# Lower Bounds for Possibly Divergent Probabilistic Programs

SHENGHUA FENG, SKLCS, Institute of Software, CAS, University of Chinese Academy of Sciences, China
MINGSHUAI CHEN[†], Zhejiang University, China
HAN SU, SKLCS, Institute of Software, CAS, University of Chinese Academy of Sciences, China
BENJAMIN LUCIEN KAMINSKI, Saarland University, Saarland Informatics Campus, Germany and University College London, United Kingdom
JOOST-PIETER KATOEN, RWTH Aachen University, Germany
NAIJUN ZHAN, SKLCS, Institute of Software, CAS, University of Chinese Academy of Sciences, China

We present a new proof rule for verifying lower bounds on quantities of probabilistic programs. Our proof rule is not confined to almost-surely terminating programs – as is the case for existing rules – and can be used to establish non-trivial lower bounds on, e.g., termination probabilities and expected values, for possibly *divergent* probabilistic loops, e.g., the well-known three-dimensional random walk on a lattice.

CCS Concepts: • **Theory of computation → Program reasoning**; • **Mathematics of computing → Probabilistic algorithms**; **Stochastic processes**.

Additional Key Words and Phrases: probabilistic programs, quantitative verification, weakest preexpectations, lower bounds, almost-sure termination, uniform integrability

**ACM Reference Format:**
Shenghua Feng, Mingshuai Chen, Han Su, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Naijun Zhan. 2023. Lower Bounds for Possibly Divergent Probabilistic Programs. *Proc. ACM Program. Lang.* 7, OOPSLA1, Article 99 (April 2023), 31 pages. https://doi.org/10.1145/3586051



## 1 INTRODUCTION

Probabilistic programs [Gordon et al. 2014; Kozen 1981; Saheb-Djahromi 1978; van de Meent et al. 2018] extend deterministic programs with stochastic behaviors, e.g., random sampling, probabilistic choice, and conditioning (via posterior observations). Probabilistic programs have witnessed numerous applications in various domains: They steer autonomous robots and self-driving cars [Evans et al. 2017; Shamsi et al. 2020], are key to describe security [Barthe et al. 2013] and quantum [Ying 2011] mechanisms, intrinsically code up randomized algorithms for solving NP-hard or even deterministically unsolvable problems (in, e.g., distributed computing [Aspnes and Herlihy 1990; Schneider 1993]), and are at the heart of modern machine learning and approximate computing [Carbin et al. 2016]. See [Barthe et al. 2020] for recent advancements in probabilistic programming.

[†]The corresponding author

Authors' addresses: Shenghua Feng, SKLCS, Institute of Software, CAS, University of Chinese Academy of Sciences, Beijing, China, fengsh@ios.ac.cn; Mingshuai Chen, Zhejiang University, Hangzhou, China, m.chen@zju.edu.cn; Han Su, SKLCS, Institute of Software, CAS, University of Chinese Academy of Sciences, Beijing, China, suhan@ios.ac.cn; Benjamin Lucien Kaminski, b.kaminski@ucl.ac.uk, Saarland University, Saarland Informatics Campus, Saarbrücken, Germany and University College London, London, United Kingdom; Joost-Pieter Katoen, RWTH Aachen University, Aachen, Germany, katoen@cs.rwth-aachen.de; Naijun Zhan, SKLCS, Institute of Software, CAS, University of Chinese Academy of Sciences, Beijing, China, znj@ios.ac.cn.


2475-1421/2023/4-ART99
https://doi.org/10.1145/3586051

Probabilistic programs, though typically relatively small in size, are hard to grasp: The crux of probabilistic programming is to treat normal-looking programs as if they were *probability distributions* [Hicks 2014; Saheb-Djahromi 1978]. Such a lift from deterministic program states to possibly infinite-support distributions (over states) renders the verification problem of probabilistic programs notoriously hard [Kaminski et al. 2019]. In particular, given a random variable $f$ (mapping program states to numbers), a key verification task is to reason about the *expected value* of $f$ after termination of a program $C$ on input $\sigma$. If $f$ is the indicator function of an event $A$, then this expected value is the *probability* that $A$ occurs upon termination of $C$. In case of a potentially *unbounded loopy program* $C$, the expected value of $f$ is often characterized as the *least fixed point* of some monotonic operator capturing the semantics of $C$ w.r.t. $f$. Computing the exact expected value of $f$ hence amounts to inferring the least fixed point which is in general highly intractable.

As a consequence, existing verification techniques for reasoning about probabilistic loops are mostly concerned with proving *upper* and/or *lower bounds* on expected values, i.e., on least fixed points. Verifying lower bounds is notably essential for establishing *total correctness* of probabilistic programs [Katoen et al. 2015; McIver and Morgan 2001] and for assessing the quality and tightness of upper bounds. For verifying a candidate upper bound $u$, the well-known principle of *Park induction* [Kozen 1985; Park 1969], or more generally, *$\kappa$-induction* [Batz et al. 2021a], suffices by "pushing $u$ through the loop semantics" *once*. Whereas for lower bounds on least fixed point, a "dual" version of Park induction is *unsound* (see Sect. 4.3).

Existing (sound) lower induction rules for probabilistic programs are confined to either (i) *bounded* random variables with a priori knowledge on the *termination probability* of the program [McIver and Morgan 2005]; or (ii) (universally) *almost-surely terminating* (AST) programs (i.e., programs that terminate with probability 1 on all inputs) and *uniformly integrable* random variables – a notion from stochastic processes, which requires reasoning about looping times and/or bounds on random variables [Hark et al. 2020]. In contrast to Park induction for upper bounds, applying these lower induction rules requires heavy proof efforts in, e.g., looking for supermartingales [Chatterjee et al. 2020] witnessing AST, checking uniform integrability, and inferring termination probabilities. *In particular, none of these rules is capable of inferring lower bounds on termination probabilities strictly less than 1*, i.e., for non-AST (aka, *divergent*) programs. Consider, e.g., the following probabilistic loop $C_{\text{3dsrw}}$ modelling the well-known *three-dimensional (3-D) random walk* on the lattice over $\mathbb{Z}^3$.[1]

$$C_{\text{3dsrw}}\colon\quad \texttt{while}\,(\,x \neq 0 \vee y \neq 0 \vee z \neq 0\,)\,\{\\ x \coloneqq x-1 \;\oplus\; x \coloneqq x+1 \;\oplus\; y \coloneqq y-1 \;\oplus\; y \coloneqq y+1 \;\oplus\; z \coloneqq z-1 \;\oplus\; z \coloneqq z+1\,\}\,.$$

The random nature underneath $C_{\text{3dsrw}}$ is fundamentally *different* from its 1- and 2-D counterparts: Pólya [1921] proved that the probability $\mathcal{P}$ that such a random walk returns to the origin at $(0,0,0)$ is strictly less than 1, indicating that $C_{\text{3dsrw}}$ does *not* terminate almost-surely. More precisely, the termination probability of $C_{\text{3dsrw}}$ starting from any neighbor location of the origin is

$$\mathcal{P} \;=\; 1 - \left(\frac{3}{(2\pi)^3}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\frac{\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z}{3-\cos x-\cos y-\cos z}\right)^{-1} \;=\; 0.3405373296\ldots \qquad (\dagger)$$

Existing verification techniques cannot tackle $C_{\text{3dsrw}}$ due to its complex nature of divergence.

In this paper, we present a new proof rule, termed the *guard-strengthening rule* for verifying lower bounds on the expected value of a potentially *unbounded* random variable $f$ for a possibly *divergent* probabilistic loop $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$. *Our proof rule employs reduction:* Suppose we aim to certify $l$ as a lower bound on the expected value of $f$ after termination of $C_{\text{loop}}$. We first forge a new loop $C'_{\text{loop}} = \texttt{while}\,(\,\varphi'\,)\,\{\,C\,\}$ out of $C_{\text{loop}}$ by *strengthening* its loop guard $\varphi$ to $\varphi'$, yielding a

[1] The iterated symbol $\oplus$ is shorthand for discrete uniform choice (in this case, with probability 1/6 each).

reduced problem where (i) the modified loop $C'_{\text{loop}}$ features a *stronger* termination property (e.g., provably AST), and (ii) both the uniform integrability of $l$ and the boundedness conditions are *easier* to verify. Our proof rule then asserts – by exploiting the "difference" between $C'_{\text{loop}}$ and $C_{\text{loop}}$ w.r.t. $f$ in terms of the *weakest preexpectation calculus* [Kozen 1985; McIver and Morgan 2005] – that *a lower bound $l$ for $C'_{\text{loop}}$ (w.r.t. a restricted form of $f$) also suffices as a lower bound for $C_{\text{loop}}$ (w.r.t. $f$)*. The former, due to (i) and (ii) by guard strengthening, can often be established by applying the aforementioned lower induction rules or – if $C'_{\text{loop}}$ has a *finite* state space – probabilistic model checking [Baier and Katoen 2008; Katoen 2016; Kwiatkowska 2003]. In this case, our proof rule can be (partially) *automated* to derive *increasingly tighter* lower bounds – as $\varphi'$ "approaches" $\varphi$ – on, e.g., the termination probability $\mathcal{P}$ of the 3-D random walk in (†), see details in Exmp. 24.

The main results of this paper are the following:

(a) We present a new proof rule via *guard strengthening* for verifying lower bounds on expected values of probabilistic programs. To the best of our knowledge, this is the first lower bound rule that admits *divergent* probabilistic loops with *unbounded* expected values.
(b) We show that the modified loops with strengthened guards feature *easily provable* almost-sure termination and uniform integrability. This eases and enlarges the use of existing proof rules for lower bounds; Moreover, we propose a novel sufficient criterion for proving uniform integrability which recognizes cases that are *out-of-reach* by existing sufficient conditions based on the optional stopping theorem [Hark et al. 2020].
(c) We show that the approximation error incurred by our guard-strengthening technique can be *arbitrarily small* thereby yielding *tight* lower bounds.
(d) We identify scenarios where our proof rule facilitates inferring quantitative properties of *infinite*-state probabilistic programs by model checking *finite*-state probabilistic models.

We demonstrate the effectiveness of our proof rule on a collection of examples, including the 3-D random walk and a real-world randomized networking protocol.

Additional background materials, elaborated proofs, and details on the examples can be found in the appendix of the full version of this paper [Feng et al. 2023].

## 2 OVERVIEW OF OUR APPROACH

In a nutshell, our idea is to transform a given potentially non-AST loop $C_{\text{loop}}$ into a provably AST loop $C'_{\text{loop}}$ and then certify lower bounds for $C'_{\text{loop}}$. Our transformation is performed in a way s.t. the expected outcome of $C'_{\text{loop}}$ is guaranteed to be a lower bound on the expected outcome of $C_{\text{loop}}$. Thus, as encoded in our proof rule, a lower bound for $C'_{\text{loop}}$ suffices as a lower bound for $C_{\text{loop}}$.

Let us demonstrate our approach by analyzing one of the most basic expected outcomes of a probabilistic loop: termination. Consider the 1-D random walk $C_{\text{1dbrw}}$ on $\mathbb{Z}$ (shown below on the left) with *biased* probability $1/3$ moving to the left and probability $2/3$ moving to the right. Due to its biased nature, this loop does not terminate almost-surely and *none of the existing proof rules*[2] *suffices* to establish non-trivial lower bounds on its termination probability; see details in Exmp. 16.

$C_{\text{1dbrw}}$: `while` $(0 < n)\,\{$ $n \coloneqq n-1\ [1/3]\ n \coloneqq n+1$ $\}$

$C^{M}_{\text{1dbrw}}$: `while` $(0 < n < M)\,\{$ $n \coloneqq n-1\ [1/3]\ n \coloneqq n+1$ $\}$

Above right, we see the modified version of this loop, $C^{M}_{\text{1dbrw}}$, which is obtained from $C_{\text{1dbrw}}$ by introducing an artificial upper bound $M \in \mathbb{N}$ on $n$ in the loop guard. This modified loop *does* terminate almost-surely and can moreover visit only finitely many different states.

[2]Referring to syntactic proof rules in the expectation-based program logic [Kozen 1985; McIver and Morgan 2005].

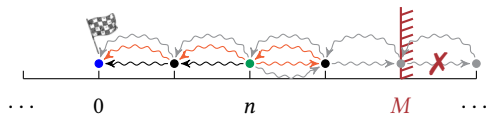


Fig. 1. Guard-strengthening effect: In $C_{1\text{dbrw}}$, all the three program traces (distinguished by colors) initiating from • *terminate* at •. In $C^{M}_{1\text{dbrw}}$, however, the gray trace crossing the "barrier" $M$ is no longer possible.

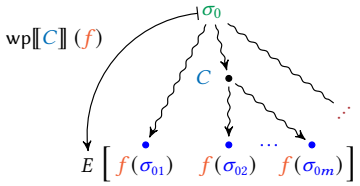


Fig. 2. Illustration of wp: $\text{wp}[\![C]\!]\,(f)\,(\sigma_0)$ determines the expected value of $f$ evaluated in the final states • reached after termination of $C$ on input $\sigma_0$; $\cdot^{\cdot^{\cdot}}$ indicates nonterminating (aka, divergent) path of $C$.

$C^{M}_{1\text{dbrw}}$ terminates by "hitting" $n \leq 0$ or $n \geq M$. The key observation is that *the probability of* $C^{M}_{1\text{dbrw}}$ *terminating at* $n \leq 0$ *is smaller than the termination probability of* $C_{1\text{dbrw}}$, since some terminating program traces of $C_{1\text{dbrw}}$ – contributing to its termination probability – are no longer possible in $C^{M}_{1\text{dbrw}}$ due to the artificial "barrier" $M$; see Fig. 1 for an illustration. Meanwhile, underapproximating the probability that $C^{M}_{1\text{dbrw}}$ terminates at $n \leq 0$ – thereby yielding a lower bound on the termination probability of $C_{1\text{dbrw}}$ – can be addressed by existing lower induction rules. In fact, since $C^{M}_{1\text{dbrw}}$ has a finite state space for any fixed $M \in \mathbb{N}$, its *exact* termination probability at $n \leq 0$ can be obtained by probabilistic model checking. Moreover, if we push the "barrier" further to the right by increasing $M$, then we obtain *increasingly tighter* lower bounds. See Exmp. 16 for a detailed analysis.

## 3 WEAKEST PREEXPECTATION REASONING

### 3.1 The Probabilistic Guarded Command Language

We consider probabilistic programs described by the simple yet Turing-complete, imperative *probabilistic guarded command language* (pGCL) [McIver and Morgan 2005] which augments Dijkstra's GCL [Dijkstra 1976] with probabilistic choices and random assignments.

*Syntax.* The syntax of a pGCL program $C$ adheres to the grammar

$$\begin{aligned} C \quad ::= \quad & \texttt{skip} \mid x \coloneqq e \mid x :\approx \mu \mid C \,;\, C \mid \{C\}\,[p]\,\{C\} \mid \\ & \texttt{if}\,(\varphi)\,\{C\}\,\texttt{else}\,\{C\} \mid \texttt{while}\,(\varphi)\,\{C\} \end{aligned}$$

where $x$ is a program variable taken from a countable set Vars, $e$ is an arithmetic expression over program variables, $\varphi$ is a quantifier-free first-order predicate over program variables, and $\mu$ denotes a discrete or continuous distribution. We do not specify the syntax of expressions $e$ and predicates $\varphi$ – they can be arbitrary as long as the corresponding evaluation functions are measurable, as is in [Szymczak and Katoen 2019]. The semantics of most program constructs – including skip, (deterministic) assignments, sequential composition, conditional, and (nested) loops – is standard. The *probabilistic choice* $\{C_1\}\,[p]\,\{C_2\}$ flips a coin with bias $p \in [0,1]$ and executes $C_1$ in case the coin yields heads, and $C_2$ otherwise. The *random assignment* $x :\approx \mu$ draws a sample from the distribution $\mu$ – either discrete or continuous – and assigns it to the program variable $x$.

*Program States.* A *program state* $\sigma$ maps every variable in Vars to its value, i.e., a real number in $\mathbb{R}$. We denote the (possibly uncountable) set of program states by $\Sigma \triangleq \{\sigma \mid \sigma\colon \texttt{Vars} \to \mathbb{R}\}$. The evaluation of expressions $e$ and guards $\varphi$ under a state $\sigma$, denoted by $e(\sigma)$ and $\varphi(\sigma)$ respectively, is standard. For instance, the evaluation of arithmetic addition is

$$(e_1 + e_2)(\sigma) \;\triangleq\; e_1(\sigma) + e_2(\sigma) \;=\; e_1\,[x/\sigma(x)] + e_2\,[x/\sigma(x)] \quad \text{for all } \; x \in \texttt{Vars}$$

where $e\,[x/\sigma(x)]$ denotes the substitution of variable $x$ by its value $\sigma(x)$ in $e$.

*Predicates.* We interpret guards in pGCL programs as predicates. A *predicate* $\varphi$ represents a subset of program states $\Sigma$. We write $\sigma \models \varphi$, reading "$\sigma$ satisfies $\varphi$", to indicate that state $\sigma$ is in the set represented by predicate $\varphi$, i.e., $\varphi(\sigma) = \texttt{true}$; and $\sigma \not\models \varphi$ otherwise. We write $\varphi_1 \implies \varphi_2$, reading "$\varphi_1$ strengthens $\varphi_2$", to indicate that under every state $\sigma \in \Sigma$, if $\varphi_1(\sigma) = \texttt{true}$ then $\varphi_2(\sigma) = \texttt{true}$.

### 3.2 The Weakest Preexpectation Calculus

To reason about quantitative properties of probabilistic programs, in particular, to lower-bound expected values of certain probabilistic quantities, we view pGCL programs as *expectation transformers* [Kaminski 2019; Kozen 1985; McIver and Morgan 2005] – a quantitative extension of the predicate-transformer calculus for non-probabilistic programs of Dijkstra [1975, 1976].

An expectation transformer acts on real-valued functions called *expectations*, which map program states to non-negative reals (extended by infinity)[3]. Note the distinction between expectations and expected values: instead of an expected value, one can think of an expectation as a *random variable*.

**Definition 1 (Expectations [Kaminski 2019]).** *The set of* expectations*, denoted by* $\mathbb{E}$*, is defined as*

$$\mathbb{E} \;\triangleq\; \left\{ f \mid f \colon \Sigma \to \mathbb{R}^{\infty}_{\geq 0} \right\} .$$

*An expectation* $f \in \mathbb{E}$ *is* finite*, written as* $f \ll \infty$*, if* $f(\sigma) < \infty$ *for all* $\sigma \in \Sigma$*;* $f \in \mathbb{E}$ *is* bounded*, if there exists* $b \in \mathbb{R}_{\geq 0}$ *such that* $f(\sigma) \leq b$ *for all* $\sigma \in \Sigma$*.*

For simplicity, a constant expectation $\lambda\sigma.\ r$ which evaluates to $r \in \mathbb{R}^{\infty}_{\geq 0}$ for every state is denoted by $r$. Similarly, given an arithmetic expression $e$, we denote by $e$ the expectation $\lambda\sigma.\ e(\sigma)$.

A partial order $\preceq$ on $\mathbb{E}$ is obtained by point-wise lifting the canonical ordering $\leq$ on $\mathbb{R}^{\infty}_{\geq 0}$, i.e.,

$$f_1 \;\preceq\; f_2 \quad \text{iff} \quad \forall \sigma \in \Sigma \colon\; f_1(\sigma) \;\leq\; f_2(\sigma) \,.$$

$(\mathbb{E}, \preceq)$ forms a complete lattice with least element 0 and greatest element $\infty$.

A pGCL program $C$ is interpreted as an expectation transformer which pushes a *postexpectation* $f \in \mathbb{E}$ (evaluated in the final states) backward through $C$ and gives a *preexpectation* $g \in \mathbb{E}$ (evaluated in the initial states). In particular, as illustrated in Fig. 2, the *weakest preexpectation of* $C$ *w.r.t.* $f$ is a *function* $g \colon \Sigma \to \mathbb{R}^{\infty}_{\geq 0}$ mapping each *initial state* $\sigma_0$ of $C$ to the corresponding expected value of $f$ evaluated in the final states reached after termination of $C$ on input $\sigma_0$:

**Definition 2 (Weakest Preexpectations [Kaminski 2019; Kozen 1985; McIver and Morgan 2005]).** *Given probabilistic program* $C$ *and initial state* $\sigma_0 \in \Sigma$*. Let* ${}^{\sigma_0}\mu_C$ *be the (sub)probability measure*[4] *over final states reached after termination of* $C$ *on input* $\sigma_0$*. Given postexpectation* $f \in \mathbb{E}$ *which is measurable w.r.t.* ${}^{\sigma_0}\mu_C$*, the* weakest preexpectation of $C$ w.r.t. $f$ *maps any initial state* $\sigma_0$ *to the expected value of* $f$ *evaluated in the final states reached after termination of* $C$ *on* $\sigma_0$*, i.e.,*[5]

$$\mathrm{wp}[\![C]\!]\,(f)\,(\sigma_0) \;\triangleq\; \int_{\Sigma} f \;\mathrm{d}\left({}^{\sigma_0}\mu_C\right) \,.$$

It is known that (i) for every measurable $f \in \mathbb{E}$, $\mathrm{wp}[\![C]\!]\,(f)$ is measurable (cf. [Szymczak and Katoen 2019, Lem. 3.2]), and (ii) the set of measurable expectations also forms a complete lattice under the partial order $\preceq$ (cf. [Szymczak and Katoen 2019, Lem. 2]). Hence, for simplicity, we abuse the notation $\mathbb{E}$ to stand for the set of *measurable* expectations throughout the rest of the paper.

[3] For simplicity, we consider the standard case of *non-negative* expectations. An arithmetic expression is thus a well-defined expectation if and only if it takes non-negative values over *all reachable* program states. See [Kaminski and Katoen 2017] for more involved techniques addressing *mixed-sign* expectations mapping to the *full* extended reals.

[4] ${}^{\sigma_0}\mu_C(\sigma) \in [0, 1]$ is the probability that, on input $\sigma_0$, $C$ terminates in the final state $\sigma$. Note that ${}^{\sigma_0}\mu_C(\Sigma) \leq 1$, where the "missing" probability mass is the probability of *nontermination* of $C$ on $\sigma_0$. A formal definition of ${}^{\sigma_0}\mu_C$ requires an (operational) semantic model of pGCL, which is out of our scope; we refer interested readers to [Dahlqvist et al. 2020].

[5] In case of a countable state space $\Sigma$, the integral can be written as a countable sum $\sum_{\sigma \in \Sigma} {}^{\sigma_0}\mu_C(\sigma) \cdot f(\sigma)$.

Table 1. Rules for the wp-transformer. $[\varphi]$ denotes the *Iverson-bracket* of $\varphi$, i.e., $[\varphi]\,(\sigma)$ evaluates to 1 if $\sigma \models \varphi$ and to 0 otherwise. For any variable $x \in \texttt{Vars}$ and any expression $e$, $f\,[x/e]$ denotes the expectation with $f\,[x/e]\,(\sigma) = f(\sigma\,[x/e])$ for any $\sigma \in \Sigma$, where $\sigma\,[x/e]\,(x) = e(\sigma)$ and $\sigma\,[x/e]\,(y) = \sigma(y)$ for all $y \in \texttt{Vars}\setminus\{x\}$.

| $C$ | $\mathbf{wp}\,[\![C]\!]\,(f)$ |
|---|---|
| `skip` | $f$ |
| $x \coloneqq e$ | $f\,[x/e]$ |
| $x :\approx \mu$ | $\int_{\mathbb{R}} f\,[x/v]\ \mathrm{d}\,\mu(v)$ |
| $C_1 \fatsemi C_2$ | $\mathsf{wp}[\![C_1]\!]\,(\mathsf{wp}[\![C_2]\!]\,(f))$ |
| $\{\,C_1\,\}\ [\,p\,]\ \{\,C_2\,\}$ | $p \cdot \mathsf{wp}[\![C_1]\!]\,(f) + (1-p) \cdot \mathsf{wp}[\![C_2]\!]\,(f)$ |
| `if` $(\,\varphi\,)\ \{\,C_1\,\}$ `else` $\{\,C_2\,\}$ | $[\varphi] \cdot \mathsf{wp}[\![C_1]\!]\,(f) + [\neg\varphi] \cdot \mathsf{wp}[\![C_2]\!]\,(f)$ |
| `while` $(\,\varphi\,)\ \{\,C'\,\}$ | $\mathsf{lfp}\ {}^{\mathsf{wp}}_{\langle\varphi,C'\rangle}\Phi_f$ |
| ${}^{\mathsf{wp}}_{\langle\varphi,C'\rangle}\Phi_f \colon \mathbb{E} \to \mathbb{E}, \quad h \mapsto [\neg\varphi] \cdot f + [\varphi] \cdot \mathsf{wp}[\![C']\!]\,(h)$ | characteristic function |

Weakest preexpectations can be determined in a *backward*, *compositional* manner; see [Feng et al. 2023, Appx. B]. In fact, the wp-*transformer* can be codified by structural induction:

**Theorem 3 (wp-Transformer** [McIver and Morgan 2005]**).** *Let* pGCL *be the set of programs in the probabilistic guarded command language. The* weakest preexpectation transformer

$$\mathsf{wp}\colon\ \mathsf{pGCL} \longrightarrow \mathbb{E} \longrightarrow \mathbb{E}$$

*adhering to the rules in* Table 1 *is well-defined; in fact,* Table 1 *coincides with* Def. 2.

A proof of Thm. 3 can be found in [Szymczak and Katoen 2019, Sect. 5], which extends the well-definedness for discrete probabilistic programs [Kaminski 2019, Thm. 4.11]. The function ${}^{\mathsf{wp}}_{\langle\varphi,C\rangle}\Phi_f$ in Table 1 is called the *characteristic function of* `while` $(\,\varphi\,)\ \{\,C\,\}$ *w.r.t.* $f$. For simplicity, we omit wp, $\varphi$, $C$, or $f$ from $\Phi$ whenever they are clear from the context. $\Phi$ is in fact a *(Scott-)continuous* – and thus *monotonic* – operator, i.e., $\Phi(\sup\{g_1 \preceq g_2 \preceq \ldots\}) = \sup \Phi(\{g_1 \preceq g_2 \preceq \ldots\})$; see [Szymczak and Katoen 2019, Lem. 3.1]. Thus by the Kleene fixed point theorem [Lassez et al. 1982], its *least fixed point* $\mathsf{lfp}\ \Phi = \sup_{n\in\mathbb{N}} \Phi^n(0) = \lim_{n\to\omega} \Phi^n(0)$ and *greatest fixed point* $\mathsf{gfp}\ \Phi = \inf_{n\in\mathbb{N}} \Phi^n(\infty) = \lim_{n\to\omega} \Phi^n(\infty)$ exist over the partial order $\preceq$ on $\mathbb{E}$.

The rules for the wp-transformer in Table 1 are compositional and, mostly, purely *syntactic*, thus providing the machinery for automating the weakest preexpectation calculus; see [Feng et al. 2023, Appx. B] for an example. One exception, however, is the transformation rule for `while`-loops: It amounts to determining the quantitative least fixed point which is often difficult or even impossible to compute [Kaminski et al. 2019]; it is thus desirable to *bound* them from above and/or from below. There are in principle two challenges (cf. [Hark et al. 2020]): (i) finding a candidate bound, and (ii) verifying that the candidate is indeed an upper or lower bound. In this paper, we aim to *verify candidate lower bounds* on $\mathsf{wp}[\![C]\!]\,(f)$ where $C$ is a (possibly nested) `while`-loop that may not terminate almost-surely. The termination probability of $C$ is captured by $\mathsf{wp}[\![C]\!]\,(1)\,(\sigma_0)$:

**Definition 4 (Almost-Sure Termination and Divergence).** *Let $C$ be a* pGCL *program and let $\sigma_0 \in \Sigma$ be an initial program state. Then $C$* terminates almost-surely on input $\sigma_0$ *iff*

$$\mathsf{wp}[\![C]\!]\,(1)\,(\sigma_0) \ = \ 1\,.$$

$C$ terminates almost-surely (AST) *iff* $C$ *terminates almost-surely on all inputs, i.e.,*

$$\mathsf{wp}[\![C]\!]\,(1) \;=\; 1\,.$$

$C$ diverges on input $\sigma_0$ *iff* $\mathsf{wp}[\![C]\!]\,(1)\,(\sigma_0) < 1$. $C$ diverges *iff* $C$ *diverges on some input* $\sigma_0$.

## 4 REASONING ABOUT LOWER BOUNDS

This section formulates our problem of proving lower bounds on $\mathsf{lfp}\,{}^{\mathsf{wp}}_{\langle\varphi,C\rangle}\Phi_f$, i.e., on the wp of a (possibly divergent) `while`-loop $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$ w.r.t. postexpectation $f$. We then give a high-level description of our approach in position to existing proof rules employing induction.

### 4.1 Problem Statement

The problem concerned in this paper can be formulated as follows.

> Given a *possibly divergent* probabilistic loop $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$, a *possibly unbounded* postexpectation $f \in \mathbb{E}$, and a *possibly unbounded* candidate lower bound $l \in \mathbb{E}$, verify that
>
> $$l \;\preceq\; \mathsf{wp}[\![C_{\text{loop}}]\!]\,(f)\;. \tag{1}$$

We present a new proof rule to address this problem: Our rule does not employ induction, rather, it *reduces* the verification of (1) with possibly divergent $C_{\text{loop}}$ and possibly unbounded $f, l \in \mathbb{E}$ to

$$l \;\preceq\; \mathsf{wp}[\![\texttt{while}\,(\,\varphi'\,)\,\{\,C\,\}]\!]\,([\neg\varphi]\cdot f) \qquad \text{with} \qquad \varphi' \implies \varphi\,. \tag{2}$$

Namely, we forge a new loop $C'_{\text{loop}} = \texttt{while}\,(\,\varphi'\,)\,\{\,C\,\}$ out of $C_{\text{loop}}$ by *strengthening* its loop guard $\varphi$ to $\varphi'$. Such guard strengthening restricts the (reachable) state space and, consequently, (i) the modified loop $C'_{\text{loop}}$ features a *stronger* termination property (e.g., becoming AST), and (ii) both the uniform integrability of $l$ and the boundedness of expectations are *easier* to verify.

Our proof rule asserts – by exploiting the difference between $\mathsf{wp}[\![C'_{\text{loop}}]\!]\,(f)$ and $\mathsf{wp}[\![C_{\text{loop}}]\!]\,(f)$ – that a lower bound $l$ w.r.t. $C'_{\text{loop}}$ satisfying (2) also suffices as a lower bound w.r.t. $C_{\text{loop}}$ satisfying (1). The former, due to guard strengthening, can often be obtained by applying existing lower induction rules (see Sect. 4.3 below) or – in case $C'_{\text{loop}}$ has a finite state space – probabilistic model checking.

### 4.2 Induction Rules for Upper Bounds

The *Park induction* principle [Park 1969] for least fixed points establishes an elegant mechanism for verifying upper bounds on weakest preexpectations:

**Theorem 5 (Park Induction for Upper Bounds** [Kaminski 2019; Kozen 1985]**).** *Let* $\Phi_f$ *be the characteristic function of* $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$ *w.r.t. postexpectation* $f \in \mathbb{E}$ *and let* $u \in \mathbb{E}$. *Then*

$$\Phi_f(u) \;\preceq\; u \quad \text{implies} \quad \mathsf{wp}[\![C_{\text{loop}}]\!]\,(f) \;\preceq\; u\,. \tag{3}$$

We call $u \in \mathbb{E}$ satisfying $\Phi_f(u) \preceq u$ a *superinvariant*. As pointed out by Hark et al. [2020], the striking power of Park induction lies in its *simplicity*: Once an appropriate candidate $u$ is found (which, however, is usually not an easy task), all we have to do is to push $u$ through $\Phi_f$ *once* and check whether it becomes smaller in terms of $\preceq$. If this is the case, we have verified that $u$ is indeed an upper bound on $\mathsf{lfp}\,\Phi_f$ and thus on the weakest preexpectation.

The *soundness* of Park induction is illustrated by the left (descending) chain in Fig. 3. We refer the readers to [Hark et al. 2020] for a formal soundness argument leveraging the Tarski-Kantorovitch principle (cf. [Jachymski et al. 2000]). See also [Batz et al. 2021a] for a strictly more general proof rule via (latticed) $k$-induction for establishing upper bounds on least fixed points.

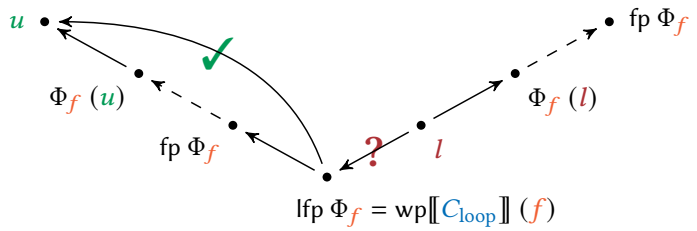


Fig. 3. Intuition of the soundness of Park induction (left branch) and the unsoundness of simple lower induction (right branch). An arrow from $g_1 \in \mathbb{E}$ to $g_2 \in \mathbb{E}$ indicates $g_1 \preceq g_2$. For Park induction, the iteration of $\Phi_f$ on $u$ converges downwards to a fixed point of $\Phi_f$ which is – by the Knaster-Tarski theorem [Knaster 1928; Lassez et al. 1982; Tarski 1955] – necessarily above lfp $\Phi_f$, thus proving wp$[\![C_{\text{loop}}]\!]\,(f) \preceq u$. For simple lower induction, however, the ascending chain $l \preceq \Phi_f(l) \preceq \ldots$ converges to a fixed point of $\Phi_f$ which is necessarily below the *greatest* fixed point gfp $\Phi_f$, but we do not know how $l$ compares to lfp $\Phi_f$.

### 4.3 Induction Rules for Lower Bounds

A "dual" version of Park induction – by flipping $\preceq$ in (3) – works for verifying lower bounds on the *greatest* fixed point gfp $\Phi_f$, but not on lfp $\Phi_f$. More precisely, for $l \in \mathbb{E}$, the rule

$$l \preceq \Phi_f(l) \quad \text{implies} \quad l \preceq \text{wp}[\![C_{\text{loop}}]\!]\,(f)\ , \qquad \lightning$$

is *unsound* in general. We call $l \in \mathbb{E}$ satisfying $l \preceq \Phi_f(l)$ a *subinvariant* and the above unsound rule *simple lower induction*. The *unsoundness* of simple lower induction is illustrated by the right (ascending) chain in Fig. 3, together with a counterexample below. We refer the readers to [Hark et al. 2020] for a formal argument again using the Tarski-Kantorovitch principle.

**Example 6 (Unsoundness of Simple Lower Induction).** Reconsider the loop $C_{\text{1dbrw}}$ in Sect. 2 with postexpectation $f = 1$; its characteristic function is $\Phi_f(h) = [n \leq 0] + [n > 0] \cdot (1/3 \cdot h(n-1) + 2/3 \cdot h(n+1))$. Observe that the constant expectation $g = 1$ is a superinvariant, since $\Phi_f(g) = 1 \preceq g$. This implies that the termination probability of $C_{\text{1dbrw}}$ is (trivially) upper-bounded by 1 (cf. Thm. 5). Meanwhile, $g$ is also a subinvariant as $g \preceq 1 = \Phi_f(g)$, which however does *not* suffice to certify 1 as a lower bound on the termination probability (recall that $C_{\text{1dbrw}}$ is non-AST; cf. Sect. 2). ◁

To retrieve soundness of lower induction, Hark et al. [2020] propose *side conditions* relying on notions of almost-sure termination (i.e., wp$[\![C_{\text{loop}}]\!]\,(1) = 1$, cf. Def. 4) and *uniform integrability* from the realm of stochastic processes. To formulate the latter, we denote by $X_n$ the random variable representing the program state after the $n$-th iteration of the loop body $C$, by $T^{\neg\varphi} \triangleq \inf\{n \in \mathbb{N} \mid X_n \models \neg\varphi\}$ the stopping time (aka, *looping time*) indicating the first time that $X_n$ hits $\neg\varphi$,[6] and by $\{X_n^{T^{\neg\varphi}}\}_{n\in\mathbb{N}}$ the corresponding stopped stochastic process, i.e., $X_n^{T^{\neg\varphi}} = X_n$ if $n \leq T^{\neg\varphi}$ and $X_n^{T^{\neg\varphi}} = X_{T^{\neg\varphi}}$ otherwise; see [Feng et al. 2023, Appx. A] for formal definitions of stopping times and stopped processes.

**Definition 7 (Uniform Integrability).** *A stochastic process $\{X_n\}_{n\in\mathbb{N}}$ on a probability space $(\Omega, \mathcal{F}, P)$ is* uniformly integrable *(*u.i.*, for short), if*

$$\lim_{R\to\infty} \sup_{n\in\mathbb{N}} E\left[|X_n| \cdot \mathbb{1}_{|X_n|\geq R}\right] = 0 \tag{4}$$

*where $\mathbb{1}_{|X_n|\geq R}$ is the indicator function, i.e., $\mathbb{1}_{|X_n|\geq R}(\omega) = 1$ if $\omega \in \{\omega \in \Omega \mid |X_n(\omega)| \geq R\}$ and 0 otherwise. Given $C_{\text{loop}} = \texttt{while}\,(\varphi)\,\{C\}$, an expectation $h \in \mathbb{E}$ is* uniformly integrable for $C_{\text{loop}}$ *if $\left\{h\left(X_n^{T^{\neg\varphi}}\right)\right\}_{n\in\mathbb{N}}$ is uniformly integrable on the probability space induced by $C_{\text{loop}}$ (cf. Sect. 5).*

[6] The looping time $T^{\neg\varphi}$ does not take into account the runtime of the loop body $C$.

Intuitively, (4) asserts that the tail expected values of $X_n$ are *uniformly* (indicated by the supremum) small in terms of the $L^1$-norm. A u.i. process $X_n$ thus satisfies $E(\lim_{n\to\infty} X_n) = \lim_{n\to\infty} E(X_n)$ if $\lim_{n\to\infty} X_n$ exists almost-surely. Now, the sound induction rule for lower bounds reads as follows.

**Theorem 8 (Hark et al.'s Induction for Lower Bounds** [Hark et al. 2020]**).** *Let $\Phi_f$ be the characteristic function of $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$ w.r.t. postexpectation $f \in \mathbb{E}$ and let $l \in \mathbb{E}$. Then*

$$\underbrace{l \preceq \Phi_f(l)}_{\text{subinvariance}} \text{ and } \underbrace{\text{wp}[\![C_{\text{loop}}]\!]\,(1) = 1 \text{ and } l \text{ is u.i. for } C_{\text{loop}}}_{\text{side conditions}} \quad \text{implies} \quad l \preceq \text{wp}[\![C_{\text{loop}}]\!]\,(f)\ . \tag{5}$$

Unlike Park induction for upper bounds, the side conditions in (5) for establishing lower bounds require extra efforts in proving almost-sure termination and uniform integrability, both of which are computationally *intractable* in general, see, e.g., [Kaminski et al. 2019]. Various techniques and tools have been developed in the literature, e.g., [Chatterjee et al. 2020; McIver et al. 2018; Moosbrugger et al. 2021], to prove almost-sure termination of (*subclasses* of) probabilistic programs. For showing uniform integrability, Hark et al. [2020] propose *sufficient* conditions based on the well-known optional stopping time theorem [Williams 1991, Chap. 10]:

**Theorem 9 (Sufficient Criteria for Uniform Integrability** [Hark et al. 2020]**).** *Given $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$, let ${}^{\sigma_0}\mathbb{P}$ be the (sub)probability measure induced by $C_{\text{loop}}$ on initial state $\sigma_0 \in \Sigma$.*[7] *Then, an expectation $h \ll\!\!\prec \infty$ is uniformly integrable for $C_{\text{loop}}$ if one of the following conditions holds:*

(a) *The looping time $T^{\neg\varphi}$ is almost-surely bounded, i.e., for any initial state $\sigma_0 \in \Sigma$, there exists $N \in \mathbb{N}$ such that ${}^{\sigma_0}\mathbb{P}(T^{\neg\varphi} \leq N) = 1$, and $\text{wp}[\![C]\!]^n(h) \ll\!\!\prec \infty$ for any $n \in \mathbb{N}$.*
(b) *The expected looping time is finite and $h$ is* conditionally difference bounded, *i.e., ${}^{\sigma_0}E[T^{\neg\varphi}] < \infty$ for any $\sigma_0 \in \Sigma$, and there exists $c \in \mathbb{R}_{\geq 0}$ such that $\text{wp}[\![C]\!]\,(|h - h(\sigma)|) \leq c$ for any $\sigma \models \varphi$.*
(c) *$h$ is bounded, i.e., there exists $c \in \mathbb{R}_{\geq 0}$ such that $h(\sigma) \leq c$ for any $\sigma \in \Sigma$.*

In summary, Hark et al.'s sound lower induction rule in Thm. 8 does not apply to *divergent* programs, and even for AST ones, it requires extra proof efforts in, e.g., looking for supermartingales [Chatterjee et al. 2020] witnessing AST and reasoning about the looping time ((a) and (b) in Thm. 9) or establishing bounds on expectations ((b) and (c) in Thm. 9) to achieve u.i..

There is an orthogonal induction rule by McIver and Morgan [2005] for *bounded* expectations:

**Theorem 10 (McIver & Morgan's Induction for Lower Bounds** [McIver and Morgan 2005]**).** *Let $\Phi_f$ be the characteristic function of $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$ w.r.t. a bounded postexpectation $f \in \mathbb{E}$, $l \in \mathbb{E}$ be a bounded expectation such that $l \preceq \Phi_f(l)$ and $[\neg\varphi] \cdot l = [\neg\varphi] \cdot f$, and $p = \text{wp}[\![C_{\text{loop}}]\!]\,(1)$ be the termination probability of $C_{\text{loop}}$. Then*

(a) *If $l = [G]$ for some predicate $G$, then $p \cdot l \preceq \text{wp}[\![C_{\text{loop}}]\!]\,(f)$.*
(b) *If $[G] \preceq p$ for some predicate $G$, then $[G] \cdot l \preceq \text{wp}[\![C_{\text{loop}}]\!]\,(f)$.*
(c) *If $\varepsilon \cdot l \preceq p$ for some $\varepsilon \in \mathbb{R}_{>0}$, then $l \preceq \text{wp}[\![C_{\text{loop}}]\!]\,(f)$.*

McIver and Morgan's lower induction rule applies to divergent programs (with termination probability $< 1$); however, it is confined to *bounded* expectations and requires a priori knowledge on the *termination probability* $p$ of a `while`-loop which is difficult to infer in general (which may in turn ask for lower bounds on $p$). In fact, Hark et al.'s induction rule generalizes McIver and Morgan's in case $C_{\text{loop}}$ is AST [Hark et al. 2020, Thm. 41]. These two proof rules, to the best of our knowledge, are the only existing (induction) rules for verifying lower bounds on weakest preexpectations.

[7] The measure ${}^{\sigma_0}\mathbb{P}$ will be formally defined in Sect. 5. Note that ${}^{\sigma_0}\mathbb{P}(T^{\neg\varphi} < \infty) = 1$ iff $\text{wp}[\![C_{\text{loop}}]\!]\,(1)\,(\sigma_0) = 1$.

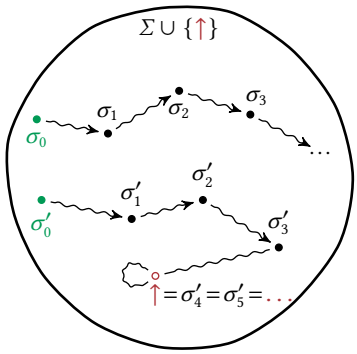


Fig. 4. Infinite traces in $\mathbb{S}$.

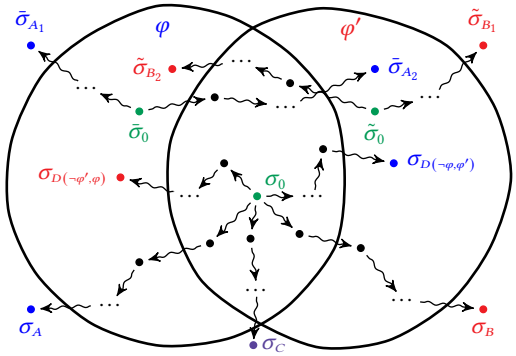


Fig. 5. Illustration of Thm. 11 (cf. (10)).

## 5 DIFFERENCES OF WEAKEST PREEXPECTATIONS

Our lower-bound proof rule reduces the verification of a probabilistic loop to that of its strengthened counterpart. To justify such a reduction, we need to *quantitatively* relate two probabilistic loops in terms of weakest preexpectations. In this section, we show how to quantify the difference of the weakest preexpectations of two `while`-loops, which differ *only* in loop guards, with respect to the same postexpectation $f \in \mathbb{E}$, namely,

$$\mathsf{wp}[\![\underbrace{\texttt{while}\,(\,\varphi\,)\,\{\,C\,\}}_{C_{\text{loop}}}]\!]\,(f) \;-\; \mathsf{wp}[\![\underbrace{\texttt{while}\,(\,\varphi'\,)\,\{\,C\,\}}_{C'_{\text{loop}}}]\!]\,(f) \tag{6}$$

where $\varphi$ and $\varphi'$ are arbitrary predicates representing subsets of $\Sigma$ (here, $\varphi'$ does *not* necessarily strengthen $\varphi$)[8] and the shared loop body $C$ itself can contain further nested `while`-loops. Such a quantification on the wp-difference forms the basis of our new proof rule (cf. Sect. 6).

Recall that the weakest preexpectation of a loop w.r.t. $f \in \mathbb{E}$ is defined (cf. Def. 2) as the integral of $f$ over the (sub)probability measure over final states reached after termination of the loop, where the termination behavior is determined by the loop guard and the loop body. Thus, to connect the two weakest preexpectations in (6), we first abstract away the loop guards $\varphi$ and $\varphi'$ and thereby obtain a certainly divergent loop $C^{\uparrow}_{\text{loop}} = \texttt{while}\,(\,\texttt{true}\,)\,\{\,C\,\}$. We then construct a probability space over the set $\mathbb{S}$ of *infinite traces* (i.e., sequences of program states) of $C^{\uparrow}_{\text{loop}}$, formally,

$$\mathbb{S} \;\triangleq\; \{\, \sigma_0\sigma_1\cdots\sigma_i\cdots \mid \sigma_0 \in \Sigma,\ \forall i \geq 1\colon \sigma_i \in \Sigma \cup \{\uparrow\} \wedge (\sigma_i = \uparrow \implies \sigma_{i+1} = \uparrow)\,\}$$

where $\sigma_i \neq \uparrow$ denotes the state in which the loop body $C$ *terminates* after its $i$-th iteration and $\uparrow$ denotes the *sink* state where $C$ diverges. See Fig. 4 for an illustration of two types of infinite traces (with or without sink states). Given a finite prefix $\pi$ of an infinite trace, the *cylinder set* of $\pi$ is $cyl(\pi) \triangleq \{s \in \mathbb{S} \mid \exists t \in (\Sigma \cup \{\uparrow\})^{\omega}\colon s = \pi t\}$, i.e., the set of infinite traces that have $\pi$ as a prefix.

The operational semantics [Dahlqvist et al. 2020] of $C^{\uparrow}_{\text{loop}}$ induces a family of (sub)probability measures – denoted by ${}^{\sigma}\mathbb{P}$ for any $\sigma \in \Sigma$ – over $\mathbb{S}$ with the smallest $\sigma$-algebra containing all cylinder

[8] Assuming $\varphi' \implies \varphi$ suffices to justify our proof rule. However, we are interested in a more general result on wp-difference with *unrelated* $\varphi$ and $\varphi'$ due to (i) symmetry in $\varphi$ and $\varphi'$; and (ii) the potential of such a general result for addressing problems beyond verifying lower bounds, e.g., sensitivity analysis [Aguirre et al. 2021; Barthe et al. 2018; Wang et al. 2020] and model repair [Bartocci et al. 2011] for probabilistic programs, as one of the interesting future directions.

sets. That is, for any finite prefix $\pi = \sigma_0\sigma_1\cdots\sigma_n$,

$$ {}^{\sigma}\mathbb{P}\left(cyl(\pi)\right) = \begin{cases} [\sigma = \sigma_0] \cdot {}^{\sigma_0}\mu_C(\sigma_1) \cdot \ldots \cdot {}^{\sigma_{i-1}}\mu_C(\sigma_i) \cdot \ldots \cdot {}^{\sigma_{n-1}}\mu_C(\sigma_n) & \text{if } \forall i \leq n\colon \sigma_i \neq \uparrow\,, \\ [\sigma = \sigma_0] \cdot {}^{\sigma_0}\mu_C(\sigma_1) \cdot \ldots \cdot {}^{\sigma_{i-1}}\mu_C(\sigma_i) \cdot (1 - {}^{\sigma_i}\mu_C(\Sigma)) & \text{if } \exists i < n\colon \sigma_i \neq \uparrow \wedge \sigma_{i+1} = \uparrow\,, \end{cases} $$

where ${}^{\sigma}\mu_C$ represents the (sub)probability measure over final states reached after termination of the loop body $C$ on input $\sigma$. The random variable $X_n\colon \mathbb{S} \to \Sigma$ representing the program state after the $n$-th iteration of the loop body $C$ is $X_n(\sigma_0\sigma_1\cdots) = \sigma_n$. Given $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$, the looping time $T^{\neg\varphi}\colon \mathbb{S} \to \mathbb{N}$ of $C_{\text{loop}}$ is defined as $T^{\neg\varphi}(s) = \inf\{n \mid X_n(s) \models \neg\varphi\}$. For any random variable $X$ over $\mathbb{S}$ and any predicate $\phi$ over $X$, we abbreviate ${}^{\sigma}\mathbb{P}(\{s \in \mathbb{S} \mid \phi(X(s))\})$ as ${}^{\sigma}\mathbb{P}(\phi(X))$, e.g., ${}^{\sigma}\mathbb{P}(T^{\neg\varphi} \geq n)$ is a shorthand for ${}^{\sigma}\mathbb{P}(\{s \in \mathbb{S} \mid T^{\neg\varphi}(s) \geq n\})$. Based on the measure ${}^{\sigma}\mathbb{P}$ defined above, the difference between $\text{wp}[\![C_{\text{loop}}]\!]\,(f)$ and $\text{wp}[\![C'_{\text{loop}}]\!]\,(f)$ as in (6) can be quantified as follows.

**Theorem 11 (wp-Difference).** *Given loops $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$ and $C'_{\text{loop}} = \texttt{while}\,(\,\varphi'\,)\,\{\,C\,\}$, then, for any postexpectation $f \in \mathbb{E}$,*[9]

$$ \begin{aligned} &\text{wp}[\![C_{\text{loop}}]\!]\,(f) - \text{wp}[\![C'_{\text{loop}}]\!]\,(f) \;=\; \\ &\qquad \text{wp}[\![\texttt{while}\,(\,\varphi \wedge \varphi'\,)\,\{\,C\,\}]\!]\,([\neg\varphi \wedge \varphi'] \cdot f) + \lambda\sigma.\, \int_A f_{C_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) - \\ &\qquad \text{wp}[\![\texttt{while}\,(\,\varphi \wedge \varphi'\,)\,\{\,C\,\}]\!]\,([\varphi \wedge \neg\varphi'] \cdot f) - \lambda\sigma.\, \int_B f_{C'_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right)\,, \end{aligned} \tag{7} $$

*where $A \subseteq \mathbb{S}$ is the set of infinite traces that hit $\neg\varphi'$ before hitting $\neg\varphi$, and $f_{C_{\text{loop}}}$ is a partial function mapping a trace $s \in \mathbb{S}$ to $f(\sigma_n)$ if $s$ hits $\neg\varphi$ for the first time at $\sigma_n$, namely,*

$$ \begin{aligned} A \;&\triangleq\; \{\,\sigma_0\sigma_1\cdots\sigma_i\cdots \in \mathbb{S} \mid \exists n \in \mathbb{N}\colon\; (\sigma_n \models \neg\varphi) \wedge (\forall i < n\colon \sigma_i \models \varphi) \wedge (\exists k < n\colon \sigma_k \models \neg\varphi')\,\}\,, \\ f_{C_{\text{loop}}}\colon \mathbb{S} &\rightharpoonup \mathbb{R}^{\infty}_{\geq 0} \qquad \sigma_0\sigma_1\cdots\sigma_i\cdots \mapsto f(\sigma_n) \quad \text{if } (\sigma_n \models \neg\varphi) \wedge (\forall i < n\colon \sigma_i \models \varphi)\;; \end{aligned} \tag{8} $$

*dually, $B \subseteq \mathbb{S}$ is the set of infinite traces that hit $\neg\varphi$ before hitting $\neg\varphi'$, and $f_{C'_{\text{loop}}}$ maps a trace $s \in \mathbb{S}$ to $f(\sigma_n)$ if $s$ hits $\neg\varphi'$ for the first time at $\sigma_n$ (the definitions of $B$ and $f_{C'_{\text{loop}}}$ are analogous to (8)).*

Proof. We prove the theorem by exploring different types of traces in $\mathbb{S}$, as depicted in Fig. 5. Given a predicate, $\phi$ let $\Diamond\phi$ be the set of all traces that *eventually* hit $\phi$, i.e.,

$$ \Diamond\phi \;\triangleq\; \{\,\sigma_0\sigma_1\cdots\sigma_i\cdots \in \mathbb{S} \mid \exists n \in \mathbb{N}\colon \sigma_n \models \phi\,\}\,. $$

By definition of the (sub)probability measure ${}^{\sigma}\mathbb{P}$, we have

$$ \begin{aligned} \text{wp}[\![C_{\text{loop}}]\!]\,(f) \;&=\; \lambda\sigma.\, \int_{\Sigma} f\,\mathrm{d}\left({}^{\sigma}\mu_{C_{\text{loop}}}\right) \;=\; \lambda\sigma.\, \int_{\Diamond(\neg\varphi)} f_{C_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right)\,, \\ \text{wp}[\![C'_{\text{loop}}]\!]\,(f) \;&=\; \lambda\sigma.\, \int_{\Sigma} f\,\mathrm{d}\left({}^{\sigma}\mu_{C'_{\text{loop}}}\right) \;=\; \lambda\sigma.\, \int_{\Diamond(\neg\varphi')} f_{C'_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right)\,. \end{aligned} \tag{9} $$

To quantify the wp-difference, we first decompose the set of traces in $\Diamond(\neg\varphi)$ and $\Diamond(\neg\varphi')$ into disjoint parts, respectively. To this end, let

$$ C \;\triangleq\; \{\,\sigma_0\sigma_1\cdots\sigma_i\cdots \in \mathbb{S} \mid \exists n \in \mathbb{N}\colon\; (\sigma_n \models \neg\varphi \wedge \neg\varphi') \wedge (\forall i < n\colon \sigma_i \models \varphi \wedge \varphi')\,\}\,, $$

and for any predicates $\phi_1, \phi_2$, let

$$ D\,(\phi_1, \phi_2) \;\triangleq\; \{\,\sigma_0\sigma_1\cdots\sigma_i\cdots \in \mathbb{S} \mid \exists n \in \mathbb{N}\colon\; (\sigma_n \models \phi_1) \wedge (\forall i \leq n\colon \sigma_i \models \phi_2)\,\}\,. $$

[9] For better understandability, (7) is formulated in the form of *difference* between weakest expectations. This formulation may raise the issue of "$\infty - \infty$", however, one can avoid this issue by shifting all the negative terms in (7) to the other side of the equation. The same reformulation tactic applies to the proof of Thm. 11.

It follows that

$$\begin{aligned}
\Diamond(\neg\varphi) &= \overbrace{A}^{\substack{\text{hitting } \neg\varphi' \text{ before hitting } \neg\varphi\\ \text{e.g., } \sigma_0\cdots\sigma_A\cdots,\tilde{\sigma}_0\cdots \text{ in Fig. 5}}} \uplus \overbrace{C}^{\substack{\text{hitting } \neg\varphi',\neg\varphi \text{ simultaneously}\\ \text{e.g., } \sigma_0\cdots\sigma_C\cdots \text{ in Fig. 5}}} \uplus \overbrace{D(\neg\varphi,\varphi')}^{\substack{\text{hitting } \neg\varphi \text{ while satisfying } \varphi'\\ \text{e.g., } \sigma_0\cdots\sigma_{D(\neg\varphi,\varphi')}\cdots \text{ in Fig. 5}}}, \\
\Diamond(\neg\varphi') &= \overbrace{B}^{\substack{\text{hitting } \neg\varphi \text{ before hitting } \neg\varphi'\\ \text{e.g., } \sigma_0\cdots\sigma_B\cdots,\tilde{\sigma}_0\cdots \text{ in Fig. 5}}} \uplus \overbrace{C}^{\substack{\text{hitting } \neg\varphi,\neg\varphi' \text{ simultaneously}\\ \text{e.g., } \sigma_0\cdots\sigma_C\cdots \text{ in Fig. 5}}} \uplus \overbrace{D(\neg\varphi',\varphi)}^{\substack{\text{hitting } \neg\varphi' \text{ while satisfying } \varphi\\ \text{e.g., } \sigma_0\cdots\sigma_{D(\neg\varphi',\varphi)}\cdots \text{ in Fig. 5}}}.
\end{aligned} \tag{10}$$

It is evident that, by definition, $f_{C_{\text{loop}}}$ and $f_{C'_{\text{loop}}}$ coincide on $C \subseteq \mathbb{S}$, i.e.,

$$\forall s \in C\colon\ f_{C_{\text{loop}}}(s) = f_{C'_{\text{loop}}}(s)\ . \tag{11}$$

For $D(\cdot,\cdot)$, we have (see [Feng et al. 2023, Lem. 32] for a more detailed proof of (12) below)

$$\begin{aligned}
\lambda\sigma.\ \int_{D(\neg\varphi,\varphi')} f_{C_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) &= \underbrace{\mathrm{wp}[\![\texttt{while}\,(\,\varphi\wedge\varphi'\,)\,\{\,C\,\}]\!]\,([\neg\varphi\wedge\varphi']\cdot f)}_{\substack{\text{evaluating } f \text{ over, e.g., } \sigma_{D(\neg\varphi,\varphi')} \text{ in Fig. 5}}}, \\
\lambda\sigma.\ \int_{D(\neg\varphi',\varphi)} f_{C'_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) &= \underbrace{\mathrm{wp}[\![\texttt{while}\,(\,\varphi\wedge\varphi'\,)\,\{\,C\,\}]\!]\,([\varphi\wedge\neg\varphi']\cdot f)}_{\substack{\text{evaluating } f \text{ over, e.g., } \sigma_{D(\neg\varphi',\varphi)} \text{ in Fig. 5}}}.
\end{aligned} \tag{12}$$

By combining the facts above, we have

$$\begin{aligned}
&\mathrm{wp}[\![C_{\text{loop}}]\!]\,(f) - \mathrm{wp}[\![C'_{\text{loop}}]\!]\,(f) = \lambda\sigma.\ \int_{\Diamond(\neg\varphi)} f_{C_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) - \lambda\sigma.\ \int_{\Diamond(\neg\varphi')} f_{C'_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) \quad \text{[by (9)]} \\
&= \lambda\sigma.\ \int_{A} f_{C_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) + \lambda\sigma.\ \int_{C} f_{C_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) + \lambda\sigma.\ \int_{D(\neg\varphi,\varphi')} f_{C_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) - \\
&\quad \lambda\sigma.\ \int_{B} f_{C'_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) - \lambda\sigma.\ \int_{C} f_{C'_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) - \lambda\sigma.\ \int_{D(\neg\varphi',\varphi)} f_{C'_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) \quad \text{[by linearity of } \textstyle\int\text{, (10)]} \\
&= \mathrm{wp}[\![\texttt{while}\,(\,\varphi\wedge\varphi'\,)\,\{\,C\,\}]\!]\,([\neg\varphi\wedge\varphi']\cdot f) + \lambda\sigma.\ \int_{A} f_{C_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) - \\
&\quad \mathrm{wp}[\![\texttt{while}\,(\,\varphi\wedge\varphi'\,)\,\{\,C\,\}]\!]\,([\varphi\wedge\neg\varphi']\cdot f) - \lambda\sigma.\ \int_{B} f_{C'_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right)\ . \quad \text{[by (11) and (12)]}
\end{aligned}$$

This completes the proof. □

**Example 12 (wp-Difference).** Consider two loops with postexpectation $f = [0 \le n \le 11]$:

$$\begin{aligned}
C_{\text{loop}}&\colon\quad \texttt{while}\,(\,0 < n < 10\,)\,\{\,n \coloneqq n-1\ [1/2]\ n \coloneqq n+1\,\}\ , \\
C'_{\text{loop}}&\colon\quad \texttt{while}\,(\,1 < n < 11\,)\,\{\,n \coloneqq n-1\ [1/2]\ n \coloneqq n+1\,\}\ .
\end{aligned}$$

Let $C_\wedge$ be the loop $\texttt{while}(1 < n < 10)\{n \coloneqq n-1\ [1/2]\ n \coloneqq n+1\}$. To illustrate Thm. 11, we show

$$\begin{aligned}
&\mathrm{wp}[\![C_{\text{loop}}]\!]\,([0 \le n \le 11]) - \mathrm{wp}[\![C'_{\text{loop}}]\!]\,([0 \le n \le 11]) \ = \\
&\qquad \mathrm{wp}[\![C_\wedge]\!]\,([n = 10]) + \lambda\sigma.\ \int_{A} f_{C_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) - \\
&\qquad \mathrm{wp}[\![C_\wedge]\!]\,([n = 1]) - \lambda\sigma.\ \int_{B} f_{C'_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right)\ .
\end{aligned} \tag{13}$$

Observe that

$$\mathrm{wp}[\![C_{\text{loop}}]\!]\,([0 \le n \le 11]) = \mathrm{wp}[\![C'_{\text{loop}}]\!]\,([0 \le n \le 11]) = [0 \le n \le 11]\ ,$$

due to the key fact that $C_{\text{loop}}$ and $C'_{\text{loop}}$ both terminate with probability 1. Thus, the left-hand side of (13) equals 0. For the right-hand side of (13), by applying Hark et al.'s induction for lower bounds and Park induction for upper bounds (details omitted), one can show that

$$\text{wp}[\![C_{\wedge}]\!]\,([n=10]) \;=\; [1 \le n \le 10] \cdot \frac{n-1}{9}\,, \qquad \text{wp}[\![C_{\wedge}]\!]\,([n=1]) \;=\; [1 \le n \le 10] \cdot \frac{10-n}{9}\,.$$

Moreover, notice that $f_{C_{\text{loop}}} = 1$ over $A$ and $f_{C'_{\text{loop}}} = 1$ over $B$, we have

$$\int_A f_{C_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) \;=\; [1 \le n < 10] \cdot \frac{10-n}{9}\,, \qquad \int_B f_{C'_{\text{loop}}}\,\mathrm{d}\left({}^{\sigma}\mathbb{P}\right) \;=\; [1 < n \le 10] \cdot \frac{n-1}{9}\,.$$

It follows that the right-hand side of (13) also equals 0. ◁

The intuition behind Thm. 11 is to decompose the weakest preexpectations into disjoint parts covering different types of traces in $\mathbb{S}$ – the type of a trace is determined by its temporal behavior of hitting $\neg\varphi$ and/or hitting $\neg\varphi'$, see (10) and Fig. 5 – thus yielding the *exact* difference between $\text{wp}[\![C_{\text{loop}}]\!]\,(f)$ and $\text{wp}[\![C'_{\text{loop}}]\!]\,(f)$ by eliminating their common parts, cf. (11). The integrals in (7) for their exclusive parts remain hard to resolve, however, it suffices to obtain a lower bound on $\text{wp}[\![C_{\text{loop}}]\!]\,(f)$ if $\varphi'$ strengthens $\varphi$ (tightness of the so-obtained lower bound is shown in Sect. 6.3):

**Corollary 13.** *Given loops* $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$ *and* $C'_{\text{loop}} = \texttt{while}\,(\,\varphi'\,)\,\{\,C\,\}$, *suppose* $\varphi' \implies \varphi$, *then, for any postexpectation* $f \in \mathbb{E}$,

$$\text{wp}[\![C_{\text{loop}}]\!]\,(f) \;\ge\; \text{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f)\,.$$

The intuition of Cor. 13 is as follows: Recall that $\text{wp}[\![C_{\text{loop}}]\!]\,(f)$ maps any initial state $\sigma_0$ to the expected value of $f$ evaluated in the final states reached after termination of $C_{\text{loop}}$ on $\sigma_0$, i.e., upon violating the loop guard $\varphi$. In order to obtain a sound lower bound on $\text{wp}[\![C_{\text{loop}}]\!]\,(f)$ via the modified loop $C'_{\text{loop}}$, we have to restrict the postexpectation $f$ to $[\neg\varphi]\cdot f$ such that $f$ is evaluated only in states violating the original guard $\varphi$ (and hence also violating the strengthened guard $\varphi'$). The proof of Cor. 13 leverages the fact that, as $\varphi' \implies \varphi$, we have $\varphi \wedge \varphi' = \varphi'$, $[\neg\varphi \wedge \varphi'] = 0$, and $B = \emptyset$ (i.e., no trace can ever hit $\neg\varphi$ before hitting $\neg\varphi'$, cf. (10)).

## 6 PROOF RULE FOR LOWER BOUNDS

In this section, we present our new proof rule for verifying lower bounds on weakest preexpectations – termed the *guard-strengthening rule* – based on the wp-difference and the guard-strengthening technique in Sect. 5. We then showcase the usefulness of this proof rule in several aspects, in particular, for reasoning about possibly divergent probabilistic programs.

**Theorem 14 (Guard Strengthening for Lower Bounds).** *Given loops* $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$, $C'_{\text{loop}} = \texttt{while}\,(\,\varphi'\,)\,\{\,C\,\}$, *and postexpectation* $f \in \mathbb{E}$, *let* $l \in \mathbb{E}$, *then the following inference rule holds:*

$$\frac{\varphi' \implies \varphi \qquad l \;\le\; \text{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f)}{l \;\le\; \text{wp}[\![C_{\text{loop}}]\!]\,(f)}\;\textsc{Guard-Strengthening}\,. \tag{14}$$

Proof. The (soundness of the) proof rule in (14) is an immediate consequence of Cor. 13. We provide in [Feng et al. 2023, Appx. C.3] an alternative proof which is *trace-agnostic* and thus simpler, yet does not contribute to our argument on the tightness of the proof rule in Sect. 6.3. □

Our guard-strengthening rule asserts that a lower bound $l$ on $\text{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f)$ suffices as a lower bound on $\text{wp}[\![C_{\text{loop}}]\!]\,(f)$ provided that $\varphi' \implies \varphi$. Such guard strengthening restricts the (reachable) state space and, consequently, (i) the modified loop $C'_{\text{loop}}$ features a *stronger* termination

property (e.g., becoming AST), and (ii) both the uniform integrability of $l$ and the boundedness of expectations are *easier* to verify. We will show that, due to these effects, our proof rule is

(a) *general* (Sect. 6.1 and 6.2): it is applicable to divergent $C_{\text{loop}}$ and unbounded $f, l \in \mathbb{E}$ where existing rules, e.g., Thm. 8 and 10, do not apply; even when $C_{\text{loop}}$ is AST, it is capable of certifying a lower bound $l$ which is *not* uniformly integrable for $C_{\text{loop}}$ (thus beyond the scope of Hark et al.'s lower induction rule [Hark et al. 2020]); moreover, it admits an orthogonal sufficient criterion for proving uniform integrability.
(b) *tight* (Sect. 6.3): the error incurred by underapproximating $\text{wp}[\![C_{\text{loop}}]\!]\,(f)$ with $\text{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f)$ approaches zero when $\varphi'$ approaches $\varphi$ appropriately; and
(c) *automatable* (Sect. 6.4): it is amenable to automation – modulo an appropriate strengthening – via probabilistic model checking in case $C'_{\text{loop}}$ has a finite state space.

**Remark.** The *completeness* of the guard-strengthening rule in (14) is evident: If $l$ is a lower bound on $\text{wp}[\![C_{\text{loop}}]\!]\,(f)$, then there always exists $\varphi'$ strengthening $\varphi$ such that $l$ is a lower bound of $\text{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f)$, as one can always choose $\varphi' = \varphi$ yielding $\text{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f) = \text{wp}[\![C_{\text{loop}}]\!]\,(f)$. This completeness argument, however, shall not be viewed as a characterization of the generality of our proof rule in *absolute* terms, as choosing $\varphi' = \varphi$ does not turn a non-AST loop into an AST one. We remark that characterizing the absolute generality of our proof rule is non-trivial as it depends highly on the "quality" of guard strengthening which may in turn rely on expert knowledge. Nonetheless, we provide useful heuristics in Sect. 6.4 for finding a "good" strengthening. ◁

**Remark.** Our guard-strengthening rule in (14) applies also to *non-probabilistic* loops $C_{\text{loop}}$. In this case, the weakest preexpectation transformer degenerates to the weakest precondition transformer of Dijkstra [1975, 1976], $l$ and $f$ become predicates partially ordered by $\Longrightarrow$, and $\cdot$ becomes $\wedge$. Our proof rule is then capable of verifying an underapproximation $l$ of the *set of initial states* on which $C_{\text{loop}}$ certainly terminates in states satisfying $f$.

### 6.1 Application to Possibly Divergent Programs

Our guard-strengthening rule reduces the verification of $l \preceq \text{wp}[\![C_{\text{loop}}]\!]\,(f)$ – with possibly divergent $C_{\text{loop}}$ and possibly unbounded $f, l \in \mathbb{E}$ – to that of $l \preceq \text{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f)$. In order to apply existing lower induction rules to the latter, one has to prove almost-sure termination of the modified loop $C'_{\text{loop}}$ and/or boundedness of expectations. To show AST, one may apply established techniques based on *supermartingales*, e.g., [Chatterjee et al. 2020; McIver et al. 2018; Moosbrugger et al. 2021]. However, due to our guard-strengthening technique, a novel, simpler argument that does not rely on the discovery of fine-tuned supermartingales often suffices to prove AST of $C'_{\text{loop}}$:

**Lemma 15 (AST Witness).** *Given $C'_{\text{loop}} = \texttt{while}\,(\varphi')\,\{C\}$, let $T^{\neg\varphi'}$ be the looping time of $C'_{\text{loop}}$. If there exist $N \in \mathbb{N}$ and $p \in (0,1)$ such that, for any initial state $\sigma \in \Sigma$,*

$$ {}^{\sigma}\mathbb{P}\left(T^{\neg\varphi'} > N\right) \;\leq\; p\,, \tag{15}$$

*then $C'_{\text{loop}}$ terminates almost-surely.*

Proof (sketch). It suffices to show that the *tail probability* ${}^{\sigma}\mathbb{P}(T^{\neg\varphi'} > n)$ (see, e.g., [Sankaranarayanan 2020]), i.e., the probability that $C'_{\text{loop}}$ does not terminate within $n \in \mathbb{N}$ steps, decreases exponentially in $n$. Then, by pushing $n$ ad infinitum, we have ${}^{\sigma}\mathbb{P}(T^{\neg\varphi'} < \infty) = 1 - \lim_{n\to\infty} {}^{\sigma}\mathbb{P}(T^{\neg\varphi'} > n) \geq 1 - 0 = 1$. Hence, $C'_{\text{loop}}$ terminates almost-surely. See details in [Feng et al. 2023, Appx. C.4] □

**Remark.** How to specify the class of loops $C_{\text{loop}}$ for which there exist strengthened counterparts $C'_{\text{loop}}$ fulfilling the AST criterion in Lem. 15 is left as an interesting open question. This may also contribute to the characterization of the absolute generality of our proof rule as discussed above. ◁

There are (semi-)automated approaches, e.g., [Chatterjee et al. 2016a], tailored for proving exponentially decreasing nontermination probabilities via *ranking-supermartingales*. In our setting, however, Lem. 15 is arguably *simpler* to verify since after strengthening the guard (by, e.g., bounding the variables, cf. Sect. 6.4), one can usually *read off* the constants $N$ and $p$ in (15) directly from the modified program with (certain) bounded variables. In fact, Lem. 15 suffices to witness AST of the modified loops in *all* the examples presented in this paper; moreover, it enables an additional sufficient condition for establishing uniform integrability which is orthogonal to Thm. 9 (cf. Sect. 6.2).

**Example 16 (1-D Biased Random Walk on $\mathbb{Z}$ [McIver et al. 2018]).** Recall the example in Sect. 2:

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

This loop terminates as soon as it reaches a state $\sigma$ with $\sigma(n) \leq 0$. We are interested in establishing non-trivial lower bounds on the termination probability of $C_{\text{1dbrw}}$, i.e., on $\text{wp}[\![C_{\text{1dbrw}}]\!]\,(f)$ with $f = 1$. However, *none of the existing proof rules applies*: (i) $C_{\text{1dbrw}}$ is not AST due to its biased nature (see, e.g., [McIver et al. 2018]), and thus Hark et al.'s lower induction [Hark et al. 2020] does not apply; moreover, (ii) it makes no sense to apply McIver and Morgan's lower induction [McIver and Morgan 2005] as having (a lower bound on) the termination probability – the quantity we aim to infer – is in turn one of the prerequisites for applying the rule (cf. Thm. 10).

We now show how our guard-strengthening rule can be used to verify non-trivial lower bounds on $\text{wp}[\![C_{\text{1dbrw}}]\!]\,(1)$. To this end, we strengthen the guard $(n > 0)$ of $C_{\text{1dbrw}}$ to $\varphi^M = (0 < n < M)$ for any fixed $M \in \mathbb{N}$, and hence obtain a modified loop which differs from $C_{\text{1dbrw}}$ only in the guard:

$$C^M_{\text{1dbrw}}\colon \quad \texttt{while}\,(\,0 < n < M\,)\,\{\,n \coloneqq n-1\ [1/3]\ n \coloneqq n+1\,\}\ .$$

Our proof rule asserts that, for any fixed $M \in \mathbb{N}$, a lower bound on $\text{wp}[\![C^M_{\text{1dbrw}}]\!]\,([n \leq 0]\cdot 1)$ suffices as a lower bound on $\text{wp}[\![C_{\text{1dbrw}}]\!]\,(1)$. Since $C^M_{\text{1dbrw}}$ terminates after at most $M-1$ steps of consecutively moving to the right with probability $(2/3)^{M-1}$, we have ${}^{\sigma}\mathbb{P}\big(T^{\neg\varphi^M} > M\big) < 1-(2/3)^{M-1}$ for any initial state $\sigma(n) \in \mathbb{Z}$. Thus, by Lem. 15, $C^M_{\text{1dbrw}}$ terminates almost-surely. We can therefore try to apply Hark et al.'s lower induction for a lower bound on $\text{wp}[\![C^M_{\text{1dbrw}}]\!]\,([n \leq 0]\cdot 1)$. Let

$$l_M \;=\; [n < 0] + [0 \leq n \leq M]\cdot\Big((1/2)^n - (1/2)^M\Big)\ .$$

It is straightforward to check (see [Feng et al. 2023, Appx. D.1]) that, for any fixed $M \in \mathbb{N}$, $l_M$ is a subinvariant, i.e.,

$$l_M \;\leq\; \Phi^M_g\,(l_M)$$

where $g = [n \leq 0]\cdot 1$ is the restricted postexpectation, and $\Phi^M_g$ is the characteristic function of $C^M_{\text{1dbrw}}$ with respect to $g$. Furthermore, $l_M$ is bounded from above by 1, and thus is uniformly integrable for $C^M_{\text{1dbrw}}$ due to condition (c) in Thm. 9. Consequently, we have

$$\begin{aligned} l_M \;&\leq\; \text{wp}[\![C^M_{\text{1dbrw}}]\!]\,([n \leq 0]\cdot 1) && \text{[by Hark et al.'s induction, cf. Thm. 8]}\\ &\leq\; \text{wp}[\![C_{\text{1dbrw}}]\!]\,(1)\ . && \text{[by guard-strengthening rule, cf. Thm. 14]}\end{aligned}$$

We therefore conclude that, for any fixed $M \in \mathbb{N}$, $l_M$ is a lower bound on $\text{wp}[\![C_{\text{1dbrw}}]\!]\,(1)$. In particular, by pushing $M$ ad infinitum – to obtain the *tightest* possible bound; this step is however *not necessary* to establish a *valid* lower bound – we have

$$\text{wp}[\![C_{\text{1dbrw}}]\!]\,(1) \;\geq\; \lim_{M\to\infty} l_M \;=\; [n < 0] + [n \geq 0]\cdot(1/2)^n\ ,$$

namely, $l_\infty = [n < 0] + [n \geq 0] \cdot (1/2)^n$ is a lower bound on $\mathsf{wp}[\![C_{\text{1dbrw}}]\!]\,(1)$. In fact, for this example, one can verify (cf. [Feng et al. 2023, Appx. D.1]) that $l_\infty$ is a superinvariant, i.e., $\Phi_f(l_\infty) \preceq l_\infty$, where $\Phi_f$ is the characteristic function of $C_{\text{1dbrw}}$ w.r.t. postexpectation $f = 1$. Thus, by Park induction (cf. Thm. 5), $l_\infty$ is also an upper bound on $\mathsf{wp}[\![C_{\text{1dbrw}}]\!]\,(1)$. We can thus claim that the *exact* least fixed point of $\Phi_f$, i.e., the *exact termination probability of* $C_{\text{1dbrw}}$ *starting from initial position* $n$, is

$$\mathsf{wp}[\![C_{\text{1dbrw}}]\!]\,(1) \;=\; l_\infty \;=\; [n < 0] + [n \geq 0] \cdot (1/2)^n \;.$$

Finally, we remark that the reasoning process in this example applies to a more general form of one-dimensional random walk with *parametrized* biased probability $p \in [0, 1/2)$, whose exact termination probability can be shown to be $[n < 0] + [n \geq 0] \cdot (p/1-p)^n$. ◁

Next, we show via the following example that even when the targeted loop $C_{\text{loop}}$ almost-surely terminates, a genuine lower bound on $\mathsf{wp}[\![C_{\text{loop}}]\!]\,(f)$ may *not* be uniformly integrable for $C_{\text{loop}}$, thereby staying out-of-reach by Hark et al.'s lower induction rule; our guard-strengthening rule, however, is capable of certifying such a lower bound.

**Example 17 (Bernoulli's St. Petersburg Paradox** [Bernoulli 1954]**).** Consider the following loop $C_{\text{pd}}$ modelling a lottery game via fair-coin tosses: The stake $b \in \mathbb{Z}_{>0}$ is doubled every time heads appears; the first time tails occurs, the game ends and the player wins all the stake in the pot:

$$C_{\text{pd}}\colon \quad \texttt{while}\,(\,a = 1\,)\,\{\,a \coloneqq 0\;[1/2]\;b \coloneqq 2 \cdot b\,\}\;.$$

We are interested in establishing non-trivial lower bounds on the expected payoff of the lottery, i.e., lower bounds on $\mathsf{wp}[\![C_{\text{pd}}]\!]\,(f)$ with $f = b$. Since $C_{\text{pd}}$ terminates by setting $a$ to 0 with probability $1/2$, we have ${}^{\sigma}\mathbb{P}(T^{a\neq 1} > 1) < 1/2$ for any initial state $\sigma$. Thus, by Lem. 15, $C_{\text{pd}}$ terminates almost-surely. However, the genuine lower bound $l_\infty = ([a \neq 1] \cdot b + [a = 1] \cdot \infty) \not\ll \infty$ – which we ultimately aim to verify – is *not* uniformly integrable for $C_{\text{pd}}$, and thus cannot be addressed by Hark et al.'s lower induction rule (see a detailed argument in [Hark et al. 2019, Appx. A]).

To apply our new proof rule, we strengthen the guard $(a = 1)$ of $C_{\text{pd}}$ to $\varphi^M = (a = 1 \wedge b < M)$ for any fixed $M \in \mathbb{Z}_{>0}$, and thereby obtain the modified loop:

$$C_{\text{pd}}^M\colon \quad \texttt{while}\,(\,a = 1 \wedge b < M\,)\,\{\,a \coloneqq 0\;[1/2]\;b \coloneqq 2 \cdot b\,\}\;.$$

Our proof rule asserts that, for any fixed $M \in \mathbb{Z}_{>0}$, a lower bound on $\mathsf{wp}[\![C_{\text{pd}}^M]\!]\,([a \neq 1] \cdot b)$ suffices as a lower bound on $\mathsf{wp}[\![C_{\text{pd}}]\!]\,(b)$. Program $C_{\text{pd}}^M$ is clearly AST (by the same argument as for $C_{\text{pd}}$); in fact, its looping time $T^{\neg\varphi^M}$ is certainly bounded by $\lceil \log_2 M \rceil$, and thus by condition (a) in Thm. 9, any expectation $h \ll \infty$ is uniformly integrable for $C_{\text{pd}}^M$ (as $\mathsf{wp}[\![C]\!]^n(h) \ll \infty$ for any $n \in \mathbb{N}$ is vacuously satisfied by $h \ll \infty$ and loop-free body $C$, which is the case for $C_{\text{pd}}^M$). Let

$$l_M \;=\; [a \neq 1] \cdot b + \sum\nolimits_{i=1}^{\lceil \log_2 M \rceil} [a = 1 \wedge (M/2^i \leq b < M/2^{i-1})] \cdot i \cdot b/2\;.$$

It is straightforward to check (see [Feng et al. 2023, Appx. D.2]) that, for any fixed $M \in \mathbb{Z}_{>0}$, $l_M$ is a subinvariant, i.e.,

$$l_M \;\preceq\; \Phi_g^M\,(l_M)$$

where $g = [a \neq 1] \cdot b$ is the restricted postexpectation, and $\Phi_g^M$ is the characteristic function of $C_{\text{pd}}^M$ with respect to $g$. Since $l_M \ll \infty$ is uniformly integrable for $C_{\text{pd}}^M$, by Thm. 8 and 14, we have

$$l_M \;\preceq\; \mathsf{wp}[\![C_{\text{pd}}^M]\!]\,([a \neq 1] \cdot b) \;\preceq\; \mathsf{wp}[\![C_{\text{pd}}]\!]\,(b)$$

for any fixed $M \in \mathbb{Z}_{>0}$. It follows that, when $M$ tends to infinity,

$$\mathsf{wp}[\![C_{\text{pd}}]\!]\,(b) \;\geq\; \lim_{M\to\infty} l_M \;=\; [a \neq 1] \cdot b + [a = 1] \cdot \infty \;=\; l_\infty\;.$$

In analogy to Exmp. 16, one can further show that $l_\infty$ suffices as a superinvariant and ultimately conclude that $\mathrm{wp}[\![C_{\mathrm{pd}}]\!]\,(b) = l_\infty$, that is, the *exact* expected value of $b$ upon termination. ◁

### 6.2 Orthogonal Sufficient Condition for Uniform Integrability

Exmp. 16 and 17 demonstrate how we can certify lower bounds on $\mathrm{wp}[\![C_{\mathrm{loop}}]\!]\,(f)$ leveraging existing sufficient criteria for uniform integrability (cf. Thm. 9) for the modified loop $C'_{\mathrm{loop}}$. The fact that $C'_{\mathrm{loop}}$ often features a strong and easily checkable termination property (cf. Lem. 15) enables an alternative sufficient condition for establishing uniform integrability:

**Theorem 18 (Orthogonal Sufficient Criterion for Uniform Integrability).** *Suppose* $C'_{\mathrm{loop}} = \texttt{while}\,(\,\varphi'\,)\,\{\,C\,\}$ *satisfies the AST condition in* Lem. 15. *Then, an expectation* $h \ll\!\!\ll \infty$ *is uniformly integrable for* $C'_{\mathrm{loop}}$ *if (i)* $C'_{\mathrm{loop}}$ *has the* bounded update property[10]*, i.e., there exists* $c \in \mathbb{R}_{\geq 0}$ *such that for any* $\sigma \in \Sigma$, ${}^{\sigma}\mathbb{P}(\forall n \in \mathbb{N}\colon |X_{n+1} - X_n| \leq c) = 1$, *and (ii)* $h$ *is bounded by a polynomial expectation*[11].

The sufficient criterion for uniform integrability in Thm. 18 is orthogonal to those in Thm. 9, in particular, to condition (b) thereof: Our new sufficient condition leverages the fact that $C'_{\mathrm{loop}}$ with a strengthened guard often features a strong and easily checkable termination property (cf. Lem. 15); it is confined to loops with the bounded update property, yet relaxes the conditional difference boundedness of $h$ to the polynomial boundedness of $h$. Our new sufficient criterion is usually *easier* to check than condition (b) in Thm. 9 in the sense that (i) in case $C'_{\mathrm{loop}}$ has a loop-free body, one can simply read off the bounded update property from the syntax of $C'_{\mathrm{loop}}$, and (ii) any (piece-wise) polynomial expectation $h$ fulfils the polynomial boundedness condition. The following example demonstrates a scenario where *none* of the conditions in Thm. 9 suffices to show uniform integrability, but our (orthogonal) sufficient condition in Thm. 18 does.

**Example 19 (1-D Symmetric Random Walk with Bounded Updates).** Consider the following loop $C_{\mathrm{bu}}$ modelling a symmetrized 1-D random walk on $\mathbb{Z}$, where we additionally increase (resp. decrease) the value of $b \in \mathbb{R}_{\geq 0}$ by 2 along every left (resp. right) move with probability $1/2$:

$$C_{\mathrm{bu}}\colon \quad \texttt{while}\,(\,n > 0\,)\,\{\,\{n \coloneqq n-1 \,\mathbin{⨟}\, b \coloneqq b+2\}\ [1/2]\ \{n \coloneqq n+1 \,\mathbin{⨟}\, b \coloneqq b-2\}\,\}\ .$$

We are interested in certifying non-trivial lower bounds on the expected value of $b$ upon termination, i.e., lower bounds on $\mathrm{wp}[\![C_{\mathrm{bu}}]\!]\,(f)$ with $f = b$. Notice that $C_{\mathrm{bu}}$ exhibits exactly the same termination behavior as the standard 1-D symmetric random walk [McIver et al. 2018], namely, $C_{\mathrm{bu}}$ terminates almost-surely yet with an *infinite* expected looping time. Moreover, the candidate lower bound $l_\infty = [n < 0] \cdot b + [n \geq 0] \cdot (b + 2 \cdot n)$ – which we ultimately aim to verify – is *unbounded*, thereby out-of-reach by Hark et al.'s lower induction (as Thm. 9 does not suffice to show u.i. of $l_\infty$).

To apply our new proof rule, we strengthen the guard $(n > 0)$ of $C_{\mathrm{bu}}$ to $\varphi^M = (0 < n < M)$ for any fixed $M \in \mathbb{Z}_{>0}$, and thereby obtain the modified loop:

$$C^M_{\mathrm{bu}}\colon \quad \texttt{while}\,(\,0 < n < M\,)\,\{\,\{n \coloneqq n-1 \,\mathbin{⨟}\, b \coloneqq b+2\}\ [1/2]\ \{n \coloneqq n+1 \,\mathbin{⨟}\, b \coloneqq b-2\}\,\}\ .$$

Our proof rule asserts that, for any fixed $M \in \mathbb{Z}_{>0}$, a lower bound on $\mathrm{wp}[\![C^M_{\mathrm{bu}}]\!]\,([n \leq 0] \cdot b)$ suffices as a lower bound on $\mathrm{wp}[\![C_{\mathrm{bu}}]\!]\,(b)$. Let

$$l_M \;=\; [n < 0] \cdot b + [0 \leq n \leq M] \cdot (b + 2 \cdot n) \cdot (1 - n/M)\ .$$

One can readily verify that, for any fixed $M \in \mathbb{Z}_{>0}$, $l_M$ is a subinvariant (see [Feng et al. 2023, Appx. D.3]). Since $C^M_{\mathrm{bu}}$ terminates after at most $M-1$ steps of consecutively increasing $n$ with

[10] The bounded update property has been used to analyze expected costs for nondeterministic probabilistic programs [Wang et al. 2019]; it subsumes the class of constant probability programs [Giesl et al. 2019].

[11] One can replace (ii) with (ii') $h$ is bounded by a *piece-wise* polynomial expectation, and the theorem still holds.

probability $(1/2)^{M-1}$, we have ${}^{\sigma}\mathbb{P}\big(T^{\neg\varphi^M} > M\big) < 1 - (1/2)^{M-1}$ for any initial state $\sigma(n) \in \mathbb{Z}$. Thus, by Lem. 15, $C_{\text{bu}}^M$ terminates almost-surely. Nevertheless, still *none of the conditions in* Thm. 9 *suffices to show that* $l_M$ *is uniformly integrable for* $C_{\text{bu}}^M$: the looping time $T^{\neg\varphi^M}$ is not almost-surely bounded, $l_M$ is not bounded due to unbounded variable $b$ thereof, and $l_M$ is not conditionally difference bounded (see [Feng et al. 2023, Appx. D.3]).

However, it is evident that our (orthogonal) sufficient criterion in Thm. 18 is fulfilled, thereby witnessing uniform integrability of $l_M$ for $C_{\text{bu}}^M$: $C_{\text{bu}}^M$ has the bounded update property, as it has a loop-free body with only bounded updates; moreover, $l_M$ meets the polynomial boundedness condition as, for any fixed $M \in \mathbb{Z}_{>0}$, $l_M$ is a piece-wise polynomial in $n$ and $b$. Consequently, by Thm. 8 and 14, we have, for any fixed $M \in \mathbb{Z}_{>0}$,

$$l_M \;\preceq\; \mathrm{wp}[\![C_{\text{bu}}^M]\!]\,([n \le 0] \cdot b) \;\preceq\; \mathrm{wp}[\![C_{\text{bu}}]\!]\,(b)\ .$$

We can eventually certify the lower bound $l_\infty$ by pushing $M$ ad infinitum:

$$\mathrm{wp}[\![C_{\text{bu}}]\!]\,(b) \;\succeq\; \lim_{M\to\infty} l_M \;=\; [n < 0] \cdot b + [n \ge 0] \cdot (b + 2 \cdot n) \;=\; l_\infty\ . \qquad \lhd$$

### 6.3 Tightness of the Guard-Strengthening Rule

Next, we discuss the error incurred by our guard-strengthening rule, i.e., the difference between $\mathrm{wp}[\![C_{\text{loop}}]\!]\,(f)$ and the established lower bound $l$ as per Thm. 14. To this end, observe that

$$\mathrm{wp}[\![C_{\text{loop}}]\!]\,(f) - l \;=\; \underbrace{\Big(\mathrm{wp}[\![C_{\text{loop}}]\!]\,(f) - \mathrm{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f)\Big)}_{\text{error incurred by guard strengthening, } \succeq 0} + \underbrace{\Big(\mathrm{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f) - l\Big)}_{\text{error incurred in the reduced problem, } \succeq 0} \tag{16}$$

where, on the right-hand side, the first summand encodes the error inherently caused by our guard-strengthening technique, whereas the second summand accounts for the error in under-approximating $\mathrm{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f)$ by $l$, which is independent of the proof rule itself. In what follows, we quantify the first summand, i.e., the error incurred by strengthening the guard.

**Lemma 20.** *Given loops* $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$ *and* $C'_{\text{loop}} = \texttt{while}\,(\,\varphi'\,)\,\{\,C\,\}$, *suppose* $\varphi' \implies \varphi$, *then, for any postexpectation* $f \in \mathbb{E}$,

$$\mathrm{wp}[\![C_{\text{loop}}]\!]\,(f) - \mathrm{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f) \;\preceq\; \mathrm{wp}[\![C'_{\text{loop}}]\!]\,([\varphi]) \cdot \sup_{\sigma\models\neg\varphi'\wedge\varphi} \mathrm{wp}[\![C_{\text{loop}}]\!]\,(f)\,(\sigma)\ .$$

Proof (sketch). As shown in the proof of Cor. 13 (cf. [Feng et al. 2023, Appx. C.2]), we have

$$\mathrm{wp}[\![C_{\text{loop}}]\!]\,(f) - \mathrm{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f) \;=\; \lambda\sigma.\ \int_A f_{C_{\text{loop}}}\ \mathrm{d}\,({}^{\sigma}\mathbb{P})\ .$$

It thus suffices to bound the integral above, which can be achieved by examining different fragments (witnessed by guard violations) of traces in $A$. See details in [Feng et al. 2023, Appx. C.6]. □

An immediate consequence of Lem. 20 yields the *tightness* of our guard-strengthening rule, that is, the error incurred by underapproximating $\mathrm{wp}[\![C_{\text{loop}}]\!]\,(f)$ by $\mathrm{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f)$ approaches zero when $\varphi'$ approaches $\varphi$ in an appropriate manner:

**Theorem 21 (Tightness of Guard Strengthening).** *Given* $C_{\text{loop}} = \texttt{while}\,(\,\varphi\,)\,\{\,C\,\}$, *suppose there exists a sequence of guards* $\{\varphi^m\}_{m\in\mathbb{N}}$ *such that* $\forall m \in \mathbb{N}\colon \varphi^m \implies \varphi$. *Let* $C_{\text{loop}}^m = \texttt{while}\,(\,\varphi^m\,)\,\{\,C\,\}$. *If one of the following two conditions holds:*

(a) $\lim_{m\to\infty} \sup_{\sigma\models\neg\varphi^m\wedge\varphi} \mathrm{wp}[\![C_{\text{loop}}]\!]\,(f)\,(\sigma) < \infty$ and $\lim_{m\to\infty} \mathrm{wp}[\![C_{\text{loop}}^m]\!]\,([\varphi]) = 0$,

(b) $\lim_{m\to\infty} \sup_{\sigma\models\neg\varphi^m\wedge\varphi} \mathrm{wp}[\![C_{\text{loop}}]\!]\,(f)\,(\sigma) = 0$,[12]

[12]Note that $\mathrm{wp}[\![C_{\text{loop}}^m]\!]\,([\varphi])$ maps a state to a *probability*, and thus $\lim_{m\to\infty} \mathrm{wp}[\![C_{\text{loop}}^m]\!]\,([\varphi]) \preceq 1 \ll \infty$.

*then, the error incurred by strengthening $\varphi$ with $\varphi^m$ converges to zero as $m$ tends to $\infty$, i.e.,*

$$\mathrm{wp}[\![C_{\mathrm{loop}}]\!]\,(f) \;=\; \lim_{m\to\infty} \mathrm{wp}[\![C^m_{\mathrm{loop}}]\!]\,([\neg\varphi]\cdot f)\ .$$

**Remark.** There are cases where both conditions (a) and (b) in Thm. 21 do *not* hold: Recall the 1-D random walk given in Sect. 2 and consider, for instance, the sequence of guards $\{\varphi^m\}_{m\in\mathbb{N}}$ with $\varphi^m = (0 < n < 9 + {}^{(2^m-1)}/_{2^m})$. We have that $\lim_{m\to\infty}\varphi^m = (0 < n < 10)$ and $\forall m \in \mathbb{N}\colon \varphi^m \implies \varphi$. However, one can verify that neither (a) nor (b) holds (as $\{\varphi^m\}_{m\in\mathbb{N}}$ do not approach $\varphi$ at all). ◁

Conditions (a) and (b) in Thm. 21 can be checked by certifying upper bounds on the weakest preexpectations therein via, e.g., Park induction, as demonstrated by the following example.

**Example 22.** Reconsider the probabilistic loops $C_{\mathrm{1dbrw}}$ and $C^M_{\mathrm{1dbrw}}$ with postexpectation $f = 1$ in Exmp. 16. We have shown – with the aid of Park induction – that $l_\infty = [n < 0] + [n \geq 0]\cdot({}^1/_2)^n$ coincides with the exact termination probability of $C_{\mathrm{1dbrw}}$ starting from initial position $n$, i.e., $\mathrm{wp}[\![C_{\mathrm{1dbrw}}]\!]\,(1) = l_\infty \lessdot \infty$. Thus, by (16), one can already conclude the tightness, namely,

$$\mathrm{wp}[\![C_{\mathrm{1dbrw}}]\!]\,(1) \;=\; \lim_{M\to\infty} \mathrm{wp}[\![C^M_{\mathrm{1dbrw}}]\!]\,([n \leq 0]\cdot 1)\ .$$

We now show that the same conclusion can be drawn via condition (b) in Thm. 21. Observe that

$$\begin{aligned}
\lim_{M\to\infty} \sup_{\sigma\models\neg\varphi^M\wedge\varphi} \mathrm{wp}[\![C_{\mathrm{1dbrw}}]\!]\,(1)\,(\sigma) \;&=\; \lim_{M\to\infty} \sup_{\sigma\models n\geq M} \mathrm{wp}[\![C_{\mathrm{1dbrw}}]\!]\,(1)\,(\sigma) \\
&\leq\; \lim_{M\to\infty} \sup_{\sigma\models n\geq M} ([n<0] + [n\geq 0]\cdot({}^1/_2)^n)\,(\sigma) \\
&\qquad\qquad [\text{by } \mathrm{wp}[\![C_{\mathrm{1dbrw}}]\!]\,(1) \preceq l_\infty] \\
&=\; \lim_{M\to\infty} \sup_{\sigma\models n\geq M} (({}^1/_2)^n)\,(\sigma) \\
&=\; \lim_{M\to\infty} ({}^1/_2)^M \;=\; 0\ .
\end{aligned}$$

Thus, by condition (b) in Thm. 21, the error incurred by strengthening $\varphi$ with $\varphi^M$ converges to zero as $M$ tends to $\infty$. ◁

### 6.4 On Automation of the Guard-Strengthening Rule

We briefly discuss the potential to *automate* our guard-strengthening rule for *finding* – rather than just verifying – lower bounds on weakest preexpectations. Such automation needs to address two questions: (i) how to find a "good" guard strengthening, i.e., $\varphi' \implies \varphi$? (ii) how to generate a non-trivial lower bound for the modified loop $C'_{\mathrm{loop}}$?

*How to Find a "Good" Strengthening?* In terms of logical strength, a good modified guard $\varphi'$ cannot be too strong, otherwise only trivial lower bound can be derived (think of $\varphi' = \texttt{false}$); it cannot be too weak either, otherwise the resulting loop $C'_{\mathrm{loop}}$ may not fulfil our conditions on termination (cf. Lem. 15) and uniform integrability (cf. Thm. 9 and 18). In general, it is hard to find – or even to characterize– the "best" strengthening $\varphi' \implies \varphi$, however, a naive strengthening pattern by simply bounding a subset $\mathcal{V} \subseteq \texttt{Vars}$ of program variables turns out to be rather effective, i.e.,

$$\varphi' \;=\; \varphi \wedge \bigwedge\nolimits_{x\in\mathcal{V}} x \diamond c\ ,$$

where $c$ is a constant and $\diamond \in \{<, >, \leq, \geq, =\}$. The subset $\mathcal{V} \subseteq \texttt{Vars}$ shall be selected in a way such that $C'_{\mathrm{loop}}$ meets our conditions on termination and uniform integrability. Such a strengthening pattern is indeed heuristic, but it suffices to produce good strengthened guards $\varphi'$ (in the sense discussed above) for *all* the examples presented in this paper – except for those in Sect. 8 where we

address limitations of the naive strengthening pattern. Investigating a potentially more advanced and clever guard-strengthening strategy is subject to future work.

*How to Generate a Non-trivial Lower Bound for $C'_{\text{loop}}$?* Due to the nice properties of $C'_{\text{loop}}$ on, e.g., termination, the desired lower bounds on $\text{wp}[\![C'_{\text{loop}}]\!]\,([\neg\varphi]\cdot f)$ can be discovered by leveraging various (semi-)automated techniques for synthesizing lower bounds on least fixed points. These include – in the probabilistic setting – techniques based on *constraint solving* [Chen et al. 2015; Feng et al. 2017; Katoen et al. 2010] and *invariant learning* [Bao et al. 2022] for generating lower bounds on weakest preexpectations of probabilistic programs, *recurrence solving* [Bartocci et al. 2019] for synthesizing moment-based invariants of (solvable) probabilistic loops, and various forms of *value iteration* [Baier et al. 2017; Hartmanns and Kaminski 2020; Quatmann and Katoen 2018] for determining reachability probabilities in Markov models. In particular, we identify a special case where computing the *exact* least fixed point can be done by *probabilistic model checking*:

**Theorem 23 (Reduction to Probabilistic Model Checking** [Baier and Katoen 2008, Chap. 10]**).** *Given $C'_{\text{loop}} = \texttt{while}\,(\,\varphi'\,)\,\{\,C\,\}$ and postexpectation $g \in \mathbb{E}$, if $C'_{\text{loop}}$ has a finite set of reachable states, then $\text{wp}[\![C'_{\text{loop}}]\!]\,(g)$ can be computed exactly by solving a system of linear equations.*

Thm. 23 holds since a finite-state loop $C'_{\text{loop}}$ can be unrolled into a finite-state Markov chain $\mathcal{M}$, and $\text{wp}[\![C'_{\text{loop}}]\!]\,(g)$ coincides with the unique solution to a linear-equation system induced by $\mathcal{M}$ with rewards modelling $g$ [Baier and Katoen 2008, Chap. 10], which can be computed symbolically by probabilistic model checkers, e.g., Storm [Dehnert et al. 2017], PRISM [Kwiatkowska et al. 2002].

*The combination of our guard-strengthening rule with Thm. 23 facilitates inferring quantitative properties of infinite-state probabilistic programs by model checking finite-state probabilistic models:*

**Example 24 (3-D Symmetric Random Walk on $\mathbb{Z}^3$** [McCrea and Whipple 1940; Montroll 1956]**).** Consider the loop $C_{\text{3dsrw}}$ modelling a symmetric random walk on the 3-D lattice over $\mathbb{Z}^3$:

$$C_{\text{3dsrw}}\colon\quad \texttt{while}\,(\,x\neq 0\,\vee\,y\neq 0\,\vee\,z\neq 0\,)\,\{\\ x\coloneqq x-1\;\oplus\;x\coloneqq x+1\;\oplus\;y\coloneqq y-1\;\oplus\;y\coloneqq y+1\;\oplus\;z\coloneqq z-1\;\oplus\;z\coloneqq z+1\,\}$$

where iterated $\oplus$ is shorthand for discrete uniform choice (in this case, with probability $1/6$ each).

The random nature underneath $C_{\text{3dsrw}}$ is fundamentally *different* from its 1- and 2-D counterparts: Pólya [1921] proved that the probability $\mathcal{P}$ that such a random walk returns to the origin at $(0,0,0)$ is strictly less than 1, indicating that $C_{\text{3dsrw}}$ does *not* terminate almost-surely. More precisely, the termination probability of $C_{\text{3dsrw}}$ starting from any neighbor location of the origin is

$$\mathcal{P}\;=\;1-\left(\frac{3}{(2\pi)^3}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\frac{\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z}{3-\cos x-\cos y-\cos z}\right)^{-1}\;=\;0.3405373296\ldots$$

which is known as one of Pólya's random walk constants. To the best of our knowledge, existing techniques cannot cope with the verification of $C_{\text{3dsrw}}$ due to its complex nature of divergence.

We now show how our guard-strengthening rule can be used to *automatically* derive lower bounds on the termination probability $\mathcal{P}$ of $C_{\text{3dsrw}}$ (i.e., with postexpectation $f=1$) by leveraging probabilistic model checking. To this end, we first bound the possible positions $(x,y,z)$ with a cube of side-length $2\cdot M$ with fixed $M\in\mathbb{Z}_{>0}$, and thereby obtain the modified loop:

$$C^{M}_{\text{3dsrw}}\colon\quad \texttt{while}\,(\,(x\neq 0\,\vee\,y\neq 0\,\vee\,z\neq 0)\,\wedge\,|x|<M\,\wedge\,|y|<M\,\wedge\,|z|<M\,)\,\{\\ x\coloneqq x-1\;\oplus\;x\coloneqq x+1\;\oplus\;y\coloneqq y-1\;\oplus\;y\coloneqq y+1\;\oplus\;z\coloneqq z-1\;\oplus\;z\coloneqq z+1\,\}\;.$$

Observe that, starting from any initial position in the lattice, the modified random walk $C^M_{\text{3dsrw}}$ can reach only a *finite* number of positions, i.e., it either (i) terminates by escaping the cube or returning to the origin, or (ii) keeps strolling within the cube. Thus, by the connection in Thm. 23, the reduced weakest preexpectation $\text{wp}[\![C^M_{\text{3dsrw}}]\!]\,([x = 0 \land y = 0 \land z = 0] \cdot 1)$ – which underapproximates the termination probability $\mathcal{P}$ of $C_{\text{3dsrw}}$ – can be computed symbolically by off-the-shelf probabilistic model checkers. For instance, Fig. 6 depicts such underapproximations produced by PRISM [Kwiatkowska et al. 2002] under different choices of $M$ (cf. the $X$-axis). In particular, Fig. 6 confirms our previous claim on tightness (cf. Thm. 21)[13]: as $M$ gets larger, the strengthening gets weaker, thus yielding tighter lower bounds on the actual termination probability $\mathcal{P}$. Fig. 6 also confirms our intuition that the further the walker is away from the origin, the lower probability returns the walker back to the origin. ◁

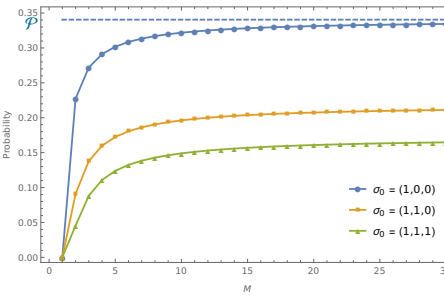


Fig. 6. Lower bounds obtained by PRISM on the termination probability of $C_{\text{3dsrw}}$ in Exmp. 24 on initial position $\sigma_0$. $\mathcal{P} \approx 0.34$ marks the true termination probability when starting from a neighbor location of the origin, e.g., $\sigma_0 = (1, 0, 0)$; the true termination probabilities on other initial positions are unknown.

**Remark.** In the aforementioned special case where our problem is reducible to model checking finite-state discrete-time Markov chains, our guard-strengthening rule acts analogously (yet on the source-code level) to the so-called *partial exploration* technique [Brázdil et al. 2014] established in probabilistic model checking. The latter constructs a sequence of increasingly precise approximations (lower and upper bounds on reachability probabilities) by (partially) exploring an incremental subset of the state space. This procedure yields a similar effect as we gradually weaken our strengthening to obtain tighter and tighter lower bounds. Since we always infer safe underapproximations, our technique may be used to generate *critical subsystems* [Ábrahám et al. 2014] serving as counterexamples to quantitative properties in probabilistic verification.

## 7 CASE STUDIES

In this section, we present a few non-trivial examples to demonstrate the effectiveness of our guard-strengthening rule in establishing lower bounds on weakest preexpectations. These examples include probabilistic programs with *continuous distributions*, *nested loops*, and *state-dependent probabilities*, as well as a real-world *randomized networking protocol*. These are all *infinite-state* programs, for some of which our proof rule can be used to prove almost-sure termination by certifying 1 as a lower bound on the termination probability.

**Example 25 (Continuous Sampling).** Consider the following loop $C_{\text{cs}}$ modelling a 1-D random walk where the branching probability is drawn form a continuous uniform distribution over $(0, 1)$.

$$
\begin{aligned}
C_{\text{cs}}\colon\quad & \texttt{while}\,(\,n > 0\,)\,\{\\
& \quad p :\approx \texttt{uniform}\,(0,1)\ \fatsemi\\
& \quad \{\,n := n - 1 \fatsemi b := b + \texttt{uniform}\,(0,1)\,\}\ [p]\ \{\,n := n + 1 \fatsemi b := b - \texttt{uniform}\,(0,1)\,\}\,\}\,.
\end{aligned}
$$

We aim to verify $l_\infty = [n < 0] \cdot b + [n \geq 0] \cdot (b + 1/2 \cdot n)$ as a lower bound on the expected value of $b$ upon termination, i.e., $l_\infty \preceq \text{wp}[\![C_{\text{cs}}]\!]\,(b)$. Observe that $l_\infty$ is unbounded; moreover, $C_{\text{cs}}$ behaves – in

[13] A rigorous proof of tightness in this example requires finding an appropriate superinvariant – like we did in Exmp. 22, yet now for the 3-D random walk – which is extremely hard and beyond the scope of this paper.

expectation – as the standard 1-D symmetric random walk [McIver et al. 2018] and thus terminates almost-surely yet with an *infinite* expected looping time. Existing proof rules hence do not apply.

We now strengthen the guard $(n > 0)$ of $C_{\text{cs}}$ to $\varphi^M = (0 < n < M)$ for any fixed $M \in \mathbb{Z}_{>0}$, and denote the so-obtained loop by $C_{\text{cs}}^M$. It can be shown (cf. [Feng et al. 2023, Appx. D.4]) that

$$l_M \;=\; [n < 0] \cdot b + [0 \le n \le M] \cdot (b + 1/2 \cdot n) \cdot (1 - n/M)$$

is a subinvariant, i.e., $l_M \preceq \Phi_g^M(l_M)$, where $g = [n \le 0] \cdot b$ is the restricted postexpectation, and $\Phi_g^M$ is the characteristic function of $C_{\text{cs}}^M$ with respect to $g$. Furthermore, $C_{\text{cs}}^M$ terminates after at most $M-1$ steps of consecutively increasing $n$ with probability at least $(1/2 \cdot 1/2)^{M-1}$ (i.e., by first drawing $p \in (1/2, 1)$ and then picking the right branch), we have ${}^{\sigma}\mathbb{P}\big(T^{\neg\varphi^M} > M\big) < 1 - (1/4)^{M-1}$ for any initial state $\sigma(n) \in \mathbb{Z}$. Thus, by Lem. 15, $C_{\text{cs}}^M$ terminates almost-surely. Moreover, $C_{\text{cs}}^M$ is uniformly integrable for $C_{\text{cs}}^M$ as witnessed by our new sufficient condition in Thm. 18. It follows that $l_M \preceq \text{wp}[\![C_{\text{cs}}^M]\!]\,([n \le 0] \cdot b) \preceq \text{wp}[\![C_{\text{cs}}]\!]\,(b)$ for any fixed $M \in \mathbb{Z}_{>0}$. We can eventually certify the lower bound $l_\infty$ by pushing $M$ ad infinitum:

$$\text{wp}[\![C_{\text{cs}}]\!]\,(b) \;\ge\; \lim_{M\to\infty} l_M \;=\; [n < 0] \cdot b + [n \ge 0] \cdot (b + 1/2 \cdot n) \;=\; l_\infty\,. \qquad \lhd$$

**Example 26 (Nested Loops).** Consider the program $C_{\text{nl}}$ with two nested loops (the outer loop alters $n \in \mathbb{Z}$ based on the outcome of the inner loop; $a, b \in \mathbb{Z}_{>0}$ are parameters satisfying $a > b$[14]):

$$\begin{aligned} C_{\text{nl}}\colon\quad & \texttt{while}\,(\,n > 0\,)\,\{\\ & \quad k \coloneqq n\,;\\ & \quad \texttt{while}\,(\,-a < k - n < b\,)\,\{\,k \coloneqq k - 1\;[1/2]\;k \coloneqq k + 1\,\}\,;\\ & \quad \texttt{if}\;(\,k < n\,)\;\{\,n \coloneqq n - 1\,\}\;\texttt{else}\;\{\,n \coloneqq n + 1\,\}\,\}\,. \end{aligned}$$

Suppose we are interested in proving a non-trivial lower bound on the termination probability $\text{wp}[\![C_{\text{nl}}]\!]\,(1)$. This problem is out-of-reach by existing proof rules as $C_{\text{nl}}$ is not AST. To apply our proof rule, we strengthen the guard of the outer loop to $\varphi^M = (0 < n < M)$ for any fixed $M \in \mathbb{Z}_{>0}$. The modified program $C_{\text{nl}}^M$ is AST due to a similar argument per Lem. 15. Let

$$l_M \;=\; [n < 0] + [0 \le n \le M] \cdot \Big((b/a)^n - (b/a)^M\Big)\,.$$

Since $l_M$ is bounded, we know by Thm. 8 and 14 that $l_M$ underapproximates the termination probability of $C_{\text{nl}}$ if $l_M$ is a subinvariant of $C_{\text{nl}}^M$. In that case, we have proved the lower bound $l_\infty$:

$$\text{wp}[\![C_{\text{nl}}]\!]\,(1) \;\ge\; \lim_{M\to\infty} l_M \;=\; [n < 0] + [n \ge 0] \cdot (b/a)^n \;\triangleq\; l_\infty\,.$$

We now show that $l_M$ is indeed a subinvariant of $C_{\text{nl}}^M$, i.e., $l_M \preceq \Phi_g^M(l_M)$, where $g = [n \le 0] \cdot 1$ is the restricted postexpectation, and $\Phi_g^M$ is the characteristic function of $C_{\text{nl}}^M$ w.r.t. $g$. Checking subinvariance in this case is more involved due to the nested feature of $C_{\text{nl}}^M$: one also has to reason about *lower bounds for the inner loop* w.r.t. a specific postexpectation depending on $l_M$. More precisely, let $C_{\text{body}}^M$ and $C_{\text{inner}}^M$ denote the loop body and the inner loop of $C_{\text{nl}}^M$, respectively. We have

$$\begin{aligned} \Phi_g^M\,(l_M) \;&=\; [n \le 0 \vee n \ge M] \cdot [n \le 0] + [0 < n < M] \cdot \text{wp}[\![C_{\text{body}}^M]\!]\,(l_M)\\ &=\; [n \le 0] + [0 < n < M] \cdot\\ &\quad\; \text{wp}[\![k \coloneqq n]\!]\,\Big(\text{wp}[\![C_{\text{inner}}^M]\!]\,\big([k < n] \cdot l_M\,[n/n-1] + [k \ge n] \cdot l_M\,[n/n+1]\big)\Big)\,. \end{aligned}$$

[14]The restriction $a > b$ makes $C_{\text{nl}}$ a non-AST program.

By combining the fact (see [Feng et al. 2023, Appx. D.5]) that, for any bounded postexpectation $h$,

$$\mathsf{wp}[\![C^M_{\text{inner}}]\!](h) \;\geq\; [(k-n<-a)\vee(k-n>b)]\cdot h + [-a\leq k-n\leq b]\cdot\left(\frac{b-(k-n)}{b+a}\cdot h\,[k/n-a]+\frac{(k-n)+a}{b+a}\cdot h\,[k/n+b]\right),$$

we can conclude the subinvariance of $l_M$, i.e., $l_M \preceq \Phi_g^M(l_M)$ (detailed in [Feng et al. 2023, Appx. D.5]). This completes our verification of the lower bound $l_\infty$. ◁

**Example 27 (Fair-in-the-Limit Random Walk [McIver et al. 2018]).** Consider the following loop $C_{\text{flrw}}$ modelling a biased random walk on $\mathbb{Z}$, where in particular, the probability of taking the next move depends on the current position of the walker:[15]

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

The further the walker moves to the right, the more fair becomes the random walk as ${}^{n+1}/_{2\cdot n+1}$ approaches $1/2$ asymptotically. We show how our guard-strengthening rule can be used to prove almost-surely termination of $C_{\text{flrw}}$ by certifying 1 as a lower bound on $\mathsf{wp}[\![C_{\text{flrw}}]\!](1)$. Similarly, we strengthen the guard $(n>0)$ to $\varphi^M = (0<n<M)$ for any fixed $M\in\mathbb{Z}_{>0}$ and denote the so-obtained loop by $C^M_{\text{flrw}}$. Since $C^M_{\text{flrw}}$ terminates after at most $M-1$ steps of consecutively moving to the left with probability at least $(1/2)^{M-1}$, $C^M_{\text{flrw}}$ terminates almost-surely by Lem. 15. Let

$$l_M \;=\; [n<0]+[0\leq n\leq M]\cdot(1-{}^{n}/_{M})\;.$$

$l_M$ is bounded from above by 1, thus is uniformly integrable for $C^M_{\text{flrw}}$ by condition (c) in Thm. 9. Furthermore, $l_M$ is a subinvariant since

$$\begin{aligned}\Phi_g(l_M) &= [n\leq 0\vee n\geq M]\cdot[n\leq 0]+[0<n<M]\cdot\left(\frac{n+1}{2n+1}\cdot l_M\,[n/n-1]+\frac{n}{2n+1}\cdot l_M\,[n/n+1]\right)\\ &= [n\leq 0\vee n\geq M]\cdot[n\leq 0]+[0<n<M]\cdot\left(\frac{n+1}{2n+1}\cdot\frac{M-n+1}{M}+\frac{n}{2n+1}\cdot\frac{M-n-1}{M}\right)\\ &= [n\leq 0]+[0<n<M]\cdot\left(1-\frac{n}{M}+\frac{1}{(2n+1)\cdot M}\right)\;\geq\; l_M\,,\end{aligned}$$

where $g=[n\leq 0]\cdot 1$ is the restricted postexpectation, and $\Phi_g^M$ is the characteristic function of $C^M_{\text{flrw}}$ with respect to $g$. This implies $l_M \preceq \mathsf{wp}[\![C_{\text{flrw}}]\!](1)$, i.e., $l_M$ is a lower bound on the termination probability of $C_{\text{flrw}}$ for any fixed $M\in\mathbb{Z}_{>0}$. In particular, when $M$ approaches infinity, we have

$$\mathsf{wp}[\![C_{\text{flrw}}]\!](1)\;\geq\;\lim_{M\to\infty} l_M \;=\; 1\,.$$

We can thus conclude that $C_{\text{flrw}}$ terminates almost-surely. ◁

**Example 28 (Zeroconf Protocol [Bohnenkamp et al. 2003]).** Consider the loop $C_{\text{zc}}$ encoding the randomized IPv4 Zeroconf protocol (parameterized in $N$) for self-configuring IP network interfaces:

$$\begin{aligned}C_{\text{zc}}\colon\quad & \mathit{start}=1\,;\,\mathit{established}=0\,;\,\mathit{probe}=0\,;\\ & \texttt{while}\,(\,\mathit{start}\leq 1\wedge\mathit{established}\leq 0\wedge\mathit{probe}<N\wedge N\geq 4\,)\,\{\\ & \quad\texttt{if}\,(\,\mathit{start}=1\,)\,\{\\ & \quad\quad\{\,\mathit{start}\coloneqq 0\,\}\;[0.5]\;\{\,\mathit{start}\coloneqq 0\,;\,\mathit{established}\coloneqq 1\,\}\,\}\\ & \quad\texttt{else}\,\{\,\{\,\mathit{probe}\coloneqq\mathit{probe}+1\,\}\;[0.001]\;\{\,\mathit{start}\coloneqq 1\,;\,\mathit{probe}\coloneqq 0\,\}\,\}\,\}\;.\end{aligned}$$

[15] The loop remains AST if we replace the biased probability ${}^{n+1}/_{2n+1}$ by ${}^{n}/_{2n+1}$. However, in this case, one needs a more advanced subinvariant in order to certify AST of the loop. See, e.g., a proof using harmonic numbers in [McIver et al. 2018].

This program is intended for providing a convenient way to analyze the probability that an unused IP address is successfully assigned to a host that is newly connected to a network of $m$ existing devices: Once being connected, the host first randomly selects an IP address from a pool of $n$ available addresses. With probability $m/n$ (instantiated as 0.5), this address is already in use; with probability $1 - m/n$, the chosen address is unused and thus the IP connection can be established (signified by terminating with *established* = 1). For the former case, the host broadcasts a "probe" message to other devices in the network asking whether the chosen IP address is already taken; If the probe is received by a device that already uses the address, it replies with a message indicating so. After receiving this message, the host to be configured restarts.[16] The probability 0.001 in the `else`-statement encodes the probability of message loss (for probe or reply). To increase reliability, the host is required to send multiple probes (upper-bounded by $N$). In fact, $C_{\text{zc}}$ encodes an *infinite family* of Markov chains (cf. [Baier and Katoen 2008, Exmp. 10.5]) by parameterizing $N$, and thus cannot be handled by probabilistic model checking.

We are interested in verifying a non-trivial lower bound 0.99999999999 on the probability $\mathcal{P}$ that, starting from a state satisfying the loop guard, $C_{\text{zc}}$ terminates in a state with *established* = 1. We do so by trying to *synthesize* a subinvariant justifying this lower bound using the recently developed tool CEGISPRO2 [Batz et al. 2023] which automates Hark et al.'s sound lower induction rule [Hark et al. 2020]. Unfortunately, CEGISPRO2 failed in this case (timeout in 20 minutes). However, by strengthening the loop guard with upper bounds on $N$, e.g., $N \leq 10$, and applying our proof rule, CEGISPRO2 found a piece-wise linear subinvariant (in less than a second; omitted due to limited space) that suffices to certify 0.99999999999 as a lower bound on $\mathcal{P}$. Overall, this demonstrates the potential of our proof rule to enable verifying lower bounds via automated techniques for practical randomized algorithms that are otherwise out-of-reach. ◁

Our proof rule applies further to a few extra case studies from the Quantitative Verification Benchmark Set [Hartmanns et al. 2019], including the (infinite family of) Bounded Retransmission protocol, the (parametrized) Rabin Mutual Exclusion algorithm, and the Coupon Collectors. These case studies are not included because they either do not bring extra insights over the Zeroconf protocol or can already be tackled by existing techniques. The main challenges of pursuing a wider range of real randomized algorithms are (i) to find good guard strengthening (cf. Sect. 8 below) and, in some cases, (ii) to find subinvariants certifying the given lower bounds. We plan to address these challenges by exploiting the potential for *automating* our proof rule as discussed in Sect. 6.4.

## 8 LIMITATIONS OF THE GUARD-STRENGTHENING RULE

There are interesting divergent programs for which the proposed guard-strengthening principle is insufficient or inadequate. We address such limitations by means of two examples.

The following example demonstrates the case where the naive strengthening heuristic is insufficient, yet a more advanced strengthening strategy suffices to establish non-trivial lower bounds.

**Example 29 (2-D Discrete Spiral).** Consider the probabilistic program

$$
\begin{array}{l}
x \coloneqq 0 \,⨟\, y \coloneqq 0 \,⨟\, \mathit{flag} \coloneqq 0 \,⨟ \\
\texttt{while}\,\big(\,(x-1)^2 + (y-1)^2 < 4\,\big)\,\{ \\
\quad \{\, x \coloneqq -1 \,⨟\, y \coloneqq -1 \,\}\; [\; {}^{|x-1|}\!/_{4} \;]\; \{ \\
\qquad \texttt{if}\;(\mathit{flag})\;\{\, y \coloneqq {}^{(3-y)}\!/_{2} \,\}\;\texttt{else}\;\{\, x \coloneqq {}^{(3-x)}\!/_{2} \,\} \,⨟\, \mathit{flag} \coloneqq 1 - \mathit{flag} \,\}\,\}\,.
\end{array}
$$

[16] The program $C_{\text{zc}}$ is an abstract model of the actual protocol in the sense that $C_{\text{zc}}$ abstracts away the need for broadcasting probe messages when the right branch of the `if`-statement is executed (in this case, no reply message will be sent anyway). Such an abstraction does not affect the analysis of the probability of terminating with *established* = 1.

This program terminates only if it visits the left branch of the probabilistic choice; otherwise, it diverges with $(x, y)$ approaching $(1, 1)$ in the shape of a 2-D discrete spiral. Note that the program is non-AST and we aim to verify lower bounds on its termination probability. Observe that, in this case, the naive strengthening heuristic $\varphi' = \varphi \wedge x < c \wedge y < c$ by adding a constant bound $c$ on program variables is *insufficient* for any $c > 3/2$ as the resulting program is still non-AST; whereas for any $c \leq 3/2$, the resulting program terminates after at most one loop iteration thus only yielding *trivial* lower bounds. In contrast, by applying the more advanced (yet intuitive) strengthening $\varphi' = \varphi \wedge (x-1)^2 + (y-1)^2 > \epsilon$ with a small constant $\epsilon > 0$ (i.e., to rule out the spiral center $(1, 1)$), we reduce the program to the case of a finite state space, which can then be tackled by probabilistic model checking for establishing *non-trivial* lower bounds. ◁

As remarked in Sect. 6.4, it is an interesting future direction to investigate potentially more advanced, (semi-)automatable guard-strengthening strategies than the heuristic used in this paper.

The following example demonstrates a corner case where no useful strengthening exists.

**Example 30 (Dummy Swapper).** Consider the probabilistic program

$$\texttt{while}\,(\,x \neq y\,)\,\{\,x \coloneqq y \;\oplus\; \{\mathit{temp} \coloneqq x \fatsemi x \coloneqq y \fatsemi y \coloneqq \mathit{temp}\} \;\oplus\; \texttt{diverge}\,\}$$

where iterated $\oplus$ is shorthand for discrete uniform choice (in this case, with probability $1/3$ each); `diverge` is syntactic sugar for `while(true){skip}`. This program is non-AST and it terminates only by visiting the leftmost branch of the uniform choice; otherwise, it diverges by either keeping swapping the values of $x$ and $y$ (i.e., pacing between two points) or diverges immediately by visiting the rightmost branch. It can be shown that *every* possible strengthening that can turn this program AST necessarily leads to $\varphi' = \texttt{false}$ – otherwise, at least one loop iteration is executed and the program diverges with probability at least $1/3$ – and therefore only yield *trivial* lower bounds. ◁

## 9 RELATED WORK

*Weakest Preexpectation Reasoning.* As a probabilistic analog to Dijkstra's predicate-transformer calculus [Dijkstra 1975, 1976], *expectation transformers* have been extensively used to reason about *quantitative* properties of probabilistic programs. wp-style reasoning goes back to the seminal work of Kozen [1983, 1985] on probabilistic propositional dynamic logic. Amongst others, Jones [1990], Morgan et al. [1996], McIver and Morgan [2005], and Hehner [2011] furthered this line of research by considering, e.g., nondeterminism and proof rules for bounding preexpectations in the presence of loops. The classical weakest preexpectation calculus has been significantly advanced in several directions to reason about, e.g., *expected runtimes* of (recursive) probabilistic programs [Kaminski et al. 2016, 2018; Olmedo et al. 2016], probabilistic programs with *conditioning* [Olmedo et al. 2018; Szymczak and Katoen 2019], *mixed-sign* postexpectations [Kaminski and Katoen 2017], *sensitivity* of probabilistic programs [Aguirre et al. 2021], probabilistic temporal logic [Morgan and McIver 1999], and quantitative separation logic [Batz et al. 2019]. A relative completeness result has been recently established by Batz et al. [2021b] for weakest preexpectation reasoning.

*Bounds on Weakest Preexpectation and Fixed Points.* There are a wide spectrum of results on establishing upper and/or lower bounds on loop semantics captured by (least) fixed points. These include (semi-)automated techniques based on *metering functions* [Frohn et al. 2020, 2016] for underapproximating runtimes of non-probabilistic programs, *constraint solving* [Chen et al. 2015; Feng et al. 2017; Katoen et al. 2010] and *invariant learning* [Bao et al. 2022] for generating lower bounds on weakest preexpectations of probabilistic programs, *recurrence solving* [Bartocci et al. 2019] for synthesizing moment-based invariants of the so-called prob-solvable probabilistic loops, *bounded model checking* [Jansen et al. 2016] for verifying probabilistic programs with nondeterminism and

conditioning, and various forms of *value iteration* [Baier et al. 2017; Hartmanns and Kaminski 2020; Quatmann and Katoen 2018] for determining reachability probabilities in Markov models.

Apart from the proof rules elaborated in this paper, Baldan et al. [2021] recently proposed lattice-theoretic proof rules for verifying lower bounds on least fixed points of *finite-state* stochastic systems. Batz et al. [2021a] invented *latticed k-induction* for establishing upper bounds on least fixed points of possibly infinite-state probabilistic programs. A closely related concept to our guard-strengthening rule is the notion of *ω-subinvariant* [Audebaud and Paulin-Mohring 2009; Jones 1990; Kaminski et al. 2016, 2018], which is a monotonically increasing sequence $\{I_n\}_{n\in\mathbb{N}}$ of expectations that are subinvariants relative to each other, i.e., $I_0 = 0$ and $I_{n+1} \preceq \Phi(I_n)$. It is known that $\sup_{n\in\mathbb{N}} I_n$ suffices as a lower bound on the least fixed point of $\Phi$, however, finding such a lower bound amounts to (i) learning a closed form of $I_n$ in $n$ by inspecting the "pattern" through fixed point iterations, and (ii) finding the supremum of the closed form. These two steps may barely save efforts compared to just inferring the supremum (limit) of the sequence $\sup_{n\in\mathbb{N}} \Phi^n(0)$ to obtain the exact least fixed point. In some of our examples, we also reason about limits of $l_M$ w.r.t. $M$ – the constant used to strengthen the guard – to get tight lower bounds. This step, however, is *not necessary* since $l_M \preceq \Phi(l_M)$ suffices already as a lower bound for any fixed $M$, that is, we only need to "push $l_M$ through the loop semantics" *once*. Moreover, for discrete-state programs with complex closed form solutions, e.g., the 3-D random walk in Exmp. 24, our rule is capable of inferring tight lower bounds (via probabilistic model checking) *without* the need for taking limits.

*Space-Restricting in Analyzing Random Walks.* There is an intellectual connection between our guard-strengthening technique and the *space-restricting* tactic in analyzing random walks over infinite, discrete lattices: Restricting an infinite-space random walk à la Feller [1950, Chap. XIV] and McCrea and Whipple [1940] by setting "barriers" and then taking the limit is analogous to our approach. Nonetheless, they both proceed, after the restriction, by *solving the recurrence equation* (aka, difference equation) to obtain a closed-form solution for the restricted random walk in order to take the limit. Although a similar methodology can be embedded in our approach – as solving recurrences is one (expensive) way to obtain subinvariants – it is *not necessary*: our proof rule *reduces* the problem and places it under the lens of probabilistic model checking. The rule is thus able to handle examples that are out-of-reach by existing verification techniques, e.g., the 3-D random walk, *without* the need for recurrence solving nor taking limits. Moreover, our proof rule codifies *ad-hoc* proof methods in probability theory as a *syntactic* rule of a program logic, and enables the use of existing proof rules on verification problems that are otherwise out-of-scope.

*Martingale-Based Reasoning.* Probabilistic program analysis via martingales was pioneered by Chakarov and Sankaranarayanan [2013, 2014] for finding upper bounds on expected runtimes and synthesizing expectation invariants. Chatterjee et al. [2016b] further extended the martingale-based approach to address nondeterminism. Barthe et al. [2016] focus on synthesizing exact martingale expressions. Fioriti and Hermanns [2015] develop a type system for uniform integrability in order to prove AST of probabilistic programs and provide upper bounds on *expected runtimes*. In contrast, Fu and Chatterjee [2019] give lower bounds on expected runtimes; Chatterjee et al. [2022] prove lower bounds on *termination probabilities*. Kobayashi et al. [2020] provide a semi-decision procedure for upper-bounding termination probabilities of probabilistic higher-order recursive programs. Chatterjee et al. [2017]; Wang et al. [2021b] focus on finding upper and lower bounds on *assertion-violation probabilities*. Ngo et al. [2018] perform automated template-driven *resource* analysis, yet infer upper bounds only. Wang et al. [2019] provide sufficient conditions for finding upper and lower bounds on *expected costs*. Wang et al. [2021a] present conditions to derive upper and lower bounds on higher moments of *expected accumulated costs*. In contrast to our proof rule, most of these martingale-based methods for inferring lower bounds address only AST probabilistic programs.

*Probabilistic Bisimulation and Equivalence.* Our result on wp-difference falls in the general scope of quantifying the "distance" between two programs. This gives rise to the notions of *probabilistic bisimulation* [Hong et al. 2019; Larsen and Skou 1991] and *probabilistic equivalence* (see [Murawski and Ouaknine 2005] over *finite* data types and [Barthe et al. 2009] for *straight-line* programs). Chen et al. [2022] recently proved a decidability result for checking the equivalence of an *infinite-state* discrete, loopy probabilistic program with a loop-free specification program (assuming *AST*). Barthe et al. [2009, 2012] developed a *relational Hoare logic* for probabilistic programs, which has been extensively used for proving program equivalence with applications in provable security and side-channel analysis. Our proof rule is dedicated to reasoning about lower bounds on expected values of possibly divergent probabilistic programs, nevertheless, it is interesting to explore its generalization in proving equivalence, bisimulation, and $\varepsilon$*-closeness* of possibly *infinite-state*, *non-AST* probabilistic programs by, e.g., embedding our proof rule in the probabilistic relational Hoare logic.

## 10 CONCLUSION

We have presented a new proof rule – based on wp-difference and the guard-strengthening technique – for verifying lower bounds on weakest preexpectations of probabilistic programs. This is the first lower bound rule that admits *divergent* probabilistic loops with possibly *unbounded* expectations. In particular, we unleash existing lower induction rules for general applicability to possibly divergent programs whose strengthened counterparts feature easily provable almost-sure termination and uniform integrability. Moreover, we have shown that the error incurred by our guard-strengthening technique can be arbitrarily small thereby yielding tight lower bounds, and in case the modified loop has a finite state space, our proof rule can be automated to generate lower bounds via probabilistic model checking. The effectiveness of our proof rule has been demonstrated on several examples. In particular, we managed to infer tight lower bounds on the termination probability of the well-known 3-D random walk on a lattice, which has not been addressed yet in the context of verification.

Future directions include (i) extending our proof rule to the expected runtime calculus for lower-bounding expected runtimes [Kaminski et al. 2016, 2018], to probabilistic programs with angelic/demonic nondeterminism [McIver and Morgan 2001, 2005] and/or with soft/hard conditioning [Olmedo et al. 2018; Szymczak and Katoen 2019]; (ii) exploring the applicability of our result on wp-difference in the context of sensitivity analysis [Aguirre et al. 2021; Barthe et al. 2018; Wang et al. 2020] and model repair [Bartocci et al. 2011] for probabilistic programs; and (iii) investigating (semi-)automated synthesis of (candidate) quantitative (sub)invariants used in our proof rule.

## ACKNOWLEDGMENTS

This work has been partially funded by the National Key R&D Program of China under grant No. 2022YFA1005101, by the NSFC under grant No. 62192732 and 62032024, by the CAS Project for Young Scientists in Basic Research under grant No. YSBR-040, by the ZJU Education Foundation's Qizhen Talent program, by the ERC Advanced Project FRAPPANT under grant No. 787914, and by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 101008233. The authors would like to thank Kevin Batz, Tim Quatmann, and anonymous reviewers for the insightful discussions on the connection respectively to $\omega$-invariants in weakest preexpectation reasoning, partial exploration in Markov models, and the space-restricting tactic in analyzing random walks.

## REFERENCES

Erika Ábrahám, Bernd Becker, Christian Dehnert, Nils Jansen, Joost-Pieter Katoen, and Ralf Wimmer. 2014. Counterexample Generation for Discrete-Time Markov Models: An Introductory Survey. In *SFM (LNCS, Vol. 8483)*. Springer, 65–121.

Alejandro Aguirre, Gilles Barthe, Justin Hsu, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja. 2021. A Pre-Expectation Calculus for Probabilistic Sensitivity. *Proc. ACM Program. Lang.* 5, POPL (2021), 1–28.

James Aspnes and Maurice Herlihy. 1990. Fast Randomized Consensus Using Shared Memory. *J. Algorithms* 11, 3 (1990), 441–461.

Philippe Audebaud and Christine Paulin-Mohring. 2009. Proofs of Randomized Algorithms in Coq. *Sci. Comput. Program.* 74, 8 (2009), 568–589.

Christel Baier and Joost-Pieter Katoen. 2008. *Principles of Model Checking*. MIT press.

Christel Baier, Joachim Klein, Linda Leuschner, David Parker, and Sascha Wunderlich. 2017. Ensuring the Reliability of Your Model Checker: Interval Iteration for Markov Decision Processes. In *CAV (II) (LNCS, Vol. 10426)*. Springer, 160–180.

Paolo Baldan, Richard Eggert, Barbara König, and Tommaso Padoan. 2021. Fixpoint Theory – Upside Down. In *FoSSaCS (LNCS, Vol. 12650)*. Springer, 62–81.

Jialu Bao, Nitesh Trivedi, Drashti Pathak, Justin Hsu, and Subhajit Roy. 2022. Data-Driven Invariant Learning for Probabilistic Programs. In *CAV (I) (LNCS, Vol. 13371)*. Springer, 33–54.

Gilles Barthe, Thomas Espitau, Luis María Ferrer Fioriti, and Justin Hsu. 2016. Synthesizing Probabilistic Invariants via Doob's Decomposition. In *CAV (I) (LNCS, Vol. 9779)*. Springer, 43–61.

Gilles Barthe, Thomas Espitau, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2018. Proving Expected Sensitivity of Probabilistic Programs. *Proc. ACM Program. Lang.* 2, POPL (2018), 57:1–57:29.

Gilles Barthe, Benjamin Grégoire, and Santiago Zanella Béguelin. 2009. Formal Certification of Code-Based Cryptographic Proofs. In *POPL*. ACM, 90–101.

Gilles Barthe, Benjamin Grégoire, and Santiago Zanella Béguelin. 2012. Probabilistic Relational Hoare Logics for Computer-Aided Security Proofs. In *MPC (LNCS, Vol. 7342)*. Springer, 1–6.

Gilles Barthe, Joost-Pieter Katoen, and Alexandra Silva (Eds.). 2020. *Foundations of Probabilistic Programming*. Cambridge University Press.

Gilles Barthe, Boris Köpf, Federico Olmedo, and Santiago Zanella Béguelin. 2013. Probabilistic Relational Reasoning for Differential Privacy. *ACM Trans. Program. Lang. Syst.* 35, 3 (2013), 9:1–9:49.

Ezio Bartocci, Radu Grosu, Panagiotis Katsaros, C. R. Ramakrishnan, and Scott A. Smolka. 2011. Model Repair for Probabilistic Systems. In *TACAS (LNCS, Vol. 6605)*. Springer, 326–340.

Ezio Bartocci, Laura Kovács, and Miroslav Stankovic. 2019. Automatic Generation of Moment-Based Invariants for Prob-Solvable Loops. In *ATVA (LNCS, Vol. 11781)*. Springer, 255–276.

Kevin Batz, Mingshuai Chen, Sebastian Junges, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja. 2023. Probabilistic Program Verification via Inductive Synthesis of Inductive Invariants. In *TACAS*. To appear.

Kevin Batz, Mingshuai Chen, Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, and Philipp Schröer. 2021a. Latticed $k$-Induction with an Application to Probabilistic Programs. In *CAV (I) (LNCS, Vol. 12760)*. Springer, 524–549.

Kevin Batz, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja. 2021b. Relatively Complete Verification of Probabilistic Programs: An Expressive Language for Expectation-Based Reasoning. *Proc. ACM Program. Lang.* 5, POPL (2021), 1–30.

Kevin Batz, Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, and Thomas Noll. 2019. Quantitative Separation Logic: A Logic for Reasoning about Probabilistic Pointer Programs. *Proc. ACM Program. Lang.* 3, POPL (2019), 34:1–34:29.

Daniel Bernoulli. 1954. Exposition of a New Theory on the Measurement of Risk. *Econometrica* 22, 1 (1954), 23–36.

Henrik C. Bohnenkamp, Peter van der Stok, Holger Hermanns, and Frits W. Vaandrager. 2003. Cost-Optimization of the IPv4 Zeroconf Protocol. In *DSN*. IEEE Computer Society, 531–540.

Tomás Brázdil, Krishnendu Chatterjee, Martin Chmelik, Vojtech Forejt, Jan Kretínský, Marta Z. Kwiatkowska, David Parker, and Mateusz Ujma. 2014. Verification of Markov Decision Processes Using Learning Algorithms. In *ATVA (LNCS, Vol. 8837)*. Springer, 98–114.

Michael Carbin, Sasa Misailovic, and Martin C. Rinard. 2016. Verifying Quantitative Reliability for Programs that Execute on Unreliable Hardware. *Commun. ACM* 59, 8 (2016), 83–91.

Aleksandar Chakarov and Sriram Sankaranarayanan. 2013. Probabilistic Program Analysis with Martingales. In *CAV (LNCS, Vol. 8044)*. Springer, 511–526.

Aleksandar Chakarov and Sriram Sankaranarayanan. 2014. Expectation Invariants for Probabilistic Program Loops as Fixed Points. In *SAS (LNCS, Vol. 8723)*. Springer, 85–100.

Krishnendu Chatterjee, Hongfei Fu, and Amir Kafshdar Goharshady. 2016a. Termination Analysis of Probabilistic Programs Through Positivstellensatz's. In *CAV (I) (LNCS, Vol. 9779)*. Springer, 3–22.

Krishnendu Chatterjee, Hongfei Fu, and Petr Novotný. 2020. Termination Analysis of Probabilistic Programs with Martingales. In *Foundations of Probabilistic Programming*, Gilles Barthe, Joost-Pieter Katoen, and Alexandra Silva (Eds.). Cambridge University Press, 221–258.

Krishnendu Chatterjee, Hongfei Fu, Petr Novotný, and Rouzbeh Hasheminezhad. 2016b. Algorithmic Analysis of Qualitative and Quantitative Termination Problems for Affine Probabilistic Programs. In *POPL*. ACM, 327–342.

Krishnendu Chatterjee, Amir Kafshdar Goharshady, Tobias Meggendorfer, and Djordje Zikelic. 2022. Sound and Complete Certificates for Quantitative Termination Analysis of Probabilistic Programs. In *CAV (I) (LNCS, Vol. 13371)*. Springer, 55–78.

Krishnendu Chatterjee, Petr Novotný, and Dorde Zikelic. 2017. Stochastic Invariants for Probabilistic Termination. In *POPL*. ACM, 145–160.

Mingshuai Chen, Joost-Pieter Katoen, Lutz Klinkenberg, and Tobias Winkler. 2022. Does a Program Yield the Right Distribution? Verifying Probabilistic Programs via Generating Functions. In *CAV (I) (LNCS, Vol. 13371)*. Springer, 79–101.

Yu-Fang Chen, Chih-Duo Hong, Bow-Yaw Wang, and Lijun Zhang. 2015. Counterexample-Guided Polynomial Loop Invariant Generation by Lagrange Interpolation. In *CAV (I) (LNCS, Vol. 9206)*. Springer, 658–674.

Fredrik Dahlqvist, Alexandra Silva, and Dexter Kozen. 2020. Semantics of Probabilistic Programming: A Gentle Introduction. In *Foundations of Probabilistic Programming*, Gilles Barthe, Joost-Pieter Katoen, and Alexandra Silva (Eds.). Cambridge University Press, 1–42.

Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, and Matthias Volk. 2017. A Storm is Coming: A Modern Probabilistic Model Checker. In *CAV (II) (LNCS, Vol. 10427)*. Springer, 592–600.

Edsger Wybe Dijkstra. 1975. Guarded Commands, Nondeterminacy and Formal Derivation of Programs. *Commun. ACM* 18, 8 (1975), 453–457.

Edsger Wybe Dijkstra. 1976. *A Discipline of Programming*. Prentice-Hall.

Owain Evans, Andreas Stuhlmüller, John Salvatier, and Daniel Filan. 2017. Modeling Agents with Probabilistic Programs. http://agentmodels.org. Accessed: 2022-7-7.

Willliam Feller. 1950. *An Introduction to Probability Theory and Its Applications*. Vol. I. John Wiley & Sons.

Shenghua Feng, Mingshuai Chen, Han Su, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Naijun Zhan. 2023. Lower Bounds for Possibly Divergent Probabilistic Programs. *CoRR* abs/2302.06082 (2023). arXiv:2302.06082

Yijun Feng, Lijun Zhang, David N. Jansen, Naijun Zhan, and Bican Xia. 2017. Finding Polynomial Loop Invariants for Probabilistic Programs. In *ATVA (LNCS, Vol. 10482)*. Springer, 400–416.

Luis María Ferrer Fioriti and Holger Hermanns. 2015. Probabilistic Termination: Soundness, Completeness, and Compositionality. In *POPL*. ACM, 489–501.

Florian Frohn, Matthias Naaf, Marc Brockschmidt, and Jürgen Giesl. 2020. Inferring Lower Runtime Bounds for Integer Programs. *ACM Trans. Program. Lang. Syst.* 42, 3 (2020), 13:1–13:50.

Florian Frohn, Matthias Naaf, Jera Hensel, Marc Brockschmidt, and Jürgen Giesl. 2016. Lower Runtime Bounds for Integer Programs. In *IJCAR (LNCS, Vol. 9706)*. Springer, 550–567.

Hongfei Fu and Krishnendu Chatterjee. 2019. Termination of Nondeterministic Probabilistic Programs. In *VMCAI (LNCS, Vol. 11388)*. Springer, 468–490.

Jürgen Giesl, Peter Giesl, and Marcel Hark. 2019. Computing Expected Runtimes for Constant Probability Programs. In *CADE (LNCS, Vol. 11716)*. Springer, 269–286.

Andrew D. Gordon, Thomas A. Henzinger, Aditya V. Nori, and Sriram K. Rajamani. 2014. Probabilistic Programming. In *FOSE*. ACM, 167–181.

Marcel Hark, Benjamin Lucien Kaminski, Jürgen Giesl, and Joost-Pieter Katoen. 2019. Aiming Low Is Harder – Inductive Proof Rules for Lower Bounds on Weakest Preexpectations in Probabilistic Program Verification. *CoRR* abs/1904.01117 (2019). arXiv:1904.01117

Marcel Hark, Benjamin Lucien Kaminski, Jürgen Giesl, and Joost-Pieter Katoen. 2020. Aiming Low Is Harder: Induction for Lower Bounds in Probabilistic Program Verification. *Proc. ACM Program. Lang.* 4, POPL (2020), 37:1–37:28.

Arnd Hartmanns and Benjamin Lucien Kaminski. 2020. Optimistic Value Iteration. In *CAV (II) (LNCS, Vol. 12225)*. Springer, 488–511.

Arnd Hartmanns, Michaela Klauck, David Parker, Tim Quatmann, and Enno Ruijters. 2019. The Quantitative Verification Benchmark Set. In *TACAS (I) (LNCS, Vol. 11427)*. Springer, 344–350.

Eric Charles Roy Hehner. 2011. A Probability Perspective. *Formal Aspects Comput.* 23, 4 (2011), 391–419.

Michael Hicks. 2014. What is Probabilistic Programming? In: *The Programming Languages Enthusiast*. http://www.pl-enthusiast.net/2014/09/08. Accessed: 2021-12-09.

Chih-Duo Hong, Anthony W. Lin, Rupak Majumdar, and Philipp Rümmer. 2019. Probabilistic Bisimulation for Parameterized Systems - (with Applications to Verifying Anonymous Protocols). In *CAV (I) (LNCS, Vol. 11561)*. Springer, 455–474.

Jacek Jachymski, Leslaw Gajek, and Piotr Pokarowski. 2000. The Tarski-Kantorovitch Principle and the Theory of Iterated Function Systems. *Bulletin of the Australian Mathematical Society* 61, 2 (2000), 247–261.

Nils Jansen, Christian Dehnert, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Lukas Westhofen. 2016. Bounded Model Checking for Probabilistic Programs. In *ATVA (LNCS, Vol. 9938)*. 68–85.

Claire Jones. 1990. *Probabilistic Non-determinism*. Ph. D. Dissertation. University of Edinburgh, UK.

Benjamin Lucien Kaminski. 2019. *Advanced Weakest Precondition Calculi for Probabilistic Programs.* Ph. D. Dissertation. RWTH Aachen University, Germany.

Benjamin Lucien Kaminski and Joost-Pieter Katoen. 2017. A Weakest Pre-expectation Semantics for Mixed-Sign Expectations. In *LICS*. IEEE Computer Society, 1–12.

Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja. 2019. On the Hardness of Analyzing Probabilistic Programs. *Acta Informatica* 56, 3 (2019), 255–285.

Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, and Federico Olmedo. 2016. Weakest Precondition Reasoning for Expected Run-Times of Probabilistic Programs. In *ESOP (LNCS, Vol. 9632)*. Springer, 364–389.

Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, and Federico Olmedo. 2018. Weakest Precondition Reasoning for Expected Runtimes of Randomized Algorithms. *J. ACM* 65, 5 (2018), 30:1–30:68.

Joost-Pieter Katoen. 2016. The Probabilistic Model Checking Landscape. In *LICS*. ACM, 31–45.

Joost-Pieter Katoen, Friedrich Gretz, Nils Jansen, Benjamin Lucien Kaminski, and Federico Olmedo. 2015. Understanding Probabilistic Programs. In *Correct System Design (LNCS, Vol. 9360)*. Springer, 15–32.

Joost-Pieter Katoen, Annabelle McIver, Larissa Meinicke, and Carroll C. Morgan. 2010. Linear-Invariant Generation for Probabilistic Programs: Automated Support for Proof-Based Methods. In *SAS (LNCS, Vol. 6337)*. Springer, 390–406.

Bronisław Knaster. 1928. Un Théorème sur les Functions D'ensembles. *Annales de la Societe Polonaise de Mathematique* 6 (1928), 133–134.

Naoki Kobayashi, Ugo Dal Lago, and Charles Grellois. 2020. On the Termination Problem for Probabilistic Higher-Order Recursive Programs. *Log. Methods Comput. Sci.* 16, 4 (2020).

Dexter Kozen. 1981. Semantics of Probabilistic Programs. *J. Comput. Syst. Sci.* 22, 3 (1981), 328–350.

Dexter Kozen. 1983. A Probabilistic PDL. In *STOC*. ACM, 291–297.

Dexter Kozen. 1985. A Probabilistic PDL. *J. Comput. Syst. Sci.* 30, 2 (1985), 162–178.

Marta Kwiatkowska, Gethin Norman, and David Parker. 2002. PRISM: Probabilistic Symbolic Model Checker. In *TOOLS*. Springer, 200–204.

Marta Z. Kwiatkowska. 2003. Model Checking for Probability and Time: From Theory to Practice. In *LICS*. IEEE Computer Society, 351.

Kim Guldstrand Larsen and Arne Skou. 1991. Bisimulation through Probabilistic Testing. *Inf. Comput.* 94, 1 (1991), 1–28.

Jean-Louis Lassez, V. L. Nguyen, and E. A. Sonenberg. 1982. Fixed Point Theorems and Semantics: A Folk Tale. *Inform. Process. Lett.* 14, 3 (1982), 112–116.

William H. McCrea and Francis J. W. Whipple. 1940. XXII.—Random Paths in Two and Three Dimensions. *Proceedings of the Royal Society of Edinburgh* 60, 3 (1940), 281–298.

Annabelle McIver and Carroll Morgan. 2001. Partial Correctness for Probabilistic Demonic Programs. *Theor. Comput. Sci.* 266, 1-2 (2001), 513–541.

Annabelle McIver and Carroll Morgan. 2005. *Abstraction, Refinement and Proof for Probabilistic Systems.* Springer.

Annabelle McIver, Carroll Morgan, Benjamin Lucien Kaminski, and Joost-Pieter Katoen. 2018. A New Proof Rule for Almost-Sure Termination. *Proc. ACM Program. Lang.* 2, POPL (2018), 33:1–33:28.

Elliot W Montroll. 1956. Random Walks in Multidimensional Spaces, Especially on Periodic Lattices. *J. Soc. Indust. Appl. Math.* 4, 4 (1956), 241–260.

Marcel Moosbrugger, Ezio Bartocci, Joost-Pieter Katoen, and Laura Kovács. 2021. The Probabilistic Termination Tool Amber. In *FM (LNCS, Vol. 13047)*. Springer, 667–675.

Carroll Morgan and Annabelle McIver. 1999. An Expectation-Transformer Model for Probabilistic Temporal Logic. *Log. J. IGPL* 7, 6 (1999), 779–804.

Carroll Morgan, Annabelle McIver, and Karen Seidel. 1996. Probabilistic Predicate Transformers. *ACM Trans. Program. Lang. Syst.* 18, 3 (1996), 325–353.

Andrzej S. Murawski and Joël Ouaknine. 2005. On Probabilistic Program Equivalence and Refinement. In *CONCUR (LNCS, Vol. 3653)*. Springer, 156–170.

Van Chan Ngo, Quentin Carbonneaux, and Jan Hoffmann. 2018. Bounded Expectations: Resource Analysis for Probabilistic Programs. In *PLDI*. ACM, 496–512.

Federico Olmedo, Friedrich Gretz, Nils Jansen, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Annabelle McIver. 2018. Conditioning in Probabilistic Programming. *ACM Trans. Program. Lang. Syst.* 40, 1 (2018), 4:1–4:50.

Federico Olmedo, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja. 2016. Reasoning about Recursive Probabilistic Programs. In *LICS*. ACM, 672–681.

David Park. 1969. Fixpoint Induction and Proofs of Program Properties. *Machine Intelligence* 5 (1969).

G. Pólya. 1921. Über eine Aufgabe der Wahrscheinlichkeitsrechnung betreffend die Irrfahrt im Straßennetz. *Math. Ann.* 84 (1921), 149–160.

Tim Quatmann and Joost-Pieter Katoen. 2018. Sound Value Iteration. In *CAV (I) (LNCS, Vol. 10981)*. Springer, 643–661.

Nasser Saheb-Djahromi. 1978. Probabilistic LCF. In *MFCS (LNCS, Vol. 64)*. Springer, 442–451.

Sriram Sankaranarayanan. 2020. Quantitative Analysis of Programs with Probabilities and Concentration of Measure Inequalities. In *Foundations of Probabilistic Programming*, Gilles Barthe, Joost-Pieter Katoen, and Alexandra Silva (Eds.). Cambridge University Press, 259–294.

Marco Schneider. 1993. Self-Stabilization. *ACM Comput. Surv.* 25, 1 (1993), 45–67.

Seyed Mahdi Shamsi, Gian Pietro Farina, Marco Gaboardi, and Nils Napp. 2020. Probabilistic Programming Languages for Modeling Autonomous Systems. In *MFI*. IEEE, 32–39.

Marcin Szymczak and Joost-Pieter Katoen. 2019. Weakest Preexpectation Semantics for Bayesian Inference - Conditioning, Continuous Distributions and Divergence. In *SETSS (LNCS, Vol. 12154)*. Springer, 44–121.

Alfred Tarski. 1955. A Lattice-Theoretical Fixpoint Theorem and Its Applications. *Pacific J. Math.* 5, 2 (1955), 285–309.

Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, and Frank Wood. 2018. An Introduction to Probabilistic Programming. *CoRR* abs/1809.10756 (2018).

Di Wang, Jan Hoffmann, and Thomas W. Reps. 2021a. Central Moment Analysis for Cost Accumulators in Probabilistic Programs. In *PLDI*. ACM, 559–573.

Jinyi Wang, Yican Sun, Hongfei Fu, Krishnendu Chatterjee, and Amir Kafshdar Goharshady. 2021b. Quantitative Analysis of Assertion Violations in Probabilistic Programs. In *PLDI*. ACM, 1171–1186.

Peixin Wang, Hongfei Fu, Krishnendu Chatterjee, Yuxin Deng, and Ming Xu. 2020. Proving Expected Sensitivity of Probabilistic Programs With Randomized Variable-Dependent Termination Time. *Proc. ACM Program. Lang.* 4, POPL (2020), 25:1–25:30.

Peixin Wang, Hongfei Fu, Amir Kafshdar Goharshady, Krishnendu Chatterjee, Xudong Qin, and Wenjun Shi. 2019. Cost Analysis of Nondeterministic Probabilistic Programs. In *PLDI*. ACM, 204–220.

David Williams. 1991. *Probability with Martingales*. Cambridge University Press.

Mingsheng Ying. 2011. Floyd-Hoare logic for Quantum Programs. *ACM Trans. Program. Lang. Syst.* 33, 6 (2011), 19:1–19:49.

Received 2022-10-28; accepted 2023-02-25