Verifying the Long-Run Behavior of Probabilistic System Models in the Presence of Uncertainty

Anonymous Author(s)

ABSTRACT

Verifying that a stochastic system is in a certain state when it has reached equilibrium has important applications. For instance, the probabilistic verification of the long-run behavior of a safety-critical system enables assessors to check whether it accepts a human *abort*-command at any time with a probability that is sufficiently high. The stochastic system is represented as probabilistic model, a long-run property is asserted and a probabilistic verifier checks the model against the property.

However, existing probabilistic verifiers do not account for the imprecision of the probabilistic parameters in the model. Due to uncertainty, the probability of any state transition may be subject to small perturbations which can have dire consequences for the veracity of the verification result. In reality, the safety-critical system may accept the *abort*-command with an insufficient probability.

In this paper, we introduce the first probabilistic verification technique that accounts for uncertainty on the verification of long-run properties of a stochastic system. We present a mathematical framework for the asymptotic analysis of the stationary distribution of a discrete-time Markov chain, making *no* assumptions about the distribution of the perturbations. Concretely, our novel technique computes upper and lower bounds on the long-run probability, given a certain degree of uncertainty about the stochastic system.

1 INTRODUCTION

Probabilistic verification is a powerful and mature technology which aims to verify systems that exhibit stochastic rather than deterministic behavior. The stochastic system is represented by a stochastic model, such as a discrete-time Markov chain (DTMC). A probabilistic model checker verifies this model against a given system property, such as the minimum (or maximum) long-run probability or reachability of a certain state. For instance, the life-cycle of software developed in a large company can be modeled as a DTMC where a state represents the current development stage of a software, and the state transition probabilities can be determined empirically from the life-cycle of software developed in the past. A probabilistic model checker, such as PRISM [14], can then check whether the long-run probability that a software is in the (error-free) deployment stage exceeds some threshold.

ESEC/FSE 2018, Lake Buena Vista, Florida, United States

DOI: 10.1145/nnnnnnn.nnnnnnn

However, probabilistic verification does not account for the imprecision of the probabilistic parameters in the model of the stochastic system. Often, the specified transition probabilities are accurate only to some degree. For instance, the transition probabilities in the DTMC model of the company's development life-cycle are computed as the sample mean over past instances. However, a sample mean is subject to variance and approximates the population mean only with *some* accuracy. The company may hire new developers, develop different software, or improve the development process. All of these contribute to our uncertainty about the *specific* probability to transition from one development stage to another.

Previous work has successfully applied perturbation theory to many aspects of probabilistic model checking, including different stochastic formalisms, different kinds of properties to be verified and different measures of perturbation distance, as well as applications of the ideas to self-adaptive software, QoS monitoring and cloud computing [16, 19–22].

This paper presents results on dealing with uncertainty when verifying long-run properties of DTMCs, where we seek to determine the probability that a system—whose stochastic behavior is subject to perturbation—will be in a particular state of interest once the system has reached a steady state, or equilibrium. We propose to model a stochastic system that is subject to uncertainty as a parameterized DTMC where the specific probability in each transition is subject to random perturbations. We introduce a probabilistic verification technique that provides an upper and lower bound on the long-run probability for each state in the parameterized DTMC. The bounds characterize the worst-case consequences of uncertainty on the (long-run) probability that the system is in a certain state at any time after it has reached equilibrium.

The verification of long-run probabilities is a very useful and versatile application of probabilistic model checking and can be applied to problems in software engineering as well as other domains. For instance, it allows us to check whether the probability that a safety-critical system accepts an external *abort*-command at an arbitrary point in time is sufficiently high. It also allows us to check which functions are most critical for a system because they are executed most often, or which web-pages on a server are important because they are visited with greater likelihood.

We introduce the pertinent concepts and evaluate our novel probabilistic verification technique using three case studies. In the first case study, we discuss the probabilistic verification of a long-run property in the development life-cycle of a large mobile app development company. In the second case study, we investigate the impact of estimating the transition probabilities in the DTMC from empirical data (i.e., using maximum-likelihood estimation). Of course, a sample is subject to some variance resulting in small perturbations for the transition probabilities. The DTMC models web-pages on a server, and the verification task is to check the importance of those web-pages given a sample of click-stream data.

In the third case study, we investigate the impact of changes to the program or its workload on determining the functions that are most critical for the program. The DTMC models the sequence of function calls in the program, and the verification task is to check the probability that a function is exercised at any time after equilibrium, given that the program or workload may slightly change.

The main contributions of this work are as follows:

(1) We introduce the first probabilistic verification technique that accounts for uncertainty in the verification of long-run properties of a stochastic system. Our technique uses perturbation analysis to provide upper and lower bounds on the system's long-run properties which represent the worst-case consequences of uncertainty.
(2) We present a mathematical framework for the asymptotic analysis of the stationary distribution of a (reducible or irreducible) DTMC when the transition probabilities are subject to a random perturbation. We make *no* assumptions about the distribution of the perturbations, reflecting the reality that the degree of uncertainty for specific state transitions is often unknown.
(3) We present a prototype implementation of our work in Python. We evaluate our technique on three case studies. Our experiments indicate that our technique is able to provide an accurate estimation of the worst-case consequences of uncertainty on verification of long-run properties in DTMCs.

The remainder of the paper is organized as follows. Section 2 introduces a running example of the life-cycle of software development. An introduction to basic concepts of probabilistic model checking of DTMCs are presented in Section 3. Section 4 explain our technical approach for analyzing the worst-case consequences of the uncertainty on probabilistic verification of long-run properties in DTMCs. Section 5 presents our experimental evaluation and results. Section 6 discusses related work. Finally, Section 7 summarizes our contributions and discusses our future work.

2 MOTIVATING EXAMPLE

2.1 A Markov Chain Model For The Software Development Life-Cycle of Mobile Apps

Throughout the paper, we exemplify the pertinent concepts and approaches based on the software development life-cycle of a large mobile app development company. For this company, each mobile app goes through the following stages:

- **Early Stage** (s_0). The app is created and actively developed. Once the app is feature-complete, it goes to *incubation* (s_1). However, at this stage an app's development can also be *abandoned* (s_3), e.g., for budget reasons.
- **Incubation** (s_1). The app implements all the intended functionality but requires some improvements and fixes. If it turns out that more features or substantial improvements are required, the app goes back to the *early stage* (s_0). If the project is discontinued, the app is *retired* (s_4). However, if the app reaches a certain maturity is goes to *deployment* (s_2). Once it is deployed, it cannot go back to *incubation* (s_1).
- **Deployment** (s_2). The (updated) app is uploaded to the app store and customers start to download and use the app. Deployment is part of the main development life-cycle. If a user reports a bug, the app is marked as *buggy* (s_6).
- **Abandoned** (s_3). The app development is discontinued at the early stage. Once the development of an app is marked as abandoned, it remains abandoned.
- **Retired** (s_4). The app development is discontinued at the incubation stage. Once an app is retired, it remains retired.
- **Repair** (s_5). A developer creates a patch for the buggy app and submits it for peer review. If the patch is accepted, the app goes back to *deployment* (s_2), i.e., the fixed version is uploaded to the app store. If the patch is rejected, the app is again marked as *buggy* (s_6).
- **Buggy** (s_6). The app contains a bug that was reported and needs to be fixed. The app is send for *repair* (s_5).

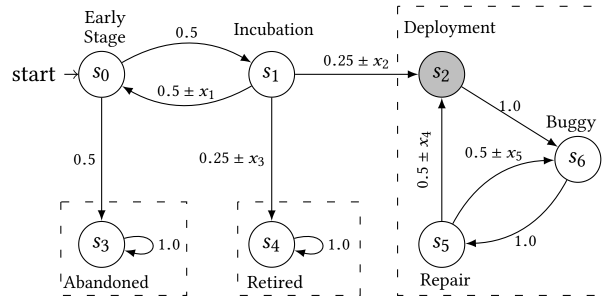

Figure 1: Motivating Example. DTMC Model of the Development Life-Cycle of a Mobile App.

To understand and improve the development life-cycle of their mobile apps, the company records the various stages that each app undergoes. For instance, it can empirically determine the proportion of apps in the *early stage* (s_0) that reach *incubation* (s_1) or that are *abandoned* (s_3). Thus, the development life-cycle can be modeled as a discrete-time Markov chain with each development phase as a *state*, each progression from phase to phase as *transition*, and the proportion of apps progressing from one phase to another as *transition probabilities*. The Markov model for the company's development life-cycle is shown in Figure 1. For now, we ignore the dashed rectangles and we consider the values of x_i in the transition probabilities to be fixed at $x_1 = \ldots = x_5 = 0$.

2.2 Problem Statement

In this work, we investigate the worst-case consequences of uncertainty on the verification of long-run properties in a discrete-time Markov chain. In the development life-cycle of an app, the company would like to understand the probability that an app, in the long-run, will be in *deployment* (s_2), i.e., graduated from incubation and not buggy. In the second column of Table 1, we can see that this probability $\boldsymbol{\pi}_{s_2} = 0.033$. It is about 25 times more likely that an app is abandoned or retired before it reaches the deployment phase ($\boldsymbol{\pi}_{s_3} + \boldsymbol{\pi}_{s_4} = 0.834$).

However, there are several sources of uncertainty in the modeling. For instance, the recorded life-cycles are only samples that are taken from a larger population; future development life-cycles

Table 1: Worst-Case Consequences of Uncertainty in the Development Life-Cycle of Mobile Apps where $\delta \leq 0.001$.

Stages	$\boldsymbol{\pi}_{s_i}$ ($\delta = 0$)	κ_{s_i}	Linear Bounds	
			$\boldsymbol{\pi}_{s_i} - \kappa_{s_i}\delta$	$\boldsymbol{\pi}_{s_i} + \kappa_{s_i}\delta$
s_0	0.000	0.000	0.0000	0.0000
s_1	0.000	0.000	0.0000	0.0000
s_2	0.033	0.067	0.0329	0.0331
s_3	0.667	0.222	0.6667	0.6672
s_4	0.167	0.333	0.1667	0.1673
s_5	0.067	0.133	0.0668	0.0671
s_6	0.067	0.133	0.0668	0.0671

may not be well-represented by the recorded ones. The company may have changed the development process since beginning of the recordings; current life-cycles may not be well-represented by earlier ones. Hence, the empirically determined transition probabilities are accurate only to some degree. What does that mean for computing the probability $\boldsymbol{\pi}_{s_2}$ that an app, in the long run, will be in the *deployment* stage (s_2)?

We model the uncertainty in the transition probabilities as small perturbations in a parameterized Markov chain, where the (symbolic) perturbation vector $\vec{x}$ is added to the transition probabilities. Figure 1 depicts the Markov chain for our motivating example parameterized with the perturbation vector $\vec{x} = \langle x_1, x_2, x_3, x_4, x_5 \rangle$.

We do *not* assume any knowledge about the (distribution of) values for $\vec{x}$. This reflects the reality that we cannot know the degree of uncertainty for specific state transitions. However, we assume that the total perturbation distance $\delta = \sum_{x_i \in \vec{x}} |x_i|$ for all parameterized state transitions can be provided (as an upper bound on the total uncertainty in the stochastic process).

Due to uncertainty, the verification of long-run properties in a Markov chain is (1) correct only within certain accuracy bounds, and (2) sensitive to uncertainty in certain states more than in others. Our approach quantifies the accuracy of the computed probability that an app, in the long run, will be in the *deployment* stage (s_2) in the form of asymptotic bounds, called *linear perturbation bounds*. Our approach will also quantify the sensitivity of the computed probability to uncertainty in each transition in the form of *condition numbers*.

For our motivating example, the company is interested in the worst-case consequences of uncertainty in the modeling of the development life-cycle if the total uncertainty was at least $\delta = 0.001$, i.e, $|x_1| + |x_2| + |x_3| + |x_4| + |x_5| \leq 0.001$. For this purpose, in the third column of Table 1, we compute a condition number κ_{s_i} that captures the effect of perturbation vector $\vec{x}$ in the computation of $\boldsymbol{\pi}_{s_i}$ such that $0 \leq i \leq 6$. As can be seen from Table 1, the greatest condition number in the model of the development life-cycle of a mobile app corresponds to the probability that an app, in the long run, will be *retired* (s_4). In other words, the long run probability of being in s_4 is the most sensitive to the perturbation vector $\vec{x}$. Based on the condition number κ_{s_i} and $\delta = 0.001$, the last column of Table 1 provides an estimate of the worst-case consequences of uncertainty in the modeling of the development life-cycle calculated as $\pm\kappa_{s_i}\delta$ and named as *linear perturbation bounds*. For instance, the linear perturbation bounds that estimate the worst-case consequences of uncertainty on verification that an app, in the long run, will be in *deployment* stage (s_2) are calculated as $\boldsymbol{\pi}_{s_2} \pm \kappa_{s_2}\delta = [0.0329, 0.0331]$.

3 BACKGROUND

3.1 Discrete-Time Markov Chains

A discrete-time Markov chain (DTMC) represents the stochastic behavior of a probabilistic system. Specifically, a DTMC is a transition system with probability distributions for the successors of each state. Formally, a DTMC is defined as follows:

Definition 1. *Discrete-Time Markov Chain (DTMC). A DTMC is a tuple $\mathcal{D} = (S, \mathbf{P}, S_{init}, AP, L)$ where S is a finite set of states, $\mathbf{P} : S \times S \rightarrow [0,1]$ is a probabilistic transition function such that $\forall s \in S, \sum_{t \in S} \mathbf{P}(s,t) = 1$, $S_{init} \subseteq S$ is a set of initial states, AP is a finite set of atomic propositions, and $L : S \rightarrow 2^{AP}$ a labeling function.*

In addition, we define $\iota_{\text{init}} : S \rightarrow [0,1]$ as the initial state distribution, such that $\sum_{s \in S} \iota_{\text{init}}(s) = 1$ and $\forall t \notin S_{init}, \iota_{\text{init}}(t) = 0$. For instance, in Figure 1 we have $S = \{s_i \mid 0 \leq i \leq 6\}$, $S_{\text{init}} = \{s_0\}$, $AP = \{$early stage, incubation, deployment, abandoned, retired, repair, buggy$\}$, and for instance $L(s_0) =$ early stage and $\mathbf{P}(s_0, s_1) = 0.5$. Furthermore, $\iota_{\text{init}}(s_0) = 1$ and $\iota_{\text{init}}(s_i) = 0$ for $i > 0$.

Definition 2. *Sub-DTMC. Let $\mathcal{D} = (S, \mathbf{P}, S_{init}, AP, L)$ be a DTMC. A sub-DTMC of $\mathcal{D}$ is a tuple $(S', \mathbf{P}')$ such that $\emptyset \neq S' \subseteq S$, and $\mathbf{P}' : S' \rightarrow [0,1]$ where for all $s, s' \in S', \mathbf{P}'(s,s') = \mathbf{P}(s,s')$ and $\sum_{s' \in S'} \mathbf{P}'(s,s') = 1$.*

The digraph of $\mathcal{D}$, denoted as $G_{(S,E)}$, is induced as there is one node for each state $s \in S$ and a direct edge $(s,t) \in E$ if only if $\mathbf{P}(s,t) > 0$. An infinite path in a DTMC $\mathcal{D}$ is a sequence of the form $\pi = s_0 s_1 s_2 \ldots$ where $s_i \in S$ and $\mathbf{P}(s_i, s_{i+1}) > 0$ for all $i \geq 0$. A finite path ϖ is a prefix of an infinite path π ending in a particular state where $\varpi = s_0 s_1 \ldots s_n$ such that $n = |\varpi|$. In this context, we say a state s_n is *reachable* from s_0 if there is a finite path ϖ.

A set of states $S' \subseteq S$ induces a strongly connected subgraph (SCS) of a DTMC $\mathcal{D}$ if only if for all $s, t \in S'$ there is a path from s to t visiting only states from S'. A strongly connected component (SCC) of $\mathcal{D}$ is a maximal (w.r.t. $\subseteq$) SCS of S. A bottom strongly connected component (BSCC) of $\mathcal{D}$ is an SCC S' from which no state outside S' is reachable. We denote BSCC($\mathcal{D}$) as the set of all BSCCs of the underlying digraph of $\mathcal{D}$.

A DTMC $\mathcal{D}$ is said *irreducible* if all its states belong to a single BSCC. Irreducibility of a DTMC is important for convergence to equilibrium as $n \rightarrow \infty$, because the convergence should be independent of any start state. In case, a DTMC is not irreducible, we say that is *reducible*. The DTMC in our motivating example is reducible, as the states belong to three BSCCs shown as dashed boxes in Figure 1.

Given a DTMC $\mathcal{D}$, the period of state s is defined as $d(s) = \gcd\{n \in \mathbb{N}_+ : \mathbf{P}^n(s,s) > 0\}$. State s is *aperiodic* if $d(s) = 1$ and *periodic* if $d(s) > 1$. In this context, a DTMC $\mathcal{D}$ is said to be *aperiodic* if all its states are aperiodic. Correspondingly, if all states of $\mathcal{D}$ are periodic then $\mathcal{D}$ is *periodic*.

3.2 Parametric Discrete-Time Markov Chains

Inspired by previous studies [8, 15, 21, 22], we use a parametric model to integrate the uncertainty which is expressed as small perturbations to transition probabilities in the DTMC.

Definition 3. *Parametric discrete-time Markov chain (PDTMC). Let $\mathcal{D} = (S, \mathbf{P}, S_{init}, AP, L)$ be a DTMC and $\vec{x} = \langle x_1, \ldots, x_m \rangle$ be a perturbation vector. A PDTMC is a tuple $\mathcal{D}[\vec{x}] = (S, \mathbf{P}[\vec{x}], S_{init}, AP, L)$ where $\mathbf{P}[\vec{x}]$ is a parametric transition function based on $\mathbf{P}$, and S, S_{init}, AP and L are defined as Definition 1.*

Figure 1 depicts a parameterized DTMC where $\vec{x} = \langle x_1, x_2, x_3, x_4, x_5 \rangle$.

3.3 Long-Run Properties in DTMCs

In probabilistic model checking of DTMCs, long-run, also named as steady-state, properties are used to analyze the reliability of probabilistic systems and to obtain performance parameters such as throughput, delay, loss probability, etc.

Formally, given a DTMC $\mathcal{D}$, the long-run properties study the limit behavior of the probability vector $\boldsymbol{\pi}^{\mathcal{D}}(t) = [\boldsymbol{\pi}_s^{\mathcal{D}}(t)]_{s \in S}$ when time tends to infinity ($\lim_{t \to \infty} \boldsymbol{\pi}^{\mathcal{D}}(t)$) [5]. This limit when exists is called *steady-state probability vector*, and it is written as $\boldsymbol{\pi}^{\mathcal{D}} = [\boldsymbol{\pi}_s^{\mathcal{D}}]_{s \in S}$. Intuitively, we can interpret $\boldsymbol{\pi}_s^{\mathcal{D}}$ as the long-run mean fraction of time the DTMC $\mathcal{D}$ is in state s.

In this paper we are interested in the computation of the long-run probability of being in a particular state s having started in initial states of a given DTMC $\mathcal{D}$. We denote this probability as $\boldsymbol{\pi}_s^{\mathcal{D}}$. For convenience, we simply mention $\boldsymbol{\pi}_s$ instead of $\boldsymbol{\pi}_s^{\mathcal{D}}$ when $\mathcal{D}$ is clear in the context.

To compute long-run probabilities in DTMCs, previous studies [5, 18] have demonstrated that the steady-state probability vector exists if the DTMC is irreducible and aperiodic. Unfortunately, it is not that case in all probabilistic models. Thus, to avoid with periodic considerations, we use the long-run probability for computing the steady-state probability vector $\boldsymbol{\pi}^{\mathcal{D}}$ of a DTMC $\mathcal{D}$. Consequently, the procedure for computing the long-run probability $\boldsymbol{\pi}_s^{\mathcal{D}}$ is based on the following conditions:

3.3.1 DTMC $\mathcal{D}$ is irreducible. Let $\mathcal{D} = (S, \mathbf{P}, S_{init}, AP, L)$ be an irreducible DTMC and given a particular state $s \in S$, we compute the long-run probability $\boldsymbol{\pi}_s^{\mathcal{D}}$ as follows:

$$\boldsymbol{\pi}_s^{\mathcal{D}} = \sum_{s' \in S} \Big\{ \lim_{n \to \infty} \frac{1}{n} \sum_{k=1}^{n} \mathbf{P}^k(s', s) \Big\}, \tag{1}$$

where $\mathbf{P}^k(s', s)$ denotes the probability of DTMC $\mathcal{D}$ being in state s after exactly k transitions when starting in state s'.

3.3.2 DTMC $\mathcal{D}$ is reducible. Let $\mathcal{D} = (S, \mathbf{P}, S_{init}, AP, L)$ be a reducible DTMC and given a particular state $s \in S$. To compute the long-run probability $\boldsymbol{\pi}_s^{\mathcal{D}}$, we first identify the set of all BSCCs of the underlying digraph of $\mathcal{D}$, denoted as $BSCC(\mathcal{D})$. Then, we evaluate whether state s belongs to any BSCC $K \in BSCC(\mathcal{D})$.

- In the affirmative case, the probability $\boldsymbol{\pi}_s^{\mathcal{D}}$ is the combination of i) the probability of reaching the BSCC K having started in the initial states of the model, denoted as $Pr^{\mathcal{D}}(\Diamond K)$, and ii) the long-run probability of state s in a sub-DTMC $\mathcal{D}(K) = (K, \mathbf{P}_K)$ of $\mathcal{D}$, induced by BSCC K, where $\mathbf{P}_K = (\mathbf{P}(s,t))_{s,t \in K}$. In short,
$$\boldsymbol{\pi}_s^{\mathcal{D}} = Pr^{\mathcal{D}}(\Diamond K) \cdot \boldsymbol{\pi}_s^{\mathcal{D}(K)}. \tag{2}$$
Since every BSCC is irreducible, we follow Equation 1 for computing the probability $\boldsymbol{\pi}_s^{\mathcal{D}(K)}$.
- Otherwise, the long-run probability $\boldsymbol{\pi}_s^{\mathcal{D}} = 0$.

4 PERTURBATION APPROACH

In this section, we present our technical approach for estimating the worst-case consequences of uncertainty on the probabilistic verification of long-run properties in DTMCs based on perturbation analysis. We first use a parameterized DTMC to model the presence of uncertainty, which is expressed as small perturbations to transition probabilities. Then, we define a *variation function* that presents a mathematical characterization of the perturbation in the verification of long-run properties against a given DTMC. In particular, this function captures the difference between the verification of a long-run property against a perturbed DTMC and its associated unperturbed DTMC. Lastly, we look for the linear fragment of the variation function that provides a useful approximate solution of this function. Based on this linear fragment and using the concept of condition numbers, we compute *linear perturbation bounds* that reveal the worst-case consequences of perturbations on the verification of long-run properties in DTMCs.

4.1 Dealing with Uncertainty

Let $\mathcal{D} = (S, \mathbf{P}, S_{\text{init}}, AP, L)$ be the *unperturbed* DTMC. We first identify the probabilistic transitions that are vulnerable to perturbations. Then, we capture the presence of these perturbations by using a perturbation vector $\vec{x} = \langle x_1, \ldots, x_m \rangle$ where m is the total number of perturbation variables in $\vec{x}$. Based on the unperturbed transition function $\mathbf{P}$, we incorporate the perturbation vector $\vec{x}$ into the parametric probabilistic transition $\mathbf{P}[\vec{x}]$, which is defined as follows:

Definition 4. *Parametric Probabilistic Transition Function. Given a perturbation vector $\vec{x} = \langle x_1, \ldots, x_m \rangle$ and a non-parametric probabilistic transition $\mathbf{P}$, $\mathbf{P}[\vec{x}]$ is the parametric probabilistic transition function such that $\sum_{s,t \in S} \mathbf{P}[\vec{x}](s,t) = 1$ and each entry $\mathbf{P}[\vec{x}](s,t)$ of $\mathbf{P}[\vec{x}]$ is defined as follows:*

$$\mathbf{P}[\vec{x}](s,t) = \begin{cases} \mathbf{P}(s,t) & \text{if } \mathbf{P}(s,t) = 0 \text{ or } \mathbf{P}(s,t) = 1 \\ \mathbf{P}(s,t) + x_i & \text{if there exists } x_i \in \vec{x} \text{ such that } vuln(s,t,x_i) \\ \mathbf{P}(s,t) & \text{otherwise} \end{cases}$$

where $vuln(s,t,x_i)$ is true if the transition from state s to state t is vulnerable to perturbation and $x_i \in \vec{x}$ represents the perturbation for the transition from s to t.

Note that a perturbed entry of the form $\mathbf{P}(s,t) + x_i$ is composed by the non-symbolic part $\mathbf{P}(s,t)$, and the perturbation variable $x_i \in \vec{x}$, which captures the presence of uncertainty.

Proposition 1. *Given a vector $\vec{x} = \langle x_1, \ldots, x_m \rangle$, let $\mathscr{I}$ be a partition on $\{1, \ldots, m\}$. A sub-vector $\vec{y} = \langle x_j, \ldots, x_k \rangle$ is an independent perturbed sub-vector in the partition $\mathscr{I}$ such that $1 \le j < k \le m$. For all $\vec{y} \in \mathscr{I}$, $\sum_{t \in S} \mathbf{P}[\vec{x}](s,t) = \sum_{t \in S} \mathbf{P}(s,t) + \sum_{x_i \in \vec{y}} x_i$ for some $s \in S$.*

Proposition 1 states that each element of $\vec{x}$ falls into an *independent* perturbed sub-vector $\vec{y}$ and all variables of a sub-vector

$\vec{y}$ are used. On the other hand, it is always assumed that the vector $\vec{x} = \langle x_1, \ldots, x_m \rangle$ is within the set $\mathcal{U} := \{\vec{x} \in \mathbb{R}^m \mid \forall \vec{y} \in \mathscr{I}, \sum_{x_i \in \vec{y}} x_i = 0, \mathbf{P}[\vec{x}] \equiv \mathbf{P}\}$. Note that we say $\mathbf{P}[\vec{x}] \equiv \mathbf{P}$ when they have exactly the same non-zero entries. In this context, we refer the parametric DTMC $\mathcal{D}[\vec{x}]$ (Definition 3) as *the perturbed* version of DTMC $\mathcal{D}$.

To illustrate the definitions above, consider again the DTMC model of the development life-cycle of a mobile app presented in Figure 1. In the following, we denote this DTMC as $\mathcal{D}^{lc}$. As we introduced in Section 2, there are several sources of uncertainty in the modeling of $\mathcal{D}^{lc}$, which are expressed as small perturbation in the probabilistic transitions of the model. As Figure 1 shows, the uncertainty in the model is depicted by the perturbation vector $\vec{x} = \langle x_1, x_2, x_3, x_4, x_5 \rangle$. Note that we only incorporated variables into transitions between 0 and 1. The presence of $\vec{x}$ in $\mathcal{D}^{lc}$ has as a consequence the definition of PMC $\mathcal{D}^{lc}[\vec{x}]$ where the only difference between both models is the parametric function $\mathbf{P}[\vec{x}]$.

Based on Proposition 1, each perturbation variable $x_i \in \vec{x}$ falls into two independent perturbed sub-vectors: $\vec{y_1} = \langle x_1, x_2, x_3 \rangle$ and $\vec{y_2} = \langle x_4, x_5 \rangle$. As a result, we have $\mathscr{I} = \{\vec{y_1}, \vec{y_2}\}$. Note that each sub-vector is associated to the outgoing transitions of a perturbed state. For example, sub-vectors $\vec{y_1}$ and $\vec{y_2}$ are associated to states s_1 and s_2, respectively.

4.2 Variation Function for Long-Run Properties in DTMCs

This section presents a mathematical characterization of the perturbation on verification of long-run properties in DTMCs. Recall that a variation function captures the difference between the computation of a long-run probability in a perturbed and unperturbed model. Following the computation of long-run probabilities described in Section 3.3, we present two variation functions: i) for irreducible DTMCs, and ii) for reducible DTMCs.

Definition 5. *Variation Function - Irreducible Case. Let $\mathcal{D} = (S, \mathbf{P}, S_{init}, AP, L)$ and $\mathcal{D}[\vec{x}] = (S, \mathbf{P}[\vec{x}], S_{init}, AP, L)$ be an irreducible DTMC and its parametric irreducible variant, respectively. A variation function of $\mathcal{D}[\vec{x}]$ against the long-run probability of a particular state $s \in S$ is $\sigma : (S, \mathcal{U}) \rightarrow [0, 1]$ defined as follows:*

$$\sigma(s, \vec{x}) = \boldsymbol{\pi}_s^{\mathcal{D}[\vec{x}]} - \boldsymbol{\pi}_s^{\mathcal{D}}. \tag{3}$$

Before presenting the variation function for reducible DTMCs, let us recall that given an unperturbed reducible DTMC $\mathcal{D}$, the computation of $\boldsymbol{\pi}_s^{\mathcal{D}} \neq 0$ if and only if $s \in K, K \in \mathrm{BSCC}(\mathcal{D})$. The value of $\boldsymbol{\pi}_s^{\mathcal{D}}$ depends on two components:

(1) The reachability probability from initial states to BSCC K, denoted as $Pr^{\mathcal{D}}(\diamond K)$.
(2) The long-run probability of state s in sub-DTMC $\mathcal{D}(K)$ if and only if $s \in$ BSCC K. This probability is denoted as $\boldsymbol{\pi}_s^{\mathcal{D}(K)}$.

Under those circumstances, we define the variation function for reducible DTMCs as follows:

Definition 6. *Variation Function - Reducible Case. Let $\mathcal{D} = (S, \mathbf{P}, S_{init}, AP, L)$ and $\mathcal{D}[\vec{x}] = (S, \mathbf{P}[\vec{x}], S_{init}, AP, L)$ be a reducible DTMC and its parametric reducible variant, respectively. Let $K \subset S$ be a BSCC. A variation function of reducible $\mathcal{D}[\vec{x}]$ with respect to the long-run probability of a particular state $s \in K$ is $\tau : (S, \mathcal{U}) \rightarrow [0, 1]$ defined as follows:*

$$\tau(s, \vec{x}) = Pr^{\mathcal{D}[\vec{x}]}(\diamond K) \cdot \boldsymbol{\pi}_s^{\mathcal{D}(K)[\vec{x}]} - Pr^{\mathcal{D}}(\diamond K) \cdot \boldsymbol{\pi}_s^{\mathcal{D}(K)}, \tag{4}$$

where $\mathcal{D}(K)[\vec{x}]$ and $\mathcal{D}(K)$ represent the perturbed and unperturbed sub-DTMC of $\mathcal{D}[\vec{x}]$ and $\mathcal{D}$, respectively.

4.3 Linear Perturbation Bounds

The key idea behind perturbation theory is to reduce a hard problem into an infinite sequence of relatively simple ones. We compute the linear perturbation bounds that capture the verification effect of uncertainty in the form of condition numbers. These bounds provide a useful approximate solution to the variation function.

We define the linear perturbation bounds i) for irreducible DTMCs, and ii) for reducible DTMCs. For each case, we first define the linear fragment of its corresponding variation function. Then, we define the condition number based on the linear fragment. Last, we compute the linear perturbation bounds based on the condition number and the total perturbation distance $\delta > 0$ of the model.

4.3.1 Irreducible Case. Let $\sigma(s, \vec{x})$ be the variation function for irreducible DTMCs with respect to the long-run probability of a particular state s (Definition 5). The linear fragment of the variation function $\sigma(s, \vec{x})$ is defined as follows:

Theorem 4.1. *Linear Fragment of $\sigma(s, \vec{x})$ - Irreducible Case. Let $\sigma(s, \vec{x})$ be the variation function for irreducible DTMCs. For any $\vec{x} \in \mathcal{U}$, the linear fragment σ_1 of σ is formulated as follows:*

$$\sigma_1(s, \vec{x}) = \sum_{s' \in S} \Big\{ \lim_{n \to \infty} \frac{1}{n} \sum_{k=1}^{n} \sum_{i+j=k-1} (\mathbf{P}^j \mathbf{P} \bigstar \mathbf{P}^i)(s', s) \Big\}, \tag{5}$$

where $\mathbf{P}\bigstar = \mathbf{P}[\vec{x}] - \mathbf{P}$.

Proof. Let us start by analyzing the variation function $\sigma(s, \vec{x}) = \boldsymbol{\pi}_s^{\mathcal{D}[\vec{x}]} - \boldsymbol{\pi}_s^{\mathcal{D}}$ (Equation 3). Using the solution of the unperturbed $\boldsymbol{\pi}_s^{\mathcal{D}}$ as an approximation to the solution of the perturbed $\boldsymbol{\pi}_s^{\mathcal{D}[\vec{x}]}$, we can express the variation function $\sigma(s, \vec{x})$ as follows:

$$\sigma(s, \vec{x}) = \sum_{s' \in S} \lim_{n \to \infty} \frac{1}{n} \hat{\mathbf{P}}(s', s), \text{ where } \hat{\mathbf{P}} = \sum_{k=1}^{n} \mathbf{P}[\vec{x}]^k - \sum_{k=1}^{n} \mathbf{P}^k. \tag{6}$$

Let $\mathbf{P}\bigstar = \mathbf{P}[\vec{x}] - \mathbf{P}$ and $\delta = \sum_{x_i \in \vec{x}} |x_i|$ be the total perturbation distance. By expanding the term $\hat{\mathbf{P}}$ of Equation 6, we have:

$$\sum_{k=1}^{n} \mathbf{P}[\vec{x}]^k - \sum_{k=1}^{n} \mathbf{P}^k = \sum_{k=1}^{n} (\mathbf{P}\bigstar + \mathbf{P})^k - \sum_{k=1}^{n} \mathbf{P}^k \tag{7}$$

As we look for the linear fragment of $\sigma(s, \vec{x})$, we ignore any term of more one copy of $\mathbf{P}\bigstar$ in Equation 7. As a result, we have:

$$\sum_{k=1}^{n} \mathbf{P}[\vec{x}]^k - \sum_{k=1}^{n} \mathbf{P}^k = \sum_{k=1}^{n} \sum_{i+j=k-1} (\mathbf{P}^j \mathbf{P}\bigstar \mathbf{P}^i) + O(\delta^2). \tag{8}$$

where $x_i \sim O(\delta)$. Based on previous equation, we conclude linear fragment σ_1 as Equation 5. □

Based on the above linear fragment σ_1, we define the condition number and the linear perturbation bounds for irreducible DTMCs as follows:

Definition 7. *Condition number - Irreducible Case. Let $\vec{h}^T$ denote the transpose of vector $\vec{h}$. Let $\vec{x} \in \mathcal{U}$ be a perturbation vector. We can write $\sigma_1(s, \vec{x})$ in the form $\sigma_1(s, \vec{x}) = \vec{x} \cdot \vec{h}^T$ for some constant vector $\vec{h}$. Thus, we can define the condition number κ_s^{irr} for irreducible DTMCs as follows:*

$$\kappa_s^{irr} = \frac{1}{2} \max_{x_i, x_j \in \vec{y}, \vec{y} \in \mathscr{I}} (\vec{h}_i - \vec{h}_j). \tag{9}$$

Definition 8. *Linear Perturbation Bound - Irreducible Case. Let $\delta = \sum_{x_i \in \vec{x}} |x_i|$ be the total perturbation distance. Let $\boldsymbol{\pi}_s^{\mathcal{D}}$ and κ_s^{irr} be the unperturbed long-run probability and the condition number of a particular state s in an irreducible DTMC $\mathcal{D}$, respectively. A pair of upper and lower linear perturbation bounds for the variation function $\sigma(s, \vec{x})$ are defined as follows:*

$$f_{irr}^{+} = \boldsymbol{\pi}_s^{\mathcal{D}} + \kappa_s^{irr} \delta. \tag{10}$$

$$f_{irr}^{-} = \boldsymbol{\pi}_s^{\mathcal{D}} - \kappa_s^{irr} \delta. \tag{11}$$

As an illustration, consider again PDTMC $\mathcal{D}^{lc}[\vec{x}]$ shown in Figure 1. As the figure shows, the dashed rectangles represent the set of BSCCs of the model: $K_1 = \{s_2, s_5, s_6\}$, $K_2 = \{s_3\}$ and $K_3 = \{s_4\}$. Suppose we are interested to estimate the worst-case effect of perturbed sub-vector $\vec{y_2} = \langle x_4, x_5 \rangle$ on the verification of the probability that an app, in the long run, will be in *deployment* (s_2). Since $s_2 \in K_1$, we only focus on BSCC K_1 that induces a sub-PDTMC $\mathcal{D}[\vec{y_2}](K_1) = (K_1, \mathbf{P}[\vec{y_2}]_{K_1})$ where $\mathbf{P}[\vec{y_2}]_{K_1} = (\mathbf{P}[\vec{x}](s,t))_{s,t \in K_1}$ and $\vec{y_2} \subset \vec{x}$. It is important to notice that the sub-PDTMC $\mathcal{D}[\vec{y_2}](K_1)$ is irreducible.

Under those circumstances, we follow the computation of linear perturbation bounds for irreducible DTMCs. First, we define the variation function $\sigma(s_2, \vec{y_2}) = \boldsymbol{\pi}_{s_2}^{\mathcal{D}[\vec{y_2}](K_1)} - \boldsymbol{\pi}_{s_2}^{\mathcal{D}(K_1)}$ (Definition 5). Then, by Theorem 4.1, we calculate $\sigma_1(s_2, \vec{y_2}) = 1.166x_4 + 0.166x_5$ as the linear fragment of σ. Next, we compute $\kappa_{s_2}^{irr} = (1.166 - 0.166)/2 = 0.5$ (Definition 7). Since the company is interested in the worst-case effect of uncertainty in the modeling of the development life-cycle if the total uncertainty was at least $\delta = 0.001$, we compute the upper bound $f_{irr}^{+} = 0.199 + (0.5)(0.001) = 0.1995$, and the lower bound $f_{irr}^{-} = 0.199 - (0.5)(0.001) = 0.1985$. Note that we consider $|x_4| + |x_5| \leq 0.001$.

4.3.2 Reducible Case. Following the same strategy as for the irreducible case, we start by defining the linear fragment of the variation function $\tau(s, \vec{x})$ for reducible DTMCs as follows:

Theorem 4.2. *Linear Fragment of $\tau(s, \vec{x})$ - Reducible Case. Let $s \in$ BSCC K. Let $\tau(s, \vec{x})$ be the variation function for reducible DTMCs with respect to long-run property of a particular state s. For any $\vec{x} \in \mathcal{U}$, the linear fragment of $\tau(s, \vec{x})$ is defined as follows:*

$$\tau_1(s, \vec{x}) = Pr^{\mathcal{D}}(\diamond K) \cdot \sigma_1(s, \vec{x}) + \rho_1(\vec{x}) \cdot \boldsymbol{\pi}_s^{\mathcal{D}(K)}, \tag{12}$$

where:

- *$Pr_{\mathcal{D}}(\diamond K)$ is the reachability probability from initial states to BSCC K,*
- *$\sigma_1(s, \vec{x})$ is the linear fragment of variation function for irreducible DTMCs w.r.t. long-run probability of s, presented in Theorem 4.1,*
- *$\rho_1(\vec{x}) = \iota_{init} \mathbf{A}^{\star}(\mathbf{A}'[\vec{x}]\mathbf{A}^{\star}\mathbf{b} + \mathbf{b}'[\vec{x}])$ is the linear fragment of a variation function $\rho_i(\vec{x})$ for reachability properties in DTMCs, and*
- *$\boldsymbol{\pi}_s^{\mathcal{D}(K)}$ is the long-run probability of state s in sub-DTMC $\mathcal{D}(K)$ induced by K.*

Proof. Let us analyze the variation function $\tau(s, \vec{x})$ presented in Equation 4. The first term of $Pr^{\mathcal{D}[\vec{x}]}(\diamond K)$ was studied by Su *et al.* [20]. The authors presented the following expansion:

$$Pr^{\mathcal{D}[\vec{x}]}(\diamond K) = Pr^{\mathcal{D}}(\diamond K) + \rho_i(\vec{x}), \tag{13}$$

where $\rho_i(\vec{x}) = \underbrace{\mathbf{A}^{\star}\mathbf{A}'[\vec{x}] \dots \mathbf{A}^{\star}\mathbf{A}'[\vec{x}]}_{\text{i-1 copies of } \mathbf{A}^{\star}\mathbf{A}'[\vec{x}]}(\mathbf{A}^{\star}\mathbf{A}'[\vec{x}]\mathbf{A}^{\star}\mathbf{b} + \mathbf{A}^{\star}\mathbf{b}'[\vec{x}])$ for each $i \geq 1$ such that $\mathbf{A} = (\mathbf{P}(s', t))_{s', t \in S \setminus K}$, $\mathbf{A}[\vec{x}] = (\mathbf{P}[\vec{x}](s', t))_{s', t \in S \setminus K}$, $\mathbf{b} = (\mathbf{P}(s', K))_{s' \in (S \setminus K)}$, $\mathbf{b}[\vec{x}] = (\mathbf{P}[\vec{x}](s', K))_{s' \in (S \setminus K)}$, $\mathbf{A}^{\star} = (\mathbf{I} - \mathbf{A})^{-1}$, $\mathbf{A}'[\vec{x}] = \mathbf{A}[\vec{x}] - \mathbf{A}$ and $\mathbf{b}'[\vec{x}] = \mathbf{b}[\vec{x}] - \mathbf{b}$.

By substituting the above expansion into Equation 4 , we can rewrite the variation function $\tau(s, \vec{x})$ as follows:

$$\tau(s, \vec{x}) = Pr^{\mathcal{D}}(\diamond K)[\boldsymbol{\pi}_s^{\mathcal{D}(K)[\vec{x}]} - \boldsymbol{\pi}_s^{\mathcal{D}(K)}] + \rho_i(\vec{x}) \cdot \boldsymbol{\pi}_s^{\mathcal{D}(K)[\vec{x}]} \tag{14}$$

It is important to note that the term $\boldsymbol{\pi}_s^{\mathcal{D}(K)[\vec{x}]} - \boldsymbol{\pi}_s^{\mathcal{D}(K)}$ in Equation 14 is the definition of variation function for *irreducible* DTMCs, denoted as $\sigma(s, \vec{x})$ (Definition 5). Thus, by substituting $\sigma(s, \vec{x})$ into Equation 14, we have:

$$\tau(s, \vec{x}) = Pr^{\mathcal{D}}(\diamond K) \cdot \sigma(s, \vec{x}) + \rho_i(\vec{x}) \cdot \boldsymbol{\pi}_s^{\mathcal{D}(K)[\vec{x}]}. \tag{15}$$

In the above equation, the term $\boldsymbol{\pi}_s^{\mathcal{D}(K)[\vec{x}]}$ is defined as:

$$\boldsymbol{\pi}_s^{\mathcal{D}(K)[\vec{x}]} = \sum_{s' \in S} \left\{ \lim_{n \to \infty} \frac{1}{n} \sum_{k=1}^{n} \mathbf{P}_K[\vec{x}]^k(s', s) \right\}. \tag{16}$$

Let $\mathbf{P}_K^{\star} = \mathbf{P}_K[\vec{x}] - \mathbf{P}_K$ and $\delta = \sum_{x_i \in \vec{x}} |x_i|$ be the total perturbation distance. In similar way than the irreducible case, expanding the term $\sum_{k=1}^{n} \mathbf{P}_K[\vec{x}]^k$, we have:

$$\sum_{k=1}^{n} \mathbf{P}_K[\vec{x}]^k = \sum_{k=1}^{n} (\mathbf{P}_K^{\star} + \mathbf{P}_K)^k \tag{17}$$

$$= \sum_{k=1}^{n} \mathbf{P}_K^k + \sum_{k=1}^{n} \sum_{\substack{i,j \geq 0 \\ i+j=k-1}} (\mathbf{P}^j \mathbf{P}_K^{\star} \mathbf{P}^i) + O(\delta^2), \tag{18}$$

where $x_i \sim O(\delta)$. It is important for calculating the linear fragment of $\tau_1(s, \vec{x})$ to ignore any term of more one copy of $\mathbf{P}_K^{\star}$ in Equation 17. Combining the above equations and Equation 15, we conclude as linear fragment of $\tau(s, \vec{x})$ as Equation 12. □

Based on linear fragment $\tau_1(s, \vec{x})$, we define the condition number and the linear perturbation bound for reducible DTMCs as follows:

Definition 9. *Condition Numbers - Reducible Case. Let $\vec{h}^T$ denote the transpose of vector $\vec{h}$. Let $\vec{x} \in \mathcal{U}$ be a perturbation vector. We can write $\tau_1(s, \vec{x})$ in the form $\tau_1(s, \vec{x}) = \vec{x} \cdot \vec{h}^T$ for some constant vector $\vec{h}$. Thus, we can define the condition number κ_s^{red} for reducible DTMCs as follows:*

$$\kappa_s^{red} = \frac{1}{2} \max_{x_i, x_j \in \vec{y}, \vec{y} \in \mathscr{I}} (\vec{h}_i - \vec{h}_j). \tag{19}$$

Definition 10. *Linear Perturbation Bound - Reducible Case. Let $\delta = \sum_{x_i \in \vec{x}} |x_i|$ be the total perturbation distance. Let $\boldsymbol{\pi}_s^{\mathcal{D}}$ and κ_s^{red} be the unperturbed long-run probability and the condition number of a particular state s in a reducible DTMC $\mathcal{D}$, respectively. A pair of* *upper and lower linear perturbation bounds for the variation function $\tau(s, \vec{x})$ are defined as follows:*

$$f_{red}^{+} = \boldsymbol{\pi}_s^{\mathcal{D}} + \kappa_s^{red}\delta. \tag{20}$$

$$f_{red}^{-} = \boldsymbol{\pi}_s^{\mathcal{D}} - \kappa_s^{red}\delta. \tag{21}$$

To exemplify the computation of linear perturbation bounds in reducible DTMCs, consider again the *reducible* PDTMC $\mathcal{D}^{lc}[\vec{x}]$ with perturbation vector $\vec{x} = \langle x_1, x_2, x_3, x_4, x_5\rangle$ in Figure 1. Suppose again we are interested to estimate the worst-case effect of $\vec{x}$ on verification of the long-run probability that an app will be in *deployment* (s_2) in the PDTMC $\mathcal{D}^{lc}[\vec{x}]$. Recall the set of BSCCs are $K_1 = \{s_2, s_5, s_6\}$, $K_2 = \{s_3\}$ and $K_3 = \{s_4\}$.

By Definition 6 and given that $s_2 \in$ BSCC K_1, we first define $\tau(s_2, \vec{x}) = Pr^{\mathcal{D}^{lc}[\vec{x}]}(\Diamond K_1) \cdot \sigma(s_2, \vec{x}) + \rho_i(\vec{x}) \cdot \boldsymbol{\pi}_{s_2}^{\mathcal{D}(K_1)[\vec{x}]}$ as the variation function, where $\boldsymbol{\pi}_{s_2}^{\mathcal{D}(K_1)[\vec{x}]}$ denotes the long-run probability that an app will be in *deployment* (s_2) in the sub-PDTMC $\mathcal{D}(K_1)[\vec{x}]$ induced by BSCC K_1.

Following Theorem 4.2, the linear fragment of $\tau(s_2, \vec{x})$ is defined as $\tau_1(s_2, \vec{x}) = Pr^{\mathcal{D}^{lc}}(\Diamond K_1) \cdot \sigma_1(s_2, \vec{x}) + \rho_1(\vec{x}) \cdot \boldsymbol{\pi}_{s_2}^{\mathcal{D}(K_1)}$ where $Pr^{\mathcal{D}^{lc}}(\Diamond K_1) = 0.166$, $\sigma_1(s_2, \vec{x}) = 1.166x_4 + 0.166x_5$ (irreducible case), $\rho_1(\vec{x}) = 0.111x_1 + 0.666x_2$ and $\boldsymbol{\pi}_{s_2}^{\mathcal{D}(K_1)} = 0.199$. Thus, we have $\tau_1(s_2, \vec{x}) = 0.0221x_1 + 0.133x_2 + 0.0648x_4 + 0.0092x_5$. Consequently, based on Definition 9, the condition number $\kappa_{s_2}^{red} = 0.067$ (shown in Table 1). And, since we are interested in the worst-case effect of $\vec{x}$ when the total perturbation distance $\delta = 0.001$, we compute the upper bound $f_{red}^{+} = 0.033 + (0.067)(0.001) = 0.0331$, and the lower bound $f_{red}^{-} = 0.033 - (0.067)(0.001) = 0.0329$.

5 EXPERIMENTAL EVALUATION

In this section, we evaluate the applicability of our approach in two case studies. Our purpose is to demonstrate that the computed asymptotic bounds provide an accurate estimation of the worst-case consequences of uncertainty on verification of long-run behavior in systems modeled as DTMCs. For this purpose, we have developed a prototype implementation in Python.

Figure 2 describes a general overview of our prototype. As can be seen from the figure, for each case study, we first build the system model as a DTMC $\mathcal{D}$. Second, based on the designer expertise, we identify transitions vulnerable to perturbations. A perturbation variable is attached to each probabilistic transition. As a result, we obtain the perturbation vector $\vec{x}$. Third, we formulate the long-run probability of being in a particular state s to be analyzed, denoted as $\boldsymbol{\pi}_s$. These aforementioned components are the input of our prototype. By following our perturbation approach described in Section 4, our prototype computes a condition number κ_s for s that captures the effect of uncertainty in the verification of $\boldsymbol{\pi}_s$. Lastly, our prototype calculates linear perturbation bounds based on the computed condition number and a given perturbation distance $\delta > 0$. All our experiments have been conducted on a machine with 3.06 GHz Intel Core Duo processors and 8 GB RAM.

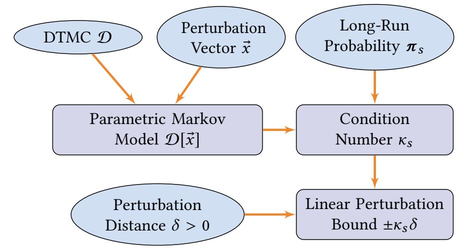

Figure 2: Computation Procedure

5.1 Clickstream Data

Model. As a first case study, we model the stochastic process of users clicking through the web-pages on a server as an *irreducible* DTMC [2, 17]. Specifically, we use public clickstream data collected for a popular news site *msnbc.com* [10]. The data spans the entire day of September 28, 1999 and it contains 989818 logs. Each log represents a clickstream. A clickstream is a sequence of web page categories visited by a given user. The categories are listed in Table 2. The DTMC $\mathcal{D}^{cl} = (S, \mathbf{P}, S_{\text{init}}, AP, L)$ consists of 17 states $S = \{s_1, \ldots, s_{17}\}$ which represent the categories of the news site. The set of initial states S_{init} is composed by the front-page (s_1). The transitions in $\mathcal{D}^{cl}$ represent the sequences of clicks from one category page to another. $\mathbf{P}(s_i, s_j)$ is calculated by counting all instances of state s_i that precede state s_j across all clickstreams. The model contains 17 states and 289 transitions.

We investigate the impact of estimating the DTMC's transition probabilities from empirical data using maximum-likelihood estimation. The clickstream sample is subject to some variance resulting in small perturbations to the estimated transition probabilities. We are interested in the probability that a user is visiting a certain webpage in the long run. Thus, we calculate the long-run probabilities $\boldsymbol{\pi}_{s_i}^{\mathcal{D}^{cl}}$ of all states s_i with $1 \leq i \leq 17$. In the following, we simply refer to $\boldsymbol{\pi}_{s_i}^{\mathcal{D}^{cl}}$ as $\boldsymbol{\pi}_{s_i}$.

Uncertainty. The DTMC $\mathcal{D}^{cl}$ that we created from the observed clickstream data is subject to sampling error on each state transition; the sampling error will compound and threaten the validity of any verification results. The clickstream data was collected on a single day. Thus, the data is merely a *sample* of the population of visitors to *msnbc.com*. Sample statistics represent population statistics only to some degree of accuracy. The DTMC's transition probabilities thus estimated are hence subject to perturbations.

To account for the sampling error, we perturbed each probabilistic transition in the Markov model $\mathcal{D}^{cl}$. In total, we added 289 perturbation variables, $\vec{x} = \langle x_1, \ldots, x_{289}\rangle$, one for each transition. After perturbing the model, we are interested in analyzing which state is affected most by the perturbation, i.e. which state is most susceptible to compounding sampling errors. For this reason, for each state s_i, we compute its corresponding condition number κ_{s_i} by using our approach. Note that since the DTMC $\mathcal{D}^{cl}$ is irreducible, we follow Definition 7 for computing κ_{s_i}.

We present experimental results for this case study in Table 2. The first two columns depict the set of 17 web page categories, which are ordered from the most visited to least visited. The next column shows the probability $\boldsymbol{\pi}_{s_i}$ that a visitor, in the long-run, visits page s_i, given the ideal scenario ($\delta = 0$). The last column depicts the corresponding condition number κ_{s_i}.

As can be seen from Table 2, in an unperturbed model, the most visited page is the front-page (15.79%), followed by BBS (10.99%)

Table 2: Experimental Results for Clickstream Data

s_i	Name	$\boldsymbol{\pi}_{s_i}$ ($\delta = 0$)	κ_{s_i}	s_i	Name	$\boldsymbol{\pi}_{s_i}$ ($\delta = 0$)	κ_{s_i}
s_1	Front-page	0.1579	4.0537	s_3	Tech	0.0401	3.4375
s_{14}	BBS	0.1099	6.9263	s_{13}	Summary	0.0377	5.5655
s_2	News	0.1019	4.3049	s_{10}	Living	0.0343	3.8085
s_8	Weather	0.0979	12.146	s_9	Health	0.0315	3.3542
s_4	Local	0.0966	5.5212	s_{11}	Business	0.0217	2.5251
s_7	Misc	0.0783	4.3953	s_{15}	Travel	0.0135	2.0452
s_6	On-air	0.0608	3.3277	s_{16}	Msn-news	0.0084	13.899
s_{12}	Sports	0.0549	4.2728	s_{17}	Msn-sports	0.0042	2.0428
s_5	Opinion	0.0504	8.7741				

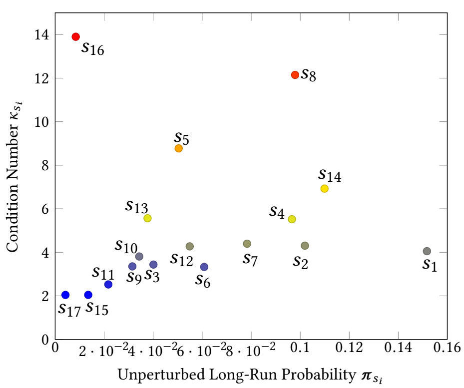

Figure 3: Relationship Between the Long-Run Probability and Condition Number of a Web-Page

and news (10.19%). On the contrary, the least visited pages are msn-news (0.42%) and msn-sport (0.84%). In Figure 3 we compare the long-run probabilities in the unperturbed model (x-axis) with the condition numbers for each state s_i. The condition numbers indicate the effect of perturbing vector $\vec{x}$ during the computation of $\boldsymbol{\pi}_{s_i}$, i.e. the impact of the unknown sampling error on the calculated long-run probabilities.

Observations. Figure 3 allows us to make several insightful observations. First, it reveals that the msn-news page (denoted as s_{16})—even though it is one of the least visited pages with $\boldsymbol{\pi}_{s_{16}} = 0.0084$—is very sensitive with respect to perturbations of the transitions probabilities. We can draw this conclusion from the large condition number, $\kappa_{s_{16}} = 13.899$. This finding indicates that small changes in the transition probabilities may have a severe impact on the number of visits to the msn-news page. Second, the sensitivity of the front-page, which is the most visited page, has a condition number ($\kappa_{s_1} = 4.0537$) that is close to the median of 4.2728. Thus, the front-page is not particularly susceptible to perturbation of transition probabilities. Finally, inspecting Figure 3 we cannot find a clear correlation between the long-run probability and the condition number. This implies that the effect of uncertainty is undiscriminating for the most and least visited web-pages.

Soundness. To evaluate whether our perturbation techniques provides a sound estimate of the worst-case consequences of uncertainty in the computation of long-run probabilities, we introduce

Table 3: Accuracy of the Linear Perturbation Bounds

s_i	Name	κ_{s_i}	$\boldsymbol{\pi}_{s_i}$ $\delta = 0$	$\boldsymbol{\pi}_{s_i}$ δ^*	$\Delta\boldsymbol{\pi}_{s_i}$	$\pm\kappa_{s_i}\delta^*$
s_1	Front-page	4.0537	0.1579	0.1368	-0.0211	0.5545
s_{16}	Msn-news	13.899	0.0084	0.0045	-0.0039	0.0625
s_8	Weather	12.146	0.0979	0.0587	-0.0392	0.7130
s_5	Opinion	8.774	0.0504	0.0337	-0.0167	0.2957

Table 3. As the table shows, we have restricted our evaluation to the most visited page (front-page) in the unperturbed scenario and three of the most sensitivity pages. To provide some intuition about the worst perturbation that might affect the model, we compute $\delta = 21.359$ as the 90% bootstrap confidence interval with around 10% clickstream data as re-sample size. In other words, 0.0726 perturbation for each transition approximately. Given that the greatest probabilities are located in $\mathbf{P}(s_i, s_i)$, these probabilities have been perturbed -0.0726. Then, $\sum \mathbf{P}(s_i, s_j) = 0.0726, i \neq j$ as a consequence of the sum of the probabilities of transitions has to normalize to 1. In short, the total distance of the perturbed model is $\delta^* = 0.1452$.

As can be seen from Table 3, for each page, we calculate the long-run probability against a non-parametric DTMC with distance δ^*. Then, we compare the perturbed and unperturbed long-run probability, denoted as $\Delta\boldsymbol{\pi}_{s_i}$, against our estimate $\pm\kappa_{s_i}\delta$ of the worst-case effect of uncertainty in the computation of $\boldsymbol{\pi}_{s_i}$. Our experimental results show that, for each table, the estimate $\pm\kappa_{s_i}\delta$ provide a safe prediction of the impact of perturbations. For example, the difference between the actual probabilities for the perturbed and unperturbed models for the front-page (s_1) is -0.0211, which lies within our estimation of ± 0.5545.

In summary, using perturbation analysis we succeed in detecting a web page category (msn-news) that is very sensitive to small changes to the transition probabilities. Even minor changes to the transition probabilities might significantly impact the frequency of visits to the msn-news pages. Perturbation analysis of discrete-time Markov chain reflects uncertainties in the model. These uncertainties include (but are not limited to) uncertainties due to sampling error and changing user behavior. The insights generated by our perturbation analysis will be very helpful for capacity planning and related activities.

5.2 Call Invocation Sequences

In the previous section we analyzed clickstream data to evaluate how the proposed perturbation technique can help to predict web server traffic and cope with uncertainties in the analyzed model. In this section we continue by investigating how the software engineering community can benefit from our approach, in particular how the technique can help during software maintenance.

Model. To this end we model the invocation sequence graph for `wc`, a program in the `coreutils 8.25` tool set. `wc` parses the standard input or alternatively a set of files and counts various attributes of the read data. By default it counts number of lines, words, and bytes. We focus on `wc` for a multitude of reasons. First, `coreutils` is widely distributed and used daily across a diverse user population. Second, `wc` has a long development history dating back to at least 1985 making it a very mature software artifact. Third, since coreutils is very mature, we expect a mature test suite which we utilize to collect transition probabilities to model a DTMC and exemplify the strength of the proposed perturbation technique. Finally, we picked the wc program from the coreutils tool set due to prior exposure to its source code. Thus, we were able to manually verify our results using existing prior knowledge.

Similar to the previous case study, we model wc as an *irreducible* DTMC $\mathcal{D}^{\text{wc}} = (S, \mathbf{P}, S_{init}, AP, L)$. The set of states S contains all unique functions called during runtime of wc. Each called function f_i is represented as a state $s_i \in S$. Additionally, S contains two pseudo states, "start" and "end", that represent program start and program termination, respectively. The set of initial states S_{init} consists of the "start" state. The transitions in $\mathcal{D}^{\text{wc}}$ represent the sequences of calls observed during runtime. We do not model function returns. In particular, $\mathbf{P}(s_i, s_j)$ is calculated by counting the fraction of calls to the function f_i represented by s_i that is directly followed by a call to the function f_j represented by state s_j. A transition f_i to f_j indicates that function f_j might be called either by f_i directly *or by any caller of* f_i. Thus, we model function call sequences not call graphs.

To gather runtime data we instrument wc with LLVM XRay 5.0.1 [1]. In particular, we use LLVM XRay's capability to collect all function enter and exit events that occurs during runtime. Next, we execute the test suite of coreutils against the instrumented wc binary. We obtain 42 trace files that contain 55, 072 logged function calls. We use these 55, 072 function calls to create $\mathcal{D}^{\text{wc}}$ as described above. The resulting model contains 46 states representing functions plus 2 pseudo states and 72 transitions.

Uncertainty. A common challenge in the context of software maintenance is to judge about the impact of a prospective change. The proposed technique can be used to analyze the sensitivity of a function with respect to call frequency if transition probabilities are perturbed. This can be helpful in the context of software maintenance to, for example, judge about the risk involved in a code change.

For example, let us assume a change is pending. In order to judge about the risk involved in applying the change to the software, especially with regard to performance, we would like to know how often the changed function f_i is expected to be executed. That is, we are interested in computing the long-run probability $\boldsymbol{\pi}_{s_i}$. If the function f_i has a high probability of being executed, then any change to this function might have a severe impact on availability and performance of the software artifact. In contrast, if the execution probability is low, then the risk involved is potentially lower. We use the previously described model capturing call invocation sequences to analyze the execution frequencies for wc. To account for uncertainties, we perturb all 40 probabilistic transition, that is $0 < \mathbf{P}(s_i, s_j) < 1$, of the model where $\vec{x} = \langle x_1, \ldots, x_{40} \rangle$. The experimental results for this case study are shown in Table 4.

The rightmost column in Table 4 contains the condition number (CN), denoted as κ_{s_i}. Due to space constraints, we present only a limited sub-set of functions of wc. In particular, we show all functions for which the condition number $\kappa_{s_i} > 10$. The functions are ordered by decreasing condition number. We also show the long-run probability $\boldsymbol{\pi}_{s_i}$ under no perturbation ($\delta = 0$).

Table 4: Experimental Results of wc for all functions with $\kappa_{s_i} > 10$

	Function	$\boldsymbol{\pi}_{s_i}^{\mathcal{D}^{wc}} (\delta = 0)$	κ_{s_i}
s_2	umaxtostr	0.32782	42.7632
s_{11}	to_uchar	0.00136	14.5389
s_3	argv_iter	0.10969	10.9580
s_8	wc_file	0.10931	10.7852
s_7	wc	0.10930	10.6524
s_6	fdadvise	0.10930	10.5196
s_4	safe_read	0.10964	10.4175
s_5	write_counts	0.10946	10.2672

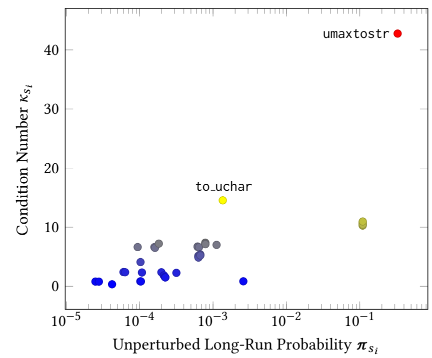

Figure 4: Relation between Functions and CNs for wc

Observations. The first entry in Table 4 describes umaxtostr. Of all the functions observed during runtime umaxtostr is most sensitive to perturbations ($\kappa_{s_2} = 42.7632$). It is also the most often executed function (with a long-run execution probability of 32.789%). Thus, any changes to umaxtostr should be associated with a higher risk. The next entry describes to_uchar. Looking at the execution probability of to_uchar (0.136%) we would predict that changes to this function are likely unproblematic as it is executed very seldom. However, the condition number for to_uchar is relatively high ($\kappa_{s_{11}} = 14.5389$). This indicates that small changes to the transition probabilities can have a severe impact on the execution frequency of to_uchar.

We investigate the call sites of to_uchar. All recorded calls to to_uchar originate from wc.c line 496 and 499. These lines are executed only if wc was instructed to count characters or words, or extract the maximum line length. Thus, how often to_uchar is executed depends strongly on the requested operation. Even though to_uchar is executed only seldom in our test environment any changes to it should be marked as risky since the selection of runtime options might severely increase its invocation rate. The high condition number for to_uchar correctly identifies this sensitivity.

Figure 4 shows the relationship between unperturbed long-run probability (log-scale) and condition number for wc. Similar to Figure 3 in the previous clickstream data scenario we cannot find a clear correlation between the long-run probability and the condition number.

Soundness. To evaluate the soundness of our perturbation estimate generated by our technique, we conducted additional experiments, depicted in Table 5. We first built a non-parametric

Table 5: Additional Experimental Results for wc

Functions		π_{s_i}	κ_{s_i}	$\delta = 1\times10^{-3}$			$\delta = 3\times10^{-3}$			$\delta = 5\times10^{-3}$		
s_i	Name	($\delta = 0$, %)		$\pi^*_{s_i}$(%)	$\Delta\pi_{s_i}$(%)	$\pm\kappa_{s_i}\delta$(%)	$\pi^*_{s_i}$(%)	$\Delta\pi_{s_i}$(%)	$\pm\kappa_{s_i}\delta$(%)	$\pi^*_{s_i}$(%)	$\Delta\pi_{s_i}$(%)	$\pm\kappa_{s_i}\delta$(%)
s_2	umaxtostr	32.79	42.7632	32.11	-0.680	±4.276	31.63	-1.160	±12.83	31.32	-1.470	±21.38
s_{11}	to_uchar	0.136	14.5389	0.138	0.002	±1.454	0.090	-0.046	±4.362	0.058	-0.078	±7.269
s_3	argv_iter	10.97	10.9580	10.82	-0.150	±1.096	10.71	-0.260	±3.287	10.63	-0.340	±5.479
s_8	wc_file	10.93	10.7852	10.77	-0.160	±1.078	10.66	-0.270	±3.236	10.59	-0.340	±5.393

model with small perturbation distances ($\delta = 1, 3, 5 \times 10^{-3}$). Second, we compute the long-run probability π_{s_i} for a sub-set of functions of `wc` in the perturbed model, denoted as $\pi^*_{s_i}$. In particular, we perturbed all probabilistic transitions in the strongly connected subgraph that performs the actual work of counting the requested properties from the input files. That is, we perturb all outgoing transitions from `umaxtostr`, `wc`, `safe_read`, `to_uchar`, `argv_iter`, `arg_iter_n_args`, and `call_freefun`. Then, we compare the perturbed and unperturbed long-run probabilities, computed as $\Delta\pi_{s_i} = \pi^*_{s_i} - \pi_{s_i}$. Lastly, we calculate the estimate $\pm\kappa_{s_i}\delta$, which provides the worst-case effect of uncertainty in the computation of π_{s_i}.

For each function the estimate $\pm\kappa_{s_i}\delta$ in Table 5 represents a safe prediction of the worst-case consequences caused by the perturbation. For example, given that $\delta = 1 \times 10^{-3}$, the difference between the actual long-run probabilities of function `to_uchar` for the perturbed and unperturbed models is $0.138 - 0.136 = 0.002$ with lies within the estimation of ± 1.454. The accuracy of this prediction is replicated where $\delta = 3, 5 \times 10^{-3}$. We obtained similar results for all functions presented in Table 5. Finally, note that `to_uchar` is affected substantially by the perturbations from a relative perspective. In contrast, the long-run probabilities of the other functions are quite stable across the different perturbation distances.

6 RELATED WORK

There is a rich body of literature that investigates finite irreducible Markov chains under perturbation. Many of those works focus on the effect of the perturbed transition matrix of a Markov chain on its steady-state distribution. Similar analysis has also been applied to queueing models [6]. One common technique is the derivation of norm-based bounds for irreducible Markov chains. Specifically, they computed a condition number κ for measuring the sensitivity of the Markov chain. Let $\mathbf{E} = (E^j_i) \in \mathbb{R}^{n\times n}$ denote the perturbation. Let $\mathbf{P}$ and $\mathbf{P'} = \mathbf{P}+\mathbf{E}$ be probability transition matrices of irreducible Markov chains with respective stationary probability vectors π and π' satisfying $\pi\mathbf{P} = \pi$, $\pi'\mathbf{P'} = \pi'$, and $\sum_i \pi_i = 1 = \sum_i \pi'_i$. Then, $\|\pi - \pi'\| \leq \kappa \cdot \|\mathbf{E}\|$ for suitable vector and matrix norms.

In those approach, the condition number κ has been calculated by using the fundamental matrix of a Markov chain as introduced by Kemeny *et al.* [11] and Schweitzer [12], the group inverse of $A \equiv \mathbf{I} - \mathbf{P}$ as presented in [9, 13], the ergodicity coefficients [7] and the mean first passage times [3]. All these approaches for computing the condition number were collected and compared by Cho and Meyer [4]. Based on their review, the smallest condition number is computed by using relevant information of the group inverse of A. This approach was proposed by Haviv *et al.* [9] and Kirkland *et al.* [13].

However, Cho and Meyer also revealed that the computation of the fundamental matrix and the group inverse are usually expensive. For this reason, the condition number in terms of the mean first passage is more feasible to be computed. Likewise, it provides an equivalent condition number to Haviv's definition based on the underlying structure of the Markov chain.

Together these studies provide valuable insights into the investigation of the sensitivity of the stationary distribution of Markov chains under perturbation. However, these previous studies compute the condition number κ based on the perturbation $\mathbf{E}$, which is a matrix of prior defined values. Even though we are interested in analyzing the effect of the perturbation in the long-run behavior of MCs, our proposal is different from these previous studies in two points. First, we ignore the defined values that the perturbation might take. Thus, we propose to capture the perturbation as symbolic variables. And, to provide an estimation of the worst-case consequences of the uncertain phenomena. Second, previous studies have not been applied in probabilistic model checking. To the best of our knowledge, no previous publication in the literature has addressed this problem using perturbation analysis.

7 CONCLUSIONS

The main goal of this paper is to estimate the worst-case consequences of uncertainty on verification of long-run properties in DTMCs. Our main contribution is a mathematical framework for asymptotic analysis in the presence of small perturbations to model probabilities on verification of long-run properties against reducible and irreducible Markov chains.

To evaluate our perturbation approach, we implemented a prototype in Python. We ran our experiments on clickstream data and call invocation sequences. These case studies have been modeled as an MC and affected by small perturbations for our evaluation purposes. For each case study, we have evaluated several long-run properties under the presence of perturbations. Our experimental results reveal the importance of our approach at estimating the effect to these perturbations and they provide crucial information for identifying the sensitivity of the long-run properties under uncertainty. This work is the first study on dealing the uncertain phenomena in the probabilistic verification of long-run properties in DTMCs by using perturbation analysis.

There are several directions for further study. For instance, backward analysis that provides the maximum permitted perturbations based on a range of verification results.

REFERENCES

[1] Dean Michael Berris, Alistair Veitch, Nevin Heintze, Eric Anderson, and Ning Wang. 2016. *XRay: A Function Call Tracing System*. Technical Report. A white paper on XRay, a function call tracing system developed at Google.

[2] Igor V Cadez, David Heckerman, Christopher Meek, Padhraic Smyth, and Steven White. 2000. Visualization of navigation patterns on a Web site using model-based clustering. In *KDD*. 280.

[3] Grace E. Cho and Carl D. Meyer. 2000. Markov Chain Sensitivity Measured by Mean First Passage Times. *Linear Algebra Appl.* 316, 1 (2000), 21–28. https://doi.org/10.1016/S0024-3795(99)00263-3

[4] Grace E. Cho and Carl D. Meyer. 2001. Comparison of Perturbation Bounds for the Stationary Distribution of a Markov Chain. *Linear Algebra Appl.* 335, 1 (2001), 137–150.

[5] Nicolas Coste. 2010. *Towards Performance Prediction of Compositional Models in GALS Designs*. Ph.D. Dissertation. University of Grenoble.

[6] Altman E., Avrachenkov K., and Núñez Queija R. 2004. Perturbation Analysis for Denumerable Markov Chains with Application to Queueing Models. *Advances in Applied Probability* 36 (2004), 839–853.

[7] Seneta E. 1988. Perturbation of the Stationary Distribution Measured by Ergodicity Coefficients. *Advances in Applied Probability* 20, 1 (1988), 228–230.

[8] Ernst Moritz Hahn, Holger Hermanns, and Lijun Zhang. 2009. Probabilistic Reachability for Parametric Markov Models. In *SPIN*. 88–106.

[9] Moshe Haviv and Ludo Van Der Heyden. 1984. Perturbation Bounds for the Stationary Probabilities of a Finite Markov Chain. *Advances in Applied Probability* 16, 4 (1984), 804–818. http://www.jstor.org/stable/1427341

[10] David Heckerman. [n. d.]. MSNBC.com Anonymous Web Data Set. http://archive.ics.uci.edu/ml/datasets/msnbc.com+anonymous+web+data. ([n. d.]). Accessed: 2018-02-05.

[11] Kemeny J. and Snell J. 1960. *Finite Markov Chains*. D. Van Nostrand, New York.

[12] Schweitzer Paul J. 1968. Perturbation Theory and Finite Markov Chains. *Journal of Applied Probability* 5, 2 (1968), 401–413.

[13] Stephen Kirkland, Michael Neumann, and Bryan Shader. 1998. Applications of Paz's Inequality to Perturbation Bounds for Markov Chains. *Linear Algebra Appl.* 268, 1 (1998), 183–196.

[14] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. 2005. Probabilistic Model Checking in Practice: Case Studies with PRISM. *SIGMETRICS Performance Evaluation Review* 32, 4, 16–21.

[15] Ruggero Lanotte, Andrea Maggiolo-Schettini, and Angelo Troina. 2007. Parametric Probabilistic Transition Systems for System Design and Analysis. *Formal Asp. Comput.* 19, 1 (2007), 93–109. https://doi.org/10.1007/s00165-006-0015-2

[16] Yamilet R. Serrano Llerena, Guoxin Su, and David S. Rosenblum. 2017. Probabilistic Model Checking of Perturbed MDPs with Applications to Cloud Computing. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2017, Paderborn, Germany, September 4-8, 2017*. 454–464. https://doi.org/10.1145/3106237.3106301

[17] Alan L. Montgomery, Shibo Li, Kannan Srinivasan, and John C. Liechty. 2004. Modeling Online Browsing and Path Analysis Using Clickstream Data. *Marketing Science* 23, 4, 579–595.

[18] Nicolas Privault. 2013. *Understanding Markov Chains: Examples and Applications*. Springer Singapore.

[19] Guoxin Su, Taolue Chen, Yuan Feng, and David S. Rosenblum. 2017. ProEva: Runtime Proactive Performance Evaluation Based on Continuous-Time Markov Chains. In *Proceedings of the 39th International Conference on Software Engineering, ICSE 2017, Buenos Aires, Argentina, May 20-28, 2017*. 484–495. http://dl.acm.org/citation.cfm?id=3097426

[20] G. Su, Y. Feng, T. Chen, and D. S. Rosenblum. 2016. Asymptotic Perturbation Bounds for Probabilistic Model Checking with Empirically Determined Probability Parameters. *IEEE Transactions on Software Engineering* 42, 7 (July 2016), 623–639. https://doi.org/10.1109/TSE.2015.2508444

[21] Guoxin Su and David S. Rosenblum. 2013. Asymptotic Bounds for Quantitative Verification of Perturbed Probabilistic Systems. In *ICFEM*. 297–312.

[22] Guoxin Su and David S. Rosenblum. 2014. Perturbation Analysis of Stochastic Systems with Empirical Distribution Parameters. In *ICSE*. 311–321.