Improving Probability Estimation Through Active Probabilistic Model Learning

Jingyi Wang[1(✉)], Xiaohong Chen[1,2], Jun Sun[1], and Shengchao Qin[3]

[1] Singapore University of Technology and Design, Singapore, Singapore
wangjyee@gmail.com, jingyi_wang@mymail.sutd.edu.sg
[2] The University of Illinois at Urbana-Champaign, Champaign, USA
[3] Teesside University, Middlesbrough, UK

Abstract. It is often necessary to estimate the probability of certain events occurring in a system. For instance, knowing the probability of events triggering a shutdown sequence 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 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 is too low or too high. Whenever a monitor issues a safety alarm, a shutdown sequence is triggered so that the system halts 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 occasionally. To show that the system satisfies certain availability requirements, we would like

This work was supported by project RG101NR0114A.

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

Z. Duan and L. Ong (Eds.): ICFEM 2017, LNCS 10610, pp. 379–395, 2017.
https://doi.org/10.1007/978-3-319-68690-5_23

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 discrete-time Markov Chain [5]) and estimate the probability of the interesting 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 readings 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 interesting 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 multiple times, the same initial distribution is applied to configure the system accordingly.

Such a method however may not be effective if the interesting 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 interesting 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 a stopping criteria is 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 [20]. The 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, which yields a weaker and more realistic requirement than other distribution manipulation techniques for rare event analysis like importance sampling [12]. In addition, 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 (short for Initial Distribution Optimizer). 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 a finite number of 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 set 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 in order (which is often the case), we denote the ith state of S as s_i, and $P(s_i, s_j)$ as p_{ij}. Given a path $s_{i_1} \dots s_{i_k}$, the probability of observing that path is $\mu_{i_1} p_{i_1 i_2} \dots p_{i_{k-1} i_k}$, denoted[2]

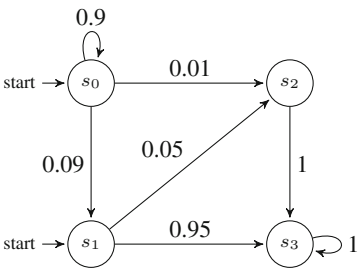

Fig. 1. An example DTMC

[2] We use P to denote transition matrices and $\mathscr{P}$ to denote the probability measure defined by P.

by $\mathscr{P}(i_1 \dots i_k)$. Which depends not only on the transition matrix P but also on the initial distribution μ. Figure 1 shows an example DTMC with four states and two initial states. Transition probabilities are labeled upon arrows, and the initial states are attached by the label "start". Not drawing an arrow means a zero transition probability.

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 distribution, how to estimate (1) its transition matrix, and (2) the probability of reaching certain states (a.k.a. the *reachability probability*)? In this paper, we assume the transition matrix is fixed, 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) when we initialize the system, and once the system is turned on, we can only observe how the system evolves through time, but not affect how it goes.

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, the actual transition matrix P can be estimated based on the experiment results (and subsequently the reachability probability can be estimated, too). In the beginning of every experiment, an initial state s is randomly generated according to the initial distribution μ. Starting from s, the system transits to the next state according to the transition matrix P, and then transits the third state, and so on, until a certain number of steps are taken and a path is obtained. Such a random path, often denoted as π, is a random variable whose distribution is fully decided by the initial distribution μ and the transition matrix P. We denote that as $\pi \sim (P, \mu)$. By abuse use of notation, we write $\Pi \sim (P, \mu)$ if Π is a set of paths that are independently generated from P and μ. Once a sample set $\Pi \sim (P, \mu)$ is obtained, an estimation function (a.k.a. an estimator) is applied on Π to generate an estimation $\widehat{P}$ of the transition matrix P. In practice, there are three commonly used estimators, which we introduce in the next definition.

Definition 2. *Let $\Pi \sim (P, \mu)$ be a set of path samples. For any states s and t, let $\#_{s,t}$ be the number of times that the one-step transition from s to t occurs in Π, and $\#_s = \Sigma_t \#_{s,t}$. We provide three commonly-used estimators $\widehat{P}$'s that are purely based on $\#_{s,t}$'s. They are*

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

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 s to t is not observed in Π because the actual $P(s,t)$ is small, $\widehat{P}_E(s,t)$ is zero by the empirical frequency estimator. The Laplace estimator overcomes this problem by adding a constant 1 to the numerator and the number of state to the denominator of the estimated transition probability. In other words, if state t is never visited after state s, the probability $\widehat{P}_L(s,t) = 1/(n + \#_s)$. 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 by actively choosing the initial state to sample from.* Ideally, it should work with any of the above-mentioned estimators. In particular, the question is: if a user is only 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? We believe this is a realistic assumption. Consider the above-mentioned SWaT test bed for example. The user can only choose a set of initial configurations following certain distribution to perform experiments 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 improve 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 $\widehat{P}$ and the current samples Π, we compute an "optimal" initial distribution μ_0 with respect to our objective at line 5. Then, we repeat from line 3, i.e., acquire more samples Π' based on μ_0, add them to Π and apply the estimator on the updated Π to obtain an updated

estimation of $\widehat{P}$. The process continues until a stopping criterion is satisfied. This approach is inspired by the expectation-maximization (EM) algorithm from statistics [16].

Algorithm 1. Sampling based on active learning

1: Let μ_0 be μ, Π be empty;
2: **while** stopping criteria unsatisfied **do**
3: Sample the system N times to obtain $\Pi' \sim (\mathcal{M}, \mu_0)$;
4: Update $\Pi = \Pi \cup \Pi'$;
5: Apply an estimator to obtain $\widehat{P}$ based on Π;
6: Set μ_0 to be the optimal initial distribution computed based on $\widehat{P}$ and Π;
7: **end while**
8: Output $\widehat{P}$;

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 such as the *mean squared error* (MSE), the standard deviation, or bias [3]. As an example, the MSE of an estimation $\widehat{P}$ is defined as

$$\mathrm{MSE}(\widehat{P}, P) = \frac{1}{(|S|)^2} \sum_{s,t \in S} (\widehat{P}(s,t) - P(s,t))^2. \tag{3.1}$$

Ideally, we would like to identify an initial distribution μ_0 such that the estimation would be most accurate. However, since the actual transition matrix P is unknown, we cannot directly compare the estimation $\widehat{P}$ and P (e.g., in term of MSE). Thus, we 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 Fig. 1, it is hard to estimate the probability of transitioning from state s_1 to s_2 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.

$$\max_{\mu_0 \in \mathcal{D}} \min_{s \in S} E(s, \mu_0, \widehat{P}, N, k) \tag{3.2}$$

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.

$$E(s, \mu_0, \widehat{P}, N, k) = N\left(\mu_0(s) + \mu_0\widehat{P}(s) + \mu_0\widehat{P}^2(s) + \cdots + \mu_0\widehat{P}^k(s)\right) \tag{3.3}$$

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.

Optimization. In each iteration, given current sample set Π, let $\gamma = \min_{s\in S} \#_s$ and $i = \text{argmin}_{s\in S}\#_s$. The optimization objective of Eq. 3.2 turns to the following.

$$\max_{\mu_0\in\mathcal{D}} E(i, \mu_0, \widehat{P}, N, k) = \max_{\mu_0\in\mathcal{D}} N\left(\mu_0(i) + \mu_0\widehat{P}(i) + \mu_0\widehat{P}^2(i) + \cdots + \mu_0\widehat{P}^k(i)\right) \tag{3.4}$$

The initial distribution μ_0 has two constraints: (1) every element of μ_0 is between 0 and 1; (2) the sum of the elements are equal to 1. This forms a standard linear optimization problem [2] which can be solved by applying a linear optimization solver like Gurobi [11]. We then generate the 'optimal' initial distribution μ_0 by solving the optimization problem stated in Eq. 3.4 over the constraints.

Convergence. In this section, 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.

The following notations are frequently used. We define $\|\cdot\|$ as the max norm $\|A\| = \max_{ij}|a_{ij}|$. Notice that the estimator $\widehat{P}$ is a random variable that is fully determined by the path samples, whose distribution is given by $\mathscr{P}$, so we simply use the same notation $\mathscr{P}(\phi)$ with ϕ being a predicate to denote the probability of ϕ being true. The choice of matrix normality in this paper is mainly a taste of flavor. Different norms will result in different actual bounds of the inequalities that will be presented later, but since our goal here is to establish that the estimation $\widehat{P}$ getting closer and closer to the actual value P, i.e., $\|\widehat{P} - P\| \to 0$, using different norms will not make a difference, thanks to the next proposition, whose proof we omit.

Proposition 1. *Suppose $\|\cdot\|_1$ and $\|\cdot\|_2$ are two matrix norms, and $A_1, A_2, \ldots$ is a sequence of $m \times n$ matrices. Then*

$$\|A_n\|_1 \to 0 \quad \textit{iff} \quad \|A_n\|_2 \to 0.$$

Definition 3. *An estimator is strongly-consistent, if $\mathscr{P}(\|\widehat{P} - P\| < \epsilon) \to 1$ as $\gamma = \min_{s\in S}\#_s \to \infty$. It is stable, if $\mathscr{P}(\|\widehat{P} - P\| < \epsilon) > 1 - \delta(\epsilon, \gamma)$, where for any $\epsilon > 0$, $\delta(\epsilon, \gamma)$ is a non-increasing function as γ increases.*

Estimators defined as above guarantee that, by optimizing our optimization objective, $\widehat{P}$ will converge to P (strongly-consistency), and $\widehat{P}$ will improve monotonically (stability). This is stated in the following Lemma.

Lemma 1. *The return 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 all 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, $\#_s$ be the number that state s is visited, and $\gamma = \min_s \#_s$. For each $1 \leq k \leq n$, we have $\#_k \geq \gamma$. Let (i, j) be the index pair such that $p_{ij} - \widehat{p}_{ij} = \|P - \widehat{P}\|$. By Chebyshev inequality, we have

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

For strong-consistency, we only need to show that for each estimator we have $\operatorname{Var}\hat{p}_{ij} \to 0$ as $\gamma = \min_s \#_s \to \infty$. For stability, we only need to show that for each estimator we have $\operatorname{Var}\hat{p}_{ij}$ is a non-increasing function as γ increases. Respectively,

Empirical Frequency Estimator. It is easy to prove

$$\operatorname{Var}\hat{p}_{ij} \leq \frac{p_{ij}(1 - p_{ij})}{\gamma} \to 0 \quad \text{as} \quad \gamma \to \infty \ \text{ and is non-increasing as } \gamma \text{ increases.}$$

Laplace Estimator. It is easy to prove

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

Good-Turing Estimator. Assume state j occurs λ times after state i, i.e., $\#_{ij} = \lambda$. From the results of [15], for $\forall\sigma > 0$, the approximate bound is:

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

In both cases, the bounds go to 0 as $\gamma \to \infty$ and is monotonically decreasing as γ increases. □

Thus, we have the following Theorem on the correctness of Algorithm 1.

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 Lemmas 1 and 2. □

Stopping Criteria of Algorithm 1. We provide two stopping criteria for two common scenarios when the algorithm is used in practice. The first is when we have a limited sampling budget, which is a very common case in reality. For instance, we can run the system only for a bounded number of times, in which case Algorithm 1 terminates when we run out of budget. The other scenario is when we require our estimation $\widehat{P}$ be as close as possible to the actual P with a high probability. That is, $\mathscr{P}(|p_{ij} - \hat{p}_{ij}| < \epsilon) > 1 - \alpha$ for all i and j, where the *sampling parameters* ϵ and α are positive numbers close to 0. Given ϵ and α, we calculate a threshold ξ on the minimum number of $\#_i$ to satisfy the requirement based on the bounds described in the proof of Lemma 2. Then we run Algorithm 1 until $\gamma = \min_{s \in S} \#_s \geq \xi$. Note that the latter scenario often results in a larger sample size given a small ϵ and α, as we observed from our experiments described in Sect. 4.2.

Approximation of Optimization. In Algorithm 1, the optimal initial distribution is calculated based on the estimation $\widehat{P}$, instead of the actual P. In the following, we aim to show that sampling according to the optimal initial distributions calculated based on $\widehat{P}$ approximates sampling according to the actual optimal initial distribution in terms of the numbers that states are visited.

Assume $\widehat{P}$ and P are $n \times n$ square matrices and their difference is bounded by ϵ. Define $\widehat{A} = I + \widehat{P} + \widehat{P}^2 + \cdots + \widehat{P}^{l-1}$ and $A = I + P + P^2 + \cdots + P^{l-1}$ be the l-step accumulation matrices of $\widehat{P}$ and P respectively for $l > 0$. The next proposition shows the difference between $\widehat{A}$ and A is bounded by $O(l^2)\epsilon$.

Proposition 2. $\|\widehat{A} - A\| \leq l(l-1)n\epsilon/2$.

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

$$\begin{aligned}\|\widehat{A} - A\| &\leq \|(I - I) + (\widehat{P} - P) + \cdots + (\widehat{P}^{l-1} - P^{l-1})\| \\ &\leq \|I - I\| + \|\widehat{P} - P\| + \cdots + \|\widehat{P}^{l-1} - P^{l-1}\| \\ &= \tau_0 + \tau_1 + \cdots + \tau_{l-1}.\end{aligned}$$

Recall that $\widehat{P}$ and P are transition matrices. We have $\tau_0 = 0$, and for any $k > 0$,

$$\begin{aligned}\tau_k &= \|\widehat{P}^k - P^k\| = \|\widehat{P}^k - \widehat{P}^{k-1}P + \widehat{P}^{k-1}P - P^k\| \\ &\leq \|\widehat{P}^{k-1}(\widehat{P} - P)\| + \|(\widehat{P}^{k-1} - P^{k-1})P\| \\ &\leq n\|\widehat{P} - P\| + \|\widehat{P}^{k-1} - P^{k-1}\| \\ &\leq n\epsilon + \tau_{k-1} \leq 2n\epsilon + \tau_{k-2} \leq \cdots \leq kn\epsilon + \tau_0 = kn\epsilon.\end{aligned}$$

Therefore $\|\widehat{A} - A\| \leq (0 + 1 + 2 + \cdots + l)n\epsilon = l(l-1)n\epsilon/2$. □

Next, we take the initial distribution μ into consideration. We use subscript i to denote the ith projection of vectors.

Proposition 3. *For any μ and i, $|(\mu A)_i - (\mu\widehat{A})_i| \leq l(l-1)n\epsilon/2$.*

Proof. Assume $\mu = (\mu_1, \ldots, \mu_n)$, $A = (a_{ij})_{n\times n}$, and $\widehat{A} = (\hat{a}_{ij})_{n\times n}$. Notice that μ is a distribution, so $\mu_1 + \cdots + \mu_n = 1$ and $0 \leq \mu_i \leq 1$, for each $i = 1, \ldots, n$, and

$$\begin{aligned} |(\mu\widehat{A})_i - (\mu A)_i| &\leq \mu_1|\hat{a}_{1i} - a_{1i}| + \cdots + \mu_n|\hat{a}_{ni} - a_{ni}| \\ &\leq \max(|\hat{a}_{1i} - a_{1i}|, \ldots, |\hat{a}_{ni} - a_{ni}|) \\ &\leq \|\widehat{A} - A\| \leq l(l-1)n\epsilon/2. \end{aligned}$$

□

Recall that our optimization goal is formula (3.2), in which the estimation $\widehat{P}$ is used as an approximation of the actual transition matrix P, and as a result, the optimal initial distribution $\hat{\mu}_{opt} = \arg\max_\mu \min_i(\mu\widehat{A})_i$ is in general not the actual (unknown) optimal initial distribution $\mu_{opt} = \arg\max_\mu \min_i(\mu A)_i$. The next proposition shows that even so, it makes little difference whether we do sampling according to $\hat{\mu}_{opt}$ or μ_{opt}, as long as our estimation $\widehat{P}$ gets close enough to the actual P.

Proposition 4. *Under previous notations and conditions, $|\min_i(\hat{\mu}_{opt}A)_i - \min_i(\mu_{opt}A)_i| \leq l(l-1)n\epsilon$.*

Proof.

$$\begin{aligned} &|\min_i(\hat{\mu}_{opt}A)_i - \min_i(\mu_{opt}A)_i| = |\min_i(\hat{\mu}_{opt}A)_i - \max_\mu \min_i(\mu A)_i| \\ \leq\ &|\min_i(\hat{\mu}_{opt}A)_i - \max_\mu \min_i(\mu\widehat{A})_i| + |\max_\mu \min_i(\mu\widehat{A})_i - \max_\mu \min_i(\mu A)_i| \\ \leq\ &|\min_i(\hat{\mu}_{opt}A)_i - \min_i(\hat{\mu}_{opt}\widehat{A})_i| + l(l-1)n\epsilon/2 \leq l(l-1)n\epsilon. \end{aligned}$$

□

Recall that $\#_s$ is the expected number of times we visit a state s when sampling according to (P, μ). In particular, we write $\widehat{\#}_s$ if paths are sampled from the optimal initial distribution obtained by solving the optimization problem (3.2), i.e., from $(P, \hat{\mu}_{opt})$. Then Proposition 4 directly leads to the next main theorem, which guarantees that solving the approximate optimization problem gives us an approximate solution to the original optimization problem, and thus justifies our approach.

Theorem 2. *Using previous notations, $|\max_\mu \min_i \#_s - \min_i \widehat{\#}_s| \leq l(l-1)n\epsilon$ if $\|\widehat{P} - P\| < \epsilon$, where l is the length of each path sample and n is the dimension of the transition matrix P and its estimation $\widehat{P}$.*

3.2 Estimating Reachability Probability

In previous sections, we have shown how to obtain an approximation $\widehat{P}$ of the actual transition matrix P. From now on we will assume such an approximation exists, and will use $\widehat{\mathscr{P}}$ to denote the probability measure that it defines. In this section, we will show one important application of such approximation $\widehat{\mathscr{P}}$, which is estimating the *reachability probability*, the probability of reaching certain states or observing certain events occurring. 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.

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

We aim to prove that estimating reachability probability using the estimation returned by Algorithm 1 also converges to the actual reachability probability and improves monotonically, no matter which estimator is used.

To facilitate discussion, let us fix a target state $s_i \in S$. We will show that

$$\widehat{\mathscr{P}}(Reach_l(s_i)) \to \mathscr{P}(Reach_l(s_i)) \quad \text{as} \quad \widehat{P} \to P.$$

Notice that $\mathscr{P}(Reach_l(s_i))$ (and similarly $\widehat{\mathscr{P}}(Reach_l(s_i))$) can be computed by

$$\mathscr{P}(Reach_l(s_i)) = \left(\mu \cdot \underbrace{P_a \cdot P_a \cdots P_a}_{l \text{ times}} \right)_i = (\mu P_a^l)_i, \tag{3.5}$$

where P_a is the amended transition matrix in which we make the state s *absorbing*, i.e., all outgoing transitions from s are replaced by a single self-loop at s.

The next proposition provides an $O(l)\epsilon$ bound on the difference between $\widehat{P}_a^l$ and P_a^l. We omit the proofs of Propositions 5 and 6 because they use the same tricks that we have seen in proving Propositions 2 and 3.

Proposition 5. *$\|\widehat{P}_a^l - P_a^l\| \leq nl\epsilon$ if $\|\widehat{P} - P\| < \epsilon$, where n is the dimension of P.*

Now let us take the initial distribution μ into consideration.

Proposition 6. *For any μ and i, if $\|\widehat{P} - P\| < \epsilon$, then $|(\mu\widehat{P}_a^l)_i - (\mu P_a^l)_i| \leq nl\epsilon$.*

This leads to the following theorem on the bound of reachability probability.

Theorem 3. *$|\widehat{\mathscr{P}}(Reach_l(s_i)) - \mathscr{P}(Reach_l(s_i))| \leq nl\epsilon$ for any state $s_i \in S$ and a bounded number of steps l.*

Proof. By Eq. (3.5) and Proposition 6. □

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 of times 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

$$\frac{|\widehat{\mathscr{P}}(Reach_k(s)) - \mathscr{P}(Reach_k(s))|}{\mathscr{P}(Reach_k(s))}.$$

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 the standard notion MSE (defined by Eq. 3.1).* 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, i.e., using the first stopping criteria.

4.1 Test Subjects

Three groups of systems are used for our evaluation. The first set contains three benchmark systems, i.e., the small example shown in Fig. 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 independent and identically distributed (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

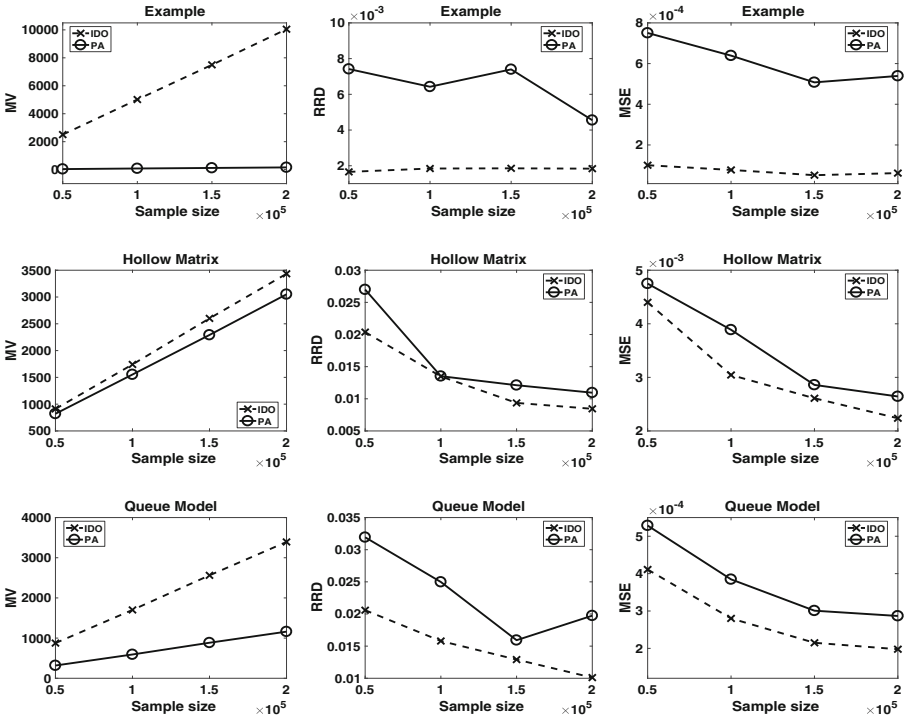

Fig. 2. Experiment results of benchmark systems.

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 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 transit from its neighboring states. Since the states of interests here are the states in the last (e.g. state 9, 10, 11), an initial distribution which favors the latter part of the initial states (e.g. state 5, 6) is more likely to reach the target states. IDO successfully identifies such an initial distribution, which subsequently leads to more visits of the target states.

Next, we present the experiment results on the random models. The results are shown in Fig. 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

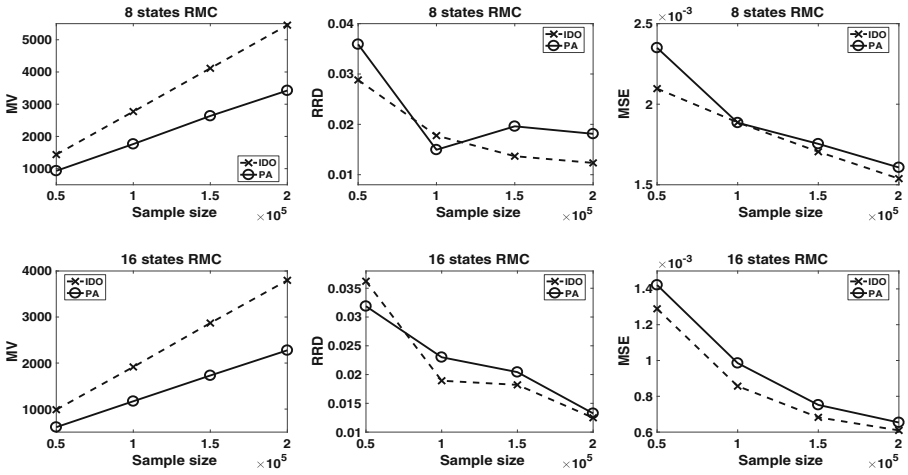

Fig. 3. Experiment results of *rmc*.

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 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 (120 k traces, each trace is a simulation of half an hour of real world system 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 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 120 K 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 interests us. Furthermore, we observe almost negligible overhead of IDO over PA in terms of running time.

Table 1. Results of SWaT.

#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

5 Conclusion and Related Work

In this paper, we proposed an active learning approach to "smartly" tune the system configurations so as to generate traces that can effectively improve our current probability estimation. We prove that our algorithm converges under three existing common estimators. Our experiment results show that our approach effectively improves random sampling in terms of probability estimation and reachability analysis (especially).

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 [14,18,19]. 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). Accessed 24 Nov 2016
3. Mean squared error – Wikipedia, the free encyclopedia (2016). Accessed 7 Dec 2016
4. Angluin, D.: Learning regular sets from queries and counterexamples. Inf. Comput. **75**(2), 87–106 (1987)
5. Baier, C., Katoen, J.-P., et al.: Principles of Model Checking, vol. 26202649. MIT Press, Cambridge (2008)
6. Barsotti, F., De Castro, Y., Espinasse, T., Rochet, P.: Estimating the transition matrix of a markov chain observed at random times. Stat. Probab. Lett. **94**, 98–105 (2014)
7. Brinker, K.: Incorporating diversity in active learning with support vector machines. In: Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), 21–24 August 2003, Washington, DC, USA, pp. 59–66 (2003)
8. Chen, Y., Nielsen, T.D.: Active learning of markov decision processes for system verification. In: 2012 11th International Conference on Machine Learning and Applications (ICMLA), vol. 2, pp. 289–294. IEEE (2012)
9. Cochran, G.: Laplace's ratio estimator. In: David, H.A. (ed.) Contributions to Survey Sampling and Applied Statistics, pp. 3–10. Academic Press, New York (1978)
10. Gale, W.A., Sampson, G.: Good-turing frequency estimation without tears*. J. Quant. Linguist. **2**(3), 217–237 (1995)
11. Gurobi Optimization Incorporation: Gurobi optimizer reference manual (2016)
12. Heidelberger, P.: Fast simulation of rare events in queueing and reliability models. ACM Trans. Model. Comput. Simul. (TOMACS) **5**(1), 43–85 (1995)
13. Lesser, K., Oishi, M.: Reachability for partially observable discrete time stochastic hybrid systems. Automatica **50**(8), 1989–1998 (2014)
14. Mao, H., Chen, Y., Jaeger, M., Nielsen, T.D., Larsen, K.G., Nielsen, B.: Learning probabilistic automata for model checking. In: Eighth International Conference on Quantitative Evaluation of Systems (QEST), pp. 111–120. IEEE (2011)
15. McAllester, D.A., Schapire, R.E.: On the convergence rate of good-turing estimators. In: COLT, pp. 1–6 (2000)
16. Moon, T.K.: The expectation-maximization algorithm. IEEE Sig. Proces. Mag. **13**(6), 47–60 (1996)
17. Tabakov, D., Vardi, M.Y.: Experimental evaluation of classical automata constructions. In: Sutcliffe, G., Voronkov, A. (eds.) LPAR 2005. LNCS (LNAI), vol. 3835, pp. 396–411. Springer, Heidelberg (2005). doi:10.1007/11591191_28
18. Wang, J., Sun, J., Qin, S.: Verifying complex systems probabilistically through learning, abstraction and refinement. arXiv preprint arXiv:1610.06371 (2016)
19. Wang, J., Sun, J., Yuan, Q., Pang, J.: Should we learn probabilistic models for model checking? a new approach and an empirical study. In: Huisman, M., Rubin, J. (eds.) FASE 2017. LNCS, vol. 10202, pp. 3–21. Springer, Heidelberg (2017). doi:10.1007/978-3-662-54494-5_1
20. Whittaker, J.A., Thomason, M.G.: A markov chain model for statistical software testing. IEEE Trans. Softw. Eng. **20**(10), 812–824 (1994)