007/3-540-48745-X_16.

[5] Susanna Donatelli. "Superposed Generalized Stochastic Petri Nets: Definition and Efficient Solution". In: *Application and Theory of Petri Nets 1994, 15th International Conference, Zaragoza, Spain, June 20-24, 1994, Proceedings*. Vol. 815. Lecture Notes in Computer Science. Springer, 1994, pp. 258–277. DOI: 10.1007/3-540-58152-9_15.

[6] Marco Ajmone Marsan. "Stochastic Petri nets: an elementary introduction". In: *Advances in Petri Nets 1989, covers the 9th European Workshop on Applications and Theory in Petri Nets, held in Venice, Italy in June 1988, selected papers*. Vol. 424. Lecture Notes in Computer Science. Springer, 1988, pp. 1–29. DOI: 10.1007/3-540-52494-0_23.

[7] Tadao Murata. "Petri nets: Properties, analysis and applications". In: *Proceedings of the IEEE* 77.4 (1989), pp. 541–580.

[8] RJ Plemmons and A Berman. *Nonnegative matrices in the mathematical sciences*. Academic Press, New York, 1979.

[9] Williams J Stewart. *Introduction to the numerical solutions of Markov chains*. Princeton Univ. Press, 1994.