Approximation of Event Probabilities in Noisy Cellular Processes[1,2]

Frédéric Didier[a], Thomas A. Henzinger[b], Maria Mateescu[a], Verena Wolf[c]

[a] *Ecole Polytechnique Fédérale de Lausanne, Switzerland*
[b] *Institute of Science and Technology, Austria*
[c] *Saarland University, Germany*

Abstract

Molecular noise, which arises from the randomness of the discrete events in the cell, significantly influences fundamental biological processes. Discrete-state continuous-time stochastic models (CTMC) can be used to describe such effects, but the calculation of the probabilities of certain events is computationally expensive.

We present a comparison of two analysis approaches for CTMC. On one hand, we estimate the probabilities of interest using repeated Gillespie simulation and determine the statistical accuracy that we obtain. On the other hand, we apply a numerical reachability analysis that approximates the probability distributions of the system at several time instances. We use examples of cellular processes to demonstrate the superiority of the reachability analysis if accurate results are required.

1. Introduction

The traditional approach for a dynamical model of cellular reaction networks is based on the assumption that the concentrations of the chemical species change continuously and deterministically in time. During the last decade, however, stochastic models with discrete state spaces have seen growing interest [9, 31, 35, 36, 46, 48, 50, 53]. The reason is that they take into account the effects of

[1]This research was supported in part by the Swiss National Science Foundation under grant 205321-111840 and by the Excellence Cluster on Multimodal Computing and Interaction at Saarland University.

[2]This paper is an extended version of [6].

molecular noise in the cell. Molecular noise has a significant influence on important processes such as gene expression [3, 8, 25, 30, 33, 49], decisions of the cell fate [1, 28, 29], and circadian oscillations [2, 16, 17].

An appropriate modeling approach for systems that are subject to molecular noise is a discrete-state continuous-time Markov process, also called *continuous-time Markov chain* (CTMC). This is particularly evident in the presence of *intrinsic noise* arising from random microscopic events in the cell, such as the location of molecules or the order of the reactions. As opposed to continuous models, the discrete-state stochastic model is able to capture the discreteness of the random events in the cell.

The evolution of such a CTMC is given by a master equation that is derived according to Gillespie's theory of stochastic chemical kinetics [13]. Since the state space grows exponentially in the number of involved chemical species, the state space of the CTMC is large, which renders its analysis difficult. Moreover, the discrete structure becomes even larger when the number of molecules in the system grows. If the populations of certain chemical species are large, their effect on the system's variance is small and they can be approximated assuming a continuous deterministic change. For species with small populations, however, a continuous approximation is not appropriate and other approximation techniques are necessary to reduce the computational effort of the analysis.

Besides the computation of cumulative measures such as expectations and variances of the populations of certain chemical species, the computation of event probabilities is important for several reasons. First, cellular process may decide probabilistically between several possibilities, e.g., in the case of developmental switches [1, 19, 36]. In order to verify, falsify, or refine the mathematical model based on experimental data, the likelihood for each of these possibilities has to be calculated. But also full distributions are of interest, such as the distribution of switching delays [30], the distribution of the time of DNA replication initiation at different origins [34], and the distribution of gene expression products [52]. Finally, many parameter estimation methods require the computation of the posterior distribution because means and variances do not provide enough information to calibrate parameters [21].

Two different families of computational approaches have been proposed and used to estimate event probabilities and approximate probability distributions. The first kind of approach is based on numerical simulation, i.e., the generation of many sample trajectories (or *simulation runs*) of the system. The second kind of approach is based on numerical reachability analysis, i.e., the propagation of the probability mass through the state space. The former approach is known as

Gillespie simulation [12], in which pseudo-random numbers are used to simulate molecular noise. Measures of interest are obtained via statistical output analysis. The main advantage of simulation is that it is easy to implement and the generation of trajectories is not limited by the size of the state space. Moreover, the precision level of the method can be easily adjusted by performing more or fewer simulation runs. For the computation of the probability of certain events, however, simulative approaches become computationally expensive, because a large number of runs have to be carried out to bound the statistical error appropriately. For estimating event probabilities, a higher precision level is necessary than for estimating cumulative measures such as expectations, and simulation becomes expensive because doubling the precision requires four times more simulation runs.

In contrast, approaches based on a numerical reachability analysis approximate probability distributions of the CTMC. As opposed to a statistical estimation of probabilities, which yields an indirect solution, the master equation is numerically solved by integrating the system's behavior over time. Standard numerical techniques are impractical for many systems because of the enormous size of the state space. Recently, however, more sophisticated numerical approximation methods have been proposed, which solve the system in an iterative fashion and consider only subsets of the state space during any given time interval [5, 22, 32, 44]. They are significantly more efficient than global analysis because they use localization optimizations (such as "sliding windows") and dynamic adaptation ("on-the-fly" generation of windows). These methods efficiently compute the probability distribution of large CTMC at several time instances up to a small approximation error. They can also be used for infinite-state systems.

In this paper, we evaluate and compare the performance of the two different approaches for the computation of probabilities of certain events, i.e., the statistical estimation using simulation and the approximation using a numerical reachability analysis. For the latter we use a particular algorithm as a representative of the whole family of numerical analysis algorithms, because we have found it to perform best. Similar to the sliding-window method [22], our algorithm performs a sequence of local analysis steps on dynamically constructed abstractions of the system. The main improvement over the sliding-window method is that our algorithm is based on adaptive uniformization [51], which allows us to consider arbitrary sets of significant states, i.e., they may be located at different parts of the state space and are not restricted to a specific window shape. Moreover, adaptive uniformization is more robust if the system under study is stiff, i.e., if the chemical reactions occur at time scales that differ by several orders of magnitude. In contrast to [22], here, for the first time, we perform a systematic experimental

performance comparison of a numerical reachability analysis with simulation.

The first example that we consider is a model of intracellular signaling through immune receptors that are involved in antigen recognition [15]. The model consists of 12 different chemical species and 19 reactions and is the most complex example that we consider. The second example is the transcription regulation of a repressor protein in bacteriophage λ [18]. In the first two examples, we approximate the probability distribution at several time instances. In the third example, which is a gene expression network [49], we compute the distribution of the time until the number of produced proteins exceeds a certain threshold. Our last example is the model of a genetic toggle switch in Escherichia coli where bistability arises from the mutually inhibitory arrangement of two repressor genes [11]. We approximate the probability distribution until the system reaches its bistable steady-state. Note that all examples that we consider are infinite in several dimensions.

We compare the running time of our numerical reachability analysis to that of the simulative approach for both examples, for different precision levels. Our results show that numerical approximation based on reachability analysis is superior to statistical estimation based on repeated simulation, especially if we increase the desired precision level. For instance, the numerical approximation of the second example needs 39 minutes for a total approximation error of 2×10^{-5}, which distributes among all states. Simulation requires more than six hours if the statistical error of a single event is to be bounded by 10^{-5} and more than sixty hours for 10^{-6}.

2. Stochastic Model

According to the theory of stochastic chemical reaction kinetics, a continuous-time Markov chain (CTMC) can be derived from a set of biochemical reactions [13, 24]. This discrete-state model has a regular structure, which gives rise to a functional description in terms of *transition class models* (TCMs) [42]. TCMs naturally represent coupled chemical reactions as each chemical reaction corresponds to a transition class. They provide, however, a more general description than a set of chemical reactions.

2.1. Transition Class Models

Consider a dynamical system with a finite number of discrete state variables such as the number of instances of some chemical species in a reaction volume. Assume that these variables change at discrete points in time. A *transition class*

provides a rule for these changes and a function for the calculation of the state-dependent *transition rate* at which a state change occurs. Let S be a countable set of states.

Definition 1. *A* transition class C *is a triple* (G, u, α) *such that (i) the guard* $G \subset S$ *is a subset of* S*, (ii)* $u : G \to S$ *is an injective update function with* $u(x) \neq x$ *for all* $x \in G$*, (iii)* $\alpha : G \to \mathbb{R}_{>0}$ *is a rate function. A* transition class model *(TCM)* $M = (y, \{C_1, \ldots, C_k\})$ *consists of an initial state* $y \in S$ *and a finite set of transition classes* $C_1, \ldots, C_k$*.*

The set G contains all states x in which a transition of type C is possible and $u(x)$ is the target state of the transition. The probability of the C-transition depends on the transition rate $\alpha(x)$ in the way explained below.

In practice, we can usually express G by a finite number of constraints on the state variables, and u and α by elementary arithmetic functions. Thus, a TCM provides a finite description of a (possibly infinite-state) system. Before we show how a CTMC is derived from a TCM, we present some examples of TCMs that describe biochemical reaction networks.

Biochemical Reaction Networks. We consider a fixed reaction volume with n different chemical species that is spatially homogeneous and in thermal equilibrium. Then, the state space of the system is given by $S = \mathbb{N}_0^n$. We assume that molecules collide randomly and that collisions may lead to chemical reactions. For a given set of chemical reactions, we construct a TCM such that each transition class corresponds to a reaction and the associated propensity function is given by the rate function α. Formally, assume that the network consists of k different chemical reactions. Let $m \in \{1, \ldots, k\}$, and let the m-th reaction be given by the stoichiometric equation

$$k_1 S_1 + \ldots + k_n S_n \to \ell_1 S_1 + \ldots + \ell_n S_n$$

where for $i \in \{1, \ldots, n\}$ the symbol S_i refers to the i-th chemical species and the stoichiometric coefficients k_i, ℓ_i are non-negative integers, which specify how many molecules of type i are consumed and how many are produced by the reaction, respectively. If $k_i > 0$ then the i-th species is called a reactant of the m-th reaction. In stoichiometric equations, terms with coefficient 0 are usually omitted and terms of the form $1S_i$ are abbreviated by S_i. The symbol $\emptyset$ abbreviates the case $0 = k_1 = \ldots = k_n$ or $0 = \ell_1 = \ldots = \ell_n$. We define the m-th transition class

$C_m = (G_m, u_m, \alpha_m)$ such that

$$\begin{aligned} G_m &= \{x = (x_1, \ldots, x_n) \in \mathbb{N}_0^n \mid x_i \geq k_i, i \in \{0, 1, \ldots\}\}, \\ u_m(x) &= x + (\ell_1 - k_1, \ldots, \ell_n - k_n), \\ \alpha_m(x) &= c \cdot \prod_{i=1}^{n} \binom{x_i}{k_i}. \end{aligned}$$

The rate function α_m takes into account that the probability of a reaction of type m is proportional to the possible number of combinations of reactant molecules, i.e., if k_i molecules of type i are needed and the current number of molecules of type S_i is x_i then $\binom{x_i}{k_i}$ is the number of possible ways to choose k_i out of x_i. The rate constant $c > 0$ depends on the temperature, the volume, and the microphysical properties of the reactant species [14].

Example 1. *We consider a simple transition class model for transcription of a gene into messenger RNA (mRNA), and subsequent translation of the latter into proteins [49]. This reaction network involves three chemical species, namely, gene, mRNA, and protein. As only a single copy of the gene exists, a state of the system is uniquely determined by the number of mRNA and protein molecules. Therefore, $S = \mathbb{N}_0^2$ and a state is a pair $(x_R, x_P) \in S$. We assume that initially there are no mRNA molecules and no proteins in the system, i.e., $y = (0, 0)$. The following four types of reactions occur in the system, namely $\emptyset \to mRNA$, $mRNA \to mRNA + P$, $mRNA \to \emptyset$, and $P \to \emptyset$. Let $m \in \{1, \ldots, 4\}$ and let $c_m > 0$ be a constant. Transition class $C_m = (G_m, u_m, \alpha_m)$ describes the m-th reaction type.*

- *We describe gene transcription by transition class C_1, which increases the number of mRNA molecules by 1. Thus, $u_1(x_R, x_P) = (x_R + 1, x_P)$. This transition class is possible in all states, i.e., $G_1 = S$. Transcription happens at the constant rate $\alpha_1(x_R, x_P) = c_1$, as only one reactant molecule (the gene) is available.*

- *We represent the translation of mRNA into protein by C_2. A C_2-transition is only possible if there is at least one mRNA molecule in the system. We set $G_2 = \{(x_R, x_P) \in S \mid x_R > 0\}$ and $u_2(x_R, x_P) = (x_R, x_P + 1)$. Note that in this case mRNA is a reactant that is not consumed. The translation rate depends linearly on the number of mRNA molecules. Therefore, $\alpha_2(x_R, x_P) = c_2 \cdot x_R$.*

- *Degradation is modeled by C_3 and C_4. Hence, $G_3 = G_2$, $G_4 = \{(x_R, x_P) \in S \mid x_P > 0\}$, $u_3(x_R, x_P) = (x_R - 1, x_P)$, and $u_4(x_R, x_P) = (x_R, x_P - 1)$. We set $\alpha_3(x_R, x_P) = c_3 \cdot x_R$ and $\alpha_4(x_R, x_P) = c_4 \cdot x_P$.*

2.2. Chemical Master Equation

A transition class model $M = (y, \{C_1, \ldots, C_k\})$ represents a time-homogeneous, discrete-state Markov process $\{X(t)\}_{t \geq 0}$, that is, a CTMC with state space S. The j-th entry of the random vector $X(t) = (X_1(t), \ldots, X_n(t))$ represents the value of the j-th state variable. Let $C_m = (G_m, u_m, \alpha_m)$, $1 \leq m \leq k$, and assume that at time $t \geq 0$ the process is in state $x \in G_m$.

The probability of a transition of type C_m occurring in the next infinitesimal time interval $[t, t + \tau)$, $\tau > 0$ is given by

$$Pr(X(t + \tau) = u_m(x) \mid X(t) = x) = \alpha_m(x) \cdot \tau.$$

Since y is the initial state of M we have $Pr(X(0) = y) = 1$, and for $x \in S$ we define the probability that X is in state x at time t by

$$p^{(t)}(x) = Pr\left(X(t) = x \mid X(0) = y\right).$$

Recall that u_m is injective. To simplify our presentation, we define the set H_m as the set of all states x for which $u_m^{-1}(x)$ is defined, that is, that can be reached by a transition of type C_m. The *chemical master equation* describes the behavior of X by the differential equation [24]

$$\frac{\partial p^{(t)}(x)}{\partial t} = \sum_{m : x \in H_m} \alpha_m(u_m^{-1}(x)) \cdot p^{(t)}(u_m^{-1}(x)) - \sum_{m : x \in G_m} \alpha_m(x) \cdot p^{(t)}(x) \ . \quad (1)$$

Unbounded Range. For realistic systems, the state space of the Markov chain is extremely large, because its size grows exponentially in the number of involved chemical species. Moreover, if upper bounds on the state variables cannot derived from certain conservation laws, their range is assumed to be infinite although in practice the number of molecules is bounded. Then from the infinite structure, we can compute bounds that are kept with a very high probability. Even though every state in the infinite state space has a non-zero probability, certain attracting regions force most of the probability mass to remain within a finite range.

Example 2. *In Ex. 1, the degradation rates $\alpha_3(x)$ and $\alpha_4(x)$ grow linearly in the state variables. Thus, the higher the number of mRNA or protein molecules the more likely is their degradation. Depending on the rate constants $c_1, \ldots, c_4$, the*

system becomes "stable" in different regions. As time approaches infinity, the main part of the probability mass will be close to a region where production and degradation of molecules cancel each other out. Below, we discuss in general under which conditions the system approaches such a stable distribution.

Holding Times and Jump Probabilities. A Markov chain $\{X(t)\}_{t\geq 0}$ defined in the way above is a *stable and conservative jump process* [4]. Thus, there exists a sequence of jump times $\{\tau(n)\}_{n\geq 0}$ and a sequence $\{\hat{X}(n)\}_{n\geq 0}$ of visited states such that

$$\tau(0) = 0 < \tau(1) < \tau(2) < \dots \text{ and } X(t) = \hat{X}(n) \text{ if } \tau(n) \leq t < \tau(n+1).$$

The distribution of the n-th holding time $\tau(n+1) - \tau(n)$ under the condition $\hat{X}(n) = x$ is negative exponentially distributed with parameter

$$\lambda(x) = \sum_{m:x\in G_m} \alpha_m(x),$$

also called *exit rate* of state x.

If the sum of all holding times is finite with positive probability, the Markov chain is said to *explode* and the limiting distribution does not exist. Explosive Markov chains are not of interest for the application area of this work since in this case the system "gets lost at infinity". It is possible to check if the Markov chain does not explode by using *Reuter's Criterion* [4]. For the remainder of our presentation we assume that the rate functions α_m are such that the Markov chain does not explode.

Assume that the n-th state of the Markov chain is x, that is, $\hat{X}(n) = x$. If at least one transition class is enabled in x, the successor state is $u_m(x)$ for some m with $x \in G_m$. The probability of successor $u_m(x)$ is given by

$$Pr(\hat{X}(n+1) = u_m(x) \mid \hat{X}(n) = x) = \tfrac{\alpha_m(x)}{\lambda(x)}.$$

The holding times and the jump probabilities play an important role for the simulation of the Markov chain, which is used to estimate the probability of a certain events.

3. Statistical Estimation of Probabilities

In this section we shortly review the basic steps that have to be carried out to estimate the probability of a certain measurable event using stochastic simulation.

Throughout this section, we will denote this event by A and its probability by γ. For the analysis of biological systems, the events of interest may be the marginal distributions or even the joint distributions of certain chemical species. For instance, A may have the form $X_j(t) = k$, that is, the number of type j molecules is k.

Estimates are obtained in two steps. In the first step, a certain number of simulation runs of the Markov chain have to be generated, and in the second step, the results of the simulation runs are analyzed.

3.1. Trajectory Generation

A realization of the Markov chain, also called *trajectory* or *run*, is the random sequence of states visited by the process. If trajectories are produced by a computer, *pseudo-random numbers* are used to artificially generate randomness [26]. The basic steps of producing a single trajectory that starts in the initial state y at time 0 are as follows:

1. Initialize time $t = 0$ and state $x = y$.
2. Generate the holding time h, i.e., a sample of a random variable being exponentially distributed with parameter $-\lambda(x)$.
3. Generate the successor state, i.e., a sample m of a discrete random variable Z that has probability distribution $P(Z = m) = \alpha_m(x)/\lambda(x)$.
4. Set $t = t + h$, $x = u_m(x)$ and go to Step 2 if $t < T$.

In Step 2, we generate the holding time of the current state x. Pseudo-random number generators usually draw from a uniform distribution. Thus, for a given random sample r_1 that is uniformly distributed on $(0, 1)$, we calculate an exponentially distributed sample by using the inverse transform method. More precisely, we compute the inverse $-\frac{\ln r_1}{\lambda(x)}$ of the cumulative distribution function of the exponential distribution. In Step 3, the same idea is used to decide, which reaction occurs next. The inverse of the cumulative distribution function of Z is given by $m = \min\{i : \sum_{j=1}^{i} \alpha_j(x) > r_2 \cdot \lambda(x)\}$, where r_2 is again a random sample that is uniformly distributed on $(0, 1)$. In the final step, the current time and the current state are updated. The simulation is terminated if the time horizon T of interest is reached and continued otherwise.

3.2. Output Analysis

The problem of estimating the probability γ of the event A can be reformulated as estimating the expectation of the random variable χ_A with

$$\chi_A(\omega) = \begin{cases} 1 & \text{if} \quad \omega \in A, \\ 0 & \text{if} \quad \omega \notin A, \end{cases}$$

where ω is a trajectory. The expectation $E[\chi_A]$ equals γ, since $E[\chi_A] = 1 \cdot Pr(\chi_A = 1) + 0 \cdot Pr(\chi_A = 0) = \gamma$. Therefore, we can resort to the standard estimation procedure for expectations. Assume that N is the number of runs that have been carried out and $Y_1, \ldots, Y_N$ are independent and identically distributed as χ_A. Thus, from the i-th run we get a realization of Y_i by checking if A has occurred or not. It is important to point out that we have to guarantee the independence of the Y_i's. This implies that we generate N independent trajectories of the Markov chain, each time with a different initial seed[1] for the pseudo-random number generator. The sample mean $\bar{Y} = \frac{1}{N}\sum_{i=1}^{N} Y_i$ is then an *unbiased* and *consistent estimator* [26] for $E[\chi_A]$. The former means that $E[\bar{Y}] = E[\chi_A]$ and the latter refers to the fact that as N increases the estimator $\bar{Y}$ becomes closer to γ. Note that $\bar{Y}$ is equal to the relative frequency of the event A. Let $\sigma^2 = VAR[\chi_A]$ be the variance of χ_A. We evaluate the quality of the estimator $\bar{Y}$ by applying the central limit theorem, which states that $\bar{Y}$ will approximately have a Normal distribution with mean $E[\chi_A] = \gamma$ and variance σ^2/N. Hence, for large N the random variable

$$Z = \frac{\bar{Y} - \gamma}{\sqrt{\sigma^2/N}}$$

has a standard Normal distribution, that is, the mean is zero and the variance is one. Knowing the distribution of Z enables us reason about the difference $|\bar{Y} - \gamma|$. Let $\beta \in [0, 1]$ be the *confidence level* and $z \in \mathbb{R}^+$ such that $\beta = Pr(|Z| \leq z)$. Then

$$\beta = Pr(|Z| \leq z) = Pr\left(\frac{|\bar{Y}-\gamma|}{\sqrt{\sigma^2/N}} \leq z\right) = Pr\left(|\bar{Y} - \gamma| \leq z\sqrt{\sigma^2/N}\right).$$

We estimate σ^2 with the sample covariance $S^2 = \frac{1}{N-1}\sum_{i=1}^{N}(Y_i - \bar{Y})^2$, which is an unbiased estimator for σ^2. Then, for large N and a large number of realizations of the *confidence interval*

$$\left[\bar{Y} - z\sqrt{S^2/N}, \bar{Y} + z\sqrt{S^2/N}\right], \tag{2}$$

β is the fraction of intervals that cover γ. It therefore measures the quality of the estimator $\bar{Y}$.

[1]The seed of a pseudo-random number generator is an initial value, on which the sequence of generated numbers depend [26].

For a practical application, two further remarks are important. Firstly, we usually choose $\beta \in \{0.95, 0.99\}$ and the corresponding value of z can be found in the table of the standard Normal distribution. Let Φ be the cumulative distribution function of the standard Normal distribution. Then, using that the Normal distribution is symmetric,

$$\Phi(z) = Pr(Z \leq z) = 1 - \tfrac{1-\beta}{2} = \tfrac{1+\beta}{2} \iff z = \Phi^{-1}\left(\tfrac{1+\beta}{2}\right).$$

Secondly, both, $\bar{Y}$ and S^2 can be computed efficiently if during the trajectory generation the realizations of the two sums $\sum_{i=1}^{N} Y_i$ and $\sum_{i=1}^{N} Y_i^2$ are calculated, since it can be easily shown that

$$S^2 = \tfrac{\sum_{i=1}^{N} Y_i^2}{N-1} - \tfrac{\left(\sum_{i=1}^{N} Y_i\right)^2}{(N-1)N}.$$

Thus, if $r \in \{0, \ldots, N\}$ is the number of times event A occurred during the N simulation runs, $\bar{Y} = r/N$ and $S^2 = \tfrac{r(N-r)}{N(N-1)}$.

If the interval in Eq. 2 is large relative to $\bar{Y}$ the quality of the estimator is poor and more simulation runs have to be carried out. For our experimental results in Section 5, we fixed the relative width of the interval to be 0.2 (which means that we have a relative error of at most 0.1) and chose confidence level $\beta = 0.95$. Thus, $z \approx 1.96$ and we can determine the number of necessary runs by bounding the relative width

$$2 \cdot \tfrac{z \cdot \sqrt{S^2/N}}{\gamma} \leq 0.2 \implies \tfrac{z^2}{0.01} \tfrac{S^2}{\gamma^2} \leq N \implies 384 \cdot \tfrac{S^2}{\gamma^2} \leq N$$

Assume now that we want to estimate the probability of events that occur at least with probability γ. Using the fact that $\sigma^2 = VAR[\chi_A] = \gamma(1-\gamma)$ and replacing S^2 by σ^2 yields $N \geq 384 \cdot \tfrac{1-\gamma}{\gamma}$ [41]. For instance, the sufficient number of runs to guarantee that probabilities, having at least the order of magnitude of 10^{-5}, are estimated with a relative error of at most 0.1 and a confidence of 95% is $N = 38,000,000$. For a detailed discussion about a sufficient number of trajectories, we refer to [40].

During the last decade more sophisticated simulation algorithms have been developed (see [39] for overview). Most of them, however, do not give exact trajectories of the Markov process but approximations and the error of this approximation is difficult to determine. Therefore, we do not use these techniques for our comparison. An alternative would be a conversion to discrete time as recently propsed by Sandmann [38]. This method, however, has the disadvantage that a tight upper bound for the exit rates of all states found during the simulation must be known a priori.

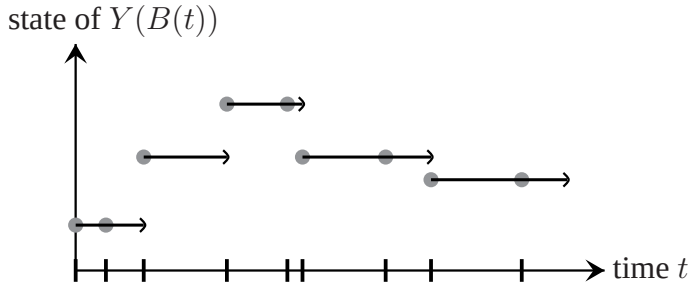

Figure 1: Construction of the process $\{Y(B(t))\}_{t\geq 0}$. The gray circles represent the state of Y and the black ticks on the x-axis the jump times of B.

4. Numerical Reachability Analysis

Instead of indirectly approximating probabilities with statistical estimation procedures, we can use a numerical reachability analysis to solve Eq. 1. An efficient solution by applying standard numerical methods is not possible, since for realistic systems the state space of the system is extremely large. An efficient approximation is, however, possible as long as the total number of involved molecules is a manageable number. We describe a method that is based on a discretization of the process and numerically approximates the probabilities $p^{(t)}(x)$ at certain time instances.

Adaptive Uniformization. We discretize the system using *adaptive uniformization*, which has been introduced by van Moorsel [51] as a variant of *standard uniformization* [20, 37, 43, 44, 54]. Numerical methods based on uniformization have the advantage that they are numerically stable and often more efficient than other methods [47].

The main idea behind uniformization methods is to construct a new stochastic process $\{Y(B(t))\}_{t\geq 0}$ such that for all states x and all times $t \geq 0$,

$$Pr\big(X(t) = x\big) = Pr\big(Y(B(t)) = x\big). \tag{3}$$

The process Y "observes" the state of the original process X at discrete points in time as illustrated in Fig. 1. The observation times are determined by a simple counting process B (see Fig. 2).

For the construction of $\{Y(B(t))\}_{t\geq 0}$, we define a sequence $S_0, S_1, \ldots$ of subsets of the state space S of the CTMC X, as well as a sequence $p_0, p_1, \ldots$ such that for $k = 0, 1, \ldots$ the function $p_k : S \to [0, 1]$ contains the state probabilities

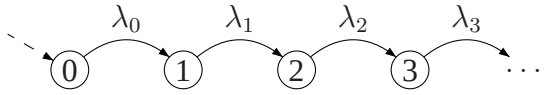

Figure 2: The birth process of the adaptive uniformization procedure.

of Y after k steps and S_k contains all states where p_k is positive. Recall that y is the initial state. At time 0, we define $S_0 = \{y\}$, $p_0(y) = 1$ and $p_0(x) = 0$ if $x \neq y$. For $k = 1, 2, \ldots$, we inductively define S_k as follows. We choose a positive *uniformization rate* $\lambda_k \geq \max_{x \in S_k} \lambda_x$ and set

$$S_{k+1} = \{x' \in S \mid \exists x \in S_k : p_k(x) \cdot q_k(x, x') > 0\}, \tag{4}$$

where, for $x \in S$,

$$q_k(x, x') = \begin{cases} \sum_{m:u_m(x)=x'} \alpha_m(x)/\lambda_k & \text{if} \quad x \neq x', \exists m : u_m(x) = x', \\ 0 & \text{if} \quad x \neq x', \nexists m : u_m(x) = x', \\ 1 - \sum_{x' \in S: x' \neq x} q_k(x, x') & \text{if} \quad x = x'. \end{cases} \tag{5}$$

For $x' \in S_{k+1}$ we set $p_{k+1}(x') = \sum_{x' \in S_k} p_k(x) \cdot q_k(x, x')$ and $p_{k+1}(x) = 0$ if $x \notin S_k$.

The value $p_k(x)$ is the probability of reaching state x after k steps in a discrete-time Markov chain $\{Y(k)\}_{k \in \mathbb{N}}$ with transition probabilities $Pr(Y(k+1) = x' \mid Y(k) = x) = q_k(x, x')$ and initial distribution $Pr(Y(0) = y) = 1$. We can reconstruct $p^{(t)}(x)$ by considering the process B that relates steps with time. Formally, let $\{B(t)\}_{t \geq 0}$ be a birth process with birth rates $\lambda_0, \lambda_1, \ldots$, that is, B has a chain structure as illustrated in Fig. 2 and starts initially in state 0 with probability one. In [51], van Moorsel has proven that Eq. (3) holds if B does not explode. Since Y and B are independent, the state probability $p^{(t)}(x)$ of the original CTMC can be expressed as

$$p^{(t)}(x) = \sum_{k=0}^{\infty} Pr(Y(k) = x) \cdot Pr(B(t) = k) = \sum_{k=0}^{\infty} p_k(x) \cdot Pr(B(t) = k). \tag{6}$$

Note that in Eq. 6, there are no negative summands involved. Moreover, p_k can be computed inductively. Lower and upper summation bounds L and U can be

obtained such that for each state x the truncation error

$$\begin{aligned} p^{(t)}(x) - \sum_{k=L}^{U} p_k(x) \cdot Pr(B(t)=k) = \sum_{\substack{0\le k<L,\\ U<k<\infty}} p_k(x) \cdot Pr(B(t)=k) \le \\ \sum_{\substack{0\le k<L,\\ U<k<\infty}} Pr(B(t)=k) = \qquad 1 - \sum_{k=L}^{U} Pr(B(t)=k) \le \epsilon \end{aligned} \tag{7}$$

can be bounded by $\epsilon > 0$. Finally, we note that from Eq. 5 it is clear that choosing the smallest possible λ_k is advantageous since this avoids high self-loop probabilities in q_k. Since $S_0 \subseteq S_1 \subseteq \ldots$ the sequence $\lambda_0, \lambda_1, \ldots$ of uniformization rates is monotonically increasing and converges to the supremum $\sup_{x\in S} \lambda(x)$.

Standard Uniformization. Standard uniformization is a special case of adaptive uniformization where a global uniformization rate $\lambda = \lambda_0 = \lambda_1 = \ldots$ has to be chosen. If each transition in the birth process occurs at a constant rate λ, the values $Pr(B(t) = k)$ follow a Poisson distribution with parameter λt. They can be calculated efficiently using the iterative procedure introduced by Fox and Glynn [10]. Standard uniformization becomes inefficient whenever λ is much larger than the exit rates $\lambda(x)$ of many states x that are involved in the computation. If the dynamics of the system is initially slow and increases as time progresses, then adaptive uniformization is more efficient, since the uniformization rate will initially be small and increase during the iteration. Finally, it will approach the global uniformization rate λ.

Approximate Discretization. In its standard form, adaptive uniformization is not appropriate for Markov chains that describe biochemical reaction networks for two reasons. Firstly, the sizes of the sets $S_0, S_1, \ldots$ grow after each step and the computational complexity for p_k becomes huge. Secondly, the birth process may become fast even if the dynamics of the system becomes slow. The reason is that after k iterations all states that are reachable within k steps from the initial state are elements of S_k. Even if the main part of the probability mass is concentrated on states with small exit rates, there may be states in S_k with a very small probability and a large exit rate. Since $\lambda_k = \max_{x\in S_k} \lambda(x)$, the transition rates of the birth process are large and the truncation point U moves to the right, which means that many iterations are necessary to achieve the desired accuracy.

Both problems mentioned above can be significantly defused by neglecting states that are very unlikely, that is, we replace Eq. 4 by

$$S_{k+1} = \{x' \in S \mid \sum\nolimits_{x\in S_k} p_k(x) \cdot q_k(x, x') > \Delta\}, \tag{8}$$

where Δ is a small positive constant. This ensures that the sizes of the sets S_k remain manageable. Moreover, the rate of the birth process corresponds to the rates of the states having "significant" probability.

The error after k steps introduced by the threshold Δ can be calculated as $1 - \sum_{x \in S_k} p_k(x)$. Note that this error increases monotonically in k since more and more probability "gets lost". Therefore we choose Δ several orders of magnitude smaller than the desired precision. For our experimental results in Section 5 we chose different values for Δ ranging from 10^{-15} till 10^{-8} in order to obtain different precision levels.

Approximate Solution of the Birth Process. We use standard uniformization to compute the probabilities $Pr(B(t) = k)$, since we can afford a high global uniformization rate (and thus, high self-loop probabilities) in this case. The reason is that the simple chain structure eases the discretization and the computational effort to solve the birth process is small compared to the calculation of the p_k. Let $\{Y_B(k)\}_{k \in \mathbb{N}_0^+}$ be the discrete-time Markov chain that results from the discretization of B and let $\{N_B(t)\}_{t \geq 0}$ be the corresponding counting process. Since we use standard uniformization, N_B is a Poisson process whose state probabilities $Pr(N_B(t) = k)$ can be computed efficiently [10]. Similar as for Y we approximately solve Y_B by neglecting states that are "left behind". Informally, we use a window (a set that contains all states within a certain range) that slides from left to right to approximate the state probabilities of Y_B. The total approximation error for the computation of the probabilities $Pr(B(t) = i)$ after k steps is then given by $1 - \sum_{i=0}^{k} Pr(B(t) = k)$.

Approximation Error. Both, the solution of Y and B gives an underapproximation of the values $p_k(x)$ and $Pr(B(t) = k)$. Thus, summing up their product according to Eq. 6 results in an underapproximation for $p^{(t)}(x)$. The final approximation error is obtained as $\delta = 1 - \sum_{x \in S_U} p^{(t)}(x)$ where U is the right truncation bound of the birth process. The probability of states that are not in S_U is approximated with zero. Note that this includes all approximation errors, i.e., the approximation error for the computation of $Pr(B(t) = k)$ and $p_k(x)$ for all $k \leq U$ and all states x, as well as the error that arises from the truncation of the infinite sum.

For our experimental results, we used the criterion in Eq. 7 to determine a truncation point U. Let $p_B(i)$ be the approximation of $Pr(B(t) = i)$ that we obtain by solving B as described above. Note that it may be the case that the terms $\sum_{x \in S_k} p_B(k)$ decrease so fast that an accuracy of ϵ can never be reached. Therefore, it is necessary to bound the total number of iterations by $\tilde{U}$ where $\tilde{U}$ is the

truncation point of the solution of the birth process using standard uniformization. For our experimental results it was never the case that we had to iterate until $\tilde{U}$, i.e. the solution of the birth process was always such that $1-\sum_{k=L}^{U} Pr(B(t)=k) \leq \epsilon$ where $U < \tilde{U}$. We chose $\epsilon = 10^{-7}$ for our results in Section 5.

Note that, alternatively, we can monitor the total error

$$\epsilon_k = 1 - \sum_{i=0}^{k} \sum_{x \in S_i} p_i(x) \cdot p_B(i)$$

after k iterations and stop the iteration if "enough" summands have been added, i.e., if a certain accuracy ϵ_k is reached. Again, this criterion is not sufficient to guarantee termination of the algorithm and an additional bound on the number of iterations is necessary.

The computational savings achieved by solving Y as well as B in the way described above are substantial. The reason is that the number of states in B and Y that are significant after k steps is several orders of magnitudes smaller than the number of all states reachable after k steps. Moreover, our experimental results show that if we choose Δ several orders of magnitude smaller than ϵ, then the desired accuracy is always achieved.

We summarize the algorithm as follows:

1. Initialize the significant set $S := \{y\}$.
2. Initialize probability functions r, p, and q on S with $r(y) := 0$, $p(y) := 1$, and $q(y) := 1$.
3. Initialize the sum of coefficients with $sum := 0$.
4. Initiailize the step count with $k := 0$.
5. While $sum < 1 - \epsilon$ and $k < \tilde{U}$
   (a) Set $\lambda_k = \max_{x \in S} \lambda(x)$.
   (b) Compute $coeff = Pr(B(t) = k)$ using λ_k.
   (c) For all $x \in S$
   For all transition classes $C_m = (G_m, u_m, \alpha_m)$
   i. If $u_m(x) \notin S$ then add $u_m(x)$ to S.
   ii. Set $prop := p(x) \cdot \alpha_m(x)/\lambda_k$.
   iii. Propagate probability $prop$ from x to $u_m(x)$ by setting $q(x) = q(x) - prop$ and $q(u_m(x)) = q(u_m(x)) + prop$.
   (d) For all states x in S
   i. If $p(x) < \Delta$, then remove x from S.

ii. Update probabilities by setting $p(x) := q(x)$.

(e) Update sum of coefficients by setting $sum = sum + \mathit{coeff}$.

(f) $k = k + 1$.

6. For all $x \in S$ set $r(x) := r(x) + \mathit{coeff} \cdot p(x)$.
7. Return r.

If a small threshold Δ is chosen, the proposed method gives accurate approximations for models where all populations are small. If the expected number of a certain population is high, then the number of significant states is large. In this case the memory requirements may exceed the memory capacities and the computation will take a long time to complete. Since high populations can be accurately approximated by deterministically and continuously changing variables, a stochastic hybrid model is more advantageous in such cases [23] which goes beyond the scope of this paper. Note that if one is interested in qualitative trends only, it is possible to get a rough idea of the dynamics of the system by choosing a much higher threshold Δ. This is similar to generating a small number of simulation runs in order to determine qualitative trends of the system.

Iteration Over Time. Our algorithm can be used in an iterative fashion to approximate the distribution of X at several time instances. To see this, first note that we can use the method described above for systems starting with arbitrary initial distributions by defining S_0 as the set of states that have an initial probability greater than Δ. After computing an approximation of $p^{(t)}(x)$ for all $x \in S$ we can use it as an initial distribution for the next step to obtain an approximation for $p^{(t')}(x)$ where $t' > t$ and the step size is $t' - t$. In this way, we obtain approximations for several time instances.

Related Work. Other approaches for an approximate numerical solution of the underlying Markov chains have been proposed [5, 32]. They differ from our approach in that they compute a finite projection of the state space that is based solely on the structure of the underlying graph. In our method, we add and neglect states in an on-the-fly fashion based on the stochastic properties of the Markov chain. Therefore, we consider a significantly smaller set of states during a certain time interval, without being less accurate. The projection algorithms include all states that are reachable within a fixed path depth. In our algorithm, for each single state, we dynamically decide if it significantly contributes to the overall solution or not. We have found this dynamic adaptation of the analysis to be essential for efficiency.

Gillespie simulation		
running time	single event error	# runs
> 500 h	10^{-8}	$> 3 \times 10^{10}$
> 50 h	10^{-7}	$> 3 \times 10^{9}$
> 5 h	10^{-6}	$> 3 \times 10^{8}$
> 30 min	10^{-5}	$> 3 \times 10^{7}$
> 3 min	10^{-4}	$> 3 \times 10^{6}$
> 18 sec	10^{-3}	$> 3 \times 10^{5}$

numerical approximation				
running time	total approx. error	$\lvert S_k \rvert$	$\max_k \lvert S_k \rvert$	Δ
10 min 31 sec	6×10^{-6}	1×10^{6}	2×10^{6}	10^{-14}
4 min 57 sec	2×10^{-5}	5×10^{5}	1×10^{6}	10^{-13}
2 min 12 sec	1×10^{-4}	3×10^{5}	6×10^{5}	10^{-12}
40 sec	5×10^{-4}	1×10^{5}	3×10^{5}	10^{-11}
15 sec	1×10^{-3}	5×10^{4}	1×10^{5}	10^{-10}

Table 1: Comparison of the running times for the signaling example.

5. Experimental Results

For our experimental results, we consider four examples from biology. Our first example is the model of intracellular signaling through receptors of the immune system considered in [15]. The second example is a model for the transcription regulation of a repressor protein in bacteriophage λ [18]. This protein is responsible for maintaining lysogeny of the λ virus in E. coli [1]. For both the first and the second example, we compute the full probability distribution for different precision levels. Our third example uses the gene expression model of Ex. 1. We calculate the distribution of the time until the number of produced proteins exceeds 500. The last example is the model of a genetic toggle switch in Escherichia coli presented in [11]. It is a prototype of a bistable system where the bistability arises from the mutually inhibitory arrangement of the repressor genes. Again, we compute the full probability distribution for different precision levels.

We implemented our direct numerical method as well as the Gillespie simula-

tion algorithm in a C++ tool called SABRE [7]. All our experiments are performed on a 3.16 GHz Intel Linux PC with 6 GB of RAM. There is no one-to-one correspondence between the statistical accuracy of the estimates that we derive via simulation and the precision of the numerical method. However, by assuming that the smallest event probability that has to be estimated is γ all results of the simulation have a "precision" of at least γ. Intuitively, we simulate often enough to reason about events that occur with a probability of at least γ. We therefore refer to γ as the *single event error*. Note that the simulation results are still subject to the statistical errors since the true values may not be covered by the confidence interval (compare Section 3.2).

The approximation error δ of the numerical method is the sum of the approximation error of *all* states in the Markov chain. Note that the probabilities of states not in S_k are underapproximated with zero and their true probabilities increase depending on how close they are to an attracting region. The error of a single state probability $p^{(t)}(x)$ is much smaller than δ but precise values for the single error are hard to obtain. A rough estimation of the single errors can be obtained by dividing the total error by the average size $|S_k|$ of the significant sets (cf. Table 2 and 3), even though δ may not be uniformly distributed on the significant set. On the other hand, δ also includes the error of insignificant states and, thus, distributes among much more states than only those in S_k.

We are comparing the two methods from the point of view of their running times. Another possibility would be to compare the memory consumption. Since we aim at computing the probability distribution of the underlying Markov chain, both methods have to store the probability of all states considered at some point in time. But this is, at least for systems with small populations, similar in both methods. We therefore focus on the running time of the algorithms.

Immune-Receptor Signaling. The signaling example involves 12 different chemical species and 19 reactions. After binding to a receptor a ligand undergoes six modifications and can generate a signal by activating a messenger [15]. Let $x = (x_1, \ldots, x_{12})$ and let $e_i \in \mathbb{N}_0^{12}$ be the vector with all entries zero except the i-th entry which is one. We define transition classes $C_i = (G_i, u_i, \alpha_i)$, $1 \leq i \leq 19$ as given below.

- Receptor-ligand binding: $G_1 = \{x \in \mathbb{N}_0^{12} \mid x_1 > 0, x_2 > 0\}$, $u_1(x) = x - e_1 - e_2 + e_3$, $\alpha_1(x) = c_1 x_1 x_2$.
- Forward modifications: For $j \in \{2, \ldots, 7\}$, we define $G_j = \{x \in \mathbb{N}_0^{12} \mid x_{j+1} > 0\}$, $u_j(x) = x - e_{j+1} + e_{j+2}$, $\alpha_j(x) = c_j x_{j+1}$.

- Backward modifications: For $j \in \{8, \ldots, 14\}$, we define $G_j = \{x \in \mathbb{N}_0^{12} \mid x_{j-5} > 0\}$, $u_j(x) = x - e_{j-5} + e_1 + e_2$, $\alpha_j(x) = c_j x_{j-5}$.
- Binding of inactive messengers: $G_{15} = \{x \in \mathbb{N}_0^{12} \mid x_9 > 0, x_{10} > 0\}$, $u_{15}(x) = x - e_9 - e_{10} + e_{11}$, $\alpha_{15}(x) = c_{15} x_9 x_{10}$.
- Unbinding of inactive messengers: $G_{16} = \{x \in \mathbb{N}_0^{12} \mid x_{11} > 0\}$, $u_{16}(x) = x - e_{11} + e_9 + e_{10}$, $\alpha_{16}(x) = c_{16} x_{11}$.
- Release of activated messengers: $G_{17} = \{x \in \mathbb{N}_0^{12} \mid x_{11} > 0\}$, $u_{17}(x) = x - e_{11} + e_9 + e_{12}$, $\alpha_{17}(x) = c_{17} x_{11}$.
- Unbinding of inactive messengers and ligands: $G_{18} = \{x \in \mathbb{N}_0^{12} \mid x_{11} > 0\}$, $u_{18}(x) = x - e_{11} + e_1 + e_2 + e_{10}$, $\alpha_{18}(x) = c_{18} x_{11}$.
- Inactivation of messengers: $G_{19} = \{x \in \mathbb{N}_0^{12} \mid x_{12} > 0\}$, $u_{19}(x) = x - e_{12} + e_{10}$, $\alpha_{19}(x) = c_{19} x_{12}$.

Following [15], the rate constants are chosen as $c_1 = 6.7 \cdot 10^{-3}$, $c_j = 0.25$ for $j \in \{2, \ldots, 7\}$, $c_j = 0.5$ for $j \in \{8, \ldots, 14\}$, $c_{15} = 1.2 \cdot 10^{-3}$, $c_{16} = 0.01$, $c_{17} = 100$, $c_{18} = 0.5$, $c_{19} = 2 \cdot 10^{-3}$ and the initial state is $x = (x_1, \ldots, x_{12})$ with $x_1 = 30$ ligands, $x_2 = 900$ receptors and $x_{10} = 10000$ messengers. We simulated the system over a time horizon of $t = 4$. In Table 1, we list the running times of our numerical method as well as the running time of the simulation. The column with header $|S_k|$ lists the average number of states in the sets $S_0, S_1, \ldots$ and $\max_k |S_k|$ lists the maximum over all these numbers of states. The columns with header Δ lists the threshold in Eq. 8.

Phage λ Model. The Phage λ model involves 6 different species and 10 reactions. Thus, a state is a vector $x = (x_1, x_2, x_3, x_4, x_5, x_6) \in \mathbb{N}_0^6$. The transition classes $C_i = (G_i, u_i, \alpha_i)$, $1 \leq i \leq 10$ are given as follows [18].

- Production of proteins: $G_1 = \{x \in \mathbb{N}_0^6 \mid x_3 > 0\}$, $u_1(x) = (x_1 + 1, x_2, x_3, x_4, x_5, x_6)$, $\alpha_1(x) = c_1 x_3$.
- Degradation of proteins: $G_2 = \{x \in \mathbb{N}_0^6 \mid x_1 > 0\}$, $u_2(x) = (x_1 - 1, x_2, x_3, x_4, x_5, x_6)$, $\alpha_2(x) = c_2 x_1$.
- Production of mRNA: $G_3 = \{x \in \mathbb{N}_0^6 \mid x_5 > 0\}$, $u_3(x) = (x_1, x_2, x_3 + 1, x_4, x_5, x_6)$, $\alpha_3(x) = c_3 x_5$.

Gillespie simulation		
running time	single event error	# runs
> 6000 h	10^{-8}	$> 3 \times 10^{10}$
> 500 h	10^{-7}	$> 3 \times 10^{9}$
67 h 22 min	10^{-6}	$> 3 \times 10^{8}$
6 h 44 min	10^{-5}	$> 3 \times 10^{7}$
40 min	10^{-4}	$> 3 \times 10^{6}$
4 min	10^{-3}	$> 3 \times 10^{5}$

numerical approximation				
running time	total approx. error	$\lvert S_k \rvert$	$\max_k \lvert S_k \rvert$	Δ
55 min 5 sec	3×10^{-6}	239792	722426	10^{-15}
39 min 16 sec	2×10^{-5}	187204	566141	10^{-14}
25 min 2 sec	2×10^{-4}	140969	427282	10^{-13}
15 min 41 sec	1×10^{-3}	101078	306130	10^{-12}
6 min 33 sec	7×10^{-3}	67540	202627	10^{-11}
3 min 12 sec	4×10^{-2}	40373	117392	10^{-10}

Table 2: Comparison of the running times for the phage λ model.

- Degradation of mRNA: $G_4 = \{x \in \mathbb{N}_0^6 \mid x_3 > 0\}$, $u_4(x) = (x_1, x_2, x_3 - 1, x_4, x_5, x_6)$, $\alpha_4(x) = c_4 x_3$.
- First dimer binding at operator site: $G_5 = \{x \in \mathbb{N}_0^6 \mid x_2, x_4 > 0\}$, $u_5(x) = (x_1, x_2 - 1, x_3, x_4 - 1, x_5 + 1, x_6)$, $\alpha_5(x) = c_5 x_2 x_4$.
- First dimer unbinding: $G_6 = \{x \in \mathbb{N}_0^6 \mid x_5 > 0\}$, $u_6(x) = (x_1, x_2 + 1, x_3, x_4 + 1, x_5 - 1, x_6)$, $\alpha_6(x) = c_6 x_5$.
- Second dimer binding at operator site: $G_7 = \{x \in \mathbb{N}_0^6 \mid x_2, x_5 > 0\}$, $u_7(x) = (x_1, x_2 - 1, x_3, x_4, x_5 - 1, x_6 + 1)$, $\alpha_7(x) = c_7 x_2 x_5$.
- Second dimer unbinding: $G_8 = \{x \in \mathbb{N}_0^6 \mid x_6 > 0\}$, $u_8(x) = (x_1, x_2 + 1, x_3,$

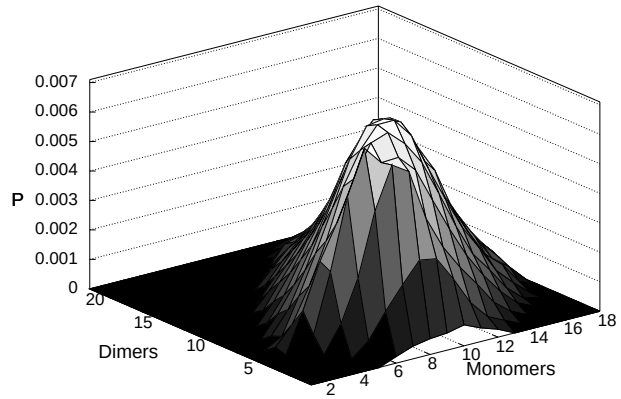

Figure 3: Probability distribution of monomers and dimers in the phage λ model.

$x_4, x_5 + 1, x_6 - 1)$, $\alpha_8(x) = c_8 x_6$.

- Dimerization: $G_9 = \{x \in \mathbb{N}_0^6 \mid x_1 > 1\}$, $u_9(x) = (x_1 - 2, x_2 + 1, x_3, x_4, x_5, x_6)$, $\alpha_9(x) = c_9 x_1(x_1 - 1)/2$.
- Dissociation into monomers: $G_{10} = \{x \in \mathbb{N}_0^6 \mid x_2 > 0\}$, $u_{10}(x) = (x_1 + 2, x_2 - 1, x_3, x_4, x_5, x_6)$, $\alpha_{10}(x) = c_{10} x_2$.

For $c_1, \ldots, c_{10}$, we choose $c_1 = 0.043$, $c_2 = 0.0007$, $c_3 = 0.0715$, $c_4 = 0.0039$, $c_5 = 1.992647 \times 10^{-2}$, $c_6 = 0.4791$, $c_7 = 1.992647 \times 10^{-4}$, $c_8 = 8.765 \times 10^{-12}$, $c_9 = 8.30269 \times 10^{-2}$, and $c_{10} = 0.5$ (see [5, 18]). The initial state of the system is given by $y = (2, 6, 0, 2, 0, 0)$ and the time horizon is $t = 300$. We approximate the probability distributions of the underlying CTMC at 100 equidistant time instances. Fig. 3 shows a plot of the distribution of dimers and monomers at time instant $t = 300$. In Table 2, we list the results of our numerical method as well as the simulation results.

Gene Expression. For the transition classes of the gene expression example we refer to Ex. 1. For the rate constants, we choose $c_1 = 0.05$, $c_2 = 0.0058$, $c_3 = 0.0029$, and $c_4 = 10^{-4}$, where c_3 and c_4 correspond to a half-life of 4 minutes for mRNA and 2 hours for the protein [49]. We compute the probability that at least 500 proteins are in the system at 100 equidistant time instances. Fig 4 shows the cumulative probability distribution of the time until the number of proteins reaches 500 for the first time (note that eventually the threshold of 500 is reached with probability one). In Table 3, we list the results for the gene expression example,

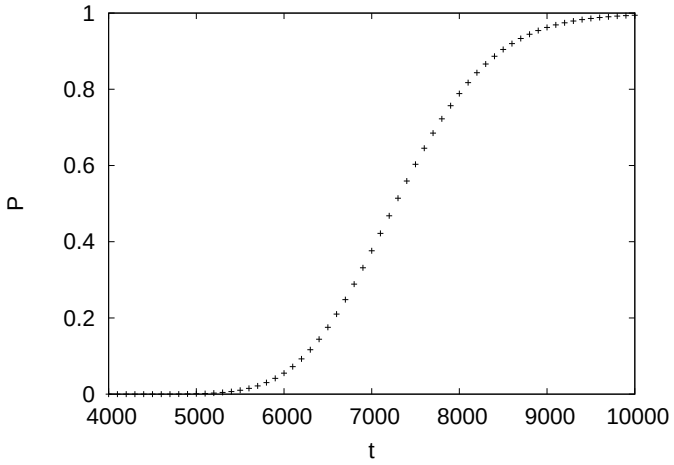

Figure 4: Cumulative probability distribution of the time until the number of proteins reaches 500 for the first time in the gene expression example.

where, as above, $|S_k|$ denotes the average number of states in the sets $S_0, S_1, \ldots$ and Δ is the threshold in Eq. 8.

Genetic Toggle Switch. The bistable toggle switch is a prototype of a genetic switch with two competing repressor proteins and four reactions [11]. The toggle switch involves two chemical species A and B and four reactions. Let $x = (x_1, x_2) \in \mathbb{N}_0^2$. The transition classes $C_i = (G_i, u_i, \alpha_i)$, $1 \leq i \leq 4$ are given as follows:

- $G_1 = N_0^2$, $u_1(x) = (x_1 + 1, x_2)$, $\alpha_1(x) = c_1/(c_2 + x_2^{\beta})$,
- $G_2 = \{x \in \mathbb{N}_0^2 \mid x_1 > 0\}$, $u_2(x) = (x_1 - 1, x_2)$, $\alpha_2(x) = c_3 \cdot x_1$,
- $G_3 = N_0^2$, $u_3(x) = (x_1, x_2 + 1)$, $\alpha_3(x) = c_4/(c_5 + x_1^{\gamma})$,
- $G_4 = \{x \in \mathbb{N}_0^2 \mid x_2 > 0\}$, $u_4(x) = (x_1, x_2 - 1)$, $\alpha_4(x) = c_6 \cdot x_2$.

For our experimental results, we chose the same parameters as Sjöberg et al. [45], that is, $c_1 = c_4 = 3 \cdot 10^3$, $c_2 = c_5 = 1.1 \cdot 10^4$, $c_3 = c_6 = 0.001$, and $\beta = \gamma = 2$. We used the initial state $x = (133, 133)$ and a time horizon of $t = 15000$. We present our experimental results in Table 4.

Gillespie simulation		
running time	single event error	# runs
> 500 h	10^{-8}	$> 3 \times 10^{10}$
> 50 h	10^{-7}	$> 3 \times 10^{9}$
> 5 h 3 min	10^{-6}	$> 3 \times 10^{8}$
> 30 min	10^{-5}	$> 3 \times 10^{7}$
> 3 min	10^{-4}	$> 3 \times 10^{6}$
> 15 sec	10^{-3}	$> 3 \times 10^{5}$

numerical approximation				
running time	total approx. error	$\lvert S_k \rvert$	max$\{S_k\}$	Δ
11 sec	2×10^{-6}	20919	23636	10^{-12}
10 sec	2×10^{-5}	19660	22469	10^{-11}
9 sec	2×10^{-4}	18180	20945	10^{-10}
7 sec	2×10^{-3}	16514	19273	10^{-9}
6 sec	2×10^{-2}	14707	17431	10^{-8}

Table 3: Comparison of the running times for the gene expression example.

Discussion. Even if we consider the total approximation error δ as a rough bound for the single error of each state probability, thus favoring simulation, the speed-up factor of the numerical approximation is large, especially if the precision increases. The necessary precision level up to which probability distributions are approximated may depend on the system under study. It is, however, important to note that the occurrence of rare biochemical events can have important effects. For instance, the spontaneous, epigenetic switching rate from the lysogenic state to the lytic state in phage λ-infected E. coli is experimentally estimated to be in the order of 10^{-7} per cell per generation [27].

6. Conclusion

We have demonstrated that, for the computation of event probabilities, a numerical reachability analysis provides an efficient alternative to simulation-based methods.

Gillespie simulation		
running time	single event error	# runs
$> 10^4$ h	10^{-8}	$> 3 \times 10^{10}$
$> 10^3$ h	10^{-7}	$> 3 \times 10^9$
> 116 h	10^{-6}	$> 3 \times 10^8$
> 11 h	10^{-5}	$> 3 \times 10^7$
> 1 h 10 min	10^{-4}	$> 3 \times 10^6$
> 7 min	10^{-3}	$> 3 \times 10^5$

numerical approximation				
running time	total approx. error	$\lvert S_k \rvert$	$\max\{S_k\}$	Δ
22 min 21 sec	6×10^{-6}	37919	42081	10^{-15}
19 min 26 sec	2×10^{-5}	35259	39372	10^{-14}
15 min 48 sec	1×10^{-4}	32521	36572	10^{-13}
12 min 29 sec	9×10^{-4}	29652	33618	10^{-12}
11 min 17 sec	9×10^{-3}	26635	30496	10^{-11}
9 min 41 sec	9×10^{-2}	23433	27136	10^{-10}

Table 4: Comparison of the running times for the genetic toggle switch example.

Even though simulation is widely used, the advantages of numerical methods increase as more sophisticated techniques become available. They reduce the computational effort, especially if accurate results are desired. Moreover, for the calibration of parameters many instances of the model have to be solved and in this case short running times for a single solution are necessary.

Until now we have analyzed examples of intrinsically stochastic systems that have been published in the literature. As future work, we are planning to apply our numerical reachability algorithm in collaboration with experimentalists working on new stochastic models. Moreover, we are planning to combine our numerical method with parameter estimation techniques.

Standard numerical reachability analysis methods are inefficient for large state spaces (in the case of high dimension and/or many molecules) and inapplicable for unbounded state spaces, and thus one resorts to simulation. We have demon-

strated that certain optimization techniques from computer science - localization, on the fly abstraction - put many examples within the reach of numerical reachability analysis. Indeed, when high accuracy is required these methods outperform simulation-based techniques.

[1] A. Arkin, J. Ross, and H. H. McAdams. Stochastic kinetic analysis of developmental pathway bifurcation in phage λ-infected E. coli cells. *Genetics*, 149:1633–1648, 1998.

[2] N. Barkai and S. Leibler. Biological rhythms: Circadian clocks limited by noise. *Nature*, 403:267–268, 2000.

[3] W. J. Blake, M. Kaern, C. R. Cantor, and J. J. Collins. Noise in eukaryotic gene expression. *Nature*, 422:633–637, 2003.

[4] P. Bremaud. *Markov Chains*. Springer, 1998.

[5] K. Burrage, M. Hegland, F. Macnamara, and R. Sidje. A Krylov-based finite state projection algorithm for solving the chemical master equation arising in the discrete modelling of biological systems. In *Proc. of the Markov 150th Anniversary Conference*, pages 21–38. Boson Books, 2006.

[6] F. Didier, T. A. Henzinger, M. Mateescu, and V. Wolf. Approximation of event probabilities in noisy cellular processes. In *CMSB*, pages 173–188, 2009.

[7] F. Didier, T. A. Henzinger, M. Mateescu, and V. Wolf. Sabre: A tool for stochastic analysis of biochemical reaction networks. In *QEST*. IEEE CS Press, 2010. To appear.

[8] M. B. Elowitz, M. J. Levine, E. D. Siggia, and P. S. Swain. Stochastic gene expression in a single cell. *Science*, 297:1183–1186, 2002.

[9] N. Fedoroff and W. Fontana. Small numbers of big molecules. *Science*, 297:1129–1131, 2002.

[10] B. L. Fox and P. W. Glynn. Computing Poisson probabilities. *Communications of the ACM*, 31(4):440–445, 1988.

[11] T. Gardner, C. Cantor, and J. Collins. Construction of a genetic toggle switch in Escherichia coli. *Nature*, 403:339 – 342, 2000.

[12] D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81(25):2340–2361, 1977.

[13] D. T. Gillespie. *Markov Processes*. Academic Press, N. Y., 1992.

[14] D. T. Gillespie. A rigorous derivation of the chemical master equation. *Physica A*, 188:404–425, 1992.

[15] B. Goldstein, J. R. Faeder, and W. S. Hlavacek. Mathematical and computational models of immune-receptor signalling. *Nat. Rev. Immunol.*, 4, 2004.

[16] D. Gonze, J. Halloy, and A. Goldbeter. Robustness of circadian rhythms with respect to molecular noise. *PNAS, USA*, 99(2):673–678, 2002.

[17] D. Gonze, J. Halloy, and A. Goldbeter. Stochastic models for circadian oscillations: Emergence of a biological rhythm. *Quantum Chemistry*, 98:228–238, 2004.

[18] J. Goutsias. Quasiequilibrium approximation of fast reaction kinetics in stochastic biochemical systems. *J. Chem. Phys.*, 122(18):184102, 2005.

[19] J. Hasty, J. Pradines, M. Dolnik, and J. J. Collins. Noise-based switches and amplifiers for gene expression. *PNAS USA*, 97:2075, 2000.

[20] A. Hellander. Efficient computation of transient solutions of the chemical master equation based on uniformization and quasi-Monte carlo. *J. Chem. Phys.*, 128(15):154109, 2008.

[21] D. A. Henderson, R. J. Boys, C. J. Proctor, and D. J. Wilkinson. Linking systems biology models to data: a stochastic kinetic model of p53 oscillations. In A. O'Hagan and M. West, editors, *Handbook of Applied Bayesian Analysis*. Oxford University Press, 2009.

[22] T. Henzinger, M. Mateescu, and V. Wolf. Sliding window abstraction for infinite Markov chains. In *Proc. CAV*, LNCS. Springer, 2009.

[23] T. A. Henzinger, M. Mateescu, L. Mikeev, and V. Wolf. Hybrid numerical solution of the chemical master equation, 2010.

[24] N. G. v. Kampen. *Stochastic Processes in Physics and Chemistry.* Elsevier, 3rd edition, 2007.

[25] A. Kierzek, J. Zaim, and P. Zielenkiewicz. The effect of transcription and translation initiation frequencies on the stochastic fluctuations in prokaryotic gene expression. *Journal of Biological Chemistry*, 276(11):8165–8172, 2001.

[26] A. Law and D. Kelton. *Simulation Modelling and Analysis*. McGraw-Hill Education, 2000.

[27] J. W. Little, D. P. Shepley, and D. W. Wert. Robustness of a gene regulatory circuit. *The EMBO Journal*, 18(15):4299–4307, 1999.

[28] R. Losick and C. Desplan. Stochasticity and Cell Fate. *Science*, 320(5872):65–68, 2008.

[29] H. Maamar, A. Raj, and D. Dubnau. Noise in gene expression determines cell fate in *Bacillus subtilis*. *Science*, 317(5837):526 – 529, 2007.

[30] H. H. McAdams and A. Arkin. Stochastic mechanisms in gene expression. *PNAS, USA*, 94:814–819, 1997.

[31] H. H. McAdams and A. Arkin. It's a noisy business! *Trends in Genetics*, 15(2):65–69, 1999.

[32] B. Munsky and M. Khammash. The finite state projection algorithm for the solution of the chemical master equation. *J. Chem. Phys.*, 124:044144, 2006.

[33] E. M. Ozbudak, M. Thattai, I. Kurtser, A. D. Grossman, and A. van Oudenaarden. Regulation of noise in the expression of a single gene. *Nature Genetics*, 31(1):69 – 73, 2002.

[34] P. Patel, B. Arcangioli, S. Baker, A. Bensimon, and N. Rhind. DNA replication origins fire stochastically in fission yeast. *Mol Biol Cell*, 17:308–316, 2006.

[35] J. Paulsson. Summing up the noise in gene networks. *Nature*, 427(6973):415–418, 2004.

[36] C. Rao, D. Wolf, and A. Arkin. Control, exploitation and tolerance of intracellular noise. *Nature*, 420(6912):231–237, 2002.

[37] W. Sandmann. Stochastic simulation of biochemical systems via discrete-time conversion. In *Proceedings of the 2nd Conference on Foundations of Systems Biology in Engineering*, pages 267–272. Fraunhofer IRB Verlag, 2007.

[38] W. Sandmann. Discrete-time stochastic modeling and simulation of biochemical networks. *Computational Biology and Chemistry*, 32(4):292 – 297, 2008.

[39] W. Sandmann. Rare event simulation methodologies in systems biology. In B. T. G. Rubino, editor, *Rare Event Simulation Using Monte Carlo Methods*, chapter 11, pages 243–265. John Wiley & Sons, 2009.

[40] W. Sandmann. Sequential estimation for prescribed statistical accuracy in stochastic simulation of biological systems. *Mathematical biosciences*, 221(1):43–53, 2009.

[41] W. Sandmann and C. Maier. On the statistical accuracy of stochastic simulation algorithms implemented in Dizzy. In *Proc. WCSB*, pages 153–156, 2008.

[42] W. Sandmann and V. Wolf. A computational stochastic modeling formalism for biological networks. In *Enformatika Transactions on Engineering, Computing and Technology*, volume 14, pages 132–137, 2006.

[43] W. Sandmann and V. Wolf. Computational probability for systems biology. In *Proc. FMSB*, volume 5054 of *LNCS*, pages 33–47. Springer, 2008.

[44] R. Sidje, K. Burrage, and S. MacNamara. Inexact uniformization method for computing transient distributions of Markov chains. *SIAM J. Sci. Comput.*, 29(6):2562–2580, 2007.

[45] P. Sjöberg, P. Lötstedt, and J. Elf. Fokker-Planck approximation of the master equation in molecular biology. *Computing and Visualization in Science*, 12:37–50, 2009.

[46] R. Srivastava, L. You, J. Summers, and J. Yin. Stochastic vs. deterministic modeling of intracellular viral kinetics. *Journal of Theoretical Biology*, 218:309–321, 2002.

[47] W. J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1995.

[48] P. S. Swain, M. B. Elowitz, and E. D. Siggia. Intrinsic and extrinsic contributions to stochasticity in gene expression. *PNAS, USA*, 99(20):12795–12800, 2002.

[49] M. Thattai and A. van Oudenaarden. Intrinsic noise in gene regulatory networks. *PNAS, USA*, 98(15):8614–8619, July 2001.

[50] T. E. Turner, S. Schnell, and K. Burrage. Stochastic approaches for modelling in vivo reactions. *Computational Biology and Chemistry*, 28:165–178, 2004.

[51] A. van Moorsel and W. Sanders. Adaptive uniformization. *ORSA Communications in Statistics: Stochastic Models*, 10(3):619–648, 1994.

[52] A. Warmflash and A. Dinner. Signatures of combinatorial regulation in intrinsic biological noise. *PNAS*, 105(45):17262–17267, 2008.

[53] D. J. Wilkinson. *Stochastic Modelling for Systems Biology*. Chapman & Hall, 2006.

[54] J. Zhang, L. T. Watson, and Y. Cao. A modified uniformization method for the solution of the chemical master equation., 2007. TR-07-31, Computer Science, Virginia Tech.