Discrete-Time Leap Method for Stochastic Simulation

Christian Rohr

Brandenburg University of Technology Cottbus-Senftenberg,
Chair of Data Structures and Software Dependability,
Postbox 10 13 44, D-03013 Cottbus, Germany,
christian.rohr@b-tu.de,
http://www-dssz.informatik.tu-cottbus.de

Abstract. In this paper, we present an approach to improve the efficiency of stochastic simulation for large and dense networks. We are using stochastic Petri nets as modelling framework. The underlying continuous-time Markov chain (CTMC) is converted to an equivalent discrete-time Markov chain (DTMC), this itself gains no efficiency. We improve the efficiency via discrete-time leaps, even though this results in an approximate method. The discrete-time leaps are done by applying the maximum firing rule, this reduces drastically the number of steps. The presented algorithm is implemented in our modelling and simulation tool Snoopy, as well as in our advanced analysis and model checking tool MARCIE. We demonstrate the approach on models of different sizes and complexities.

Keywords: discrete-time leap method, stochastic Petri net, approximate stochastic simulation, maximum firing rule, binomial distribution, weighted random shuffle

1 Introduction

Throughout the past decades typical biological models increased in their size and complexity, because of advances in systems and molecular biology, in particular through the high-throughput omic technologies. This progress gained momentum for the arise of multi-scale modelling [10]. Furthermore, Smallbone and Mendes raised issues of large-scale metabolic models that simulation methods have to take care of [19]: "...the inherent stiffness of genome-scale models. Since cells need to produce some metabolites (such as ATP) at a much higher rate than others (such as zymosterol), metabolic processes will necessarily be taking place at different timescales. As such, systems biology tools are needed that can robustly simulate models of this size and with these numerical instabilities." This all together demands for faster and more efficient simulation algorithms, even at the expense of greater approximation.

We present an approach to improve the efficiency of stochastic simulation for large and dense networks. We are using stochastic Petri nets as modelling framework, because of the well defined mathematical background and the nice graphical representation. The underlying continuous-time Markov chain (CTMC) is converted to an equivalent discrete-time Markov chain (DTMC), this itself gains no efficiency. We improve the efficiency via discrete-time leaps. The discrete-time leaps are done by applying the maximum firing rule, this reduces drastically the needed number of steps. The algorithm is called *discrete-time leap method* for the simulation of stochastic Petri nets.

We start by giving the mandatory definitions and continue with the presentation of the discrete-time leap method. Afterwards we demonstrate our approach on some case-studies of different sizes and complexities.

2 Preliminaries

A **Petri net** is a 4-tuple $\mathcal{PN} = (P, T, f, m_0)$ with a finite set of places $P = \{p_1, p_2, \dots, p_m\}$, defining the state of the net and a finite set of transitions $T = \{t_1, t_2, \dots, t_n\}$, defining the state changes in the net. The sets of places and transitions are disjoint and not empty. Places and transitions are connected via directed arcs and defined by the arc weight (multiplicity) function $f\colon ((P \times T) \cup (T \times P)) \to \mathbb{N}$. We define any marking m as a mapping $m\colon P \to \mathbb{N}_0$. It maps the set of places onto the set of natural numbers, where $m(p)$ defines the number of tokens in place $p \in P$. If appropriate and the context precludes confusion, we treat a place like a variable and write simply p instead of $m(p)$. An initial marking is denoted by m_0. The set of pre-places (input places) of transition t is defined as ${}^\bullet t = \{p \in P \mid f(p,t) > 0\}$. The set of post-places (output places) of transition t is defined as $t^\bullet = \{p \in P \mid f(t,p) > 0\}$.

A transition $t \in T$ is called *enabled* in marking m, if $\forall p \in {}^\bullet t\colon m(p) \geq f(p,t)$. If a transition $t \in T$ is enabled in m, it may fire. The firing of a transition t occurs in two parts. First, the transition removes the amount of tokens $f(p,t)$ from each of its pre-places ${}^\bullet t$. Second, it adds the amount of tokens $f(t,p)$ to each of its post-places $t^\bullet$. The change in the marking induced by firing transition t is denoted by

$$\Delta t = \{p \in {}^\bullet t \cup t^\bullet \colon \Delta t(p) = f(t,p) - f(p,t)\}. \tag{1}$$

When t in m fires, a new marking $m' = m + \Delta t$ is reached. This is denoted by $m \xrightarrow{t} m'$. The firing itself does not consume any time and takes place atomically. This defines the *standard firing rule*.

A transition t may still be enabled after firing once, i.e., it is possible to fire transition t in a sequence like this $m \xrightarrow{t} m' \xrightarrow{t} m'' \xrightarrow{t} \dots \xrightarrow{t} m^n$ until it is not enabled any more. The number of occurrences of transition t in such a sequence starting in marking m is named *enabledness (concurrency) degree* and is defined as

$$\mathrm{ed}_t = min_{p \in {}^\bullet t}\left(\left\lfloor \frac{m(p)}{f(p,t)} \right\rfloor\right). \tag{2}$$

The standard firing rule given above can be extended by the enabledness degree in the following way

$$m' = m + \Delta t \cdot \mathrm{ed}_t(m). \tag{3}$$

Usually referred to as *maximum (auto-concurrency)* firing rule.

A **stochastic Petri net** ($\mathcal{SPN}$) builds on $\mathcal{PN}$, but transitions have an exponentially distributed firing delay, characterised by the firing rate λ. A transition may lose its enabledness while waiting for the delay to expire. The firing itself does not consume time and follows the standard Petri net firing rule. The firing rates are typically transition-specific and marking-dependent. They are defined by stochastic firing rate functions, also known as propensity or hazard functions. The mapping $V : T \to H$, where H is the set of *hazard* functions, associates to each transition a function h_t from H. The sum of all transition rates in a marking m is denoted by $E(m) = \sum_{t \in T} h_t(m)$.

The semantics of a $\mathcal{SPN}$ is a **continuous time Markov chain** (CTMC). It is a stochastic process with an exponential probability distribution that has the *Markov property*. That means, its future behaviour depends only on its current state and not on former behaviour. The definition of a CTMC is comparable with the reachability graph, except that the arcs are labelled with the stochastic firing rate of the transition. A CTMC of a $\mathcal{SPN}$ is a tuple $\text{CTMC}_{\mathcal{SPN}} = (\mathcal{R}_{SPN}(m_0), \mathbf{Q}, m_0)$ with $\mathcal{R}_{SPN}(m_0)$ denoting the state space of the underlying net, the transition rate matrix $\mathbf{Q} = \mathcal{R}_{SPN}(m_0) \times \mathcal{R}_{SPN}(m_0) \to \mathbb{R}_0^+$ and m_0 the initial state. Now, the probability of a transition t enabled in state m to fire (which results in state m') within τ time units is

$$1 - e^{-\mathbf{Q}(m,m')\cdot\tau}. \tag{4}$$

When working with biological systems modelled as $\mathcal{SPN}$, **stochastic simulation** is a suitable analysis technique. Although in principle known a long time before, Gillespie was the first who developed a supporting theory for stochastic simulation of chemical kinetics [8,7]. He presented the stochastic simulation algorithm (SSA). Basically, it performs the following steps:

Starting from the initial marking m_0, one has to repeatedly fire transitions. In order to fire a transition, one must answer two questions:

1. When will the next transition fire?
2. Which transition will fire next?

The enabled transitions in the net compete in a race condition. The fastest one determines the next marking and the simulation time elapses. In the new marking, the race condition starts anew.

The SSA creates a sequence of discrete random variables $X(\tau)$. The discrete random variable $X_p(\tau)$ describes the number of tokens on place $p \in P$ at time τ. The system state (marking) at time τ is thus a discrete n-dimensional random vector $X(\tau) = (X_{p_1}(\tau), \ldots, X_{p_n}(\tau)) \in \mathcal{X}$. The time $\Delta\tau$ to the next transition is an exponentially distributed random variable with mean $1/E(m)$; the probability density function (pdf) is

$$P(\Delta\tau \mid m) = E(m) \cdot e^{-E(m)\cdot\Delta\tau}. \tag{5}$$

The next transition to fire is a discrete random variable with probability mass function (pmf):

$$P(t \mid m) = h_t(m)/E(m). \tag{6}$$

Given the system is in state $X(\tau)$, the probability that a transition $t \in T$ will occur in the time interval $[\tau, \tau + \Delta\tau)$ is given by:

$$\begin{aligned} P(\Delta\tau, t \mid m) &= E(m) \cdot e^{-E(m)\cdot\Delta\tau} \cdot h_t(m)/E(m) \\ &= h_t(m) \cdot e^{-E(m)\cdot\Delta\tau}. \end{aligned} \tag{7}$$

3 Discrete-time Leap Method

The discrete-time leap method (δ-leaping for short) aims at the simulation of stochastic Petri nets used for modelling biochemical reaction networks. In an attempt to decrease the time complexity of the simulation algorithm, we exploit the uniformization of the underlying CTMC in combination with the maximum firing rule.

The idea of converting a CTMC into a DTMC goes back to [14]. Similar methods are known as *uniformization* or *randomization*, see [20]. The DTMC is defined stochastically identical to the CTMC, i.e., the original CTMC is represented by a DTMC where the times are implicitly driven by a Poisson process. It can be shown that this DTMC behaves equivalently to the CTMC [18].

Generating paths through the DTMC is as expensive as for the CTMC and we would not gain any efficiency by doing it in an exact way. That's why, we are leaping over several states. That means, all enabled transitions that are not mutually exclusive, are forced to fire within one leap. When the net is filled up with tokens, every transition will fire within every leap. Furthermore, we embody the maximum firing rule and let each transition fire concurrently to itself.

3.1 Transition firing

How often a transition is allowed to fire concurrently depends on its enabledness degree and is determined randomly at each step.

$$\text{firing rate} \approxeq \mathit{random}[0, \text{enablness degree}] \tag{8}$$

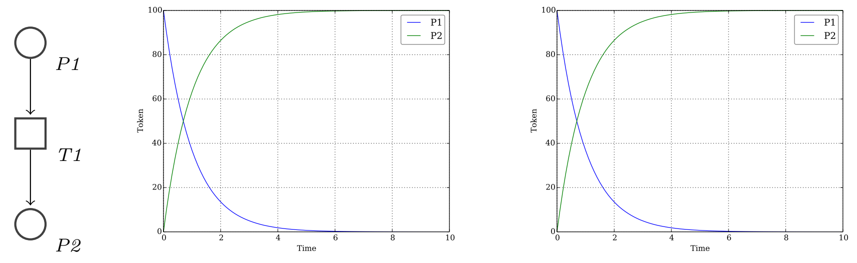
The construction of the DTMC induces that the times between transitions are all exponentially distributed. Hence, these times are randomized by a Poisson process. Since for the uniformized DTMC the number of transitions in any time interval of length δ has a Poisson distribution with rate λ. The Poisson distribution with rate λ is an approximation of the bounded discrete binomial distribution with two parameters k and pr according to the Poisson limit theorem:

$$\lambda = k \cdot pr. \tag{9}$$

The binomial distribution is used to model the number of successes in a sequence of k independent yes/no experiments with a probability pr to succeed. In our case, the enabledness degree corresponds to the sequence's length $k = \mathrm{ed}_t$. The success probability pr is deduced from equation (4), because of the exponentially distributed times between transitions. Given out of ed_t maximum firings and a firing rate h_t, we compute the probability pr for transition t in marking m for δ units of time as follows

$$pr = \begin{cases} 1 - e^{-\frac{h_t(m)}{\mathrm{ed}_t(m)} \cdot \delta} & \mathrm{ed}_t(m) > 0 \\ 0 & \textit{otherwise.} \end{cases} \tag{10}$$

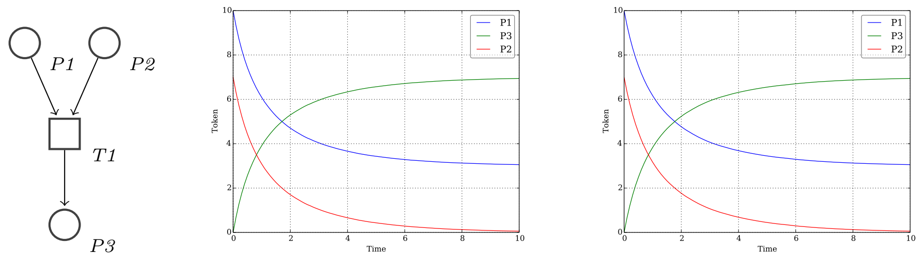
This leads us to a good approximation of the stochastic simulation results. We compare the simulation results of δ-leaping, with $\delta = 0.1$, against the direct method [8] on the most common reaction types in biochemical reaction networks, i.e., first and second order reactions. The first order reaction $P1 \rightarrow P2$ is shown in Fig. 1. It uses mass-action kinetics with rate constant 1. The results of δ-leaping match the results of the direct method.



(a) $\mathcal{PN}$ (b) δ-leaping (c) direct method

Fig. 1. First order reaction: $P1 \rightarrow P2$

The second order reaction $P1 + P2 \rightarrow P3$ uses mass-action kinetics with rate constant 0.1. The results shown in Fig. 2 are quite as good as in Fig. 1.



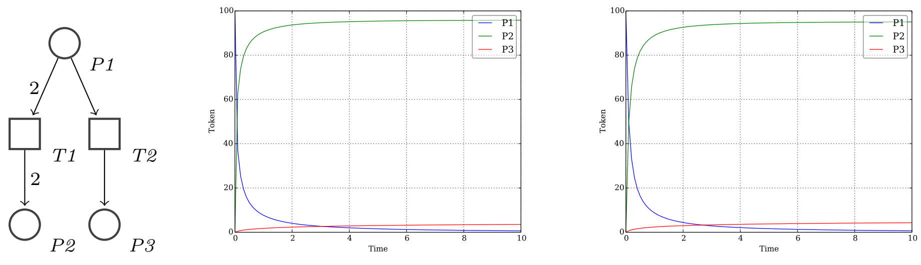
(a) $\mathcal{PN}$ (b) δ-leaping (c) direct method

Fig. 2. Second order reaction: $P1 + P2 \rightarrow P3$

3.2 Dependent Subnets

We discussed the firing of single and independent transitions, but a typical model consists of much more than that. There are two types of subnets that need some attention: *conflicts* and *sequences*.

A conflict exists inside a Petri net, if two or more transitions share a pre-place, see Fig. 3. If the amount of tokens on this place is just as much as the arc weights, one has to determine, which transition will fire. This is not an issue in the stochastic simulation algorithm. In an exact stochastic simulation only one transition is selected at a time and there is no need for further conflict resolution. As in the standard conflict resolution for Petri nets, we have to choose a transition non-deterministically. The standard conflict resolution proposes a non-deterministic selection according to a uniform distribution on the number of affected transitions. But this is not the best solution for us, because this does not pay attention to the firing rates of the transitions. We are aiming at a *weighted* non-deterministic selection, which accounts for the firing rates of transitions as well.

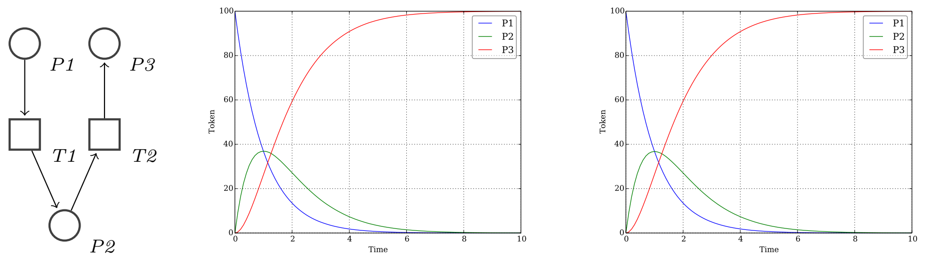


(a) $\mathcal{PN}$ (b) δ-leaping (c) direct method

Fig. 3. Conflict: $P1 \rightarrow P2$ and $P1 \rightarrow P3$

The handling of transition sequences, see Fig. 4, is closely related to the conflict resolution. We embody the maximum firing rule and force every transition to fire (if enabled) in one time step. We shuffle the transitions and let them fire (if enabled) sequentially to approximate the stochastic behaviour.

Luckily, we can treat both issues, transitions in conflict or in sequence, in one solution. We generate a weighted random sequence of all transitions $t \in T$ in each step and let them fire (if enabled) sequentially. The serial firing precludes additional attempts to solve conflicts. Our algorithm is based on the modern version of the Fisher–Yates shuffle [4] introduced in [2]. Algorithm 1 incorporates Bernoulli sampling to realize a shuffling in accordance to transition weights. The weight computation in equation (11) is inspired by the mass-action kinetics and approximates the expected firing rate, if the transition would be enabled. It is a



(a) $\mathcal{PN}$ (b) δ-leaping (c) direct method

Fig. 4. Sequence: $P1 \rightarrow P2 \rightarrow P3$

heuristic method, but it provided the best results in our experiments.

$$w_t = \sum_{p \in {}^{\bullet}t} f(p,t)^{f(p,t)} \tag{11}$$

We get quite meaningful results using our weighted random shuffle algorithm for the simulation of conflicts (Fig. 3) and sequences (Fig. 4). The results are very close to stochastic simulation. The heuristic method, used to compute the transition weights, needs further attention in future work for non mass-action kinetics.

3.3 Algorithm

So far, we discussed the essential steps of the discrete-time leap method. Here, we integrate the various steps in one algorithm, which is given in detail in Algorithm 2. Each simulation run starts with the initialization phase (Algorithm 2, line 2–4), where the simulation time, the marking, and the transition sequence are set. The

Algorithm 1 Weighted random shuffle

```
Require: transition sequence $T$, transition weights $W$, $|T| = |W| = n$
Ensure: shuffled transition sequence
 1: procedure WeightedRandomShuffle(transitions $T$, weights $W$)
 2:     for $i = n;\ i > 1;\ i \leftarrow i - 1$ do
 3:         $r \leftarrow$ uniformRandomInteger($[1, i-1]$)
 4:         $t_1 \leftarrow T_{n-i},\ t_2 \leftarrow T_{n-i+r}$
 5:         $w \leftarrow W_{t_2}/(W_{t_1} + W_{t_2})$
 6:         if bernoulliSampling($w$) $= 1$ then
 7:             swap($T_{n-i}, T_{n-i+r}$)
 8:         end if
 9:     end for
10:     return $T$
11: end procedure
```

Algorithm 2 δ-leaping algorithm

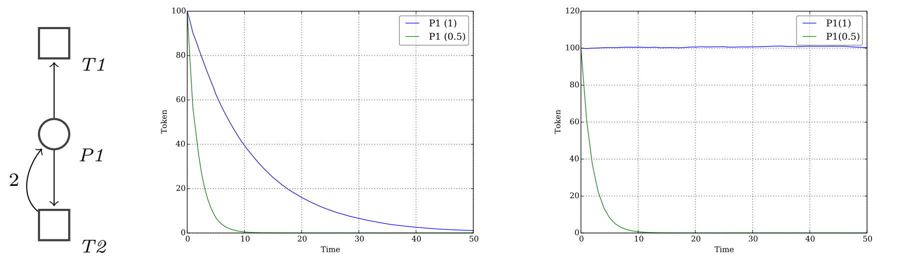
```
Require: $\mathcal{SPN}$ with initial marking $m_0$, time interval $[\tau_0, \tau_{max}]$, time step $\delta$, runs
    $r_{max}$, weights $W$
Ensure: marking $m$ at time point $\tau_{max}$
 1: for $r = 0;\ r < r_{max};\ r \leftarrow r + 1$ do
 2:     time $\tau \leftarrow \tau_0$
 3:     marking $m \leftarrow m_0$
 4:     $T_r \leftarrow T$
 5:     while $\tau \leq \tau_{max}$ do
 6:         $T_r \leftarrow$ weightedRandomShuffle$(T_r)$
 7:         for all transitions $t_j \in T_r$ do
 8:             $k \leftarrow$ enablednessDegree$(t_j, m)$
 9:             $h \leftarrow$ firingRate$(t_j, m)$
10:             if $k > 0$ then
11:                 $f \leftarrow$ binomialSampling$(a, (1 - e^{-\frac{h}{k} \cdot \delta}))$
12:                 $m \leftarrow m + f \cdot \Delta t_j$
13:             end if
14:         end for
15:         $\tau \leftarrow \tau + \delta$
16:     end while
17: end for
```

next step is the generation of the weighted random sequence of transitions using the *weightedRandomShuffle* algorithm (Algorithm 2, line 6). After that for each transition the following steps are done (Algorithm 2, line 7–14), compute the enabledness degree k, the firing rate h, and pick a random number f according to the binomial distribution defined by k and equation (10), finally the transition fires f-times. The simulation time is elapsed by δ time unit. All these steps are performed until the end of the simulation time interval τ_{max} is reached. Just like for stochastic simulation, one has to do several simulation runs, in order to get reasonable results. The overall result is the mean of all runs.

3.4 Caveat

The presented results of the discrete-time leap method approximate the stochastic simulation algorithm quite well. This might not be true for all kinds of biological mechanisms in a stochastic Petri net model, as it is the case for, e.g., a simplified birth-death process, shown in Fig. 5. The result of the stochastic simulation using mass-action kinetics with a rate constant of 1 shows a steady state of *P1* at the initial marking. Transitions *T1* and *T2* are both likely to fire and thus, fire alternately in average. The token consumed by *T1* is produced by *T2* and vice versa.

Due to the maximum firing rule, the result of the discrete-time leap method differs. Let us assume an initial marking of 100 tokens in *P1* and each transition consumes 50% of the tokens in each firing for the purpose of demonstration. There exist two possible firing sequences $S_1 = [T1, T2]$ and $S_2 = [T2, T1]$. The



(a) $\mathcal{PN}$ (b) *delta*-leaping (c) direct method

Fig. 5. Simplified birth-death process, it shows the results for different rate constants of *T2*, i.e., $c_{T2} = 1$ (blue) and $c_{T2} = 0.5$ (green)

execution of S_1 results in the following path $m_0(100) \xrightarrow{T1} m_1(50) \xrightarrow{T2} m_2(75)$ and S_2 generates $m_0(100) \xrightarrow{T2} m_1(150) \xrightarrow{T1} m_2(75)$. Both sequences decrease the amount of tokens on *P1*. So *P1* approaches zero tokens in the long term in any case. This contradicts the exact stochastic simulation results. Equal results can be obtained by adapting the rate constant of the stochastic transition, i.e., setting the rate constant of *T2* to 0.5.

4 Case Studies

In this section, we demonstrate our approach on models of the RKIP inhibited ERK pathway and the mitogen-activated protein kinase, and on two genome scale metabolic models of *E.coli* K-12. All Petri nets were modelled with Snoopy [12] and analysed with MARCIE [13]. The experiments were carried out on a MacPro with 2×Intel® Xeon® @ 2.26GHz and 32GB RAM running Mac OSX 10.11.

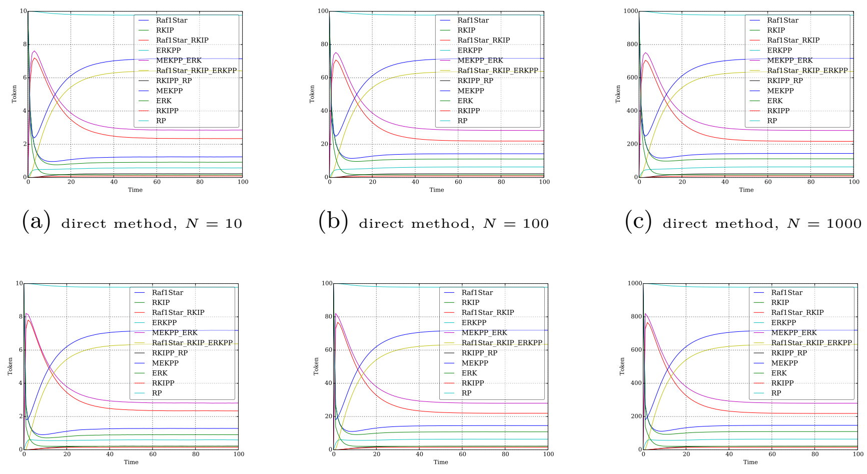
We compared the discrete-time leap method using $\delta = 1$ with stochastic simulation using Gillespie's direct method [8].

4.1 RKIP inhibited ERK pathway

This model shows the influence of the raf kinase inhibitor protein (RKIP) on the extracellular signal regulated kinase (ERK) signalling pathway. A model of non-linear ordinary differential equations was originally published in [1]. Later on, it was discussed as qualitative and continuous Petri nets in [5], and as stochastic Petri net in [9]. The Petri net $\mathcal{PN}_{ERK}$ comprises 11 places and 11 transitions connected by 34 arcs.

The model is scalable by the initial amount of tokens N in the places *RKIP*, *MEKpp*, *ERK* and *RP*. All transition rate functions of $\mathcal{SPN}_{ERK}$ use mass action kinetics with rate constants taken from [9].

We performed experiments with different values of $N = \{10, 100, 1000\}$. The simulation results are given in Fig. 6 and the run-times are provided in Table 1.



(a) direct method, $N = 10$ (b) direct method, $N = 100$ (c) direct method, $N = 1000$

(d) δ-leaping, $N = 10$ (e) δ-leaping, $N = 100$ (f) δ-leaping, $N = 1000$

Fig. 6. $\mathcal{SPN}_{ERK}$ with $N = \{10, 100, 1000\}$ and 100 000 simulation runs

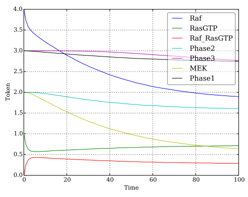
For all instances of N the results of direct method and δ-leaping match quite well. For $N = 10$ the simulation run-time for 100 000 simulation runs for direct method is lower than for δ-leaping. The simulation run-time increases for both algorithms with $N = 100$, but the discrete-time leap method is a little faster now. For $N = 1000$ the results follow the ones before, the run-time for direct method increases by a factor of 10, whereas the run-time for δ-leaping increases by only 30%.

4.2 Mitogen-activated Protein Kinase

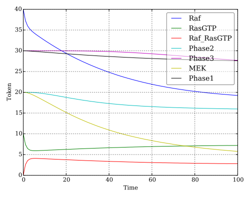
The mitogen-activated protein kinase (MAPK) is the core of the ERK/MAPK pathway that can, for example, carry cell division and differentiation signals from the cell membrane to the nucleus. The model was published in [15] and later on, discussed as stochastic Petri net in [11].

The Petri net $\mathcal{SPN}_{MAPK}$ comprises 22 places and 30 transitions connected by 90 arcs. The model is scalable by the initial amount of tokens in six places. All transition rate functions of $\mathcal{SPN}_{MAPK}$ use mass action kinetics with rate constants taken from [11].

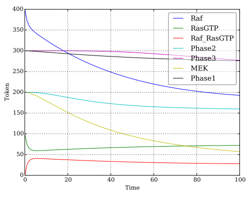
We performed experiments with the following 3 different values of $N = \{1, 10, 100\}$. The simulation results for all three instances of N are given in Fig. 7. They match quite well for the given places, as well as for the others. The run-time behaviour for this model develops in a comparable way to SPN_{ERK}, see Table 1. The run-time of the direct method in the first instance $N = 1$ is lower than for δ-leaping, but it increases much faster in the other instances of N.



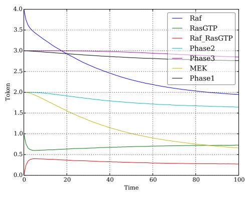
(a) direct method, $N = 1$



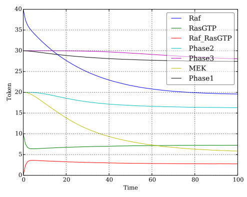
(b) direct method, $N = 10$



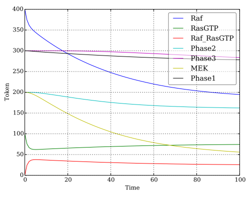
(c) direct method, $N = 100$



(d) δ-leaping $N = 1$



(e) δ-leaping, $N = 10$



(f) δ-leaping, $N = 100$

Fig. 7. $\mathcal{SPN}_{MAPK}$ with $N = \{1, 10, 100\}$ and 100 000 simulation runs

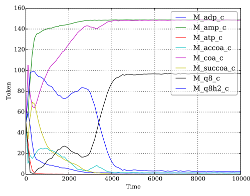
4.3 Reduced *E.coli* K-12 Genome Scale Metabolic model

This reduced model has been developed to illustrate the basic structure of the whole genome metabolism of *E.coli* K-12. The reduction was originally done by hand [17] and subsequently used for comparison with the results of an automated procedure [3]. The used version of the model is currently part of investigations concerning structural issues and was provided by [6].

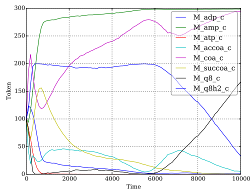
The Petri net comprises 93 places and 172 transitions connected by 589 arcs. The model is scalable by the initial amount of tokens in 12 places belonging to a place invariant. It has some places with a high connectivity (Fig. 9(a)), e.g., $M\_h\_c(52)$, $M\_h2o\_c(27)$ and $M\_h\_e(26)$. These places are involved in many reactions and changing their values leads to many updates of transition rates. This has a strong influence on the overall speed of the stochastic simulation.

We performed experiments with the following 3 different values of $N = \{50, 100, 500\}$. The simulation results in Fig. 8 and the run-times in Table 1 are quite interesting. The trajectories of M_adp_c, M_amp_c and M_atp_c correlate quite well in the direct method and δ-leaping. The traces of M_accoa_c, M_coa_c and M_succoa_c show some similarities, but M_accoa_c reaches a non zero steady state for $N = 500$ in direct method, whereas it goes to zero in δ-leaping. The reason for this, might be the different firing rules. The plots of M_q8_c and M_q8h2_c differ the most.

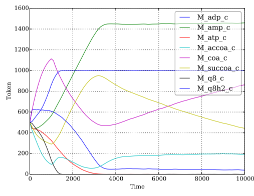
The model is still under investigation for its structural correctness and there are no kinetic rate constants available; so divergent simulation results have been expected, as well as the run-time variations differing in several orders of magnitude, which speak for themselves. Here, the direct method is clearly out of race.



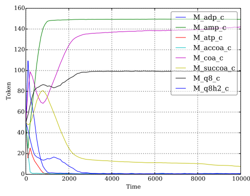
(a) direct method, $N = 50$



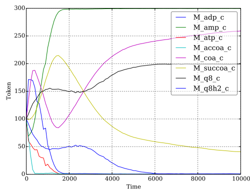
(b) direct method, $N = 100$



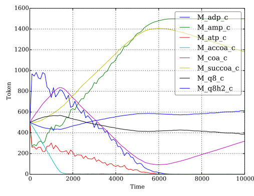
(c) direct method, $N = 500$



(d) δ-leaping $N = 50$



(e) δ-leaping, $N = 100$

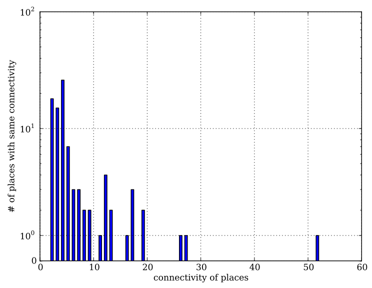


(f) δ-leaping, $N = 500$

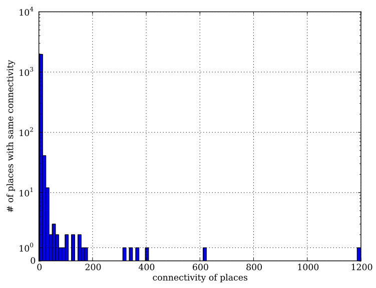
Fig. 8. *E.coli* core with $N = \{50, 100, 500\}$ and 100 simulation runs

4.4 *E.coli* K-12 Genome Scale Metabolic model

This model describes the whole genome metabolism of *E.coli* K-12 iJO1366 substrain MG1655, and is one of the 55 GEM models published by Monk et al. [16]. We have chosen this strain as an example, because it was the first strain of *E.coli* to be sequenced; it is considered to be the best curated model and thus it has been used as the basis for reconstructing the models of the other strains of *E.coli*. The used version of the model is currently part of investigations concerning structural issues and was provided by [6].



(a) Reduced *E.coli* K-12 core model

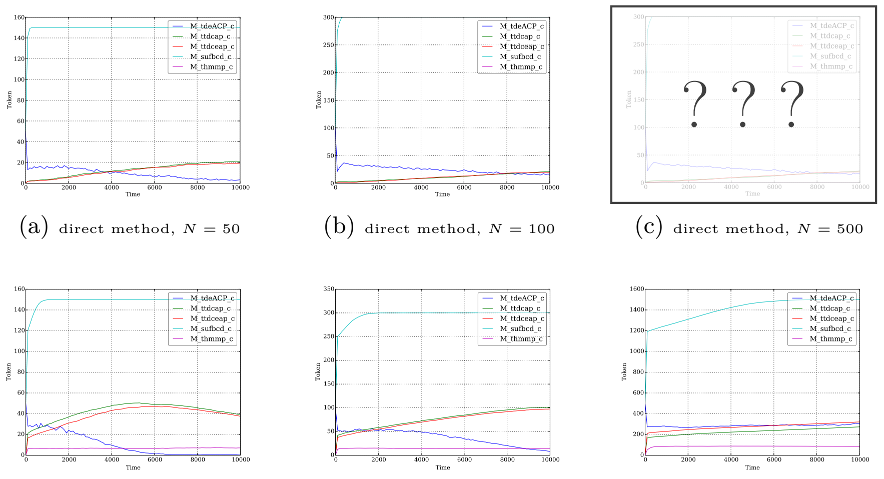


(b) *E.coli* K-12 model

Fig. 9. Connectivity of the reduced *E.coli* K-12 core model (a) and the *E.coli* K-12 genome scale metabolic model (b).

The Petri net comprises 2046 places and 3703 transitions connected by 13001 arcs. The model is scalable by the initial amount of tokens in 101 of 2046 places. The model is supposed to be rather dense. This is confirmed by looking at the place connectivity in Fig. 9(b). The top six places at the connectivity ranking are *M_h_c* with 1198 arcs, *M_h2o_c* with 617 arcs, *M_atp_c* with 402 arcs, *M_h_p* with 369 arcs, *M_pi_c* with 339 arcs and *M_adp_c* with 314 arcs. Places with such high connectivity have a large impact on the performance of stochastic simulation, because the greater connectivity of a place, the more transitions change the number of tokens on this place and the more transition rates have to be evaluated each time this happens.

We performed experiments with the following 3 different values of $N = \{50, 100, 500\}$. The simulation results given in Fig. 10, as well as the run-times



(a) direct method, $N = 50$ (b) direct method, $N = 100$ (c) direct method, $N = 500$

(d) δ-leaping $N = 50$ (e) δ-leaping, $N = 100$ (f) δ-leaping, $N = 500$

Fig. 10. *E.coli* K-12 with $N = \{50, 100, 500\}$ and 10 simulation runs

in Table 1 are quite interesting. The plots of the randomly chosen places are comparable to some extent, but others differ a lot. This is not surprising, because the model is under investigation and there are no kinetic parameters available. We were not able to finish one stochastic simulation run within 40 days for $N = 500$. The simulation run-times for δ-leaping are moderate for such a big and dense model. The scaling parameter N has little influence on the simulation run-time of δ-leaping.

5 Conclusions

We presented the discrete-time leap method for the simulation of stochastic Petri nets. It converts the underlying CTMC into a stochastically equivalent DTMC.

Table 1. Comparison of run-times for the direct method (a) and δ-leaping (b). All models were parametrised with N and simulated with different numbers of simulation runs. † is placed, if the simulation did not finish in reasonable time (>40 days).

model	N	1 run		10 runs		100 runs		100 000 runs	
		a	b	a	b	a	b	a	b
ERK	10	<1s	<1s	<1s	<1s	<1s	<1s	9s	35s
ERK	100	<1s	<1s	<1s	<1s	<1s	<1s	1m21s	46s
ERK	1000	<1s	<1s	<1s	<1s	<1s	<1s	13m27s	1m5s
MAPK	1	<1s	<1s	<1s	<1s	<1s	<1s	12s	1m7s
MAPK	10	<1s	<1s	<1s	<1s	<1s	<1s	1m47s	2m1s
MAPK	100	<1s	<1s	<1s	<1s	<1s	<1s	16m50s	2m38s
E.coli core	50	13s	<1s	2m1s	3s	50m8s	50s	†	11h20m
E.coli core	100	1m47s	<1s	18m4s	3s	6h56m	56s	†	11h50m
E.coli core	500	46m30s	<1s	10h31m	4s	7d10h	1m12s	†	14h10m
E.coli K-12	50	18h33m	9s	10d16h	1m31s	†	33m1s	†	12d13h
E.coli K-12	100	2d17h	9s	40d7h	1m32s	†	33m4s	†	12d14h
E.coli K-12	500	†	9s	†	1m32s	†	33m20s	†	13d12h

Generating paths through the DTMC is as expensive as for the CTMC and we would not gain any efficiency by doing it in an exact way. That's why, we are leaping over several states. The discrete time model and the maximum firing rule in combination with binomial sampling and weighted random shuffling of the transitions make an efficient simulation algorithm, that leads to comparable results. The weighted random shuffle is in need for further improvements to obtain even better approximations of the stochastic simulation.

We can conclude that the discrete-time leap method computes reasonable results and it has a very good run-time performance especially for larger and dense networks. It is less sensitive to higher number of tokens and thus higher transition rates than stochastic simulation algorithms in terms of run-time. This recommends δ-leaping for models, which stochastic simulation is not capable of simulating in reasonable time, e.g., genome scale metabolic models. Furthermore, it is suitable for in silico experiments, where the differences between modified models are of interest, such as knock-out scenarios of certain species or reactions. For particular models it might be necessary to adapt the kinetic rate constants in order to obtain similar results, see Section 3.4. Here, further investigations are needed if we can compute the required adaptations from the net structure.

The discrete-time leap method is implemented in our tools Snoopy [12] and MARCIE [13]. Both are free available for non-commercial use on our web page.

References

1. Cho, K.H., Shin, S.Y., Kim, H.W., Wolkenhauer, O., McFerran, B., Kolch, W.: Mathematical modeling of the influence of RKIP on the ERK signaling pathway. In: Proc. CMSB 2003. pp. 127–141. LNCS 2602, Springer (2003)

2. Durstenfeld, R.: Algorithm 235: Random permutation. Commun. ACM 7(7), 420– (Jul 1964), http://doi.acm.org/10.1145/364520.364540
3. Erdrich, P., Steuer, R., Klamt, S.: An algorithm for the reduction of genome-scale metabolic network models to meaningful core models. BMC systems biology 9(1) (2015)
4. Fisher, R.A., Yates, F., et al.: Statistical tables for biological, agricultural and medical research. Oliver and Boyd, Edinburgh, 6th edn. (1963)
5. Gilbert, D., Heiner, M.: From Petri nets to differential equations - an integrative approach for biochemical network analysis. In: Proc. ICATPN 2006. pp. 181–200. LNCS 4024, Springer (2006)
6. Gilbert, D., Heiner, M.: personal communication (2016)
7. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. J Phys Chem 81(25), 2340 – 2361 (December 1977)
8. Gillespie, D.: A General Method for Numerically Simulating the Stochastic Time Evolution of Coupled Chemical Species. J Comput Phys 22, 403–434 (1976)
9. Heiner, M., Donaldson, R., Gilbert, D.: Petri Nets for Systems Biology, in Iyengar, M.S. (ed.), Symbolic Systems Biology: Theory and Methods. Jones and Bartlett Publishers, Inc. (2010)
10. Heiner, M., Gilbert, D.: Biomodel engineering for multiscale systems biology. Progress in Biophysics and Molecular Biology 111(2-3), 119–128 (April 2013), http://www.sciencedirect.com/science/article/pii/ S0079610712001071?v=s5
11. Heiner, M., Gilbert, D., Donaldson, R.: Petri nets in systems and synthetic biology. In: SFM. pp. 215–264. LNCS 5016, Springer (2008)
12. Heiner, M., Herajy, M., Liu, F., Rohr, C., Schwarick, M.: Snoopy - a unifying Petri net tool. In: Proc. PETRI NETS 2012. LNCS, vol. 7347, pp. 398—407. Springer (June 2012)
13. Heiner, M., Rohr, C., Schwarick, M.: MARCIE - Model checking And Reachability analysis done effiCIEntly. In: Colom, J., Desel, J. (eds.) Proc. PETRI NETS 2013. LNCS, vol. 7927, pp. 389—399. Springer (June 2013)
14. Jensen, A.: Markoff chains as an aid in the study of markoff processes. Scandinavian Actuarial Journal 1953(sup1), 87–91 (1953)
15. Levchenko, A., Bruck, J., Sternberg, P.: Scaffold proteins may biphasically affect the levels of mitogen-activated protein kinase signaling and reduce its threshold properties. Proc Natl Acad Sci USA 97(11), 5818–5823 (2000)
16. Monk, J.M., Charusanti, P., Azizb, R.K., Lermand, J.A., Premyodhinb, N., Orth, J.D., Feist, A.M., Palsson, B.Ø.: Genome-scale metabolic reconstructions of multiple *Escherichia coli* strains highlight strain-specific adaptations to nutritional environments. PNAS 110(50), 20338–20343 (2013)
17. Orth, J.D., Fleming, R.M., Palsson, B.Ø.: Reconstruction and use of microbial metabolic networks: the core Escherichia coli metabolic model as an educational guide. EcoSal Plus 4(1) (2010)
18. Sandmann, W.: Brief Communication: Discrete-time stochastic modeling and simulation of biochemical networks. Comput. Biol. Chem. 32, 292–297 (August 2008), http://dl.acm.org/citation.cfm?id=1385698.1385866
19. Smallbone, K., Mendes, P.: Large-scale metabolic models: From reconstruction to differential equations. Industrial Biotechnology 9(4), 179–184 (2013)
20. Stewart, W.: Introduction to the Numerical Solution of Markov Chains. Princeton Univ. Press (1994)