Improving Probability Estimation through Active Probabilistic Model Learning

Jingyi Wang[1], Xiaohong Chen[1,2], Sun Jun[1], Shengchao Qin[3]

[1]Singapore University of Technology and Design
[2]The University of Illinois at Urbana-Champaign
[3]Teesside University

Abstract. It is often necessary to estimate the probability of certain events occurring in a given system. For instance, knowing the probability of events triggering a shutdown sequence in a system allows us to estimate the availability of the system. One approach is to run the system multiple times and then construct a probabilistic model to estimate the probability. When the probability of the event to be estimated is low, many system runs are necessary in order to generate an accurate estimation. For complex cyber-physical systems, each system run is costly and time-consuming, and thus it is important to reduce the number of system runs while providing accurate estimation. In this work, we assume that the user can actively tune the initial configuration of the system before the system runs and answer the following research question: how should the user set the initial configuration so that a better estimation can be learned with fewer system runs. The proposed approach has been implemented and evaluated with a set of benchmark models, random generated models, and a real-world water treatment system.

1 Introduction

It is often necessary to estimate the probability of certain events occurring in a given system. In the following, we describe a real-world scenario where such a task arises. The SWaT testbed[1] at Singapore University of Technology and Design (SUTD) is a complex water treatment system that consists of multiple phases including filtering and chemical dosing, etc. The system is safety critical and has built-in monitors that check violation of safety requirements. For instance, water-level monitors are put in place to check whether the level of water in tanks are too low or too high. Whenever a monitor issues a safety alarm, a shutdown sequence is triggered so that the system is brought to a halt and expert engineers are called upon to inspect the system. Such a design guarantees that safety violation is detected at runtime, at the cost of potentially shutting the system down from time to time. To show that the system satisfies certain availability requirements, we would like to show that the likelihood of triggering the shutdown sequence is low, i.e., the probability of shutdown triggering events occurring is below a certain threshold.

One way to solve the problem is to run the system multiple times, observe how the system evolves through time, construct a probabilistic model of the system (i.e., a

[1] http://itrust.sutd.edu.sg/research/testbeds/secure-water-treatment-swat/

discrete-time Markov Chain) and estimate the probability of the interested events based on the model. To observe how the system evolves at runtime, we can introduce a logger in the system to record the system state, e.g., to log the sensor reading of the water level and the status of the valves and pumps in the system. To construct the Markov Chain model, we can apply an estimation function to estimate the transition probability between system states. Commonly used estimators include empirical frequency, Laplace estimator [9] and Good-Turing estimator [10]. To estimate the probability of the interested events occurring, we additionally need an initial probability distribution, i.e., the probability of having certain initial configuration of the system (e.g., the initial water level of the tanks and the status of the actuators), which we can often obtain either through historical data or expert experience. When we run the system for multiple times, the same initial distribution is applied to configure the initial system state accordingly.

Such a method however may not be effective if the interested events have low probability. For instance, some events may only be triggered under certain particular initial configurations. For instance, the event of water underflow may only occur when the initial water level is set to be near the boundary and the water valve is set to drain the water. If there are a large number of possible initial configurations and these particular initial configurations have low probability according to the initial distribution, it would take many system runs so that we can trigger the events for a sufficient number of times and estimate their probability accurately. However, conducting an experiment to run a system like the water treatment system (or other real-world cyber-physical systems) often has non-negligible cost. Thus, it is desirable to reduce the number of system runs while being able to accurately estimate the transition probability based on which we compute the probability of the interested events.

In this work, we propose to smartly configure the system initially so that during the system runs, "interesting" system transitions are more likely to be triggered (than configuring the system according to the originally given initial distribution). Our idea is to first get an initial estimation of the transition probabilities, based on which we calculate an 'optimal' initial distribution which we should follow to conduct further experiments. Intuitively, an initial distribution is considered optimal if the estimation of the transition probabilities based on the experiment results according to the initial distribution is more accurate than other initial distributions. Afterwards, we run the system multiple times according to the optimal initial distribution and update the estimation of the transition probabilities accordingly. We repeat the process until some stopping criteria are satisfied.

Our method can be viewed as an active learning method for Markov Chain models [5], which are useful in modeling and analyzing a wide range of systems. Our method is designed to learn a Markov Chain model actively in a particular setting. That is, we assume that a prior initial distribution is given, and we are allowed to tune the initial probability distribution but not the transition probability distributions. To the best of our knowledge, our method is the first active learning method for Markov Chain models. Our method is not restricted to one particular way of estimating transition probability. We show that our method works for common estimation techniques like empirical frequency, Laplace estimator and Good-Turing estimator. In order to evaluate the

effectiveness of our approach, we implemented a prototype tool in Java called IDO. We set up experiments to compare IDO with alternative approaches. The experiment results show that IDO always estimates more accurately with the same number of system runs, or requires fewer system runs to achieve the same level of accuracy. Our test subjects include several benchmark systems, a set of randomly generated models, and the SWaT testbed mentioned above.

2 Problem Definition

We will formally state the problem that we consider in this paper upon discrete-time Markov chains (DTMCs), a widely-used formalization that models probabilistic transition system with finite states [20]. Before that, we will present a succinct review of DTMCs and introduce our notations.

2.1 The Model

Definition 1. *A discrete-time Markov Chain (DTMC) is a tuple $\mathcal{M} = (S, S_0, P, \mu)$ with a finite nonempty set of states S, a nonempty subset of initial states $S_0 \subseteq S$, a transition matrix $P : S \times S \to [0, 1]$, and an initial probability distribution μ over initial states. A path is a nonempty sequence of states starting with an initial state.*

Note that different from the standard definition, we distinguish a set of initial states from the rest and constrain that the initial distribution μ only assign probabilities to initial states. The value $P(s, s')$ (where $s, s' \in S$) is the conditional probability of visiting s' given the current state is s. When the set of states S is indexed or enumerated, which is often the case, we write p_{ij} for the probability $P(s_i, s_j)$, where $s_i, s_j \in S$. That is the reason why $P = (p_{ij})$ is called transition *matrix*. Given a path $i_1 \dots i_k$, the probability of observing that path is $\mu_{i_1} p_{i_1 i_2} \dots p_{i_{k-1} i_k}$, denoted[2] $\mathbf{P}(i_1 \dots i_k)$, which depends not only on the transition matrix P but also on the initial distribution μ.

Figure 1 shows a simple DTMC with four states and two initial states. Transition probabilities are labeled upon arrows between states, and the initial distribution is attached by the label "start". By convention, not drawing an arrow means the corresponding transition probability is zero.

2.2 The Problem

The problem we investigate in this work can be defined as follows. Given a system that is modeled as a DTMC with a fixed set of states and an initial probability distribution, how to estimate (A) its transition matrix and (B) the probability of reaching certain states? In this paper, we assume the transition matrix is fixed and an unchanged one, but we can try different initial configurations when running the system. In our SWaT testbed [1], for example, we can set different levels of water in tanks (and/or other configurations such as the initial pH value of the water) before we run the system, and

[2] We use P to denote transition matrices and $\mathbf{P}$ to denote probability measure.

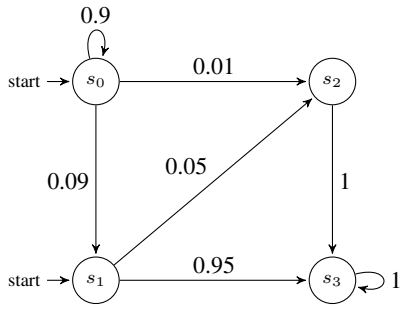

Fig. 1: An example DTMC

once the system is turned on, we can only observe how the system evolves through time, but not influence how it evolves.

Contrary to our approach, a *passive* approach to solve the problem is to set the initial configuration of the system as the given initial distribution μ and run the system multiple times. After a couple times of experiments, transition probability P can be estimated (and subsequently the probability of reaching certain state). That is, an initial state s_1 is generated randomly according to μ, configure the system so that it starts with state s_1, let the system evolve according to the transition probability P; obtain a new state s_2, and so on until a bounded number of steps are taken. We write $\pi \sim (\mathcal{M}, \mu)$ to denote that π is a sample obtained this way with initial distribution μ. Let $\mathit{\Pi}$ be a set of paths of $\mathcal{M}$. We write $\mathit{\Pi} \sim (\mathcal{M}, \mu)$ to denote that $\mathit{\Pi}$ is a set of samples obtained with the initial probability distribution μ.

Once a set of samples $\mathit{\Pi} \sim (\mathcal{M}, \mu)$ is obtained, an estimation function (a.k.a. an estimator) could be applied to generate an estimation $\widehat{P}$ of the transition matrix P. In the following, we briefly introduce three different estimators which are commonly used.

Definition 2. *Let $\mathit{\Pi} \sim (\mathcal{M}, \mu)$ be a set of samples. Given any state s, let $\#s$ be the number of times state s is visited by samples in $\mathit{\Pi}$. For a pair of states s and t, let $\#(s,t)$ be the number of one-step transition from state s to t in $\mathit{\Pi}$.*

- *The empirical frequency estimator sets $P(s,t)$ to be $\#(s,t)/\#s$;*
- *The Laplace estimator [9] sets $P(s,t)$ to be $(1+\#(s,t))/(n+\#s)$, where n is the total number of states in $\mathcal{M}$;*
- *The Good-Turing estimator [10] sets $P(s,t)$ to be $\frac{(\#(s,t)+1)\times N_{\#(s,t)+1}}{\#s \times N_{\#(s,t)}}$ where $N_r = |\{t \in S | \#(s,t) = r\}|$ is the number of states which are visited after s exactly r times in $\mathit{\Pi}$.*

The empirical frequency estimator estimates the transition probability based on the frequency. It may be problematic if the system contains transitions with low probability. That is, if a transition from state s to t is not observed in $\mathit{\Pi}$ because the actual $P(s,t)$ is small, $P(s,t)$ is estimated to be zero by the empirical frequency estimator. The Laplace estimator overcomes this problem by adding a constant 1 to both the numerator and denominator of the estimated transition probability. In other words, if state t is never visited after state s, the probability $P(s,t)$ is estimated to be $1/(n+\#s)$. The Good-Turing

estimator is widely used when the amount of samples is relatively small compared to the number of states. We skip the discussion on how the Good-Turing estimator works intuitively and refer the readers to [10] for comprehensive discussion on when different estimators are effective. Once an estimation of P is obtained, we can calculate the probability of reaching certain state straightforwardly using methods like probabilistic model checking [5].

All the above-mentioned estimators guarantee that they converge to an accurate estimate of P with an unbounded number of samples. It might however take a large number of samples in order to obtain an accurate estimate of P. In practice, we may not be able to run a complex system (like the SWaT tested) for many times since each run has non-negligible cost in terms of money and time. In this work, we aim to develop a method which allows us to reduce the number of samples required to generate an accurate estimate of P. Ideally, it should work with any of the above-mentioned estimators. In particular, the question is: if a user is allowed to tune the initial distribution (e.g., by initializing the system using a probability distribution of initial configuration/inputs different from μ_0), how should she/he tune the initial distribution so that we can estimate P more effectively? Note that we assume that we can neither change the transition probability nor assigning non-zero probability to non-initial states. We believe this is a realistic assumption. Consider for example the above-mentioned SWaT test bed. The user can decide to perform multiple experiments by choosing a set of initial configurations following certain distribution and simply observe how the system evolves afterwards.

3 Our approach

Our approach is inspired and built on the idea of "active learning", one that has been studied extensively in automaton learning (e.g., [4]) and classifier learning (e.g., [7]). The basic idea is to sample smartly, based on the current estimation $\widehat{P}$ of the transition matrix, so as to obtain informative samples that effectively improves the estimation.

The overall algorithm is shown in Algorithm 1. Initially, since we do not have any knowledge of P, we obtain samples of the system with the user-provided initial distribution μ. After obtaining some number of samples, we apply an estimator to obtain an estimate $\widehat{P}$ at line 4. Based on $\widehat{P}$, we compute an "optimal" initial distribution with respects to our objective at line 5. Then, we repeat from line 3, i.e., acquire more samples based on μ, and apply the estimator again to obtain an updated estimation of $\widehat{P}$. The process continues until a stopping criteria is satisfied. This approach is inspired by the expectation-maximization (EM) algorithm from statistics [16].

3.1 Estimating Transition Probability

In order to identify the "optimal" initial distribution, we must firstly identify our analysis objective. In this work, our overall objective is to estimate the transition probability and reachability probability. In the following, we first focus on estimating the transition matrix P. The accuracy of an estimation $\widehat{P}$ of P can be measured using measurements

Algorithm 1: act

1: Let μ_0 be μ;
2: **while** not time out **do**
3: Sample the system to obtain $\Pi \sim (\mathcal{M}, \mu_0)$;
4: Apply an estimator to obtain $\widehat{P}$ based on Π;
5: Set μ_0 to be the optimal initial distribution computed based on $\widehat{P}$;
6: **end while**
7: Output $\widehat{P}$;

like mean squared error (MSE), the standard deviation or bias [3]. As an example, the following shows the definition of MSE.

$$MSE(\widehat{P}, P) = \frac{1}{(\#S)^2} \sum_{s,t \in S} (\widehat{P}(s,t) - P(s,t))^2$$

Ideally, we would like to identify an initial distribution μ_0 such that the estimation would be most accurate. However, since P is unknown, we cannot directly compare the estimation $\widehat{P}$ and P (e.g., in term of MSE). We thus need to define an alternative optimization objective. The optimization objective is important as it should guarantee that not only will we eventually learn P accurately, but also we will do it in a more effective way than sampling according to μ. Intuitively, a sample is most useful in improving our estimation if it can help eliminate most uncertainty in our current learning result. In general, if a state is rarely visited by the training samples, the estimation of the transition probability from this state is likely to be inaccurate, whereas transition probabilities from a state which is often visited is likely to be estimated more accurately. For instance, given the DTMC in Figure 1, it is hard to estimate the probability of transitioning from state s_1 to s_3 if a limited number of samples are available and s_1 is visited only a few times. Based on this observation, the following optimization objective is adopted.

$$\max_{\mu_0 \in \mathcal{D}} \min_{s \in S} E(s, \mu_0, \widehat{P}, N, k) \tag{3.1}$$

where $\mathcal{D}$ is the set of all initial distributions that only assigns non-zero probability to initial states; and $E(s, \mu_0, \widehat{P}, N, k)$ is the expected number of times a state s is visited if we sample N paths (each of which with k transitions) according to the initial distribution μ_0 and the transition matrix $\widehat{P}$. It is defined as follows.

$$E(s, \mu_0, \widehat{P}, N, k) = N \left(\mu_0(s) + \mu_0 \widehat{P}(s) + \mu_0 \widehat{P}^2(s) + \cdots + \mu_0 \widehat{P}^k(s) \right)$$

where $\mu_0 \widehat{P}^l(s)$ is the probability of visiting state s after l transitions. Intuitively, we would like to identify an initial distribution μ_0 so as to maximize the probability of visiting the least likely state to be visited within k steps.

Applying standard optimization techniques [2, 11], we can identify the 'optimal' initial distribution μ_0 based on the above optimization objective. In the following, we discuss why the above objective works. In particular, we show that it guarantees we

would always converge to an accurate estimation of P and our estimation $\widehat{P}$ monotonically improves, no matter which of the three above-mentioned estimators are used.

In the following, we introduce some notations that will be frequently used below. We use the norm $\|\cdot\|$ that is defined as $\|A\| = \max_{i,j} |a_{ij}|$, where $A = (a_{ij})$ is a square matrix. The normality $\|\cdot\|$ has the following properties.

- $\|A + B\| \leq \|A\| + \|B\|$
- $\|AB\| \leq \|A\|\|B\|$

Definition 3. *An estimator is strongly-consistent, if* $P(\|\widehat{P} - P\| < \epsilon) \to 1 \quad as \quad N = \min_{s \in S} \#s \to \infty$. *where function* $P(\phi)$ *with* ϕ *being a predicate is the probability of* ϕ *being true. It is stable, if* $P(\|\widehat{P} - P\| < \epsilon) > 1 - \delta(\epsilon, N)$, *where for any* $\epsilon > 0$, $\delta(\epsilon, N)$ *is a non-increasing function as* N *increases.*

Lemma 1. *The returned value of Algorithm 1 converges to the exact transition matrix* P *if an estimator is strongly-consistent and stable.*

Proof. Recall that our algorithm samples according to an initial distribution which maximizes $\min_{s \in S} E(s, \mu_0, \widehat{P}, N, k)$ during each iteration. As it goes to ∞, by the definition of a strongly-consistent and stable estimator, the maximum difference between two entries of P and $\widehat{P}$ converges to 0. Thus, for every entry (i, j), we have $|P_{ij} - \widehat{P}_{ij}| \leq \|P - \widehat{P}\| \to 0$, i.e., the estimation $\widehat{P}$ converges to P. □

Next, we establish that all of the above-mentioned estimators are strongly-consistent and stable.

Lemma 2. *The empirical frequency estimator, Laplace estimator and Good-Turing estimator are all strongly-consistent and stable.*

Proof. Let n be the number of states in the DTMC, $\#s_k$ be the number that $k-$th state is visited, and $N = \min_k \#s_k$. For each $1 \leq k \leq n$, we have $\#s_k \geq N$. Let (i, j) be the index pair such that $P_{ij} - \widehat{P}_{ij} = \|P - \widehat{P}\|$. By Chebyshev inequality, we have

$$\mathbf{P}(\|\widehat{P} - P\| < \epsilon) = \mathbf{P}(|\hat{p}_{ij} - p_{ij}| < \epsilon) \geq 1 - \frac{1}{\epsilon^2} \mathrm{Var}\, \hat{p}_{ij}$$

For strongly-consistency, we only need to show that for each estimator we have $\mathrm{Var}\, \hat{p}_{ij} \to 0$ as $N = \#s_i \to \infty$. For stability, we only need to show that for each estimator we have $\mathrm{Var}\, \hat{p}_{ij}$ is a non-increasing function as N increases. We discuss all the three estimators respectively in the following:
Empirical frequency estimator. It is easy to prove

$$\mathrm{Var}\, \hat{p}_{ij} = \frac{p_{ij} \cdot (1 - p_{ij})}{N} \leq \frac{1}{N} \to 0 \ \text{ as } \ N \to \infty \ \text{ and is non-increasing as } N \text{ increases.}$$

Laplace estimator. It is easy to prove

$$\mathrm{Var}\, \hat{p}_{ij} \leq \frac{N}{4(N+n)^2} \sim \frac{1}{N} \to 0 \quad \text{as} \quad N \to \infty \ \text{ and is non-increasing as } N \text{ increases.}$$

Good-Turing estimator. Assume state j occures k times after state i. From [15], we have the following for $\forall\sigma > 0, \forall^{\sigma} S$:

$$|\hat{p_{ij}} - p_{ij}| \leq \frac{k+2}{\#s_i - k} + \sqrt{\frac{2\ln(\frac{3}{\sigma})}{m}} \cdot \left[\frac{k+1}{1-k/\#s_i} + k + \sqrt{2k\ln(\frac{3m}{\sigma})} + 2\ln(\frac{3m}{\sigma})\right]$$
$$= \frac{k+2}{N-k} + \sqrt{\frac{2\ln(\frac{3}{\sigma})}{m}} \cdot \left[\frac{k+1}{1-k/N} + k + \sqrt{2k\ln(\frac{3m}{\sigma})} + 2\ln(\frac{3m}{\sigma})\right]$$

Approximately, the bound is

$$\|P - \widehat{P}\| = |\hat{p_{ij}} - p_{ij}| \leq \begin{cases} 2\ln(3N/\sigma)\sqrt{\frac{2\ln 3/\sigma}{N}} & \text{if } k \text{ small compared to } \ln\frac{3N}{\sigma} \\ 2k\sqrt{\frac{2\ln 3/\sigma}{N}} & \text{if } k \text{ large compared to } \ln\frac{3N}{\sigma} \end{cases}.$$

In both cases, the bound goes to 0 as $N \to \infty$ and is monotonically decreasing as N increases. □

Theorem 1. *The estimation $\widehat{P}$ returned by Algorithm 1 eventually converges to P for the empirical frequency estimator, Laplace estimator and Good-Turing estimator.*

Proof. By Lemma 1 and Lemma 2. □

In our approach, we calculate the optimal initial distribution based on an estimation $\widehat{P}$, instead of the actual P. In the following, we aim to quantify the distance between the optimal initial distribution calculated based on $\widehat{P}$ and that based on P. Assume $\|P - \widehat{P}\| \leq \epsilon$. Let A and B be the l-step accumulation matrix of P and $\widehat{P}$, respectively. That is, $A = I + P + P^2 + \cdots + P^{l-1}$ and $B = I + \widehat{P} + \widehat{P}^2 + \cdots + \widehat{P}^{l-1}$.

Proposition 1 *For $l \geq 1$, $\|A - B\| \leq l(l-1)\epsilon/2$.*

Proof. Let $\delta_k = \|P^k - \widehat{P}^k\|$ for any $k \geq 0$, then

$$\begin{aligned}\|A - B\| &\leq \|(I - I) + (P - \widehat{P}) + \cdots + (P^{l-1} - \widehat{P}^{l-1})\| \\ &\leq \|I - I\| + \|P - \widehat{P}\| + \cdots + \|P^{l-1} - \widehat{P}^{l-1}\| \\ &= \delta_0 + \delta_1 + \cdots + \delta_{l-1}.\end{aligned}$$

As P and $\widehat{P}$ are transition matrices whose elements are between zero and one. So for any $k > 0$,

$$\begin{aligned}\delta_k &= \|P^k - \widehat{P}^k\| = \|P^k - P^{k-1}\widehat{P} + P^{k-1}\widehat{P} - \widehat{P}^k\| \\ &\leq \|P^{k-1}(P - \widehat{P})\| + \|\widehat{P}(P^{k-1} - \widehat{P}^{k-1})\| \\ &\leq \|P^{k-1}\|\|P - \widehat{P}\| + \|\widehat{P}\|\|P^{k-1} - \widehat{P}^{k-1}\| \\ &\leq 1 \cdot \epsilon + 1 \cdot \delta_{k-1} = \epsilon + \delta_{k-1} \leq 2\epsilon + \delta_{k-2} \leq \cdots \leq k\epsilon + \delta_0 \\ &= k\epsilon.\end{aligned}$$

Therefore $\|A - B\| \leq l(l-1)\epsilon/2$. □

Approximation of optimization. Recall our objective as written in formula (3.1). Also note that the expectation of $\#s_i$ (given that the transition matrix is P, the initial distribution is μ and the index $i \in \{1, 2, \ldots, n\}$) is $\mathrm{E}\,\#s_i = (\mu A)_i$ where A is the accumulation matrix of P. Therefore, we consider the following objective

$$|\max_{\mu} \min_{i} (\mu A)_i - \min_{i} (\hat{\mu} A)_i| < \delta \quad \text{if} \quad \|P - \widehat{P}\| < \epsilon, \tag{3.2}$$

where $\hat{\mu} = \arg\max_{\mu} \min_i (\mu B)_i$.

Proposition 2 *For any μ and i, $|(\mu A)_i - (\mu B)_i| \leq l(l-1)\epsilon/2$.*

Proof. Note that every element of μ and A are between zero and one, it is easy to observe that $|(\mu A)_i - (\mu B)_i| < \|A - B\| \leq l(l-1)\epsilon/2$. □

Then we have the following proof sketch for inequality 3.2.

Proof.

$$\begin{aligned}
|\max_{\mu} \min_{i} (\mu A)_i - \min_{i} (\hat{\mu} A)_i| &\leq |\max_{\mu} \min_{i} (\mu A)_i - \max_{\mu} \min_{i} (\mu B)_i| \\
&\quad + |\max_{\mu} \min_{i} (\mu B)_i - \min_{i} (\hat{\mu} A)_i| \\
&= |\max_{\mu} \min_{i} (\mu A)_i - \max_{\mu} \min_{i} (\mu B)_i| \\
&\quad + |\min_{i} (\hat{\mu} B)_i - \min_{i} (\hat{\mu} A)_i| \\
&\leq l(l-1)\epsilon/2.
\end{aligned}$$

□

Remark 1. All the above happen with a probability of $1 - \delta(\epsilon, N)$.

Thus, we have the following theorem.

Theorem 2. *$|\max_{\mu} \min_i \#s_i - \min_i \#\widehat{s}_i| < \delta$ if $\|\widehat{P} - P\| < \epsilon$, where s_i is the sample set from the initial distribution μ and $\widehat{s}_i$ is the sample set from the initial distribution $\hat{\mu}$, which is the optimal distribution given that the transition matrix is $\widehat{P}$.*

Note that all s_i and $\widehat{s}_i$ are sample sets from the transition matrix P.

3.2 Estimating Reachability Pobability

Recall that our objective is also to estimate the probability of certain event occurring or equivalently reaching a certain state. This can be done based on the estimated transition matrix $\widehat{P}$. Given a DTMC $\mathcal{M} = (S, S_0, P, \mu)$, the probability of reaching state t from state s within l steps is defined as follows.

$$\mathbf{P}_{\mathcal{M}}(Reach_l(s,t)) = \begin{cases} 1 & \text{if} \quad s = t \wedge l \geq 0, \\ 0 & \text{if} \quad s \neq t \wedge l = 0, \\ \sum_{x \in S} \mathbf{P}_{\mathcal{M}}(Reach_{l-1}(x,t)) P(s,x) & \text{otherwise.} \end{cases}$$

We write $\mathbf{P}$ instead of $\mathbf{P}_{\mathcal{M}}$ and $\widehat{\mathbf{P}}$ instead of $\mathbf{P}_{\widehat{\mathcal{M}}}$ for conciser notation.

We now aim to prove that under the optimization objective stated in Equation 3.1, our algorithm for estimating reachability probability also converges to the actual reachability probability and improves monotonically, no matter which estimator is used.

Let us fix a target state s. We will show that $\widehat{\mathbf{P}}(Reach_l(s))$ converges to $\mathbf{P}(Reach_l(s))$ as the number of samples approaches infinity. Recall from Definition 3 and Lemma 2, for all the three estimators, we have $\mathbf{P}(\|\widehat{P} - P\| < \epsilon) \to 1 \quad \text{as} \quad N = \min_i \#s_i \to \infty$ and $\mathbf{P}(\|\widehat{P} - P\| < \epsilon) > 1 - \delta(\epsilon, N)$, where for any $\epsilon > 0$, $\delta(\epsilon, N)$ is non-increasing as N increases.

Remind that the reachability probabilities can be obtained by computing the transient probabilities according to the following steps. First, amend P to P_a by making state s absorbing, i.e., all outgoing transitions of s is replaced by a single self-loop at s. Then, the reachability probability of s can be calculated as:

$$\mathbf{P}(Reach_l(s)) = \mu \cdot \underbrace{P_a \cdot P_a \cdots P_a}_{l \text{ times}}(s) = \mu \cdot P_a^l(s) \tag{3.3}$$

Proposition 3 *If $\|P - \widehat{P}\| < \epsilon$, then $\|P_a^l - \widehat{P}_a^l\| \leq l\epsilon$.*

Proof. It is straightforward to see that $\|P_a - \widehat{P}_a\| < \epsilon$. Now Let $\delta_k = \|P_a^k - \widehat{P}_a^k\|$ for $k \geq 0$. Then we have

$$\begin{aligned}
\delta_k &= \|P_a^k - \widehat{P}_a^k\| = \|P_a^k - P_a^{k-1}\widehat{P}_a + P_a^{k-1}\widehat{P}_a - \widehat{P}_a^k\| \\
&\leq \|P_a^{k-1}(P_a - \widehat{P}_a)\| + \|\widehat{P}_a(P_a^{k-1} - \widehat{P}_a^{k-1})\| \\
&\leq \|P_a^{k-1}\|\|P_a - \widehat{P}_a\| + \|\widehat{P}_a\|\|P_a^{k-1} - \widehat{P}_a^{k-1}\| \\
&\leq \|P_a^{k-1}\|\|P - \widehat{P}\| + \|\widehat{P}_a\|\|P_a^{k-1} - \widehat{P}_a^{k-1}\| \\
&\leq 1 \cdot \epsilon + 1 \cdot \delta_{k-1} = \epsilon + \delta_{k-1},
\end{aligned}$$

and $\delta_0 = 0$. By induction, we know $\|P_a^l - \widehat{P}_a^l\| = \delta_l < l\epsilon$. □

Proposition 3 in fact tells us that the estimation error ϵ accumulates linearly when propagating. And that immediately leads us to the next proposition.

Proposition 4 *$|\widehat{\mathbf{P}}(Reach_l(s)) - \mathbf{P}(Reach_l(s))| \leq l\epsilon$ for any state s and a bounded step l.*

Proof. By Equation 3.3 and Proposition 3. □

Theorem 3. *$\mathbf{P}\left(|\widehat{\mathbf{P}}(Reach_l(s)) - \mathbf{P}(Reach_l(s))| \leq l\epsilon\right) \to 1$ for any state s and a bounded step l, as $N = \min_i \#s_i \to \infty$ and $\mathbf{P}\left(|\widehat{\mathbf{P}}(Reach_l(s)) - \mathbf{P}(Reach_l(s))| \leq l\epsilon\right) > 1 - \delta(\epsilon, N)$, where for any $\epsilon > 0$, $\delta(\epsilon, N)$ is non-increasing as N increases.*

Proof. By Definition 3, Lemma 2 and Proposition 4. □

4 Evaluation

We have developed a prototype implementation of our approach called IDO in 4k lines of Java code. To evaluate the effectiveness of IDO, we compare it with the passive approach (i.e., random sampling, referred to as PA) using the following metrics.

- Firstly, we count the minimum number that a state is visited among all the reachable states (referred to as MV). Note that this is precisely our optimization objective. By measuring MV, we aim to check whether the optimization technique we adopt has worked as expected. The larger the MV value is, the better.
- Secondly, we compare the estimated probability of reaching a state of interest. We use the relative difference from the estimated reachability probability to the actual reachability probability (RRD) as a measure, which is defined as follows.

$$\frac{|\mathbf{P}_{\widehat{\mathcal{M}}}(Reach_k(s)) - \mathbf{P}_{\mathcal{M}}(Reach_k(s))|}{\mathbf{P}_{\mathcal{M}}(Reach_k(s))}$$

  A smaller RRD indicates a more precise estimation of reachability probability. If there are more than one state of interest, we calculate the average RRD.
- Thirdly, we compare the estimated transition matrix $\widehat{P}$ using standard notions like MSE. Similarly, a smaller MSE indicates a more precise estimation.

For a fair comparison, the experiments are designed such that the number of samples used are the same for IDO and PA.

4.1 Test Subjects

In order to answer the above research questions, three groups of systems are used for our evaluation. The first set contains three benchmark system, i.e., the small example shown in Figure 1 and two systems from the literature [6]. One is the *queuing model*. In a queuing model, customers arrive at a station for service. States represent the number of persons in the queue. We consider a scenario where the only possible transitions are the arrival or departure of someone. The number of transitions between two consecutive observations are assumed to be i.i.d.. For convenience, we assume a maximum number of persons in the queue to be 10. Thus, the Markov chain has 11 possible states and a transition matrix as follows:

$$M = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.53 & 0 & 0.47 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.65 & 0 & 0.35 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.45 & 0 & 0.55 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.30 & 0 & 0.70 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.62 & 0 & 0.38 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.68 & 0 & 0.32 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.64 & 0 & 0.36 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.52 & 0 & 0.48 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.61 & 0 & 0.39 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{4.1}$$

We set the first 6 states as initial states and assume an initial uniform distribution over the 6 states. The reachability probability of interest is the probability of reaching the last 3 states in 11 (which is the number of states) steps. Based on the above model, the precise reachability probability can be calculated as: 0.0444, 0.0194 and 0.0075 respectively. The other model is the *hollow matrix*. This case study deals with Markov chains that changes state at each transition. The transition matrix is as follows.

$$P = \begin{bmatrix} 0 & 0.992 & 0.0003 & 0.0005 \\ 0.98 & 0 & 0.01 & 0.01 \\ 0.40 & 0.13 & 0 & 0.47 \\ 0.42 & 0.20 & 0.38 & 0 \end{bmatrix} \tag{4.2}$$

We set the first 2 states as initial states and assume a distribution $(0.99, 0.01)$ over them. The reachability probabilities of interests are reaching the last 2 states in 4 (number of states) steps, which are 0.0147 and 0.0159 respectively.

The second group is a set of randomly generated models (referred to as *rmc*). These models are generated with different numbers of states and transition densities using an approach similar to the approach in [17]. For reachability analysis, we choose first half of the states to be the initial states and assume a uniform initial distribution over them. We select those states with reachability probability less than 0.05 as states of interest, since we are interested in improving reachability probability of low probability states.

The last group contains the SWaT testbed [1]. SWaT is a real world complex system which involves a series of water treatments process from raw water like ultra-filtration, chemical dosing, dechlorination through an ultraviolet system, etc. The system is safety critical and ideally we want to accurately estimate the probability of reaching some *bad* states, like tank overflow or underflow, abnormal water pH level, etc. Modeling SWaT is challenging and thus we would like to have a way of estimating the transition probability as well as some reachability probability. SWaT has many configurable parameters which can be set before the system starts and it can be restarted if necessary. However, restarting SWaT is non-trivial as we have to follow a predefined shutdown sequence and thus we would like to obtain some precise estimation with as few restarts as possible. In our experiment, we focus on tank overflow or underflow. We select states with normal tank levels as initial states and assume a uniform initial distribution over them. Furthermore, we select states with abnormal tank level as states of interest.

4.2 Experiment Results

We first show the experiment results on the benchmark systems. Figure 2 presents the comparison of IDO and PA in terms of MV, RRD, and MSE respectively for the three benchmark systems. The first row shows the results of the first example. It can be observed that MV of IDO improves linearly as we increase the number of samples, whereas MV of PA remains almost zero due to the low probability of reaching some states according to the original initial distribution. IDO has significantly better estimation of both the reachability probability (in terms of RRD) as well as the transition probability (in terms of MSE). The second row shows the results of the hollow matrix. It can be observed that the improvement of MV and the probability estimation are not

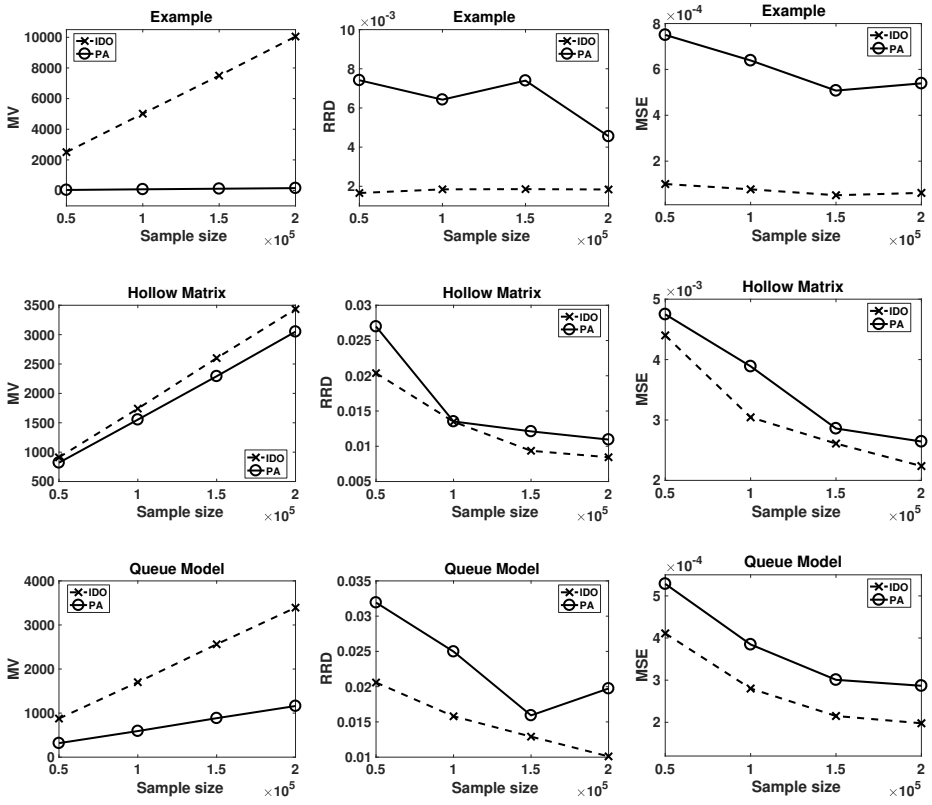

Fig. 2: Experiment results of benchmark systems.

as significant as for the first example. A closer investigation shows that the reason is that its two initial states have very high probability of transitioning to each other. As a result, adjusting the initial distribution does not effectively change how the other states are visited. The third row shows the results of the queuing model. We observe a noticeable improvement in terms of MV, RRD ad MSE. This is because that a state of the queuing model can only be reached from its previous and next states. IDO successfully identifies an initial distribution which favors the latter part of the initial states (e.g. state 5, 6), which subsequently leads to more visits of states of interest.

Next, we present the experiment results on the random models. The results are shown in Figure 3. We consider random models with 8 states or 16 states. We randomly generate a set of 20 models of 8 states and 20 models of 16 states and present the average results, to avoid the influence of randomness. It can be observed that IDO improves MV, RRD and MSE in almost all the cases. On one hand, we observe from the results that as the state number increases, IDO's improvement over PA in terms of MSE goes down. The reason is that IDO targets to improve the worst estimated transitions, while MSE is computed in terms of all transitions. Consequently, the improvement is

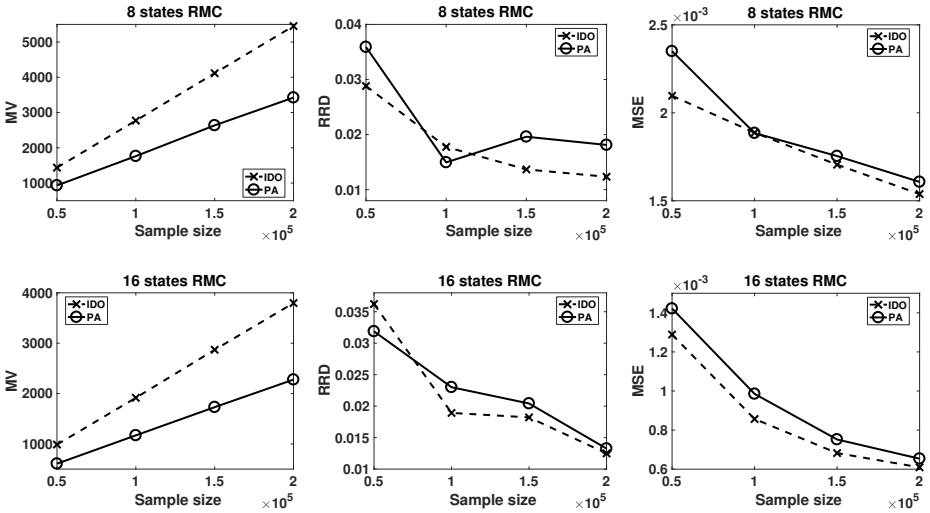

Fig. 3: Experiment results of *rmc*.

flattened with a large number of transitions. On the other hand, we observe a more and more significant improvement in terms of reachability estimation when the number of states increases. This is due to the fact that IDO actively selects an initial distribution which is likely to visit the target state most often, which effectively improves the estimation. In comparison, random sampling using a uniform initial distribution may visit the target states much less. We remark this is an important advantage of IDO since for complex systems, there are often many states and we are often interested in estimating the probability of certain unlikely states (e.g., unsafe states). Considering the extreme case when only one initial state s_0 would lead to visit of some target state s_t. If the number of initial states is large, there is a very low probability to sample s_0 through uniform random sampling. As a result, s_t is rarely visited. In comparison, IDO optimizes the initial distribution and samples from s_0 with probability 1.

Lastly, we report the results on the SWaT testbed. One difficulty we face in evaluating the effectiveness of our approach for SWaT is that we do not have the actual DTMC model. To overcome this problem, we run a large number of simulations (120k traces, each trace is a simulation of half an hour with uniform initial distribution), and then apply empirical frequency to estimate the transition probability, which we regard as an approximation of the actual transition matrix. We remark that all the traces of SWaT are generated using a simulator which fully mimics the testbed, and for each trace the running time of the simulator is scaled less than the running time of the actual system. Note that we define a state of SWaT as a combination of sensor values. Since the target states that we are looking into are overflow and underflow of water tanks, we collect those sensors that indicate water levels to encode states. In other words, we abstract away the internal states for simplicity. We further abstract the sensor values (which

#state	MV		RRD		MSE		Time cost(s)	
	IDO	PA	IDO	PA	IDO	PA	IDO	PA
64	19	8	6.31	7.13	4.58E-4	3.93E-4	12553	12601
216	3	1	43	59.7	5.49E-4	4.86E-4	13700	12785

Table 1: Experiment results of SWaT.

are continuous float variables) into discrete states. Two different abstraction are experimented, one with 64 states and the other with 216 states. In our experiment, we generate the first estimation $\widehat{P}$ based on 5000 traces (randomly selected from the 120K traces). Afterwards, we iteratively refine the estimation using IDO by adding and learning from additional 5000 traces each time. The total number of traces used by IDO and PA are the same. Similarly, we compare the MV, MSE and RRD of a set of target states, whose reachability probabilities are less than 0.01, for IDO and PA respectively. The results are shown in Table 1. It can be observed that the MSE is expectedly not improving as we have many states to take average on. However, we can see from the results of RRD that IDO effectively improves our estimation of probability of water tank overflow or underflow which interest us. Furthermore, we observe almost negligible overhead of IDO over PA in terms of running time.

5 Conclusion and Related Work

This paper is our continuous work in applying learning probabilistic models for verifying complex systems like cyber-physical systems [19, 18]. The proposed approach is to "smartly" tune the system configurations so as to generate traces that can effectively improve our current probability estimation. We prove that our algorithm will converge under various existing popular estimators. The experiment results show that our approach effectively improves random sampling in terms of probability estimation and reachability analysis (especially). In the future, we plan to conduct a complete case study on SWaT combining active learning, abstraction, and refinement. The idea of this work is also promising to be applied to 'active' program testing by tuning program input to discover more interesting program states rather than random testing.

This work is mainly related to the following three lines of work. Firstly, it is a further effort in the recent trend of learning probabilistic models for model checking [14]. Instead of learning from fixed data, we propose to actively sample the system for more informative traces to make learning more effective for reachability analysis [13]. Such an active learning idea is applied in [8] to learn Markov decision process actively by choosing optimal actions in each step. Secondly, importance sampling [12] is another approach of smart sampling, but it may require us to change the probability distribution in the process of system operation, which is sometimes unrealistic for cyber-physical systems. Our work differs in that we only require to tune the initial distribution by adjusting the initial configuration of the system. Lastly, this work relies and works on a variety of estimators [9, 10, 15], which are designed for different applications.

References

1. http://itrust.sutd.edu.sg/research/testbeds/secure-water-treatment-swat/.
2. Linear programming — Wikipedia, the free encyclopedia, 2016. [Online; accessed 24-November-2016].
3. Mean squared error — Wikipedia, the free encyclopedia, 2016. [Online; accessed 7-December-2016].
4. Dana Angluin. Learning regular sets from queries and counterexamples. *Inf. Comput.*, 75(2):87–106, 1987.
5. Christel Baier, Joost-Pieter Katoen, et al. *Principles of model checking*, volume 26202649. MIT press Cambridge, 2008.
6. Flavia Barsotti, Yohann De Castro, Thibault Espinasse, and Paul Rochet. Estimating the transition matrix of a markov chain observed at random times. *Statistics & Probability Letters*, 94:98–105, 2014.
7. Klaus Brinker. Incorporating diversity in active learning with support vector machines. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, pages 59–66, 2003.
8. Yingke Chen and Thomas Dyhre Nielsen. Active learning of markov decision processes for system verification. In *Machine Learning and Applications (ICMLA), 2012 11th International Conference on*, volume 2, pages 289–294. IEEE, 2012.
9. G Cochran. Laplace s ratio estimator. *Contributions to survey sampling and applied statistics (New York, 1978)*, pages 3–10, 1978.
10. William A Gale and Geoffrey Sampson. Good-turing frequency estimation without tears*. *Journal of Quantitative Linguistics*, 2(3):217–237, 1995.
11. Inc. Gurobi Optimization. Gurobi optimizer reference manual, 2016.
12. Cyrille Jegourel, Axel Legay, and Sean Sedwards. Cross-entropy optimisation of importance sampling parameters for statistical model checking. In *International Conference on Computer Aided Verification*, pages 327–342. Springer, 2012.
13. Kendra Lesser and Meeko Oishi. Reachability for partially observable discrete time stochastic hybrid systems. *Automatica*, 50(8):1989–1998, 2014.
14. Hua Mao, Yingke Chen, Manfred Jaeger, Thomas D Nielsen, Kim G Larsen, and Brian Nielsen. Learning probabilistic automata for model checking. In *Eighth International Conference on Quantitative Evaluation of Systems (QEST), 2011*, pages 111–120. IEEE, 2011.
15. David A McAllester and Robert E Schapire. On the convergence rate of good-turing estimators. In *COLT*, pages 1–6, 2000.
16. Todd K Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, 1996.
17. Deian Tabakov and Moshe Y. Vardi. Experimental evaluation of classical automata constructions. In *Logic for Programming, Artificial Intelligence, and Reasoning, 12th International Conference, LPAR 2005, Montego Bay, Jamaica, December 2-6, 2005, Proceedings*, pages 396–411, 2005.
18. Jingyi Wang, Jun Sun, and Shengchao Qin. Verifying complex systems probabilistically through learning, abstraction and refinement. *arXiv preprint arXiv:1610.06371*, 2016.
19. 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 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings*, pages 3–21, 2017.
20. James A Whittaker and Michael G Thomason. A markov chain model for statistical software testing. *IEEE Transactions on Software engineering*, 20(10):812–824, 1994.