Global PAC Bounds for Learning Discrete Time Markov Chains

Hugo Bazille, Blaise Genest, Cyrille Jegourel, Jun Sun

Global PAC Bounds for Learning Discrete Time Markov Chains*,**

Hugo Bazille[1], Blaise Genest[1], Cyrille Jegourel[2], and Jun Sun[3]

[1] Univ Rennes, CNRS & Rennes 1, France
{hbazille,bgenest}@irisa.fr
[2] Singapore University of Technology and Design, Singapore
cyrille.jegourel@gmail.com
[3] Singapore Management University, Singapore
junsun@smu.edu.sg

Abstract Learning models from observations of a system is a powerful tool with many applications. In this paper, we consider learning Discrete Time Markov Chains (DTMC), with different methods such as *frequency estimation* or *Laplace smoothing*. While models learnt with such methods converge asymptotically towards the exact system, a more practical question in the realm of trusted machine learning is how accurate a model learnt with a limited time budget is. Existing approaches provide bounds on how close the model is to the original system, in terms of bounds on *local* (transition) probabilities, which has unclear implication on the *global* behavior.
In this work, we provide *global bounds on the error* made by such a learning process, in terms of global behaviors formalized using *temporal logic*. More precisely, we propose a learning process ensuring a bound on the error in the probabilities of these properties. While such learning process cannot exist for the full LTL logic, we provide one ensuring a bound that is uniform over all the formulas of CTL. Further, given one time-to-failure property, we provide an improved learning algorithm. Interestingly, frequency estimation is sufficient for the latter, while Laplace smoothing is needed to ensure non-trivial uniform bounds for the full CTL logic.

1 Introduction

Discrete-Time Markov Chains (DTMC) are commonly used in model checking to model the behavior of stochastic systems [3,4,7,25]. A DTMC is described by a set of states and transition probabilities between these states. The main issue with modeling stochastic systems using DTMCs is to obtain the transition probabilities. One appealing approach to overcome this issue is to observe the system and to *learn automatically* these transition probabilities [8,29], e.g., using frequency estimation or Laplace (or additive) smoothing [12]. Frequency

* All authors have contributed equally.

** Jun Sun's research is supported by the National Research Foundation Singapore under its AI Singapore Programme (Award Number: AISG-RP-2019-012)

estimation works by observing a long run of the system and estimating each individual transition by its empirical frequency. However, in this case, the unseen transitions are estimated as zeros. Once the probability of a transition is set to zero, the probability to reach a state could be tremendously changed, e.g., from 1 to 0 if the probability of this transition in the system is small but non-zero. To overcome this problem, when the set of transitions with non-zero probability is known (but not their probabilities), Laplace smoothing assigns a positive probability to the unseen transitions, i.e., by adding a small quantity both to the numerator and the denominator of the estimate used in frequency estimation. Other smoothing methods exist, such as Good-Turing [15] and Kneser-Sey estimations [7], notably used in natural language processing. Notwithstanding smoothing generates estimation biases, all these methods converge asymptotically to the exact transition probabilities.

In practice, however, there is often limited budget in observing and learning from the system, and the validity of the learned model is in question. In trusted machine learning, it is thus crucial to measure how the learned model differs from the original system and to provide practical guidelines (e.g., on the number of observations) to guarantee some control of their divergence.

Comparing two Markov processes is a common problem that relies on a notion of divergence. Most existing approaches focus on deviations between the probabilities of local transitions (e.g., [10,26,5]). However, a single deviation in a transition probability between the original system and the learned model may lead to large differences in their global behaviors, even when no transitions are overlooked, as shown in our example 1. For instance, the probability of reaching certain state may be magnified by paths which go through the same deviated transition many times. It is thus important to use a measure that quantifies the differences over global behaviors, rather than simply checking whether the differences between the individual transition probabilities are low enough.

Technically, the knowledge of a lower bound on the transition probabilities is often assumed [14,1]. While it is a soft assumption in many cases, such as when all transition probabilities are large enough, it is less clear how to obtain such a lower bound in other cases, such as when a very unlikely transition exists (e.g., a very small error probability). We show how to handle this in several cases: learning a Markov chain accurate w.r.t. this error rate, or learning a Markov chain accurate over all its global behaviors, which is possible if we know the underlying structure of the system (e.g., because we designed it, although we do not know the precise transition probabilities which are governed by uncertain forces). For the latter, we define a new concept, namely *conditioning* of a DTMC.

In this work, we model global behaviors using temporal logics. We consider Linear Temporal Logic (LTL) [23] and Computational Tree Logic (CTL) [11]. Agreeing on all formulas of LTL means that the first order behaviors of the system and the model are the same, while agreeing on CTL means that the system and the model are bisimilar [2]. Our goal is to provide stopping rules in the learning process of DTMCs that provides Probably Approximately Correct (PAC) bounds on the error in probabilities of every property in the logic between

the model and the system. In Section 2, we recall useful notions on DTMCs and PAC-learning. We point out related works in Section 3. Our main contributions are as follows:

- In Section 4, we show that it is impossible to learn a DTMC accurate for all LTL formulas, by adapting a result from [13].
- We provide in Section 6 a learning process bounding the difference in probability *uniformly over all CTL properties*. To do so, we use Laplace smoothing, and we provide rationale on choosing the smoothing parameter.
- For the particular case of a time-to-failure property, notably used to compute the mean time between failures of critical systems (see e.g., [24]), we provide tighter bounds in Section 5, based on frequency estimation.

In Section 4, we formally state the problem and the specification that the learning process must fulfill. We also show our first contribution: the impossibility of learning a DTMC, accurate for all LTL formulas. Nevertheless, we prove in Section 5 our second contribution: the existence of a global bound for the time-to-failure properties, notably used to compute the mean time between failures of critical systems (see e.g., [24]) and provide an improved learning process, based on frequency estimation. In Section 6, we present our main contribution: a global bound guaranteeing that the original system and a model learned by Laplace smoothing have similar behaviors for all the formulas in CTL. We show that the error bound that we provide on the probabilities of properties is close to optimal. We evaluate our approach in Section 7 and conclude in Section 8.

2 Background

In this section, we introduce the notions and notations used throughout the paper. A stochastic system $\mathcal{S}$ is interpreted as a set of interacting components in which the state is determined randomly with respect to a global probability measure described below.

Definition 1 (Discrete-Time Markov Chains). *A Discrete-Time Markov Chain is a triple $\mathcal{M} = (S, \mu, A)$ where:*

- *S is a finite set of states;*
- *$\mu : S \to [0, 1]$ is an initial probability distribution over S;*
- *$A : S \times S \to [0, 1]$ is a transition probability matrix, such that for every $s \in S$, $\sum_{s' \in S} A(s, s') = 1$.*

We denote by m the cardinal of S and $A = (a_{ij})_{1 \leq i,j \leq m} = (A(i, j))_{1 \leq i,j \leq m}$ the probability matrix. Figs. 1 and 2 show the graph of two DTMCs over 3 states $\{s_1, s_2, s_3\}$ (with $\mu(s_1) = 1$). A run is an infinite sequence $\omega = s_0 s_1 \cdots$ and a path is a finite sequence $\omega = s_0 \cdots s_l$ such that $\mu(s_0) > 0$ and $A(s_i, s_{i+1}) > 0$ for all i, $0 \leq i \leq l$. The length $|\omega|$ of a path ω is its number of transitions.

The cylinder set of ω, denoted $C(\omega)$, consists of all the runs starting by a path ω. Markov chain $\mathcal{M}$ underlies a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, where Ω is the

set of all runs from $\mathcal{M}$; $\mathcal{F}$ is the sigma-algebra generated by all the cylinders $C(\omega)$ and $\mathbb{P}$ is the unique probability measure [31] such that $\mathbb{P}(C(s_0 \cdots s_l)) = \mu(s_0) \prod_{i=1}^{l} A(s_{i-1}, s_i)$. For simplicity, we assume a unique initial state s_0 and denote $\mathbb{P}(\omega) = \mathbb{P}(C(\omega))$. Finally, we sometimes use the notation $\mathbb{P}_i^A$ to emphasize that the probability distribution is parameterized by the probability matrix A, and the starting state is i.

2.1 PAC-learning for properties

To analyze the behavior of a system, properties are specified in temporal logic (e.g., LTL or CTL, respectively introduced in [23] and [11]). Given a logic $\mathcal{L}$ and φ a property of $\mathcal{L}$, decidable in finite time, we denote $\omega \models \varphi$ if a path ω satisfies φ. Let $z : \Omega \times \mathcal{L} \to \{0, 1\}$ be the function that assigns 1 to a path ω if $\omega \models \varphi$ and 0 otherwise. In what follows, we assume that we have a procedure that draws path ω with respect to $\mathbb{P}^A$ and outputs $z(\omega, \varphi)$. Further, we denote $\gamma(A, \varphi)$ the probability that a path drawn with respect to $\mathbb{P}^A$ satisfies φ. We omit the property or the matrix in the notation when it is clear from the context. Finally, note that the behavior of $z(., \varphi)$ can be modeled as a Bernoulli random variable Z_φ parameterized by the mean value $\gamma(A, \varphi)$.

Probably Approximately Correct (PAC) learning [27] is a framework for mathematical analysis of machine learning. Given $\varepsilon > 0$ and $0 < \delta < 1$, we say that a property φ of $\mathcal{L}$ is PAC-learnable if there is an algorithm $\mathcal{A}$ such that, given a sample of n paths drawn according to the procedure, with probability of at least $1-\delta$, $\mathcal{A}$ outputs in polynomial time (in $1/\varepsilon$ and $1/\delta$) an approximation of the average value for Z_φ close to its exact value, up to an error less than or equal to ε. Formally, φ is PAC-learnable if and only if $\mathcal{A}$ outputs an approximation $\hat{\gamma}$ such that:

$$\mathbb{P}\left(|\gamma - \hat{\gamma}| > \varepsilon\right) \leq \delta \tag{1}$$

Moreover, if the above statement for algorithm $\mathcal{A}$ is true for every property in $\mathcal{L}$, we say that $\mathcal{A}$ is a PAC-learning algorithm for $\mathcal{L}$.

2.2 Monte-Carlo estimation and algorithm of Chen

Given a sample W of n paths drawn according to $\mathbb{P}^A$ until φ is satisfied or violated (for φ such that with probability 1, φ is eventually satisfied or viol-

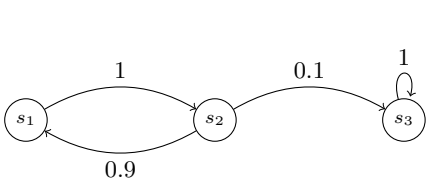

Figure 1: An example of DTMC $\mathcal{M}_1$

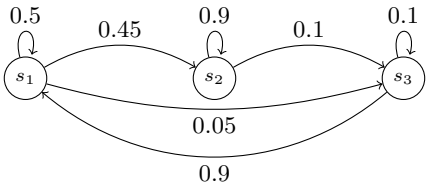

Figure 2: DTMC $\mathcal{M}_2$

ated), the crude Monte-Carlo estimator, denoted $\hat{\gamma}_W(A,\varphi)$, of the mean value for the random variable Z_φ is given by the empirical frequency: $\hat{\gamma}_W(A,\varphi) = \frac{1}{n}\sum_{i=1}^{n} z(\omega_i) \approx \gamma(A,\varphi)$.

The Okamoto inequality [22] (also called the Chernoff bound in the literature) is often used to guarantee that the deviation between a Monte-Carlo estimator $\hat{\gamma}_W$ and the exact value γ by more than $\varepsilon > 0$ is bounded by a predefined confidence parameter δ. However, several sequential algorithms have been recently proposed to guarantee the same confidence and accuracy with fewer samples[4]. In what follows, we use the Massart bound [**?**], implemented in the algorithm of Chen [6].

Theorem 1 (Chen bound). *Let $\varepsilon > 0$, δ such that $0 < \delta < 1$ and $\hat{\gamma}_W$ be the crude Monte-Carlo estimator, based on n samples, of probability γ.*

If $n \geq \frac{2}{\varepsilon^2}\log\left(\frac{2}{\delta}\right)\left[\frac{1}{4} - (|\frac{1}{2} - \hat{\gamma}_W| - \frac{2}{3}\varepsilon)^2\right]$,

$$\mathbb{P}(|\gamma - \hat{\gamma}_W| > \varepsilon) \leq \delta.$$

To ease the readability, we write $n_{\text{succ}} = \sum_{i=1}^{n} z(\omega_i)$ and $H(n, n_{\text{succ}}, \epsilon, \delta) = \frac{2}{\varepsilon^2}\log\left(\frac{2}{\delta}\right)\left[\frac{1}{4} - (|\frac{1}{2} - \hat{\gamma}_W| - \frac{2}{3}\varepsilon)^2\right]$. When it is clear from the context, we only write $H(n)$. Then, the algorithm $\mathcal{A}$ that stops sampling as soon as $n \geq H(n)$ and outputs a crude Monte-Carlo estimator for $\gamma(A,\varphi)$ is a PAC-learning algorithm for φ. The condition over n is called the stopping criteria of the algorithm. As far as we know, this algorithm requires fewer samples than the other sequential algorithms (see e.g., [18]). Note that the estimation of a probability close to $1/2$ likely requires more samples since $H(n)$ is maximized in $\hat{\gamma}_W = 1/2$.

3 Related work

Our work shares similar statistical results (see Section 2.3) with Statistical Model Checking (SMC) [31]. However, the context and the outputs are different. SMC is a simulation-based approach that aims to estimate one probability for a given property [9,28], within acceptable margins of error and confidence [17,18,32]. A challenge in SMC is posed by unbounded properties (e.g., fairness) since the sampled executions are finite. Some algorithms have been proposed to handle unbounded properties but they require the knowledge of the minimal probability transition of the system [14,1], which we avoid. While this restriction is light in many contexts, such as when every state and transition appears with a sufficiently high probability, contexts where probabilities are unknown and some are very small seems much harder to handle. In the following, we propose 2 solutions not requiring this assumption. The first one is the closest to SMC: we learn a Markov chain accurate for a given time-to-error property, and it does not require knowledge on the Markov chain. The second one is much more ambitious than SMC as it learns a Markov chain accurate for *all* its global behaviors, formalized as all properties of a temporal logic; it needs the assumption that the set

[4] We recall the Okamoto-Chernoff bound in the extended version(as well as the Massart bound), but we do not use it in this work.

of transitions is known, but not their probabilities nor a lower bound on them. This assumption may seem heavy, but it is reasonable for designers of systems, for which (a lower bound on) transition probabilities are not known (e.g. some error rate of components, etc).

For comparison with SMC, our final output is the (approximated) transition matrix of a DTMC rather than one (approximated) probability of a given property. This learned DTMC can be used for different purposes, e.g. as a component in a bigger model or as a simulation tool. In terms of performances, we will show that we can learn a DTMC w.r.t. a given property with the same number of samples as we need to estimate this property using SMC (see Section 5). That is, there is no penalty to estimate a DTMC rather than estimate one probability, and we can scale as well as SMC. In terms of expressivity, we can handle unbounded properties (e.g. fairness properties). Even better, we can learn a DTMC accurate uniformly over a possibly infinite set of properties, e.g. all formulas of CTL. This is something SMC is not designed to achieve.

Other related work can be cited: In [13], the authors investigate several distances for the estimation of the difference between DTMCs. But they do not propose algorithms for learning. In [16], the authors propose to analyze the learned model a posteriori to test whether it has some good properties. If not, then they tweak the model in order to enforce these properties. Also, several PAC-learning algorithms have been proposed for the estimation of stochastic systems [5,10] but these works focus on local transitions instead of global properties.

4 Problem statement

In this work, we are interested to learn a DTMC model from a stochastic system $\mathcal{S}$ such that the behaviors of the system and the model are similar. We assume that the original system is a DTMC parameterized by a matrix A of transition probabilities. The transition probabilities are unknown, but the set of states of the DTMC is assumed to be known.

Our goal is to provide a learning algorithm $\mathcal{A}$ that guarantees an accurate estimation of $\mathcal{S}$ with respect to certain global properties. For that, a sampling process is defined as follows. A path (i.e., a sequence of states from s_0) of $\mathcal{S}$ is observed, and at steps specified by the sampling process, a reset action is performed, setting $\mathcal{S}$ back to its initial state s_0. Then another path is generated. This process generates a set W of paths, called traces, used to learn a matrix $\hat{A}_W$. Formally, we want to provide a learning algorithm that guarantees the following specification:

$$\mathbb{P}(\mathcal{D}(A, \hat{A}_W) > \varepsilon) \leq \delta \tag{2}$$

where $\varepsilon > 0$ and $\delta > 0$ are respectively *accuracy* and *confidence* parameters and $\mathcal{D}(A, \hat{A}_W)$ is a measure of the divergence between A and $\hat{A}_W$.

There exist several ways to specify the divergence between two transition matrices, e.g., the Kullback-Leibler divergence [19] or a distance based on a

matrix norm. However, the existing notions remain heuristic because they are based on the difference between the individual probabilistic transitions of the matrix. We argue that what matters in practice is often to quantify the similarity between the global behaviors of the systems and the learned model.

In order to specify the behaviors of interest, we use a property φ or a set of properties Ψ on the set of states visited. We are interested in the difference between the probabilities of φ (i.e., the measure of the set of runs satisfying φ) with respect to A and $\hat{A}_W$. We want to ensure that this difference is less than some predefined ε with (high) probability $1-\delta$. Hence, we define:

$$\mathcal{D}_\varphi(A, \hat{A}_W) = |\gamma(A, \varphi) - \gamma(\hat{A}_W, \varphi)| \tag{3}$$

$$\mathcal{D}_\Psi(A, \hat{A}_W) = \max_{\varphi \in \Psi}(\mathcal{D}_\varphi(A, \hat{A}_W)) \tag{4}$$

Our problem is to construct an algorithm which takes the following as inputs:

- confidence δ, $0 < \delta < 1$,
- absolute error $\varepsilon > 0$, and
- a property φ (or a set of properties Ψ),

and provides a learning procedure sampling a set W of paths, outputs $\hat{A}_W$, and terminates the sampling procedure while fulfilling Specification (2), with $\mathcal{D} = \mathcal{D}_\varphi$ $(= \mathcal{D}_\Psi)$.

In what follows, we assume that the confidence level δ and absolute error ε are fixed. We first start with a negative result: if Ψ is the set of LTL formulas [2], such a learning process is impossible.

Theorem 2. *Given $\varepsilon > 0$, $0 < \delta < 1$, and a finite set W of paths randomly drawn with respect to a DTMC A, there is no learning strategy such that, for every LTL formula φ,*

$$\mathbb{P}(|\gamma(A, \varphi) - \gamma(\hat{A}_W, \varphi)| > \varepsilon) \leq \delta \tag{5}$$

Note that contrary to Theorem 1, the deviation in Theorem 2 is a difference between two exact probabilities (of the original system and of a learned model). The theorem holds as long as $\hat{A}_W$ and A are not strictly equal, no matter how $\hat{A}_W$ is learned. To prove this theorem, we show that, for any number of observations, we can always define a sequence of LTL properties that violates the specification above. It only exploits a single deviation in one transition. The proof, inspired by a result from [13], is given in the extended version.

Example 1. We show in this example that in general, one needs to have some knowledge on the system in order to perform PAC learning - either a positive lower bound $\ell > 0$ on the lowest probability transition, as in [14,1], or the support of transitions (but no knowledge on their probabilities), as we use in Section 6. Further, we show that the latter assumption does not imply the former, as even if no transitions are overlooked, the error in some reachability property can

be arbitrarily close to 0.5 even with arbitrarily small error on the transition probabilities.

Let us consider DTMCs $A, \hat{A}, \hat{B}$ in Fig. 3, and formula $\mathbf{F}\, s_2$ stating that s_2 is eventually reached. The probabilities to satisfy this formula in $A, \hat{A}, \hat{B}$ are respectively $\mathbb{P}^A(\mathbf{F}\, s_2) = \frac{1}{2}$, $\mathbb{P}^{\hat{A}}(\mathbf{F}\, s_2) = \frac{2\tau-\eta}{4\tau} = \frac{1}{2} - \frac{\eta}{4\tau}$ and $\mathbb{P}^{\hat{B}}(\mathbf{F}\, s_2) = 0$.

Assume that A is the real system and that $\hat{A}$ and $\hat{B}$ are DTMCs we learned from A. Obviously, one wants to avoid learning $\hat{B}$ from A, as the probability of $\mathbf{F}\, s_2$ is very different in $\hat{B}$ and in $\hat{A}$ (0 instead of 0.5). If one knows that $\tau > \ell$ for some lower bound $\ell > 0$, then one can generate enough samples from s_1 to evaluate τ with an arbitrarily small error $\frac{\eta}{2} << \ell$ on probability transitions with an arbitrarily high confidence, and in particular learn a DTMC similar to $\hat{A}$.

On the other hand, if one knows there are transitions from s_1 to s_2 and to s_3, then immediately, one does not learn DTMC $\hat{B}$, but a DTMC similar to DTMC $\hat{A}$ (using e.g. Laplace smoothing [12]). While this part is straightforward with this assumption, evaluating τ is much harder when one does not know a priori a lower bound $\ell > 0$ such that $\tau > \ell$. That is very important: while one can make sure that the error $\frac{\eta}{2}$ on probability transitions is arbitrarily small, if τ is unknown, then it could be the case that τ is as small as $\frac{\eta}{2(1-\varepsilon)} > \frac{\eta}{2}$, for a small $\varepsilon > 0$. This gives us $\mathbb{P}^{\hat{A}}(\mathbf{F}\, s_2) = \frac{1}{2} - \frac{1-\varepsilon}{2} = \frac{\varepsilon}{2}$, which is arbitrarily small, whereas $\mathbb{P}^A(\mathbf{F}\, s_2) = 0.5$, leading to a huge error in the probability to reach s_2. We work around that problem in Section 6 by defining and computing the *conditioning* of DTMC $\hat{A}$. In some particular cases, as the one discussed in the next section, one can avoid that altogether (actually, the conditioning in these cases is perfect (=1), and it needs not be computed explicitly).

5 Learning for a time-to-failure property

In this section, we focus on property φ of reaching a failure state s_F from an initial state s_0 without re-passing by the initial state, which is often used for assessing the failure rate of a system and the mean time between failures (see e.g., [24]). We assume that with probability 1, the runs eventually re-pass by s_0 or reach s_F. Also, without loss of generality, we assume that there is a unique failure state s_F in A. We denote $\gamma(A, \varphi)$ the probability, given DTMC A, of satisfying property φ, i.e., the probability of a failure between two visits of s_0.

Assume that the stochastic system $\mathcal{S}$ is observed from state s_0. Between two visits of s_0, property φ can be monitored. If s_F is observed between two

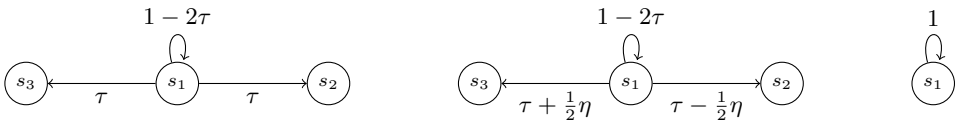

Figure 3: Three DTMCs $A, \hat{A}, \hat{B}$ (from left to right), with $0 < \eta < 2\tau < 1$

instances of s_0, we say that the path $\omega = s_0 \cdot \rho \cdot s_F$ satisfies φ, with $s_0, s_F \notin \rho$. Otherwise, if s_0 is visited again from s_0, then we say that the path $\omega = s_0 \cdot \rho \cdot s_0$ violates φ, with $s_0, s_F \notin \rho$. We call *traces* paths of the form $\omega = s_0 \cdot \rho \cdot (s_0 \vee s_F)$ with $s_0, s_F \notin \rho$. In the following, we show that it is sufficient to use a *frequency estimator* to learn a DTMC which provides a good approximation for such a property.

5.1 Frequency estimation of a DTMC

Given a set W of n traces, we denote n_{ij}^W the number of times a transition from state i to state j has occurred and n_i^W the number of times a transition has been taken from state i.

The *frequency estimator* of A is the DTMC $\hat{A}_W = (\hat{a}_{ij})_{1 \leq i,j \leq m}$ given by $\hat{a}_{ij} = \frac{n_{ij}^W}{n_i^W}$ for all i, j, with $\sum_{i=1}^m n_i^W = \sum_{i=1}^m \sum_{j=1}^m n_{ij}^W = |W|$. In other words, to learn $\hat{A}_W$, it suffices to count the number of times a transition from i to j occurred, and divide by the number of times state i has been observed. The matrix $\hat{A}_W$ is trivially a DTMC, except for states i which have not been visited. In this case, one can set $\hat{a}_{ij} = \frac{1}{m}$ for all states j and obtain a DTMC. This has no impact on the behavior of $\hat{A}_W$ as i is not reachable from s_0 in $\hat{A}_W$.

Let $\hat{A}_W$ be the matrix learned using the frequency estimator from the set W of traces, and let A be the real probabilistic matrix of the original system $\mathcal{S}$. We show that, in the case of time-to-failure properties, $\gamma(\hat{A}_W, \varphi)$ is equal to the crude Monte Carlo estimator $\hat{\gamma}_W(A, \varphi)$ induced by W.

5.2 PAC bounds for a time-to-failure property

We start by stating the main result of this section, bounding the error between $\gamma(A, \varphi)$ and $\gamma(\hat{A}_W, \varphi)$:

Theorem 3. *Given a set W of n traces such that $n = \lceil H(n) \rceil$, we have:*

$$\mathbb{P}\left(|\gamma(A, \varphi) - \gamma(\hat{A}_W, \varphi)| > \varepsilon\right) \leq \delta \tag{6}$$

where $\hat{A}_W$ is the frequency estimator of A.

To prove Theorem (3), we first invoke Theorem 1 to establish:

$$\mathbb{P}\left(|\gamma(A, \varphi) - \hat{\gamma}_W(A, \varphi)| > \varepsilon\right) \leq \delta \tag{7}$$

It remains to show that $\hat{\gamma}_W(A, \varphi) = \gamma(\hat{A}_W, \varphi)$:

Proposition 1. *Given a set W of traces, $\gamma(\hat{A}_W, \varphi) = \hat{\gamma}_W(A, \varphi)$.*

It might be appealing to think that this result can be proved by induction on the size of the traces, mimicking the proof of computation of reachability probabilities by linear programming [2]. This is actually not the case. The remaining of this section is devoted to proving Proposition (1).

We first define $q_W(u)$ the number of occurrences of sequence u in the traces of W. Note that u can be a state, an individual transition or even a path. We also use the following definitions in the proof.

Definition 2 (Equivalence). *Two sets of traces W and W' are equivalent if for all $s, t \in S$, $\frac{q_W(s \cdot t)}{q_W(s)} = \frac{q_{W'}(s \cdot t)}{q_{W'}(s)}$.*

We define a set of traces W' equivalent with W, implying that $\hat{A}_W = \hat{A}_{W'}$. This set W' of traces satisfies the following:

Lemma 1. *For any set of traces W, there exists a set of traces W' such that:*
(i) W and W' are equivalent,
(ii) for all $r, s, t \in S$, $q_{W'}(r \cdot s \cdot t) = \dfrac{q_{W'}(r \cdot s) \times q_{W'}(s \cdot t)}{q_{W'}(s)}$.

The proof of Lemma 1 is provided in the extended version. In Lemma 1, (i) ensures that $\hat{A}_{W'} = \hat{A}_W$ and (ii) ensures the equality between the proportion of runs of W' passing by s and satisfying γ, denoted $\hat{\gamma}^s_{W'}$, and the probability of reaching s_F before s_0 starting from s with respect to $\hat{A}_{W'}$. Formally,

Lemma 2. *For all $s \in S$, $\mathbb{P}_s^{\hat{A}_{W'}}(\text{reach } s_f \text{ before } s_0) = \hat{\gamma}^s_{W'}$.*

Proof. Let S_0 be the set of states s with no path in $\hat{A}_{W'}$ from s to s_f without passing through s_0. For all $s \in S_0$, let $p_s = 0$. Also, let $p_{s_f} = 1$. Let $S_1 = S \setminus (S_0 \cup \{s_f\})$. Consider the system of equations (8) with variables $(p_s)_{s \in S_1} \in [0,1]^{|S_1|}$:

$$\forall\, s \in S_1, \quad p_s = \sum_{t=1}^{m} \hat{A}_{W'}(s,t) p_t \tag{8}$$

The system of equations (8) admits a unique solution according to [2] (Theorem 10.19. page 766). Then, $(\mathbb{P}_s^{\hat{A}_{W'}}(\text{reach } s_f \text{ before } s_0))_{s \in S_1}$ is trivially a solution of (8). But, since W' satisfies the conditions of Lemma 1, we also have that $(\hat{\gamma}^s_{W'})_{s \in S_1}$ is a solution of (8), and thus we have the desired equality. □

Notice that Lemma 2 does not hold in general with the set W. We have:

$$\begin{aligned}
\hat{\gamma}_W(A, \varphi) &= \hat{\gamma}^{s_0}_W \quad \text{(by definition)} \\
&= \hat{\gamma}^{s_0}_{W'} \quad \text{(by Lemma 1)} \\
&= \mathbb{P}_{s_0}^{\hat{A}_{W'}}(\text{reach } s_f \text{ before } s_0) \quad \text{(by Lemma 2)} \\
&= \mathbb{P}_{s_0}^{\hat{A}_W}(\text{reach } s_f \text{ before } s_0) \quad \text{(by Lemma 1)} \\
&= \gamma(\hat{A}_W, \varphi) \quad \text{(by definition).}
\end{aligned}$$

That concludes the proof of Proposition 1. It shows that learning can be as efficient as statistical model-checking on comparable properties.

6 Learning for the full CTL logic

In this section, we learn a DTMC $\hat{A}_W$ such that $\hat{A}_W$ and A have similar behaviors over all CTL formulas. This provides a much stronger result than on time-to-failure property, e.g., properties can involve liveness and fairness, and more importantly they are not known before the learning. Notice that PCTL [2] cannot be used, since an infinitesimal error on one > 0 probability can change the probability of a PCTL formula from 0 to 1. (State)-CTL is defined as follows:

Definition 3. *Let Prop be the set of state names. (State)-CTL is defined by the following grammar* $\varphi ::= \bot \mid \top \mid p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathbf{AX}\varphi \mid \mathbf{EX}\varphi \mid \mathbf{AF}\varphi \mid \mathbf{EF}\varphi \mid \mathbf{AF}\varphi \mid \mathbf{EG}\varphi \mid \mathbf{AG}\varphi \mid \mathbf{E}(\varphi\mathbf{U}\varphi) \mid \mathbf{A}(\varphi\mathbf{U}\varphi)$*, with* $p \in Prop$*.* **E***(xists) and* **A***(ll) are quantifiers on paths, ne***X***t,* **G***lobally,* **F***inally and* **U***ntil are path-specific quantifiers. Notice that some operators are redundant. A minimal set of operators is* $\{\top, \vee, \neg, \mathbf{EG}, \mathbf{EU}, \mathbf{EX}\}$*.*

As we want to compute the probability of *paths* satisfying a CTL formula, we consider the set Ψ of *path-CTL* properties, that is formulas φ of the form $\varphi = \mathbf{X}\varphi_1$, $\varphi = \varphi_1\mathbf{U}\varphi_2$, $\varphi = \mathbf{F}\varphi_1$ or $\varphi = \mathbf{G}\varphi_1$, with φ_1, φ_2 (state)-CTL formulas. For instance, the property considered in the previous section is $(\neg s_0)\mathbf{U}s_F$.

In this section, for the sake of simplicity, the finite set W of traces is obtained by observing paths till a state is seen twice on the path. Then, the reset action is used and another trace is obtained from another path. That is, a trace ω from W is of the form $\omega = \rho \cdot s \cdot \rho' \cdot s$, with $\rho \cdot s \cdot \rho'$ a loop-free path.

As explained in example 1, some additional knowledge on the system is necessary. In this section, we assume that the support of transition probabilities is known, i.e., for any state i, we know the set of states j such that $a_{ij} \neq 0$. This assumption is needed both for Theorem 5 and to apply Laplace smoothing.

6.1 Learning DTMCs with Laplace smoothing

Let $\alpha > 0$. For any state s, let k_s be the number of successors of s, that we know by hypothesis, and $T = \sum_{s \in S} k_s$ be the number of non-zero transitions. Let W be a set of traces, n_{ij}^W the number of transitions from state i to state j, and $n_i^W = \sum_j n_{ij}^W$. The *estimator for W with Laplace smoothing α* is the DTMC $\hat{A}_W^\alpha = (\hat{a}_{ij})_{1 \leq i,j \leq m}$ given for all i, j by:

$$\hat{a}_{ij} = \frac{n_{ij}^W + \alpha}{n_i^W + k_i\alpha} \text{ if } a_{ij} \neq 0 \quad \text{and} \quad \hat{a}_{ij} = 0 \text{ otherwise}$$

In comparison with the frequency estimator, the Laplace smoothing adds for each state s a term α to the numerator and k_s times α to the denominator. This preserves the fact that $\hat{A}_W^\alpha$ is a Markov chain, and it ensures that $\hat{a}_{ij} \neq 0$ iff $a_{ij} \neq 0$. In particular, compared with the frequency estimator, it avoids creating zeros in the probability tables.

6.2 Conditioning and Probability Bounds

Using Laplace smoothing slightly changes the probability of each transition by an additive offset η. We now explain how this small error η impacts the error on the probability of a CTL property.

Let A be a DTMC, and A_η be a DTMC such that $A_\eta(i,j) \neq 0$ iff $A(i,j) \neq 0$ for all states i, j, and such that $\sum_j |A_\eta(i,j) - A(i,j)| \leq \eta$ for all states i. For all states $s \in S$, let $R(s)$ be the set of states i such that there exists a path from i to s. Let $R_*(s) = R(s) \setminus \{s\}$. Since both DTMCs have the same support, R (and also R_*) is equal for A and A_η. Given m the number of states, the conditioning of A for $s \in S$ and $\ell \leq m$ is:

$$\mathrm{Cond}_s^\ell(A) = \min_{i \in R_*(s)} \mathbb{P}_i^A(\mathbf{F}_{\leq \ell} \neg R_*(s))$$

i.e., the minimal probability from state $i \in R_*(s)$ to move away from $R_*(s)$ in at most ℓ steps. Let ℓ_s be the minimal value such that $\mathrm{Cond}_s^{\ell_s}(A) > 0$. This minimal ℓ_s exists as $\mathrm{Cond}_s^m(A) > 0$ since, for all $s \in S$ and $i \in R_*(s)$, there is at least one path reaching s from i (this path leaves $R_*(s)$), and taking a cycle-free path, we obtain a path of length at most m. Thus, the probability $\mathbb{P}_i^A(\mathbf{F}_{\leq m} \neg R_*(s))$ is at least the positive probability of the cylinder defined by this finite path. Formally,

Theorem 4. *Denoting φ the property of reaching state s in DTMC A, we have:*

$$|\gamma(A,\varphi) - \gamma(A_\eta,\varphi)| < \frac{\ell_s \cdot \eta}{Cond_s^{\ell_s}(A)}$$

Proof. Let v_s be the stochastic vector with $v_s(s) = 1$. We denote $v_0 = v_{s_0}$. Let $s \in S$. We assume that $s_0 \in R_*(s)$ (else $\gamma(A,\varphi) = \gamma(A_\eta,\varphi)$ and the result is trivial). Without loss of generality, we can also assume that $A(s,s) = A_\eta(s,s) = 1$ (as we are interested in reaching s at any step). With this assumption:

$$|\gamma(A,\varphi) - \gamma(A_\eta,\varphi)| = \lim_{t\to\infty} |v_0 \cdot (A^t - A_\eta^t) \cdot v_s|$$

We bound this error, through bounding by induction on t:

$$E(t) = \max_{i \in R_*(s)} |v_i \cdot (A^t - A_\eta^t) \cdot v_s|$$

We then have trivially:

$$|\gamma(A,\varphi) - \gamma(A_\eta,\varphi)| \leq \lim_{t\to\infty} E(t)$$

Note that for $i = s$, $\lim_{t\to\infty} v_i \cdot (A^t) \cdot v_s = 1 = \lim_{t\to\infty} v_i \cdot A_\eta^t \cdot v_s$, and thus their difference is null.

Let $t \in \mathbb{N}$. We let $j \in R_*(s)$ such that $E(t) = |v_j \cdot (A^t - A_\eta^t) \cdot v_s|$.

By the triangular inequality, introducing the term $v_j \cdot A^{\ell_s} A_\eta^{t-k} \cdot v_s - v_j \cdot A^{\ell_s} A_\eta^{t-k} \cdot v_s = 0$, we have:

$$E(t) \leq |v_j \cdot (A_\eta^t - A^{\ell_s} A_\eta^{t-\ell_s}) \cdot v_s| + |(v_j \cdot A^{\ell_s}) \cdot (A_\eta^{t-\ell_s} - A^{t-\ell_s}) \cdot v_s|$$

We separate vector $(v_j \cdot A^{\ell_s}) = w_1 + w_2 + w_3$ in three sub-stochastic vectors w_1, w_2, w_3: vector w_1 is over $\{s\}$, and thus we have $w_1 \cdot A_\eta^{t-\ell_s} = w_1 = w_1 \cdot A^{t-\ell_s}$, and the term cancels out. Vector w_2 is over states of $R_*(s)$, with $\sum_{i \in R_*} w_2[i] \leq (1 - \mathrm{Cond}_s^{\ell_s}(A))$, and we obtain an inductive term $\leq (1 - \mathrm{Cond}_s^{\ell_s}(A))E(t - \ell_s)$. Last, vector w_3 is over states not in $R(s)$, and we have $w_3 \cdot A_\eta^{t-\ell_s} \cdot v_s = 0 = w_3 \cdot A^{t-\ell_s} \cdot v_s$, and the term cancels out.

We also obtain that $|v_j \cdot (A_\eta^t - A^{\ell_s} A_\eta^{t-\ell_s}) \cdot v_s| \leq \ell_s \cdot \eta$. Thus, we have the inductive formula $E(t) \leq (1 - \mathrm{Cond}_s^{\ell_s}(A))E(t - \ell_s) + \ell_s \cdot \eta$. It yields for all $t \in \mathbb{N}$:

$$E(t) \leq (\ell_s \cdot \eta) \sum_{i=1}^{\infty} (1 - \mathrm{Cond}_s^{\ell_s}(A))^i$$

$$E(t) \leq \frac{\ell_s \cdot \eta}{Cond_s^{\ell_s}(A)}$$

□

We can extend this result from reachability to formulas of the form $S_0 \mathbf{U} S_F$, where S_0, S_F are subsets of states. This formula means that we reach the set of states S_F through only states in S_0 on the way.

We define $R(S_0, S_F)$ to be the set of states which can reach S_F using only states of S_0, and $R_*(S_0, S_F) = R(S_0, S_F) \setminus S_F$. For $\ell \in \mathbb{N}$, we let:

$$\mathrm{Cond}_{S_0,S_F}^{\ell}(A) = \min_{i \in R_*(S_0,S_F)} \mathbb{P}_i^A(\mathbf{F}_{\leq \ell} \neg R_*(S_0, S_F) \vee \neg S_0).$$

Now, one can remark that $\mathrm{Cond}_{S_0,S_F}(A) \geq \mathrm{Cond}_{S,S_F}(A) > 0$. Let $\mathrm{Cond}_{S_F}^{\ell}(A) = \mathrm{Cond}_{S,S_F}^{\ell}(A)$. We have $\mathrm{Cond}_{S_0,S_F}^{\ell}(A) \geq \mathrm{Cond}_{S_F}^{\ell}(A)$. As before, we let $\ell_{S_F} \leq m$ be the minimal ℓ such that $\mathrm{Cond}_{S_F}^{\ell}(A) > 0$, and obtain:

Theorem 5. *Denoting φ the property $S_0 \mathbf{U} S_F$, we have, given DTMC A:*

$$|\gamma(A, \varphi) - \gamma(A_\eta, \varphi)| < \frac{\ell_{S_F} \cdot \eta}{Cond_{S_F}^{\ell_{S_F}}(A)}$$

We can actually improve this conditioning: we defined it as the probability to reach S_F or $S \setminus R(S, S_F)$. At the price of a more technical proof, we can obtain a better bound by replacing S_F by the set of states $R_1(S_F)$ that have probability 1 to reach S_F. We let $\overline{R_*}(S_F) = R(S, S_F) \setminus R_1(S_F)$ the set of states that can reach S_F with < 1 probability, and define the *refined conditioning* as follows:

$$\overline{\mathrm{Cond}}_{S_F}^{\ell}(A) = \min_{i \in \overline{R_*}(S_F)} \mathbb{P}_i^A(\mathbf{F}_{\leq \ell} \neg \overline{R_*}(S_F))$$

6.3 Optimality of the conditioning

We show now that the bound we provide in Theorem 4 is close to optimal.

Consider again DTMCs $A, \hat{A}$ in Fig. 3 from example 1, and formula $\mathbf{F}\, s_2$ stating that s_2 is eventually reached. The probabilities to satisfy this formula in $A, \hat{A}$ are respectively $\mathbb{P}^A(\mathbf{F}\, s_2) = \frac{1}{2}$ and $\mathbb{P}^{\hat{A}}(\mathbf{F}\, s_2) = \frac{1}{2} - \frac{\eta}{4\tau}$. Assume that A is the real system and that $\hat{A}$ is the DTMC we learned from A.

As we do not know precisely the transition probabilities in A, we can only compute the conditioning on $\hat{A}$ and not on A (it suffices to swap A and A_η in Theorem 4 and 5 to have the same formula using $\mathrm{Cond}(A_\eta) = \mathrm{Cond}(\hat{A})$). We have $R(s_2) = \{s_1, s_2\}$ and $R_*(s_2) = \overline{R_*}(s_2) = \{s_1\}$. The probability to stay in $R_*(s_2)$ after $\ell_{s_2} = 1$ step is $(1 - 2\tau)$, and thus $\mathrm{Cond}^1_{\{s_2\}}(\hat{A}) = \overline{\mathrm{Cond}}^1_{\{s_2\}}(\hat{A}) = 1-(1-2\tau) = 2\tau$. Taking $A_\eta = \hat{A}$, Theorem 5 tells us that $|\,\mathbb{P}^A(\mathbf{F}\, s_2) - \mathbb{P}^{\hat{A}}(\mathbf{F}\, s_2)| \leq \frac{\eta}{2\tau}$. Notice that on that example, using $\ell_{s_2} = m = 3$, we obtain $\mathrm{Cond}^3_{\{s_2\}}(\hat{A}) = 1 - (1 - 2\tau)^3 \approx 6\tau$, and we find a similar bound $\approx \frac{3\eta}{6\tau} = \frac{\eta}{2\tau}$.

Compare our bound with the exact difference $|\,\mathbb{P}^A(\mathbf{F}\, s_2) - \mathbb{P}^{\hat{A}}(\mathbf{F}\, s_2)| = \frac{1}{2} - (\frac{1}{2} - \frac{\eta}{4\tau}) = \frac{\eta}{4\tau}$. Our upper bound only has an overhead factor of 2, even while the conditioning is particularly bad (small) in this example.

6.4 PAC bounds for $\sum_j |\hat{\boldsymbol{A}}_{\boldsymbol{W}}(\boldsymbol{i},\boldsymbol{j}) - \boldsymbol{A}(\boldsymbol{i},\boldsymbol{j})| \leq \boldsymbol{\eta}$

We use Theorem 1 in order to obtain PAC bounds. We use it to estimate individual transition probabilities, rather than the probability of a property.

Let W be a set of traces drawn with respect to A such that every $\omega \in W$ is of the form $\omega = \rho \cdot s \cdot \rho' \cdot s$. Recall for each state i, j of S, n_i^W is the number of transitions originating from i in W and n_{ij}^W is the number of transitions ss' in W. Let $\delta' = \frac{\delta}{m_{\text{stoch}}}$, where m_{stoch} is the number of *stochastic* states, i.e., with at least two outgoing transitions.

We want to sample traces until the empirical transition probabilities $\frac{n_{ij}^W}{n_i^W}$ are relatively close to the exact transition probabilities a_{ij}, for all $i, j \in S$. For that, we need to determine a stopping criteria over the number of state occurrences $(n_i)_{1 \leq i \leq m}$ such that:

$$\mathbb{P}\left(\exists i \in S, \sum_j \left| a_{ij} - \frac{n_{ij}^W}{n_i^W} \right| > \varepsilon\right) \leq \delta$$

First, note that for any observed state $i \in S$, if $a_{ij} = 0$ (or $a_{ij} = 1$), then with probability 1, $\frac{n_{ij}^W}{n_i^W} = 0$ (respectively $\frac{n_{ij}^W}{n_i^W} = 1$). Thus, for all $\varepsilon > 0$, $|a_{ij} - \frac{n_{ij}^W}{n_i^W}| < \varepsilon$ with probability 1. Second, for two distinct states i and i', the transition probabilities $\frac{n_{ij}^W}{n_i^W}$ and $\frac{n_{i'j'}^W}{n_{i'}^W}$ are independent for all j, j'.

Let $i \in S$ be a stochastic state. If we observe n_i^W transitions from i such that $n_i^W \geq \frac{2}{\varepsilon^2} \log\left(\frac{2}{\delta'}\right) \left[\frac{1}{4} - \left(\max_j |\frac{1}{2} - \frac{n_{ij}^W}{n_i^W}| - \frac{2}{3}\varepsilon\right)^2\right]$, then, according to Theorem 1,

$\mathbb{P}\left(\bigvee_{j=1}^{m} |a_{ij} - \frac{n_{ij}^W}{n_i^W}| > \varepsilon\right) \leq \delta'$. In particular, $\mathbb{P}\left(\max_{j\in S} |a_{ij} - \frac{n_{ij}^W}{n_i^W}| > \varepsilon\right) \leq \delta'$. Moreover, we have:

$$\begin{aligned}\mathbb{P}\left(\bigvee_{j=1}^{m} \max_{j\in S} |a_{ij} - \frac{n_{ij}^W}{n_i^W}| > \varepsilon\right) &\leq \sum_{j=1}^{m} \mathbb{P}\left(\max_{j\in S} |a_{ij} - \frac{n_{ij}^W}{n_i^W}| > \varepsilon\right) \\ &\leq m_{\text{stoch}}\delta' \\ &\leq \delta\end{aligned}$$

In other words, the probability that "there exists a state $i \in S$ such that the deviation between the exact and empirical outgoing transitions from i exceeds ε" is bounded by δ as soon as for each state $i \in S$, n_i^W satisfies the stopping rule of the algorithm of Chen using ε and the corresponding δ'. This gives the hypothesis $\sum_j |A_\eta(i,j) - A(i,j)| \leq \epsilon$ for all states i of Section 6.2.

6.5 A Matrix $\hat{A}_W$ accurate for all CTL properties

We now use Laplace smoothing in order to ensure the other hypothesis $A_\eta(i,j) \neq 0$ iff $A(i,j) \neq 0$ for all states i, j. For all $i \in S$, we define the Laplace offset depending on the state i as $\alpha_i = \frac{(n_i^W)^2\varepsilon}{10\cdot k_i^2 \max_j n_{ij}^W}$, where k_i is the number of transitions from state i. This ensures that the error from Laplace smoothing is at most one tenth of the statistical error. Let $\alpha = (\alpha_i)_{1\leq i\leq m}$. From the sample set W, we output the matrix $\hat{A}_W^\alpha = (\hat{a}_{ij})_{1\leq i,j\leq m}$ with Laplace smoothing α_i for state i, i.e.:

$$\hat{a}_{ij} = \frac{n_{ij}^W + \alpha_i}{n_i^W + k_i\alpha_i} \text{ if } a_{ij} \neq 0 \quad \text{and} \quad \hat{a}_{ij} = 0 \text{ otherwise}$$

It is easy to check that we have for all $i, j \in S$: $\left|\hat{a}_{ij} - \frac{n_{ij}^W}{n_i^W}\right| \leq \frac{\varepsilon}{10\cdot k_i}$

That is, for all states i, $\sum_j \left|\hat{a}_{ij} - \frac{n_{ij}^W}{n_i^W}\right| \leq \frac{\varepsilon}{10}$. Using the triangular inequality:

$$\mathbb{P}\left(\exists i \in S, \sum_j |a_{ij} - \hat{a}_{ij}| > \frac{11}{10}\varepsilon\right) \leq \delta$$

For all $i \in S$, let $H^*(n_i^W, \epsilon, \delta') = \max_{j\in S} H(n_i^W, n_{ij}^W, \epsilon, \delta')$ be the maximal Chen bound over all the transitions from state i. Let $B(\hat{A}_W^\alpha) = \max_{S_F} \frac{\ell_{S_F}}{\text{Cond}_{S_F}^{\ell_{S_F}}(\hat{A}_W^\alpha)}$. Since in Theorem 5, the original model and the learned one have symmetric roles, by applying this theorem on $\hat{A}_W^\alpha$, we obtain that:

Theorem 6. *Given a set W of traces, for $0 < \epsilon < 1$ and $0 < \delta < 1$, if for all $i \in S$, $n_i^W \geq \left(\frac{11}{10}B(\hat{A}_W^\alpha)\right)^2 H^*(n_i^W, \epsilon, \delta')$, we have for any CTL property φ:*

$$\mathbb{P}(|\gamma(A,\varphi) - \gamma(\hat{A}_W^\alpha, \varphi)|) > \varepsilon) \leq \delta \tag{9}$$

Proof. First, $\hat{a}_{ij} \neq 0$ iff $a_{ij} \neq 0$, by definition of $\hat{A}^{\alpha}_{W}$. Second, $\mathbb{P}(\exists i, \sum_j |a_{ij} - \hat{a}_{ij}| > \frac{11}{10}\varepsilon) \leq \delta$. We can thus apply Theorem 5 on $\hat{A}^{\alpha}_{W}, A$ and obtain (9) for φ any formula of the form $S_1 \mathbf{U} S_2$. It remains to show that for any formula $\varphi \in \Psi$, we can define $S_1, S_2 \subseteq S$ such that φ can be expressed as $S_1 \mathbf{U} S_2$.

Consider the different cases: If φ is of the form $\varphi = \varphi_1 \mathbf{U} \varphi_2$ (it subsumes the case $\varphi = \mathbf{F}\varphi_1 = \top \mathbf{U} \varphi_1$) with φ_1, φ_2 CTL formulas, we define S_1, S_2 as the sets of states satisfying φ_1 and φ_2, and we have the equivalence (see [2] for more details). If $\varphi = X\varphi_2$, define $S_1 = \emptyset$ and S_2 as the set of states satisfying φ_2.

The last case is $\varphi = \mathbf{G}\varphi_1$, with φ_1 a CTL formula. Again, we define S_1 the set of states satisfying φ_1, and S_2 the set of states satisfying the CTL formula $\mathbf{AG}\varphi_1$. The probability of the set of paths satisfying $\varphi = \mathbf{G}\varphi_1$ is exactly the same as the probability of the set of paths satisfying $S_1 \mathbf{U} S_2$. □

6.6 Algorithm

We give more details about the learning process of a Markov Chain, accurate for every CTL formula. For completeness, we also provide in the extended versiona similar algorithm for a time-to-failure property.

A path ω is observed from s_0 till a state is observed twice. Then ω is added to W and the reset operation is performed. We use Laplace smoothing to compute the corresponding matrix $\hat{A}^{\alpha}_{W}$. The error bound is computed on W, and a new path ω' is then being generated if the error bound is not as small as desired.

This algorithm is guaranteed to terminate since, as traces are generated, with probability 1, n^W_s tends towards ∞, $\hat{A}^{\alpha}_{W}$ tends towards A, and $B(\hat{A}^{\alpha}_{W})$ tends towards $B(A)$.

Algorithm 1: Learning a matrix accurate for CTL

Data:
$\mathcal{S}, s_0, \delta, \varepsilon$

$W := \emptyset$
$m = |S|$
for all $s \in S$, $n^W_s := 0$
Compute $\hat{A} := \hat{A}^{\alpha}_{W}$
Compute $B := B(\hat{A})$
while $\exists s \in S, n^W_s < \left(\frac{11}{10} B(\hat{A})\right)^2 H^*(n^W_s, \epsilon, \frac{\delta}{m})$ **do**
    Generate a new trace $\omega := s_0\, \rho\, s_1\, \rho'\, s_1$, and reset $\mathcal{S}$
    for all $s \in S$, $n^W_s := n^W_s + n^{\{\omega\}}_s$
    add ω to W
    Compute $\hat{A} := \hat{A}^{\alpha}_{W}$
    Compute $B := B(\hat{A})$

Output: $\hat{A}^{\alpha}_{W}$

7 Evaluation and Discussion

In this section, we first evaluate Algorithm 1 on 5 systems which are crafted to evaluate the algorithm under different conditions (e.g., rare states). The objective of the evaluation is to provide some idea on how many samples would be sufficient for learning accurate DTMC estimations, and compare learning for all properties of CTL and learning for one time-to-failure property.

Then, we evaluate our algorithm on very large PRISM systems (millions or billions of states). Because of the number of states, we cannot learn a DTMC accurate for all properties of CTL there: it would ask to visit every single state a number of times. However, we can learn a DTMC for one specific (unbounded) property. We compare with an hypothesis testing algorithm from [30] which can handle the same unbounded property through a reachability analysis using the topology of the system.

7.1 Evaluation on crafted models

We first describe the 5 systems: Systems 1 and 2 are three-state models described in Fig. 1 and Fig. 2. Systems 3 (resp. 5) is a 30-state (resp. 200-states) clique in which every individual transition probability is 1/30 (resp. 1/200). System 4 is a 64-state system modeling failure and repair of 3 types of components (3 components each, 9 components in total), see the extended versionfor a full description of the system, including a PRISM [20] model for the readers interested to investigate this system in details.

We tested time-to-failure properties by choosing as failure states s_3 for Systems 1, 2, 3, 5, and the state where all 9 components fail for System 4. We also tested Algorithm 1 (for full CTL logic) using the refined conditioning $\overline{\text{Cond}}$. We performed our algorithms 100 times for each model, except for full CTL on System 4, for which we only tested once since it is very time-consuming. We report our results in Table 1 for $\varepsilon = 0.1$ and $\delta = 0.05$. In particular, we output for

	System 1	System 2	System 3	System 4	System 5
# states	3	3	30	64	200
# transitions	4	7	900	204	40,000
# events for time-to-failure	191 (16%)	991 (10%)	2,753 (7.4%)	1,386 (17.9%)	18,335 (7.2%)
# events for full CTL	1,463 (12.9%)	4,159 (11.7%)	8,404 (3.8%)	1,872,863	79,823 (1.7%)

Table 1: Average number of observed events N (and relative standard deviation in parenthesis) given $\varepsilon = 0.1$ and $\delta = 0.05$ for a time-to-failure property and for the full CTL logic using the refined conditioning $\overline{\text{Cond}}$.

each model its number of states and transitions. For each (set of) property, we provide the average number of observations (i.e. the number of samples times their average length) and the relative standard deviation (in parenthesis, that is the standard deviation divided by the average number of observed events).

The results show that we can learn a DTMC with more than 40000 stochastic transitions, such that the DTMC is accurate for all CTL formulas. Notice that for some particular systems such as System 4, it can take a lot of events to be observed before Algorithm 1 terminates. The reason is the presence of rare states, such as the state where all 9 components fail, which are observed with an extremely small probability. In order to evaluate the probabilities of CTL properties of the form: "if all 9 components fail, then CTL property φ is satisfied", this state needs to be explored many times, explaining the high number of events observed before the algorithm terminates. On the other hand, for properties that do not involve the 9 components failing as prior, such as time-to-failure, one does not need to observe this state even once to conclude that it has an extremely small probability to happen. This suggests that efficient algorithms could be developed for subsets of CTL formulas, e.g., in defining a subset of important events to consider. We believe that Theorem 4 and 5 could be extended to handle such cases. Over different runs, the results stay similar (notice the rather small relative standard deviation).

Comparing results for time-to-failure (or equivalently SMC) and for the full CTL logic is interesting. Excluding System 4 which involves rare states, the number of events that needs to be observed for the full CTL logic is 4.3 to 7 times more. Surprisingly, the highest difference is obtained on the smallest System 1. It is because every run of System 1 generated for time-to-failure is short ($s_1 s_2 s_1$ and $s_1 s_2 s_3$). However, in Systems 2,3 and 5, samples for time-to-failure can be much longer, and the performances for time-to-failure (or equivalently SMC) is not so much better than for learning a DTMC accurate for all CTL properties.

For the systems we tested, the unoptimized Cond was particularly large (more than 20) because for many states s, there was probability 0 to leave $R(s)$, and hence $\ell(s)$ was quite large. These are the cases where $\overline{\text{Cond}}$ is much more efficient, as then we can choose $\ell_s = 1$ as the probability to reach s from states in $R(s)$ is 1 ($R_1(s) = R(s)$ and $\overline{R_*(s)} = \emptyset$). We used $\overline{\text{Cond}}$ in our algorithm.

Finally, we evaluate experimental confidence by comparing the time-to-failure probabilities in the learned DTMC and the original system. We repeat our algorithms 1000 times on System 1 and 2 (with $\varepsilon = 0.1$ and $\delta = 0.05$). These probabilities differ by less than ε, respectively 999 and 995 times out of 1000. Specification (2) is thus largely fulfilled (the specification should be ensured 950 out of 1000 times), that empirically endorses our approach. Hence, while our PAC bound over-approximates the confidence in the learned system (which is unavoidable), it is not that far from experimental values.

7.2 Evaluation on large models

We also evaluated our algorithm on large PRISM models, ranging from hundreds of thousands to billions of states. With these numbers of states, we cannot use

the more ambitious learning over all the properties of CTL, which would need to visit every states a number of times. However, we can use our algorithm for learning a DTMC which is accurate given a particular (unbounded) property: it will visit only a fraction of the states, which is enough to give a model accurate for that property, with a well-learned kernel of states and some other states representatives for the remaining of the runs. We consider three test-cases from PRISM, satisfying the property that the sample stops with a conclusion (yes or no) with probability 1. Namely, *herman, leader* and *egl*.

Our prototype tool used in the previous subsection is implemented in Scilab: it cannot simulate very large systems of PRISM. Instead, we use PRISM to generate the samples needed for the learning. Hence, we report the usual Okamoto-Chernoff bound on the number of samples, which is what is implemented in PRISM. We also compare with the Massart bound used by the Chen algorithm (see Section 2.2), which is implemented in our tool and is more efficient as it takes into account the probability of the property.

For each model, we report its parameters, its *size*, i.e. its number of states, the number of *samples* needed using the Massart bound (the conservative Okamoto-Chernoff bound is in parenthesis), and the average *path length*. For comparison, we consider an hypothesis testing algorithm from [30] which can also handle unbounded properties. It uses the knowledge of the topology to do reachabil-

		our learning method		sampling with reachability analysis [30]	
Model name	size	samples	path length	samples	path length
herman(17)	129M	506 (38K)	27	219	30
herman(19)	1162M	506 (38K)	40	219	38
herman(21)	10G	506 (38K)	43	219	48
leader(6,6)	280K	506 (38K)	7.4	219	7
leader(6,8)	$>$ 280K	506 (38K)	7.4	(MO)	(MO)
leader(6,11)	$>$ 280K	506 (38K)	7.3	(MO)	(MO)
egl(15,10)	616G	38K (38K)	470	1100	201
egl(20,15)	1279T	38K (38K)	930	999	347
egl(20,20)	1719T	38K (38K)	1200	(TO)	(TO)

Table 2: Results for $\varepsilon = 0.01$ and $\delta = 0.001$ of our algorithm compared with sampling with reachability analysis [30], as reported in [14], page 20. Numbers of samples needed by our method are given by the Massart bound (resp. by the Okamoto-Chernoff bound in parenthesis). TO and MO means time out ($>$ 15 minutes on an Opteron 6134) and memory out ($>$ 5GB) respectively.

ity analysis to stop the sampling if the property cannot be reached anymore. Hypothesis testing is used to decide with high confidence whether a probability exceeds a threshold or not. This requires less samples than SMC algorithms which estimate probabilities, but it is also less precise. We chose to compare with this algorithm because as in our work, it does not require knowledge on the probabilities, such as a lower bound on the transition probabilities needed by e.g. [14]. We do not report runtime as they cannot be compared (different platforms, different nature of result, etc.).

There are several conclusions we can draw from the experimental results (shown in Table 2). First, the number of samples from our algorithm (Chen algorithm implementing the Massart bound) are larger than in the algorithm from [30]. This is because they do hypothesis testing, which requires less samples than even estimating the probability of a property, while we learn a DTMC accurate for this property. For *herman* and *leader*, the difference is small (2.5x), because it is a case where the Massart bound is very efficient (80 times better than Okamoto-Chernoff implemented in PRISM). The *egl* system is the worst-case for the Massart bound (the probability of the property is $\frac{1}{2}$), and it coincides with Okamoto-Chernoff. The difference with [30] is 40x in that case. Also, as shown in *egl*, paths in our algorithm can be a bit larger than in the algorithm from [30], where they can be stopped early by the reachability analysis. However, the differences are never larger than 3x. On the other hand, we learn a model representative of the original system for a given property, while [30] only provide a yes/no answer to hypothesis testing (performing SMC evaluating the probability of a property with the Massart bound would give exactly the same number of samples as we report for our learning algorithm). Last, the reachability analysis from [30] does time out or memory out on some complex systems, which is not the case with our algorithm.

8 Conclusion

In this paper, we provided theoretical grounds for obtaining global PAC bounds when learning a DTMC: we bound the error made between the behaviors of the model and of the system, formalized using temporal logics. While it is not possible to obtain a learning framework for LTL properties, we provide it for the whole CTL logic. For subsets of CTL, e.g. for a fixed timed-to-failure property, we obtain better bounds, as efficient as Statistical MC. Overall, this work should help in the recent trends of establishing trusted machine learning [16].

Our techniques are useful for designers of systems for which probabilities are governed by uncertain forces (e.g. error rates): in this case, it is not easy to have a lower bound on the minimal transition probability, but we can assume that the set of transitions is known. Technically, our techniques provides rationale to set the constant in Laplace smoothing, otherwise left to an expert to set.

Some cases remain problematic, such as systems where states are visited very rarely. Nevertheless, we foresee potential solutions involving rare event simulation [21]. This goes beyond the scope of this work and it is left to future work.

References

1. Pranav Ashok, Jan Kretínský, and Maximilian Weininger. PAC statistical model checking for markov decision processes and stochastic games. In *Computer Aided Verification - 31st International Conference, CAV 2019, New York City, NY, USA, July 15-18, 2019, Proceedings, Part I*, volume 11561 of *Lecture Notes in Computer Science*, pages 497–519. Springer, 2019.
2. Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.
3. Luca Bortolussi and Guido Sanguinetti. Learning and Designing Stochastic Processes from Logical Constraints. In *Quantitative Evaluation of Systems - 10th International Conference, QEST, Buenos Aires, Argentina*, pages 89–105, 2013.
4. Manuele Brambilla, Carlo Pinciroli, Mauro Birattari, and Marco Dorigo. Property-driven design for swarm robotics. In *International Conference on Autonomous Agents and Multiagent Systems, AAMAS, Valencia, Spain*, pages 139–146, 2012.
5. Jorge Castro and Ricard Gavaldà. Towards Feasible PAC-Learning of Probabilistic Deterministic Finite Automata. In *Grammatical Inference: Algorithms and Applications, 9th International Colloquium, ICGI, Saint-Malo, France*, pages 163–174, 2008.
6. Jianhua Chen. Properties of a New Adaptive Sampling Method with Applications to Scalable Learning. In *Web Intelligence, Atlanta*, pages 9–15, 2013.
7. Stanley F. Chen and Joshua Goodman. An Empirical Study of Smoothing Techniques for Language Modeling. *Computer Speech and Language*, 13(4):359–394, 1999.
8. Yingke Chen, Hua Mao, Manfred Jaeger, Thomas Dyhre Nielsen, Kim Guldstrand Larsen, and Brian Nielsen. Learning Markov Models for Stationary System Behaviors. In *NASA Formal Methods - 4th International Symposium, NFM, Norfolk, VA, USA*, pages 216–230, 2012.
9. H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Ann. Math. Statist.*, 23(4):493–507, 1952.
10. Alexander Clark and Franck Thollard. PAC-learnability of Probabilistic Deterministic Finite State Automata. *Journal of Machine Learning Research*, 5:473–497, 2004.
11. Edmund M. Clarke and E. Allen Emerson. Design and Synthesis of Synchronization Skeletons Using Branching-Time Temporal Logic. In *Logics of Programs, Workshop, Yorktown Heights, New York, USA, May 1981*, pages 52–71, 1981.
12. William G. Cochran. *Contributions to Survey Sampling and Applied Statistics*, chapter Laplace's ratio estimator, pages 3–10. Academic Press, New York, 1978.
13. Przemyslaw Daca, Thomas A. Henzinger, Jan Kretínský, and Tatjana Petrov. Linear Distances between Markov Chains. In *27th International Conference on Concurrency Theory, CONCUR 2016, August 23-26, 2016, Québec City, Canada*, pages 20:1–20:15, 2016.
14. Przemyslaw Daca, Thomas A. Henzinger, Jan Kretínský, and Tatjana Petrov. Faster Statistical Model Checking for Unbounded Temporal Properties. *ACM Trans. Comput. Log.*, 18(2):12:1–12:25, 2017.
15. William A. Gale and Geoffrey Sampson. Good-Turing Frequency Estimation Without Tears. *Journal of Quantitative Linguistics*, pages 217–37, 1995.
16. Shalini Ghosh, Patrick Lincoln, Ashish Tiwari, and Xiaojin Zhu. Trusted Machine Learning: Model Repair and Data Repair for Probabilistic Models. In *AAAI-17 Workshop on Symbolic Inference and Optimization*, 2017.

17. T. Hérault, R. Lassaigne, F. Magniette, and S. Peyronnet. Approximate Probabilistic Model Checking. In *VMCAI*, volume 2937 of *LNCS*, pages 307–329, 2004.
18. Cyrille Jégourel, Jun Sun, and Jin Song Dong. Sequential Schemes for Frequentist Estimation of Properties in Statistical Model Checking. In *Quantitative Evaluation of Systems - 14th International Conference, QEST, Berlin, Germany*, pages 333–350, 2017.
19. Solomon Kullback and Richard A. Leibler. On Information and Sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.
20. M. Z. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *CAV*, pages 585–591, 2011.
21. Axel Legay, Sean Sedwards, and Louis-Marie Traonouez. Rare Events for Statistical Model Checking an Overview. In *Reachability Problems - 10th International Workshop, RP, Aalborg, Denmark*, pages 23–35, 2016.
22. P. Massart. The tight constant in the Dvoretzky-Kiefer-Wolfowitz inequality. *The Annals of Probability*, 18:1269–1283, 1990.
23. Masashi Okamoto. Some Inequalities Relating to the Partial Sum of Binomial Probabilities. *Annals of the Institute of Statistical Mathematics*, 10:29–35, 1958.
24. Amir Pnueli. The Temporal Logic of Programs. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA*, pages 46–57, 1977.
25. Ad Ridder. Importance Sampling Simulations of Markovian Reliability Systems Using Cross-Entropy. *Annals OR*, 134(1):119–136, 2005.
26. Dorsa Sadigh, K. Driggs-Campbell, A. Puggelli, W. Li, V. Shia, R. Bajcsy, A. Sangiovanni-Vincentelli, S. Sastry, and S. Seshia. Data-driven probabilistic modeling and verification of human driver behavior. In *In Formal Verification and Modeling in Human-Machine Systems - AAAI Spring Symposium*, 2014.
27. Chris Sherlaw-Johnson, Steve Gallivan, and Jim Burridge. Estimating a Markov Transition Matrix from Observational Data. *The Journal of the Operational Research Society*, 46(3):405–410, 1995.
28. Leslie G. Valiant. A Theory of the Learnable. *Commun. ACM*, 27(11):1134–1142, 1984.
29. Abraham Wald. Sequential tests of statistical hypotheses. *The Annals of Mathematical Statistics*, 16(2):117–186, 1945.
30. Jingyi Wang, Jun Sun, Qixia Yuan, and Jun Pang. Should We Learn Probabilistic Models for Model Checking? A New Approach and An Empirical Study. In *Fundamental Approaches to Software Engineering - 20th International Conference, FASE, Uppsala, Sweden*, pages 3–21, 2017.
31. Håkan L. S. Younes, Edmund M. Clarke, and Paolo Zuliani. Statistical verification of probabilistic properties with unbounded until. In *SBMF'10*, pages 144–160, 2010.
32. Håkan L. S. Younes and Reid G. Simmons. Probabilistic verification of discrete event systems using acceptance sampling. In *Computer Aided Verification, 14th International Conference, CAV ,Copenhagen, Denmark*, pages 223–235, 2002.
33. P. Zuliani, A. Platzer, and E. M. Clarke. Bayesian Statistical Model Checking with Application to Stateflow/Simulink Verification. *FMSD*, 43(2):338–367, 2013.