Rare Event Handling in Signalling Cascades

Benoît Barbot[1], Serge Haddad[1], Monika Heiner[2], and Claudine Picaronny[1]

[1] LSV, ENS Cachan & CNRS & Inria, France
{barbot, haddad, picaronny}@lsv.ens-cachan.fr
[2] Brandenburg University of Technology, Cottbus, Germany
monika.heiner@b-tu.de

Abstract Signalling cascades are a recurrent pattern of biological regulatory systems whose analysis has deserved a lot of attention. It has been shown that stochastic Petri nets are appropriate to model such systems and evaluate the probabilities of specific properties. Such an evaluation can be done numerically when the combinatorial state space explosion is manageable or statistically otherwise. However, when the probabilities to be evaluated are too small, random simulation requires more sophisticated techniques for the handling of rare events. Here we show how such involved methods can be successfully applied for signalling cascades. More precisely, we study three relevant properties of a signalling cascade with the help of the Cosmos tool. Our experiments point out interesting dependencies between quantitative parameters of the regulatory system and its transient behaviour. In addition, they demonstrate that we can go beyond the capabilities of Marcie, which provides one of the most efficient numerical solvers.

1 Introduction

Signalling cascades. Signalling processes play a crucial role for the regulatory behaviour of living cells. They mediate input signals, i.e. the extracellular stimuli received at the cell membrane, to the cell nucleus, where they enter as output signals the gene regulatory system. Understanding signalling processes is still a challenge in cell biology. To approach this research area, biologists design and explore signalling networks, which are likely to be building blocks of the signalling networks of living cells. Among them are the type of signalling cascades which we investigate in our paper.

A signalling cascade is a set of reactions which can be grouped into levels. At each level a particular enzyme is produced (e.g. by phosphorylation); the level generally also includes the inverse reactions (e.g., dephosphorylation). The system constitutes a cascade since the enzyme produced at some level is the catalyser for the reactions at the next level. The catalyser of the first level is usually considered to be the input signal, while the catalyser produced by the last level constitutes the output signal. The transient behaviour of such a system presents a characteristic shape, the quantity of every enzyme increases to some stationary value. In addition, the increases are temporally ordered w.r.t.

the levels in the signalling cascade. This behaviour can be viewed as a signal travelling along the levels, and there are many interesting properties to be studied like the travelling time of the signal, the relation between the variation of the enzymes of two consecutive levels, etc.

In [11], it has been shown how such a system can be modelled by a Petri net which can either be equipped with continuous transition firing rates leading to a continuous Petri net which determines a set of differential equations or by stochastic transition firing rates leading to a stochastic Petri net. This approach emphasises the importance of Petri nets which, depending on the chosen semantics, permit to investigate particular properties of the system. In this paper we wish to explore the influence of stochastic features on the signalling behaviour, and thus we focus on the use of stochastic Petri nets.

Analysis of stochastic Petri nets can be performed either numerically or statistically. The former approach is much faster than the latter and provides exact results up to numerical approximations, but its application is limited by the memory requirements due to the combinatory explosion of the state space.

Statistical evaluation of rare events. Statistical analysis means to estimate the results by evaluating a sufficient number of simulations. However, standard simulation is unable to efficiently handle *rare events*, i.e. properties whose probability of satisfaction is tiny. Indeed the number of trajectories to be generated in order to get an accurate interval confidence for rare events becomes prohibitively huge. Thus *acceleration* techniques [16] have been designed to tackle this problem whose principles consist in (1) favouring trajectories that satisfy the property, and (2) numerically adjusting the result to take into account the bias that has been introduced. This can be done by *splitting* the most promising trajectories [14] or *importance sampling* [10], i.e. modifying the distribution during the simulation. In previous work [4], some of us have developed an original importance sampling method based on the design and numerical analysis of a reduced model in order to get the importance coefficients. First proposed for checking "unbounded until" properties over models whose semantics is a discrete time Markov chain, it has been extended to also handle "bounded until" properties and continuous time Markov chains [5].

Our contribution. In this paper we consider three families of properties for signalling cascades that are particularly relevant for the study of their behaviour and that are (depending on a scaling parameter) potentially rare events. From an algorithmic point of view, this case study raises interesting issues since the combinatorial explosion of the model quickly forbids the use of numerical solvers and its intricate (quantitative) behaviour requires elaborated and different abstractions depending on the property to be checked.

Due to these technical difficulties, the signalling cascade analysis has led us to substantially improve our method and in particular the way we obtain the final confidence interval. From a biological point of view, experiments have pointed out interesting dependencies between the scaling parameter of the model and the probability of satisfying a property.

Organisation. In Section 2 we present the biological background, the signalling cascade under study and its properties. Then in Section 3 after some recalls on Markov chains and stochastic Petri nets, we model signalling cascades by SPNs. In Section 4, we develop our method for handling rare events. Then in Section 5 we report and discuss the results of our experiments. Finally in Section 6, we conclude and give some perspectives to our work. Additional details about algorithmic considerations are provided in the appendix.

2 Signalling cascades

In technical terms, signalling cascades can be understood as networks of biochemical reactions transforming input signals into output signals. By this way, signalling processes determine crucial decisions a cell has to make during its development, such as cell division, differentiation, or death. Malfunction of these networks may potentially lead to devastating consequences on the organism, such as outbreak of deceases or immunological abnormalities. Therefore, cell biology tries to increase our understanding of how signalling cascades are structured and how they operate. However, signalling networks are generally hard to observe and often highly interconnected, and thus signalling processes are not easy to follow. For this reason, typical building blocks are designed instead, which are able to reproduce observed input/output behaviours.

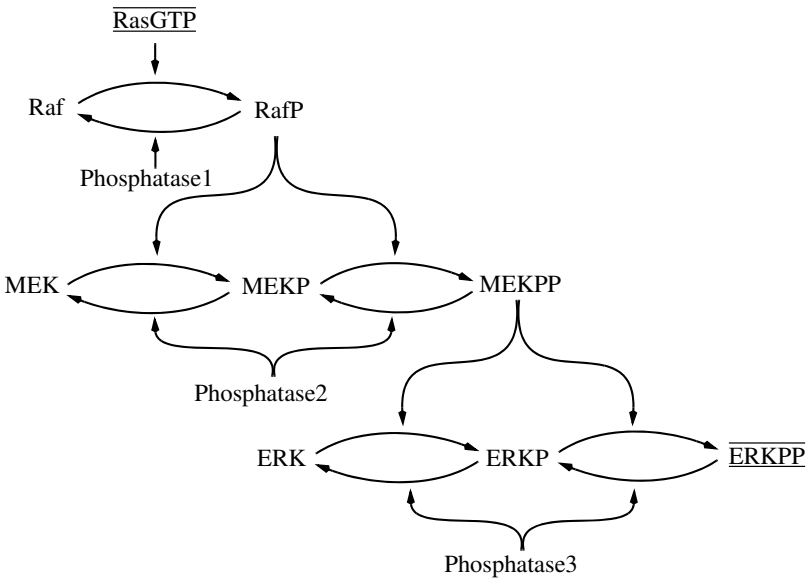

Figure 1. The general scheme of the considered three-level signalling cascade; RasGTP serves as input signal and ERKPP as output signal.

The case study we have chosen for our paper is such a signalling building block: the mitogen-activated protein kinase (MAPK) cascade [15]. This is the core of the ubiquitous ERK/MAPK network that can, among others, convey cell division and differentiation signals from the cell membrane to the nucleus. The description starts at the RasGTP complex which acts as an enzyme (kinase) to phosphorylate Raf, which phosphorylates MAPK/ERK Kinase (MEK), which in turn phosphorylates Extracellular signal Regulated Kinase (ERK). We consider RasGTP as the input signal and ERKPP (activated ERK) as the output signal. This cascade (RasGTP $\rightarrow$ Raf $\rightarrow$ MEK $\rightarrow$ ERK) of protein interactions is known to control cell differentiation, while the strength of the effect depends on the ERK activity.

The scheme in Figure 1 describes the typical modular structure for such a signalling cascade, see [7]. Each layer corresponds to a distinct protein species. The protein Raf in the first layer is only singly phosphorylated. The proteins in the two other layers, MEK and ERK respectively, can be singly as well as doubly phosphorylated. In each layer, forward reactions are catalysed by kinases and reverse reactions by phosphatases (Phosphatase1, Phosphatase2, Phosphatase3). The kinases in the MEK and ERK layers are the phosphorylated forms of the proteins in the previous layer. Each phosphorylation/dephosphorylation step applies mass action kinetics according to the pattern $A+E \rightleftharpoons AE \rightarrow B+E$. This pattern reflects the mechanism by which enzymes act: first building a complex with the substrate, which modifies the substrate to allow for forming the product, and then disassociating the complex to release the product; for details see [6].

Having the wiring diagram of the signalling cascade, a couple of interesting questions occur whose answers would shed some additional light on the subject under investigation. Among them are an assessment of the signal strength in each level, and specifically of the output signal. We will consider these properties in Sections 5.1 and 5.2. The general scheme of the signalling cascade also suggests a temporal order of the signal propagation in accordance with the level order. What cannot be derived from the structure is the extend to which the signals are built simultaneously; we will discuss this in Section 5.3.

3 Petri net modelling

3.1 Stochastic Petri nets

Due to its graphical representation and bipartite nature, Petri nets are highly appropriate to model biochemical networks. When equipped with a stochastic semantics, yielding stochastic Petri nets (SPN) [1], they can be used to perform quantitative analysis.

Definition 1 (SPN). *A stochastic Petri net $\mathcal{N}$ is defined by:*

- *a finite set of places P;*
- *a finite set of transitions T;*
- *a backward (resp. forward) incidence matrix* **Pre** *(resp.* **Post***) from $P \times T$ to $\mathbb{N}$.*

– *a set of state-dependent rates of transitions* $\{\mu_t\}_{t\in T}$ *such that* μ_t *is a mapping from* $\mathbb{N}^P$ *to* $\mathbb{R}_{>0}$.

A *marking* m of an SPN $\mathcal{N}$ is an item of $\mathbb{N}^P$. A transition t is fireable in marking m if for all $p \in P$ $m(p) \geq \mathbf{Pre}(p,t)$. Its firing leads to marking m' defined by: for all $p \in P$ $m'(p) = m(p) - \mathbf{Pre}(p,t) + \mathbf{Post}(p,t)$. It is denoted either as $m \xrightarrow{t} m'$ or as $m \xrightarrow{t}$ omitting the next marking. Let $\sigma = \sigma_1 \dots \sigma_n \in T^*$, then σ is fireable from m and leads to m' if there exists a sequence of markings $m = m_0, m_1, \dots, m_n$ such that for all $0 \leq k < n$, $m_k \xrightarrow{\sigma_k} m_{k+1}$. This firing is also denoted $m \xrightarrow{\sigma} m'$. Let m_0 be an initial marking, the reachability set $Reach(\mathcal{N}, m_0)$ is defined by: $Reach(\mathcal{N}, m_0) = \{m \mid \exists \sigma \in T^* \ m_0 \xrightarrow{\sigma} m\}$. The initialised SPNs $(\mathcal{N}, m_0)$ that we consider do not have deadlocks: for all $m \in Reach(\mathcal{N}, m_0)$ there exists $t \in T$ such that $m \xrightarrow{t}$.

The semantics of an SPN is a continuous time Markov chain (CTMC). For further developments, in particular the use of uniformisation, we also introduce discrete time Markov chains (DTMC).

Definition 2 (DTMC). *A discrete time Markov chain* $\mathcal{C}$ *is defined by:*

– *a set of states* S *with an initial state* s_0*;*
– *a transition probability matrix* $\mathbf{P}$ *of size* $S \times S$.

The state of the chain at (discrete) time n *is a random variable* S_n *defined inductively by* $\Pr(S_0 = s_0) = 1$ *and,*

$$\Pr(S_{n+1} = s' \mid S_n = s, S_{n-1} = s_{n-1}, \dots, S_0 = s_0) = \Pr(S_{n+1} = s' \mid S_n = s) = \mathbf{P}(s, s')$$

A continuous time Markov Chain is a DTMC in which exit rates are associated with states. A rate assigns to an exponential distribution to the sojourn time in the corresponding state.

Definition 3 (CTMC). *A continuous time Markov Chain* $\mathcal{C}$ *is defined by:*

– *a set of states* S *with an initial state* s_0*;*
– *a transition matrix* $\mathbf{P}$ *of size* $S \times S$.
– *a rate* λ_s *associated with every state* $s \in S$*;*

The behaviour of $\mathcal{C}$ *is defined by two families of random variables:* $\{S_n\}_{n\in\mathbb{N}}$ *and* $\{T_n\}_{n\in\mathbb{N}}$. S_n *denotes the state of the chain after* n *steps while* T_n *denotes the time elapsed in state* S_n *before the next step.* S_n *is defined as for a DTMC while* T_n *is defined by:*

$$\begin{aligned}\Pr(T_n \leq \tau \mid S_0 = s_0, ..., S_n = s, T_0 \leq \tau_0, ..., T_{n-1} \leq \tau_{n-1}) \\ = \Pr(T_n \leq \tau \mid S_n = s) = 1 - e^{\lambda_s \cdot \tau}\end{aligned}$$

The DTMC defined by $\mathbf{P}$ is called *embedded chain*. It observes the change of state, independently of the time elapsed in the state. In the following, we assume that the rates are bounded, i.e. there exists some finite μ such that for every s, $\lambda_s \leq \mu$. Using this hypothesis, it can be shown that for every $\tau \geq 0$,

the following random variable X_τ is defined almost everywhere and denotes the state of $\mathcal{C}$ at (continuous) time τ.

$$X_\tau = S_{N(\tau)} \text{ where } N(\tau) = \min(\{n \mid \sum_{k=0}^{n} T_k > \tau\})$$

Let $\pi(\tau)$ be the distribution of X_τ. Then $\pi(\tau)$ fulfills the following differential equation system:

$$\frac{d\pi}{d\tau} = \pi\mathbf{Q} \text{ where for all } s' \neq s \ \mathbf{Q}(s,s') = \lambda_s \mathbf{P}(s,s') \text{ and } \mathbf{Q}(s,s) = -\sum_{s' \neq s} \mathbf{Q}(s,s')$$

Thus $\{\pi(\tau)\}_{\tau \in \mathbb{R}_{\geq 0}}$ is fully defined by the *infinitesimal generator* $\mathbf{Q}$.

An SPN is a high-level model whose operational semantics is a CTMC $\mathcal{C}$. In a state, enabled transition of the Petri net randomly selects an execution time according to a Poisson process. Then the transition with earliest firing is selected to fire yielding the new marking.

Definition 4 (CTMC of a SPN). *Let $\mathcal{N}$ be a stochastic Petri net and m_0 be an initial marking. Then the CTMC associated with $(\mathcal{N}, m_0)$ is defined by:*

- *the set of states is Reach$(\mathcal{N}, m_0)$;*
- *the transition matrix $\mathbf{P}$ is defined by:*

$$\mathbf{P}(m,m') = \frac{\sum_{m \xrightarrow{t} m'} \mu_t(m)}{\sum_{m \xrightarrow{t}} \mu_t(m)}$$

- *the rate λ_m is defined by: $\lambda_m = \sum_{m \xrightarrow{t}} \mu_t(m)$*

3.2 Running case study

We now explain how to model our running case study in the Petri net framework. The signalling cascade is made of several phosphorylation/dephosphorylation steps, which are built on mass/action kinetics. Each step follows the pattern $A + E \rightleftharpoons AE \to B + E$ and is modelled by a small Petri net component depicted in Figure 2. The mass action kinetics is expressed by the rate of the transitions. The marking-dependent rate of each transition is equal to the product of the number of tokens in all its incoming places up to a multiplicative constant given by the biological behaviour (summing up dependencies on temperature, pressure, volume, etc.).

The whole reaction network based on the general scheme of a three-level double phosphorylation cascade, as given in Figure 1, is modelled by the Petri net in Figure 3. The input signal is the number of tokens in the place RasGTP and the output signal is the number of tokens in the place ERKPP.

This signalling cascade model represents a self-contained and closed system. It is covered with P- and T-invariants, specifically each layer in the cascade forms

a P-invariant consisting of all states a protein can undergo; thus the model is bounded. Assuming an appropriate initial marking, the model is also live and reversible; see [11] for more details, where this Petri net has been developed and analysed in the qualitative, stochastic and continuous modelling paradigms. In our paper we extend these analysis techniques for handling properties corresponding to rare events.

We introduce a scaling factor N to parameterise how many tokens are spent to specify the initial marking. Increasing the scaling parameter can be interpreted in two different ways: either an increase of the biomass circulating in the closed system (if the biomass value of one token is kept constant), or an increase of the resolution (if the biomass value of one token inversely decreases, called level concept in [11]). The kind of interpretation does not influence the approach we pursue in this paper.

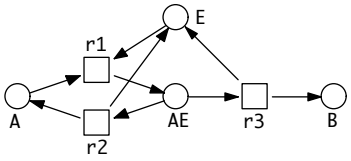

Figure 2. Petri net pattern for mass action kinetics $A + E \rightleftharpoons AE \to B + E$.

Table 1. Development of the state space for increasing N.

N	number of states	N	number of states
1	24,065 (4)	6	769,371,342,640 (11)
2	6,110,643 (6)	7	5,084,605,436,988 (12)
3	315,647,600 (8)	8	27,124,071,792,125 (13)
4	6,920,337,880 (9)	9	122,063,174,018,865 (14)
5	88,125,763,956 (10)	10	478,293,389,221,095 (14)

Increasing N means to increase the size of the state space and thus of the CTMC, as shown in Table 1 which has been computed with the symbolic analysis tool MARCIE [12]. As expected, the explosion of the state space prevents numerical model checking for higher N and thus calls for statistical model checking.

Furthermore, increasing the number of states means to actually decrease the probabilities to be in a certain state, as the total probability of 1 is fixed. With the distribution of the probability mass of 1 over an increasingly huge number of states, we obtain sooner or later states with very tiny probabilities, and thus rare events. Neglecting rare events is usually appropriate when focusing on the averaged behaviour. But they become crucial when certain jump processes such as mutations under rarely occurring conditions are of interest.

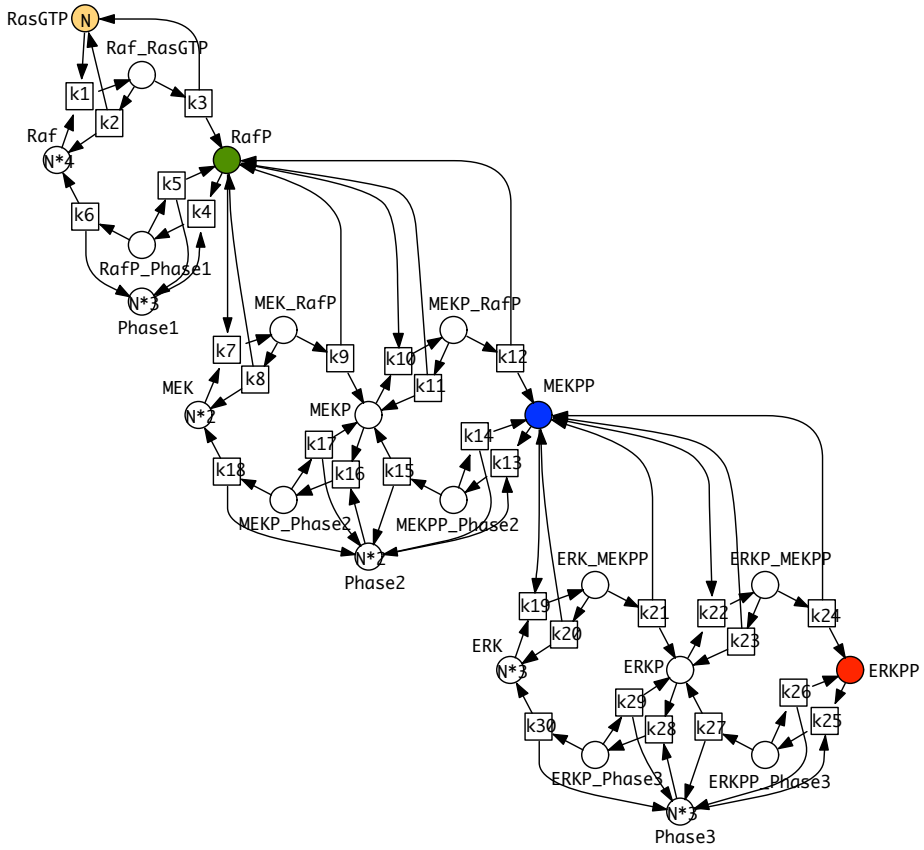

Figure 3. Petri net modelling the three-level signalling cascade given in Figure 1; k_i are the kinetic constants for mass action kinetics, N the scaling parameter.

4 Statistical model checking with rare events

Our method is based on our previous works [4,5] but improves them in some aspects. It requires several technical developments organised in five sections.

- In Section 4.1, we recall standard results about Markov chains and temporal logic LTL.
- In Section 4.2, we recall statistical simulation with a focus on rare events and importance sampling.
- In Section 4.3, we explain the core of our approach: performing an importance sampling based on an abstract model.
- In Section 4.4, we depict all the stages of the complete methodology.
- The memory requirements – while much lesser than for numerical evaluation – may still become inhibitory. Thus, we propose in Section 4.5 several algorithms offering a trade-off between time and space.

4.1 Markov chain recalls for model checking

Uniformisation. A CTMC is said to be *uniform* when the rate $\lambda = \lambda_s$ is independent from s. Given a uniform chain, the distribution $\pi(\tau)$ is obtained by the following formula:

$$\pi(\tau) = \sum_{n \geq 0} \frac{e^{-\lambda\tau}(\lambda\tau)^n}{n!} \mathbf{P}^n(s_0, s)$$

Indeed using the uniform hypothesis, $\frac{e^{-\lambda\tau}(\lambda\tau)^n}{n!}$ is the probability that n transitions take place in interval $[0, \tau]$ and $\mathbf{P}^n(s_0, s)$ is the probability to be in state s after n transitions.

Given a non-uniform chain with bounded rates, it is a standard approach to transform it in a uniform Markov chain with the same distribution $\pi(\tau)$. One selects some upper bound of the rates (say λ), considers λ as the uniform transition rate and sets a transition matrix $\mathbf{P}^{(\lambda)}$ defined by:

$$\forall s \neq s' \in S \ \mathbf{P}^{(\lambda)}(s, s') = \frac{\lambda_s}{\lambda}\mathbf{P}(s, s') \text{ and } \mathbf{P}^{(\lambda)}(s, s) = 1 - \sum_{s' \neq s} \mathbf{P}^{(\lambda)}(s, s')$$

In [8], an accurate computation of $\pi(\tau)$ is designed by truncating the infinite sum. Given two numerical requirements α and β, truncation points n^- and n^+ and values $\{c_n\}_{n^- \leq n \leq n^+}$ are determined such that:

$$\forall n^- \leq n \leq n^+ \ c_n(1 - \alpha - \beta) \leq \frac{e^{-\lambda\tau}(\lambda\tau)^n}{n!} \leq c_n \text{ and}$$

$$\sum_{n < n^-} \frac{e^{-\lambda\tau}(\lambda\tau)^n}{n!} \leq \alpha \text{ and } \sum_{n > n^+} \frac{e^{-\lambda\tau}(\lambda\tau)^n}{n!} \leq \beta$$

Model checking of Markov chains. In the context of model checking, the states of a chain $\mathcal{C}$ are labelled with atomic propositions that they fulfil. Given state s, $\alpha(s)$ denotes the set of propositions satisfied by s. We denote $S_x = \{s \in S \mid x \in \alpha(s)\}$, $S_{\overline{x}} = \{s \in S \mid x \notin \alpha(s)\}$, $S_{xy} = \{s \in S \mid x \in \alpha(s) \wedge y \in \alpha(s)\}$, etc.

The problem we address here is the computation of the probability that a random path starting from a fixed state s (and in particular from the initial state) satisfies a formula φ^I where φ is a formula of temporal logic HASL [3] (a variant of LTL) that can be translated in a finite automaton and $I = [0, \tau]$ is an interval. For sake of conciseness and readability, we only detail the case $\varphi = aUb$ and informally explain how to handle a general formula. Recall that a path satisfies $aU^{[0,\tau]}b$ if there exists $\tau' \leq \tau$ such that a is satisfied along the path at every instant less than τ' and b is satisfied at instant τ'.

When $\mathcal{C}$ is a DTMC, let us denote $\mu_n(s)$, the probability that a random path starting from s fulfils $aU^{[0,n]}b$. These probabilities can be shown to be the solution of the following system of equations ($\mathbf{1}_E$ denotes the indicator function of set E).

$$\begin{cases} \forall u \ \forall s \in S_{\overline{ab}} & \mu_u(s) = 0 \\ \forall u \ \forall s \in S_b & \mu_u(s) = 1 \text{ and } \mu_0 = \mathbf{1}_{S_b} \\ \forall u > 0 \ \forall s \in S_{a\overline{b}} & \mu_u(s) = \sum_{s' \in S} \mathbf{P}(s, s')\mu_{u-1}(s') \end{cases} \quad (1)$$

Observe that $\mu_u(s)$ is increasing w.r.t. u and $\mu_\infty(s) = \lim_{u\to\infty} \mu_u(s)$ is the probability that a random path starting from s fulfils aUb.

The previous equations suggest a way to transform the model checking problem into a reachability problem which is more appropriate for our method. More precisely we interpret $\mu_n(s)$ in a DTMC $\mathcal{C}$ as a reachability probability in a DTMC $\mathcal{C}_n$ defined as follows.

- $S_n = S_{a\bar{b}} \times [1, n] \cup \{s_-, s_+\}$
- s_-, s_+ are *absorbing* states: $\mathbf{P}_n(s_-, s_-) = \mathbf{P}_n(s_+, s_+) = 1$
- $\forall s, s' \in S_{a\bar{b}}\ \forall u > 1\ \mathbf{P}_n((s, u), (s', u-1)) = \mathbf{P}(s, s')$, $\mathbf{P}_n((s, u), s_-) = \sum_{s' \in S_{\overline{ab}}} \mathbf{P}(s, s')$, $\mathbf{P}_n((s, u), s_+) = \sum_{s' \in S_b} \mathbf{P}(s, s')$
- $\forall s\ \mathbf{P}_n((s, 1), s_+) = \sum_{s' \in S_b} \mathbf{P}(s, s')$, $\mathbf{P}_n((s, 1), s_-) = 1 - \mathbf{P}_n((s, 1), s_+)$
- The other transition probabilities are null.

Observe that the probability to reach s_+ or s_- from any state is equal to 1. Moreover by construction, it holds $\mu(s, u) = \mu_u(s)$ where $\mu(s, u)$ is the probability to reach s_+ in $\mathcal{C}_n$ starting from (s, u).

When $\mathcal{C}$ is a CTMC, let denote $\mu_\tau(s)$ the probability that a random path starting from s fufills $aU^{[0,\tau]}b$. Observe that $\mu_\infty(s)$ only depends on the embedded DTMC (and in fact is equal to the corresponding value in the DTMC). Using uniformisation with bounding rate λ, $\mu_\tau(s)$ fulfils:

$$\mu_\tau(s) = \sum_{n \in \mathbb{N}} \frac{e^{-\lambda\tau}(\lambda\tau)^n}{n!} \mu_n(s)$$

where $\mu_n(s)$ corresponds to the satisfaction probability of $aU^{[0,n]}b$ in the embedded DTMC of the uniformised CTMC.

In case of a formula specified by a *deterministic* finite automaton, the *synchronised product* of the Markov chain and the automaton is still a Markov chain with the addition of an absorbing rejection state and merging of all accepting states into another absorbing state. Then, the considered probability is the probability to reach the accepting state in the interval $[0, \tau]$.

4.2 Statistical model checking and rare events

Simulation recalls. The statistical approach for evaluating the expectation $\mathbf{E}(X)$ of a random variable X related to a random path in a Markov chain is generally based on three parameters: the number of simulations K, the confidence level γ, and the width of the confidence interval lg (see [2]). Once the user provides two parameters, the procedure computes the remaining one. Then it performs K simulations of the Markov chain and outputs a confidence interval $[L, U]$ with a width of at most lg such that $\mathbf{E}(X)$ belongs to this interval with a probability of at least γ. More precisely, depending on the hypotheses, the confidence level has two interpretations: (1) either the confidence level is truly ensured, or (2) the required probability is only asymptotically valid (when K goes to infinity using

central limit theorem). The two usual hypotheses for providing a *true* confidence level are either that the distribution of X is known up to a parameter (e.g. Bernoulli law with unknown success probability) or that the random variable is bounded allowing to exploit Chernoff-Hoeffding bounds [13].

Statistical evaluation of a reachability probability. Let $\mathcal{C}$ be a DTMC with two absorbing states s_+ or s_-, such that the probability to reach s_+ or s_- from any state is equal to 1. Assume one wants to estimate p, the probability to reach s_+. Then the simulation step consists in generating K paths of $\mathcal{C}$ which end in an absorbing state. Let K_+ be the number of paths ending in state s_+. The random variable K_+ follows a binomial distribution with parameters p and K. Thus the random variable $\frac{K_+}{K}$ has a mean value p and since the distribution is parametrised by p, a true confidence level can be ensured. Unfortunately, when $p \ll 1$, the number of paths required for a small confidence interval is too large to be simulated. This issue is known as the *rare event* problem.

Importance sampling. In order to tackle the rare event problem, the importance sampling method uses for the generation of paths a modified DTMC $\mathcal{C}'$, with the same state space, but modified transition matrix $\mathbf{P}'$. $\mathbf{P}'$ must satisfy:

$$\mathbf{P}(s, s') > 0 \Rightarrow \mathbf{P}'(s, s') > 0 \vee s' = s_- \tag{2}$$

which means that this modification cannot remove transitions that have not s_- as target, but can add new transitions. The method maintains a correction factor called L initialised to 1; this factor represents the *likelihood* of the path. When a path crosses a transition $s \to s'$ with $s' \neq s_-$, L is updated by $L \leftarrow L\frac{\mathbf{P}(s,s')}{\mathbf{P}'(s,s')}$. When a path reaches s_-, L is set to zero. If $\mathbf{P}' = \mathbf{P}$ (i.e. no modification of the chain), the value of L when the path reaches s_+ (resp. s_-) is 1 (resp. 0). Let V_s (resp. W_s) be the random variable associated with the final value of L for a path starting in x in the original model $\mathcal{C}$ (resp. in $\mathcal{C}'$). By definition, the expectation $\mathbf{E}(V_{s_0}) = p$ and by construction of the likelihood, $\mathbf{E}(W_{s_0}) = p$. Of course, a useful importance sampling should reduce the variance of W_{s_0} w.r.t. to the one of V_{s_0} equal to $p(1-p)$.

4.3 Abstraction for importance sampling

The importance sampling method relies on a choice of a biased distribution that will artificially increase the frequency of the observed rare event during the simulation. The choice of this distribution is crucial for the efficiency of the method and usually cannot be found without a deep understanding of the system to be studied.

The core of our method deals with a DTMC on which it computes the probability to satisfy formula $aU^{[0,n]}b$. It relies on building an abstract smaller (but still close) DTMC that we call the *reduced model*. On this reduced model, we perform numerical computations to obtain the distribution corresponding to the importance sampling.

Definition 5. *Let $\mathcal{C}$ be a DTMC, a DTMC $\mathcal{C}^\bullet$ is called a* reduction *of $\mathcal{C}$ by a function f that maps S to $S^\bullet$, the state space of $\mathcal{C}^\bullet$, if for all $s \in S$:*

- *$a \in \alpha(s)$ (resp. $b \in \alpha(s)$) iff $a \in \alpha^\bullet(f(s))$ (resp. $b \in \alpha^\bullet(f(s))$)*
- *$\forall 0 < u \leq n\ \mu_u^\bullet(f(s)) = 0 \Rightarrow \mu_u(s) = 0$*
  where $\mu_u^\bullet(s^\bullet)$ denotes the probability that a random path in $\mathcal{C}^\bullet$ starting from $s^\bullet$ satisfies $aU^{[0,u]}b$.

Given $s \in S$ and $0 < u \leq n$, let $h_u(s) = \frac{1}{\mu_u^\bullet(f(s))} \sum_{s' \in S} \mathbf{P}(s,s')\mu_{u-1}^\bullet(f(s'))$. The next definition introduces an important sampling based on the reduced chain.

Definition 6. *Let $\mathcal{C}$ be a DTMC and $\mathcal{C}^\bullet$ be a reduction of $\mathcal{C}$ by f. Then $\mathbf{P}'_n$ is the transition matrix on S_n the state space of $\mathcal{C}_n$ defined by:*
Let s be a state of $S_{a\bar{b}}$ and $0 < u \leq n$,

1. *if $\mu_u^\bullet(f(s)) = 0$ then for all $s' \in S_n$, $\mathbf{P}'_n((s,u),s') = \mathbf{P}_n((s,u),s')$*
2. *if $\mu_u^\bullet(f(s)) > 0$ and $u > 1$ and $h_u(s) \leq 1$ then*
   a. *$\forall s' \in S_{a\bar{b}}\ \mathbf{P}'_n((s,u),(s',u-1)) = \frac{\mu_{u-1}^\bullet(f(s'))}{\mu_u^\bullet(f(s))}\mathbf{P}(s,s')$*
   b. *$\mathbf{P}'_n((s,u),s_+) = \frac{1}{\mu_u^\bullet(f(s))} \sum_{s' \in S_b} \mathbf{P}(s,s')$*
   c. *$\mathbf{P}'_n((s,u),s_-) = 1 - h_u(s)$*
3. *if $\mu_u^\bullet(f(s)) > 0$ and $u > 1$ and $h_u(s) > 1$ then*
   a. *$\forall s' \in S_{a\bar{b}}\ \mathbf{P}'_n((s,u),(s',u-1)) = \frac{\mu_{u-1}^\bullet(f(s'))}{h_u(s)\mu_u^\bullet(f(s))}\mathbf{P}(s,s')$*
   b. *$\mathbf{P}'_n((s,u),s_+) = \frac{1}{h_u(s)\mu_u^\bullet(f(s))} \sum_{s' \in S_b} \mathbf{P}(s,s')$*
   c. *$\mathbf{P}'_n((s,u),s_-) = 0$.*
4. *if $\mu_1^\bullet(f(s)) > 0$ and $h_1(s) \leq 1$ then*
   a. *$\mathbf{P}'_n((s,1),s_+) = \frac{1}{\mu_1^\bullet(f(s))} \sum_{s' \in S_b} \mathbf{P}(s,s')$*
   b. *$\mathbf{P}'_n((s,1),s_-) = 1 - h_1(s)$.*
5. *if $\mu_1^\bullet(f(s)) > 0$ and $h_1(s) > 1$ then*
   a. *$\mathbf{P}'_n((s,1),s_+) = \frac{1}{h_1(s)\mu_1^\bullet(f(s))} \sum_{s' \in S_b} \mathbf{P}(s,s')$*
   b. *$\mathbf{P}'_n((s,\tau),s_-) = 0$.*

Let us explain the intuition underlying the previous definition. Consider a "successful" path $(s_0,n),(s_1,n-1)\ldots,(s_k,n-k),s_+$. If all steps of this path correspond to cases 2.a, 2.b or 4.a then the likelihood of this path is:

$$\frac{\mu_n^\bullet(f(s_0))}{\mu_{n-1}^\bullet(f(s_1))} \cdots \frac{\mu_1^\bullet(f(s_k))}{1} = \mu_n^\bullet(f(s_0))$$

If some step of this path corresponds to cases 3.a, 3.b or 3.a then the (current) likelihood is multiplied by some $h_u(s) > 1$ thus the final likelihood of the whole path is greater than $\mu_n^\bullet(f(s_0))$. By definition, an unsuccessful path yields a null likelihood. The next proposition summarises these observations.

Proposition 1 ([4]). *Let $\mathcal{C}$ be a DTMC and $\mathcal{C}^\bullet$ be a reduction. The importance sampling based on matrix of $\mathbf{P}'_n$ of definition 6 has the following property: For all s and all $0 < u \leq n$ such that $\mu(s,u) > 0$, $W_{(s,u)}$ is a random variable whose range is included in $\{0\} \cup [\mu_u^\bullet(f(s)), \infty[$.*

Without additional properties of the reduction, it is difficult to analyse the variance of $W_{(s_0,n)}$. However, based on this proposition we can deduce two criteria for a "good" reduction: (1) the probability $\mu_n^\bullet(f(s_0))$ should be as small as possible and (2) with high probability, during simulation the current state (s,u) should satisfy $h_u(s) \leq 1$. In case the second property is always satisfied (see [4]), $W_{s_0,n}$ is a rescaled Bernoulli distribution with variance $\mu_n(s_0)\mu_n^\bullet(f(s_0)) - \mu_n^2(s_0)$. When the first condition is satisfied, it is much smaller than the original variance $\mu_n(s_0) - \mu_n^2(s_0)$.

From DTMC to CTMC statistical model checking. So, given some DTMC and some time horizon n, the previous method returns for state s, a confidence interval $I_n = [L_n, U_n]$ and a threshold probability ε_n such that $\Pr(\mu_n(s) \notin I_n) \leq \varepsilon_n$. In [4], it is also shown how to obtain $I_\infty = [L_\infty, U_\infty]$ and ε_∞ such that $\Pr(\mu_\infty(s) \notin I_\infty) \leq \varepsilon_\infty$.

Then for a uniformised (by λ) CTMC, using the previous results on its embedded DTMC we obtain a procedure returning a statistical evaluation of $\mu_\tau(s)$. The threshold probability is defined by:

$$\sum_{n=n^-}^{n^+} \varepsilon_n + \varepsilon_\infty \tag{3}$$

and the confidence interval I is defined by:

$$I = \sum_{n=n^-}^{n^+} [c_n(1-\alpha-\beta), c_n] \cdot I_n + [0, \alpha U_{n^-}] + [0, \beta U_\infty] \tag{4}$$

where $[a,b] \cdot [a',b'] = [aa', bb']$, $[a,b] + [a',b'] = [a+a', b+b']$.

The correctness of such a procedure is straightforward. However a naive implementation would require to apply statistical model checking of formulas $aU^{[0,n]}b$ for all n between n^- an n^+ and such a number can be large.

A more tricky alternative consists in producing all trajectories until time horizon n^+ and updating the simulation results at the end of a trajectory for all the intervals $[0,n]$ with $n^- \leq n \leq n^+$ as follows. If the trajectory has reached s^- then it is an unsuccessful trajectory for all intervals. Otherwise if it has reached s^+ at time τ then for all $n \geq \tau$ it is a successful trajectory and for all $n < \tau$ it is unsuccessful. Doing this way, every trajectory contributes to all evaluation, and we significantly increase the sample size without increasing computational cost. The accuracy of the results is improved. However, this requires that the importance sampling associated with time interval $[0, n^+]$ is also appropriate for the other intervals and in particular with time interval $[0, n^-]$.

4.4 Our methodology

Based on the previous developments, we describe a methodology to perform statistical model checking for an SPN $\mathcal{N}$, using importance sampling to estimate the tiny probability $p = \mu_\tau(m_0)$ in several steps, where m_0 is the initial marking.

1. Specify an SPN $(\mathcal{N}^\bullet, m_0^\bullet)$ and a reduction function f from $Reach(\mathcal{N}, m_0)$ to $Reach(\mathcal{N}^\bullet, m_0^\bullet)$.
2. Fix some uniform rate λ for the uniformisation of the embedded DTMC of $\mathcal{N}$. Then, given some numerical precisions α and β, determine the lower and upper bounds n^- and n^+ and coefficients c_n $(n^- \leq n \leq n^+)$ for the Poisson distribution with parameter $\lambda\tau$ (see section 4.3).
3. Numerically compute the distributions $\{\mu_n^\bullet\}_{0<n\leq n^+}$ on the reduced net.
4. Use these distributions to perform importance sampling on the simulation of the initial net to estimate $\mu_u(s)$ for $n^- \leq u \leq n^+$ and $u = \infty$. Weight and combine the confidence intervals with equation (4) in order to return the final interval.

The first step requires some understanding of the system to design the appropriately reduced Markov chain. Steps 2 and 3 are standard computations and do not need additional explanations. Depending on the size of the reachability set of the reduced model, the fourth step requires a different space-time trade-off to put off computational abilities as described below.

4.5 Algorithmic considerations

Table 2. Compared complexities

Complexity	Algorithm 1	Algorithm 2	Algorithm 3	Algorithm 4
Space	mn^+	$2m\sqrt{n^+}$	$m \log n^+$	$2m$
Time for the precomputation	$\Theta(mn^+)$	$\Theta(mn^+)$	$\Theta(mn^+)$	0
Additional time for the simulation	0	$\Theta(mn^+)$	$\Theta(mn^+ \log(n^+))$	$\Theta(m(n^+)^2)$

The easiest way to perform importance sampling is to compute the whole family of vectors $\{\mu_n^\bullet\}_{0<n\leq n^+}$ before the simulation starts as described in Algorithm 1. When the memory requirement of this algorithm exceeds the resources of the machine, Algorithm 2 (resp. Algorithm 3) maintains only a subfamily of size $2\sqrt{n^+}$ (of size $\log(n^+)$). Finally, Algorithm 4 computes the appropriate vector at each step of the simulation. The time and memory consumption of the four algorithms are described in Table 2 where m is the size of the reachability set of the reduced net. The space unit is the storage of a float, and the space complexity takes only into account the memory used to store the vectors. All methods require also to store the transition probability matrix and the simulation state. The algorithms are fully described in the appendix.

Algorithm 1

`Precomputation`$(u, \mu_0^\bullet, P_0^\bullet)$ **Result**: Ls
`// List` Ls `fulfills` $Ls(i) = \mu_i^\bullet$
$v \leftarrow \mu_0^\bullet$
for $i = 1$ **to** u **do**
  $v \leftarrow P_0^\bullet v \quad Ls(i) \leftarrow v$

Algorithm 2

`Precomputation`$(u, \mu_0^\bullet, P_0^\bullet)$ **Result**: Ls, K
`// List` Ls `fulfills` $Ls(i) = \mu_{i \cdot l}^\bullet$
$l \leftarrow \lfloor \sqrt{u} \rfloor \quad w \leftarrow \mu_0^\bullet$
for i **from** *1* **to** $\lfloor \frac{u}{l} \rfloor l$ **do**
  $w \leftarrow P_0^\bullet w$ **if** $i \mod l = 0$ **then**
    $Ls(\frac{i}{l}) \leftarrow w$
`// List` K `contains` $\mu^\bullet_{\lfloor \frac{u}{l} \rfloor l+1}, \ldots, \mu_u^\bullet$
for i **from** $\lfloor \frac{u}{l} \rfloor l + 1$ **to** u **do**
  $w \leftarrow P_0^\bullet w \quad K(i \mod l) \leftarrow w$

`Stepcomputation`$(v, l, P_0^\bullet, K, Ls)$
  `// Updates` K `when needed`
if $v \mod l = 0$ **then**
  $w \leftarrow Ls(\frac{v}{l} - 1)$
  for i **from** $(\frac{v}{l} - 1)l + 1$ **to** $v - 1$ **do**
    $w \leftarrow P_0^\bullet w \quad K(i \mod l) \leftarrow w$

Algorithm 3

`Precomputation`$(u, \mu_0^\bullet, P_0^\bullet)$ **Result**: Ls
`// Ls fulfills` $Ls(i) = \mu^\bullet_{\sum_{j=i}^k a_{u,j} 2^j}$
$k \leftarrow \lfloor \log_2(u) \rfloor + 1 \quad v \leftarrow \mu_0^\bullet \quad Ls(k+1) \leftarrow v$
for i **from** k **downto** 0 **do**
  if $a_{u,i} = 1$ **then**
    for j **from** 1 **to** 2^i **do**
      $w \leftarrow P_0^\bullet w$
  $Ls(i) \leftarrow w$

`Stepcomputation`$(v, l, P_0^\bullet, Ls)$ `// Ls is updated accordingly to` $v - 1$
$i_0 \leftarrow \min(i \mid a_{v,i} = 1) \quad w \leftarrow Ls(i_0 + 1)$
$Ls(i_0) \leftarrow v$
for i **from** $i_0 - 1$ **downto** 0 **do**
  for $j = 1$ **to** 2^i **do**
    $w \leftarrow P_0^\bullet w$
  $Ls(i) \leftarrow w$

Algorithm 4

`Stepcomputation`$(u, \mu_0^\bullet, P_0^\bullet)$ **Result**: v
`// Vector` v `equal to` $\mu_u^\bullet$
$v \leftarrow \mu_0^\bullet$
for $i = 1$ **to** u **do**
  $v' \leftarrow P_0^\bullet v \quad v \leftarrow v'$

5 Experiments

Confidence interval. In all statistical computations that we have conducted here we use the importance sampling defined in Proposition 1. Consequently the result of each trajectory of the simulation is a realisation of a random variable, taking values in $\{0\} \cup [\mu_u^\bullet(f(s)), \infty[$. We have to compute a confidence interval from theses realisations but there is no approach that is more appropriate than other ones for this. We select three ways to compute this confidence interval.

1. The more classical way to compute confidence intervals is to suppose that the distribution is Gaussian; this is asymptotically valid if the variance is finite, thanks to the central limit theorem. But we cannot estimate the accuracy of the approximation.
2. Another method is to use a pseudo Chernoff-Hoeffding bound. Whenever the random variable is bounded, this method is asymptotically valid. In our case we will use the maximal value observed during the simulation as an upper bound.
3. The last method consists in recording the minimal and maximal observed values and returns them as confidence intervals.

As this random variable, say W, takes its values in $\{0\} \cup [\mu^{\bullet}_{n+}(f(s)), \infty[$, we introduce the Bernoulli variable T whose value is 0 when W is 0, and 1 when $W \geq \mu^{\bullet}_{n+}(f(s)) > 0$. The same simulation gives us a confidence interval for $\mathbf{E}(T) = \Pr(T = 1)$, say $[L_T, U_T]$, and a confidence interval for the conditional $\mathbf{E}(W|T = 1)$, say $[L, U]$. As $\mathbf{E}(W) = \mathbf{E}(W|T = 1)\Pr(T = 1)$, we use $[L_T L, U_T U]$ as a confidence interval for $\mathbf{E}(W)$.

We have analysed three properties, the latter two are inspired by [11]. For all properties we use a reduced model with a smaller scaling factor. But the scaling factor is not homogeneously applied over all parts of the model. More details on the construction of the reduced models are provided later.

All the experiments have been carried out with our tool Cosmos [3] on a machine with 16 cores running at 2 GHz and 32 GB of memory.

5.1 Maximal peak of the output signal

We start off with a time-bounded reachability property assessing the strength of the output signal: "What is the probability to reach within 10 time units a state where the total mass of ERK is doubly phosphorylated?", yielding the quantity:

$$p_1 = \Pr(\text{True } \mathbf{U}^{\leq 10}(\text{ERKPP} = 3N))$$

The inner formula is parameterised with N, the scaling factor for the initial marking. The reduced model that we design for Cosmos uses different scaling factors for the three levels in the signalling cascade. The first two levels which are based on Raf and MEK always use a scaling factor of 1, whereas the last level involving ERK uses a scaling factor of N. The second column of Table 3 shows the reduction factor of the reduced model. The mapping function from the original model to the reduced one decreases the number of tokens in each place in a consistent way.

Table 3. Computational complexity related to p_1

N	Cosmos			Marcie	
	reducing factor	time(s)	memory	time(s)	memory
1	-	-	-	4	514MB
2	38	20,072	3,811MB	326	801MB
3	558	15,745	15,408MB	43,440	13,776MB
4	4667	40,241	3,593MB	Out of Memory: >32GB	
5	27353	51,120	19,984MB		

We perform experiments with both Cosmos and Marcie. The time and memory consumption for increasing values of N are reported in Table 3. For each value of N we run one million of trajectories with Cosmos. For $N = 2$ and $N = 3$, we use the first algorithm which is the fastest, but for $N = 4$ and $N = 5$ we use the second algorithm in order to make the computation fit into the memory. We observe that the time consumption significantly increases between

$N = 3$ and $N = 4$ when we switch from Algorithm 1 to Algorithm 2. MARCIE suffers an exponential increasing w.r.t. both time and space resources. When $N = 3$, it is slower than COSMOS and it is unable to handle the case $N = 4$.

Table 4. Numerical values associated with p_1

N	COSMOS			MARCIE
	CI Gaussian	CI Chernoff	CI MinMax	Result
1				2.07E-12
2	[3.75E-027,5.88E-026]	[3.75E-027,4.54E-025]	[3.75E-27,1.57E-23]	8.18E-26
3	[4.34E-042,1.72E-039]	[4.34E-042,1.82E-038]	[4.43E-42,1.87E-37]	2.56E-39
4	[1.54E-057,8.54E-056]	[1.54E-057,1.98E-055]	[1.78E-57,7.05E-55]	Out of Memory
5	[3.97E-073,2.33E-070]	[3.97E-073,7.30E-070]	[5.44E-73,2.24E-69]	

Table 4 depicts the values returned by the two tools: MARCIE returns a single value, whereas COSMOS returns three types of confidence intervals with confidence level set to 0.99. The three confidence intervals correspond to the three methods for bounding the likelihood of trajectories. We observe that confidence intervals computed by the Gaussian analysis always underestimate the result, the ones computed by Chernoff-Hoeffding underestimate it for $N = 3$, and the larger ones computed from the minimal and maximal observation always contain the result computed with MARCIE when it is available. An analysis of the likelihood is detailed in appendix.

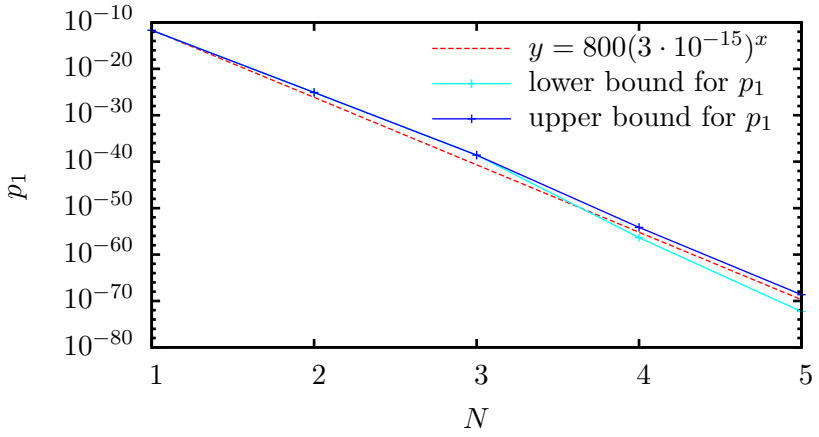

Figure 4. Observing an exponential dependency

Figure 4 illustrates the dependency of p_1 with respect to the scaling factor N. It appears that the probability p_1 depends on N in an exponential way. This striking phenomenon could be interpreted by biologists.

5.2 Conditional maximal signal peak

The network structure of each level in the signalling cascade ensures cyclic behaviour, i.e. phosphorylated proteins, serving as signal for the next level, can also be dephosphorylated again, which corresponds to a decrease of the signal strength. Thus, an intriguing property of the signalling cascade is the probability of a further increase of the signal strength under the condition that a certain strength has already been reached. We explore this property for the first level in the signalling cascade, i.e. *RafP*, and ask specifically for the probability to reach its maximal strength: "What is the probability of the concentration of *RafP* to continue its increase and reach its maximal strength, when starting in a state where the concentration is for the first time at least L?". Please note, this is a special use case of the general pattern introduced in [11].

$$p_2 = \Pr_\pi((\text{RafP} \geq L)\ \mathbf{U}\ (\text{RafP} > L'))$$

where π is the distribution over states when satisfying for the first time state formula $\text{RafP} \geq L$.

This formula is parameterised with the two thresholds L and L'. Observing that $L, L' \in [0, 4N]$ for $RafP$, the results for increasing N and L, with $L < L'$ and $L' = 4N - 1$ are reported in Table 5 (confidence intervals are computed by Chernoff-Hoeffding method). As before MARCIE cannot handle the case $N = 3$ but here the bottleneck is the execution time.

At first sight it was unclear how p_2 evolves when L increases as L' remains fixed. Experiments point out that p_2 increases by a least one magnitude order when L is incremented. This phenomenon could also be interpreted by biologists.

Table 5. Numerical values associated with p_2

N	L	L'	COSMOS		MARCIE		
			confidence interval	time	result	time	memory
2	2	7	[2.39e-13 , 1.07e-09]	31	5.55e-10	90	802 MB
2	3	7	[2.18e-10 , 6.92e-08]	110	6.64e-08	136	816 MB
2	4	7	[9.33e-08 , 3.54e-05]	256	3.01e-06	276	798 MB
2	5	7	[1.16e-05 , 6.08e-04]	1000	7.16e-05	759	801 MB
2	6	7	[5.42e-04 , 1.21e-03]	5612	1.27e-03	3180	804 MB
3	5	11	[1.82e-12 , 9.78e-09]	459	time > 48 hours		
3	6	11	[3.41e-010 , 9.66e-08]	1428			
3	7	11	[1.81e-08 , 2.23e-06]	7067			
3	8	11	[8.72e-07 , 2.71e-06]	4460			
3	9	11	[1.42e-06 , 4.59e-05]	4301			
3	10	11	[2.69e-04 , 9.34e-04]	6420			
4	10	15	[5.12e-09 , 2.75e-08]	8423			
4	11	15	[8.23e-08 , 2.97e-07]	7157			
4	12	15	[9.84e-07 , 1.86e-06]	18730			

5.3 Signal propagation

To demonstrate that the increases of the signals are temporally ordered w.r.t. the levels in the signalling cascade, and by this way proving the travelling of the signals along the levels, we explore the following property: "What is the probability that, given the initial concentrations of *RafP*, $MEKPP$ and $ERKPP$ being zero, the concentration of *RafP* rises above some level L while the concentrations of $MEKPP$ and $ERKPP$ remain at zero, i.e. *RafP* is the first species to react?". While this property has its focus on the beginning of the signalling cascade, it is obvious how to extend the investigation by further properties covering the entire signalling cascade.

$$p_3 = \Pr((\text{MEKPP} = 0) \wedge (\text{ERKPP} = 0))\mathbf{U}(\text{RafP} > L))$$

This formula is parameterised with L for $L \in [0, 4N]$. The results for increasing N and $L = 4N - 1$ are given in Table 6. Obviously, the probability of a further increase for $L = 4N$ is zero. On the one hand as reported in Table 6, the event is less rare than the other properties and on the other hand, the reduced model seems less appropriate for importance sampling. We also observed that as expected the probability decreases with respect to L.

Table 6. Experiments associated with p_3

N	L	Cosmos	
		confidence interval	time
2	5	[9.81e-06 , 0.0518]	102
2	6	[7.8e-05 , 3.16e-04]	102
2	7	[5.78e-07 , 9.89e-06]	103
3	9	[0 , 2.16e-06]	3963

6 Conclusion

We have studied rare events in signalling cascades with the help of an improved importance sampling method implemented in Cosmos. As demonstrated by means of our scalable case study, our method has been able to cope with huge models that could not be handled neither by numerical computations nor by standard simulations. In addition, analysis of the experiments has pointed out interesting dependencies between the scaling parameter and the quantitative behaviour of the model.

In future work we intend to incorporate other types of quantitative properties, such as the mean time a signal needs to exceed a certain threshold, the mean travelling time from the input to the output signal, or the relation between the variation of the enzymes of two consecutive levels. We also plane to analyse other biological systems for which the evaluation of tiny probabilities might be relevant like mutation rates in growing bacterial colonies [9]. Finally, to improve usability, we aim at developing a methodology based on a structural analysis of the net to semi-automatically derive an appropriate reduced model for importance sampling.

References

1. M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with generalized stochastic Petri nets*. John Wiley & Sons, Inc., 1994.
2. L. J. Bain and M. Engelhardt. *Introduction to Probability and Mathematical Statistics, Second Edition*. Duxbury Classic Series, 1991.
3. P. Ballarini, H. Djafri, M. Duflot, S. Haddad, and N. Pekergin. HASL: An expressive language for statistical verification of stochastic models. In *Proceedings of the 5th International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS'11)*, pages 306–315, Cachan, France, May 2011.
4. B. Barbot, S. Haddad, and C. Picaronny. Coupling and importance sampling for statistical model checking. In C. Flanagan and B. König, editors, *TACAS*, volume 7214 of *Lecture Notes in Computer Science*, pages 331–346. Springer, 2012.
5. B. Barbot, S. Haddad, and C. Picaronny. Importance sampling for model checking of continuous time Markov chains. In Petre Dini and Pascal Lorenz, editors, *Proceedings of the 4th International Conference on Advances in System Simulation (SIMUL'12)*, pages 30–35, Lisbon, Portugal, November 2012. XPS.
6. R. Breitling, D. Gilbert, M. Heiner, and R. Orton. A structured approach for the engineering of biochemical network models, illustrated for signalling pathways. *Briefings in Bioinformatics*, 9(5):404–421, September 2008.
7. V. Chickarmane, B. N. Kholodenko, and H. M. Sauro. Oscillatory dynamics arising from competitive inhibition and multisite phosphorylation. *Journal of Theoretical Biology*, 244(1):68–76, January 2007.
8. B. L. Fox and P. W. Glynn. Computing Poisson probabilities. *Commun. ACM*, 31(4):440–445, 1988.
9. D. Gilbert, M. Heiner, F. Liu, and N. Saunders. Colouring Space - A Coloured Framework for Spatial Modelling in Systems Biology. In JM Colom and J Desel, editors, *Proc. PETRI NETS 2013*, volume 7927 of *LNCS*, pages 230–249. Springer, June 2013.
10. P. W. Glynn and D. L Iglehart. Importance sampling for stochastic simulations. *Management Science*, 35(11):1367–1392, 1989.
11. M. Heiner, D. Gilbert, and R. Donaldson. Petri nets for systems and synthetic biology. In M. Bernardo, P. Degano, and G. Zavattaro, editors, *SFM 2008*, volume 5016 of *LNCS*, pages 215–264. Springer, 2008.
12. M. Heiner, C. Rohr, and M. Schwarick. MARCIE - Model checking And Reachability analysis done effiCIEntly . In J.M. Colom and J. Desel, editors, *Proc. PETRI NETS 2013*, volume 7927 of *LNCS*, pages 389–399. Springer, 2013.
13. W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):pp. 13–30, 1963.
14. P. L'Ecuyer, V. Demers, and B. Tuffin. Rare events, splitting, and quasi-Monte Carlo. *ACM Trans. Model. Comput. Simul.*, 17(2), 2007.
15. A. Levchenko, J. Bruck, and P.W. Sternberg. 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. G. Rubino and B. Tuffin. *Rare Event Simulation using Monte Carlo Methods*. Wiley, 2009.

7 Appendix

7.1 Algorithmic considerations

Here, we describe in detail the four algorithms that we can use in order to the space-time trade-off in the algorithmic part.

We denote by m the number of states of the embedded Markov chain $\mathcal{C}^\bullet$ of the reduced net. A simulation takes at most u steps going through states $(s_{n^+}, n^+), \ldots, (s_1, 1), s_\pm$ where $s_{n^+} = s_0$ and $s_\pm \in \{s_+, s_-\}$. In state (s_v, v), we compute the distribution $P'_u((s_v, v), -)$ (cf. definition 6), which requires the values of $\mu^\bullet_v(f(s))$ and $\mu^\bullet_{v-1}(f(s'))$, for each possible target state s' from s_v.

Thanks to equations 1, the vectors $\{\mu^\bullet_v\}_{0<v\leq u}$ may be computed iteratively one from the other with complexity $\Theta(mdu)$. More precisely, we derive from $\mathbf{P}^\bullet$, matrix $\mathbf{P}^\bullet_0$, a square (substochastic) matrix, indexed by $S_{a\bar{b}} \cup s_+$ and defined by $\forall s, s' \in S_{a\bar{b}}$:
$\mathbf{P}^\bullet_0(s, s') = \mathbf{P}^\bullet(s, s'), \mathbf{P}^\bullet_0(s, s_+) = \sum_{s'' \in S_b} \mathbf{P}^\bullet(s, s'')$
$\mathbf{P}^\bullet_0(s_+, s_+) = 1, \mathbf{P}^\bullet_0(s_+, s') = 0$
Then $\mu^\bullet_v = \mathbf{P}^\bullet_0 \cdot \mu^\bullet_{v-1}$ and $\mu^\bullet_0$ is null except $\mu^\bullet_0(s_+) = 1$. But for large values of u, the space complexity to store them becomes intractable and the challenge is to obtain a space-time trade-off. So we propose four methods. The methods consist of a precomputation stage and a simulation stage. Their difference lies in the information stored during the first stage and the additional numerical computations during the second stage. In the precomputation, most methods compute iteratively the u vectors $\mu^\bullet_v = (P^\bullet_0)^v(\mu^\bullet_0)$ for v from 1 to u.

1. The first method is the "natural" implementation. It consists in storing all these vectors during the precomputation stage and then proceeding to the simulation without any additional numerical computations. The precomputation stage is described in algorithm 4.5 presented in the appendix. The storage of vectors $\{\mu^\bullet_v\}_{v\leq u}$ is the main memory requirement.
2. Let $l(< u)$ be an integer. In the precomputation stage, the second method only stores the $\lfloor \frac{u}{l} \rfloor + 1$ vectors $\mu^\bullet_\tau$ with τ multiple of l in list Ls and $\mu^\bullet_{l\lfloor \frac{u}{l} \rfloor+1}, \ldots, \mu^\bullet_u$ in list K (see the precomputation stage of algorithm 4.5). During the simulation stage, in a state (s, τ), with $\tau = ml$, the vector $\mu^\bullet_{\tau-1}$ is present neither in Ls nor in K. So the method uses the vector $\mu^\bullet_{l(m-1)}$ stored in Ls to compute iteratively all vectors $\mu^\bullet_{l(m-1)+i} = P^{\bullet i}(\mu^\bullet_{l(m-1)})$ for i from 1 to $l-1$ and store them in K (see the step computation stage of algorithm 4.5). Then it proceeds to l consecutive steps of simulation without anymore computations. We choose l close to $\sqrt{u}$ in order to minimize the space complexity of such a factorization of steps.
3. Let $k = \lfloor \log_2(u) \rfloor + 1$. In the precomputation stage, the third method only stores $k+1$ vectors in Ls. More precisely, initially using the binary decomposition of u ($u = \sum_{i=0}^k a_{u,i} 2^i$), the list Ls of $k+1$ vectors consists of $w_{i,v} = \mu^\bullet_{\sum_{j=i}^k a_{v,j} 2^j}$, for all $1 \leq i \leq k+1$ (see the precomputation step of algorithm 4.5). During the simulation stage in a state (s, v), with the binary decomposition of v ($v = \sum_{i=0}^k a_{v,i} 2^i$), the list Ls consists of $w_{i,v} = \mu^\bullet_{\sum_{j=i}^k a_{v,j} 2^j}$,

for all $1 \leq i \leq k+1$. Observe that the first vector $w_{1,v}$ is equal to $\mu_v^\bullet$. We obtain $\mu_{v-1}^\bullet$ by updating Ls according to $v-1$. Let us describe the updating of the list performed by the stepcomputation of algorithm 4.5. Let i_0 be the smallest index such that $a_{v,i_0} = 1$. Then for $i > i_0$, $a_{v-1,i} = a_{v,i}$, $a_{v-1,i_0} = 0$ and for $i < i_0$, $a_{v-1,i} = 1$. The new list Ls is then obtained as follows. For $i > i_0$ $w_{i,v-1} = w_{i,v}$, $w_{i_0,v-1} = w_{i_0-1,v}$. Then the vectors for $i_0 < i$, the vectors $w_{i,v-1}$ are stored along iterated $2^{i_0-1} - 1$ matrix-vector products starting from vector $w_{i_0,v-1}$: $w(j, v-1) = P_0^{\bullet 2^j} w(j+1, v-1)$.
The computation associated with v requires $1 + 2 + \cdots + 2^{i_0-1}$ products matrix-vector , i.e. $\Theta(md2^{i_0})$. Noting that the bit i is reset at most $m2^{-i}$ times, the complexity of the whole computation is $\sum_{i=1}^{k} 2^{k-i}\Theta(md2^i) = \Theta(mdn^+ \log(n^+))$.

4. The fourth method naively consists in computing vector $\mu_v^\bullet$ from the initial vector at each step. In this method we only need to store two copies of the vector. The required amount of space is the same as the one required by a numerical model checker on the same system. This method in not really practical but it give a lower bound on the required memory.

7.2 Experimental analysis of the likelihood

We describe here some technical reading of the simulation results obtained for property p_1 (from subsection 5.1). To analyse these results, we have first to study the distribution of the random variable $W_{(s,n^+)}$ which is the likelihood of the trajectories. Proposition 1 guarantees that values of $W_{(s,n^+)}$ lay in $\{0\} \cup [\mu_{n^+}^\bullet(f(s)), \infty[$. We observe for each trajectory the value of its likelihood at the end of the simulation. Figure 5 shows an histogram of the distribution of the strictly positive values of $W_{(s,n^+)}$ as well as their impact on the mean values of $W_{(s,n^+)}$. We simulate the system with $N = 2$ and for the formula p_1 with a discrete horizon of 615 which is the right truncation point return by the Fox Glynn algorithm for this property. This figure is in logarithmic scale for abscissa and distribution of impact. The distribution of trajectories is in linear scale and has been scaled to fit in the same figure. The total number of trajectories is 69000, 49001 of them end with the 0 value. We can see from this figure that most of the trajectories end with a value close to 2.10^{-35}, and that a few trajectories have a value close to 10^{-32}, but the contributions to the mean of the two set of trajectories are similar. This means that an estimator of the mean value of $W_{(s,u)}$ will underestimate the true expected value of $W_{(s,u)}$.

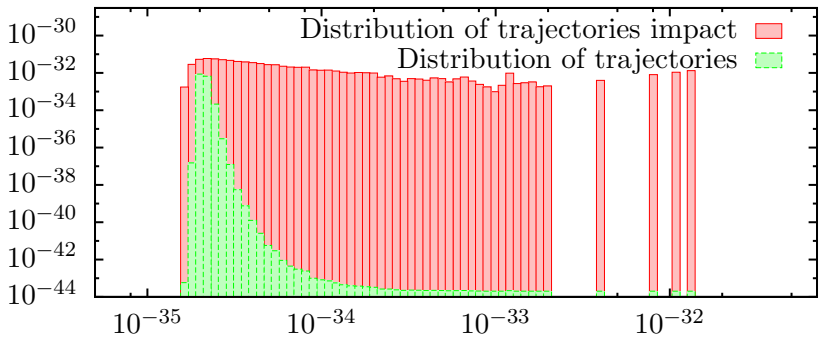

Figure 5. Distribution of trajectories and their contribution