
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
- $P = \{p_0, p_1, \ldots, p_{n-1}\}$ is a finite set of places;
- $T = \{t_0, t_1, \ldots, t_{m-1}\}$ 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 $P \cup T \neq \emptyset$ [4].

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

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

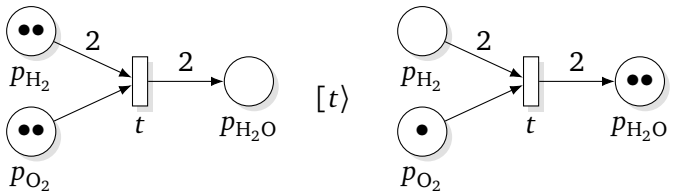



Figure 2.1: A Petri net model of the reaction of hydrogen and oxygen.

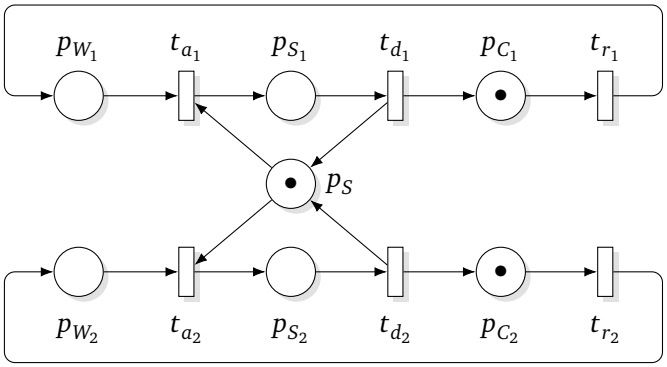



Figure 2.2: The *SharedResource* Petri net model.

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 $M_0 \rightsquigarrow M$.

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

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

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 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 – 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 [2].

**Definition 2.2** 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 2.3** 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

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

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*.

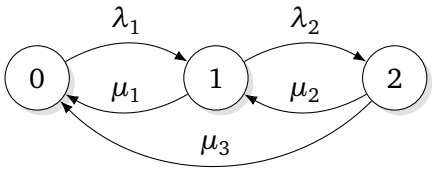
- The off-diagonal elements $q[x,y] \geq 0$ describe the state transitions. In state $x$ the CTMC will jump to state $y$ at the next state transition with probability $-q[x,y]/q[x,x]$. Equivalently, there is expontentially distributed countdown in the state $x$ for each $y : q[x,y] > 0$ with *transition 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 $Q\mathbf{1}^{\mathrm{T}} = 0$.

For more algebraic properties of infinitesimal generator matrices, we refer to **TODO Markov biblia** and **TODO Nonnegative matrices**.

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 $\lambda_1$ and $\lambda_2$ respectively, while transitions in the reverse direction have rates $\mu_1$ and $\mu_2$. The transition form state 2 to 0 is also possible with rate $\mu_3$.

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 2.4** A *Continous-time Markov Reward Process* over a finite state space $S = \{0, 2, \ldots, n-1\}$ 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 $Q(\theta)$ and the reward rate vector $\mathbf{r}(\theta)$ may depend on some *parameters* $\theta \in \mathbb{R}^m$. 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 2.5** 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 The sensitivity $\mathbf{s}_i$ of the steady-state probability vector $\boldsymbol{\pi}$, the system of linear equations

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

is solved.

**TODO Sensitivity masik definicioja, amikor a numerikus stabilitast vizsgaljuk...**

## 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 (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 [3]. After the delay associated with an enabled transition is elapsed the transition fires *atomically* are transitions delays are reset.

**Definition 2.6** A Stochastic Petri Net is a pair $SPN = (PN, \Lambda)$, where $PN = (P, T, F, W, M_0)$ is a Petri net and $\Lambda : P \to \mathbb{R}^+$ is a transition rate function.

### 2.3.1 Stochastic reward nets

## 2.4 Kronecker algebra

**Definition 2.7** 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:
$$A \otimes (B \otimes C) = (A \otimes B) \otimes C,$$
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 $I_2$.

*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

*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] 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.

[3] 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.

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