Importance Sampling for Model Checking of Continuous Time Markov Chains

Benoît Barbot, Serge Haddad, Claudine Picaronny

May 2012

Research report LSV-12-08
Importance Sampling for Model Checking of Continuous Time Markov Chains

Benoît Barbot, Serge Haddad, Claudine Picaronny
LSV, ENS Cachan & CNRS & INRIA Cachan, France
\{barbot,haddad,picaronny\}@lsv.ens-cachan.fr

Abstract—Statistical model-checking is an alternative verification technique applied on stochastic systems whose size is beyond numerical analysis ability. Given a model (most often a Markov chain) and a formula, it provides a confidence interval for the probability that the model satisfies the formula. In previous contributions, we have overtaken the main limitation of the statistical approach, i.e. the computation time explosion associated with the evaluation of very small probabilities. These methods were only valid for discrete time Markov chains and untimed properties. Here combining the uniformization technique and generalizing our previous techniques, we design a method which applies to continuous time Markov chains and formulas of a timed temporal logic. The corresponding algorithm has been implemented in our tool COSMOS. We present experimentations on a relevant system, with drastic time reductions w.r.t. standard statistical model checking.

Keywords—statistical model checking, rare events, importance sampling, coupling, uniformization

I. INTRODUCTION

Model checking of probabilistic real time systems. Many complex systems exhibit probabilistic behaviour either in an inherent way or through interaction with unreliable environment (communication protocols, biological systems…). When embedded in safety-critical applications, it is crucial to fiably ensure the correct behaviour of these systems. Model checking is an efficient technique to verify properties of a system behaviour [1]. Thanks to its algorithmic simplicity, it has been successfully implemented in a variety of tools. The system is described as a formal model and required properties are expressed by some temporal logic. It can handle models governed by continuous time and probabilistic choice and properties specified by temporal logics like CSL [2]. For instance, such logics can express probabilistic real-time properties like “the probability that the airbag fails to deploy within 10ms is less than $10^{-3}$”.

Statistical model checking. Model-checking of stochastic systems may be undertaken using numerical or statistical techniques. Whenever it is possible, the numerical method provides a high accuracy as the result is exact up to numerical approximations. Its applicability is nevertheless restricted, as it suffers from state explosion. Statistical sampling techniques only provide a probabilistic guarantee of correctness. But they handle a much broader class of models. As soon as it is possible to perform a Monte-Carlo simulation the model, we obtain a probabilistic framing of the expected value by generating a big sample of trajectories of the system and computing the ratio of these trajectories that satisfy a given property. This usually requires a very small amount of space compared to the numerical method and allows to deal with huge models [3].

Rare events. The main drawback of the statistical model checking is its inefficiency in dealing with very small probabilities. The size of the sample of simulations required to estimate these small probabilities exceeds achievable capacities. This difficulty is known as the rare event problem. Several methods have been developed to cope with this problem whose main one is importance sampling. Importance sampling method is based on a modification of the underlying probability distribution in such a way that the rare events occur much more frequently. Most of the techniques related to importance sampling are based on heuristics and cannot provide any confidence interval for the estimated probability. In [4], we proposed an efficient method based on importance sampling to estimate in a reliable way (the first one with a true, and not an approximate, confidence interval) the very small probability of a standard “Until” property $(aUb)$ using coupling theory and we applied it on a large variety of case studies, modelled by discrete time Markov chains (DTMC).

Our contribution. While DTMC is an appropriate model for probabilistic reasoning, continuous time Markov chains (CTMC) are required for simultaneous timed and probabilistic assessments. So here, we deal with CTMC and timed temporal formula. More precisely, we consider formulas $\varphi^I$ where $\varphi$ is specified by a finite automaton and $I = [0, \tau]$ is a bounded interval. We statistically estimate the (tiny) probability that a random path generated by the CTMC is accepted by the automaton before instant $\tau$. In order to design and prove the correctness of the method we proceed in three stages:

- We show using uniformisation [5] that the confidence interval for the estimation can be computed from confidence intervals of several estimations in the embedded DTMC of the CTMC.
- Our importance sampling approach for time bounded reachability in DTMC is then developed by generalizing the one we have proposed in [4], based on the mapping of the original model to a reduced one and on coupling [6].
- However contrary to the original approach, the memory requirements are no more negligible and depend on the
considered (discrete) time interval. Thus we propose three algorithms with a different trade-off between time and space so that we can handle very large time intervals.

To the best of our knowledge, our method is the first one among importance sampling methods for CTMC that provides a confidence interval. Furthermore we have implemented it in the statistical model checker COSMOS [7]. We tested our tool on two classical relevant models getting impressive time and/or memory reductions.

Organisation. In section II, we give a state of the art related to rare event handling emphasizing the continuous time framework. The next three sections describe our method. In section III the general scheme is presented. The theoretical framework for estimating the time bounded reachability in DTMC is detailed in section IV. The different algorithmic solutions are developed and discussed in section V. Afterwards we present two experimentations in section VI. Finally in section VII, we conclude and give some perspectives to this work.

II. STATE OF THE ART

Statistical model checking. Whenever numerical model checking fails, models with huge stochastic process are handled by statistical model checking. The corresponding techniques randomly generate a (large) set of execution paths and check whether the paths fulfill the formula. The result is a probabilistic estimation of the satisfaction given by a confidence interval [8]. In principle, it only requires to maintain a current state (and some numerical values in case of a non Markovian process). Furthermore no regenerative assumption is required and it is easier to parallelise the methods. Several tools include statistical model checking: COSMOS [7], GREATSPN [9], PRISM [10], UPPAAL [11], YMER [12].

Rare events. Model checking of probabilistic systems is particularly important for events which have disastrous consequences (loss of human life, financial ruin, etc.), but occur with tiny probability. Unfortunately statistical model checking of rare events triggers a computation time explosion, forbidding its use. Thus acceleration techniques [13] have been introduced to cope with this problem. The two main families of methods are splitting and importance sampling.

Importance sampling. Importance sampling methods [14] generate paths of a system whose probability distribution of transitions has been changed to increase the probability of the event to occur. A weight is then affected to each path to correct the introduced bias. In the discrete case (Markov chains), there is an optimal change of distribution, defining an estimator with null variance. In practice, this distribution takes the required probability as a value, so cannot be computed. But this theoretical result provides a general guideline to choose a new distribution, allowing to design efficient heuristics for some classes of models. In our case, we are to substitute to the Bernoulli random variable corresponding to the occurrence of the rare event, another one with same mean value (the probability of event occurrence) but smaller variance.

The modification of the distribution can be performed at the model level (called static) or at the Markov chain level (called dynamic). The static importance sampling requires no additional memory but in general provides a smaller reduction of variance than the dynamic importance sampling. It may not be sufficient even for asymptotical optimality [15]. In a continuous setting, the optimal distribution cannot be obtained this way (see [13],paragraph 2.2 for a counter-example). In full generality, the dynamic importance sampling [16] requires to maintain a memory whose size is proportional to the size of the Markov chain which is exactly what one wants to avoid. To the best of our knowledge, theoretical results have been obtained for importance sampling but none of them includes any reliable confidence interval for the mean value.

Rare events in a continuous context. Monte-Carlo simulation of complex dynamical systems governed by stochastic processes require to estimate some rare events probabilities in a continuous setting. It has been intensely studied for very specific models, mainly using the splitting method, in order to develop specific algorithms to increase the efficiency of simulations of rare events in particle transport in nuclear physics [17], delay time distributions and losses in some telecommunication networks [18] or in air traffic modeling [19]. Importance sampling method [20] has been justified for the simulation of coupled molecular reactions in biological or genetic networks, but with no clue on how to choose the importance sampling.

III. OVERALL APPROACH

Our approach requires several technical developments that we organize in three sections.

- The core of our approach is a theoretical framework (based on coupling) for statistical model checking of discrete time Markov chains (DTMC). This framework yields, under some assumptions, to an importance sampling which provides both a confidence interval and a reduction variance for the estimation of the satisfaction probability of a timed temporal formula in DTMCs. This part is described in section IV.
- Since the memory requirements while much lesser than for numerical evaluation are still important we propose three algorithms with a different trade-off between time and space in section V.
- The overall approach applied to continuous time Markov chains (CTMC) consists in uniformizing the chain and combining a numerical algorithm with the previous statistical algorithms in order to design an
importance sampling with the same properties for estimation of the satisfaction probability of formula in CTMCs. This part is described in the next paragraphs.

A. Markov chains

**Definition 1:** A discrete time Markov chain (DTMC) $C$ is defined as a set of states $S$, an initial state $s_0$, and a transition probability matrix $P$ of size $S \times S$. The state of the chain at (discrete) time $n$ is a random variable $X_n$ defined inductively by $Pr(X_0 = s_0) = 1$ and $Pr(X_{n+1} = s' \mid X_n = s, X_{n-1} = s_{n-1}, \ldots, X_0 = s_0) = Pr(X_{n+1} = s' \mid X_n = s) = P(s, s')$.

**Definition 2:** A continuous time Markov Chain (CTMC) $C$ is defined by:

- a set of states $S$ with an initial state $s_0$;
- a rate $\lambda_x$ associated with every state $s \in S$;
- a transition matrix $P$ of size $S \times S$.

The behaviour of $C$ is defined by two families of random variables: $\{X_n\}_{n\in\mathbb{N}}$ and $\{T_n\}_{n\in\mathbb{N}}$. $X_n$ denotes the state of the chain after $n$ steps while $T_n$ denotes the time elapsed in state $X_n$ before the next step. The time interval between the time instants $T_n$

- $X_n$ defined inductively by $Pr(X_0 = s_0) = 1$ and $Pr(X_{n+1} = s' \mid X_n = s, X_{n-1} = s_{n-1}, \ldots, X_0 = s_0) = Pr(X_{n+1} = s' \mid X_n = s) = P(s, s')$.
- $T_n$ is a random variable distributed as a negative exponential, whose rate is $\lambda_{X_n}$. That is to say:

\[
Pr(T_n \leq \tau \mid X_0 = s_0, \ldots, X_n = s, T_0 \leq \tau_0, \ldots, T_{n-1} \leq \tau_{n-1}) = \lambda_{X_n} \tau
\]

The DTMC defined by $P$ 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 $\mu$ 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 $Y_{\tau}$ is defined almost everywhere and denotes the state of $C$ at (continuous) time $\tau$.

\[
Y_{\tau} = X_{n(\tau)} \text{ where } n(\tau) = \min\left\{n \mid \sum_{k=0}^{n} t_k > \tau \right\}
\]

**Example.** Figure 1 represents a CTMC modelling a tandem queue system. This system contains two queues, the number of clients in the first queue is represented on the horizontal axis and the number of clients in the second one is represented on the vertical axis. In the initial state $s_0$, the two queues are empty. Given some state, a new client comes in the first queue with rate $\rho_0$, a client leaves the first queue for the second one with rate $\rho_1$ and a client leaves the second queue and exits with rate $\rho_2$.

Usually the modeller does not specify its system with a Markov chain. He rather defines a higher level model $M$ (a queueing network, a stochastic Petri net, etc.), whose operational semantic is a Markov chain $C$.

B. Model checking of Markov chains

In the context of model checking, the states of chain $C$ are labelled with atomic propositions that they fulfill. Given state $s$, $\alpha(s)$ denotes the set of propositions satisfied by $s$. We denote $S_\alpha = \{s \in S \mid x \in \alpha(s)\}$, $S_\neg \alpha = \{s \in S \mid x \notin \alpha(s)\}$, $S_{\neg x} = \{s \in S \mid x \in \alpha(s) \land y \in \neg \alpha(s)\}$, 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 $\varphi$ where $\varphi$ is specified by 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 for along the path at every instant less than $\tau$ and $b$ is satisfied at instant $\tau$.

In a DTMC, we denote $\mu_n(s)$, the probability that a random path starting from $s$ fulfills $aU[0,n]b$. These probabilities can be shown to be the solution of the following system of equations ($I_E$ denotes the indicator function of set $E$).

\[
\begin{cases}
\forall u \in S_{\neg \alpha} \quad \mu_u(s) = 0 \\
\forall u \in S_\alpha \quad \mu_u(s) = 1 \\
\forall u > 0 \in S_{\neg \alpha} \quad \mu_u(s) = \sum_{s' \in S} P(s, s') \mu_{u-1}(s')
\end{cases}
\]

(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 $s$ fulfills $aUb$.

Similarly in a CTMC, we denote $\mu_\tau(s)$ the probability that a random path starting from $s$ fulfills $aU[0,\tau]b$. Observe that $\mu_\infty(s)$ only depends on the embedded DTMC (and in fact is equal to the corresponding value in the DTMC).

These probabilities can be shown to be the solutions of the following differential equation system:

\[
\begin{cases}
\forall u \in S_{\neg \alpha} \quad \mu_u(s) = 0 \\
\forall u \in S_\alpha \quad \mu_u(s) = 1 \\
\forall u > 0 \in S_{\neg \alpha} \quad \frac{d\mu_u(s)}{dt} = \lambda_s \left( (P(s, s) - 1) \mu_s(s) + \sum_{s' \in S} P(s, s') \mu_{s'}(s') \right)
\end{cases}
\]

Figure 1. CTMC for the tandem queues
D. Statistical model checking of a uniform CTMC

Assume we have a statistical procedure that, given some DTMC and some time horizon \( n \), returns for state \( s \), a (random) confidence interval \( I_n \) and threshold probability \( \varepsilon_n \) such that:

\[
\Pr(\mu_n(s) \notin I_n) \leq \varepsilon_n
\]

This procedure also computes a bound \( u_{p_k} \) (with \( k \in \mathbb{N} \cup \{\infty\} \)) and a threshold probability \( \varepsilon'_k \) such that

\[
\Pr(\sup_{n<k} \mu_n(s) > u_{p_k}) \leq \varepsilon'_k
\]

Assume furthermore that we have a numerical procedure\(^1\) that, given some rate \( \lambda \) and numerical precisions \( \alpha, \beta \), computes indices \( n^- \) and \( n^+ \) and coefficients \( c_n \) such that:

\[
c_n(1 - \alpha - \beta) \leq \frac{e^{-\lambda \tau} (\tau)^n}{n!} \leq c_n \quad \text{for} \quad n^- \leq n \leq n^+
\]

Then:

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

Then we combine the two procedures to obtain a statistical evaluation of \( \mu_\tau(s) \), as follows. The threshold probability is defined by:

\[
\sum_{n=n^-}^{n^+} \varepsilon_n + \varepsilon'_{n^-} + \varepsilon'_\infty \tag{2}
\]

and the confidence interval \( I \) is defined by:

\[
I = \sum_{n=n^-}^{n^+} [c_n(1 - \alpha - \beta), c_n] \cdot I_n + [0, \alpha u_{p_{n^-}}] + [0, \beta u_{p_\infty}]
\]

where \([a, b] \cdot [a', b'] = [aa', bb']\) and \([a, b] + [a', b'] = [a + a', b + b']\). The correctness of such a procedure is straightforward.

IV. IMPORTANCE SAMPLING FOR STATISTICAL MODEL CHECKING OF DTMC

A. Importance sampling for reachability analysis of DTMC

We consider a DTMC \( C \) with two absorbing states \( s_+ \) or \( s_- \), i.e., \( P(s_-, s_-) = P(s_+, s_+) = 1 \). We require that the probability to reach \( s_+ \) or \( s_- \) from any state is equal to 1.

The statistical approach for reachability evaluation consists in generating \( K \) paths of \( C \) which end in an absorbing state.

Let \( K_+ \) be the number of paths ending in the \( s_+ \) state. 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 a variance \( \frac{p(1-p)}{K} \). When \( K \) goes to infinity the variance goes to 0. In order to be more precise on the estimation, we formally define what is a confidence interval.

\(^1\)As described in [21].
Definition 3: Let \( X_1, \ldots, X_n \) be independent random variables following a common distribution including a parameter \( \theta \). Let \( 0 < \gamma < 1 \) be a confidence level. Then a confidence interval for \( \theta \) with level at least \( \gamma \) is given by two random variables \( m(X_1, \ldots, X_n) \) and \( M(X_1, \ldots, X_n) \) such that for all \( \theta \):

\[
\Pr\left( m(X_1, \ldots, X_n) \leq \theta \leq M(X_1, \ldots, X_n) \right) \geq \gamma
\]

For standard parametrized distributions like the normal or the Bernoulli ones, it is possible to compute confidence intervals [8]. Thus, given a number of paths \( K \) and a confidence level \( 1 - \varepsilon \), the method produces a confidence interval. As discussed before when \( p \ll 1 \), the number of paths required for a small confidence interval is too large to be simulated.

The importance sampling method uses a modified transition matrix \( \mathbf{P}' \) during the generation of paths. \( \mathbf{P}' \) must satisfy:

\[
\mathbf{P}(s, s') > 0 \Rightarrow \mathbf{P}'(s, s') > 0 \lor s' = s_-
\]

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 \( L \) initialized 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 \( \mathbb{E}(V_s) = p \). The following proposition establishes the correctness of the method.

Proposition 1: \( \mathbb{E}(W_{s_+}) = p \).

Proof: In all states, the probability to reach \( s_- \) or \( s_+ \) is equal to 1. Then thanks to a classic result on Markov chains the expected value of the r.v. \( V_s \) is the unique solution of the following system of equations:

\[
\mathbb{E}(V_{s_-}) = 0 \land \mathbb{E}(V_{s_-}) = 1 \land \forall s \notin \{s_- , s_+ \} \mathbb{E}(V_s) = \sum_{s' \neq s_-} \mathbf{P}(s, s') \mathbb{E}(V_{s'})
\]

We now write the corresponding system for \( \mathbf{P}' \) with correction factor:

\[
\mathbb{E}(W_{s_-}) = 0 \land \mathbb{E}(W_{s_-}) = 1 \land \forall s \notin \{s_- , s_+ \} \mathbb{E}(W_s) = \sum_{s' \neq s_- \land \mathbf{P}'(s, s') > 0} \mathbf{P}'(s, s') \left( \frac{\mathbf{P}(s, s')}{\mathbf{P}'(s, s')} \right) \mathbb{E}(W_{s'})
\]

Thanks to the restriction of equation 4, the two systems are equal after simplification, and we have \( \mathbb{E}(W_{s_0}) = \mathbb{E}(V_{s_0}) = p \).

A good choice of \( \mathbf{P}' \) should reduce the variance of \( W_{s_0} \) w.r.t. to variance of \( V_{s_0} \). The following proposition shows that there exists a matrix \( \mathbf{P}' \) which leads to a null variance. We denote the probability to reach \( s_+ \) starting from \( s \) by \( \mu(s) \).

Proposition 2: Let \( \mathbf{P}' \) be defined by:

- \( \forall s \) such that \( \mu(s) \neq 0 \), \( \mathbf{P}'(s, s') = \frac{\mu(s')}{\mu(s)} \mathbf{P}(s, s') \)
- \( \forall s \) such that \( \mu(s) = 0 \), \( \mathbf{P}'(s, s') = \mathbf{P}(s, s') \)

Then for all \( s \), the variance of \( W_s \) is null: \( \mathbb{V}(W_s) = 0 \).

Proof: If \( \mu(s) = 0 \) then all trajectories starting in \( s \) end in \( s_- \). Therefore the variance is null.

If \( \mu(s) \neq 0 \), thanks to the equation

\[
\mu(s) = \sum_{s' | \mu(s') > 0} \mathbf{P}(s, s') \mu(s')
\]

\( \mathbf{P}'(s, -) \) is a distribution. A trajectory starting from a state \( s \) with \( \mu(s) > 0 \) visits only states \( s' \) with \( \mu(s') > 0 \), so it ends in \( s_+ \). Denoting by \( s = u_0, \ldots, u_t = s_+ \) such a trajectory, the value \( L \) is equal to \( \mu(u_t)/\mu(u_1) \cdots \mu(u_{t-1})/\mu(u_t) = \mu(s) \).

This result has a priori no practical application since it requires the knowledge of \( \mu \) for all states, whereas we only want to estimate \( \mu(s_0) \! \! . \)

B. From model checking to reachability

We now relate the computation of \( \mu(s) \) in DTMC \( \mathcal{C} \) to a reachability problem in a Markov chain \( \mathcal{C}_u \) which depends both on \( \mathcal{C} \) and on \( I \).

Markov chain \( \mathcal{C}_u \) (with \( u \) finite) is defined by:

- \( S_u = S_{\mathcal{C}} \times \{1, u\} \cup \{s_-, s_+\} \)
- \( s_-, s_+ \) are absorbing states:
  \( \mathbf{P}_u(s_-, s_-) = \mathbf{P}_u(s_+, s_+) = 1 \)
  \( \forall s, s', v > 1 \mathbf{P}_u((s, v), (s', v - 1)) = \mathbf{P}(s, s') \)
  \( \mathbf{P}_u((s, v), s_-) = \sum_{s' \in S_{\mathcal{C}}} \mathbf{P}(s, s') \mathbf{P}_u(s', s_-) \)
  \( \mathbf{P}_u((s, v), s_+) = \sum_{s' \in S_{\mathcal{C}}} \mathbf{P}(s, s') \mathbf{P}_u(s', s_+) \)
- \( \forall s \mathbf{P}_u((s, 1), s_+)) = \sum_{s' \in S_{\mathcal{C}}} \mathbf{P}(s, s') \mathbf{P}_u(s', s_+) \)
- \( \mathbf{P}_u((s, 1), s_-)) = 1 - \mathbf{P}_u((s, 1), s_+) \)
- 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, \( \mu(s, v) = \mu_v(s) \) where \( \mu(s, v) \) is the probability to reach \( s_+ \) in \( \mathcal{C}_u \) starting from \( (s, v) \) and \( \mu_v(s) \) is the probability of satisfying formula \( aU[0,v]b \) in \( \mathcal{C} \).

The definition of Markov chain \( \mathcal{C}_\infty \) is simpler:

- \( S_\infty = S_{\mathcal{C}} \cup \{s_-, s_+\} \)
- \( s_-, s_+ \) are absorbing states:
  \( \mathbf{P}_\infty(s_-, s_-) = \mathbf{P}_\infty(s_+, s_+) = 1 \)
  \( \forall s, s' \in S_{\mathcal{C}} \mathbf{P}_\infty(s, s') = \mathbf{P}(s, s') \)
  \( \mathbf{P}_\infty(s_-, s_-) = \sum_{s' \in S_{\mathcal{C}}} \mathbf{P}(s, s') \mathbf{P}_\infty(s', s_-) \)
  \( \mathbf{P}_\infty(s_+, s_+) = \sum_{s' \in S_{\mathcal{C}}} \mathbf{P}(s, s') \mathbf{P}_\infty(s', s_+) \)

In \( \mathcal{C}_\infty \), the probability to reach \( s_+ \) or \( s_- \) from any state is not necessary equal to 1. So in order to apply our technique,
we require that in $\mathcal{C}$ the probability that an infinite random path always lies in $S_{\mathcal{G}}$ is 0. Then by construction, $\mu(s) = \mu_\infty(s)$ where $\mu(s)$ is the probability to reach $s_+$ in $\mathcal{C}_\infty$ and $\mu_\infty(s)$ is the probability of satisfying formula $aUb$ in $\mathcal{C}$.

In case of a formula specified by a deterministic finite automaton, the synchronized product is appropriate for $u = \infty$ since one has already transformed the model checking in a reachability problem. When $u$ is finite, one unfolds the synchronized product similarly as it is done in the definition of $\mathcal{C}_u$.

Example. Figure 3 describes chain $\mathcal{C}_u$ associated with the example.

C. Coupling for model checking

The coupling method [6] is a classical method for comparing two stochastic processes, applied in different contexts (establishing ergodicity of a chain, stochastic ordering, bounds, etc.). In the sequel we develop a coupling framework for model checking. A coupling between two DTMCs is a chain whose space is a subset of the product of the two spaces which satisfies: (1) the projection of the product chain on any of its components behaves like the original corresponding chain, (2) an additional constraint which depends on the property to be proved (here related to the until formula). For our needs, we only need to define the coupling of a chain with itself.

Definition 4: Let $\mathcal{C} = (S, P)$ be a labelled DTMC and $(a, b)$ a pair of atomic propositions. A coupling of $\mathcal{C}$ w.r.t. $(a, b)$ is a DTMC $\mathcal{C}^\otimes = (S^\otimes, P^\otimes)$ such that:

- $S^\otimes \subseteq S \times S$
- $\forall s \neq s' \in S \Rightarrow (s, s') \in S^\otimes$,
- $P((s, s'), (s_1, s_1')) = \sum_{(s_1, s_1') \in S^\otimes} P^\otimes((s, s'), (s_1, s_1'))$ and $\forall s' \neq s' \in S \forall (s, s') \in S^\otimes$,
- $P((s', s_1')) = \sum_{(s_1', s_1') \in S^\otimes} P^\otimes((s', s_1'), (s_1', s_1'))$,
- $\forall (s, s') \in S^\otimes \Rightarrow \exists s \in S_b \Rightarrow s = s_b$,
- $\forall (s, s') \in S^\otimes \Rightarrow \exists s \in S_{ab} \Rightarrow s' = s_{ab}$

The set $S^\otimes$ defines a coupling relation of the chain.

In case of a formula specified by a deterministic finite automaton, one considers the synchronized product and the last conditions become: (1) if $s'$ is the accepting state then $s = s'$ and (2) if $s$ is the rejecting state then $s = s'$.

The following proposition allows to compare probabilities without any numerical computation. As before, $\mu_a(s)$ denotes the probability that a random path starting from $s$ satisfies $aU[0,u]b$.

Proposition 3: Let $\mathcal{C}^\otimes$ be a coupling of $\mathcal{C}$ related to $(a, b)$ then for all $(s, s') \in S^\otimes$, we have:

$\forall u \mu_a(s) \geq \mu_a(s')$ (and thus $\mu_\infty(s) \geq \mu_\infty(s')$)

Proof: We first observe that the property on the coupling only depends on $a, b$. So we prove the property by induction on $u$.

Let $(s, s') \in S^\otimes$, since $\mu_{[0,0]} = 1_{S_a}$ and $s' \in S_b \Rightarrow s \in S_b$,

$\mu_{[0,0]}(s) \geq \mu_{[0,0]}(s')$.

Now we prove the property for interval $[0, u]$ with finite $u$ by induction on $u > 0$ with basis case $u = 0$ already proved. If $s \in S_b$ then $\mu_{[0,u]}(s) = 1 \geq \mu_{[0,u]}(s')$

If $s' \in S_b$ then $s \in S_b$ and $\mu_{[0,u]}(s) = \mu_{[0,u]}(s') = 1$

If $s \in S_{ab}$ then $s' \in S_{ab}$ and $\mu_{[0,u]}(s) = \mu_{[0,u]}(s') = 0$

If $s' \in S_{ab}$ then $\mu_{[0,u]}(s') = 0 \leq \mu_{[0,u]}(s)$

The last case to consider is $s, s' \in S_{ab}$.

$\mu_{[0,u]}(s) = \sum_{s_1 \in S} P((s, s_1) \mu_{[0,u-1]}(s_1))$

$= \sum_{s_1 \in S} \sum_{(s_1, s_1') \in S^\otimes} P^\otimes((s, s_1'), (s_1, s_1')) \mu_{[0,u-1]}(s_1)$

$\geq \sum_{s_1 \in S} \sum_{(s_1, s_1') \in S^\otimes} P^\otimes((s, s_1'), (s_1, s_1')) \mu_{[0,u-1]}(s_1)$

$= \sum_{s_1 \in S} \sum_{(s_1, s_1') \in S^\otimes} P^\otimes((s', s_1'), (s_1, s_1')) \mu_{[0,u-1]}(s_1)$

$= \sum_{s_1 \in S} \sum_{(s_1, s_1') \in S^\otimes} P^\otimes((s', s_1'), (s_1, s_1')) \mu_{[0,u-1]}(s_1)$

$\geq \mu_{[0,u-1]}(s')$. Since $\mu_{[0,\infty]}(s) = \lim_{u \to \infty} \mu_{[0,u]}(s)$, this proves the result for interval $[0, \infty]$.

Example. Let us illustrate coupling for the DTMC represented in figure 4 and called $\mathcal{C}^\ast$. This chain is obtained from the embedded DTMC of the tandem queues by lumping together states which have the same number of clients and at least $R$ clients in the second queue. Its set of state is $S^\ast = \mathbb{N} \times \{0..R\}$. Consider the coupling of this chain with itself defined by $S^\ast = \{(n_1, n_2), (n_2, n_2) | n_1 + n_2 \geq n_1^\prime + n_2^\prime \land n_1 \geq n_1^\prime\}$. Consider now proposition a “There is at least one client in some queue” and proposition b “The sum of the number of clients in both queues is at least $H$” where $H$ is some constant. (we consider that the initial state $s_0$ is the state with one client in the first queue to avoid trivial evaluation). Then $S^\otimes$ is a coupling relation w.r.t. $(a, b)$.

D. An importance sampling framework with variance reduction and confidence interval

The proposed method performs statistical model checking on $\mathcal{C}$ by statistically computing a reachability probability in $\mathcal{C}_u$ using importance sampling with associated matrix obtained by numerical model checking on a reduced chain with requirements given below.

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

\begin{align*}
R & = \rho_{\mathcal{C}^\ast}(s) = \sum_{s' \in S_{\mathcal{C}^\ast}} P^\ast((s, s')) \rho_{\mathcal{C}}(s') \\
R - 1 & = \rho_{\mathcal{C}^\ast}(s) = \sum_{s' \in S_{\mathcal{C}^\ast}} P^\ast((s, s')) \rho_{\mathcal{C}}(s') \\
\end{align*}
which implies the previous one (see later proposition 4) is by $s_u, \mu$, a random variable (r.v.) $V$ that $\mu$ is obtained from the original chain by applying the construction of $C^*$ for this reduction.

Two states $s$ and $s'$ are equivalent if $f(s) = f(s')$, in other words $f^{-1}$ defines equivalence classes for this reduction.

**Example.** In the example of tandem queues, the reduced chain $C^*$ is obtained from the original chain by applying the following function to the state space.

$$f(n_1, n_2) = \begin{cases} (n_1, n_2) & \text{if } n_1 \leq R \\ (n_1 + n_2 - R, R) & \text{otherwise} \end{cases}$$

The intuition behind this reduction is to block clients in the first queue when there are $R$ clients in the second one, thus increasing the probability of a global overflow.

Given some reduced chain $C^*$, our goal is to replace the random variable (r.v.) $V_S$ which takes value in $\{0, 1\}$ by a r.v. $W_S$ which takes value in $\{0, \mu^*(f(s))\}$. This requires that $\mu_u(s_0) \leq \mu^*(f(s_0))$. By applying an homogeneity principle, we get the stronger requirement $\forall s \in S, \forall 0 \leq v \leq u, \mu_u(s) \leq \mu_u^*(f(s))$. In fact, the appropriate requirement which implies the previous one (see later proposition 4) is expressed by the next definition.

**Definition 6:** Let $C$ be a DTMC and $C^*$ a reduction of $C$ by $f$. $C^*$ is a reduction with guaranteed variance if for all $s \in S$ such that $\mu^*_u(f(s)) > 0$ and $\forall 0 < \tau \leq u$ we have:

$$\sum_{s' \in S} \mu^*_u(f(s')) \cdot P(s, s') \leq \mu^*_u(f(s)) \quad (7)$$

Given $s \in S$ and $0 < \tau \leq u$, let $h_\tau(s)$ be defined by $h_\tau(s) = \sum_{s' \in S} \frac{\mu^*_u(f(s'))}{\mu^*_u(f(s))} P(s, s')$. We can now construct an efficient important sampling based on a reduced chain with guaranteed variance.

**Definition 7:** Let $C$ be a DTMC and $C^*$ be a reduction of $C$ by $f$ with guaranteed variance. Then $P'_u$ is the transition matrix on $S_{0u}$ the state space of $C_u$ defined by:

$$P'_u(s, s') = \begin{cases} P_u(s, s') & \text{if } s' \in S_{0u} \\ 0 & \text{otherwise} \end{cases}$$

The following proposition justifies the definition of $P'_u$.

**Proposition 4:** Let $C$ be a DTMC and $C^*$ be a reduction with guaranteed variance. The importance sampling based on matrix of $P'_u$ of definition 7 has the following properties:

- For all $s$ and all $0 < \tau \leq u$ such that $\mu(s, \tau) > 0$, $W(s, \tau)$ is a random variable which has value in $\{0, \mu^*_u(f(s))\}$.
- $\mu_u(s) \leq \mu^*_u(f(s))$ and $V(W(s, \tau)) = \mu_u(s)\mu^*_u(f(s)) - \mu^*_u(s)$.
- One can compute a confidence interval for this importance sampling.

**Proof:** Let $s = (s, v) = u_0, \ldots, u_l = s_+$ be a trajectory starting in $s$ ending in $s_+$. We observe that due to the construction of $C_u$, any $u_k$ can be written as $(s_k, v - k)$ with $0 \leq k < l$.

As the trajectory avoids $s_-$, its value is:

$$\mu^*_u(f(s)) \cdot P_u(s, s_+) = \mu^*_u(f(s_+)) \cdot P_u(s_+, s_+)$$

We know that $E(W(s,v)) = \mu_u(s)$, then $P(W(s,v) = \mu^*_u(f(s))) = P_u(s, s_+) = \frac{\mu^*_u(s)}{\mu^*_u(f(s))}$. This implies that $\mu_u(s) \leq \mu^*_u(f(s))$ and $V(W(s,v)) = \mu_u(s)\mu^*_u(f(s)) - \mu^*_u(s)$. As $W(s,v)$ takes only two values, as for a Bernoulli law, it is possible to compute a confidence interval.

Since $\mu_u(s_0) < 1$, $V(W_{s_0}) = \mu_u(s_0)$. If $\mu_u(s_0) \leq \mu^*_u(f(s_0))$, we obtain $V(W(s_0, u)) \approx \mu_u(s_0)\mu^*_u(f(s_0))$, so the variance is reduced by a factor $\frac{\mu^*_u(s_0)}{\mu^*_u(f(s_0))}$. In the case where $\mu_u(s_0)$ and $\mu^*_u(f(s_0))$ have same magnitude order, the reduction of variance is even bigger.
Unfortunately, Inequality (7) requires to compute the functions \( \mu_{n}^{*} \) in order to check that \( C^{*} \) is a reduction with guaranteed variance. We are looking for a structural requirement that does not involve the computation of \( \mu_{n}^{*} \).

**Proposition 5:** Let \( C \) be a DTMC, \( C^{*} \) be a reduction of \( C \) by \( f \). Assume there exists a family of functions \( (g_{s})_{s \in S} \), \( g_{s} : \{ t \mid P(s, t) > 0 \} \rightarrow S' \) such that:

1. \( \forall s \in S, \forall t' \in S' \), \( P^{*}(f(s), t') = \sum_{s' \mid g(s', s') = t'} P(s, s') \)
2. \( \forall s, t \in S \) such that \( P(s, t) > 0 \forall 0 \leq \tau < u \mu_{\tau}^{*}(f(t)) \leq \mu_{n}^{*}(g_{s}(t)) \)

Then \( C^{*} \) is a reduction of \( C \) with guaranteed variance.

**Proof:** Let \( s \) be a state of \( S \) and \( v > 0 \). We partition the terms of the sum of the inequality (7) according to their images by the function \( g_{s} \):

\[
\sum_{s' \mid P(s, s') > 0} \mu_{n-1}^{*}(f(s')) \cdot P(s, s') = \sum_{s' \in S' \mid g_{s}(s') = s} \mu_{n-1}^{*}(f(s')) \cdot P(s, s')
\]

We apply the second hypothesis:

\[
\leq \sum_{s \in S^*} \sum_{s' \mid g_{s}(s') = s} \mu_{n-1}^{*}(s^*) \cdot P(s, s') = \sum_{s^* \in S^*} \mu_{n-1}^{*}(s^*) \sum_{s' \mid g_{s}(s') = s} P(s, s')
\]

Then the first hypothesis yields:

\[
= \sum_{s^* \in S^*} \mu_{n-1}^{*}(s^*) P(f(s), s^*)
\]

This term is equal to \( \mu_{n}^{*}(f(s)) \) thanks to the equation (5) applied to the Markov chain \( C^{*} \).

The family of functions \( (g_{s}) \) assigns to each transition of \( C \) starting from \( s \) a transition of \( C^{*} \) starting from \( f(s) \). The first condition can be checked by straightforward examination of the probability transition matrices. The second condition still involves the mapping \( \mu^{*} \) but here there are only comparisons between its values. Thanks to proposition 3, it can be proved by exhibiting a coupling of \( C^{*} \) with itself.

**Example.** To apply the method on the example it remains to specify the family of functions \( (g_{s})_{s \in S} \):

\[
g_{(n_{1}, n_{2})}(n_{1}, n_{2}) = f(n_{1}, n_{2})
g_{(n_{1}, n_{2})}(n_{1} + 1, n_{2}) = f(n_{1} + 1, n_{2})
g_{(n_{1}, n_{2})}(n_{1} - 1, n_{2} + 1) = f(n_{1} - 1, n_{2} + 1)
g_{(n_{1}, n_{2})}(n_{1}, n_{2} - 1) = \begin{cases} (n_{1}, n_{2} - 1) & \text{if } n_{2} \leq R \\ (n_{1} + n_{2} - R, R - 1) & \text{else} \end{cases}
\]

The condition 2 always trivially holds except for the last case with \( n_{2} > R \). We have to check that \( \mu_{n}^{*}(n_{1} + n_{2} - 1 - R, R) \leq \mu_{n}^{*}(n_{1} + n_{2} - R, R - 1) \). As \( (n_{1} + n_{2} - R, R - 1), (n_{1} + n_{2} - 1 - R, R) \) belongs to the coupling relation the inequality holds.

**V. ALGORITHMIC CONSIDERATIONS**

Based on the previous developments, we describe a methodology to perform statistical model checking using importance sampling to estimate the tiny probability \( p = \mu_{\tau}(s_{0}) \) in several steps.

1. Specify a model \( M^{*} \) with embedded DTMC \( C^{*} \), a function \( f \) and a family of functions \( (g_{s})_{s \in S} \). The specification of this family is done at the level of models \( M \) and \( M^{*} \) as shown in the example of tandem queues.

2. Specify a coupling satisfying the required properties in order to ensure \( C^{*} \) is a reduction with guaranteed variance (Proposition 5).

3. Fix some uniform rate \( \lambda \) for the uniformization of \( C \). Then, given some numerical precisions \( \alpha \) and \( \beta \), determine the lower and upper bounds \( n^{-} \) and \( n^{+} \) and coefficients \( c_n \) \( (n^{-} \leq n \leq n^{+}) \) for the Poisson distribution with parameter \( \lambda t \) (see section III-D).

4. Compute the distributions \( \{ \mu_{n}^{u} \}_{0 \leq n \leq n^{+}} \) (numerical computations using equations (1) on \( C^{*} \)).

5. Use these distributions to perform importance sampling on the simulation of the initial model in order to estimate \( \mu_{u}(s_{0}) \) for \( n^{-} \leq u \leq n^{+} \) and \( u = \infty \). We generate a large sample of trajectories using the transition system corresponding to matrix \( P_{u}^{'} \) (definition 7) and compute along each path the likelihood \( L \) in order to obtain an estimation with accurate confidence interval.

6. Weight and combine the confidence intervals with equation (3) in order to return the final interval. Since \( \mu_{i}(s_{0}) \) is increasing w.r.t. \( i \), \( u_{i} \) can be chosen to be the upper bound of the confidence interval returned for \( \mu_{i}(s_{0}) \). Similarly the upper bound \( u_{\infty} \) can be chosen to be the upper bound of the confidence interval returned for \( \mu_{\infty}(s_{0}) \) (see [4] for the case \( \infty \)) The threshold probability is computed by equation (2).

The first step requires some understanding of the system to design the appropriate reduced chain. The proof of coupling is done by hand but could be mechanized with a proof assistant. The steps 3, 4 and 6 are standard computations and do not need additional explanations. We now describe in detail the fifth step since it gives algorithmic problems.

We denote by \( m \) the number of states of the Markov chain \( C^{*} \) and by \( d \) the maximum of outdegrees of vertices of \( C^{*} \). Let us remark that in typical modelings, \( d \) is very small compared to \( m \). A simulation takes at most \( u \) steps going through states \( (s_{u}, u), \ldots, (s_{1}, 1), s_{-} \) where \( s_{u} = s_{0} \) and \( s_{-} \in \{ s_{+}, s_{-} \} \). In state \( (s_{v}, v) \), we compute the distribution \( P_{u}^{'}((s_{v}, v), -) \) (cf. definition 7), which requires the values of \( \mu_{n}^{*}(f(s)) \) and \( \mu_{n-1}^{*}(f(s')) \), for each possible target state \( s' \) from \( s_{v} \).

Thanks to equations 1, the vectors \( \{ \mu_{n}^{u} \}_{0 \leq u \leq u} \) may be computed iteratively one from the other with complexity \( \Theta(m d u) \). More precisely, we derive from \( P^{*} \), matrix \( P_{0}^{*} \),
a square (substochastic) matrix, indexed by $S_0 \cup s_+$ and defined by $\forall s, s' \in S_0$: 
$P^*_0(s, s') = P^*(s, s')$, $P^*(s, s_+) = \sum_{s'' \in S_0} P^*(s, s'')$ 
$P^*_0(s_+, s_+) = 1$, $P^*_0(s_+, s') = 0$

Then $\mu_0^* = P^*_0 \cdot \mu_{-1}$ and $\mu_0$ is null except $\mu_0^*(s_+) = 1$.

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 three 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, both methods compute iteratively the $u$ vectors $\mu^*_v = (P^*_0)^v(\mu_0^*)$ 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 1 presented in the appendix. The storage of vectors $\{\mu^*_v\}_{v \leq u}$ 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 $\mu^*_v$ with $\tau$ multiple of $l$ in list $L_s$ and $\mu_{l \cdot i}^*, \mu_{l \cdot i+1}^*, \ldots, \mu^*_u$ in list $K$ (see the precomputation stage of algorithm 2). During the simulation stage, in a state $(s, \tau)$, with $\tau = ml$, the vector $\mu^*_{\tau-1}$ is present neither in $L_s$ nor in $K$. So the method uses the vector $\mu_{l \cdot (m-1)}^*$ stored in $L_s$ to compute iteratively all vectors $\mu^*_{l \cdot (m-1) + i} = P^* (\mu^*_{l \cdot (m-1)})$ for $i$ from $1$ to $l-1$ and store them in $K$ (see the step computation stage of algorithm 2). 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 $L_s$. More precisely, initially using the binary decomposition of $u$ ($u = \sum_{i=0}^{k} a_i, 2^i$), the list $L_s$ of $k+1$ vectors consists of $w_{k,v} = \mu_{\sum_{j=0}^{k} a_j 2^j}$, for all $1 \leq i \leq k+1$ (see the precomputation step of algorithm 3). During the simulation stage in a state $(s, v)$, with the binary decomposition of $v$ ($v = \sum_{i=0}^{k} a_i, 2^i$), the list $L_s$ consists of $w_{k,v} = \mu_{\sum_{j=0}^{k} a_j 2^j}$, for all $1 \leq i \leq k+1$. Observe that the first vector $w_{k,1}$ is equal to $\mu^*_1$.

We obtain $\mu^*_{v-1}$ by updating $L_s$ according to $v$. Let us describe the updating of the list performed by the step computation of algorithm 3. Let $i_0$ be the smallest index such that $a_{i_0} = 1$. Then for $i > i_0$, $a_{i_1} = a_{i_0}$, $a_{i-1} = 0$ and for $i < i_0$, $a_{i-1} = 1$. The new list $L_s$ is then obtained as follows. For $i < i_0$ $w_{k+1,v} = w_{k,v}$, $w_{i_0,v} = w_{i_0,v-1}$. Then the vectors for $i = i_0$ are stored along iterated $2^{i_0-1}$ matrix-vector products starting from vector $w_{i_0,v-1}$: $w(j, v-1) = P^* w(j+1, v-1)$.

The computation associated with $v$ requires $1 + 2 + \ldots + 2^{i_0-1}$ products matrix-vector, i.e. $\Theta(mld^{2u})$. Noting that the bit $i$ is reset at most $m2^{-i}$ times, the complexity of the whole computation is $\sum_{i=1}^{k} 2^{k-i} \Theta(ml2^i) = \Theta(mdu \log(u))$.

The three methods are numbered according to their decreasing space complexity. The corresponding space-time trade-off is summarized by table 1, where the space unit is the storage of a float.

### Algorithm 1:

**Precomputation**($u, \mu_0^*, P_0^*$)

**Result:** $L_s$

// List $L_s$ fulfills $L_s(i) = \mu^*_i$

1. $v \leftarrow \mu_0^*$
2. for $i = 1$ to $u$
   3. $v \leftarrow P_0^* v$
   4. $L_s(i) \leftarrow v$

### Algorithm 2:

**Precomputation**($u, \mu_0^*, P_0^*$)

**Result:** $L_s, K$

// List $L_s$ fulfills $L_s(i) = \mu^*_i$

1. $l \leftarrow \lfloor \sqrt{u} \rfloor$
2. $w \leftarrow \mu_0^*$
3. for $i$ from $l$ to $\lfloor \frac{u}{l} \rfloor$
   4. $w \leftarrow P_0^* w$
   5. if $i \mod l = 0$
      6. $L_s(i) \leftarrow w$

// List $K$ contains $\mu^*_{\lfloor \frac{u}{l} \rfloor + 1}, \ldots, \mu_0^*$

7. for $i$ from $\lfloor \frac{u}{l} \rfloor + 1$ to $u$
   8. $w \leftarrow P_0^* w$
   9. $K(i \mod l) \leftarrow w$

**Stepcomputation($v, l, P_0^*, K, L_s$)**

// Updates $K$ when needed

10. if $v \mod l = 0$
   11. $w \leftarrow L_s(\frac{v}{l} - 1)$
   12. for $i$ from $\lfloor \frac{v}{l} \rfloor + 1$ to $v - 1$
      13. $w \leftarrow P_0^* w$
      14. $K(i \mod l) \leftarrow w$

### VI. Experimentation

#### A. Implementation

**Tools.** Our experiments have been performed on COSMOS, a statistical model checker whose input model is a stochas-
**Algorithm 3:**

Precomputation\((u, \mu_0^\bullet, P_0^\bullet)\)

Result: \(L_s\)

// \(L_s\) fulfills \(L_s(i) = \mu_{k-u}^\bullet \sum_{j=1}^{u} a_{u,j} 2^j\)

1. \(k \leftarrow \lceil \log_2(u) \rceil + 1\)
2. \(v \leftarrow \mu_0^\bullet\)
3. \(L_s(k+1) \leftarrow v\)
4. for \(i\) from \(k\) downto 0 do
   5. if \(a_{u,i} = 1\) then
      6. for \(j\) from 1 to \(2^i\) do
         7. \(w \leftarrow P_0^\bullet w\)
     8. \(L_s(i) \leftarrow w\)
5. Stepcomputation\((v, l, P_0^\bullet, L_s)\)

// \(L_s\) is updated accordingly to \(v - 1\)

10. \(i_0 \leftarrow \min(i \mid a_{u,i} = 1)\)
11. \(w \leftarrow L_s(i_0 + 1)\)
12. \(L_s(i_0) \leftarrow v\)
13. for \(i\) from \(i_0 - 1\) downto 0 do
   14. for \(j\) to 1 to \(2^i\) do
      15. \(w \leftarrow P_0^\bullet w\)
   16. \(L_s(i) \leftarrow w\)

<table>
<thead>
<tr>
<th>Complexity</th>
<th>Method 1</th>
<th>Method 2</th>
<th>Method 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space</td>
<td>(m\dfrac{u}{u})</td>
<td>(2m\sqrt{u})</td>
<td>(m\log u)</td>
</tr>
<tr>
<td>Time for the precomputation</td>
<td>(\Theta(mdu))</td>
<td>(\Theta(mdu))</td>
<td>(\Theta(mdu))</td>
</tr>
<tr>
<td>Additional time for the simulation</td>
<td>0</td>
<td>(\Theta(mdu))</td>
<td>(\Theta(mdu\log(u)))</td>
</tr>
</tbody>
</table>

**Table 1: COMPARED COMPLEXITIES**

Adaptation of COSMOS. In addition to the implementation of our algorithms, we have done two main modifications on the tool in order to integrate our method. First, the probabilities of the Poisson distribution are computed by a freely available implementation of the Fox-Glynn algorithm [22]. Second, COSMOS sequentially generates a batch of trajectories. In our context this is highly inefficient since the numerical computations of \(\mu_n^\bullet\) required by algorithms 2 and 3 should be repeated for every trajectory. So one generates a bunch of trajectories in parallel step by step. Different sizes of bunches are possible but they cannot exceed the size required for the numerical computations. Based on the asymptotic time and space cost of these computations, we handle \(m^2\) trajectories.

**B. Global Overflow in Tandem Queues**

**Generalisation of the example.** The first experimentation consists of tandem queues with \(k\) queues generalising the illustrative example of previous sections. A client arrives in the first queue with rate \(\rho_0\). In queue \(i\) \((i < k)\), a client is served with rate \(\rho_i\) and then go to the next queue. In the last queue, clients leave the system with rate \(\rho_k\). In the reduced model, the length (number of clients) of every queue except the first one is bounded by some parameter \(R\), equipped with a coupling relation.

**Proof of guaranteed variance**

**Proof:** For our purpose, we enrich the usual definition of Markov chains with labels on transitions and a notion of equivalence of Markov chains.

**Definition 8:** A discrete time Markov chain (DTMC) \(C\) is defined by a set of states \(S\), an initial state \(S_0\), a finite set of events \(E\), a successor function \(\delta : S \times E \rightarrow S\), and a function \(p : S \times E \rightarrow [0,1]\) with the property that for all \(s \in S\), \(\sum_{e \in E} p(s,e) = 1\). We define the transition probability matrix \(P\) of size \(S \times S\) by:

\[
\forall s, s' \in S, \quad P(s,s') = \sum_{\delta(s,e) = s'} p(s,e)
\]

Two Markov chains are **equivalent** if they have the same set of states \(S\), respective probability distribution matrices \(\mathbf{P}\), \(\mathbf{P}'\) and for all state \(s\) with \(q = P(s,s)\), \(q' = P'(s,s)\) we have the following equalities:

\[
\forall s' \neq s, \quad \frac{P(s,s')}{1-q} = \frac{P'(s,s')}{1-q'}
\]

This equivalence is used in an implicit way in the proofs. We will often omit self-loops: we consider that a self-loop always exists with a probability such that the sum of all outgoing transitions probability is equal to 1.

We explicit the embedded DTMC associated with the CTMC defined by the \(k\) tandem queues models. The property for which one estimates the probability is: “The system contains \(H\) clients before returning to the idle state (no clients)”.

- The set of states \(S\) is \(\{(n_1,\ldots,n_k) \mid \sum_{i=1}^{k} n_i \leq H\}\) with initial state \(S_0 = (1,0,\ldots,0)\). For a state \(s = (n_1,\ldots,n_k)\), \(n_i\) represents the number of clients in the \(i^{th}\) queue.

---

\(^2\)Here, \(K\) means Kilobyte, \(M\) means Megabyte and \(G\) means Gigabyte.
The set of events is $E = \{\gamma_0, \gamma_1, \cdots, \gamma_k\}$. $\gamma_0$ labels the entrance of a client in the first queue, which follows a Poisson law with rate $\rho_0$. For $i > 0$, $\gamma_i$ labels the service of a client in the $i^{th}$ queue, which follows a Poisson law with rate $\rho_i$. When this happens, the client leaves the $i^{th}$ queue for the next one if $i < k$ or leaves the $k^{th}$ queue.

Let us describe the transitions leaving a state $s = (n_1, n_2, \cdots, n_k)$:

- If $\sum_{i=1}^{k} n_i = H$ then $\delta(s, \gamma_0) = s$
- else $\delta(s, \gamma_0) = (n_1 + 1, n_2, \cdots, n_k)$

For all $i \leq k - 1$

- If $n_i > 0$ then $\delta(s, \gamma_i) = (n_1, \cdots, n_i - 1, n_{i+1} + 1, \cdots, n_k)$
- If $n_k > 0$ then $\delta(s, \gamma_k) = (n_1, n_2, \cdots, n_k - 1)$

In the reduced model, the length of all queues except the first one is bounded by $R$. The function $\delta^*$ is the transition function in the reduced chain.

$\delta^*(s, \gamma_i) = s$ if $1 \leq i < k$ and $n_{i+1} = R$.

In all other cases $\delta^* = \delta$.

The reduction function $f$ is defined using for each queue $i$ two recursive functions $f_i$ which basically computes the $i^{th}$ component of $f$ and $r_i$.

\[
\begin{align*}
    f_k(s) &= \min(n_k, R) \\
    r_k(s) &= \max(n_k - R, 0) \\
    f_i(s) &= \min(n_i + r_{i+1}(s), R) \text{ if } 1 < i < k \\
    r_i(s) &= \max(n_i + r_{i+1}(s) - R, 0) \text{ if } 1 < i < k \\
    f_i(s) &= n_i + r_2(s)
\end{align*}
\]

The reduction function $f$ is then defined by:

\[f(s) = (f_1(s), f_2(s), \cdots, f_k(s))\]

We define the family of functions $g_{s}(g_{s'})_{s \in S}$:

\[g_x(\delta(s, \gamma_i)) = \delta^*(f(s), \gamma_i) \text{ for } 0 \leq i \leq k\]

We define the coupling relation $S^\circ$:

Let $s = (n_1, \cdots, n_k)$ and $s' = (n'_1, \cdots, n'_k)$ be in $S$. $(s, s') \in S^\circ$ iff

\[
\begin{align*}
    n_1 &\geq n'_1 \\
    \land \quad n_1 + n_2 &\geq n'_1 + n'_2 \\
    \land \quad \cdots \\
    \land \quad n_1 + n_2 + \cdots + n_k &\geq n'_1 + n'_2 + \cdots + n'_k
\end{align*}
\]

Let us prove that this relation is a coupling relation.

We need to check that for all couples $(s, s')$ in the relation, all successors also belong to the relation. We define the property $P_t(s)$ to be true if and only if $n_i > 0 \land n_{i+1} < R$ for all $0 < i < k$.

1) For transition $\gamma_0$, the successor of $(s, s')$ is $((n_1 + 1, \cdots, n_2), (n'_1 + 1, \cdots, n'_2))$ which is inside the relation.

2) For transition $\gamma_i$ with $0 < i < k$, the successor of $(s, s')$ is $((n_1, \cdots, n_i - 1 P_i(s), n_{i+1} + 1 P_i(s'), \cdots, n_k), (n'_1, \cdots, n'_i - 1 P_i(s'), n_{i+1} + 1 P_i(s'), \cdots, n'_k))$ All the conditions except condition (i) are trivially satisfied as the corresponding sums are not modified.

Let us check the $i^{th}$ condition:

- If $n_1 + n_2 + \cdots + n_i > n'_1 + n'_2 + \cdots + n'_i$ then $n_1 + \cdots + n_i - 1 P_i(s) \geq n'_1 + \cdots + n'_i - 1 P_i(s')$ and the condition (i) is satisfied.
propositions.

So let us check that the model satisfies the hypotheses of Proposition 5. First condition holds because the set of transitions \( \{ \gamma_0, \gamma_1, \ldots, \gamma_k \} \) is the same for the reduced model and the original one:

\[
\forall s \in S, \forall t^* \in S^*, \mathbf{P}^\star(f(s), t^*) = \sum_{e \in E} \mathbf{P}^\star(f(s), e) = \sum_{e \in E} p(f(s), e) = \sum_{e \in E} p(s, e) = \sum_{s' \mid g(s, s') = t^*} \mathbf{P}(s, s').
\]

Thanks to the definition of \( g_0 \) and to the coupling relation that we have established for \( \hat{C}^\star \), the second condition holds if we prove that:

\[
\forall 0 \leq i \leq k \quad (\delta^\star(f(s), \gamma_i), f(\delta(s, \gamma_i))) \in S^\otimes
\]

We denote \( s = (n_1, \ldots, n_k) \) and \( f(s) = (n_1^*, \ldots, n_k^*) \). We first observe that if \( i < k \) and \( n_i^* > 0 \) and \( n_{i+1} < R \) then \( ((n_1^*, \ldots, n_k^*), (n_1^* - 1, n_{i+1}^* + 1, n_i^*)) \in S^\otimes \). We also observe that: \( n_i^* < R \) implies \( n_i \leq n_i^* \).

Case \( i = 0 \)

\[
f(\delta(s, \gamma_0)) = \delta^\star(f(s), \gamma_i) = (n_1^* + 1, n_2^*, \ldots, n_k^*).
\]

Case \( 0 < i \) and \( n_i^* = 0 \)

Then \( n_i \leq n_i^* = 0 \).

So \( \delta(s, \gamma_i) = s \) and \( f(\delta(s, \gamma_i))) = f(s) \).

On the other hand, \( \delta^\star(f(s), \gamma_i)) = f(s) \).

Case \( 0 < i < k \) and \( n_i^* = 0 \) and \( n_{i+1}^* = R \)

Then by definition \( \delta^\star(f(s), \gamma_i)) = f(s) \).

On the other hand:

Either \( n_i = 0 \) and \( \delta(s, \gamma_i)) = s \).

Or \( (\delta(s, \gamma_i)) = (n_1, \ldots, n_i - 1, n_i + 1, \ldots, n_k) \).

Since \( n_{i+1}^* = R \),

\[
f_{i+1}(n_1, \ldots, n_i - 1, n_i + 1, n_k) = f_{i+1}(s), \quad r_{i+1}(n_1, \ldots, n_i - 1, s_i + 1, n_k) = r_{i+1}(s) + 1
\]

and so using the recursive definition,

\[
f_i(n_1, \ldots, n_i - 1, n_i + 1, \ldots, n_k) = f_i(s)
\]

and (when \( i > 1 \))

\[
r_i(n_1, \ldots, n_i - 1, n_i + 1, \ldots, n_k) = r_i(s)
\]

which establishes that \( f(\delta(s, \gamma_i)) = f(s) \).

Case \( 0 < i < k \) and \((i < k \) and \( n_{i+1}^* < R) \) or \( i = k \)

\[
\delta^\star(f(s), \gamma_i)) = (n_1^*, \ldots, n_i^* - 1, n_{i+1}^* + 1, \ldots, n_k^*).
\]

On the other hand,

Since \( n_{i+1}^* < R \) (or \( i = k \)), one has \( r_{i+1}(s) = 0 \).

So \( n_i \geq n_i^* > 0 \).

\[
\delta(s, \gamma_i)) = (n_1, \ldots, n_i - 1, n_i + 1, \ldots, n_k).
\]

First one has for all \( j > i \),

\[
f_j((n_1, \ldots, n_i - 1, n_i + 1, \ldots, n_k)) = n_i^*.
\]

Then since \( n_{i+1}^* < R \),

\[
f_{i+1}(n_1, \ldots, n_i - 1, n_i + 1, \ldots, n_k) = n_{i+1}^* + 1
\]

and \( r_{i+1}(n_1, \ldots, n_i - 1, n_i + 1, \ldots, n_k)) = 0 \).

Let \( i' \leq i \) be the greatest index such that either \( \sum_{j=i'} n_j \leq (i - i') + 1 \) or \( i' = 1 \).

Then by definition of the \( f_j \)'s and \( r_i \)'s:

\[
\forall i' < j \leq i \quad f_j((n_1, \ldots, n_i - 1, n_{i+1} + 1, \ldots, n_k)) = n_i^*.
\]

\[
\forall j < i' \quad f_i((n_1, \ldots, n_i - 1, n_{i+1} + 1, \ldots, n_k)) = n_i^*.
\]

Since \( i' \leq i \), it is routine to check that:

\[
((n_1^*, \ldots, n_i - 1, n_{i+1} + 1, \ldots, n_k^*), (n_1^*, \ldots, n_{i+1} + 1, \ldots, n_k^*)) \in S^\otimes
\]

So hypotheses of Proposition 5 are satisfied.

Modeling considerations. This example is a classical benchmark for importance sampling. It has also practical interest as a standard modeling of networks [23]. Such a modeling allows to accurately dimension a network for a given loadwork. Furthermore it illustrates the advantage of dynamic importance sampling w.r.t. the static one [15].

We choose the parameters of the system and of the formula as follows. \( \rho_0 = 0.25 \) and for all \( 1 \leq i \leq \rho_i = 0.375 \).

The threshold corresponding to the number of clients for the considered formula is \( H = 50 \). We study the behaviour of the methods for different values of \( k \). We have chosen for the reduced model \( R = 5 \) as we experimentally found that this value of \( R \) yields a tight confidence interval. The horizon \( \tau \) for the “bounded until” formula is set to 100.

The confidence level for the simulation of the embedded DTMC corresponds to \( \varepsilon_n = 10^{-6} \) for every \( n \). As described in the next paragraph, \( \alpha \) can be set to 0. We choose \( \beta = 10^{-10} \). Finally we have generated 1000 simulations.
to estimate every $\mu_n(s_0)$.

**Preliminary observation.** We plotted in figure 5 the curves $\mu_n(s)$, $\sum_{n=0}^{\infty} \frac{n^\lambda}{n!} \mu_n(s)$ and $\sum_{n=0}^{\infty} \frac{n^\lambda}{n!} \mu_n(s)$. We observe that for $n < 50$, $\mu_n(s_0) = 0$ whereas the Poisson probability for such a $n$ is not null. Therefore we choose a bigger left truncation index $n^-$ for the uniformisation than the one given by the Fox-Glynn algorithm. As no error is done by the left truncation, $e^- = 0$ and $\alpha = 0$. On the right part of the Poisson distribution we notice that after the maximum ($n = 100$) the curve decreases while the curve of $\mu_n$ increases. Thus the maximum of the product is shifted to the right compared to the maximum of the Poisson probabilities. In order to get a confidence interval of $10^{-1} \mu_n(s_0)$ a big enough right truncation index is required. For $\beta = 10^{-10}$ Fox-Glynn algorithm proposes a right truncation index equal to 206.

**Analysis of confidence interval.** We collected our results with respective time and space consumption for the three algorithms and PRISM in table II. We also computed the value $\mu_\infty$ with $\epsilon_\infty = 0.001$ estimated with COSMOS with method described in [4]. The overall threshold probability is equal to $(n^+ - n^-) \epsilon_\infty = 157 - 10^{-10} + 0.001 = 0.001157$. In all experiments, the width of the confidence interval is ten times smaller than the estimated value. Moreover when the numerical computation terminates, the result belongs to the confidence interval. With our choice of $\beta$, the contribution of the right truncation of the Poisson distribution to the length of the confidence interval is several magnitude orders less than the contribution associated with the statistical estimations. So in order to reduce this length, we should increase the number of simulations letting unchanged the truncation index $n^+$.

**Analysis of numerical and statistical PRISM.** We compare our method to numerical and statistical model checking done by PRISM. Due to the rarity of the considered event the statistical approach always fails returning 0. We observe that for small models ($k \leq 4$), PRISM numerical model checker is faster and uses less memory than COSMOS. For $k = 5$, our method is 10 times faster and uses up to 28 times less memory. For $k \geq 6$, PRISM crashes due to a lack of space memory.

**Comparison of the three methods.** While the empirical storage behaviour of the three methods follows the theoretical study, memory does not constitute a bottleneck until $k = 8$. For this value, memory required by method 1 is too important. In order for method 2 to fail, farther time horizons should be chosen.

**Parallelism.** COSMOS can take advantage of a multicore architecture to speed up the simulations. Table II reports experimentations done without exploiting parallelism. In order to analyze the improvements obtained by the parallelism, we performed additional simulations on 10 parallels threads. Since numerical computations are not parallelised, the speed up for method 1 (resp. methods 2 and 3) is about 7 (resp. about 6 and 5).

**C. Parallel Random Walk**

The second experiment consists of a parallel random walk of $k$ walkers. A walk is done between position 1 and position $\max$ starting in position $\max/2$ and ends up in the extremal positions. At every round, some random walker can randomly go in either direction. However, when walkers $i$ and $i + 1$ are in the same position, walker $i$ can only go toward $1$. This model is a paradigm of failure tolerant systems in which the same task is performed by several processors (the walkers). The result of the task is obtained by getting (if it exists) the result returned by a majority of processors. Every processor returns a correct result when reaching position 1 and an unpredicted result when reaching position $\max$. So the probability that a majority of walkers have reached position $\max$ corresponds to the failure probability of the whole system.

The Markov chain associated with the model has $\max^k$ states. In order to get a reduced model, we remove all synchronisation between walkers. Behaviours of all walkers are now independent and thus a state of the reduced system is now defined by the number of walkers in each position. Proposition 5 holds for this reduced model. Intuitively, removing synchronisation between walkers increases the probability to reach position $\max$.

**Proof of guaranted variance**

**Proof:**

The Petri net depicted in figure 6 models a parallel random walk of $k$ walkers. A walk is done between position 1 and position $\max$ starting in position $\max/2$ and ends up in the extremal positions. At every round, some random walker can randomly go in either direction. However, when walkers $i$ and $i + 1$ are in the same position, walker $i$ can only go toward 1. We represent this figure the walker $i$ and his interactions with walker $i + 1$. Transition $A_{i,j}$ (resp. $R_{i,j-1}$) corresponds to a step toward $\max$ (resp. 1).

This model has $\max^k$ states. In order to get a reduced model, we remove all synchronisation between walkers. Behaviours of all walkers are now independent and thus a state of the reduced system is now defined by the number of walkers in each position. The size of the reduced system is $\max^{(k_1 + \max_{1-1})}$.

Proposition 5 holds for this reduced model: the required inequality is satisfied because removing synchronisation between walkers increases the probability to reach position $\max$.

Table III shows the experimental results with the following parameters. The rate for a walker to go toward position...
max is 0.3 and the rate to go toward position 1 is 0.7. We choose \( \text{max} = 10 \). The simulation is stopped when the confidence interval width reaches one half of the estimated value. Our method handles huge models (with size up to \( 10^{11} \)) with small probabilities \( (10^{-11}) \) with reasonable time and space complexities. On the contrary, the \text{PrISM} statistical model checker and numerical model checker fail due to either the low probability (for every value of \( k \)) or the size of the system (when \( k \geq 10 \)).

VII. Conclusion

We proposed a method of statistical model checking in order to compute with accuracy a tiny probability associated with a timed temporal formula on a CTMC. We obtain a reliable confidence interval bounding this value. We have developed a theoretical framework justifying the validity of a confidence interval and ensuring the reduction of the variance. As the memory requirements (which depend on the time horizon) put a kurb on the efficiency of the method, we propose three algorithms with a different trade-off between time and space. We have implemented these algorithms in the statistical model checker \text{Cosmos} and we have done experiments on two classical relevant models.

We plan to go further in several directions. Our first goal is to deal with infinite models whose reduction yields an infinite one and more expressive language logical formula like CSL-TA [24]. We also consider to mechanise the proofs of coupling using an assistant prover, since it consists in checking parametrised inequalities. Finally we aim at defining formalisms on which the reduced model can be automatically produced.

REFERENCES


