A Unified Framework for Quantitative Analysis of Probabilistic Programs

Shenghua Feng[1*], Tengshun Yang[2*], Mingshuai Chen[3(✉)], and Naijun Zhan[4,2(✉)]

[1] Zhongguancun Laboratory, Beijing, China
[2] Institute of Software, CAS, University of Chinese Academy of Sciences, China
[3] Zhejiang University, Hangzhou, China
[4] School of Computer Science, Peking University, China

Abstract. Verifying probabilistic programs requires reasoning about various probabilistic behaviors, e.g., random sampling, nondeterminism, and conditioning, against multiple quantitative properties, e.g., assertion-violation probabilities, moments, and expected running times. It is desirable and theoretically significant to have a unified framework which can deal with quantitative analysis of programs with different probabilistic behaviors and properties. In this paper, we present a unified framework for the quantitative analysis of probabilistic programs, which incorporates and extends existing results on the analysis of termination, temporal properties, and expected cost. We show that these quantitative properties of a general probabilistic program can be characterized as solutions to equation systems of the corresponding Markov chain counterpart with a possibly uncountable state space. Based on such characterization, we propose sufficient conditions to establish upper and lower bounds on these quantitative properties. Moreover, we demonstrate how our approach can be adapted to address inference problems in Bayesian programming.

Keywords: Probabilistic programs · Quantitative analysis · Markov chains · Equation systems

1 Introduction

Probabilistic programming [39,29,55] is a novel programming paradigm that extends classical programming languages with statements such as probabilistic branching and sampling. Probabilistic programs provide a powerful model for randomized algorithms [9], artificial intelligence [13,52], reliability engineering [15], network protocols [27,41], etc. These applications involve safety-critical domains such as autonomous vehicles and spacecraft. Thus, formal analysis of these probabilistic (control) programs has become increasingly significant – as is mandatory per many standards – and has undergone a recent surge of interest.

[*] Both authors contributed equally to this work.

Compared to the formal analysis of classical programs that mostly focuses on *qualitative* properties – which are represented as the intersection of safety and liveness [4] – the analysis of probabilistic programs concerns *quantitative* properties in many situations. Prominent problems of quantitative analysis of probabilistic programs include (1) *expected running time analysis* [34,35,28,1,30], for reasoning about the expected termination time; (2) *temporal property analysis* [57,22,54], for, e.g., estimating the probability that an assertion eventually happens ($\Diamond$ operator) or the probability that one assertion always holds until another assertion holds ($\mathbf{U}$ operator); (3) *cost analysis* [58,56,48], for analyzing the expected accumulated resource consumption incurred by program transitions and termination;[5] and (4) *expectation analysis* [32,30,11], for deriving the bounds for weakest pre-expectation.

In the literature, there are various approaches proposed to attack the aforementioned properties for programs with different probabilistic behaviors. Among these approaches, there are two lines of work leveraging martingale-based reasoning and weakest pre-expectation (wp) reasoning, respectively:

- *Martingale-based reasoning.* For expected running time analysis, [16] employs martingales to find upper bounds of expected running time for programs without nondeterminism. [20] further extends the martingale-based technique to handle programs with nondeterminism, and obtains sound conditions for finding upper bounds of expected running time. [30] proposes conditions to verify lower bounds on expected running time for programs without nondeterminism. For temporal property analysis, [57] focuses on finding upper and lower bounds of assertion violations for programs without nondeterminism. For cost analysis, [48] considers the transition cost to be non-negative and bounded. [58] further considers general transition costs for programs with at most one type of nondeterminism and provides sufficient conditions for finding upper and lower bounds. Recently, [56] presents conditions to derive upper and lower bounds on higher moments of expected accumulated cost.
- *Weakest pre-expectation reasoning.* Weakest pre-expectation reasoning [44] is a reasoning technique for probabilistic programs, which extends the classical weakest precondition calculus for classical imperative programs. [35] presents a wp-style calculus for obtaining bounds on expected running time. In [33], a weakest pre-expectation style calculus is proposed for reasoning about expected values of mixed-sign random variables after termination[6] for programs without nondeterminism. Recently, [26] introduces a guard-strengthening rule to infer lower bounds of wp for possibly diverging probabilistic programs.

To summarize, extensive research has been conducted on analyzing probabilistic programs, targeting diverse objectives such as termination, expected running time, temporal properties, and expected cost. Common to many of these

[5] Expected running time analysis can be seen as an instance of expected cost.
[6] This can be interpreted as termination cost in cost analysis.

methods is a typical procedure: They reduce the verification process (specifically, the determination of upper and lower bounds on desired quantities) to synthesizing functions that meet specific conditions. However, the methodologies for expressing different types of analyses can vary significantly. Additionally, each analytical approach imposes different side conditions on the probabilistic programs under examination. These include constraints on the non-negativity of expectation functions [44], boundedness of transition cost functions [48], and limitations on the boundedness of variable updates [58], to name just a few. It is both desirable and of theoretical importance to develop a unified framework capable of addressing all aforementioned issues in a consistent manner, thereby allowing for the seamless integration of various analytical results.

Synopsis of Our Approach and Contributions. We propose a unified framework that facilitates reasoning about various quantitative properties of probabilistic programs. This approach enables the integration and extension of existing analyses of probabilistic programs, including, but not limited to, expected running time, cost, and temporal property analyses. We demonstrate that these quantitative properties of a general probabilistic program can be represented as solutions to equation systems associated with its Markov chain counterpart with a possibly uncountable state space. Based on this characterization, we propose sufficient conditions for establishing upper and lower bounds on these quantitative properties and discuss how our approach can be extended to Markov decision processes. Furthermore, we illustrate how our framework can be adapted to tackle inference problems in Bayesian programming.

Paper Organization. In Sect. 2, we first introduce basic notations on Markov chains and probabilistic programs and then formulate the problem of interest. Sect. 3 outlines sufficient conditions for estimating the quantitative properties of Markov chains, based on which, Sect. 4 illustrates the method for consistently encapsulating earlier findings in the quantitative analysis of probabilistic programs. In Sect. 5, we demonstrate how our approach can be adapted to address inference problems in Bayesian programming. Sect. 6 concludes the paper.

Related Work. In addition to the above-mentioned techniques using martingale- and weakest pre-expectation-based reasoning, quantitative analysis of probabilistic programs can also be handled by other methods, e.g., value iteration-based algorithms for approximating reachability probabilities of Markov systems [8,31,50], characteristic function-based forward inference techniques [23,37], distribution approximation (in terms of, e.g., moment series) using density estimation in statistics [36,38], coupling-based techniques for differential privacy, convergence, and program equivalence [2,3], etc. Among them, moment-based methods exhibit high efficiency and have been used to address, e.g., Bayesian network properties including exact inference [53] and termination analysis [45,46]. Moreover, [47] and [5] enlarge the scope of moments derivation to enable wider applications of moment-based methods. [40,56] bound higher central moments for running times and other monotonically increasing quantities.

2 Preliminaries

2.1 Probability Theory

Let $\mathbb{N}$, $\mathbb{R}$, $\mathbb{R}^+$ be respectively the set of natural, real, and non-negative real numbers. Let $(f)^+$ and $(f)^-$ denote the positive and negative part of f, i.e. $(f)^+ = \max\{f, 0\}$, $(f)^- = \max\{-f, 0\}$, and thus $f = (f)^+ - (f)^-$, $a \wedge b$ denote the minimum between a and b. For any set A, $\mathbf{1}_A$ denotes the indicator function that maps s to 1 if $s \in A$ and 0 otherwise.

A *probability space* is a triple $(\Omega, \mathcal{F}, \mathbb{P})$, where Ω is a sample space, $\mathcal{F} \subseteq 2^\Omega$ is a σ-algebra on Ω, and $\mathbb{P}\colon \mathcal{F} \to [0, 1]$ is a probability measure on the measurable space $(\Omega, \mathcal{F})$. For any measurable space $(\Omega, \mathcal{F})$, denote the set of probability measure on Ω by $\mathcal{D}(\Omega)$. The support of the probability measure $\mathbb{P}$ on $(\Omega, \mathcal{F})$ is

$$\mathtt{supp}(\mathbb{P}) = \overline{\{\, A \in \mathcal{F} \mid \mathbb{P}(A) \neq 0 \,\}},$$

where $\overline{H}$ denote the closure of H for any set H. If Ω is a finite set and $\mathcal{F} = 2^\Omega$, then $\mathtt{supp}(\mathbb{P})$ is the set of all elements in Ω that have positive probability. A *random variable* X defined on the probability space $(\Omega, \mathcal{F}, P)$ is a $\mathcal{F}$-measurable function $X\colon \Omega \to \mathbb{R} \cup \{-\infty, +\infty\}$; its *expectation* (w.r.t. $\mathbb{P}$) is denoted by $E[X]$; For any set $A \subseteq \Omega$, $E[X \cdot \mathbf{1}_A]$ is also denoted by $E[X; A]$. Let $\mathcal{F}' \subseteq \mathcal{F}$ is a sub-σ-algebra, a *conditional expectation* of X w.r.t. $\mathcal{F}'$ is a $\mathcal{F}'$-measurable random variable denoted by $E[X \mid \mathcal{F}']$, such that $E[X \cdot \mathbf{1}_A] = E[E[X \mid \mathcal{F}'] \cdot \mathbf{1}_A]$ for all $A \in \mathcal{F}'$. A collection $\{\mathcal{F}_n \mid n \in \mathbb{N}\}$ of σ-algebras in $\mathcal{F}$ is a *filtration* if $\mathcal{F}_n \subseteq \mathcal{F}_{n+k}$ for $n, k \in \mathbb{N}$. A random variable $T\colon \Omega \to [0, \infty]$ is called a *stopping time* w.r.t. some filtration $\{\mathcal{F}_n \mid n \in \mathbb{N}_0\}$ of $\mathcal{F}$ if $\{T \leq n\} \in \mathcal{F}_n$ for all $n \in \mathbb{N}$.

A stochastic process $\{X_n\}_{n\in\mathbb{N}}$ adapted to a filtration $\{\mathcal{F}_n \mid n \in \mathbb{N}\}$ is called a *supermartingale* (resp. *submartingale*) if $E[X_n] < \infty$ for any $n \in \mathbb{N}_0$ and $E[X_m \mid \mathcal{F}_n] \leq X_n$ (resp. $E[X_m \mid \mathcal{F}_n] \geq X_n$) for all $m \leq n$. That is, the conditional expected value of any future observation, given all past observations, is no larger (resp. smaller) than the most recent observation. $\{X_n\}_{n\in\mathbb{N}}$ is a martingale if it is both supermartingale and submartingale. A set of random variables $\{X_n\}_{n\in\mathbb{N}}$ is *uniformly integrable*, if

$$\lim_{M\to\infty} \sup_n E[|X_n| \cdot \mathbf{1}_{|X_n|\geq M}] = 0.$$

A *Markov chain* (MC) is a tuple $\mathcal{M} = (S, \mathbf{P})$, where S is a set of states endowed with $\sigma-$algebra $\mathcal{F}_S$, and $P\colon S \times \mathcal{F}_S \to [0, 1]$ is the transition probability function such that for all state s, $\mathbf{P}(s, \cdot)$ is a probability measure over $\mathcal{F}_S$. For $s \in S$, the set of infinite paths of $\mathcal{M}$ starting from s is $\mathtt{Paths}^{\mathcal{M}}(s) = \{\, \pi = s_0 s_1 \ldots \in S^\omega \mid s_0 = s \,\}$, the set of all infinite paths of $\mathcal{M}$ is $\mathtt{Paths}^{\mathcal{M}} = \cup_{s\in S} \mathtt{Paths}^{\mathcal{M}}(s)$. Following a standard procedure [25], for any initial state $s \in S$, we can construct a probability measure over the set of infinite paths $\mathtt{Paths}^{\mathcal{M}}$, with $\sigma-$algebra generated by all cylinder sets. We denote this probability measure by $\mathbb{P}_s$, where s is the initial state. Let X_n be the random process that represents the system state after n^{th} transition, formally,

$$X_n\colon \mathtt{Paths}^{\mathcal{M}} \to S,$$
$$s_0 s_1 \ldots s_n \ldots \mapsto s_n.$$

```
1 :  while (x ≥ 0){
2 :    if (x ≥ 4)
3 :      { x = x − 1 [0.4] x = x + 1 }
4 :    else
5 :      { x = x − 1 [0.5] x = x + 1 }
6 :  }.
7 :
```

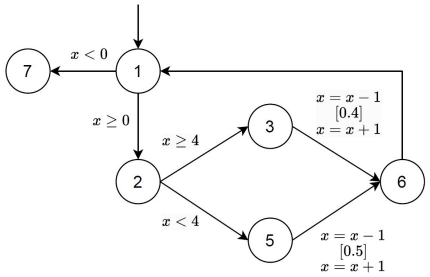

Fig. 1: A probabilistic program (left) and its CFG (right), where location label $l_{\mathsf{in}} = 1$ and $l_{\mathsf{end}} = 7$ represent the starting and ending point of the program, respectively.

The $\sigma-$algebra generated by $X_1, X_2, \ldots, X_n$ is denoted by $\mathcal{F}_n$, $E_s[\cdot]$ denotes taking expectation w.r.t. probability measure $\mathbb{P}_s$.

2.2 Probabilistic Programs

In this subsection, we describe the syntax of the probabilistic programs under investigation:

$$C ::= \mathsf{skip} \mid x := e \mid x :\approx \mu \mid C; C \mid \{C\}\ [p]\ \{C\} \mid \mathsf{if}\ (\varphi)\ \{C\}\ \mathsf{else}\ \{C\} \mid \mathsf{while}\ (\varphi)\ \{C\},$$

where x is a program variable taken from a countable set Vars of variables, φ is a formula over program variables that is a Boolean combination of arithmetic inequalities, and μ is a predefined probability distribution. The semantics of most statements, including skip, assignment, sequential composition, conditional, and the while statement, follow their standard meaning in imperative programs. The semantics of a statement $\{C_1\}[p]\{C_2\}$ is a probabilistic choice that flips a coin with bias $p \in [0, 1]$ and executes the statement C_1 if the coin yields head, and C_2 otherwise. Note that the probabilistic choice statement $\{C_1\}\ [p]\ \{C_2\}$ is the syntactic sugar for $x :\approx \mathsf{Bernoulli}(p); \mathsf{if}\ (x = 0)\ \{C_1\}\ \mathsf{else}\ \{C_2\}$, where $\mathsf{Bernoulli}(p)$ equals 0 with probability p and equals 1 with $1-p$. The semantics of a statement $x :\approx \mu$ samples a value according to the predefined distribution μ (including both discrete and continuous distributions) and assigns the value to the variable x.

Given a probabilistic program C, its semantics can be interpreted as a discrete-time Markov chain over its underlying control flow graph (e.g. Fig. 1). The state space is defined as the Cartesian product of the program location (or program counter) L and the variable valuation Val, where Val represents the set of all mappings from program variables to their respective values. There are two special locations in L: l_{in} represents the starting location of the program, and l_{end} represents the ending location of the program.

Definition 1 (Operational Semantics of Probabilistic Programs [18,16]). *Given a probabilistic program C, its operational semantics can be characterized*

by a Markov chain $\mathcal{M}_C = (S_C, \mathbf{P}_C)$*, where* $S_C \triangleq L \times \mathsf{Val}$*,* $\mathbf{P}_C$ *is the transition probability over* S_c*.*

Therefore, *analyzing a probabilistic program is essentially the same as analyzing its underlying Markov model.* In the following, we will investigate the quantitative properties of the Markov chain and demonstrate its relationship with the corresponding probabilistic program in Sect. 4.

2.3 Quantitative Analysis of MC

In this subsection, we formalize the notations for quantitative analysis of MC and formulate the problem investigated in this paper.

LTL-style notations for quantitative properties. Given an MC $\mathcal{M} = (S, \mathbf{P})$. For any $A, B \subseteq S$, let $\Diamond A$ denote the set of paths in $\texttt{Paths}^{\mathcal{M}}$ that can reach A eventually, and $A \,\mathbf{U}\, B$ stand for the set of paths staying in A before visiting B. Formally, for any $\pi = s_0 s_1 \ldots \in \texttt{Paths}^{\mathcal{M}}$,

$$\begin{aligned} &\pi \models \Diamond A \quad \text{iff} \quad \exists i \geq 0,\, s_i \in A, \\ &\pi \models A \,\mathbf{U}\, B \quad \text{iff} \quad \exists i \geq 0,\, s_i \in B \text{ and } \forall j < i,\, s_j \in A. \end{aligned}$$

The probability for $\Diamond A$ and $A \,\mathbf{U}\, B$ to hold in state s is denoted by $\mathbb{P}_s(s \models \Diamond A)$ and $\mathbb{P}_s(s \models A \,\mathbf{U}\, B)$, i.e.

$$\begin{aligned} &\mathbb{P}_s(s \models \Diamond A) = \mathbb{P}_s(\{\pi \in \texttt{Paths}^{\mathcal{M}}(s) \mid \pi \models \Diamond A\}), \\ &\mathbb{P}_s(s \models A \,\mathbf{U}\, B) = \mathbb{P}_s(\{\pi \in \texttt{Paths}^{\mathcal{M}}(s) \mid \pi \models A \,\mathbf{U}\, B\}). \end{aligned}$$

In addition to temporal operator $\Diamond$ and $\mathbf{U}$, we also consider the expected cumulated cost in MC. Suppose $\mathcal{M}$ is also associated with a cost function $\mathit{cost}\colon S \to \mathbb{R}$. Intuitively, the value $\mathit{cost}(s)$ stands for the cost paid on leaving state s. The *accumulated cost* for a finite path $\widehat{\pi} = s_0 s_1 \ldots s_n$ is defined by

$$\mathit{cost}(\widehat{\pi}) = \mathit{cost}(s_0) + \mathit{cost}(s_1) + \ldots + \mathit{cost}(s_{n-1}).$$

Based on accumulated cost for finite paths, we now consider the accumulated cost paid along an infinite path until reaching A, formally, for any $\pi = s_0 s_1 \ldots \in \texttt{Paths}^{\mathcal{M}}$, define

$$\mathit{cost}(\pi, \Diamond A) = \begin{cases} \mathit{cost}(s_0 \ldots s_n) & \text{if } s_n \in A,\, \forall i < n,\, s_i \notin A \\ \lim\limits_{n \to \infty} \mathit{cost}(s_0 \ldots s_n) & \text{if } \pi \not\models \Diamond A \end{cases}$$

The expected cost until reaching A from initial state s for MC $\mathcal{M} = (S, \mathbf{P})$ is then defined by

$$\texttt{ExpCost}(s \models \Diamond A) = E_s\left[\mathit{cost}(\pi, \Diamond A)\right].$$

Intuitively, $\texttt{ExpCost}(s \models \Diamond A)$ represents the averaged cost until reaching A over all infinite paths w.r.t. probability measure $\mathbb{P}_s$.

Problem Formulation. Given a possibly infinite MC $\mathcal{M} = (S, \mathbf{P})$, estimate the following quantitative properties

$$\mathbb{P}_s(s \models \Diamond A), \qquad \mathbb{P}_s(s \models A \,\mathbf{U}\, B), \qquad \texttt{ExpCost}(s \models \Diamond A)\ , \tag{1}$$

and relate them to the quantitative analysis of probabilistic programs.

3 Quantitative Analysis of Markov Chains

In this section, we show how to estimate (upper and lower bounds) the quantitative properties in Eq. (1). In order to characterize these properties uniformly, we first propose a novel value function $V(s)$ which can represent $\mathbb{P}^s(s \models \Diamond A)$, $\mathbb{P}^s(s \models A \,\mathbf{U}\, B)$, and $\texttt{ExpCost}(s \models \Diamond A)$ by choosing different parameters, then show $V(x)$ is a solution to a typical equation system, and finally propose sufficient conditions to find upper and lower bounds on $V(x)$.

The key ingredient of $V(x)$ is a properly defined stopping time. For any set of states $H \subseteq S$, let T_H be the random variable that represents the transition time of an infinite path before reaching H, formally,

$$T_H \colon \texttt{Paths}^{\mathcal{M}} \to \mathbb{N},$$
$$s_0 s_1 s_2 \ldots \mapsto \begin{cases} \inf\{\, n \in \mathbb{N} \mid s_n \in H \,\} & \text{if exists } k,\ s_k \in H; \\ \infty & \text{if forall } n,\ s_n \notin H. \end{cases}$$

Recall that X_i represents the system state after i^{th} transition; the random variable representing the system state upon reaching H is defined by X_{T_H}, formally,

$$X_{T_H} \colon \texttt{Paths}^{\mathcal{M}} \to S,$$
$$s_0 s_1 s_2 \ldots \mapsto s_k \text{ if } \inf\{\, n \in \mathbb{N} \mid s_n \in H \,\} = k.$$

We now present the definition of value function $V(s)$:

Definition 2. *Given Markov chain $\mathcal{M} = (S, \mathbf{P})$ and $H \subseteq S$, for any function $f \colon S \to \mathbb{R}$, $g \colon S \to \mathbb{R}$, value function $V(s)$ is defined as follows*[7]*,*

$$V(s) = E_s\left[\sum_{i=0}^{T_H - 1} f(X_i) + \mathbf{1}_{T_H < \infty} \cdot g(X_{T_H})\right]. \tag{2}$$

Intuitively, if $f(s)$ and $g(s)$ represent the cost on state s, then $V(s)$ is the expected cost upon reaching H over all infinite paths w.r.t. transition cost f and termination cost g. By choosing different function f, g, and set H, value function $V(s)$ can represent $\mathbb{P}^s(s \models \Diamond A)$, $\mathbb{P}^s(s \models A \,\mathbf{U}\, B)$, and $\texttt{ExpCost}(s \models \Diamond A)$ uniformly. In detail, for any set $A, B \subseteq S$,

[7] Here we implicitly assume summation $\sum_{i=0}^{T_H - 1} f(X_i)$ converges (including converging to infinity) almost surely. If this is not the case, we may view $V(s)$ as an interval function. The convergence issue is similar to the well-definedness of the Lebesgue integral. See Appx. A for details.

1. let $H = A$, and
$$f = 0, \quad g(s) = \begin{cases} 1 & \text{if } s \in A \\ 0 & \text{if } s \in S \backslash A \end{cases},$$
then
$$V(s) = \mathbb{P}^s(s \models \Diamond A).$$

2. let $H = (S \backslash A) \cup B$, and
$$f = 0, \quad g(s) = \begin{cases} 1 & \text{if } s \in B \\ 0 & \text{otherwise} \end{cases},$$
then
$$V(s) = \mathbb{P}^s(s \models A \,\mathbf{U}\, B).$$

Proof. By Def. 2, we have
$$\begin{aligned} V(s) &= E_s\left[\mathbf{1}_{T_H < \infty} \cdot g(X_{T_H})\right] \\ &= \mathbb{P}_s\big(\{\pi \in \texttt{Paths}^{\mathcal{M}}(s) \mid T_H(\pi) < \infty, g(X_{T_H})(\pi) = 1\}\big). \end{aligned}$$
For any infinite path $\pi \in \texttt{Paths}^{\mathcal{M}}(s)$, $T_H(\pi) < \infty$ and $g(X_{T_H})(\pi) = 1$ if and only if for the first time π hits $H = (S \backslash A) \cup B$, it hits B, which implies $\pi \models A \,\mathbf{U}\, B$. Thus we have $V(s) = \mathbb{P}^s(s \models A \,\mathbf{U}\, B)$.

3. let $H = A$, and
$$f(s) = \mathit{cost}(s), \quad g(s) = 0,$$
then
$$V(s) = \texttt{ExpCost}(s \models \Diamond A).$$

Note that the proofs for 1 and 3 are straightforward and are therefore omitted for brevity.

Therefore, to estimate quantitative properties in Eq. (1), it suffices to estimate value function $V(s)$. We first show that $V(s)$ serves as the solution to a related equation system by leveraging the Markov property.

Theorem 1. *Given a Markov chain $\mathcal{M} = (S, \mathbf{P})$, value function $V(s)$ satisfies the following equation system:*
$$\begin{aligned} \int_S V(t)\, \mathbf{P}(s, \mathrm{d}t) + f(s) = V(s), \quad \textit{if } s \in S \backslash H, \\ V(s) = g(s), \quad \textit{if } s \in H. \end{aligned} \tag{3}$$

Proof. For any $s \in H$, $V(s) = g(s)$ trivially holds. For any $s \in X \backslash H$, as $\{X_i\}_{i \in \mathbb{N}}$ is a Markov process, by Markov property and properties of conditional expecta-

tion, we have

$$\begin{aligned}
&E_s\left[\sum_{i=0}^{T_H-1} f(X_i) + \mathbf{1}_{T_H<\infty}\cdot g(X_{T_H}) \,\middle|\, X_1\right] \\
&= f(s) + E_s\left[\sum_{i=1}^{T_H-1} f(X_i) + \mathbf{1}_{T_H<\infty}\cdot g(X_{T_H}) \,\middle|\, X_1\right] \\
&= f(s) + E_{X_1}\left[\sum_{i=0}^{T_H-1} f(X_i) + \mathbf{1}_{T_H<\infty}\cdot g(X_{T_H})\right] \\
&= f(s) + V(X_1).
\end{aligned}$$

Taking expectation w.r.t. probability $\mathbb{P}_s$ on both sides, we have

$$E_s\left[E_s\left[\sum_{i=0}^{T_H-1} f(X_i) + \mathbf{1}_{T_H<\infty}\cdot g(X_{T_H})\middle|\, X_1\right]\right] = f(s) + E_s\left[V(X_1)\right],$$

where the left-hand side equals $V(s)$ by properties of conditional expectation, and the right-hand side is equivalent to taking expectation w.r.t. probability measure $\mathbf{P}(s,\cdot)$ by the construction of $\mathbb{P}_s$, thus we have

$$V(s) = \int_S V(t)\,\mathbf{P}(s,\mathrm{d}t) + f(s).$$

This completes the proof. □

Remark 1 (Connections to Probabilistic Model Checking). When Markov chain $\mathcal{M} = (S, \mathbf{P})$ is finite, Eq. (3) reduces to a linear system due to the finiteness of the state space. This system corresponds exactly to the set of equations used to characterize the temporal properties of Markov chains in probabilistic model checking [7]. □

Although Thm. 1 establishes that $V(s)$ is a solution to Eq. (3), computing $V(s)$ explicitly is infeasible (for infinite Markov chain), as Eq. (3) may have multiple solutions. Even when Eq. (3) has a unique solution, solving it remains challenging. Therefore, a more practical approach involves approximating $V(s)$ by determining its upper and lower bounds.

The classical Fatou's Lemma, as presented in [25], validates the interchange of integration and limits. In this work, we extend Fatou's Lemma to suit our specific requirements.

Lemma 1 (Generalized Fatou's lemma). *Let $\{X_n\}_{n\in\mathbb{N}}$ be a sequence of random variables,*

- *If $\{(X_n)^-\}_{n\in\mathbb{N}}$ is uniformly integrable, then*

$$E[\liminf_n X_n] \le \liminf_n E[X_n].$$

– *If* $\{(X_n)^+\}_{n\in\mathbb{N}}$ *is uniformly integrable, then*

$$\limsup_n E[X_n] \le E[\limsup_n X_n].$$

Proof. For the first part, let $X = \liminf_n X_n$. Due to the uniform integrability of $\{(X_n)^-\}_{n\in\mathbb{N}}$, for any $\epsilon > 0$, there exists c such that $E[X_n^-; (X_n)^- > c] \le \epsilon$ for any $n \in \mathbb{N}$ (recalling $E[X; A]$ denotes $E[X \cdot \mathbf{1}_A]$). Since $X + c \le \liminf_n (X_n + c)^+$, we have

$$E[X] + c \le E[\liminf_n (X_n + c)^+] \le \liminf_n E[(X_n + c)^+].$$

Moreover, we have that

$$(X_n + c)^+ = X_n + c + (X_n + c)^- \le X_n + c + \mathbf{1}_{X_n < -c} \cdot (X_n)^-$$

holds, which implies

$$\begin{aligned} E[X] + c &\le \liminf_n E[(X_n + c)^+] \\ &\le \liminf_n (E[X_n] + c + E[(X_n)^-; (X_n)^- > c]) \\ &\le \liminf_n (E[X_n] + c + \epsilon). \end{aligned}$$

Since ϵ is an arbitrary positive number, the result follows. For the second part, substituting $\{X_n\}_{n\in\mathbb{N}}$ with $\{-X_n\}_{n\in\mathbb{N}}$, the result follows similarly. □

Given a Markov chain $\mathcal{M} = (S, \mathbf{P})$, random variable ${}^uY_n$ is defined by

$${}^uY_n \triangleq u(X_n) + \sum_{i=0}^{n-1} f(X_i) \tag{4}$$

for any function $u\colon S \to \mathbb{R}$, where X_i represents system state after i^{th} transition. We may omit prescript u in ${}^uY_n$ when it is clear from the context. The following theorem indicates how to find upper bounds on value function $V(x)$.

Theorem 2. *Given a Markov chain* $\mathcal{M} = (S, \mathbf{P})$*, suppose there exists a function* $u\colon S \to \mathbb{R}$ *such that the process*

$$\{({}^uY_{n\wedge T_H})^-\}_{n\in\mathbb{N}}$$

is uniformly integrable, then $u(s) \ge V(s)$ *for* $s \in S$ *if one of the following conditions holds:*

– $\mathbb{P}_s(T_H < \infty) = 1$ *for any* $s \in S$*, and*

$$\begin{aligned} \int_S u(t)\,\mathbf{P}(s, \mathrm{d}t) + f(s) \le u(s), &\quad \text{if } s \in S\backslash H, \\ u(s) \ge g(s), &\quad \text{if } s \in H. \end{aligned} \tag{5}$$

– *the following equation holds*

$$\begin{aligned} \int_S u(t)\,\mathbf{P}(s,\mathrm{d}t) + f(s) \le u(s), \quad \textit{if } s \in S\backslash H, \\ u(s) \ge 0, \quad \textit{if } s \in S\backslash H, \\ u(s) \ge g(s), \quad \textit{if } s \in H. \end{aligned} \tag{6}$$

Proof. For any $s \in H$, Eq. (5) and Eq. (6) imply $u(s) \ge g(s) = V(s)$. For any $s \in S\backslash H$, Eq. (5) and Eq. (6) imply

$$\int_S u(t)\,\mathbf{P}(s,\mathrm{d}t) + f(s) \le u(s), \quad \text{if } s \in S\backslash H.$$

Thus for any $\pi \in \mathtt{Paths}^{\mathcal{M}}$, if $T_H(\pi) > n+1$, then

$$\begin{aligned} E_s[Y_{(n+1)\wedge T_H} \mid \mathcal{F}_n](\pi) &= E_s\left[u(X_{n+1}) + \sum_{i=0}^{n+1} f(X_i) \mid \mathcal{F}_n\right](\pi) \\ &= E_s\left[u(X_{n+1}) \mid \mathcal{F}_n\right](\pi) + \sum_{i=0}^{n} f(X_i)(\pi) \\ &= \int_S u(t)\,\mathbf{P}(X_n,\mathrm{d}t)(\pi) + f(X_n)(\pi) + \sum_{i=0}^{n-1} f(X_i)(\pi) \\ &\le u(X_n)(\pi) + \sum_{i=0}^{n-1} f(X_i)(\pi) = Y_n(\pi) = Y_{n\wedge T_H}(\pi), \end{aligned}$$

and if $T_H(\pi) \le n$, then

$$Y_{(n+1)\wedge T_H}(\pi) = Y_{n\wedge T_H}(\pi).$$

In this case, we still have

$$E_s[Y_{(n+1)\wedge T_H}(\pi) \mid \mathcal{F}_n] \le Y_{n\wedge T_H}(\pi)$$

Then it follows that $\{Y_{n\wedge T_H}\}_{n\in\mathbb{N}}$ is a supermartingale. By properties of supermartingale, we have

$$E_s[Y_{n\wedge T_H}] \le E_s[Y_0] = u(s).$$

holds for any $n \in \mathbb{N}$. Moreover, since $\{(Y_{n\wedge T_H})^-\}_{n\in\mathbb{N}}$ is uniformly integrable, properties of supermartingale [25] imply $\lim_n Y_{n\wedge T_H})$ converges almost surely, thus Lem. 1 further implies

$$E_s[\lim_{n\to\infty} Y_{n\wedge T_H}] \le \lim_{n\to\infty} E_s[Y_{n\wedge T_H}] \le u(s)$$

If $\mathbb{P}_s(T_H < \infty) = 1$ for any $s \in S$ and Eq. (5) holds, then $u(s) \ge g(s)$ for $s \in H$, thus

$$\begin{aligned} \lim_{n\to\infty} Y_{n\wedge T_H} &= \lim_{n\to\infty}\left(u(X_{n\wedge T_H}) + \sum_{i=0}^{n\wedge T_H - 1} f(X_i)\right) \\ &= u(X_{T_H}) + \sum_{i=0}^{T_H-1} f(X_i) \ge g(X_{T_H}) + \sum_{i=0}^{T_H-1} f(X_i) \end{aligned}$$

By taking expectation on both sides, we have

$$V(s) \le E_s[\lim_{n\to\infty} Y_{n\wedge T_H}] \le u(s)$$

If Eq. (6) holds, then $u(s) \ge g(s)$ for $s \in H$, and $u(s) \ge 0$ for $s \in S\backslash H$, thus

$$\begin{aligned}\lim_{n\to\infty} Y_{n\wedge T_H} &= \lim_{n\to\infty}\left(u(X_{n\wedge T_H}) + \sum_{i=0}^{n\wedge T_H-1} f(X_i)\right)\\ &\ge \mathbf{1}_{T_H<\infty}\cdot u(X_{T_H}) + \sum_{i=0}^{T_H-1} f(X_i)\\ &\ge \mathbf{1}_{T_H<\infty}\cdot g(X_{T_H}) + \sum_{i=0}^{T_H-1} f(X_i).\end{aligned}$$

Thus, we still have $V(s) \le E_s[\lim_{n\to\infty} Y_{n\wedge T_H}] \le u(s)$, This completes the proof. □

Sufficient conditions for finding lower bounds on value function $V(s)$ can be dually formulated as follows.

Theorem 3. *Given a Markov chain $\mathcal{M} = (S, \mathbf{P})$, suppose there exists a function $u\colon S \to \mathbb{R}$ such that the process*

$$\{({}^{u}Y_{n\wedge T_H})^{+}\}_{n\in\mathbb{N}}$$

is uniformly integrable, then $u(s) \le V(s)$ for $s \in S$ if one of the following conditions holds:

- $\mathbb{P}_s(T_H < \infty) = 1$ *for any $s \in S$, and*

$$\begin{aligned}\int_S u(t)\,\mathbf{P}(s,\mathrm{d}t) + f(s) \ge u(s), &\quad \text{if } s \in S\backslash H\\ u(s) \le g(s), &\quad \text{if } s \in H\end{aligned} \tag{7}$$

- *the following equation holds*

$$\begin{aligned}\int_S u(t)\,\mathbf{P}(s,\mathrm{d}t) + f(s) \ge u(s), &\quad \text{if } s \in S\backslash H\\ u(s) \le 0, &\quad \text{if } s \in S\backslash H\\ u(s) \le g(s), &\quad \text{if } s \in H\end{aligned} \tag{8}$$

Proof. The result follows from Thm. 2 if we replace f, g with $-f$, $-g$, then finding lower bounds on the original $V(s)$ is equivalent to finding upper bounds on the new $V(s)$. □

As demonstrated in Thm. 2 and Thm. 3, determining upper and lower bounds necessitates verifying the uniform integrability of $\{({}^{u}Y_{n\wedge T_H})^{-}\}_{n\in\mathbb{N}}$ and $\{({}^{u}Y_{n\wedge T_H})^{+}\}_{n\in\mathbb{N}}$, respectively. This verification is challenging by definition. However, the classical Optional Stopping Theorem offers sufficient conditions to confirm uniform integrability, simplifying the process.

Theorem 4 (Optional Stopping Theorem [25,60]). *Let T be a stopping time w.r.t. $\mathcal{F}_n$, and $\{X_n\}_{n\in\mathbb{N}}$ is a stochastic process adapted to $\mathcal{F}_n$, such that $E[X_n] < \infty$ for all $n \in \mathbb{N}$, Process $\{X_{n\wedge T}\}_{n\in\mathbb{N}}$ is uniformly integrable if one of the following conditions holds:*

- *T is bounded almost surely, i.e. there exists $N \in \mathbb{N}$ such that $\mathbb{P}(T \leq N) = 1$;*
- *$X_{n\wedge T}$ is bounded, i.e. there exists constant $C \in \mathbb{R}^+$ such that $|X_{n\wedge T}| \leq C$ almost surely;*
- *$E[T] < \infty$ and $X_{n\wedge T}$ is conditional difference bounded, i.e. there exists $M > 0$, such that for any $n \in \mathbb{N}$,*

$$E[|X_{(n+1)\wedge T} - X_{n\wedge T}| \mid \mathcal{F}_n] \leq M.$$

then $\{X_{n\wedge T}\}_{n\in\mathbb{N}}$ is uniformly integrable.

Based on optional stopping theorem, the following results ensure the uniform integrability of $\{({}^uY_{n\wedge T_H})^-\}_{n\in\mathbb{N}}$ and $\{({}^uY_{n\wedge T_H})^+\}_{n\in\mathbb{N}}$.

Lemma 2. *The stochastic process $\{({}^uY_{n\wedge T_H})^-\}_{n\in\mathbb{N}}$ (resp. $\{({}^uY_{n\wedge T_H})^+\}_{n\in\mathbb{N}}$) is uniformly integrable if one of the following conditions hold.*

- *T_H is bounded almost surely, i.e. there exists $N \in \mathbb{N}$ such that $\mathbb{P}_s(T_H \leq N) = 1$ for any $s \in S$;*
- *u^- (resp. u^+) is bounded and $f^- = 0$ (resp. $f^+ = 0$);*
- *$E_s[T_H] < \infty$ for any $s \in S$, f^- (resp. f^+) is bounded, and there exists $C \in \mathbb{R}^+$, such that*

$$\begin{aligned} &\int_S \left|u^-(t) - u^-(s)\right| \mathbf{P}(s, \mathrm{d}t) \leq C \\ (resp.\ &\int_S \left|u^+(t) - u^+(s)\right| \mathbf{P}(s, \mathrm{d}t) \leq C) \end{aligned} \tag{9}$$

for any $s \in S \backslash H$.

Proof. We give a proof for $\{({}^uY_{n\wedge T_H})^-\}_{n\in\mathbb{N}}$, the proof for $\{({}^uY_{n\wedge T_H})^+\}_{n\in\mathbb{N}}$ is similar. By Eq. (4), we have

$$0 \leq ({}^uY_n)^- \leq u^-(X_n) + \sum_{i=0}^{n-1} f^-(X_i)$$

It suffices to prove the uniform integrability of $Z_{n\wedge T_H}$, where Z_n is defined by

$$Z_n \triangleq u^-(X_n) + \sum_{i=0}^{n-1} f^-(X_i)$$

The above three sufficient conditions directly correspond to three sufficient conditions formulated in Thm. 4. □

Remark 2. Eq. (9) can be relaxed to

$$\int_S |u(t) - u(s)| \mathbf{P}(s, \mathrm{d}t) \leq C$$

since $|u^-(t) - u^-(s)| \leq |u(t) - u(s)|$ and $|u^+(t) - u^+(s)| \leq |u(t) - u(s)|$ for any $t, s \in S$. □

3.1 Extension to MDP

Our approach can be readily adapted to the quantitative analysis of Markov Decision Processes (MDPs). An MDP represents a *demonic* nondeterministic extension of a Markov Chain and reduces to a Markov Chain when a scheduler resolves the nondeterminism.

Considering demonic nondeterminism, the value function is generalized to

$$V(s) = \sup_{\sigma} E_s \left[\sum_{i=0}^{T_H - 1} f(X_i^{\sigma}) + \mathbf{1}_{T_H < \infty} \cdot g(X_{T_H}^{\sigma}) \right], \tag{10}$$

where σ represents a scheduler that resolves nondeterminism. Also, the equational characterization for determining upper and lower bounds of V must account for the calculation of suprema across the action set. For instance, Eq. (5), which characterizes the upper bound for a Markov Chain, extends to

$$\begin{aligned} \sup_{a \in \mathsf{Act}(s)} \int_S u(t)\, \mathbf{P}(s, a, \mathrm{d}t) + f(s) \le u(s), \quad & \text{if } s \in S \backslash H, \\ u(s) \ge g(s), \quad & \text{if } s \in H \end{aligned} \tag{11}$$

for the MDP scenario, where $\mathsf{Act}(s)$ represents the action set at state s. Solving Eq. (11) establishes an upper bound on the value function V. Similarly, Eq. (7) for the lower bound extends to

$$\begin{aligned} \sup_{a \in \mathsf{Act}(s)} \int_S u(t)\, \mathbf{P}(s, a, \mathrm{d}t) + f(s) \ge u(s), \quad & \text{if } s \in S \backslash H, \\ u(s) \le g(s), \quad & \text{if } s \in H \end{aligned} \tag{12}$$

in the MDP context. It is important to note that while the computational complexity of solving Eq. (11) (i.e., finding the upper bound) remains consistent with the Markov Chain case, finding solutions for Eq. (12) (i.e., determining the lower bound) is considerably more challenging, as it involves computing the suprema over all possible actions.

4 Quantitative Analysis of Probabilistic Programs

In this section, we establish a link between the quantitative analysis of the Markov chain in Sect. 3 and the quantitative analysis of probabilistic programs. We will demonstrate how varying the parameters in $V(x)$ can represent different quantitative properties of a probabilistic program within the program's semantic Markov chain.

Recall that for any program C, there exists a corresponding Markov chain $\mathcal{M}_C = (S_C, \mathbf{P}_C)$, with S_C is defined as $L \times \mathsf{Val}$, representing the product of program counters and variable valuations. Within L, there are two notable counters: the starting counter l_{in} and the ending counter l_{end}.

4.1 Assertion Violation

Assertion violation analysis constitutes a fundamental problem in the quantitative assessment of probabilistic programs. This analysis aims to estimate the probability that a given assertion will be violated before the program terminates. Assertion violation was first considered in [17], and further investigated in [21,22] via concentration inequality. In [57], the authors demonstrate that the probability of an assertion violation represents the least solution to a specific equation. They also propose sufficient conditions for establishing upper and lower bounds on this probability. In this subsection, we will re-establish the results in [57] within our own framework.

Suppose the assertion we aim to avoid is $B \subseteq L \times \mathsf{Val}$, and let l_{end} denote the ending location, then $\mathbb{P}_s\left(s \models \left((L \setminus l_{\mathsf{end}}) \times \mathsf{Val}\right) \mathbf{U}\, B\right)$ is exactly the assertion violation probability with undesirable set B. According to Sect. 3,

$$V(s) \;=\; \mathbb{P}_s\left(s \models \left((L \setminus l_{\mathsf{end}}) \times \mathsf{Val}\right) \mathbf{U}\, B\right) \;=\; \text{Assertion violation probability}$$

with

$$H = (l_{\mathsf{end}} \times \mathsf{Val}) \cup B, \quad f = 0, \quad g(s) = \begin{cases} 1 & \text{if } s \in B \\ 0 & \text{otherwise.} \end{cases}$$

Thus Thm. 2 and Thm. 3 provide sufficient conditions for obtaining upper and lower bounds on assertion violation probability.

4.2 Expected Running time

Expected running time is one of the most important properties of probabilistic programs. There are various works considering to calculate the bounds for the expected running time, e.g., [16,20] employs martingale techniques to find upper bound of expected running time for programs, [34,35] present a wp-style calculus for obtaining bounds on expected running time, [30] proposes conditions to verify lower bound on expected running time for programs without nondeterminism. In this subsection, we show how our approach entails the proof rules for establishing upper and lower bounds on expected running time proposed in [30].

Since l_{end} is the ending location, we have

$$V(s) \;=\; E[T_H] \;=\; \text{Expected running time of } C$$

with

$$H = (l_{\mathsf{end}} \times \mathsf{Val}), \quad f = 1, \quad g(s) = 0.$$

Thus Thm. 2 and Thm. 3 are exactly the same[8] canonical proof rules for upper and lower bound proposed in [16,30].

[8] Note in this case, side conditions (i.e. uniformly integrability) in Thm. 2 trivially holds, thus Thm. 2 is essentially the Park's induction rule.

4.3 Expected Accumulated Cost

Expected running time analysis can be naturally generalized[9] to cost analysis, i.e., calculating the expected resource consumption of the programs. In [48], the authors first consider the expected cost with bounded non-negative transition cost, [58] further considers mixed-sign transition costs (i.e. both positive and negative cost) for probabilistic programs with nondeterminism and derives upper and lower bounds for the expected cost, [56] presents conditions to derive bounds on higher moments of expected accumulated cost, and so forth. Again, we demonstrate how our methods establish the results in [58].

Suppose one-step transition cost is denoted by $cost\colon L \times \mathsf{Val} \to \mathbb{R}$, then $\texttt{ExpCost}(s \models \Diamond(l_{\mathsf{end}} \times \mathsf{Val})$ represents the expected cost of program C. Thus

$$V(s) \;=\; \texttt{ExpCost}(s \models \Diamond(l_{\mathsf{end}} \times \mathsf{Val})) \;=\; \text{Expected Cost of } C$$

with

$$H = (l_{\mathsf{end}} \times \mathsf{Val}), \quad f = cost, \quad g(s) = 0.$$

Therefore, Thm. 2 and Thm. 3 provide sufficient conditions for obtaining upper and lower bounds on expected cost, which are the same as [58].

4.4 Expectation Analysis

The weakest-precondition calculus [24] offers a logical framework for formally reasoning about classical programs. Its probabilistic counterpart, the weakest pre-expectation calculus [44], extends this framework to accommodate probabilistic programs. This extension provides a robust deductive verification framework for analyzing probabilistic behaviors. Intuitively, the weakest pre-expectation transformer $\mathsf{wp}\,[\![C]\!]\,(g)$ represents the expected value of g after program C terminates. Within our framework, we have

$$V(s) \;=\; \mathsf{wp}\,[\![C]\!]\,(g)\,(s) \;=\; \mathbb{E}[g(X_{T_H}) \cdot \mathbf{1}_{T_H<\infty}],$$

with

$$H = (l_{\mathsf{end}} \times \mathsf{Val}), \quad f = 0, \quad g = g.$$

Thus Thm. 2 and Thm. 3 also incorporate results about upper and lower bounding $\mathsf{wp}\,[\![C]\!]\,(g)$ proposed in [30].

Example 1. Consider the following 1-D biased random walk,

$$C_{\text{1dbrw}}\colon \quad \texttt{while}\,(\,n > 0\,)\,\{\,n \coloneqq n-1\ [2/3]\ n \coloneqq n+1\,\}\ .$$

We are interested in bounding the expected running time of C_{1dbrw}. According to Sect. 4.2, it suffices to solve Eq. (5) with $f = 1$, $g = 0$:

$$\begin{aligned} \frac{2}{3}u(x-1) + \frac{1}{3}u(x+1) + 1 \le u(x), &\ \text{ if } x > 0, \\ u(x) \ge 0 \quad\ , &\ \text{ if } x \le 0; \end{aligned} \tag{13}$$

[9] The expected running time analysis is a special case of cost analysis, where the transition cost can be taken as a constant (e.g., 1).

to find upper bounds on expected running time and solve Eq. (7) with $f = 1$, $g = 0$:

$$\begin{aligned} \frac{2}{3}u(x-1) + \frac{1}{3}u(x+1) + 1 \geq u(x), &\quad \text{if } x > 0, \\ u(x) \leq 0 \quad , &\quad \text{if } x \leq 0; \end{aligned} \tag{14}$$

to find lower bounds. Eqs. (13) and (14) can be encoded as semidefinite programming problems [19], and further solved by an off-the-shelf solver like Mosek [6]. □

5 Extension to Bayesian Programming

In this section, we demonstrate how our approach can be adapted to address inference problems in Bayesian programming.

Bayesian programming [55,51] is a specific programming paradigm that models Bayesian models as probabilistic programs. In a nutshell, compared with (standard) probabilistic programs, Bayesian programming languages have one specific construct: `score` (a.k.a. `observe`) [14,10], which is used to record the likelihood of observed data in the form of "`score`(`weight`)". The syntax of our Bayesian probabilistic programming language is

$$\begin{aligned} C ::= &\ \mathsf{skip} \mid x := e \mid x :\approx \mu \mid C; C \mid \{C\}\ [p]\ \{C\} \mid \\ &\ \mathsf{if}\ (\varphi)\ \{C\}\ \mathsf{else}\ \{C\} \mid \mathsf{while}\ (\varphi)\ \{C\} \mid \mathsf{score}(\mathsf{weight}) \end{aligned}$$

where parameter `weight` can be a number $w \in \mathbb{R}$ or a probability density function. The semantics of `score` can be interpreted as weighting the current execution with the parameter `weight`[10].

Example 2 (Pedestrian [42,43,13,59]). A pedestrian has got lost on the way home and only knows that she is a uniform random distance between 0 and 3 km from her house. She repeatedly walks a uniform random distance of at most 1 km in either direction, until she arrives at her house. Upon her arrival, an odometer tells her that she has walked 1.1 km in total. However, this odometer was once broken and the measured distance is normally distributed around the true distance with a standard deviation of 0.1 km. The movement can be modeled by the probabilistic program in Fig. 2, together with its underlying CFG.

In Bayesian programming, one central problem is to infer the normalized posterior distribution of Bayesian programming. We show that our unified framework can be extended to infer the normalized posterior distribution for Bayesian programming by a direct adaption of value function $V(s)$.

[10] `score` is more expressive and general than `observe`. Actually, the relation of the keyword `observe` and `score` is $\texttt{score}(f(D)) = \texttt{observe}\ D\ \texttt{from}\ p$, where p is a predefined probabilistic distribution, f is the corresponding probability density function, and D is a measurable set. See more details in [10]. When using `observe` to filter the admissible executions with a Boolean condition D (usually called hard conditioning), we also can express it with $\mathsf{if}\ (D)\ \{\mathsf{score}(1)\}\ \mathsf{else}\ \{\mathsf{score}(0)\}$.

```
        start :≈ uniform(0, 3)
        p := start; d := 0;
l_in :  while(p ≥ 0){
          r :≈ uniform(0, 1);
          {p := p − r [0.5] p := p + r}
          d := d + r;
          }
        score(pdf(normal(d, 0.1), 1.1));
l_end : return start
```

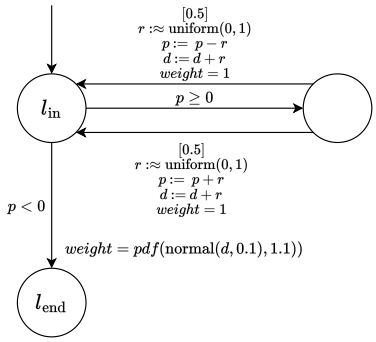

Fig. 2: Pedestrian random walk (left) and its simplified CFG (right), where location label l_{in} and l_{end} represent the starting and ending point of the program, respectively.

Given a measurable set U, the normalized posterior distribution of Bayesian program P w.r.t. set U is defined by:

$$\mathsf{posterior}(U) = \frac{Z_U}{Z_P},$$

where Z_U is the expected accumulated weights that terminal states lie in $l_{\mathsf{end}} \times U$, and Z_P is the normalising constant[11]. Thus, the key challenge in bounding the normalized posterior distribution lies in finding upper and lower bounds on the expected accumulated weights Z_U and Z_P.

Problem Formulation. Given a probabilistic program P and a measurable set U, we aim to estimate the expected accumulated weights Z_P and Z_U after the termination of program P.

To address this problem, we refine the value function $V(s)$ by incorporating a weight factor and eliminating the transition cost, yielding the following formulation:

$$V(s) = E_s\left[\prod_{i=0}^{T_H-1} \texttt{weight}(X_i) \cdot \mathbf{1}_{T_H<\infty} \cdot g(X_{T_H})\right],$$

where $\texttt{weight}(X_i)$ represents[12] the weight factor at state X_i.

[11] We assume $Z_P < \infty$.

[12] If the program location associated with X_i contains no `score` statement, the weight factor defaults to 1, i.e. $\texttt{weight}(X_i) = 1$.

Based on the adapted value function, if H and g are further defined by

$$H = (l_{\mathsf{end}} \times \mathsf{Val}), \quad g(s) = \begin{cases} 1 & \text{if } s \in U \\ 0 & \text{otherwise.} \end{cases}$$

then $V(s) = Z_U$. Similarly, $V(s) = Z_P$ if $H = (l_{\mathsf{end}} \times \mathsf{Val})$ and $g(s) = 1$. Thus, it suffices to find upper and lower bounds on value $V(s)$. Let

$$^{u}Y_n \triangleq u(X_n) \cdot \prod_{i=0}^{n-1} \mathtt{weight}(X_i), \tag{15}$$

for any $u : S \to \mathbb{R}$, the following results, analogous to Thm. 2, present sufficient conditions to obtain upper bounds on $V(s)$.

Theorem 5. *Given a Bayesian Program P and its underlying Markov chain $\mathcal{M} = (S, \mathbf{P})$, suppose there exists a function $u\colon S \to \mathbb{R}$ such that the process*

$$\{(^{u}Y_{n\wedge T_H})^{-}\}_{n\in\mathbb{N}}$$

is uniformly integrable, then $u(s) \geq V(s)$ for $s \in S$ if one of the following conditions holds:

- $\mathbb{P}_s(T_H < \infty) = 1$ *for any $s \in S$, and*

$$\begin{aligned} \int_S u(\mathrm{d}t)\, \mathbf{P}(s, \mathrm{d}t) \cdot \mathtt{weight}(s) \leq u(s), \quad & \text{if } s \in S\backslash H \\ u(s) \geq g(s), \quad & \text{if } s \in H \end{aligned} \tag{16}$$

- *the following equation holds*

$$\begin{aligned} \int_S u(\mathrm{d}t)\, \mathbf{P}(s, \mathrm{d}t) \cdot \quad \mathtt{weight}(s) \leq u(s), \quad & \text{if } s \in S\backslash H \\ u(s) \geq 0, \quad & \text{if } s \in S\backslash H \\ u(s) \geq g(s), \quad & \text{if } s \in H \end{aligned} \tag{17}$$

Remark 3. A similar result can be obtained for deriving lower bounds on $V(s)$. □

Proof. The proof closely resembles the proof of Thm. 2. Since Eq. (16) holds, one can directly verify $\{Y_{n\wedge T_H}\}_{n\in\mathbb{N}}$ is a supermartingale. Thus, by properties of supermartingale, we have

$$E_s[Y_{n\wedge T_H}] \leq E_s[Y_0] = u(s).$$

holds for any $n \in \mathbb{N}$. by taking limit on both sides and following a similar argument in Thm. 2, we obtain $u(s) \geq V(s)$. □

Example 3. We simply illustrate how our extended framework is applied to address the Bayesian inference through Exmp. 2. Without loss of generality, we take the calculation of the upper bound and lower bound for Z_P as an example. According to Thm. 5, we have the following characterization for the upper bound of Z_P:

$$\begin{aligned} 0.5\mathbb{E}_r[u_1(p-r,d+r)] + 0.5\mathbb{E}_r[u_1(p+r,d+r)] &\leq u_1(p,d), \quad \text{if } p \geq 0 \wedge s \in l_{\mathsf{in}}, \\ pdf(normal(d,0.1),1.1) \cdot u_2(p,d) &\leq u_1(p,d), \quad \text{if } p < 0 \wedge s \in l_{\mathsf{in}}, \\ u_2(p,d) &\geq 1 \quad , \quad \text{if } s \in l_{\mathsf{end}}, \end{aligned} \tag{18}$$

where u_1 is the value function on location l_{in} and u_2 is the value function on location l_{end}. Eq. (18) can be encoded into semidefinite programming problems using Putinar's Positivstellensatz [49], and further solved by a off-the-shelf solver, e.g. Mosek [6]. □

Remark 4. The sufficient conditions for deriving upper and lower bounds on $V(x)$ (or expected accumulated weights) constitute the central finding in [59]. In this paper, we re-establish these results within our unified framework. □

6 Conclusion

We have introduced a unified framework for the quantitative analysis of probabilistic programs, which encompasses termination analysis, temporal property analysis, cost analysis, and expectation analysis. We illustrate how our approach can be adapted to tackle inference problems in Bayesian programming. In the future, we aim to expand our framework to encompass a broader spectrum of probabilistic models, such as weighted programs discussed in [12], and to explore efficient computational algorithms for determining upper and lower bounds.

Acknowledgments. We dedicate this article to our dear colleague Joost-Pieter Katoen on the occasion of his 60th birthday, who has been tirelessly pushing the limits of, amongst others, (automated) quantitative analysis of probabilistic programs. This work has been funded by the National Key R&D Program of China under grant No. 2022YFA1005101, by the NSFC under grant No. 62192732, by the ZJNSF Major Program under grant No. LD24F020013, by the Fundamental Research Funds for the Central Universities of China under grant No. 226-2024-00140, and by the ZJU Education Foundation's Qizhen Talent program.

References

1. A. Abate, M. Giacobbe, and D. Roy. Learning probabilistic termination proofs. In A. Silva and K. R. M. Leino, editors, *Computer Aided Verification - 33rd International Conference, CAV 2021, Virtual Event, July 20-23, 2021, Proceedings, Part II*, volume 12760 of *Lecture Notes in Computer Science*, pages 3–26. Springer, 2021.

2. A. Albarghouthi and J. Hsu. Constraint-based synthesis of coupling proofs. In H. Chockler and G. Weissenbacher, editors, *Computer Aided Verification - 30th International Conference, CAV 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings, Part I*, volume 10981 of *Lecture Notes in Computer Science*, pages 327–346. Springer, 2018.
3. A. Albarghouthi and J. Hsu. Synthesizing coupling proofs of differential privacy. *Proc. ACM Program. Lang.*, 2(POPL):58:1–58:30, 2018.
4. B. Alpern and F. B. Schneider. Defining liveness. *Inf. Process. Lett.*, 21(4):181–185, 1985.
5. D. Amrollahi, E. Bartocci, G. Kenison, L. Kovács, M. Moosbrugger, and M. Stankovic. Solving invariant generation for unsolvable loops. In G. Singh and C. Urban, editors, *Static Analysis - 29th International Symposium, SAS 2022, Auckland, New Zealand, December 5-7, 2022, Proceedings*, volume 13790 of *Lecture Notes in Computer Science*, pages 19–43. Springer, 2022.
6. E. D. Andersen, C. Roos, and T. Terlaky. On implementing a primal-dual interior-point method for conic quadratic optimization. *Math. Program.*, 95(2):249–277, 2003.
7. C. Baier and J. Katoen. *Principles of model checking*. MIT Press, 2008.
8. C. Baier, J. Klein, L. Leuschner, D. Parker, and S. Wunderlich. Ensuring the reliability of your model checker: Interval iteration for markov decision processes. In R. Majumdar and V. Kuncak, editors, *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, volume 10426 of *Lecture Notes in Computer Science*, pages 160–180. Springer, 2017.
9. G. Barthe, M. Gaboardi, B. Grégoire, J. Hsu, and P. Strub. Proving differential privacy via probabilistic couplings. In M. Grohe, E. Koskinen, and N. Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 749–758. ACM, 2016.
10. G. Barthe, J.-P. Katoen, and E. A. Silva. *Foundations of Probabilistic Programming*. Cambridge University Press, 2020.
11. K. Batz, M. Chen, S. Junges, B. L. Kaminski, J. Katoen, and C. Matheja. Probabilistic program verification via inductive synthesis of inductive invariants. In S. Sankaranarayanan and N. Sharygina, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 29th International Conference, TACAS 2023, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2022, Paris, France, April 22-27, 2023, Proceedings, Part II*, volume 13994 of *Lecture Notes in Computer Science*, pages 410–429. Springer, 2023.
12. K. Batz, A. Gallus, B. L. Kaminski, J. Katoen, and T. Winkler. Weighted programming: a programming paradigm for specifying mathematical models. *Proc. ACM Program. Lang.*, 6(OOPSLA1):1–30, 2022.
13. R. Beutner, C. L. Ong, and F. Zaiser. Guaranteed bounds for posterior inference in universal probabilistic programming. In R. Jhala and I. Dillig, editors, *PLDI '22: 43rd ACM SIGPLAN International Conference on Programming Language Design and Implementation, San Diego, CA, USA, June 13 - 17, 2022*, pages 536–551. ACM, 2022.
14. J. Borgström, U. D. Lago, A. D. Gordon, and M. Szymczak. A lambda-calculus foundation for universal probabilistic programming. In J. Garrigue, G. Keller, and E. Sumii, editors, *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming, ICFP 2016, Nara, Japan, September 18-22, 2016*, pages 33–46. ACM, 2016.

15. M. Carbin, S. Misailovic, and M. C. Rinard. Verifying quantitative reliability for programs that execute on unreliable hardware. In A. L. Hosking, P. T. Eugster, and C. V. Lopes, editors, *Proceedings of the 2013 ACM SIGPLAN International Conference on Object Oriented Programming Systems Languages & Applications, OOPSLA 2013, part of SPLASH 2013, Indianapolis, IN, USA, October 26-31, 2013*, pages 33–52. ACM, 2013.
16. A. Chakarov and S. Sankaranarayanan. Probabilistic program analysis with martingales. In N. Sharygina and H. Veith, editors, *Computer Aided Verification - 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings*, volume 8044 of *Lecture Notes in Computer Science*, pages 511–526. Springer, 2013.
17. A. Chakarov, Y. Voronin, and S. Sankaranarayanan. Deductive proofs of almost sure persistence and recurrence properties. In M. Chechik and J. Raskin, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 22nd International Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*, volume 9636 of *Lecture Notes in Computer Science*, pages 260–279. Springer, 2016.
18. K. Chatterjee, H. Fu, and A. K. Goharshady. Termination analysis of probabilistic programs through positivstellensatz's. In S. Chaudhuri and A. Farzan, editors, *Computer Aided Verification - 28th International Conference, CAV 2016, Toronto, ON, Canada, July 17-23, 2016, Proceedings, Part I*, volume 9779 of *Lecture Notes in Computer Science*, pages 3–22. Springer, 2016.
19. K. Chatterjee, H. Fu, and A. K. Goharshady. Termination analysis of probabilistic programs through positivstellensatz's. *CoRR*, abs/1604.07169, 2016.
20. K. Chatterjee, H. Fu, P. Novotný, and R. Hasheminezhad. Algorithmic analysis of qualitative and quantitative termination problems for affine probabilistic programs. In R. Bodík and R. Majumdar, editors, *Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2016, St. Petersburg, FL, USA, January 20 - 22, 2016*, pages 327–342. ACM, 2016.
21. K. Chatterjee, H. Fu, P. Novotný, and R. Hasheminezhad. Algorithmic analysis of qualitative and quantitative termination problems for affine probabilistic programs. *ACM Trans. Program. Lang. Syst.*, 40(2):7:1–7:45, 2018.
22. K. Chatterjee, P. Novotný, and D. Zikelic. Stochastic invariants for probabilistic termination. In G. Castagna and A. D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, pages 145–160. ACM, 2017.
23. M. Chen, J. Katoen, L. Klinkenberg, and T. Winkler. Does a program yield the right distribution? - verifying probabilistic programs via generating functions. In S. Shoham and Y. Vizel, editors, *Computer Aided Verification - 34th International Conference, CAV 2022, Haifa, Israel, August 7-10, 2022, Proceedings, Part I*, volume 13371 of *Lecture Notes in Computer Science*, pages 79–101. Springer, 2022.
24. E. W. Dijkstra. *A Discipline of Programming*. Prentice-Hall, 1976.
25. R. Durrett. *Probability: theory and examples*, volume 49. Cambridge University Press, 2019.
26. S. Feng, M. Chen, H. Su, B. L. Kaminski, J. Katoen, and N. Zhan. Lower bounds for possibly divergent probabilistic programs. *Proc. ACM Program. Lang.*, 7(OOPSLA1):696–726, 2023.
27. N. Foster, D. Kozen, K. Mamouras, M. Reitblatt, and A. Silva. Probabilistic netkat. In P. Thiemann, editor, *Programming Languages and Systems - 25th European Symposium on Programming, ESOP 2016, Held as Part of the European*

Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings, volume 9632 of *Lecture Notes in Computer Science*, pages 282–309. Springer, 2016.
28. H. Fu and K. Chatterjee. Termination of nondeterministic probabilistic programs. In C. Enea and R. Piskac, editors, *Verification, Model Checking, and Abstract Interpretation - 20th International Conference, VMCAI 2019, Cascais, Portugal, January 13-15, 2019, Proceedings*, volume 11388 of *Lecture Notes in Computer Science*, pages 468–490. Springer, 2019.
29. A. D. Gordon, T. A. Henzinger, A. V. Nori, and S. K. Rajamani. Probabilistic programming. In J. D. Herbsleb and M. B. Dwyer, editors, *Proceedings of the on Future of Software Engineering, FOSE 2014, Hyderabad, India, May 31 - June 7, 2014*, pages 167–181. ACM, 2014.
30. M. Hark, B. L. Kaminski, J. Giesl, and J. Katoen. Aiming low is harder: induction for lower bounds in probabilistic program verification. *Proc. ACM Program. Lang.*, 4(POPL):37:1–37:28, 2020.
31. A. Hartmanns and B. L. Kaminski. Optimistic value iteration. In S. K. Lahiri and C. Wang, editors, *Computer Aided Verification - 32nd International Conference, CAV 2020, Los Angeles, CA, USA, July 21-24, 2020, Proceedings, Part II*, volume 12225 of *Lecture Notes in Computer Science*, pages 488–511. Springer, 2020.
32. B. L. Kaminski. *Advanced weakest precondition calculi for probabilistic programs*. PhD thesis, RWTH Aachen University, Germany, 2019.
33. B. L. Kaminski and J. Katoen. A weakest pre-expectation semantics for mixed-sign expectations. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12. IEEE Computer Society, 2017.
34. B. L. Kaminski, J. Katoen, C. Matheja, and F. Olmedo. Weakest precondition reasoning for expected run-times of probabilistic programs. In P. Thiemann, editor, *Programming Languages and Systems - 25th European Symposium on Programming, ESOP 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*, volume 9632 of *Lecture Notes in Computer Science*, pages 364–389. Springer, 2016.
35. B. L. Kaminski, J. Katoen, C. Matheja, and F. Olmedo. Weakest precondition reasoning for expected runtimes of randomized algorithms. *J. ACM*, 65(5):30:1–30:68, 2018.
36. A. Karimi, M. Moosbrugger, M. Stankovic, L. Kovács, E. Bartocci, and E. Bura. Distribution estimation for probabilistic loops. In E. Ábrahám and M. Paolieri, editors, *Quantitative Evaluation of Systems - 19th International Conference, QEST 2022, Warsaw, Poland, September 12-16, 2022, Proceedings*, volume 13479 of *Lecture Notes in Computer Science*, pages 26–42. Springer, 2022.
37. L. Klinkenberg, K. Batz, B. L. Kaminski, J. Katoen, J. Moerman, and T. Winkler. Generating functions for probabilistic programs. In M. Fernández, editor, *Logic-Based Program Synthesis and Transformation - 30th International Symposium, LOPSTR 2020, Bologna, Italy, September 7-9, 2020, Proceedings*, volume 12561 of *Lecture Notes in Computer Science*, pages 231–248. Springer, 2020.
38. A. Kofnov, M. Moosbrugger, M. Stankovic, E. Bartocci, and E. Bura. Moment-based invariants for probabilistic loops with non-polynomial assignments. In E. Ábrahám and M. Paolieri, editors, *Quantitative Evaluation of Systems - 19th International Conference, QEST 2022, Warsaw, Poland, September 12-16, 2022, Proceedings*, volume 13479 of *Lecture Notes in Computer Science*, pages 3–25. Springer, 2022.

39. D. Kozen. Semantics of probabilistic programs. *J. Comput. Syst. Sci.*, 22(3):328–350, 1981.
40. S. Kura, N. Urabe, and I. Hasuo. Tail probabilities for randomized program runtimes via martingales for higher moments. In T. Vojnar and L. Zhang, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 25th International Conference, TACAS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings, Part II*, volume 11428 of *Lecture Notes in Computer Science*, pages 135–153. Springer, 2019.
41. M. Z. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In G. Gopalakrishnan and S. Qadeer, editors, *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings*, volume 6806 of *Lecture Notes in Computer Science*, pages 585–591. Springer, 2011.
42. C. Mak, C. L. Ong, H. Paquet, and D. Wagner. Densities of almost surely terminating probabilistic programs are differentiable almost everywhere. In N. Yoshida, editor, *Programming Languages and Systems - 30th European Symposium on Programming, ESOP 2021, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 - April 1, 2021, Proceedings*, volume 12648 of *Lecture Notes in Computer Science*, pages 432–461. Springer, 2021.
43. C. Mak, F. Zaiser, and L. Ong. Nonparametric hamiltonian monte carlo. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 7336–7347. PMLR, 18–24 Jul 2021.
44. A. McIver and C. Morgan. *Abstraction, Refinement and Proof for Probabilistic Systems*. Monographs in Computer Science. Springer, 2005.
45. M. Moosbrugger, E. Bartocci, J. Katoen, and L. Kovács. Automated termination analysis of polynomial probabilistic programs. In N. Yoshida, editor, *Programming Languages and Systems - 30th European Symposium on Programming, ESOP 2021, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 - April 1, 2021, Proceedings*, volume 12648 of *Lecture Notes in Computer Science*, pages 491–518. Springer, 2021.
46. M. Moosbrugger, E. Bartocci, J. Katoen, and L. Kovács. The probabilistic termination tool amber. In M. Huisman, C. S. Pasareanu, and N. Zhan, editors, *Formal Methods - 24th International Symposium, FM 2021, Virtual Event, November 20-26, 2021, Proceedings*, volume 13047 of *Lecture Notes in Computer Science*, pages 667–675. Springer, 2021.
47. M. Moosbrugger, M. Stankovic, E. Bartocci, and L. Kovács. This is the moment for probabilistic loops. *Proc. ACM Program. Lang.*, 6(OOPSLA2):1497–1525, 2022.
48. V. C. Ngo, Q. Carbonneaux, and J. Hoffmann. Bounded expectations: resource analysis for probabilistic programs. In J. S. Foster and D. Grossman, editors, *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2018, Philadelphia, PA, USA, June 18-22, 2018*, pages 496–512. ACM, 2018.
49. M. Putinar. Positive polynomials on compact semi-algebraic sets. *Indiana University Mathematics Journal*, 42(3):969–984, 1993.
50. T. Quatmann and J. Katoen. Sound value iteration. In H. Chockler and G. Weissenbacher, editors, *Computer Aided Verification - 30th International Conference,*

CAV 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings, Part I, volume 10981 of *Lecture Notes in Computer Science*, pages 643–661. Springer, 2018.
51. T. Rainforth. *Automating inference, learning, and design using probabilistic programming*. PhD thesis, University of Oxford, 2017.
52. A. Ścibior, Z. Ghahramani, and A. D. Gordon. Practical probabilistic programming with monads. In B. Lippmeier, editor, *Proceedings of the 8th ACM SIGPLAN Symposium on Haskell, Haskell 2015, Vancouver, BC, Canada, September 3-4, 2015*, pages 165–176. ACM, 2015.
53. M. Stankovic, E. Bartocci, and L. Kovács. Moment-based analysis of bayesian network properties. *Theor. Comput. Sci.*, 903:113–133, 2022.
54. T. Takisaka, Y. Oyabu, N. Urabe, and I. Hasuo. Ranking and repulsing supermartingales for reachability in randomized programs. *ACM Trans. Program. Lang. Syst.*, 43(2):5:1–5:46, 2021.
55. J. van de Meent, B. Paige, H. Yang, and F. Wood. An introduction to probabilistic programming. *CoRR*, abs/1809.10756, 2018.
56. D. Wang, J. Hoffmann, and T. W. Reps. Central moment analysis for cost accumulators in probabilistic programs. In S. N. Freund and E. Yahav, editors, *PLDI '21: 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation, Virtual Event, Canada, June 20-25, 20211*, pages 559–573. ACM, 2021.
57. J. Wang, Y. Sun, H. Fu, K. Chatterjee, and A. K. Goharshady. Quantitative analysis of assertion violations in probabilistic programs. In S. N. Freund and E. Yahav, editors, *PLDI '21: 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation, Virtual Event, Canada, June 20-25, 20211*, pages 1171–1186. ACM, 2021.
58. P. Wang, H. Fu, A. K. Goharshady, K. Chatterjee, X. Qin, and W. Shi. Cost analysis of nondeterministic probabilistic programs. In K. S. McKinley and K. Fisher, editors, *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2019, Phoenix, AZ, USA, June 22-26, 2019*, pages 204–220. ACM, 2019.
59. P. Wang, H. Fu, T. Yang, G. Li, and L. Ong. Template-based static posterior inference for bayesian probabilistic programming. *CoRR*, abs/2307.13160, 2023.
60. D. Williams. *Probability with martingales*. Cambridge University Press, 1991.

A Additional remarks on value function

In Def. 2, the value function $V(s)$ for Markov chains is defined by

$$V(s) = E_s\left[\sum_{i=0}^{T_H-1} f(X_i) + \mathbf{1}_{T_H<\infty} \cdot g(X_{T_H})\right] \tag{19}$$

which implicitly assume that $\sum_{i=0}^{T_H-1} f(X_i)$ converges [13] (including converging to infinity) almost surely. If it is not the case, we may define $V(s)$ as an interval,

[13] Note $\sum_{i=0}^{T_H-1} f(X_i)$ is an infinite sum iff $T_H = \infty$, thus if $T_H < \infty$ almost surely, $\sum_{i=0}^{T_H-1} f(X_i)$ also converges almost surely.

which contains all possible expected cost during the execution of the system. Formally, let

$$L(s) = E_s\left[\liminf_{n\to\infty} \sum_{i=0}^{n\wedge T_H-1} f(X_i) + \mathbf{1}_{T_H<\infty}\cdot g(X_{T_H})\right] \tag{20}$$

$$U(s) = E_s\left[\limsup_{n\to\infty} \sum_{i=0}^{n\wedge T_H-1} f(X_i) + \mathbf{1}_{T_H<\infty}\cdot g(X_{T_H})\right] \tag{21}$$

we then define $V(s) = [L, U]$. If $V(s)$ is an interval, the equation system in Thm. 1 is not applicable, but the sufficient conditions for finding upper and lower bounds proposed in Thm. 2 and Thm. 3 are still valid in the sense that $u(s) \geq V(s) = [L(s), U(s)]$ means $u(s) \geq U(s)$, and $u(s) \leq V(s) = [L(s), U(s)]$ means $u(s) \leq L(s)$.

For Markov chain, the value function $V(s)$ in Def. 2 is defined by

$$V(s) = E_s\left[\sum_{i=0}^{T_H-1} f(X_i) + \mathbf{1}_{T_H<\infty}\cdot g(X_{T_H})\right] \tag{22}$$

which also implicitly assume $\sum_{i=0}^{T_H-1} f(X_i)$ converges almost surely. If it is not the case, we may define $V(s)$ as an interval, formally, let

$$L(s) = E_s\left[\liminf_{n\to\infty} \sum_{i=0}^{n\wedge T_H-1} f(X_i) + \mathbf{1}_{T_H<\infty}\cdot g(X_{T_H})\right]$$

$$U(s) = E_s\left[\limsup_{n\to\infty} \sum_{i=0}^{n\wedge T_H-1} f(X_i) + \mathbf{1}_{T_H<\infty}\cdot g(X_{T_H})\right]$$

then $V(s)$ is defined by

$$V(s) = [L(s), U(s)].$$

In this case, the equation system for $V(s)$ proposed in Thm. 1 is not applicable, but the conditions for finding upper and lower bounds proposed in Thm. 2 and Thm. 3 are still valid in the sense that

$$u(s) \geq V(s) = [L(s), U(s)]$$

means $u(s) \geq U(s)$, and

$$u(s) \leq V(s) = [L(s), U(s)]$$

means $u(s) \leq L(s)$.