Statistical Model Checking for Cyber-Physical Systems*

Edmund M. Clarke and Paolo Zuliani

Computer Science Department
Carnegie Mellon University
Pittsburgh, PA, USA
{emc,pzuliani}@cs.cmu.edu

Abstract. Statistical Model Checking is useful in situations where it is either inconvenient or impossible to build a concise representation of the global transition relation. This happens frequently with cyber-physical systems: Two examples are verifying Stateflow-Simulink models and in reasoning about biochemical reactions in Systems Biology. The main problem with Statistical Model Checking is caused by rare events. We describe how Statistical Model Checking works and demonstrate the problem with rare events. We then describe how Importance Sampling with the Cross-Entropy Technique can be used to address this problem.

1 Introduction

Cyber-Physical Systems are characterized by the tight interaction between a digital computing component (the Cyber part) and a continuous-time dynamical system (the Physical part). The concept is better explained by examples. A modern airliner governed by the autopilot is a typical Cyber-Physical System (CPS). The autopilot is a software which provides inputs to the aircraft's engines and flight control surfaces (*e.g.*, rudder, flaps, *etc.*) on the basis of various sensor readings and an appropriate control law. The autopilot greatly reduces the pilot's workload and can improve the aircraft's fuel economy. Another example of CPS is a car equipped with an Anti-lock Braking System. The ABS modulates braking power to avoid a complete lock-up of the car's wheels in hard braking or low adherence situations. In this way, the friction between the tires and the road surface is maintained, thereby allowing the driver to keep control of the vehicle and improving safety.

Cyber-Physical Systems enjoy wide adoption in our society, even in safety-critical applications, but are difficult to reason about. In particular, to automatically prove behavioral properties of a CPS is exceedingly difficult. One of the

* This research was sponsored by the National Science Foundation under contracts no. CNS0926181 and no. CNS0931985, the SRC under contract no. 2005TJ1366, General Motors under contract no. GMCMUCRLNV301, the Air Force (Vanderbilt University) under contract no. 18727S3, the GSRC under contract no. 1041377 (Princeton University), the Office of Naval Research under award no. N000141010188, and DARPA under contract FA8650-10-C-7077.

obstacles is due to the fact that currently we do not know how to interface formal verification techniques for the cyber part with the well-established engineering techniques used to design the physical part of the system [12]. Another obstacle is that most CPSs feature stochastic effects, because of uncertainties present in the system components or the environment. For example, a flight control system needs to be able to cope with (possibly) unreliable readings from sensors, or to recognize and react appropriately when hit by "random" cosmic radiation at high altitudes. As a result, fully formal verification of a CPS is currently not possible, while validation boils down to extensive system simulations and bench/live tests. However, in the past decade there has been progress towards formal verification for CPSs.

In this paper we single out one particular verification technique that aims at tackling both obstacles above: Statistical Model Checking [22,21,16,5]. This technique addresses the verification problem for general stochastic systems, *i.e.*, to compute the probability that a stochastic model satisfies a given temporal logic property. For example, we would like to know the probability of a fuel-control system failing to ensure an optimal air-fuel flow ratio, given unreliable readings from the engine's sensors. We express such properties in Bounded Linear Temporal Logic (BLTL), a variant of LTL [13] in which the temporal operators are equipped with time bounds. As CPS models, we use a stochastic version of control systems modeled in Stateflow/Simulink - the *de facto* standard tool for embedded system design.

Numerical methods [1,2,3,4,7] have been developed to compute with high precision the probability that a stochastic system satisfies a temporal logic formula, but they are generally only feasible for systems with up to 10^8-10^9 states [10,18]. The state space of modern CPSs very often exceeds this limit (or is infinite), hence the need for methods such as Statistical Model Checking, which solve the verification problem for stochastic systems in a less precise, yet rigorous and more efficient way.

Statistical model checking addresses the verification problem as a statistical inference problem: it samples behaviors (simulations) of the system model, checks their conformance with respect to the temporal formula, and finally applies a *statistical estimation* technique to compute an approximate value for the probability that the formula is satisfied. The returned value will be, with high probability, close to the true probability that the formula holds. The key observation behind statistical model checking's efficiency is that for large, complex systems, simulation is generally easier and faster than building a concise representation of the global transition relation of the system.

Statistical model checking was introduced by Younes [20], and phrased as a hypothesis testing problem. In that setting, the task is to decide whether the temporal formula is satisfied with a probability greater than a given threshold. Later work [6,16] generalized statistical model checking using statistical estimation techniques (*e.g.*, the Chernoff bound). Hypothesis-testing methods are more efficient than estimation techniques when the probability that the formula holds is distant from the user-specified threshold [19]. Sequential Bayesian techniques for both hypothesis testing and estimation were introduced in [8,23] and shown to perform very well.

The main problem with statistical model checking is caused by rare events, *i.e.*, temporal formulae whose satisfaction probability is very small. When estimating the probability of such formulae, the number of simulations needed to ensure a good estimate becomes unfeasible. In this paper we show that Importance Sampling and the Cross-Entropy method can efficiently address this problem.

2 Background

Statistical model checking is essentially a Monte Carlo technique, since it is based on randomized sampling of simulations of a stochastic model. In this Section, we first describe the temporal logic used to express properties and how statistical model checking works. Next, we give a summary of the Monte Carlo method and the rare-event problem.

2.1 Statistical Model Checking

We start by defining the Bounded Linear Temporal Logic (BLTL) [11,8]. For a model $\mathcal{M}$, we denote by SV the finite set of real-valued state variables. An Atomic Proposition (AP) over SV is a Boolean predicate of the form $y{\sim}v$, where $y \in SV$, $\sim$ is one of $\{\geq, \leq, =\}$, and $v \in \mathbb{R}$. A BLTL property is built on a finite set of Boolean predicates over SV using Boolean connectives and temporal operators. The syntax of the logic is given by the following grammar:

$$\phi ::= y{\sim}v \,|\, (\phi_1 \vee \phi_2) \,|\, (\phi_1 \wedge \phi_2) \,|\, \neg\phi_1 \,|\, (\phi_1 \mathbf{U}^t \phi_2),$$

where $\sim \in \{\geq, \leq, =\}$, $y \in SV$, $v \in \mathbb{Q}$, and $t \in \mathbb{Q}_{\geq 0}$.

The formula $\phi_1 \mathbf{U}^t \phi_2$ holds true if and only if, *within* time t, ϕ_2 will be true and ϕ_1 will hold until then. Bounded versions of the usual $\mathbf{F}$ and $\mathbf{G}$ operators are easily defined: $\mathbf{F}^t\phi = \mathit{true}\ \mathbf{U}^t\phi$ requires ϕ to hold true within time t; $\mathbf{G}^t\phi = \neg\mathbf{F}^t\neg\phi$ requires ϕ to hold true up to time t. Also, BLTL can be seen as a sublogic of Metric Temporal Logic [9].

The semantics of BLTL is defined with respect to executions (traces) of $\mathcal{M}$. A trace σ is a sequence $(s_0, t_0), (s_1, t_1), \ldots$, with the meaning that the system moved to state s_{i+1} after having sojourned for time t_i in state s_i. We assume non-Zeno behavior about $\mathcal{M}$, *i.e.*, for any trace σ it must be $\sum_{i=0}^{\infty} t_i = \infty$. In other words, the system cannot make an infinite number of transitions in a finite amount of time. This assumption is necessary for ensuring termination of statistical model checking.

The fact that a trace σ satisfies the BLTL property ϕ is denoted by $\sigma \models \phi$. We denote the trace suffix starting at step i by σ^i, where σ^0 denotes the full trace σ.

Definition 1. *The* semantics *of BLTL for a trace σ^k ($k \in \mathbb{N}$) is:*

- $\sigma^k \models AP$ iff AP holds true in state s_k;
- $\sigma^k \models \phi_1 \vee \phi_2$ iff $\sigma^k \models \phi_1$ or $\sigma^k \models \phi_2$;
- $\sigma^k \models \phi_1 \wedge \phi_2$ iff $\sigma^k \models \phi_1$ and $\sigma^k \models \phi_2$;
- $\sigma^k \models \neg\phi_1$ iff $\sigma^k \models \phi_1$ does not hold;
- $\sigma^k \models \phi_1 \mathbf{U}^t \phi_2$ iff $\exists i \geq 0$ such that

  a) $\sum_{l=0}^{i-1} t_{k+l} \leq t$, *and*

  b) $\sigma^{k+i} \models \phi_2$, *and*

  c) $\forall\ 0 \leq j < i$, $\sigma^{k+j} \models \phi_1$.

Statistical model checking is based on checking system simulations, *i.e.*, *finite* traces (naturally, simulations need to be finite in length). Therefore, one has to prove that $\sigma \models \phi$ has a well-defined semantics and will not change its truth-value by continuing the simulation. In [23] we proved well-definedness and the fact that a finite prefix of the trace is sufficient for BLTL model checking, which is crucial for termination.

Definition 2. *[11,23] The* sampling bound $\#(\phi) \in \mathbb{Q}_{\geq 0}$ *of a BLTL formula* ϕ *is defined as:*

$$\begin{aligned}
\#(y \sim v) &= 0\\
\#(\neg\phi_1) &= \#(\phi_1)\\
\#(\phi_1 \vee \phi_2) &= \max(\#(\phi_1), \#(\phi_2))\\
\#(\phi_1 \wedge \phi_2) &= \max(\#(\phi_1), \#(\phi_2))\\
\#(\phi_1 \mathbf{U}^t \phi_2) &= t + \max(\#(\phi_1), \#(\phi_2))
\end{aligned}$$

Since we assumed non-zenoness, any trace will reach the sampling bound with a finite prefix (not necessarily of the same length). We have the following lemma.

Lemma 1. *[23] For any BLTL formula* ϕ *and trace* σ, *the relation* $\sigma \models \phi$ *is well-defined and can be checked using only a* finite prefix *of* σ *of duration* $\#(\phi)$.

The verification problem for a stochastic system $\mathcal{M}$ and a BLTL formula ϕ is the following: to compute the probability that $\mathcal{M}$ satisfies ϕ. We are in particular interested in discrete-time stochastic systems, since statistical model checking is based on simulation. The problem is well-posed, as it can be shown that the set of traces of $\mathcal{M}$ satisfying ϕ is measurable, thereby defining the probability p that $\mathcal{M}$ satisfies ϕ [22].

Suppose now that the stochastic system $\mathcal{M}$ satisfies the BLTL formula ϕ with some (unknown) probability $p = \text{Prob}\{\sigma \mid \sigma \models \phi\}$. The key idea behind statistical model checking [22] is that the behavior of $\mathcal{M}$ (with respect to property ϕ) can be modeled by a Bernoulli random variable with success parameter p. This random variable can be repeatedly evaluated via system simulation in the following way. Let σ be a trace of $\mathcal{M}$, then one can define the Bernoulli random variable B that returns 1 if $\sigma \models \phi$ and 0 otherwise. In other words, the probability mass function of B is

$$\begin{aligned}
&\text{Prob}(B(\sigma) = 1) = p \qquad && (\sigma \models \phi) \qquad (1)\\
&\text{Prob}(B(\sigma) = 0) = 1 - p \qquad && (\sigma \models \neg\phi)
\end{aligned}$$

Therefore, by running a simulation of $\mathcal{M}$ and by checking ϕ on the resulting trace we can obtain a sample of B.

2.2 The Monte Carlo Method

We consider the problem of estimating the probability of rare events in a stochastic CPS by means of randomized (*i.e.*, Monte Carlo) techniques. An event is said to be *rare* when its probability of occurrence is very low, say 10^{-8}. The Monte Carlo approach for estimating probabilities is by means of relative frequencies. Let X be a random variable defined over a probability space $(\Omega, \mathcal{F}, \mathrm{P})$. Suppose we want to estimate $p = \mathrm{P}(X \in B)$, the probability that X belongs to a given Borel set B. We first obtain a number of independent realizations of $I_B(X)$, the indicator function of B — $I_B(x)$ is 1 if $x \in B$ ("$X \in B$ has occurred"), 0 otherwise — and then compute their average to estimate p.

The theoretical justification of the Monte Carlo method is the strong law of large numbers. It states that if $X_1, X_2, \ldots$ is a sequence of independent and identically distributed (iid) random variables with $\mathrm{E}[|X_1|] < \infty$, then

$$\mathrm{P}\left(\lim_{n\to\infty} \frac{S_n}{n} = \mu\right) = 1$$

where $S_n = X_1 + \cdots + X_n$ and $\mu = \mathrm{E}[X_1]$. This means that the measure of the set of sample points for which $\frac{S_n}{n}$ converges to μ is 1. Therefore, we can approximate μ by taking the average of a *finite* number of realizations (samples) of X_1, since we know that the average will not converge to μ only for a negligible subset of realizations (a set of measure 0).

Returning to our problem of estimating $\mathrm{P}(X \in B) = p$ for a given random variable X and Borel set B, note that the random variable $I_B(X)$ is a Bernoulli of success parameter p, that is, $\mathrm{P}(I_B(X) = 1) = p$. Also, note that $p = \mathrm{E}[I_B(X)]$. Now, given a finite sequence $X_1, \ldots, X_N$ of random variables iid as X, the *crude Monte Carlo estimator* $\hat{p} = \frac{1}{N}\sum_{i=1}^{N} I_B(X_i)$ will converge to p as $N \to \infty$ (with probability 1) by the strong law of large numbers. The estimator $\hat{p}$ is readily shown to be *unbiased* (*i.e.*, $\mathrm{E}[\hat{p}] = p$) and its variance is:

$$\mathrm{Var}(\hat{p}) = \frac{\mathrm{Var}(I_B(X))}{N} \ .$$

Also, from the central limit theorem it follows that for large N the distribution of $\hat{p}$ is approximately a normal distribution of mean $\hat{p}$ and variance $\mathrm{Var}(I_B(X))/N$. The variance of $\hat{p}$ will thus tends to 0 as we increase the sample size N, leading to more precise estimates. However, a small variance does not necessarily imply a good estimate.

The *relative error* associated with the estimate $\hat{p}$ is an important quantity for assessing the quality of an estimator, especially in the rare-event case ($p \ll 1$). It is defined as the ratio

$$\mathrm{RE}(\hat{p}) = \frac{\sqrt{\mathrm{Var}(\hat{p})}}{\mathrm{E}[\hat{p}]}$$

and intuitively it is a "measure" of the accuracy of the estimator $\hat{p}$ with respect to its standard deviation. Since the crude Monte Carlo estimator is unbiased, the sample $X_1, \ldots, X_N$ is iid, and $p \ll 1$, it follows that

$$\mathrm{RE}(\hat{p}) = \frac{\sqrt{\mathrm{Var}(I_B(X))/N}}{p} = \frac{\sqrt{p(1-p)}}{p\sqrt{N}} \approx \sqrt{\frac{1}{Np}}\ .$$

Now, if N is kept constant and $p \to 0$, it follows that $\mathrm{RE}(\hat{p}) \to \infty$. For example, to estimate $p = 10^{-8}$ with a relative error of 0.01 we would need about $N \approx \frac{1}{p\mathrm{RE}^2(\hat{p})} = 10^{12}$ samples — an unfeasible quantity. Therefore, in order to keep the relative error low as $X \in B$ becomes rarer, we need to increase the sample size, thereby meaning that crude Monte Carlo is not an efficient technique for estimating very low probabilities. Alternatively, one can try to find another estimator whose variance is smaller than $\mathrm{Var}(\hat{p})$, for a given sample size. Importance sampling is a technique for devising estimators with reduced variance, and thus with low relative error.

3 Importance Sampling

Importance Sampling is a variance-reduction technique for the Monte Carlo method, developed in the late 1940s. Here we present a brief overview of Importance Sampling — more details and applications can be found, for example, in Srinivasan's book [17].

3.1 Basics

We consider the more general case of estimating $c = \mathrm{E}[g(X)] < \infty$ for a random variable X and a measurable function $g{:}\mathbb{R} \to \mathbb{R}^{\geqslant 0}$. (By defining $g(X) = I_B(X)$ we recover the previous case.) We assume that the distribution of X is absolutely continuous with respect to the Lebesgue measure, and denote by f the corresponding density. The *crude Monte Carlo* (MC) estimator is

$$\hat{c} = \frac{1}{N}\sum_{i=1}^{N} g(X_i)$$

where $X_1, \ldots, X_N$ be random variables iid with density f. By the strong law of large numbers, $\hat{c}$ converges to c with probability 1. Also, it is unbiased, and its variance is

$$\mathrm{Var}(\hat{c}) = \frac{1}{N}(\mathrm{E}[g^2(X)] - c^2)\ . \tag{2}$$

In our statistical model checking setting, we are interested in determining the probability that a stochastic system satisfies a certain temporal logic formula ϕ. In this setting, the random variables $X_1, \ldots, X_N$ are independent executions (simulations) $\sigma_1, \ldots, \sigma_N$ of the system, represented by time series of the system variables (traces). The function g is just the model checker that verifies whether a trace satisfies ϕ. Therefore, given a trace σ the random variable $g(\sigma)$ is again a Bernoulli — 1 if the trace σ satisfies ϕ, and 0 otherwise. Also, it is the random variable previously defined in (1).

We now introduce Importance Sampling. Suppose we had another (absolutely continuous) distribution for X, with corresponding density f_*, such that the ratio f/f_* is well-defined. The entire theory of importance sampling rests upon the following fundamental identity:

$$\begin{aligned} c &= \mathrm{E}[g(X)] \\ &= \int_{\mathbb{R}} g(x)f(x)\ dx \\ &= \int_{\mathbb{R}} g(x)\frac{f(x)}{f_*(x)}f_*(x)\ dx \\ &= \int_{\mathbb{R}} g(x)W(x)f_*(x)\ dx \\ &= \mathrm{E}_*[g(X)W(X)] \end{aligned} \tag{3}$$

where $\mathrm{E}_*[\cdot]$ denotes expectation with respect to the density f_*. The term $W(x) = \frac{f(x)}{f_*(x)}$ is the *weighting* function, or *likelihood ratio.* Naturally, for all x such that $g(x)f(x) > 0$, it must be $f_*(x) > 0$. The density f_* is known as the *biasing* (or *proposal*) density.

The *Importance Sampling* (IS) estimator is

$$\hat{c}_{\mathrm{IS}} = \frac{1}{N}\sum_{i=1}^{N} g(X_i)W(X_i)$$

where $W(x) = f(x)/f_*(x)$ is the likelihood ratio and $X_1, \ldots, X_N$ are random variables iid with density f_* (the biasing density). The IS estimator is unbiased by (3), and its variance is:

$$\mathrm{Var}(\hat{c}_{\mathrm{IS}}) = \frac{1}{N}(\mathrm{E}_*[g^2(X)W^2(X)] - c^2)\ . \tag{4}$$

The crucial problem in importance sampling is to find a biasing density such that the variance (4) of the IS estimator is smaller than the variance (2) of the crude MC estimator.

It turns out that there exists a biasing density which can minimize the variance (4) of the IS estimator. In particular, it is easy to verify that when the function g is non-negative the following *optimal biasing* density actually results in a zero-variance estimator:

$$f_*(x) = \frac{g(x)f(x)}{c}\ .$$

But in practice it is difficult to sample from f_*, since it depends on $c = \mathrm{E}[g(X)]$, the (unknown) quantity we are trying to estimate. Therefore, instead of trying to come up with the optimal density, it may be preferable to search in a parametrized family of densities for a biasing density "close" to the optimal one. This is exactly the approach taken by the cross-entropy method.

4 The Cross-Entropy Method

The cross-entropy method was introduced in 1999 by Rubinstein [14]. Assume that the original (or nominal) density f of X belongs to a parametric family $\{f(\cdot, u) \,|\, u \in \mathcal{U}\}$, and in particular $f(\cdot) = f(\cdot, v)$ for some fixed $v \in \mathcal{U}$. (For example, a common family is the *natural exponential family.*) The method chooses the biasing density from the family such that the Kullback-Leibler divergence between the optimal biasing density and the chosen density is minimal.

The cross-entropy method has two basic steps:

1. find a density with minimal Kullback-Leibler divergence with respect to the optimal biasing density;
2. perform importance sampling with the biasing density computed in step 1 to estimate $\mathrm{E}[g(X)]$.

We will see that step 1 actually requires to sample X. In practice, the number of samples generated for step 2 will be larger than for step 1.

Definition 3. *The* Kullback-Leibler *divergence of two densities f, h is*

$$\mathcal{D}(f, h) = \int_{\mathbb{R}} f(x) \ln \frac{f(x)}{h(x)}\; dx.$$

The Kullback-Leibler divergence is also known as the *cross-entropy* (CE). Formally, $\mathcal{D}$ is not a distance, since it is not symmetric, *i.e.*, $\mathcal{D}(f, h) \neq \mathcal{D}(h, f)$ in general. However, it can be shown that $\mathcal{D}$ is always non-negative, and that $\mathcal{D}(f, h) = 0$ iff $f = h$. Therefore, the CE can be useful in assessing how close two densities are.

We recall that our task is to estimate $c = \mathrm{E}[g(X)]$, where X is a random variable with density f and g is a non-negative, measurable function. We want to find a density in the parametric family such that the CE with the optimal biasing density f_* is minimal. Therefore, we need to solve the minimization problem:

$$u^* = \operatorname*{argmin}_{u \in \mathcal{U}}\; \mathcal{D}(f_*(\cdot), f(\cdot, u))$$

where $f_*(x) = g(x)f(x, v)/c$ is the optimal biasing density. This can be turned into a maximization problem as follows:

$$\begin{aligned}
\operatorname*{argmin}_{u \in \mathcal{U}}\; \mathcal{D}(f_*(\cdot), f(\cdot, u)) &= \operatorname*{argmin}_{u \in \mathcal{U}}\; \mathrm{E}_* \left[\ln \frac{f_*(X)}{f(X, u)} \right] \\
&= \operatorname*{argmin}_{u \in \mathcal{U}} \int_{\mathbb{R}} f_*(x) \ln f_*(x)\, dx - \int_{\mathbb{R}} f_*(x) \ln f(x, u)\, dx \\
&= \operatorname*{argmax}_{u \in \mathcal{U}} \int_{\mathbb{R}} f_*(x) \ln f(x, u)\, dx \\
&= \operatorname*{argmax}_{u \in \mathcal{U}} \int_{\mathbb{R}} g(x) f(x, v) \ln f(x, u)\, dx \\
&= \operatorname*{argmax}_{u \in \mathcal{U}} \mathrm{E}[g(X) \ln f(X, u)]
\end{aligned}$$

where in the second step we used the fact is $\mathcal{D}$ is non-negative and that the first integral does not depend on u. It turns out that for certain families of densities the maximization problem can be solved *analytically* [15, Chapter 3].

We now assume that $\mathbf{X}$ is a random vector, *i.e.*, $\mathbf{X}{:}\Omega \to \mathbb{R}^n$, which implies that function g must be defined over $\mathbb{R}^n$. Note that this does not change what we obtained so far. The optimal parameter $\mathbf{u}^* = \text{argmax}_{u \in \mathcal{U}} \, \mathrm{E}[g(\mathbf{X}) \ln f(\mathbf{X}, u)]$ when $\mathbf{X}$ is in an exponential family of distributions is:

$$u_j^* = \frac{\mathrm{E}[g(\mathbf{X}) X_j]}{\mathrm{E}[g(\mathbf{X})]}$$

where $\mathbf{u}^* = (u_1^*, \ldots, u_n^*)$ and X_j is the j-th component of $\mathbf{X}$.

The optimal parameter thus depends on the quantity we wish to estimate, *i.e.*, $\mathrm{E}[g(\mathbf{X})]$, and therefore u^* needs itself to be estimated by MC simulation. In the one-dimensional case we have that

$$u^* = \frac{\mathrm{E}[g(X) X]}{\mathrm{E}[g(X)]}$$

and u^* may be estimated from a sample $X_1, \ldots, X_N$ iid with density f (the nominal density) as:

$$\hat{u}^* = \frac{\sum_{i=1}^N g(X_i) X_i}{\sum_{i=1}^N g(X_i)} \; . \tag{5}$$

However, in statistical model checking $g(X_i)$ is either 1 or 0 — a sample trace either satisfies a temporal logic property or it does not. Furthermore, in the rare event case it will be very unlikely to "see" a sample trace that satisfies the temporal logic property, which means that for reasonable sample sizes Eq.(5) would just give $\frac{0}{0}$.

The problem can be circumvented by noting that

$$u^* = \frac{\mathrm{E}[g(X) X]}{\mathrm{E}[g(X)]} = \frac{\mathrm{E}_w[g(X) W(X, w) X]}{\mathrm{E}_w[g(X) W(X, w)]}$$

where $W(x, w) = f(x)/f(x, w)$ and $w \in \mathcal{U}$ is an arbitrary parameter (recall that $f(x) = f(x, v)$ is the nominal density of X). Note that the expectation is computed with respect to the biased density $f(\cdot, w)$. Again, u^* can be estimated by

$$\hat{u}^* = \frac{\sum_{i=1}^N g(X_i) W(X_i, w) X_i}{\sum_{i=1}^N g(X_i) W(X_i, w)} \tag{6}$$

where each X_i is distributed as $f(\cdot, w)$. Basically, we use importance sampling with a biasing density given by the parameter w. Intuitively, w would have to be chosen in such a way that the estimator (6) is well-defined. This means that w should substantially increase the probability of the event $g(X) = 1$. In the literature w is know as the *tilting parameter.*

In the random vector case, we have samples $\mathbf{X}_1, \ldots, \mathbf{X}_N$ iid as $f(\cdot, w)$ and the j-th component of the optimal parameter $\mathbf{u}^*$ is estimated by

$$\hat{u}_j^* = \frac{\sum_{i=1}^{N} g(\mathbf{X}_i) W(\mathbf{X}_i, w) X_{ij}}{\sum_{i=1}^{N} g(\mathbf{X}_i) W(\mathbf{X}_i, w)}$$

where X_{ij} is the j-th component of $\mathbf{X}_i$.

5 Experiments

We report preliminary results showing that our technique can be utilized to efficiently address the rare-event problem in statistical model checking. We have considered an example of CPS that is part of the Stateflow/Simulink package demos. The model[1] describes a fault-tolerant fuel control system for a gasoline engine. It detects sensor failures, and dynamically adjusts the control law to provide seamless operation. The system aims at keeping the air-fuel ratio close to the *stoichiometric* ratio of 14.6. The "correct" fuel rate is estimated by taking into account sensor readings for the amount of oxygen present in the exhaust gas (EGO), for the engine speed, throttle command and manifold absolute pressure. In the event of a single sensor fault, *e.g.*, the EGO sensor, the system detects the situation, computes an estimate for the sensor's reading, and operates the engine with a higher fuel flow rate. If two or more sensors fail, the engine is shut down, since the system cannot reliably control the air-fuel ratio.

The Stateflow control logic of the system has a total of 24 locations, grouped in 6 parallel states. The Simulink part of the system is described by several nonlinear equations and a linear differential equation with a switching condition. Overall, this model provides a representative summary of the important features of a CPS.

Our stochastic system is obtained by introducing random faults in the EGO, speed and manifold pressure sensors. We model the faults by three independent Poisson processes with different arrival rates. When a fault occurs, it is "repaired" with a fixed service time of one second (*i.e.*, the sensor remains in fault condition for one second, then it resumes normal operation). The model has no free inputs, since the throttle command provides a periodic triangular input, and the nominal speed is never changed. This ensures that, once we set the three fault rates, for any given temporal logic property ϕ the probability that the model satisfies ϕ does not change.

For our experiments we model checked the following BLTL formula ϕ:

$$\phi = \mathbf{F}^{100}\mathbf{G}^{1}(FuelFlowRate = 0)).$$

Informally, we would like to estimate the probability that within 100 seconds the fuel flow rate stays at zero for one second. The nominal fault rates for the

[1] More information on the model is available at `http://mathworks.com/products/simulink/demos.html?file=/products/demos/shipping/simulink/sldemo_fuelsys.html`

three sensors are all equal to 1/3600. Since engine shutdown occurs when two or more sensors are faulty, the probability that the system satisfies ϕ is likely to be very close to 0. To compute the optimal biasing density we used tilting rates all equal to 1/10.

In the table below we report our preliminary results. We performed two experiments, depending on the number of samples used to compute the optimal CE rates (step 1) and in the importance sampling phase (step 2). In the table we report the estimate for the probability that ϕ holds, the (approximate) relative error, and the total computation time (*i.e.*, simulation, model checking, and CE method). The experiments have been performed on a 2.2GHz Opteron 6174 computer running Matlab R2010b on Linux (64-bit).

		Estimate	**RE**	**Time (h)**
Samples	**step 1** : 1,000 **step 2** : 10,000	5.1×10^{-15}	0.47	1.7
	step 1 : 10,000 **step 2** : 100,000	2.17×10^{-14}	0.13	17.8

From the magnitude of the probability estimates, we see that a crude Monte Carlo estimation would require about 10^{14} samples just to obtain one "success" sample. With feasible sample sizes of the order of 10^5, the Monte Carlo estimator would most likely return 0, thus incurring in a high error. Techniques based on confidence interval computation (*e.g.*, Chernoff bound) would require even larger sample sizes.

6 Conclusions

Statistical model checking efficiently addresses verification by combining the Monte Carlo method with temporal logic model checking. The technique is especially useful for verifying systems with very large state spaces, such as cyber-physical systems. The main problem with statistical model checking is caused by rare events. We have showed that Importance Sampling and the Cross-Entropy method can address this problem. In particular, we have successfully verified a representative example of cyber-physical system coded as a Stateflow-Simulink model, for which traditional verification techniques are not feasible.

References

1. Baier, C., Clarke, E.M., Hartonas-Garmhausen, V., Kwiatkowska, M.Z., Ryan, M.: Symbolic model checking for probabilistic processes. In: Degano, P., Gorrieri, R., Marchetti-Spaccamela, A. (eds.) ICALP 1997. LNCS, vol. 1256, pp. 430–440. Springer, Heidelberg (1997)
2. Baier, C., Haverkort, B.R., Hermanns, H., Katoen, J.-P.: Model-checking algorithms for continuous-time Markov chains. IEEE Trans. Software Eng. 29(6), 524–541 (2003)

3. Ciesinski, F., Größer, M.: On probabilistic computation tree logic. In: Baier, C., Haverkort, B.R., Hermanns, H., Katoen, J.-P., Siegle, M. (eds.) Validation of Stochastic Systems. LNCS, vol. 2925, pp. 147–188. Springer, Heidelberg (2004)
4. Courcoubetis, C., Yannakakis, M.: The complexity of probabilistic verification. Journal of the ACM 42(4), 857–907 (1995)
5. Grosu, R., Smolka, S.A.: Carlo Model Checking. In: Halbwachs, N., Zuck, L.D. (eds.) TACAS 2005. LNCS, vol. 3440, pp. 271–286. Springer, Heidelberg (2005)
6. Hérault, T., Lassaigne, R., Magniette, F., Peyronnet, S.: Approximate probabilistic model checking. In: Steffen, B., Levi, G. (eds.) VMCAI 2004. LNCS, vol. 2937, pp. 73–84. Springer, Heidelberg (2004)
7. Hinton, A., Kwiatkowska, M.Z., Norman, G., Parker, D.: PRISM: A tool for automatic verification of probabilistic systems. In: Hermanns, H. (ed.) TACAS 2006. LNCS, vol. 3920, pp. 441–444. Springer, Heidelberg (2006)
8. Jha, S.K., Clarke, E.M., Langmead, C.J., Legay, A., Platzer, A., Zuliani, P.: A Bayesian approach to Model Checking biological systems. In: Degano, P., Gorrieri, R. (eds.) CMSB 2009. LNCS, vol. 5688, pp. 218–234. Springer, Heidelberg (2009)
9. Koymans, R.: Specifying real-time properties with metric temporal logic. Real-time Systems 2(4), 255–299 (1990)
10. Kwiatkowska, M.Z., Norman, G., Parker, D.: Symmetry reduction for probabilistic model checking. In: Ball, T., Jones, R.B. (eds.) CAV 2006. LNCS, vol. 4144, pp. 234–248. Springer, Heidelberg (2006)
11. Maler, O., Nickovic, D.: Monitoring temporal properties of continuous signals. In: Lakhnech, Y., Yovine, S. (eds.) FORMATS 2004 and FTRTFT 2004. LNCS, vol. 3253, pp. 152–166. Springer, Heidelberg (2004)
12. Parnas, D.L.: Really rethinking 'Formal Methods'. IEEE Computer 43(1), 28–34 (2010)
13. Pnueli, A.: The temporal logic of programs. In: FOCS, pp. 46–57. IEEE, Los Alamitos (1977)
14. Rubinstein, R.Y.: The cross-entropy method for combinatorial and continuous optimization. Methodology and Computing in Applied Probability 2, 127–190 (1999)
15. Rubinstein, R.Y., Kroese, D.P.: The Cross-Entropy Method. Springer, Heidelberg (2004)
16. Sen, K., Viswanathan, M., Agha, G.: Statistical model checking of black-box probabilistic systems. In: Alur, R., Peled, D.A. (eds.) CAV 2004. LNCS, vol. 3114, pp. 202–215. Springer, Heidelberg (2004)
17. Srinivasan, R.: Importance Sampling. Springer, Heidelberg (2002)
18. Younes, H.L.S., Clarke, E.M., Zuliani, P.: Statistical verification of probabilistic properties with unbounded until. In: Davies, J. (ed.) SBMF 2010. LNCS, vol. 6527, pp. 144–160. Springer, Heidelberg (2011)
19. Younes, H.L.S., Kwiatkowska, M.Z., Norman, G., Parker, D.: Numerical vs. statistical probabilistic model checking. STTT 8(3), 216–228 (2006)
20. Younes, H.L.S., Musliner, D.J.: Probabilistic plan verification through acceptance sampling. In: AIPS Workshop on Planning via Model Checking, pp. 81–88 (2002)
21. Younes, H.L.S., Simmons, R.G.: Probabilistic verification of discrete event systems using acceptance sampling. In: Brinksma, E., Larsen, K.G. (eds.) CAV 2002. LNCS, vol. 2404, pp. 223–235. Springer, Heidelberg (2002)
22. Younes, H.L.S., Simmons, R.G.: Statistical probabilistic model checking with a focus on time-bounded properties. Inf. Comput. 204(9), 1368–1409 (2006)
23. Zuliani, P., Platzer, A., Clarke, E.M.: Bayesian statistical model checking with application to Stateflow/Simulink verification. In: HSCC, pp. 243–252 (2010)