An Effective Heuristic for Adaptive Importance Splitting in Statistical Model Checking

Cyrille Jegourel, Axel Legay and Sean Sedwards

{cyrille.jegourel,axel.legay,sean.sedwards}@inria.fr

Abstract Statistical model checking avoids the intractable growth of states associated with numerical model checking by estimating the probability of a property from simulations. Rare properties pose a challenge because the relative error of the estimate is unbounded. In [13] we describe how importance splitting may be used with SMC to overcome this problem. The basic idea is to decompose a logical property into nested properties whose probabilities are easier to estimate. To improve performance it is desirable to decompose the property into many equi-probable *levels*, but logical decomposition alone may be too coarse.
In this article we make use of the notion of a *score function* to improve the granularity of a logical property. We show that such a score function may take advantage of heuristics, so long as it also rigorously respects certain properties. To demonstrate our importance splitting approach we present an optimal adaptive importance splitting algorithm and an heuristic score function. We give experimental results that demonstrate a significant improvement in performance over alternative approaches.

1 Introduction

Model checking offers the possibility to automatically verify the correctness of complex systems or detect bugs [7]. In many practical applications it is also useful to quantify the probability of a property (e.g., system failure), so the concept of model checking has been extended to probabilistic systems [1]. This form is frequently referred to as *numerical* model checking.

To give results with certainty, numerical model checking algorithms effectively perform an exhaustive traversal of the states of the system. In most real applications, however, the state space is intractable, scaling exponentially with the number of independent state variables (the 'state explosion problem' [6]). Abstraction and symmetry reduction may make certain classes of systems tractable, but these techniques are not generally applicable. This limitation has prompted the development of *statistical* model checking (SMC), which employs an executable model of the system to estimate the probability of a property from simulations.

SMC is a Monte Carlo method which takes advantage of robust statistical techniques to bound the error of the estimated result (e.g., [18,22]). To quantify a property it is necessary to observe the property, while increasing the number of observations generally increases the confidence of the estimate. Rare properties

are often highly relevant to system performance (e.g., bugs and system failure are required to be rare) but pose a problem for statistical model checking because they are difficult to observe. Fortunately, rare event techniques such as *importance sampling* [14,16] and *importance splitting* [15,16,20] may be successfully applied to statistical model checking.

Importance sampling and importance splitting have been widely applied to specific simulation problems in science and engineering. Importance sampling works by estimating a result using weighted simulations and then compensating for the weights. Importance splitting works by reformulating the rare probability as a product of less rare probabilities conditioned on levels that must be achieved.

Recent work has explicitly considered the use of importance sampling in the context of statistical model checking [19,11,2,12]. Some limitations of importance sampling are discussed in [13]. In particular, it remains an open problem to quantify the performance of the importance sampling *change of measure*. A further numerical challenge arises from properties and systems that require very long simulations. In these cases the change of measure is only very subtly different to the original measure and may be difficult to represent with standard fixed length data types.

Earlier work [21,10] extended the original applications of importance splitting to more general problems of computational systems. In [13] we proposed the use of importance splitting for statistical model checking, specifically linking the concept of levels and score functions to temporal logic. In that work we considered two algorithms which make use of a fixed number of simulations per level. The first algorithm is based on fixed levels, chosen a priori, whose probabilities may not be equal. The second algorithm finds levels adaptively, evenly distributing the probability between them. In so doing, the adaptive algorithm reduces the relative error of the final estimate.

In what follows we show that importance splitting is an effective technique for statistical model checking and discuss the important role of the score function. We motivate and demonstrate the need for fine grained score functions that allow adaptive algorithms to find optimal levels. We then present a fine grained heuristic score function and an optimal adaptive importance splitting algorithm that improve on the performance of previous algorithms. We perform a set of experiments to illustrate both advantages and drawbacks of the technique.

The remainder of the paper is organised as follows. Section 2 defines the notation used in the sequel and introduces the basic notions of SMC applied to rare properties. Section 3 introduces the specific notions of importance splitting and score functions. Section 4 gives our importance splitting algorithms, while Section 5 illustrates their use on the dining philosophers protocol.

2 Statistical model checking rare events

We consider stochastic discrete-event systems. This class includes any stochastic process that can be thought of as occupying a single state for a duration of time before an instantaneous transition to a new state. In particular, we consider

systems described by discrete and continuous time Markov chains. Sample execution paths can be generated by efficient discrete-event simulation algorithms (e.g., [9]). Execution paths are sequences of the form $\omega = s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} s_2 \xrightarrow{t_2} ...$, where each s_i is a state of the model and $t_i \in \mathbb{R} > 0$ is the time spent in the state s_i (the delay time) before moving to the state s_{i+1}. In the case of discrete time, $t_i \equiv 1, \forall i$. When we are not interested by the times of jump epochs, we denote a path $\omega = s_0 s_1 ....$ The length of path ω includes the initial state and is denoted $|\omega|$. A prefix of ω is a sequence $\omega_{\leq k} = s_0 s_1 ... s_k$ with $k < |\omega| \in \mathbb{N}$. We denote by $\omega_{\geq k}$ the suffix of ω starting at s_k.

In the context of SMC we consider properties specified in bounded temporal logic, which may evaluate to true or false when applied to a specific path. Given a stochastic system and a bounded temporal logic property φ, our objective is to calculate the probability γ that an arbitrary execution trace ω satisfies φ, denoted $\gamma = \mathrm{P}(\omega \models \varphi)$. Let Ω be the set of paths induced by the initial state of the system, with $\omega \in \Omega$ and f a probability measure over Ω. To decide the truth of a particular trace ω', we define a model checking function z from Ω to $\{0, 1\}$ that takes the value 1 if $\omega' \models \varphi$ and 0 if $\omega' \not\models \varphi$. Thus,

$$\gamma = \int_{\Omega} z(\omega)\, \mathrm{d}f \qquad \text{and} \qquad \tilde{\gamma} = \frac{1}{N} \sum_{i=1}^{N} z(\omega_i)$$

N denotes the number of simulations and ω_i is sampled according to f. Note that the estimate $\tilde{\gamma}$ is distributed according to a binomial distribution with parameters N and γ. Hence $\mathrm{Var}(\tilde{\gamma}) = \gamma(1-\gamma)/N$ and for $\gamma \to 0$, $\mathrm{Var}(\tilde{\gamma}) \approx \gamma/N$.

When a property is not rare there are useful bounding formulae (e.g., the Chernoff bound [18]) that relate *absolute* error, confidence and the required number of simulations to achieve them. As the property becomes rarer, however, absolute error ceases to be useful and it is necessary to consider *relative* error, defined as the standard deviation of the estimate divided by its expectation. For the binomial random variable described above the relative error of an estimate is given by $\sqrt{\gamma(1-\gamma)}/N\gamma$, which is unbounded as $\gamma \to 0$. In standard Monte Carlo simulation, γ is the expected fraction of executions in which the rare event will occur. If the number of simulation runs is significantly less than $1/\gamma$, as is often necessary when γ is very small, no occurrences of the rare property will likely be observed. A number of simulations closer to $100/\gamma$ is desirable to obtain a reasonable estimate. Hence, importance sampling and importance splitting have been developed to reduce the number of simulations required or, equivalently, to reduce the variance of the rare event and so achieve greater confidence for a given number of simulations. In this work we focus on importance splitting.

3 Importance splitting

The earliest application of importance splitting is perhaps that of [14,15], where it is used to calculate the probability that neutrons pass through certain shielding materials. This physical example provides a convenient analogy for the more

general case. The system comprises a source of neutrons aimed at one side of a shield of thickness T. It is assumed that neutrons are absorbed by random interactions with the atoms of the shield, but with some small probability γ it is possible for a neutron to pass through the shield. The distance travelled in the shield can then be used to define a set of increasing levels $l_0 = 0 < l_1 < l_2 < \cdots < l_n = T$ that may be reached by the paths of neutrons. Importantly, reaching a given level implies having reached all the lower levels. Though the overall probability of passing through the shield is small, the probability of passing from one level to another can be made arbitrarily close to 1 by reducing the distance between the levels.

These concepts can be generalised to simulation models of arbitrary systems, where a path is a simulation trace. By denoting the abstract level of a path as l, the probability of reaching level l_i can be expressed as $\mathrm{P}(l > l_i) = \mathrm{P}(l > l_i \mid l > l_{i-1})\mathrm{P}(l > l_{i-1})$. Defining $\gamma = \mathrm{P}(l > l_n)$ and observing $\mathrm{P}(l > l_0) = 1$, it is possible to write

$$\gamma = \prod_{i=1}^{n} \mathrm{P}(l > l_i \mid l > l_{i-1}) \tag{1}$$

Each term of the product (1) is necessarily greater than or equal to γ. The technique of importance splitting thus uses (1) to decompose the simulation of a rare event into a series of simulations of conditional events that are less rare. There have been many different implementations of this idea, but a generalised procedure is as follows.

Assuming a set of increasing levels is defined as above, at each level a number of simulations are generated, starting from a distribution of initial states that correspond to reaching the current level. The procedure starts by estimating $\mathrm{P}(l \geq l_1 | l \geq l_0)$, where the distribution of initial states for l_0 is usually given (often a single state). Simulations are stopped as soon as they reach the next level; the final states becoming the empirical distribution of initial states for the next level. Simulations that do not reach the next level (or reach some other stopping criterion) are discarded. In general, $\mathrm{P}(l \geq l_i | l \geq l_{i-1})$ is estimated by the number of simulation traces that reach l_i, divided by the total number of traces started from l_{i-1}. Simulations that reached the next level are continued from where they stopped. To avoid a progressive reduction of the number of simulations, the generated distribution of initial states is sampled to provide additional initial states for new simulations, thus replacing those that were discarded.

In physical and chemical systems, distances and quantities may provide a natural notion of level that can be finely divided. In the context of model checking arbitrary systems, variables may be Boolean and temporal logic properties may not contain an obvious notion of level. To apply importance splitting to statistical model checking it is necessary to define a set of levels based on a sequence of logical properties, φ_i, that have the characteristic

$$\varphi = \varphi_n \Rightarrow \varphi_{n-1} \Rightarrow \cdots \Rightarrow \varphi_0 \tag{2}$$

Each φ_i is a strict restriction of the property φ_{i-1}, formed by the conjunction of φ_i with property ψ_i, such that $\varphi_i = \varphi_{i-1} \wedge \psi_i$, with $\varphi_0 \equiv \top$. Hence, φ_i can be

written $\varphi_i = \bigwedge_{j=1}^{i} \psi_j$. This induces a strictly nested sequence of sets of paths $\Omega_i \subseteq \Omega$:

$$\Omega_n \subset \Omega_{n-1} \subset \cdots \subset \Omega_0$$

where $\Omega_i = \{\omega \in \Omega : \omega \models \varphi_i\}$, $\Omega_0 \equiv \Omega$ and $\forall \omega \in \Omega, \omega \models \varphi_0$. Thus, for arbitrary $\omega \in \Omega$,

$$\gamma = \prod_{i=1}^{n} \mathrm{P}(\omega \models \varphi_i \mid \omega \models \varphi_{i-1}),$$

which is analogous to (1).

A statistical model checker works by constructing an automaton to accept traces that satisfy the specified property. In the context of SMC, importance splitting requires that the state of this automaton be included in the final state of a trace that reaches a given level. In practice, this means storing the values of the counters of the loops that implement the time bounded temporal operators.

The choice of levels is crucial to the effectiveness of importance splitting. To minimise the relative variance of the final estimate it is desirable to choose levels that make $\mathrm{P}(\omega \models \varphi_i \mid \omega \models \varphi_{i-1})$ the same for all i (see, e.g., [8]). A simple decomposition of a property may give levels with widely divergent conditional probabilities, hence [13] introduces the concept of a *score function* and techniques that may be used to increase the possible granularity of levels. Given sufficient granularity, a further challenge is to define the levels. In practice these might be guessed or found by trial and error, but Section 4 gives algorithms that find optimal levels adaptively.

Score functions

The goal of a score function S is to discriminate good paths from bad with respect to the property of interest. This is often expressed as a function from paths to $\mathbb{R}$, assigning higher values to paths which more nearly satisfy the overall property. Standard statistical model checking can be seen as a degenerate case of splitting, in the sense that computing $P(\omega \models \varphi)$ is equivalent to compute $P(S(\omega) \geq 1)$ with the functional equality $S = z$, where z is the Bernoulli distributed model checking function.

Various ways to decompose a temporal logic property are proposed in [13]. Given a sequence of nested properties $\varphi_0 \Leftarrow \varphi_1 \Leftarrow \cdots \Leftarrow \varphi_n = \varphi$, one may design a function which directly correlates logic to score. For example, a simple score function may be defined as follows:

$$S(\omega) = \sum_{k=1}^{n} \mathbb{1}(\omega \models \varphi_k)$$

$\mathbb{1}(\cdot)$ is an indicator function taking the value 1 when its argument is true and 0 otherwise.

Paths that have a higher score are clearly better because they satisfy more of the overall property. However in many applications the property of interest

may not have a suitable notion of levels to exploit; the logical levels may be too coarse or may distribute the probability unevenly. For example, given the dining philosophers problem presented in section 5, we know that from a thinking state, a philosopher must pick one fork and then a second one before eating, but there is no obvious way of creating a finer score function from these logical subproperties and actually, the probability of satisfying a subproperty from a state such that the previous subproperty is satisfied is too low (about 0.06, see Table 1). For these cases it is necessary to design a more general score function which maps a larger sequence of nested set of paths to a set of nested intervals of $\mathbb{R}$.

Denoting an arbitrary path by ω and two path prefixes by ω' and ω'', an ideal score function S satisfies the following property:

$$S(\omega') \geq S(\omega'') \iff \mathrm{P}(\omega \models \varphi \mid \omega') \geq \mathrm{P}(\omega \models \varphi \mid \omega'') \tag{3}$$

Intuitively, (3) states that prefix ω' has greater score than prefix ω'' if and only if the probability of satisfying φ with paths having prefix ω' is greater than the probability of satisfying φ with paths having prefix ω''.

Designing a score function which satisfies (3) is generally infeasible because it requires a huge analytical work based on a detailed knowledge of the system and, in particular, of the probability of interest. However, the minimum requirement of a score function is much less demanding. Given a set of nested properties $\varphi_1, \ldots \varphi_i, \ldots, \varphi_n$ satisfying (2), the requirement of a score function is that $\omega \models \varphi_i \iff S(\omega) \geq \tau_i$, with $\tau_i > \tau_{i-1}$ a monotonically increasing set of numbers called thresholds. Even a simple score function with few levels (e.g., $n = 2$) could nevertheless provide an unbiased estimate with a likely smaller number of traces than a standard Monte Carlo estimation.

When no formal levels are available, an effective score function may still be defined using heuristics that only loosely correlate increasing score with increasing probability of satisfying the property. In particular, a score function based on coarse logical levels may be given increased granularity by using heuristics between the levels. For example, a time bounded property, not explicitly correlated to time, may become increasingly less likely to be satisfied as time runs out (i.e., with increasing path length). A plausible heuristic in this case is to assign higher scores to shorter paths. A similar heuristic has been used for importance sampling, under the assumption that the mass of probability in the optimal change of measure is centred on short, direct paths [19]. In the context of importance splitting, the assumption is that shorter paths that satisfy the sub-property at one level are more likely to satisfy the sub-property at the next level because they have more time to do so. We make use of this heuristic in Section 4.

4 Importance splitting algorithms

We give three importance splitting pseudo-algorithms based on [4]; in the first one, levels are fixed and defined a priori, the number of levels is an input of the algorithm; in the second one, levels are found adaptively with respect to a

predefined probability, the number of levels is a random variable and is not an input anymore; the third one is an extension of the second where the probability to cross a level from a previous stage is set to its maximum. By N we denote the number of simulations performed at each level. Thresholds, denoted τ, are usually but not necessarily defined as values of score function $S(\omega)$, where ω is a path. τ_φ is the minimal threshold such that $S(\omega) \geq \tau_\varphi \iff \omega \models \varphi$. τ_k is the k^{th} threshold and ω_i^k is the i^{th} simulation on level k. $\tilde{\gamma}_k$ is the estimate of γ_k, the k^{th} conditional probability $\mathrm{P}(S(\omega) \geq \tau_k \mid S(\omega) \geq \tau_{k-1})$.

4.1 Fixed level algorithm

The fixed level algorithm follows from the general description given in Section 3. Its advantages are that it is simple, it has low computational overhead and the resulting estimate is unbiased. Its disadvantage is that the levels must often be guessed by trial and error – adding to the overall computational cost.

In Algorithm 1, $\tilde{\gamma}$ is an unbiased estimate (see, e.g., [8]). Furthermore, from Proposition 3 in [4], we can deduce the following $(1-\alpha)$ confidence interval:

$$CI = \left[\tilde{\gamma}\left(\frac{1}{1+\frac{z_\alpha \sigma}{\sqrt{N}}}\right), \tilde{\gamma}\left(\frac{1}{1-\frac{z_\alpha \sigma}{\sqrt{N}}}\right)\right] \qquad \text{with} \qquad \sigma^2 \geq \sum_{k=1}^{M} \frac{1-\gamma_k}{\gamma_k}, \quad (4)$$

where z_α is the $1-\frac{\alpha}{2}$ quantile of the standard normal distribution. Hence, with confidence $100(1-\alpha)\%$, $\gamma \in CI$. For any fixed M, the minimisation problem

$$\min \sum_{k=1}^{M} \frac{1-\gamma_k}{\gamma_k} \qquad \text{with constraint} \qquad \prod_{k=1}^{M} \gamma_k = \gamma$$

implies that σ is reduced by making all γ_k equal.

For given γ, this motivates fine grained score functions. When it is not possible to define γ_k arbitrarily, the confidence interval may nevertheless be reduced by increasing N. The inequality for σ arises because the independence of initial states diminishes with increasing levels: unsuccessful traces are discarded and new initial states are drawn from successful traces. Several possible ways to minimise these dependence effects are proposed in [4]. In the following, for the sake of simplicity, we assume that this goal is achieved. In the confidence interval, σ is estimated by the square root of $\sum_{k=1}^{M} \frac{1-\tilde{\gamma}_k}{\tilde{\gamma}_k}$.

4.2 Adaptive level algorithm

The cost of finding good levels must be included in the overall computational cost of importance splitting. An alternative to trial and error is to use an adaptive level algorithm that discovers its own optimal levels.

Algorithm 2 is an adaptive level importance splitting algorithm presented first in [5]. It works by pre-defining a fixed number N_k of simulation traces to retain at each level. With the exception of the last level, the conditional

Algorithm 1: Fixed levels

Let $(\tau_k)_{1 \leq k \leq M}$ be the sequence of thresholds with $\tau_M = \tau_\varphi$
Let *stop* be a termination condition
$\forall j \in \{1, \ldots, N\}$, set prefix $\tilde{\omega}_j^1 = \epsilon$ (empty path)
for $1 \leq k \leq M$ **do**
  $\forall j \in \{1, \ldots, N\}$, using prefix $\tilde{\omega}_j^k$, generate path ω_j^k until $(S(\omega_j^k) \geq \tau_k) \vee stop$
  $I_k = \{\forall j \in \{1, \ldots, N\} : S(\omega_j^k) \geq \tau_k\}$
  $\tilde{\gamma}_k = \frac{|I_k|}{N}$
  $\forall j \in I_k$, $\tilde{\omega}_j^{k+1} = \omega_j^k$
  $\forall j \notin I_k$, let $\tilde{\omega}_j^{k+1}$ be a copy of ω_i^k with $i \in I_k$ chosen uniformly randomly
$\tilde{\gamma} = \prod_{k=1}^{M} \tilde{\gamma}_k$

Algorithm 2: Adaptive levels

Let $\tau_\varphi = \min \{S(\omega) \mid \omega \models \varphi\}$ be the minimum score of paths that satisfy φ
Let N_k be the pre-defined number of paths to keep per iteration
$k = 1$
$\forall j \in \{1, \ldots, N\}$, generate path ω_j^k
repeat
  Let $T = \left\{S(\omega_j^k), \forall j \in \{1, \ldots, N\}\right\}$
  Find maximum $\tau_k \in T$ such that $|\{\tau \in T : \tau > \tau_k\}| \geq N - N_k$
  $\tau_k = \min(\tau_k, \tau_\varphi)$
  $I_k = \{j \in \{1, \ldots, N\} : S(\omega_j^k) > \tau_k\}$
  $\tilde{\gamma}_k = \frac{|I_k|}{N}$
  $\forall j \in I_k$, $\omega_j^{k+1} = \omega_j^k$
  for $j \notin I_k$ **do**
    choose uniformly randomly $l \in I_k$
    $\tilde{\omega}_j^{k+1} = \max_{|\omega|} \left\{\omega \in pref(\omega_l^k) : S(\omega) < \tau_k\right\}$
    generate path ω_j^{k+1} with prefix $\tilde{\omega}_j^{k+1}$
  $M = k$
  $k = k + 1$
until $\tau_k > \tau_\varphi$;
$\tilde{\gamma} = \prod_{k=1}^{M} \tilde{\gamma}_k$

probability of each level is then nominally N_k/N. Making N_k all equal minimizes the overall relative variance and is only possible if the score function has sufficient granularity.

Use of the adaptive algorithm may lead to gains in efficiency (no trial and error, reduced overall variance), however the final estimate has a bias of order $\frac{1}{N}$, i.e., $\mathrm{E}(\tilde{\gamma}) = \gamma\left(1 + \mathcal{O}(N^{-1})\right)$. The overestimation (potentially not a problem when estimating rare critical failures) is negligible with respect to σ, such that the confidence interval remains that of the fixed level algorithm. Furthermore, under some regularity conditions, the bias can be asymptotically corrected. The estimate of γ has the form $r_0{\gamma_0}^{M_0}$, with $M_0 = M-1$, $r_0 = \gamma{\gamma_0}^{-M_0}$ and $\frac{E[\tilde{\gamma}]-\gamma}{\gamma} \sim \frac{M_0}{N}\frac{1-\gamma_0}{\gamma_0}$ when N goes to infinity. Using the expansion

$$\tilde{\gamma} = \gamma\left(1 + \frac{1}{\sqrt{N}}\sqrt{M_0\frac{1-\gamma_0}{\gamma_0} + \frac{1-r_0}{r_0}}Z + \frac{1}{N}M_0\frac{1-\gamma_0}{\gamma_0} + o\left(\frac{1}{N}\right)\right),$$

with Z a standard normal variable, $\tilde{\gamma}$ is corrected by dividing it by $1 + \frac{M_0(1-\gamma_0)}{N\gamma_0}$. See [4] for more details.

4.3 Optimized adaptive level algorithm

Algorithm 3 defines an optimized adaptive level importance splitting algorithm. The variance of the estimate $\tilde{\gamma}$ is:

$$Var(\tilde{\gamma}) = \frac{p^2}{N}\left(n_0\frac{1-\gamma_0}{\gamma_0} + \frac{1-r_0}{r_0} + o(N^{-1})\right)$$

and the function $f : \gamma_0 \longmapsto \frac{1-\gamma_0}{-\gamma_0\, log\, \gamma_0}$ is strictly decreasing on $]0,1[$. Increasing γ_0 therefore decreases the variance. Ideally, this value is $\gamma_0 = 1 - \frac{1}{N}$ but it is more realistic to fix this value for each iteration k at $\gamma_0 = 1 - \frac{N_k}{N}$, with N_k the number of paths achieving the minimal score. Another advantage of this optimized version is that, although the number of steps before the algorithm terminates is more important, we only rebranch a few discarded traces (ideally only 1) per iteration.

Remark about rebranching At the end of iteration k, we end up with an estimate of γ_k and an approximation $\tilde{l}_k$ of the first entrance state distribution into level k. The discarded traces must be rebranched over a successful prefix with respect to distribution $\tilde{l}_k$. In practise, to decrease the variance, we do not pick uniformly an index of a successful path but a cycle of indexes of successful paths. In doing so we avoid the unlikely but possible rebranching of all the discarded traces from the same state.

Let I_k and J_k be respectively the sets of indexes of successful and discarded prefixes. We denote by respectively $I_k(j)$ and $J_k(j)$ the j-th index of I_k and J_k. Let $\mathfrak{S}_{|I_k|}$ be the set of permutations of $\{1, \cdots, |I_k|\}$ and ι an element of $\mathfrak{S}_{|I_k|}$.

Algorithm 3: Optimized adaptive levels

Let $\tau_\varphi = \min\{S(\omega) \mid \omega \models \varphi\}$ be the minimum score of paths that satisfy φ
$k = 1$
$\forall j \in \{1, \ldots, N\}$, generate path ω_j^k
repeat
  Let $T = \left\{S(\omega_j^k), \forall j \in \{1, \ldots, N\}\right\}$
  $\tau_k = \min T$
  $\tau_k = \min(\tau_k, \tau_\varphi)$
  $I_k = \{j \in \{1, \ldots, N\} : S(\omega_j^k) > \tau_k\}$
  $\tilde{\gamma}_k = \frac{|I_k|}{N}$
  $\forall j \in I_k$, $\omega_j^{k+1} = \omega_j^k$
  for $j \notin I_k$ **do**
    choose uniformly randomly $l \in I_k$
    $\tilde{\omega}_j^{k+1} = \max_{|\omega|} \left\{\omega \in \mathit{pref}(\omega_l^k) : S(\omega) < \tau_k\right\}$
    generate path ω_j^{k+1} with prefix $\tilde{\omega}_j^{k+1}$
  $M = k$
  $k = k + 1$
until $\tau_k > \tau_\varphi$;
$\tilde{\gamma} = \prod_{k=1}^{M} \tilde{\gamma}_k$

We then build randomly a $|J_k|$-length vector $\tilde{J}_k$ with elements of I_k. We choose uniformly cycle ι of $\mathfrak{S}_{|I_k|}$ and repeat the chosen cycle if $N - |I_k| \geq |I_k|$. The first $|J_k|$ elements are the respective elements of $\tilde{J}_k$. Finally, we assign to discarded prefix $\omega_{J_k(j)}$ the successful prefix $\omega_{\tilde{J}_k(j)} = \omega_{I_k((\iota(j)-1 \text{ modulo } |I_k|)+1)}$.

This circular sampling has the advantage to resample perfectly with respect to distribution $\tilde{l}_k$.

5 Case study: dining philosophers protocol

We have adapted a case study from the literature to illustrate the use of heuristic-based score functions and of the optimized adaptive splitting algorithm with statistical model checking. We have defined a rare event in the well known probabilistic solution [17] of Dijkstra's dining philosophers problem . In this example, there are no natural counters to exploit, so levels must be constructed by considering 'lumped' states.

A number of philosophers sit at a circular table with an equal number of chopsticks; a chopstick being placed within reach of two adjacent philosophers. Philosophers think and occasionally wish to eat from a communal bowl. To eat, a philosopher must independently pick up two chopsticks: one from the left and one from the right. Having eaten, the philosopher replaces the chopsticks and returns to thinking. A problem of concurrency arises because a philosopher's neighbour(s) may have already taken the chopstick(s). Lehmann and Rabin's solution [17] is to allow the philosophers to make probabilistic choices.

We consider a model of 150 'free' philosophers [17]. The number of states in the model is more than 10^{177}; 10^{97} times more than the estimated number of protons in the universe. The possible states of an individual philosopher can be abstracted to those shown in Fig. 1.

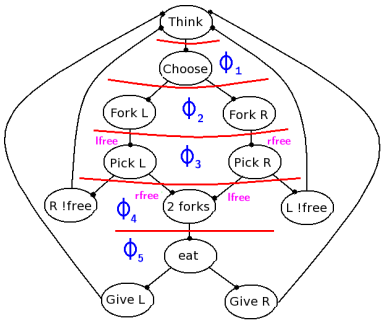

Figure 1. Abstract dining philosopher.

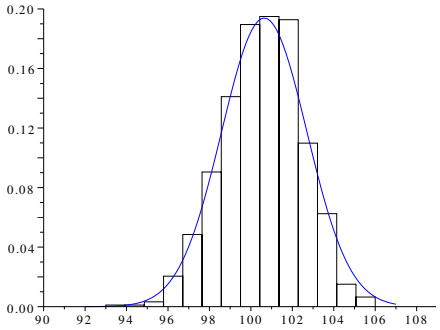

Figure 2. Empirical number of levels.

Think is the initial state of all philosophers. In state Choose, the philosopher makes a choice of fork he will try to get first. The transitions labelled by lfree or rfree in Fig. 1 are dependent on the availability of respectively left or right chopsticks. All transitions are controlled by stochastic rates and made in competition with the transitions of other philosophers. With increasing numbers of philosophers, it is increasingly unlikely that a specific philosopher will be satisfied (i.e., that the philosopher will reach the state eat) within a given number of steps from the initial state. We thus define a rare property $\varphi = \mathbf{F}^t\mathsf{eat}$, with t initially 30, denoting the property that a given philosopher will reach state eat within 30 steps. Thus, using the states of the abstract model, we decompose φ into nested properties $\varphi_0 = \mathbf{F}^t\mathsf{Think} = \top$, $\varphi_1 = \mathbf{F}^t\mathsf{Choose}$, $\varphi_2 = \mathbf{F}^t\mathsf{Try}$, $\varphi_3 = \mathbf{F}^t 1^{\mathsf{st}}\mathsf{stick}$, $\varphi_4 = \mathbf{F}^t 2^{\mathsf{nd}}\mathsf{stick}$, $\varphi_5 = \mathbf{F}^t\mathsf{eat}$. The red lines crossing the transitions indicate these formal levels on the graph.

Monte Carlo simulations with PLASMA statistical model checker With such a large state space it is not possible to obtain a reference result with numerical model checking. We therefore performed extensive Monte Carlo simulations using the parallel computing capability of the PLASMA statistical model checker [11,3]. The experiment generated 300 million samples using 255 cores and took about 50 minutes. Our reference probability is thus approximately equal to 1.59×10^{-6} with 95%-confidence interval $\left[1.44 \times 10^{-6}; 1.72 \times 10^{-6}\right]$.

Recall and Experiment protocol Table 1 recalls, given a score function, that the parameters of each algorithm for an experiment are the number n of simulations used at the first iteration and the distance between levels (usually

constant) in the fixed level algorithm or a probability between levels for the adaptive algorithms.

initial parameters n, $\tau_k - \tau_{k-1}$, γ_0	fixed alg.	adaptive alg.	optimized alg
Number n of path at first iteration	YES	YES	YES
Step between levels ($\tau_k - \tau_{k-1}$)	YES	NO	NO
conditionnal probability γ_0	NO	YES	NO

Table 1. Parameters in each ISp algorithm.

Note that the conditionnal probability in the optimized algorithm is a function of n more than an independent parameter.

Four types of importance splitting experiments are driven. The first one uses the simple score function and the fixed algorithm, the second uses the heuristic score function and the fixed-level algorithm (with different step values). The third algorithm uses the adaptive-level algorithm with different γ_0 parameters and finally the fourth set of experiments uses the optimized version of the adaptive algorithm.

For each set of experiments and chosen parameters, experiments are repeated 100 times in order to check the reliability of our results. In what follows, we remind which statistical notions are exploited and why:

- Number of experiments: used to estimate the variance of the estimator.
- Number of path per iteration: it is a parameter of the algorithm, equal to the number of paths that we use to estimate a conditionnal probability.
- Number of levels: known in the fixed algorithm, variable in the adaptive algorithms. In the second case, an average is provided.
- Time in seconds: the average of the 100 experiments is provided.
- The mean estimate is the estimator $\tilde{\gamma}$ of the probability of interest. The average of the 100 estimators is provided.
- The relative standard deviation of $\tilde{\gamma}$ is estimated with the 100 final estimators γ. A reliable estimator must have a low relative standard deviation (roughly ≤ 0.3).
- The mean value of γ_k is the average of the mean values of the conditionnal probabilities in an experiment. It is variable in the fixed algorithm and supposed to be a constant γ_0 in the adaptive algorithms. Because of the discreteness of the score function, the value is only almost constant and slightly lower than γ_0.
- The relative standard deviation of γ_k is the average of the relative standard deviations of the conditionnal probabilities in an experiment. By construction, the value in the adaptive algorithms must be low.

Comparison between logical and heuristic score function Let ω be a path of length $t = 30$. For each prefix $\omega_{\leq j}$ of length j, we define the following

function:

$$\Psi(\omega_{\leq j}) = \sum_{k=0}^{n} \mathbb{1}(\omega_{\leq j} \models \varphi_k) - \frac{\{\sum_{k=1}^{n} \mathbb{1}(\omega_{\leq j} \models \varphi_k)\} - j}{\sum_{k=1}^{n} \mathbb{1}(\omega_{\leq j} \models \varphi_k) - (t+1)}$$

We define score of ω as follows:

$$S(\omega) = \max_{1 \leq j \leq K} \Psi(\omega_{\leq j})$$

In the following experiment this score function is defined for any path of length $t+1$, starting in the initial state 'all philosophers think'. The second term of Ψ is a number between 0 and 1, linear in j such that the function gives a greater score to paths which satisfy a greater number of sub-properties φ_k and discriminates between two paths satisfying the same number of sub-properties by giving a greater score to the shortest path. A score in $]i-1, i]$ implies that a prefix of the path satisfied at most φ_i. We then compare results with the simple score function $S(\omega) = \sum_{k=1}^{n} \mathbb{1}(\omega \models \varphi_k)$.

Statistics	Simple score function	Heuristic score function		
number of experiments	100	100	100	100
number of path per iteration	1000	1000	1000	1000
number of levels	5	20	40	80
Time in seconds (average)	6.95	13.42	16.64	21.56
mean estimate $\times 10^6$ (average)	0.01	0.59	1	1.37
mean value of $\tilde{\gamma_k}$	0.06	0.53	0.73	0.86
relative standard deviation of $\tilde{\gamma_k}$	1.04	0.36	0.22	0.15

Table 2. Comparison between fixed-level algorithms.

The experiments are repeated 100 times in order to demonstrate and improve the reliability of the results. Each conditional probability γ_k is estimated with a sample of 1000 paths.

For simplicity we consider a linear growing of score thresholds when we use the fixed-level algorithm. The simple score function thresholds increase by 1 between each level. When using the heuristic score function, we performed three sets of experiments involving an increase of 0.2, 0.1 and 0.05 of the thresholds. These partitions imply respectively 5, 20, 40 and 80 levels.

Table 2 shows that the simple score function likely gives a strong underestimation. It is due to the huge decrease of value of conditional probabilities between the logical levels. All the estimated conditional probabilities are small and imply a large theoretical relative variance ($V(\tilde{\gamma})/E[\tilde{\gamma}]$). The final levels are difficult to cross and have probabilities close to 0. A sample size of 1000 paths is obviously not enough for the last step. On average $\tilde{\gamma}_5 = 0.003$ and in one case the last step is not satisfied by any trace, such that the estimate is equal to zero.

If a threshold is not often exceeded, it implies that traces will be rebranched from a very small set of first entrance states at the next level. This leads to significant relative variance between experiments. A further problem is that the conditional estimate is less efficient if γ_k is small. Increasing the number of evenly spaced levels decrease *a priori* more smoothly the conditional probabilities and reinforce the reliability of the results as soon as the relative standard deviation of conditional probabilities decreases enough. In the experiments, as expected, the mean value of conditional probabilities is positively correlated to the number of levels (respectively 0.06, 0.53, 0.73 and 0.86) and negatively correlated to the relative standard deviation of conditional probabilities. The results with 40 and 80 levels give results that are apparently close to the reference estimate, but are nevertheless consistently underestimates. This suggests that the number of simulations per level is too low.

Two questions arise: how to detect that the simulation is not efficient or robust and how to improve the results. In answer to the first, there are no general criteria for judging the quality of an importance splitting estimator. However, assuming that experiments are repeated a few times, a large relative error of the estimators (roughly ≥ 0.5), a very low value of conditional probability estimates, or a large relative error of conditional probability estimates (roughly ≥ 0.2) are good warnings. As for the second question, a way to improve results with the fixed level algorithm is simply to increase the number of paths per level or to increase the number of levels, for the reasons given above.

5.1 comparison between fixed and adaptive algorithm

The following section illustrates that adaptive algorithms give significantly more reliable results for slightly increased time. In the following set of experiments we use the adaptive algorithm with three predefined γ_0: 0.6, 0.75 and 0.9. Because of the granularity of the score function, conditional probabilities are not equal at each iteration, but their values are kept under control because their relative standard deviation does not vanish (≤ 0.2). We use 1000 sample paths per level and repeat the experiments 100 times.

γ_0	0.6	0.75	0.9
number of experiments	100	100	100
number of path per iteration	1000	1000	1000
number of levels (average)	22	34	65
Time in seconds (average)	14.53	16.78	20.05
mean estimate $\times 10^6$ (average)	0.78	1.14	1.58
relative standard deviation of $\tilde{\gamma}$	0.26	0.25	0.23
mean value of $\tilde{\gamma_k}$	0.55	0.68	0.83
relative standard deviation of $\tilde{\gamma_k}$	0.2	0.16	0.12

Table 3. Comparison between adaptive algorithms.

As we increased the desired γ_0, the number of levels and time increase. However, the final estimate with $\gamma_0 = 0.9$ matches the Monte Carlo estimator and the relative standard deviation is minimized. In this experiment the number of levels found adaptively is on average 65. Even with mean value of conditional probabilities smaller than in the 80-fixed-level experiment, the results show better convergence, a slightly better speed and lower standard deviation.

5.2 Comparison with the optimized adaptive algorithm

Statistics	Importance splitting				MC
number of experiments	100	100	100	100	1
number of path per iteration	100	200	500	1000	10 million
Time in seconds (average)	1.73	4.08	11.64	23.77	> 5 hours
mean estimate $\times 10^6$ (average)	1.52	1.59	1.58	1.65	1.5
standard deviation $\times 10^6$	1.02	0.87	0.5	0.38	0.39
95%-confidence interval $\times 10^6$	[1.34; 1.74]	[1.48; 1.72]	[1.54; 1.63]	[1.64; 1.66]	[0.74; 2.26]

Table 4. Comparison between optimized adaptive algorithms.

This section illustrates a set of experiments using the optimized adaptive algorithm. As previously, we repeated experiments 100 times to check reliability of our results. For each experiment we use a different number of initial paths: 100, 200, 500 and 1000. In order to give an idea of the gain of time, we also executed a Monte Carlo experiment using 10^7 paths. The 95%-confidence intervals are given by (4) for the importance splitting experiments and by the standard confidence interval $\left[\tilde{\gamma} \pm 1.96 \times \sqrt{\frac{\tilde{\gamma}(1-\tilde{\gamma})}{N}}\right]$ for Monte Carlo experiment. As the experiments are repeated several times, we approximate the relative standard deviation σ by the standard deviation of the estimates divided by the average of the estimates, instead of assuming full independence between levels and so taking $\sigma \approx \sum_{k=1}^{m} \frac{1-\gamma_k}{\gamma_k}$. Our approach is more pessimistic and in practise requires the experiment to be repeated a few times. However, even doing so, the results are much more accurate than the Monte Carlo approach. For example, 100 initial paths are used in the first experiment. Roughly speaking, the paths cross on average 100 other levels and only 11% are rebranched each time. So, only 1200 paths are generated and provide in less than 2 seconds an estimate and a confidence interval strictly included in the Monte Carlo confidence interval. This represents a gain greater than 10^4 with respect to the Monte Carlo experiment.

Figure 2 illustrates empirically the convergence of the number of levels to a Gaussian with low variance (4.23) with respect to the mean of levels (100.65). Although this fact is only empirical, knowing that the variance is low has some importance whenever the time budget is critical for more extensive experiments.

6 Conclusion

We have presented an effective heuristic to improve the granularity of score functions for importance splitting. The logical properties used in statistical model checking can thus be decomposed into a greater number of levels, allowing the use of high performance adaptive algorithms. We have presented an optimized adaptive algorithm and shown how, in combination with our heuristic score function, it significantly improves on the performance of the alternatives. As future work, we would like to develop a Chernoff bound and sequential hypothesis test to complement the confidence interval presented here.

References

1. C. Baier and J.-P. Katoen. *Principles of Model Checking (Representation and Mind Series)*. The MIT Press, 2008.
2. 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 *LNCS*, pages 331–346. Springer, 2012.
3. B. Boyer, K. Corre, A. Legay, and S. Sedwards. PLASMA-lab: A flexible, distributable statistical model checking library. In K. Joshi, M. Siegle, M. Stoelinga, and P. D'Argenio, editors, *Quantitative Evaluation of Systems*, volume 8054 of *Lecture Notes in Computer Science*, pages 160–164. Springer Berlin Heidelberg, 2013.
4. F. Cérou, P. Del Moral, T. Furon, and A. Guyader. Sequential Monte Carlo for rare event estimation. *Statistics and Computing*, 22:795–808, 2012.
5. F. Cérou and A. Guyader. Adaptive multilevel splitting for rare event analysis. *STOCHASTIC ANALYSIS AND APPLICATIONS*, 25:417–443, 2007.
6. E. Clarke, E. A. Emerson, and J. Sifakis. Model checking: algorithmic verification and debugging. *Commun. ACM*, 52(11):74–84, Nov. 2009.
7. E. M. Clarke, Jr., O. Grumberg, and D. A. Peled. *Model checking*. MIT Press, Cambridge, MA, USA, 1999.
8. P. Del Moral. *Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications*. Probability and Its Applications. Springer, 2004.
9. D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81:2340–2361, 1977.
10. P. Glasserman, P. Heidelberger, P. Shahabuddin, and T. Zajic. Multilevel splitting for estimating rare event probabilities. *Oper. Res.*, 47(4):585–600, Apr. 1999.
11. C. Jegourel, A. Legay, and S. Sedwards. A Platform for High Performance Statistical Model Checking – PLASMA. In C. Flanagan and B. König, editors, *TACAS*, volume 7214 of *LNCS*, pages 498–503. Springer, 2012.
12. C. Jegourel, A. Legay, and S. Sedwards. Cross-Entropy Optimisation of Importance Sampling Parameters for Statistical Model Checking. In P. Madhusudan and S. A. Seshia, editors, *CAV*, volume 7358 of *LNCS*, pages 327–342. Springer, 2012.
13. C. Jégourel, A. Legay, and S. Sedwards. Importance splitting for statistical model checking rare properties. In *CAV*, pages 576–591, 2013.
14. H. Kahn. Random sampling (Monte Carlo) techniques in neutron attenuation problems. *Nucleonics*, 6(5):27, 1950.

15. H. Kahn and T. E. Harris. Estimation of Particle Transmission by Random Sampling. In *Applied Mathematics*, volume 5 of *series 12*. National Bureau of Standards, 1951.
16. H. Kahn and A. W. Marshall. Methods of Reducing Sample Size in Monte Carlo Computations. *Operations Research*, 1(5):263–278, November 1953.
17. D. Lehmann and M. O. Rabin. On the Advantage of Free Choice: A Symmetric and Fully Distributed Solution to the Dining Philosophers Problem (Extended Abstract). In *Proc. 8th Ann. Symposium on Principles of Programming Languages*, pages 133–138, 1981.
18. M. Okamoto. Some inequalities relating to the partial sum of binomial probabilities. *Annals of the Institute of Statistical Mathematics*, 10:29–35, 1959.
19. D. Reijsbergen, P.-T. de Boer, W. Scheinhardt, and B. Haverkort. Rare event simulation for highly dependable systems with fast repairs. *Performance Evaluation*, 69(7–8):336 – 355, 2012.
20. M. N. Rosenbluth and A. W. Rosenbluth. Monte Carlo Calculation of the Average Extension of Molecular Chains. *Journal of Chemical Physics*, 23(2), February 1955.
21. M. Villén-Altamirano and J. Villén-Altamirano. RESTART: A Method for Accelerating Rare Event Simulations. In J. W. Cohen and C. D. Pack, editors, *Queueing, Performance and Control in ATM*, pages 71–76. Elsevier, 1991.
22. A. Wald. Sequential Tests of Statistical Hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186, June 1945.