Disproving (Positive) Almost-Sure Termination of Probabilistic Term Rewriting via Random Walks*

Jan-Christoph Kassing, Henri Nagel, Alexander Schlecht, and Jürgen Giesl

RWTH Aachen University, Aachen, Germany

Abstract. In recent years, numerous techniques were developed to automatically prove termination of different kinds of probabilistic programs. However, there are only few automated methods to *disprove* their termination. In this paper, we present the first techniques to automatically disprove (positive) almost-sure termination of probabilistic term rewrite systems. Disproving termination of non-probabilistic systems requires finding a finite representation of an infinite computation, e.g., a loop of the rewrite system. We extend such qualitative techniques to probabilistic term rewriting, where a quantitative analysis is required. In addition to the existence of a loop, we have to count the number of such loops in order to embed suitable random walks into a computation, thereby disproving termination. To evaluate their power, we implemented all our techniques in the tool AProVE.

1 Introduction

While termination is undecidable in general, automatic techniques to prove termination are crucial in many applications. However, proving *non-termination* is equally important, e.g., to detect bugs in programs. While termination and non-termination of ordinary programs has been studied for decades, the work on automated termination analysis of *probabilistic programs* is quite recent. Such programs may have *probabilistic choices*, which allow the program to proceed in several different ways. Probabilistic programming is used to deal with uncertainty in data and has applications in many areas such as machine learning and Bayesian inference [18].

In the probabilistic setting, requiring all executions to be finite is a hard restriction. Instead, one usually considers less restrictive notions of termination like (positive) almost-sure termination. A program is *almost-surely terminating* (`AST`) if every computation terminates with probability 1. A strictly stronger notion is *positive* `AST` (`PAST`) [5, 36], where every computation must have a finite expected number of steps. It is well known that `PAST` implies `AST` but not vice versa.

Term rewriting [2] is a fundamental concept to transform and evaluate expressions, and techniques to analyze termination of term rewrite systems (TRSs) are used for termination analysis of programs in many languages. Term rewriting was adapted to the probabilistic setting in [1, 4, 5]. While techniques to automatically prove `AST` and `PAST` of probabilistic TRSs (PTRSs) were developed in [1, 24–27], up to now there were no approaches for non-termination of PTRSs. In this paper, we develop the first techniques to automatically disprove `AST` and `PAST` of PTRSs.

* funded by the DFG Research Training Group 2236 UnRAVeL

Contribution. We introduce a general approach to disprove termination of PTRSs by embedding non-terminating random walks into rewrite sequences of PTRSs (Thm. 6). First, we embed random walks via *occurrences* of *loops* (Thm. 10, 14, and 16), which were previously used to disprove termination of TRSs, see, e.g., [14]. Second, we embed random walks via looping *pattern terms* (Thm. 27), which were previously used to prove non-looping non-termination of TRSs [10]. While the original techniques were *qualitative* (*find the existence of a loop*), the probabilistic setting requires *quantitative* analyses (*find the number of possible loops*). We develop a dynamic programming algorithm (Alg. 1) to solve the counting problems that arise in this setting, e.g., to determine how many instantiations of a term t occur in a term s at orthogonal positions. We implemented all our techniques in the tool AProVE and perform an experimental evaluation to demonstrate their applicability.

Related Work. Techniques to prove non-termination of non-probabilistic term rewriting have been developed for decades, see, e.g., [10, 11, 14, 34, 35]. Tools analyzing (non-)termination of TRSs participate annually in the *Termination Competition* (*TermComp*) [16]. A related problem to proving non-termination is analyzing reachability (or (in)feasibility). Techniques for analyzing reachability in term rewriting were presented in, e.g., [21, 32, 38, 43].

For imperative programs, there are also numerous techniques for proving non-termination, e.g., [7, 9, 12, 13, 20, 28, 30], and tools for termination analysis of imperative programs compete annually in *TermComp* and in the software verification competition (*SV-COMP*) [3]. In addition to specific techniques for non-termination, there also exist *decision* procedures for termination on certain subclasses of programs, e.g., [6, 22, 23, 31, 40, 42]. To automatically disprove `AST` or `PAST`, we are only aware of approaches for probabilistic *imperative* programs: there exists a technique via repulsing supermartingales [8], and a technique via fixpoints and Park induction [39]. Based on [8], martingale-based proof rules which can also disprove `AST` and `PAST` were implemented in the tool Amber [33].

The most common non-termination technique for TRSs is the detection of loops. Thus, to disprove `AST` or `PAST` of probabilistic term rewriting, in this work we lift the idea of disproving termination via loops to the probabilistic setting, and consider embeddings based on loops.

Structure. We present preliminaries on probability theory, random walks, and (probabilistic) term rewriting in Sect. 2. Then, we introduce our novel approach to disprove termination of PTRSs via an embedding of random walks in Sect. 3. More precisely, in Sect. 4 we show how to embed random walks based on loops and develop a dynamic programming algorithm to solve the arising counting problems for occurrences of terms. In Sect. 5, we show how to embed random walks based on looping pattern terms. We evaluate our implementation and conclude in Sect. 6. The proofs of our results can be found in App. A.

2 Preliminaries

In Sect. 2.1, we start with basic concepts of probability theory and define the notion of a random walk (program). Then, we recapitulate ordinary term rewriting in Sect. 2.2, and probabilistic term rewriting in Sect. 2.3.

2.1 Probability Theory and Random Walks

We start by recapitulating *random walk programs (without direct termination)* as in [17], which we will use as lower bounds to disprove (positive) almost-sure termination of probabilistic term rewrite systems in Sect. 4 and 5. For an introduction to general random walks, see, e.g., [19, 29, 37].

In probability theory, one wants to assign probabilities to possible *outcomes* $\alpha \in \Omega$ of a random process. Since assigning probabilities to individual outcomes is often not feasible, one considers a set of *events* $\mathfrak{A} \subseteq 2^{\Omega}$ instead, where an event is a subset of outcomes, and assigns probabilities to each event. Such an *event space* $\mathfrak{A}$ has to contain Ω itself and it must be closed under complement and countable unions. The pair $(\Omega, \mathfrak{A})$ is called a *measurable space*. A *probability space* $(\Omega, \mathfrak{A}, \mathbb{P})$ extends a measurable space $(\Omega, \mathfrak{A})$ by a *probability measure* $\mathbb{P}$ which maps every event from $\mathfrak{A}$ to a probability between 0 and 1 such that $\mathbb{P}(\Omega) = 1$, $\mathbb{P}(\varnothing) = 0$, and $\mathbb{P}(\biguplus_{i \geq 0} A_i) = \sum_{i \geq 0} \mathbb{P}(A_i)$ for every family of pairwise disjoint events $A_i \in \mathfrak{A}$. As in [17], we use the measurable space $(\mathbb{Z}^{\omega}, \mathfrak{A}_{\mathrm{cyl}})$ on infinite words defined by the typical cylinder construction used in MDP theory. We call the possible outcomes $\alpha \in \mathbb{Z}^{\omega}$ *runs* and a finite word $\alpha \in \mathbb{Z}^*$ a *prefix run*. Let $\alpha(i)$ denote the number at position i in α.

A *random walk (program)* μ is a function $\mu : \mathbb{Z} \to \mathbb{R}_{\geq 0}$ whose *support* $\mathrm{Supp}(\mu) = \{x \in \mathbb{Z} \mid \mu(x) > 0\}$ is finite and satisfies $\sum_{x \in \mathrm{Supp}(\mu)} \mu(x) = 1$. The function μ represents the transition relation of a random walk on $\mathbb{Z}$. If the current value of the random walk is $y > 0$, then for any $x \in \mathbb{Z}$, $\mu(x)$ is the probability that the random walk transitions from y to $y + x$. We stop when we reach a number ≤ 0 and do not perform any transition steps anymore.[1] For a prefix run $\alpha \in \mathbb{Z}^*$, let $\langle \alpha \rangle \subseteq \mathbb{Z}^{\omega}$ denote the set of all infinite words that have α as prefix (called a *cylinder set*). The probability measure $\mathbb{P}^{\mu}_{x_0}$ (given some start value $x_0 \in \mathbb{Z}$) of a random walk is defined on the event space $\mathfrak{A}_{\mathrm{cyl}} = \{\langle \alpha \rangle \mid \alpha \in \mathbb{Z}^*\}$, i.e., on the cylinder sets $\langle \alpha \rangle$ of all prefix runs α. Given a finite word $\alpha = \alpha(0) \ldots \alpha(k)$, we define $\mathbb{P}^{\mu}_{x_0}(\langle \alpha \rangle) = 0$ if $\alpha(0) \neq x_0$, and $\mathbb{P}^{\mu}_{x_0}(\langle \alpha \rangle) = \prod_{i=0}^{\infty} \mu(\alpha(i+1) - \alpha(i))$ otherwise.

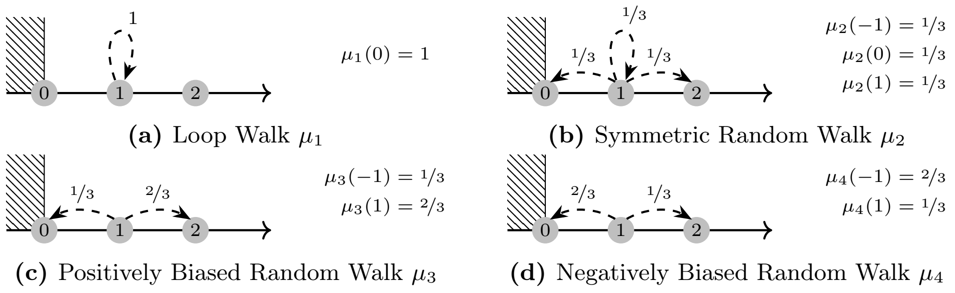
Depending on μ and its *expected value* $\mathbb{E}(\mu) = \sum_{x \in \mathrm{Supp}(\mu)} x \cdot \mu(x)$, we characterize four different types of random walks:

1. If $\mu(0) = 1$, then μ is a *loop walk*.
2. If $\mu(0) < 1$ and $\mathbb{E}(\mu) = 0$, then μ is a *symmetric random walk*.
3. If $\mu(0) < 1$ and $\mathbb{E}(\mu) > 0$, then μ is a *positively biased random walk*.
4. If $\mu(0) < 1$ and $\mathbb{E}(\mu) < 0$, then μ is a *negatively biased random walk*.

In Fig. 1 one can see examples for all four different types, where we start at $x_0 = 1$.

We are interested in the probability of termination and in the expected number of steps it takes to terminate. The random walk μ *certainly terminates* if there is no infinite word α such that $\alpha(i) > 0$ and $\mu(\alpha(i+1) - \alpha(i)) > 0$ holds for all $i \in \mathbb{N}$. This only holds if we have $\mu(x) = 0$ for all $x \geq 0$. Since this is a rather restrictive requirement, we are more interested in the *probability of termination*. We say that a run α *terminates* if there exists an $n \in \mathbb{N}$ such that $\alpha(n) \leq 0$, and the termination length $|\alpha|$ is defined as the smallest such n. Let $\langle \mathtt{SN} \rangle = \biguplus_{n \in \mathbb{N}} \biguplus_{\alpha \in \mathbb{Z}^{\omega}, |\alpha| = n} \langle \alpha \rangle$ be the event of all terminating runs. (Here, as usual, "$\mathtt{SN}$" stands for "$\underline{\text{s}}$trong $\underline{\text{n}}$ormalization", which is equivalent to "termination".) Then the *probability of termination* of a

[1] Ordinary random walks [37] do not stop and are identically distributed everywhere.



(a) Loop Walk μ_1 **(b)** Symmetric Random Walk μ_2

(c) Positively Biased Random Walk μ_3 **(d)** Negatively Biased Random Walk μ_4

Fig. 1: Four Different Random Walks μ_1, μ_2, μ_3, and μ_4

random walk μ starting in x_0 is $\mathbb{P}^{\mu}_{x_0}(\langle \texttt{SN} \rangle) = \sum_{n \in \mathbb{N}} \sum_{\alpha \in \mathbb{Z}^\omega, |\alpha| = n} \mathbb{P}^{\mu}_{x_0}(\langle \alpha \rangle)$.

A random walk μ is *almost-surely terminating* (AST) if we have $\mathbb{P}^{\mu}_{x_0}(\langle \texttt{SN} \rangle) = 1$ for all $x_0 \in \mathbb{Z}$. In addition, one is also interested in the expected number of steps it takes to terminate. We define the *expected derivation length* $\mathrm{edl}(\mu, x_0) \in \mathbb{R}_{\geq 0} \cup \{\infty\}$ of a random walk μ starting in x_0 as $\mathrm{edl}(\mu, x_0) = \infty$ if μ is not AST, and otherwise, as $\mathrm{edl}(\mu, x_0) = \sum_{n \in \mathbb{N}} \sum_{\alpha \in \mathbb{Z}^\omega, |\alpha| = n} n \cdot \mathbb{P}^{\mu}_{x_0}(\langle \alpha \rangle)$. A random walk μ is *positively almost-surely terminating* (PAST) if we have $\mathrm{edl}(\mu, x_0) < \infty$ for all $x_0 \in \mathbb{Z}$.

With this characterization, as in [17, Thm. 18], classical results on random walks [37] yield the following classification w.r.t. AST and PAST.

Theorem 1 (AST and PAST of Random Walks). *Let μ be a random walk.*

1. *If μ is a loop walk, then μ is neither* AST *nor* PAST.
2. *If μ is a positively biased random walk, then μ is neither* AST *nor* PAST.
3. *If μ is a symmetric random walk, then μ is* AST *but not* PAST.
4. *If μ is a negatively biased random walk, then μ is* AST *and* PAST.

Example 2. Based on Thm. 1, μ_1 and μ_3 are neither AST nor PAST. The random walk μ_2 is AST but not PAST, and μ_4 is both AST and PAST.

2.2 Rewriting

Next, we recapitulate abstract rewriting and term rewriting, see, e.g., [2]. An *abstract reduction system* (ARS) on a set A is a binary relation $\to \subseteq A \times A$. Instead of $(a, b) \in \to$ one writes $a \to b$ to indicate that b is a direct successor of a. For any such relation $\to$ and any $n \in \mathbb{N}$, we define $\to^n$ as $\to^0 = \{(a, a) \mid a \in A\}$ and $\to^{n+1} = \to^n \circ \to$, where "$\circ$" denotes composition of relations. Moreover, $\to^* = \bigcup_{n \in \mathbb{N}} \to^n$ and $\to^+ = \bigcup_{n \in \mathbb{N}_{>0}} \to^n$. Let $\texttt{NF}_\to$ denote the set of all objects that are in *normal form* w.r.t. $\to$, i.e., $a \in \texttt{NF}_\to$ if there is no $b \in A$ with $a \to b$.

The set $\mathcal{T}(\Sigma, \mathcal{V})$ of all *terms* over a finite set of *function symbols* $\Sigma = \biguplus_{k \in \mathbb{N}} \Sigma_k$ and an infinite set of *variables* $\mathcal{V}$ is the smallest set with $\mathcal{V} \subseteq \mathcal{T}(\Sigma, \mathcal{V})$, and if $f \in \Sigma_k$ and $t_1, \ldots, t_k \in \mathcal{T}(\Sigma, \mathcal{V})$, then $f(t_1, \ldots, t_k) \in \mathcal{T}(\Sigma, \mathcal{V})$. If Σ and $\mathcal{V}$ are clear from the context, we just write $\mathcal{T}$ instead of $\mathcal{T}(\Sigma, \mathcal{V})$. For example, $\mathsf{gt}(\mathsf{s}(x), \mathsf{0})$ is a term over a signature where gt has arity 2, s has arity 1, and $\mathsf{0}$ has arity 0. A *substitution* is a function $\sigma : \mathcal{V} \to \mathcal{T}(\Sigma, \mathcal{V})$ where $\mathrm{dom}(\sigma) = \{x \in \mathcal{V} \mid \sigma(x) \neq x\}$ is finite, and we often write $x\sigma$ instead of $\sigma(x)$. If $\mathrm{dom}(\sigma) = \{x_1, \ldots, x_n\}$ and $x_i = s_i$ for all $1 \leq i \leq n$, then we also write $\sigma = [x_1/s_1, \ldots, x_n/s_n]$. Substitutions homomorphically extend to terms: if $t = f(t_1, \ldots, t_k) \in \mathcal{T}$ then $t\sigma = f(t_1\sigma, \ldots, t_k\sigma)$. Thus, for a substitution σ with $x\sigma = \mathsf{s}(\mathsf{0})$ we obtain $\mathsf{s}(x)\sigma = \mathsf{s}(\mathsf{s}(\mathsf{0}))$. For any term $t \in \mathcal{T}$, the set of *positions* $\mathrm{Pos}(t)$ is the smallest subset of $\mathbb{N}^*$ with $\varepsilon \in \mathrm{Pos}(t)$, and if $t = f(t_1, \ldots, t_k)$ then for all $1 \leq j \leq k$ and all $\pi \in \mathrm{Pos}(t_j)$ we have $j.\pi \in \mathrm{Pos}(t)$. By $\mathrm{Pos}_X(t) \subseteq \mathrm{Pos}(t)$ we denote all positions of symbols or variables from $X \subseteq \Sigma \cup \mathcal{V}$ in t. A position π_1 is *above* π_2 if π_1 is a (not necessarily proper) prefix of π_2. If π_1 is not above π_2 and π_2 is not above π_1, then they are called *orthogonal* (denoted $\pi_1 \perp \pi_2$). If $\pi \in \mathrm{Pos}(t)$ then $t|_\pi$ denotes the subterm starting at position π and $t[r]_\pi$ denotes the term that results from replacing the subterm $t|_\pi$ at position π with the term $r \in \mathcal{T}$. We write $t \trianglelefteq s$ if t is a subterm of s and $t \triangleleft s$ if t is a *proper* subterm of s (i.e., if $t \trianglelefteq s$ and $t \neq s$). For example, we have $\mathrm{Pos}(\mathsf{gt}(\mathsf{s}(x), \mathsf{0})) = \{\varepsilon, 1, 1.1, 2\}$, $\mathsf{gt}(\mathsf{s}(x), \mathsf{0})|_2 = \mathsf{0}$, $\mathsf{gt}(\mathsf{s}(x), \mathsf{0})[\mathsf{s}(y)]_2 = \mathsf{gt}(\mathsf{s}(x), \mathsf{s}(y))$, and $\mathsf{s}(y) \triangleleft \mathsf{gt}(\mathsf{s}(x), \mathsf{s}(y))$. A *context* C is a term from $\mathcal{T}(\Sigma \uplus \{\Box\}, \mathcal{V})$ which contains exactly one occurrence of the constant $\Box$ (called "hole"). If $C|_\pi = \Box$, then $C[s]$ is a shorthand for $C[s]_\pi$.

A *term rewrite rule* $\ell \to r$ is a pair of terms $(\ell, r) \in \mathcal{T} \times \mathcal{T}$ such that $\mathcal{V}(r) \subseteq \mathcal{V}(\ell)$ and $\ell \notin \mathcal{V}$, where $\mathcal{V}(t)$ is the set of all variables occurring in $t \in \mathcal{T}$. A *term rewrite system* (TRS) $\mathcal{R}$ is a finite set of term rewrite rules, and it induces an ARS $(\mathcal{T}, \to_\mathcal{R})$ on terms where $s \to_\mathcal{R} t$ holds if there is a $\pi \in \mathrm{Pos}(s)$, a rule $\ell \to r \in \mathcal{R}$, and a substitution σ such that $s|_\pi = \ell\sigma$ and $t = s[r\sigma]_\pi$. We sometimes simply refer to $\mathcal{R}$ instead of $\to_\mathcal{R}$. As an example, the following TRS $\mathcal{R}_{\mathsf{gt}}$ computes the "**g**reater **t**han" function on natural numbers (represented by $\mathsf{0}$ and the successor function s).

$$\mathsf{gt}(\mathsf{s}(x), \mathsf{s}(y)) \to \mathsf{gt}(x, y) \qquad \mathsf{gt}(\mathsf{s}(x), \mathsf{0}) \to \mathsf{true} \qquad \mathsf{gt}(\mathsf{0}, y) \to \mathsf{false}$$

Here, we have $\mathsf{gt}(\mathsf{s}(\mathsf{0}), \mathsf{s}(\mathsf{0})) \to_{\mathcal{R}_{\mathsf{gt}}} \mathsf{gt}(\mathsf{0}, \mathsf{0}) \to_{\mathcal{R}_{\mathsf{gt}}} \mathsf{false}$, where $\mathsf{false} \in \mathtt{NF}_{\mathcal{R}_{\mathsf{gt}}}$.

2.3 Probabilistic Rewriting

A *probabilistic* ARS has finite multi-distributions on the right-hand sides of its rewrite rules. A finite *multi-distribution* μ on a set $A \neq \varnothing$ is a finite multiset of pairs $(p : a)$, where $0 < p \leq 1$ is a probability and $a \in A$, such that $\sum_{(p:a) \in \mu} p = 1$. Let $\mathrm{FDist}(A)$ denote the set of all finite multi-distributions on A. For $\mu \in \mathrm{FDist}(A)$, its *support* is the multiset $\mathrm{Supp}(\mu) = \{a \mid (p : a) \in \mu \text{ for some } p\}$. A *probabilistic abstract reduction system* (PARS) is a pair $(A, \to)$ such that $\to \in A \times \mathrm{FDist}(A)$.

A *probabilistic term rewrite rule* $\ell \to \mu$ is a pair $(\ell, \mu) \in \mathcal{T} \times \mathrm{FDist}(\mathcal{T})$ such that $\ell \notin \mathcal{V}$ and $\mathcal{V}(r) \subseteq \mathcal{V}(\ell)$, and a *probabilistic TRS* (PTRS) is a finite set of probabilistic term rewrite rules. Similar to TRSs, a PTRS $\mathcal{P}$ induces a PARS $(\mathcal{T}, \to_\mathcal{P})$ with $s \to_\mathcal{P} \{p_1 : t_1, \ldots, p_k : t_k\}$ if there is a position $\pi \in \mathrm{Pos}(s)$, a rule $\ell \to \{p_1 : r_1, \ldots, p_k : r_k\} \in \mathcal{P}$, and a substitution σ such that $s|_\pi = \ell\sigma$ and $t_j = s[r_j\sigma]_\pi$ for all $1 \leq j \leq k$. Again, we sometimes refer to $\mathcal{P}$ instead of $\to_\mathcal{P}$. Consider the PTRS $\mathcal{P}_{\mathsf{geo}}$ with the only rule $\mathsf{geo}(x) \to \{1/2 : \mathsf{geo}(\mathsf{s}(x)),\ 1/2 : x\}$. When starting with the term $\mathsf{geo}(\mathsf{0})$, repeated rewriting yields $\mathsf{s}^k(\mathsf{0})$ (representing $k \in \mathbb{N}$) with a probability of $(1/2)^{k+1}$, i.e., a geometric distribution.

To track rewrite sequences of a PARS $(A, \to)$ with their probabilities, we consider (possibly infinite) *rewrite sequence trees (RSTs)* [27]. The nodes v of a $\to$-RST are labeled by pairs $(p_v : a_v)$ of a probability $p_v \in (0, 1]$ and an object $a_v \in A$, where the probability at the root is 1. For each node v with successors $w_1, \ldots, w_k$,

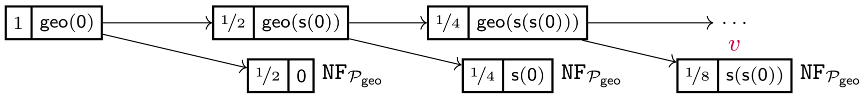


Fig. 2: A $\mathcal{P}_{\mathsf{geo}}$-RST $\mathfrak{T}$ with $\mathrm{root}(\mathfrak{T}) = \mathsf{geo}(0)$ and $\mathrm{h}(\mathfrak{T}) = \omega$. The node v is labeled by the probability $p_v = 1/8$ and the term $a_v = \mathsf{s}(\mathsf{s}(0))$, and it has depth $\mathrm{d}(v) = 3$.

the edge relation represents a rewrite step, i.e., $a_v \to \{\frac{p_{w_1}}{p_v} : a_{w_1}, \ldots, \frac{p_{w_k}}{p_v} : a_{w_k}\}$. For a $\to$-RST $\mathfrak{T}$, $V(\mathfrak{T})$ denotes its set of nodes, $\mathrm{root}(\mathfrak{T})$ is the object at its root, and $\mathrm{Leaf}(\mathfrak{T})$ denotes its set of leaves. The *depth* $\mathrm{d}(v)$ of a node $v \in V(\mathfrak{T})$ is the number of steps it takes to reach v from the root. The *height* of a tree $\mathfrak{T}$ is $\mathrm{h}(\mathfrak{T}) = \sup_{v \in V(\mathfrak{T})} \mathrm{d}(v) \in \mathbb{N} \cup \{\omega\}$. An example for a $\mathcal{P}_{\mathsf{geo}}$-RST is shown in Fig. 2.

For a $\to$-RST $\mathfrak{T}$, its *termination probability* is $|\mathfrak{T}| = \sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v$. A PARS $(A, \to)$ is *almost-surely terminating* (AST) if $|\mathfrak{T}| = 1$ holds for all $\to$-RSTs $\mathfrak{T}$, i.e., if the probability of termination is always 1. However, AST is not sufficient to guarantee that the expected number of rewrite steps is finite. The *expected derivation length* of a $\to$-RST $\mathfrak{T}$ is $\mathrm{edl}(\mathfrak{T}) = \infty$ if $|\mathfrak{T}| < 1$, and $\mathrm{edl}(\mathfrak{T}) = \sum_{v \in \mathrm{Leaf}(\mathfrak{T})} \mathrm{d}(v) \cdot p_v$, otherwise. For the $\mathcal{P}_{\mathsf{geo}}$-RST $\mathfrak{T}$ in Fig. 2 we obtain $\mathrm{edl}(\mathfrak{T}) = 1 \cdot 1/2 + 2 \cdot 1/4 + 3 \cdot 1/8 + \ldots = 2$, so in expectation we perform 2 rewrite steps. A PARS $(A, \to)$ is *positively almost-surely terminating* (PAST) if $\mathrm{edl}(\mathfrak{T})$ is finite for all $\to$-RSTs $\mathfrak{T}$. These notions of AST and PAST for PARSs are equivalent to the ones in [1, 5, 24] where AST and PAST are defined via a lifting of $\to$ to multisets or via stochastic processes.

3 Disproving AST and PAST via Loops and Embeddings

The most common approach to disprove termination of a TRS $\mathcal{R}$ is by finding *loops*. A loop is a sequence $t \to_{\mathcal{R}}^{+} C[t\sigma]$ for some term t, context C, and substitution σ. Indeed, such a loop gives rise to an infinite rewrite sequence $t \to_{\mathcal{R}}^{n} C[t\sigma] \to_{\mathcal{R}}^{n} C[C[t\sigma]\sigma] \to_{\mathcal{R}} \ldots$ for some $n \in \mathbb{N}_{>0}$. However, to disprove AST or PAST of a PTRS $\mathcal{P}$, it is not sufficient to find a loop in the *non-probabilistic variant* $\mathrm{np}(\mathcal{P}) = \{\ell \to r \mid \ell \to \mu \in \mathcal{P}, r \in \mathrm{Supp}(\mu)\}$.

Example 3. Consider the PTRS $\mathcal{P}_1$ with the rule $\mathsf{g}(x) \to \{2/3 : x, 1/3 : \mathsf{g}(\mathsf{g}(x))\}$ modeling a negatively biased random walk on the number of g's. Here, $\mathrm{np}(\mathcal{P}_1)$ contains the two rules $\mathsf{g}(x) \to x$ and $\mathsf{g}(x) \to \mathsf{g}(\mathsf{g}(x))$. The latter gives rise to the loop $\mathsf{g}(x) \to C[\mathsf{g}(x)\sigma]$ with $C = \mathsf{g}(\Box)$ and the identity substitution σ. However, $\mathcal{P}_1$ corresponds to the random walk μ_4 from Fig. 1, and thus, it is PAST.

Instead of $\mathrm{np}(\mathcal{P})$, one has to consider $\mathcal{P}$-RSTs. If there is a $\mathcal{P}$-RST with root t and an instance of t in every leaf, then $\mathcal{P}$ is not AST and thus, also not PAST.

Theorem 4 (Disproving AST via Loops). *Let $\mathcal{P}$ be a PTRS and $\mathfrak{T}$ be a $\mathcal{P}$-RST with $\mathrm{h}(\mathfrak{T}) > 0$ such that $\mathrm{root}(\mathfrak{T}) = t$ and for every $v \in \mathrm{Leaf}(\mathfrak{T})$ there is a context C_v and a substitution σ_v such that $t_v = C_v[t\sigma_v]$. Then $\mathcal{P}$ is neither AST nor PAST.*

However, Thm. 4 can only be used to disprove AST for a very restricted class of PTRSs. To handle more complicated examples, instead of requiring that a looping term t occurs in every leaf of a $\mathcal{P}$-RST, one has to find a PARS $(A, \to)$ and a $\to$-RST $\mathfrak{T}$ where it is known that $|\mathfrak{T}| < 1$ (to disprove AST) or $\mathrm{edl}(\mathfrak{T}) = \infty$ (to disprove PAST) holds. In addition, one has to find a $\mathcal{P}$-RST $\mathfrak{T}'$ and an *embedding*

$\mathsf{e} : V(\mathfrak{T}) \to V(\mathfrak{T}')$ which ensures $|\mathfrak{T}'| \leq |\mathfrak{T}|$ and $\mathrm{edl}(\mathfrak{T}) \leq \mathrm{edl}(\mathfrak{T}')$. If $|\mathfrak{T}| < 1$, then we have disproven `AST` of $\mathcal{P}$, and if $\mathrm{edl}(\mathfrak{T}) = \infty$, then we have disproven `PAST` of $\mathcal{P}$.

Definition 5 (RST-Embedding). *Let $i \in \{1, 2\}$, let $(A_i, \to_i)$ be a PARS, and let $\mathfrak{T}_i$ be a $\to_i$-RST. An embedding from $\mathfrak{T}_1$ to $\mathfrak{T}_2$ is an injective mapping $\mathsf{e} : V(\mathfrak{T}_1) \to V(\mathfrak{T}_2)$ such that $p_v = p_{\mathsf{e}(v)}$ for all $v \in V(\mathfrak{T}_1)$ (the probability of v in $\mathfrak{T}_1$ and the probability of its image $\mathsf{e}(v)$ in $\mathfrak{T}_2$ are the same) and if there is a path from a node v to w in $\mathfrak{T}_1$, then there is a path from $\mathsf{e}(v)$ to $\mathsf{e}(w)$ in $\mathfrak{T}_2$ as well.*

Theorem 6 (Lower Bounds via Embeddings). *Let $(A_1, \to_1)$ and $(A_2, \to_2)$ be two PARSs, and let $\mathfrak{T}_i$ be a $\to_i$-RST for $i \in \{1, 2\}$ such that there exists an embedding from $\mathfrak{T}_1$ to $\mathfrak{T}_2$. Then, $|\mathfrak{T}_2| \leq |\mathfrak{T}_1|$ and $\mathrm{edl}(\mathfrak{T}_1) \leq \mathrm{edl}(\mathfrak{T}_2)$.*

4 Embedding Random Walks Based on Term Occurrences

To embed symmetric or even positively biased random walks, we again search for a loop where t rewrites to $C[t\sigma]$, but now we also consider the *probabilities* and the *number of loops* represented by the right-hand side $C[t\sigma]$.

Example 7. Consider the PTRS $\mathcal{P}_2$ with the rule $\mathsf{g}(x) \to \{{}^2\!/\!_3 : x, {}^1\!/\!_3 : \mathsf{g}(\mathsf{g}(\mathsf{g}(x)))\}$ modeling a symmetric random walk on the number of g's, because their number increases in each rule application by ${}^2\!/\!_3 \cdot (-1) + {}^1\!/\!_3 \cdot 2 = 0$ in expectation. While $\mathcal{P}_1$ from Ex. 3 is `PAST`, $\mathcal{P}_2$ is not. The difference is that the symbol g occurs three instead of two times in the second choice of the distribution on the right-hand side.

Ex. 7 illustrates that it is important how often a looping term like $\mathsf{g}(x)$ occurs on the right-hand side. In this example, the three subterms of $\mathsf{g}(\mathsf{g}(\mathsf{g}(x)))$ that are instantiations of $\mathsf{g}(x)$ can indeed be counted separately, because they do not overlap at a non-variable position. Thus, we first solve the problem of finding the maximal number of such non-overlapping instantiations (called *occurrences*).

Definition 8 (Term Occurrences, $\triangleleft_\pi$). *Let $t, s \in \mathcal{T}$ be terms. We say that t* occurs *in s at position π (denoted $t \triangleleft_\pi s$) if $s|_\pi = t\sigma$ for some substitution σ. Two positions π_1 and π_2 are* overlapping *w.r.t. t if there is a position $\tau \in \mathrm{Pos}_\Sigma(t)$ such that $\pi_1 = \pi_2.\tau$ or $\pi_2 = \pi_1.\tau$. Let $\mathrm{NO}(t, s)$ denote the set of all sets of* pairwise **n**on-overlapping **o**ccurrences *of t in s. So for $S \subseteq \mathrm{Pos}(s)$ we have $S \in \mathrm{NO}(t, s)$ iff*

- *if $\pi \in S$ then $t \triangleleft_\pi s$, and*
- *if $\pi_1, \pi_2 \in S$ with $\pi_1 \neq \pi_2$ then π_1, π_2 are non-overlapping w.r.t. t.*

Let $\mathrm{maxNO}(t, s) = \max_{S \in \mathrm{NO}(t,s)} |S|$ *be the maximal cardinality of a set in* $\mathrm{NO}(t, s)$.

So if $t \triangleleft_{\pi_1} s$ and $t \triangleleft_{\pi_2} s$ where π_1, π_2 are non-overlapping w.r.t. t, then π_1, π_2 are either orthogonal, or we have $s|_{\pi_1} = t\sigma_1$ and $t\sigma_2 \trianglelefteq \sigma_1(x)$ (or $s|_{\pi_2} = t\sigma_2$ and $t\sigma_1 \trianglelefteq \sigma_2(x)$) for some variable $x \in \mathcal{V}(t)$ and substitutions σ_1, σ_2.

For example, we have $\mathsf{g}(x) \triangleleft_\pi \mathsf{g}(\mathsf{g}(\mathsf{g}(x)))$ for all $\pi \in \{\varepsilon, 1, 1.1\}$ and $\mathrm{NO}(\mathsf{g}(x), \mathsf{g}(\mathsf{g}(\mathsf{g}(x))))$ consists of all subsets of $\{\varepsilon, 1, 1.1\}$. Thus, $\mathrm{maxNO}(\mathsf{g}(x), \mathsf{g}(\mathsf{g}(\mathsf{g}(x)))) = 3$. So there are three non-overlapping occurrences of the term $\mathsf{g}(x)$ in $\mathsf{g}(\mathsf{g}(\mathsf{g}(x)))$, i.e., these occurrences can indeed all be counted when analyzing non-termination.

On the other hand, $\mathsf{g}(\mathsf{g}(x)) \triangleleft_\pi \mathsf{g}(\mathsf{g}(\mathsf{g}(x)))$ only holds for $\pi \in \{\varepsilon, 1\}$, and we have $\mathrm{NO}(\mathsf{g}(\mathsf{g}(x)), \mathsf{g}(\mathsf{g}(\mathsf{g}(x)))) = \{\varnothing, \{\varepsilon\}, \{1\}\}$, but $\{\varepsilon, 1\} \notin \mathrm{NO}(\mathsf{g}(\mathsf{g}(x)), \mathsf{g}(\mathsf{g}(\mathsf{g}(x))))$, because ε and 1 are overlapping w.r.t. $\mathsf{g}(\mathsf{g}(x))$, since $1 \in \mathrm{Pos}_\Sigma(\mathsf{g}(\mathsf{g}(x)))$. Thus, we

Algorithm 1: Compute maxNO(t, s) for $t \notin \mathcal{V}$

```
1  $q \leftarrow$ EmptyList()
2  for $s' \trianglelefteq s$ do
3      $\alpha_{s'} \leftarrow 0$, $\beta_{s'} \leftarrow 0$, $\gamma_{s'} \leftarrow 0$                // Initialize values
4  q.enqueue(leaves)                                          // Start with the leaves
5  while not q.isEmpty() do
6      $s' \leftarrow$ q.dequeue()
7      $\gamma_{s'} \leftarrow 1$
8      if $s'$ is leaf then
9          $\alpha_{s'} \leftarrow \begin{cases} 1, & \text{if } t \blacktriangleleft_\varepsilon s' \\ 0, & \text{otherwise} \end{cases}$   // $t$ matches $s'$ / // $t$ does not match $s'$
10     else
11         $\beta_{s'} \leftarrow \sum_{s'=f(v_1,\ldots,v_k)} \sum_{j=1}^{k} \alpha_{v_j}$
12         $\alpha_{s'} \leftarrow \begin{cases} \max\left\{\beta_{s'}, \sum_{\pi \in \mathrm{Pos}_{\mathcal{V}}(t)} \alpha_{s'|_\pi} + 1\right\}, & \text{if } t \blacktriangleleft_\varepsilon s' \\ \beta_{s'}, & \text{otherwise} \end{cases}$   // $t$ matches $s'$ / // $t$ does not match $s'$
13     if $\forall v \in s'$.parent().children() : $\gamma_v = 1$      // All siblings of $s'$ processed
14      then
15         q.enqueue($s'$.parent())                              // Enqueue parent
16 return $\alpha_s$
```

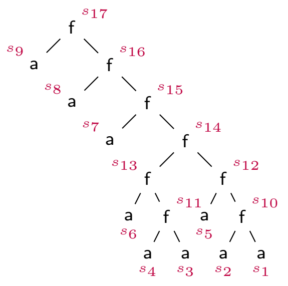
just obtain maxNO$(\mathsf{g}(\mathsf{g}(x)), \mathsf{g}(\mathsf{g}(\mathsf{g}(x)))) = 1$, i.e., we can only count one instead of two occurrences of the term $\mathsf{g}(\mathsf{g}(x))$ in $\mathsf{g}(\mathsf{g}(\mathsf{g}(x)))$. Such overlapping occurrences cannot be counted separately, because rewriting one occurrence may interfere with the possibility to rewrite the other one, see Ex. 12.

Computing maxNO(t, s) can be done via a dynamic programming algorithm traversing the tree structure of s from the leaves to the root, see Alg. 1. For every node, i.e., every subterm $s' \trianglelefteq s$, we compute two numbers: (1) $\alpha_{s'} = \mathrm{maxNO}(t, s')$ and (2) $\beta_{s'} = \max_{S \in \mathrm{NO}(t,s'), \varepsilon \notin S} |S|$, i.e., the maximal number of non-overlapping occurrences of t in s' if we do not consider an occurrence of t at the root of s'. For the leaves, we set $\beta_{s'} = 0$ and $\alpha_{s'} = 1$ if $t \blacktriangleleft_\varepsilon s'$ and $\alpha_{s'} = 0$ otherwise. For each inner node where s' has the form $f(v_1, \ldots, v_k)$, we set $\beta_{s'} = \sum_{1 \leq j \leq k} \alpha_{v_j}$, and then check whether there is an occurrence of t at the root of s', i.e., whether $t \blacktriangleleft_\varepsilon s'$. If not, then we simply set $\alpha_{s'} = \beta_{s'} = \sum_{1 \leq j \leq k} \alpha_{v_j}$. If there is an occurrence of t at the root of s', then $\alpha_{s'} = \max\{\beta_{s'}, \sum_{\pi \in \mathrm{Pos}_{\mathcal{V}(t)}} \alpha_{s'|_\pi} + 1\}$, i.e., $\alpha_{s'}$ is the maximum of $\beta_{s'}$ and the number obtained when considering the root position and maxNO for all subterms of s' corresponding to variable positions of t. In Alg. 1, we use a flag $\gamma_{s'}$ where $\gamma_{s'} = 1$ indicates that we have already computed the values $\alpha_{s'}$ and $\beta_{s'}$, and otherwise we have $\gamma_{s'} = 0$. Concerning the runtime of Alg. 1, the algorithm has to consider each subterm s' of s and check whether t matches s', which runs in time $\mathcal{O}(|s'|)$, where $|s'|$ is the number of positions in s'. So Alg. 1 runs in $\mathcal{O}(|s|^2)$.

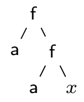
Example 9. As an example, Fig. 3 shows the dynamic programming table for the two terms $t = \mathsf{f}(\mathsf{a}, \mathsf{f}(\mathsf{a}, x))$ and $s = \mathsf{f}(\mathsf{a}, \mathsf{f}(\mathsf{a}, \mathsf{f}(\mathsf{a}, \mathsf{f}(\mathsf{f}(\mathsf{a}, \mathsf{f}(\mathsf{a}, \mathsf{a})), \mathsf{f}(\mathsf{a}, \mathsf{f}(\mathsf{a}, \mathsf{a}))))))$ computed by Alg. 1. There exists no occurrence of t in s_i for any $1 \leq i \leq 11$. For s_{12} and s_{13} there is one occurrence of t at the root. Therefore, we have $\alpha_{s_{12}} = \alpha_{s_{13}} = 1$. At s_{14} we have no occurrence of t at the root, but two occurrences below the root, thus $\alpha_{s_{14}} = \beta_{s_{14}} = 2$. The same holds for s_{15}. Then, we have another occurrence at the root of s_{16} that does not overlap with the occurrences at the roots of s_{12} and

Subterm s'	$\alpha_{s'}$	$\beta_{s'}$
s_1–s_{11}	0	0
s_{12}–s_{13}	1	0
s_{14}	2	2
s_{15}	2	2
s_{16}	3	2
s_{17}	3	3

(a) Dynamic Programming Table



(b) Tree Structure of s



(c) Tree Structure of t

Fig. 3: Computation of $\mathrm{maxNO}(t, s)$

s_{13}, so $\alpha_{s_{16}} = 3$. Finally, there is another occurrence at the root of s_{17}, but it is overlapping (w.r.t. t) with the preceding occurrence at the root of s_{16}. Thus, we obtain $\mathrm{maxNO}(t, s) = \alpha_s = \alpha_{s_{17}} = 3$.

We now embed random walks based on the maximal number of non-overlapping occurrences of a looping term t. In general, we start with a $\mathcal{P}$-RST $\mathfrak{T}$ with $\mathrm{root}(\mathfrak{T}) = t$ where t is *linear* (we will explain the reason for the linearity requirement in Ex. 13). As usual, a term $t \in \mathcal{T}$ is *linear* if $|t|_x \leq 1$ for all $x \in \mathcal{V}$, where $|t|_x$ denotes the number of positions $\pi \in \mathrm{Pos}(t)$ such that $t|_\pi = x$.

Then, we extend the rewrite sequence tree $\mathfrak{T}$, i.e., we rewrite the terms in its leaves, until we reach a point where there are enough non-overlapping occurrences of t in the leaves in order to disprove `AST` or `PAST`. In practice, one stops after reaching a suitable finite RST $\mathfrak{T}$, but the following theorem also holds for infinite RSTs $\mathfrak{T}$. We can then use the (finite) tree $\mathfrak{T}$ as a (finite) representation of an infinite $\mathcal{P}$-RST $\mathfrak{T}^\infty$, where we can embed a random walk that disproves `AST` or `PAST`.

Theorem 10 (Embedding Random Walks via Occurrences (1)). *Let $\mathcal{P}$ be a PTRS and let $\mathfrak{T}$ be a $\mathcal{P}$-RST with $\mathrm{h}(\mathfrak{T}) > 0$ and $\mathrm{root}(\mathfrak{T}) = t$, where t is linear. If we have $\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxNO}(t, t_v) > 1$, then $\mathcal{P}$ is not* `AST`. *Moreover, if $\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxNO}(t, t_v) \geq 1$, then $\mathcal{P}$ is not* `PAST`.

So for every leaf v, we multiply its probability p_v with the number of non-overlapping occurrences of t in the term t_v of the leaf. This number of occurrences in the leaves gives rise to a random walk μ with $\mu(x) = \sum_{v \in \mathrm{Leaf}(\mathfrak{T}), x = \mathrm{maxNO}(t, t_v) - 1} p_v$ for all $x \in \mathbb{Z}$. By repeatedly rewriting an innermost[2] occurrence of t in the leaves of the tree according to the rules used in $\mathfrak{T}$, we obtain an infinite $\mathcal{P}$-RST $\mathfrak{T}^\infty$, where we can embed the computation of the random walk μ. This yields the desired lower bound. Note that it is not a problem if different instantiations of t occur at different positions in a leaf. The reason is that if t starts a loop, then so does every instantiation of t.

Example 11 (Embedding via Occurrences and Innermost Rewriting). Consider the PTRS $\mathcal{P}_3$ with the rules $\mathsf{g}(x) \to \{1/2 : x, 1/2 : \mathsf{f}(x)\}$ and $\mathsf{f}(x) \to \{1 : \mathsf{g}(\mathsf{g}(x))\}$. If

[2] The innermost strategy must be used due to rules where a variable occurs more often on the left-hand side than on the right-hand side. Such rules might decrease the number of occurrences of t in the arguments when rewriting at a non-innermost position.

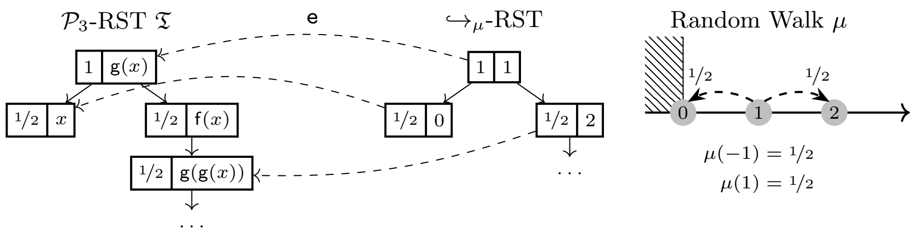


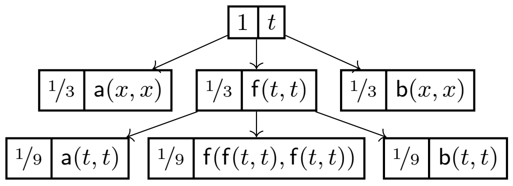
Fig. 4: Embedding of a Random Walk into a $\mathcal{P}_3$-RST to Disprove PAST of $\mathcal{P}_3$

we count the occurrences of $\mathsf{g}(x)$, then the $\mathcal{P}_3$-RST $\mathfrak{T}$ gives rise to the symmetric random walk μ both depicted in Fig. 4, which represents a "lower bound" for the termination behavior of $\mathfrak{T}$. Here, we represent μ as a PARS $(\mathbb{Z}, \hookrightarrow_\mu)$ with $y \hookrightarrow_\mu \{(p : y + \mathrm{maxNO}(t, t_v) - 1) \mid v \in \mathrm{Leaf}(\mathfrak{T})\}$ for every $y > 0$. The $\hookrightarrow_\mu$-RST with infinite expected derivation length can be embedded in the tree that results from extending $\mathfrak{T}$ by rewriting the innermost subterm $\mathsf{g}(x)$ repeatedly according to the rules used in $\mathfrak{T}$. So since μ is not PAST, $\mathcal{P}_3$ is not PAST either. However, since μ is AST, this does not yield any information about whether $\mathcal{P}_3$ is AST.

Example 12 (Non-Overlappingness is Required). Non-overlappingness of the different occurrences of t in a leaf guarantees that rewriting an innermost occurrence of t does not interfere with the possibility to rewrite the other occurrences of t later on. To see this, consider the PTRS $\mathcal{P}_4$ with the rule $\mathsf{g}(\mathsf{g}(x)) \to \{1/3 : x, 2/3 : \mathsf{g}(\mathsf{g}(\mathsf{g}(x)))\}$, which is AST. If one counted both occurrences of $\mathsf{g}(\mathsf{g}(x))$ in the term $\mathsf{g}(\mathsf{g}(\mathsf{g}(x)))$ in spite of their overlap, then one could embed the random walk μ_3 from Fig. 1, and thus, falsely disprove AST of $\mathcal{P}_4$. Here, the problem is that rewriting the innermost subterm $\mathsf{g}(\mathsf{g}(x))$ of $\mathsf{g}(\mathsf{g}(\mathsf{g}(x)))$ could yield $\mathsf{g}(x)$, i.e., then the outermost occurrence of $\mathsf{g}(\mathsf{g}(x))$ in $\mathsf{g}(\mathsf{g}(\mathsf{g}(x)))$ would be "destroyed".

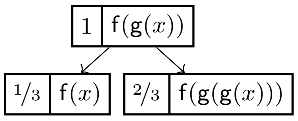
Example 13 (Linearity of t is Required). Linearity of t is required in Thm. 10, because otherwise rewriting an innermost occurrence of t in a leaf may "destroy" other occurrences of t in that leaf. As an example, consider the PTRS $\mathcal{P}_5$ with the rule $\mathsf{f}(x,x) \to \{1/3 : \mathsf{a}, 1/3 : \mathsf{b}, 1/3 : \mathsf{f}(\mathsf{f}(x,x), \mathsf{f}(x,x))\}$. If we count the occurrences of the term $\mathsf{f}(x,x)$, then the $\mathcal{P}_5$-RST $\mathfrak{T}$ where we perform a single rewrite step starting in $\mathsf{f}(x,x)$ gives rise to the random walk μ_5 with $\mu_5(-1) = 2/3$ and $\mu_5(2) = 1/3$, since $\mathrm{maxNO}(\mathsf{f}(x,x), \mathsf{f}(\mathsf{f}(x,x), \mathsf{f}(x,x))) = 3$. Since μ_5 is not PAST, we would falsely disprove PAST of $\mathcal{P}_5$. But in fact, $\mathcal{P}_5$ is PAST. The problem is that rewriting the proper subterms of $\mathsf{f}(\mathsf{f}(x,x), \mathsf{f}(x,x))$ may yield terms like $\mathsf{f}(\mathsf{a}, \mathsf{b})$, where the two arguments of f are not equal. Thus, rewriting an innermost occurrence of $t = \mathsf{f}(x,x)$ in $\mathsf{f}(\mathsf{f}(x,x), \mathsf{f}(x,x))$ may "destroy" the occurrence of t at the root.

So when rewriting innermost occurrences of t according to the rules used in $\mathfrak{T}$, we need linearity of t. Instead, one could also try to rewrite outermost occurrences. Then, instead of linearity of t, we have to require that there is no leaf $v \in \mathrm{Leaf}(\mathfrak{T})$ where a variable occurs more often in the looping term t than in the term t_v of the leaf v. A $\mathcal{P}$-RST $\mathfrak{T}$ is ***non-variable-decreasing*** (nvd) if it satisfies $|\mathrm{root}(\mathfrak{T})|_x \leq |t_v|_x$ for all $v \in \mathrm{Leaf}(\mathfrak{T})$ and all $x \in \mathcal{V}$. Rewriting an outermost occurrence of t according to an nvd $\mathcal{P}$-RST $\mathfrak{T}$ does not affect any other non-overlapping occurrences of t.



1 | t
$^1/_3$ | $\mathsf{a}(x,x)$
$^1/_3$ | $\mathsf{f}(t,t)$
$^1/_3$ | $\mathsf{b}(x,x)$
$^1/_9$ | $\mathsf{a}(t,t)$
$^1/_9$ | $\mathsf{f}(\mathsf{f}(t,t),\mathsf{f}(t,t))$
$^1/_9$ | $\mathsf{b}(t,t)$

Fig. 5: $\mathcal{P}_5'$-RST



1 | $\mathsf{f}(\mathsf{g}(x))$
$^1/_3$ | $\mathsf{f}(x)$
$^2/_3$ | $\mathsf{f}(\mathsf{g}(\mathsf{g}(x)))$

Fig. 6: $\mathcal{P}_6$-RST

Theorem 14 (Embedding Random Walks via Occurrences (2)). *Let $\mathcal{P}$ be a PTRS and let $\mathfrak{T}$ be an* nvd *$\mathcal{P}$-RST with $\mathrm{h}(\mathfrak{T}) > 0$ and $\mathrm{root}(\mathfrak{T}) = t$. If we have $\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxNO}(t, t_v) \underset{(-)}{\geq} 1$, then $\mathcal{P}$ is not* (P)`AST`.

Example 15 (Embedding via Occurrences and Outermost Rewriting). Consider the PTRS $\mathcal{P}_5'$ (similar to $\mathcal{P}_5$ from Ex. 13) with the rule $\mathsf{f}(x,x) \to \{^1/_3 : \mathsf{a}(x,x), {}^1/_3 : \mathsf{b}(x,x), {}^1/_3 : \mathsf{f}(\mathsf{f}(x,x),\mathsf{f}(x,x))\}$. Rewriting at the outermost position yields the nvd $\mathcal{P}_5'$-RST $\mathfrak{T}$ in Fig. 5, where $t = \mathsf{f}(x,x)$. Because of $\mathrm{maxNO}(t, \mathsf{a}(t,t)) = \mathrm{maxNO}(t, \mathsf{b}(t,t)) = 2$, $\mathrm{maxNO}(t, \mathsf{f}(\mathsf{f}(t,t),\mathsf{f}(t,t))) = 7$, and $\mathrm{maxNO}(t, \mathsf{a}(x,x)) = \mathrm{maxNO}(t, \mathsf{b}(x,x)) = 0$, we obtain $\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxNO}(t, t_v) = {}^{11}/_9 > 1$. Thus, by Thm. 14 this disproves `AST` of $\mathcal{P}_5'$.

If t is not linear and the RST $\mathfrak{T}$ is not nvd, then we can only count ***o****rthogonal* **o**ccurrences. The maximal number of orthogonal occurrences of t in s is $\mathrm{maxOO}(t,s) = \max\{|S| \mid S \in \mathrm{NO}(t,s), \forall \pi_1, \pi_2 \in S \text{ with } \pi_1 \neq \pi_2 : \pi_1 \perp \pi_2\}$.

To compute $\mathrm{maxOO}(t,s)$, we can adjust Alg. 1 at Line 12 in the case of $t \trianglelefteq_\varepsilon s'$. Instead of setting $\alpha_{s'}$ to the maximum of $\beta_{s'}$ and $\sum_{\pi \in \mathrm{Pos}_\mathcal{V}(t)} \alpha_{s'|_\pi} + 1$, we set $\alpha_{s'}$ to $\max\{\beta_{s'}, 1\}$, because now we do not count occurrences below another occurrence anymore. The runtime of the adjusted algorithm is still in $\mathcal{O}(|s|^2)$.

Theorem 16 (Embedding Random Walks via Occurrences (3)). *Let $\mathcal{P}$ be a PTRS and let $\mathfrak{T}$ be a $\mathcal{P}$-RST with $\mathrm{h}(\mathfrak{T}) > 0$ and $\mathrm{root}(\mathfrak{T}) = t$. If we have $\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxOO}(t, t_v) \underset{(-)}{\geq} 1$, then $\mathcal{P}$ is not* (P)`AST`.

To automate Thm. 10, 14, and 16, we have to find a $\mathcal{P}$-RST satisfying one of the two constraints. To this end, we first search for a term t that loops at all. Here, we reuse the non-probabilistic loop detection algorithms from [14] on the non-probabilistic variant $\mathrm{np}(\mathcal{P})$. After finding a loop of $\mathrm{np}(\mathcal{P})$ starting with a term t, we first check whether t is linear. If t is linear, then we reconstruct the corresponding $\mathcal{P}$-RST $\mathfrak{T}$ for this path. On all remaining terms u in the leaves of $\mathfrak{T}$, we check whether we can possibly reach a term s from u such that $t \trianglelefteq_\pi s$ for some $\pi \in \mathrm{Pos}(s)$. This is undecidable in general, but we use the *symbol transition graph* from [38] as a sufficient criterion. On those terms u where we may potentially reach such a term s, we extend the RST by performing further *rewrite steps*, in order to obtain leaves that contain more occurrences of t. This is performed repeatedly until we have constructed an RST that satisfies one of the conditions of Thm. 10 or until one reaches a certain threshold for the number of rewrite steps. In the latter case, we try to find another looping term to generate an RST in order to embed a suitable random walk. If the looping term t is not linear, then we proceed in an analogous way in order to apply Thm. 14 or Thm. 16, depending on whether the initial tree $\mathfrak{T}$ is nvd or not. In the former case we compute both values $\mathrm{maxNO}(t, t_v)$ and $\mathrm{maxOO}(t, t_v)$ for every leaf v, since rewriting the leaves of $\mathfrak{T}$ may yield a tree that is not nvd anymore.

5 Embedding Random Walks Based on Pattern Terms

Instead of counting the occurrences of a single term, we can also count the number of *instantiations* applied to a certain *base term*. To simplify the presentation, we only consider orthogonal occurrences in this section. However, similar to Sect. 4, corresponding approaches that consider pairwise non-overlapping occurrences and use innermost or outermost rewriting are possible as well.

Example 17. Consider $\mathcal{P}_6 = \{\mathsf{f}(\mathsf{g}(x)) \to \{1/3 : \mathsf{f}(x), 2/3 : \mathsf{f}(\mathsf{g}(\mathsf{g}(x)))\}\}$ modeling a positively biased random walk on the number of g's *directly below an* f in a term and the corresponding $\mathcal{P}_6$-RST in Fig. 6. Here, it is not enough to count the occurrences of $\mathsf{f}(\mathsf{g}(x))$ to disprove `AST`, but instead we have to count the occurrences of $\mathsf{g}(x)$ below an f symbol. Moreover, note that the f in $\mathsf{f}(x)$ is crucial. The PTRS $\mathcal{P}_6'$ with the rule $\mathsf{f}(\mathsf{g}(x)) \to \{1/3 : x, 2/3 : \mathsf{f}(\mathsf{g}(\mathsf{g}(x)))\}$ does not model a random walk anymore, since from any term $\mathsf{f}(\mathsf{g}^n(x))$ we have the option to directly stop and rewrite to $\mathsf{g}^{n-1}(x)$ (removing the outer f). So $\mathcal{P}_6$ is not `AST`, but $\mathcal{P}_6'$ is even `PAST`.

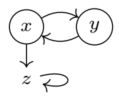
We formalize this with the concept of a *pattern term* $\langle t, \sigma\rangle$ that represents all terms which result from applying the *pumping substitution* σ repeatedly to the *base term* t, i.e., $\langle t, \sigma\rangle$ represents the terms t, $t\sigma$, $t\sigma^2$, etc. For example, if $t = \mathsf{f}(x)$ and $\sigma = [x/\mathsf{g}(x)]$, then $\langle t, \sigma\rangle$ represents the terms $\mathsf{f}(x)$, $\mathsf{f}(\mathsf{g}(x))$, $\mathsf{f}(\mathsf{g}(\mathsf{g}(x)))$, etc. A similar notion was defined in [10] to disprove *non-looping non-termination*. In contrast, here we want to *count loops*, and we want to count how often the substitution σ is applied. Therefore, we have to distinguish $t\sigma^m$ and $t\sigma^{m'}$ whenever $m \neq m'$.

Definition 18 (Pattern Term). *A pair $\langle t, \sigma\rangle$ is a* pattern term *if $t\sigma^m \neq t\sigma^{m'}$ holds whenever $m \neq m'$. We call t the* base term *and σ the* pumping substitution.

To check whether $\langle t, \sigma\rangle$ is a pattern term, we use the *variable transition graph*.

Definition 19 (Variable Transition Graph). *For any substitution σ, G_σ is the graph that has all variables as nodes and there is an edge from x to y if $y \in \mathcal{V}(x\sigma)$. For any term t, the* variable transition graph *of σ w.r.t. t is the subgraph $G_{\sigma,t}$ of G_σ which contains only those nodes that are reachable from a node in $\mathcal{V}(t)$.*

Example 20. Let $t = \mathsf{f}(x, y)$ and $\sigma = [x/\mathsf{f}(z, y), y/x]$. The graph $G_{\sigma,t}$ is shown on the right. There is an edge from z to z because $z \notin \mathrm{dom}(\sigma)$, and hence, $z\sigma = z$. The nodes from $\mathcal{V}(t) = \{x, y\}$ are marked by circles.



The variable transition graph yields a computable criterion to detect pattern terms.

Lemma 21 (Detecting Pattern Terms). *For a term t and a substitution σ, $\langle t, \sigma\rangle$ is a pattern term iff some cycle of $G_{\sigma,t}$ contains a variable x such that $x\sigma \notin \mathcal{V}$.*

Example 22. The pair $\langle t, \sigma\rangle$ with $t = \mathsf{f}(x, y)$ and $\sigma = [x/\mathsf{f}(z, y), y/x]$ is a pattern term, because $G_{\sigma,t}$ has a cycle that contains the variable x and $x\sigma = \mathsf{f}(z, y) \notin \mathcal{V}$.

Next, we consider the problem of counting orthogonal pattern term occurrences.

Definition 23 (Pattern Occurrences). *Let $\langle t, \sigma \rangle$ be a pattern term and let $s \in \mathcal{T}$ be a term. We say that $\langle t, \sigma \rangle$* occurs with multiplicity $m_\pi \in \mathbb{N}$ *at position π in s (denoted by $\langle t, \sigma \rangle \trianglelefteq_\pi^{m_\pi} s$) if there is an occurrence $t \trianglelefteq_\pi s$ and m_π is the maximal number such that $t\sigma^{m_\pi} \trianglelefteq_\pi s$. For every set $S \in \mathrm{NO}(t, s)$, let $m_S = \sum_{\pi \in S} m_\pi$. Let* $\mathrm{maxOPO}(t, \sigma, s) = \max\{m_S \mid S \in \mathrm{NO}(t, s), \forall \pi_1, \pi_2 \in S$ *with* $\pi_1 \neq \pi_2 : \pi_1 \perp \pi_2\}$ *denote the maximal value that one can obtain by adding all multiplicities for a set S of pairwise orthogonal occurrences of $\langle t, \sigma \rangle$ in s.*

Example 24. As an example, consider the term $t = \mathsf{f}(x)$, the substitution $\sigma = [x/\mathsf{g}(x)]$, and $s = \mathsf{c}(\mathsf{f}(\mathsf{g}(x)), \mathsf{f}(\mathsf{g}(\mathsf{g}(x))))$. Then $\langle t, \sigma \rangle$ occurs in s at position 1 with multiplicity 1, and at position 2 with multiplicity 2. Thus, we have $m_1 = 1$ and $m_2 = 2$. Since $\mathrm{NO}(t, s) = \{\varnothing, \{1\}, \{2\}, \{1, 2\}\}$ and both occurrences are at orthogonal positions, we have $\mathrm{maxOPO}(t, \sigma, s) = m_{\{1,2\}} = 1 + 2 = 3$.

To compute $\mathrm{maxOPO}(t, \sigma, s)$, we use Alg. 1 with $t\sigma$ instead of t. Moreover, we adjust Alg. 1 at Line 12 in the case of $t\sigma \trianglelefteq_\varepsilon s'$. Instead of setting $\alpha_{s'}$ to $\max\{\beta_{s'}, 1\}$ as in the computation of $\mathrm{maxOO}(t\sigma, s)$, we set $\alpha_{s'}$ to $\max\{\beta_{s'}, m\}$ where m is the multiplicity of the occurrence of $\langle t, \sigma \rangle$ at the root of s'. Note that we only consider orthogonal occurrences here. So if we consider the occurrence at the root of s', then occurrences below the root are ignored. To find m, we check for occurrences of $t\sigma$, $t\sigma^2$, …, $t\sigma^{|s|}$. Compared to Alg. 1, we have to perform at most $|s| - 1$ additional matching checks in each iteration, resulting in a runtime of $\mathcal{O}(|s|^3)$.

Example 25. Consider the pattern term $\langle t', \sigma \rangle$ with base term $t' = \mathsf{f}(\mathsf{a}, x)$ and pumping substitution $\sigma = [x/\mathsf{f}(\mathsf{a}, x)]$, and let $t = \mathsf{f}(\mathsf{a}, \mathsf{f}(\mathsf{a}, x))$ and s be as in Ex. 9. Note that $t = \mathsf{f}(\mathsf{a}, x)[x/\mathsf{f}(\mathsf{a}, x)]$. We have occurrences of $\langle t', \sigma \rangle$ with multiplicity 0 at s_{10}, s_{11}, and s_{15}; with multiplicity 1 at s_{12}, s_{13}, and s_{16}; and with multiplicity 2 at the root s_{17}. Since we only consider occurrences at orthogonal positions, the maximum is obtained by adding the multiplicities for the orthogonal subterms s_{12} and s_{13} or by considering the multiplicity at the root. Thus, we obtain $\mathrm{maxOPO}(t', \sigma, s) = \alpha_s = \alpha_{s_{17}} = 2$.

As demonstrated by the PTRS $\mathcal{P}_6'$ in Ex. 17, if a term $t\sigma^m$ can be rewritten to a term without any occurrence of t, then this does not mean that the multiplicity is reduced by m, but it may mean that one directly reaches a normal form. Hence, then this pattern cannot be used to disprove `(P)AST`. So a pattern $\langle t, \sigma \rangle$ may only be used for disproving `(P)AST` if we have a $\mathcal{P}$-RST where every leaf contains an occurrence of t. Moreover, if we have a $\mathcal{P}$-RST $\mathfrak{T}$ that starts with $t\sigma^1$ and has an occurrence $\langle t, \sigma \rangle \trianglelefteq_\pi^q t_v$ in a leaf v, then we also need that the tree can be "generalized" from 1 to an arbitrary multiplicity $m \in \mathbb{N}_{>0}$, i.e., rewriting the term $t\sigma^m$ using the same rules as in $\mathfrak{T}$ at the same positions must result in a leaf v' with an occurrence $\langle t, \sigma \rangle \trianglelefteq_\pi^{q+m-1} t_{v'}$. For any occurrence $\langle t, \sigma \rangle \trianglelefteq_\pi^q t_v$, let $\kappa_{v,\pi}$ be the substitution such that $t\sigma^q \kappa_{v,\pi} = t_v|_\pi$. Then we ensure this generalization property by requiring commutation of σ with all these substitutions $\kappa_{v,\pi}$.

Definition 26 (Pattern Tree). *Let $\mathcal{P}$ be a PTRS, let $\mathfrak{T}$ be a $\mathcal{P}$-RST, and let $\langle t, \sigma \rangle$ be a pattern term. $\mathfrak{T}$ is a $\mathcal{P}$*-pattern tree *for $\langle t, \sigma \rangle$ if $\mathrm{root}(\mathfrak{T}) = t\sigma$, for every* *leaf $v \in \mathrm{Leaf}(\mathfrak{T})$ there exists an occurrence $t \trianglelefteq_\pi t_v$, and whenever $t\sigma^q\kappa_{v,\pi} = t_v|_\pi$ for some $q \geq 0$, some $\pi \in \mathrm{Pos}(t_v)$, and some substitution $\kappa_{v,\pi}$, then the pumping substitution σ commutes with $\kappa_{v,\pi}$, i.e., $\sigma\kappa_{v,\pi} = \kappa_{v,\pi}\sigma$.*

For Ex. 17, the $\mathcal{P}_6$-RST in Fig. 6 is a pattern tree for the pattern term $\langle t, \sigma\rangle$ where $t = \mathsf{f}(x)$ and $\sigma = [x/\mathsf{g}(x)]$. Here, the substitutions $\kappa_{v_1,\varepsilon}$ and $\kappa_{v_2,\varepsilon}$ for the two leaves are the identity, which commutes with σ. Indeed, this tree can be "generalized" to arbitrary multiplicities $m > 0$, because the corresponding $\mathcal{P}_6$-RST starting with $\mathsf{f}(\mathsf{g}^m(x))$ at the root has $\mathsf{f}(\mathsf{g}^{m-1}(x))$ and $\mathsf{f}(\mathsf{g}^{2+m-1}(x))$ in its leaves.

Similar to Thm. 16, by rewriting orthogonal occurrences, we result in a technique to embed random walks via patterns in a $\mathcal{P}$-RST.

Theorem 27 (Embedding RWs via Patterns). *Let $\mathcal{P}$ be a PTRS, $\langle t, \sigma\rangle$ be a pattern term, and let $\mathfrak{T}$ be a $\mathcal{P}$-pattern tree for $\langle t, \sigma\rangle$ with $\mathrm{h}(\mathfrak{T}) > 0$. If we have $\sum_{v\in\mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxOPO}(t, \sigma, t_v) \underset{(-)}{>} 1$, then $\mathcal{P}$ is not* `(P)AST`.

Example 28. Using Thm. 27, we can now embed the positively biased random walk μ_3 from Fig. 1 with $\mu_3(-1) = {}^1\!/_3$ and $\mu_3(1) = {}^2\!/_3$ in the $\mathcal{P}_6$-RST of Fig. 6 and disprove `AST`.

Example 29. By counting the number of applications of the pumping substitution $\sigma = [x/\mathsf{g}(y), y/\mathsf{f}(x)]$ to the base term $t = \mathsf{c}(y, x)$, we can disprove `AST` of the PTRS $\mathcal{P}_7$ with the only rule $\mathsf{c}(\mathsf{f}(x), \mathsf{g}(y)) \to \{{}^1\!/_3 : \mathsf{c}(y, x), {}^2\!/_3 : \mathsf{c}(\mathsf{f}(\mathsf{g}(y)), \mathsf{g}(\mathsf{f}(x)))\}$ via Thm. 27. Again, we can embed the random walk μ_3 in the RST corresponding to the only rewrite rule, because the child $t = \mathsf{c}(y, x)$ has the probability ${}^1\!/_3$, and the second child $t\sigma^2 = \mathsf{c}(\mathsf{f}(\mathsf{g}(y)), \mathsf{g}(\mathsf{f}(x)))$ has the probability ${}^2\!/_3$.

Remark 30. Note that Thm. 27 is *not* a generalization of Thm. 16. We can disprove `PAST` of the PTRS $\mathcal{P}_8$ with the rule $\mathsf{g} \to \{{}^1\!/_2 : \mathsf{a}, {}^1\!/_2 : \mathsf{c}(\mathsf{g}, \mathsf{g})\}$ via Thm. 16 by simply taking the tree $\mathfrak{T}$ corresponding to the only rewrite rule and counting the occurrences of g. However, there is no pattern term $\langle t, \sigma\rangle$ such that $t\sigma = \mathsf{g}$. Indeed, `PAST` of $\mathcal{P}_8$ cannot be disproven via Thm. 27.

To automate Thm. 27, we adapt our implementation of Thm. 16. After finding the pattern $t\sigma$, we have to rewrite the leaves until we obtain a pattern tree. Note that we can directly stop if our reachability analysis shows that some leaf cannot reach a term containing an occurrence of t.

6 Evaluation and Conclusion

We presented the first techniques to disprove `(P)AST` of PTRSs automatically. To this end, we embed random walks in rewrite sequence trees, based on counting occurrences of terms or multiplicities of patterns. In this way, qualitative approaches to detect non-termination of non-probabilistic TRSs based on loops or pattern terms can now be lifted to the probabilistic setting, where a quantitative analysis is required. Our approach can be based on any algorithm to detect loops for standard non-probabilistic TRSs.

We implemented our new contributions in our termination prover AProVE [15]. Currently, we run our techniques to prove and to disprove termination of PTRSs in parallel and stop once one of the techniques succeeds. For proving termination, we use the probabilistic *dependency pair* (DP) framework [26, 27], which modularizes termination proofs such that one can apply different techniques to different subproblems. In the future, we plan to integrate our new techniques to disprove AST or PAST into the DP frameworks for AST [27] and PAST [26], respectively. Then the DP framework can help to detect those parts of a PTRS which are potentially non-terminating such that one can restrict the search for non-termination proofs to these parts. Moreover, we also plan to analyze whether there are interesting subclasses of PTRSs where AST or PAST is decidable.

To evaluate the power of our new techniques, we used the benchmark set of all 138 PTRSs from the *Termination Problem Data Base* [41], i.e., the benchmarks used for the annual *Termination Competition* [16]. They contain 138 typical probabilistic programs, including examples with complicated probabilistic structure and probabilistic algorithms on lists and trees. Note that this set was mainly developed to evaluate techniques that can prove AST or PAST. Therefore, most of the examples are indeed AST, and we added 20 more examples that are not AST or not PAST, which express typical bugs in implementations.

Category	Thm. 4	Thm. 10	Thm. 14	Thm. 16	Thm. 27	AProVE
AST	8	18	19	16	13	25
PAST	8	33	34	28	27	49

We performed our experiments on a computer with an Apple M4 CPU and 16 GB of RAM, and a timeout of 30 seconds was used for each example. The table above shows the individual results for each of our novel theorems, and "AProVE" denotes the combination of all techniques as implemented in our tool. Note that AProVE can *prove* AST for 69 of the 158 examples and *prove* PAST for 32 of the 158 examples. So at most $158 - 69 = 89$ examples may be non-AST and AProVE can disprove AST for 25 of them. Similarly, at most 126 examples may be non-PAST and AProVE disproves PAST for 49 of them.

The experimental results for Thm. 4 confirm that one indeed needs more elaborate techniques than just a direct lifting of the loop detection technique to the probabilistic setting. Our experiments show that each of our theorems has its own benefits. More precisely, for each of the five theorems, there exist examples that can only be solved by this theorem but not by any of the other four theorems.

Since ours is the first approach to disprove (P)AST of PTRSs automatically, we could not compare with other tools for termination analysis of PTRSs. While there exist techniques [8, 39] and the tool Amber [33] for disproving (P)AST of imperative programs, an experimental comparison would be problematic due to the fundamental differences between the considered languages.

For further details on our experiments and precise results for each theorem on each benchmark, to access the collection of benchmarks, and for instructions on how to run AProVE via its *web interface* or locally, we refer to: https://aprove-developers.github.io/DisprovingPTRSTermination/

Acknowledgements. We thank Florian Frohn and Carsten Fuhs for a joint discussion on initial ideas for this paper during CADE '23 in Rome.

References

[1] M. Avanzini, U. Dal Lago, and A. Yamada. "On Probabilistic Term Rewriting". In: *Sci. Comput. Program.* 185 (2020). DOI: 10.1016/j.scico.2019.102338.

[2] F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998. DOI: 10.1017/CBO9781139172752.

[3] D. Beyer and J. Strejček. "Improvements in Software Verification and Witness Validation: *SV-COMP* 2025". In: *Proc. TACAS '25.* LNCS 15698. Website of *SV-COMP*: https://sv-comp.sosy-lab.org/. 2025, pp. 151–186. DOI: 10.1007/978-3-031-90660-2_9.

[4] O. Bournez and C. Kirchner. "Probabilistic Rewrite Strategies. Applications to ELAN". In: *Proc. RTA '02.* LNCS 2378. 2002, pp. 252–266. DOI: 10.1007/3-540-45610-4_18.

[5] O. Bournez and F. Garnier. "Proving Positive Almost-Sure Termination". In: *Proc. RTA '05.* LNCS 3467. 2005, pp. 323–337. DOI: 10.1007/978-3-540-32033-3_24.

[6] M. Braverman. "Termination of Integer Linear Programs". In: *Proc. CAV '06.* LNCS 4144. 2006, pp. 372–385. DOI: 10.1007/11817963_34.

[7] M. Brockschmidt, T. Ströder, C. Otto, and J. Giesl. "Automated Detection of Non-Termination and NullPointerExceptions for Java Bytecode". In: *Proc. FoVeOOS '12.* LNCS 7421. 2012, pp. 123–141. DOI: 10.1007/978-3-642-31762-0_9.

[8] K. Chatterjee, P. Novotný, and Ð. Žikelić. "Stochastic Invariants for Probabilistic Termination". In: *Proc. POPL '17.* 2017, pp. 145–160. DOI: 10.1145/3009837.3009873.

[9] H.-Y. Chen, B. Cook, C. Fuhs, K. Nimkar, and P. O'Hearn. "Proving Nontermination via Safety". In: *Proc. TACAS '14.* LNCS 8413. 2014, pp. 156–171. DOI: 10.1007/978-3-642-54862-8_11.

[10] F. Emmes, T. Enger, and J. Giesl. "Proving Non-Looping Non-Termination Automatically". In: *Proc. IJCAR '12.* LNCS 7364. 2012, pp. 225–240. DOI: 10.1007/978-3-642-31365-3_19.

[11] J. Endrullis and H. Zantema. "Proving Non-Termination by Finite Automata". In: *Proc. RTA '15.* LIPIcs 36. 2015, pp. 160–176. DOI: 10.4230/LIPICS.RTA.2015.160.

[12] F. Frohn and C. Fuhs. "A Calculus for Modular Loop Acceleration and Non-Termination Proofs". In: *Int. J. Softw. Tools Technol. Transf.* 24 (2022), pp. 691–715. DOI: 10.1007/s10009-022-00670-2.

[13] F. Frohn and J. Giesl. "Proving Non-Termination by Acceleration Driven Clause Learning (Short Paper)". In: *Proc. CADE '23.* LNCS 14132. 2023, pp. 220–233. DOI: 10.1007/978-3-031-38499-8_13.

[14] J. Giesl, R. Thiemann, and P. Schneider-Kamp. "Proving and Disproving Termination of Higher-Order Functions". In: *Proc. FroCoS '05.* LNCS 3717. 2005, pp. 216–231. DOI: 10.1007/11559306_12.

[15] J. Giesl, C. Aschermann, M. Brockschmidt, F. Emmes, F. Frohn, C. Fuhs, J. Hensel, C. Otto, M. Plücker, P. Schneider-Kamp, T. Ströder, S. Swiderski, and R. Thiemann. "Analyzing Program Termination and Complexity

Automatically with AProVE". In: *J. Autom. Reason.* 58.1 (2017), pp. 3–31. DOI: 10.1007/s10817-016-9388-y.

[16] J. Giesl, A. Rubio, C. Sternagel, J. Waldmann, and A. Yamada. "The Termination and Complexity Competition". In: *Proc. TACAS '19.* LNCS 11429. Website of *TermComp*: https://termination-portal.org/wiki/Termination_Competition. 2019, pp. 156–166. DOI: 10.1007/978-3-030-17502-3_10.

[17] J. Giesl, P. Giesl, and M. Hark. "Computing Expected Runtimes for Constant Probability Programs". In: *Proc. CADE '19.* LNCS 11716. 2019, pp. 269–286. DOI: 10.1007/978-3-030-29436-6_16.

[18] A. D. Gordon, T. A. Henzinger, A. V. Nori, and S. K. Rajamani. "Probabilistic Programming". In: *Proc. FOSE '14.* 2014, pp. 167–181. DOI: 10.1145/2593882.2593900.

[19] G. Grimmett and D. Stirzaker. *Probability and Random Processes.* Oxford University Press, 2020.

[20] A. Gupta, T. A. Henzinger, R. Majumdar, A. Rybalchenko, and R.-G. Xu. "Proving Non-Termination". In: *Proc. POPL '08.* 2008, pp. 147–158. DOI: 10.1145/1328438.1328459.

[21] R. Gutiérrez and S. Lucas. "Automatically Proving and Disproving Feasibility Conditions". In: *Proc. IJCAR '20.* LNCS 12167. 2020, pp. 416–435. DOI: 10.1007/978-3-030-51054-1_27.

[22] M. Hark, F. Frohn, and J. Giesl. "Termination of Triangular Polynomial Loops". In: *Formal Methods Syst. Des.* 65.1 (2025), pp. 70–132. DOI: 10.1007/S10703-023-00440-Z.

[23] M. Hosseini, J. Ouaknine, and J. Worrell. "Termination of Linear Loops over the Integers". In: *Proc. ICALP '19.* LIPIcs 132. 2019. DOI: 10.4230/LIPIcs.ICALP.2019.118.

[24] J.-C. Kassing and J. Giesl. "Proving Almost-Sure Innermost Termination of Probabilistic Term Rewriting Using Dependency Pairs". In: *Proc. CADE '23.* LNCS 14132. 2023, pp. 344–364. DOI: 10.1007/978-3-031-38499-8_20.

[25] J. Kassing and J. Giesl. "From Innermost to Full Probabilistic Term Rewriting: Almost-Sure Termination, Complexity, and Modularity". In: *Log. Methods Comput. Sci.* 21.4 (2025). DOI: 10.46298/LMCS-21(4:28)2025.

[26] J.-C. Kassing, L. Spitzer, and J. Giesl. "Dependency Pairs for Expected Innermost Runtime Complexity and Strong Almost-Sure Termination of Probabilistic Term Rewriting". In: *Proc. PPDP '25.* 2025. DOI: 10.1145/3756907.3756917.

[27] J.-C. Kassing and J. Giesl. "The Annotated Dependency Pair Framework for Almost-Sure Termination of Probabilistic Term Rewriting". In: *Sci. Comput. Program.* 251 (2026), p. 103417. DOI: 10.1016/j.scico.2025.103417.

[28] D. Larraz, K. Nimkar, A. Oliveras, E. Rodríguez-Carbonell, and A. Rubio. "Proving Non-termination Using Max-SMT". In: *Proc. CAV '14.* LNCS 8559. 2014, pp. 779–796. DOI: 10.1007/978-3-319-08867-9_52.

[29] G. F. Lawler and V. Limic. *Random Walk: A Modern Introduction.* Cambridge University Press, June 2010.

[30] J. Leike and M. Heizmann. "Geometric Nontermination Arguments". In: *Proc. TACAS '18.* LNCS 10806. 2018, pp. 266–283. DOI: 10.1007/978-3-319-89963-3_16.

[31] N. Lommen and J. Giesl. "Targeting Completeness: Using Closed Forms for Size Bounds of Integer Programs". In: *Proc. FroCoS '23*. LNCS 14279. 2023, pp. 3–22. DOI: 10.1007/978-3-031-43369-6_1.

[32] S. Lucas and R. Gutiérrez. "Use of Logical Models for Proving Infeasibility in Term Rewriting". In: *Inf. Process. Lett.* 136 (2018), pp. 90–95. DOI: 10.1016/j.ipl.2018.04.002.

[33] M. Moosbrugger, E. Bartocci, J. Katoen, and L. Kovács. "Automated Termination Analysis of Polynomial Probabilistic Programs". In: *Proc. ESOP '21*. LNCS 12648. 2021, pp. 491–518. DOI: 10.1007/978-3-030-72019-3_18.

[34] É. Payet. "Loop Detection in Term Rewriting Using the Eliminating Unfoldings". In: *Theor. Comput. Sc.* 403 (2008), pp. 307–327. DOI: 10.1016/j.tcs.2008.05.013.

[35] É. Payet. "Non-Termination in Term Rewriting and Logic Programming". In: *J. Autom. Reason.* 68.4 (2024). DOI: 10.1007/s10817-023-09693-z.

[36] N. Saheb-Djahromi. "Probabilistic LCF". In: *Proc. MFCS '78*. LNCS 64. 1978, pp. 442–451. DOI: 10.1007/3-540-08921-7_92.

[37] F. Spitzer. *Principles of Random Walk*. Vol. 34. Springer, 2001.

[38] C. Sternagel and A. Yamada. "Reachability Analysis for Termination and Confluence of Rewriting". In: *Proc. TACAS '19*. LNCS 11427. 2019, pp. 262–278. DOI: 10.1007/978-3-030-17462-0_15.

[39] T. Takisaka, Y. Oyabu, N. Urabe, and I. Hasuo. "Ranking and Repulsing Supermartingales for Reachability in Randomized Programs". In: *ACM Trans. Program. Lang. Syst.* 43 (2021). DOI: 10.1145/3450967.

[40] A. Tiwari. "Termination of Linear Programs". In: *Proc. CAV '04*. LNCS 3114. 2004, pp. 70–82. DOI: 10.1007/978-3-540-27813-9_6.

[41] TPDB. *Termination Problem Data Base*. 2025. URL: https://github.com/TermCOMP/TPDB-ARI.

[42] M. Xu and Z.-B. Li. "Symbolic Termination Analysis of Solvable Loops". In: *J. Symb. Comput.* 50 (2013), pp. 28–49. DOI: 10.1016/j.jsc.2012.05.005.

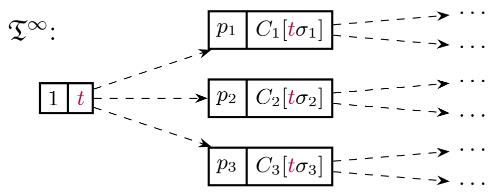
[43] A. Yamada. "Term Orderings for Non-Reachability of (Conditional) Rewriting". In: *Proc. IJCAR '22*. 13385. 2022, pp. 248–267. DOI: 10.1007/978-3-031-10769-6_15.

A Appendix

In this appendix, we give all proofs for our lemmas and theorems.

Theorem 4 (Disproving $\mathtt{AST}$ via Loops). *Let $\mathcal{P}$ be a PTRS and $\mathfrak{T}$ be a $\mathcal{P}$-RST with $\mathrm{h}(\mathfrak{T}) > 0$ such that $\mathrm{root}(\mathfrak{T}) = t$ and for every $v \in \mathrm{Leaf}(\mathfrak{T})$ there is a context C_v and a substitution σ_v such that $t_v = C_v[t\sigma_v]$. Then $\mathcal{P}$ is neither $\mathtt{AST}$ nor $\mathtt{PAST}$.*

Proof. We construct an infinite $\mathcal{P}$-RST $\mathfrak{T}^\infty$ based on $\mathfrak{T}$ where we always rewrite one of the occurrences of t in a leaf according to the rules used to generate $\mathfrak{T}$. Since $\mathfrak{T}$ has an occurrence of t in every leaf, the resulting tree $\mathfrak{T}^\infty$ has no leaves, and therefore, $\mathcal{P}$ is neither $\mathtt{AST}$ nor $\mathtt{PAST}$.



We only have to show that if we rewrite a term $s = C[t\sigma]$ for some context C and some substitution σ using the same rules as in $\mathfrak{T}$, then we result in a tree $\mathfrak{T}_s$ where all leaves contain an occurrence of t again. Let τ be the position of the hole in C. If we rewrite at position π in $\mathfrak{T}$, then we rewrite at position $\tau.\pi$ in the tree $\mathfrak{T}_s$. For every leaf $v \in \mathrm{Leaf}(\mathfrak{T})$ there is a corresponding leaf $v' \in \mathrm{Leaf}(\mathfrak{T}_s)$ with $t_{v'} = C[t_v\sigma]$. Since there exists an occurrence of t within t_v, i.e., $t_v = C'[t\sigma']$, there is an occurrence of t within $t_{v'} = C[t_v\sigma] = C[C'[t\sigma']\sigma]$ as well. □

Theorem 6 (Lower Bounds via Embeddings). *Let $(A_1, \to_1)$ and $(A_2, \to_2)$ be two PARSs, and let $\mathfrak{T}_i$ be a $\to_i$-RST for $i \in \{1, 2\}$ such that there exists an embedding from $\mathfrak{T}_1$ to $\mathfrak{T}_2$. Then, $|\mathfrak{T}_2| \leq |\mathfrak{T}_1|$ and $\mathrm{edl}(\mathfrak{T}_1) \leq \mathrm{edl}(\mathfrak{T}_2)$.*

Proof. Let $\mathrm{Pre}(v)$ be the set of all (not necessarily direct) predecessors of a node $v \in \mathrm{Leaf}(\mathfrak{T}_2)$ including the node v itself. Similarly, let $\mathrm{Post}(v)$ be the set of all (not necessarily direct) successors of a node $v \in \mathrm{Leaf}(\mathfrak{T}_2)$ including the node v itself. Finally, by $\mathrm{Post}_{\mathrm{Leaf}}(v)$ we denote the set of all leaves in $\mathrm{Post}(v)$.

We start with some basic observations.

I. Since we have $p_v = p_{\mathsf{e}(v)}$ for all $v \in \mathrm{Leaf}(\mathfrak{T}_1)$ by definition of an embedding (Def. 5) and $p_{\mathfrak{r}} = 1$ for the root node $\mathfrak{r}$ of $\mathfrak{T}_1$ by definition of a $\to$-RST, we must map the root $\mathfrak{r}$ of $\mathfrak{T}_1$ to some node $v \in V(\mathfrak{T}_2)$ with $p_v = p_{\mathsf{e}(\mathfrak{r})} = p_{\mathfrak{r}} = 1$.

II. The nodes $v_1, \ldots, v_n$ of $\mathfrak{T}_2$ with probability 1, i.e., $v \in V(\mathfrak{T}_2)$ with $p_v = 1$, form a connected path. Let v' be the node with the greatest depth within this path. Every node $w \in V(\mathfrak{T}_2)$ with $p_w < 1$ is a successor of v', i.e., $w \in \mathrm{Post}(v')$. Thus, together with I. we obtain that every node $w \in V(\mathfrak{T}_2)$ has a predecessor $v \in \mathrm{Pre}(w)$ such that $v = \mathsf{e}(u)$ for some $u \in V(\mathfrak{T}_1)$.

III. Every $w \in \operatorname{Leaf}(\mathfrak{T}_2)$ has a predecessor $v \in \operatorname{Pre}(w)$ that is the image of a leaf $u \in \operatorname{Leaf}(\mathfrak{T}_1)$ in $\mathfrak{T}_1$ (i.e., $\mathtt{e}(u) = v$).

Assume for a contradiction that this statement is false. By II. we know that there is at least some $v \in \operatorname{Pre}(w)$ that is the image of some node $u \in V(\mathfrak{T}_1)$. Let v be the predecessor with the largest depth among all such predecessors that are the image of some node $u \in V(\mathfrak{T}_1)$. If u is not a leaf, then there exist direct successors $u_1, \ldots, u_k$ of u in $\mathfrak{T}_1$ with $p_u = \sum_{j=1}^{k} p_{u_j}$. Since $\mathtt{e}$ is injective, we have $\mathtt{e}(u_j) \neq v$ for all $1 \leq j \leq k$. By definition of an embedding, we have $p_v = p_{\mathtt{e}(u)} = p_u = \sum_{j=1}^{k} p_{u_j} = \sum_{j=1}^{k} p_{\mathtt{e}(u_j)}$. Now w would be a node that is a successor of v but it does not have any $p_{\mathtt{e}(u_j)}$ as predecessor (since v was the deepest predecessor being an image of a node from $\mathfrak{T}_1$). But since the probabilities of all $\mathtt{e}(u_1), \ldots, \mathtt{e}(u_k)$ already add up to p_v, and $\mathtt{e}(u_1), \ldots, \mathtt{e}(u_k)$ must also be successors of v by the definition of an embedding, this would imply that $p_w = 0$, which is a contradiction.

Next, we show $|\mathfrak{T}_2| \leq |\mathfrak{T}_1|$. If $v \in \operatorname{Leaf}(\mathfrak{T}_2)$, then there has to exist a $w \in \operatorname{Pre}(v)$ that is the image of a leaf $u \in \operatorname{Leaf}(\mathfrak{T}_1)$ in $\mathfrak{T}_1$ (i.e., $\mathtt{e}(u) = w$). Note that by the definition of embeddings we have $p_w = p_{\mathtt{e}(u)} = p_u$. Moreover, we have $\sum_{v \in \operatorname{Post}_{\operatorname{Leaf}}(w)} p_v \leq p_w$, i.e., the sum of the probabilities of successors of w cannot be higher than the probability of p_w itself, because following the rewrite sequence cannot increase the probability. Thus, we obtain

$$\begin{array}{lll} |\mathfrak{T}_2| = \sum_{v \in \operatorname{Leaf}(\mathfrak{T}_2)} p_v & = \sum_{u \in \operatorname{Leaf}(\mathfrak{T}_1)} \sum_{v \in \operatorname{Post}_{\operatorname{Leaf}}(\mathtt{e}(u))} p_v & \\ \quad\;\; \leq \sum_{u \in \operatorname{Leaf}(\mathfrak{T}_1)} p_{\mathtt{e}(u)} & = \sum_{u \in \operatorname{Leaf}(\mathfrak{T}_1)} p_u & = |\mathfrak{T}_1|. \end{array}$$

Next, we consider the expected derivation length. If $|\mathfrak{T}_2| < 1$, then we have $\operatorname{edl}(\mathfrak{T}_2) = \infty$ and thus, the claim is trivial. Hence, let $|\mathfrak{T}_2| = 1$. Due to $|\mathfrak{T}_2| \leq |\mathfrak{T}_1|$, this also implies $|\mathfrak{T}_1| = 1$. Note that a path from the root to a leaf v in $\mathfrak{T}_2$ is at least as long as the path from the root to the corresponding leaf u in $\mathfrak{T}_1$ (where $\mathtt{e}(u) = w$ for a $w \in \operatorname{Pre}(v)$), since $\mathtt{e}$ is injective and "path preserving". Therefore, we have $\operatorname{d}(u) \leq \operatorname{d}(v)$ for all $v \in \operatorname{Post}_{\operatorname{Leaf}}(\mathtt{e}(u))$, and thus

$$\begin{aligned} \operatorname{edl}(\mathfrak{T}_2) &= \textstyle\sum_{u \in \operatorname{Leaf}(\mathfrak{T}_1)} \sum_{v \in \operatorname{Post}_{\operatorname{Leaf}}(\mathtt{e}(u))} \operatorname{d}(v) \cdot p_v \\ &\geq \textstyle\sum_{u \in \operatorname{Leaf}(\mathfrak{T}_1)} \sum_{v \in \operatorname{Post}_{\operatorname{Leaf}}(\mathtt{e}(u))} \operatorname{d}(u) \cdot p_v \\ &= \textstyle\sum_{u \in \operatorname{Leaf}(\mathfrak{T}_1)} \operatorname{d}(u) \cdot \sum_{v \in \operatorname{Post}_{\operatorname{Leaf}}(\mathtt{e}(u))} p_v \\ &= \textstyle\sum_{u \in \operatorname{Leaf}(\mathfrak{T}_1)} \operatorname{d}(u) \cdot p_{\mathtt{e}(u)} \\ &= \textstyle\sum_{u \in \operatorname{Leaf}(\mathfrak{T}_1)} \operatorname{d}(u) \cdot p_u \\ &= \operatorname{edl}(\mathfrak{T}_1) \end{aligned}$$

To conclude $\sum_{v \in \operatorname{Post}_{\operatorname{Leaf}}(\mathtt{e}(u))} p_v = p_{\mathtt{e}(u)}$ in the fourth step, note that $|\mathfrak{T}_2| = 1$ implies that we have $p_w = \operatorname{Post}_{\operatorname{Leaf}(w)} p_v$ for every node w of $\mathfrak{T}_2$. □

Theorem 10 (Embedding Random Walks via Occurrences (1)). *Let $\mathcal{P}$ be a PTRS and let $\mathfrak{T}$ be a $\mathcal{P}$-RST with $\operatorname{h}(\mathfrak{T}) > 0$ and $\operatorname{root}(\mathfrak{T}) = t$, where t is linear. If we have $\sum_{v \in \operatorname{Leaf}(\mathfrak{T})} p_v \cdot \operatorname{maxNO}(t, t_v) > 1$, then $\mathcal{P}$ is not* `AST`. *Moreover, if $\sum_{v \in \operatorname{Leaf}(\mathfrak{T})} p_v \cdot \operatorname{maxNO}(t, t_v) \geq 1$, then $\mathcal{P}$ is not* `PAST`.

Proof. If we have $|\mathfrak{T}| = \sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v < 1$, then $\mathfrak{T}$ is itself a witness that $\mathcal{P}$ is not `AST` and not `PAST`. Therefore, we now consider the case $|\mathfrak{T}| = 1$.

We define the random walk μ by $\mu(x) = \sum_{v \in \mathrm{Leaf}(\mathfrak{T}), x = \mathrm{maxNO}(t,t_v)-1} p_v$ for all $x \in \mathbb{Z}$. Note that

$$\begin{aligned}\mathbb{E}(\mu) &= \textstyle\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot (\mathrm{maxNO}(t,t_v) - 1) \\ &= (\textstyle\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxNO}(t,t_v)) - (\textstyle\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v) \\ &= (\textstyle\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxNO}(t,t_v)) - 1 \qquad\qquad (\text{since } \textstyle\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v = 1)\end{aligned}$$

Therefore, if $\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxNO}(t,t_v) > 1$, then we have $\mathbb{E}(\mu) > 0$ and by Thm. 1, μ is not `AST`. Similarly, if $\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxNO}(t,t_v) \geq 1$, then we have $\mathbb{E}(\mu) \geq 0$ and by Thm. 1, μ is not `PAST`.

We can represent μ as a PARS $(\mathbb{Z}, \hookrightarrow_\mu)$ with $y \hookrightarrow_\mu \{(p : y + \mathrm{maxNO}(t,t_v) - 1) \mid v \in \mathrm{Leaf}(\mathfrak{T})\}$ for every $y > 0$. Let $\mathfrak{T}_1$ be the infinite $\hookrightarrow_\mu$-RST that starts at 1. Note that $|\mathfrak{T}_1| < 1$ if μ is not `AST`, and $\mathrm{edl}(\mathfrak{T}_1) = \infty$ if μ is not `PAST`. The only difference between μ and the PARS $(\mathbb{Z}, \hookrightarrow_\mu)$ is that $\hookrightarrow_\mu$ is defined via multi-distributions which may have multiple different pairs $(p_1 : x), \ldots, (p_k : x)$ for the same number x, while μ maps every value x to a single probability ($\mu(x) = \sum_{i=1}^{k} p_i$). However, since for every number x there is only a single rewrite rule with x as its left-hand side, the termination probability and expected derivation length are equal for μ and $(\mathbb{Z}, \hookrightarrow_\mu)$.

We construct an infinite $\mathcal{P}$-RST $\mathfrak{T}^\infty$ based on $\mathfrak{T}$ where we always rewrite an innermost occurrence of t in a leaf according to the rules used to generate $\mathfrak{T}$. If we have $\mathrm{maxNO}(t,t_v) = 0$ in a leaf v, then it remains a leaf in $\mathfrak{T}^\infty$. We have to prove that $\mathfrak{T}^\infty$ behaves like the rewrite sequence tree of our random walk μ.

We first show that if we have a term $s \in \mathcal{T}$ with $\mathrm{maxNO}(t,s) = k \in \mathbb{N}_{>0}$, then instead of the tree $\mathfrak{T}$ whose root is labeled with t, we can construct a tree $\mathfrak{T}_s$ whose root is labeled with s, where $\mathfrak{T}_s$ is like $\mathfrak{T}$ when rewriting an innermost occurrence of t in s instead, according to the rules used in $\mathfrak{T}$. Thus, we can define a bijective embedding $\mathtt{e}_s : V(\mathfrak{T}) \to V(\mathfrak{T}_s)$ and obtain $\mathrm{maxNO}(t, t_{\mathtt{e}_s(v)}) \geq \mathrm{maxNO}(t,t_v) + k - 1$ for all leaves $v \in \mathrm{Leaf}(\mathfrak{T})$. (While k was the original number of occurrences of t in s, this number is now modified according to the random walk μ.) To prove this inequation, let $\{\pi_1, \ldots, \pi_k\} \in \mathrm{NO}(t,s)$ be a witness for the k pairwise non-overlapping occurrences of t in s. The term s has the form $s = C[t\delta]$ for some context C with a hole at position π_1 and substitution δ. Let π_1 be the position of an innermost occurrence of t, i.e., there is no π_j with $1 \leq j \leq k$ strictly below π_1. To construct $\mathfrak{T}_s$, we rewrite $C[t\delta]$ according to $\mathfrak{T}$ at position π_1. This is possible, since if t can be rewritten via the rules used in $\mathfrak{T}$ then so does every instantiation of t. Let $v \in \mathrm{Leaf}(\mathfrak{T})$ and $\mathtt{e}_s(v) \in \mathrm{Leaf}(\mathfrak{T}_s)$ be the corresponding leaf in $\mathfrak{T}_s$ which we get by following the same path in $\mathfrak{T}_s$ as in $\mathfrak{T}$. We have $t_{\mathtt{e}_s(v)} = C[t_v\delta]$. Since t is linear and $\{\pi_1, \ldots, \pi_k\}$ are pairwise non-overlapping w.r.t. t, we still have $\{\pi_2, \ldots, \pi_k\} \in \mathrm{NO}(t, C[t_v\delta])$. Moreover, all occurrences of t in t_v also exist in $t_v\delta$. Finally, the occurrences of t in $t_v\delta$ are non-overlapping with $\{\pi_2, \ldots, \pi_k\}$ because they are below π_1. Thus, we get $\mathrm{maxNO}(t, t_{\mathtt{e}_s(v)}) = \mathrm{maxNO}(t, C[t_v\delta]) = \mathrm{maxNO}(t, t_v\delta) + k - 1 \geq \mathrm{maxNO}(t,t_v) + k - 1$.

Now we define the embedding $\mathtt{e} : V(\mathfrak{T}_1) \to V(\mathfrak{T}^\infty)$ from the $\hookrightarrow_\mu$-RST by induction on the depth of the node in $\mathfrak{T}_1$. Moreover, for every node $w \in V(\mathfrak{T}_1)$ that is labeled with the number a_w in $\mathfrak{T}_1$, we prove $a_w \leq \mathrm{maxNO}(t, t_{\mathtt{e}(w)})$.

We start by mapping the root w of $\mathfrak{T}_1$ to the root of $\mathfrak{T}^\infty$. Here, we have $a_w = 1$ and $\mathrm{maxNO}(t, t_{\mathtt{e}(w)}) = \mathrm{maxNO}(t,t) = 1$.

Now assume that we have already defined the mapping for a node $w \in V(\mathfrak{T}_1)$ and we have $a_w \leq \mathrm{maxNO}(t, t_{\mathtt{e}(w)})$. Every child u of w in $V(\mathfrak{T}_1)$ is labeled by a number of the form $a_u = a_w + x$ with $x \in \mathbb{Z}$ and $\mu(x) > 0$. Thus, by the definition of μ, there is a $v \in \mathrm{Leaf}(\mathfrak{T})$ with $x = \mathrm{maxNO}(t, t_v) - 1$. Consider the tree $\mathfrak{T}_{t_{\mathtt{e}(w)}}$. As $a_w \leq \mathrm{maxNO}(t, t_{\mathtt{e}(w)})$, we know that if w has a child in $\mathfrak{T}_1$, then $1 \leq a_w \leq \mathrm{maxNO}(t, t_{\mathtt{e}(w)})$. Thus, by the inequation proved above, there is a bijective embedding $\mathtt{e}_{t_{\mathtt{e}(w)}}$ of $\mathfrak{T}$ into $\mathfrak{T}_{t_{\mathtt{e}(w)}}$ where $\mathrm{maxNO}(t, t_{\mathtt{e}_{t_{\mathtt{e}(w)}}(v)}) \geq \mathrm{maxNO}(t, t_v) + k - 1$ for $k = \mathrm{maxNO}(t, t_{\mathtt{e}(w)})$. As $a_w \leq k$, this implies $\mathrm{maxNO}(t, t_{\mathtt{e}_{t_{\mathtt{e}(w)}}(v)}) \geq \mathrm{maxNO}(t, t_v) + a_w - 1$. In $\mathfrak{T}^\infty$, the node $\mathtt{e}(w)$ eventually reaches a node corresponding to $\mathtt{e}_{t_{\mathtt{e}(w)}}(v)$. Thus, $\mathtt{e}(u)$ is defined to be this node. Then we have $a_u = a_w + x = a_w + \mathrm{maxNO}(t, t_v) - 1 \leq \mathrm{maxNO}(t, t_{\mathtt{e}_{t_{\mathtt{e}(w)}}(v)}) = \mathrm{maxNO}(t, t_{\mathtt{e}(u)})$. Injectivity, path preservation, and equal probabilities for the embedding $\mathtt{e}$ follow by construction.

By Thm. 6, we get $|\mathfrak{T}^\infty| \leq |\mathfrak{T}_1|$ and $\mathrm{edl}(\mathfrak{T}^\infty) \geq \mathrm{edl}(\mathfrak{T}_1)$. So if $|\mathfrak{T}_1| < 1$ holds, then $\mathcal{P}$ is not `AST`, and if $\mathrm{edl}(\mathfrak{T}_1) = \infty$ holds, then $\mathcal{P}$ is not `PAST`. □

Theorem 14 (Embedding Random Walks via Occurrences (2)). *Let $\mathcal{P}$ be a PTRS and let $\mathfrak{T}$ be an* nvd *$\mathcal{P}$-RST with* $\mathrm{h}(\mathfrak{T}) > 0$ *and* $\mathrm{root}(\mathfrak{T}) = t$. *If we have* $\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxNO}(t, t_v) \underset{(-)}{>} 1$*, then $\mathcal{P}$ is not* `(P)AST`.

Proof. The proof is analogous to the one of Thm. 10, but now we construct the infinite $\mathcal{P}$-RST $\mathfrak{T}^\infty$ based on $\mathfrak{T}$ by always rewriting an outermost occurrence of t.

Again, if $s \in \mathcal{T}$ with $\mathrm{maxNO}(t, s) = k \in \mathbb{N}_{>0}$, then instead of the tree $\mathfrak{T}$ whose root is labeled with t, we show that we can construct a tree $\mathfrak{T}_s$ whose root is labeled with s, where $\mathfrak{T}_s$ is like $\mathfrak{T}$ when rewriting an outermost occurrence of t in s instead, according to the rules used in $\mathfrak{T}$. As in the proof of Thm. 10, we can define a bijective embedding $\mathtt{e}_s : V(\mathfrak{T}) \to V(\mathfrak{T}_s)$ and obtain $\mathrm{maxNO}(t, t_{\mathtt{e}_s(v)}) \geq \mathrm{maxNO}(t, t_v) + k - 1$ for all leaves $v \in \mathrm{Leaf}(\mathfrak{T})$. To prove this inequation, let $\{\pi_1, \ldots, \pi_k\} \in \mathrm{NO}(t, s)$ be a witness for the k pairwise non-overlapping occurrences in s. The term s has the form $s = C[t\delta]$ for some context C with a hole at position π_1 and substitution δ. Let π_1 be the position of an outermost occurrence of t, i.e., there is no π_j with $1 \leq j \leq k$ strictly above π_1. We rewrite $C[t\delta]$ according to $\mathfrak{T}$ at position π_1. Since $\mathfrak{T}$ is non-variable-decreasing, for every position $\tau \in \mathrm{Pos}_{\mathcal{V}}(t)$ and every leaf $v \in \mathrm{Leaf}(\mathfrak{T})$, there exists a (unique) position $\tau' \in \mathrm{Pos}_{\mathcal{V}}(t_v)$ with $t|_\tau = t_v|_{\tau'}$.

As in the proof of Thm. 10, we get $t_{\mathtt{e}_s(v)} = C[t_v\delta]$. All occurrences of t at positions that are orthogonal to π_1 are still present in $t_{\mathtt{e}_s(v)}$. All occurrences at a position below π_1, i.e., $\pi_j = \pi_1.\kappa$, can be written as $\pi_j = \pi_1.\tau.\beta$ for some position $\beta \in \mathbb{N}^*$ and $\tau \in \mathrm{Pos}_{\mathcal{V}}(t)$. Moreover, since $t \blacktriangleleft_{\pi_j} C[t\delta]$ we get $t \blacktriangleleft_\beta \delta(x)$ for the variable $x = t|_\tau$. By the previous paragraph, we can find a (unique) position $\tau' \in \mathrm{Pos}_{\mathcal{V}}(t_v)$ with $t|_\tau = t_v|_{\tau'}$, and therefore, all such occurrences are still present in $t_{\mathtt{e}_s(v)} = C[t_v\delta]$, because we still have $t \blacktriangleleft_{\pi_1.\tau'.\beta} C[t_v\delta] \iff t \blacktriangleleft_\beta \delta(x)$ for the variable $x = t_v|_{\tau'} = t|_\tau$. Note that there are no occurrences above π_1

as it is an outermost occurrence. Thus, we obtain $\{\pi_2, \ldots, \pi_k\} \in \mathrm{NO}(t, C[t_v\delta])$. Moreover, all occurrences of t in t_v also exist in $t_v\delta$. Finally, the occurrences of t in t_v are non-overlapping with $\{\pi_2, \ldots, \pi_k\}$. Hence, we get $\mathrm{maxNO}(t, t_{\mathsf{e}_s(v)}) = \mathrm{maxNO}(t, C[t_v\delta]) = \mathrm{maxNO}(t, t_v\delta) + k - 1 \geq \mathrm{maxNO}(t, t_v) + k - 1$ as desired. The rest of the proof (i.e., the definition of the embedding from the $\hookrightarrow_\mu$-RST to $\mathfrak{T}^\infty$) is as in the proof of Thm. 10. $\square$

Theorem 16 (Embedding Random Walks via Occurrences (3)). *Let $\mathcal{P}$ be a PTRS and let $\mathfrak{T}$ be a $\mathcal{P}$-RST with* $\mathrm{h}(\mathfrak{T}) > 0$ *and* $\mathrm{root}(\mathfrak{T}) = t$. *If we have* $\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxOO}(t, t_v) \underset{(-)}{>} 1$, *then $\mathcal{P}$ is not* `(P)AST`.

Proof. Compared to the proof of Thm. 10, we now define the random walk μ via $\mathrm{maxOO}(t, t_v)$ instead of $\mathrm{maxNO}(t, t_v)$. The remaining proof only differs in the proof of the inequation $\mathrm{maxOO}(t, t_{\mathsf{e}_s(v)}) \geq \mathrm{maxOO}(t, t_v) + k - 1$, where $\mathrm{maxOO}(t, s) = k \in \mathbb{N}_{>0}$.

Let $\{\pi_1, \ldots, \pi_k\} \in \mathrm{NO}(t, s)$ be a witness for the k orthogonal occurrences of t in s. The term s has the form $s = C[t\delta]$ for some context C with a hole at position π_1 and substitution δ. We rewrite $C[t\delta]$ according to $\mathfrak{T}$ at position π_1, and get $t_{\mathsf{e}_s(v)} = C[t_v\delta]$.

All the occurrences of t at the positions $\pi_2, \ldots, \pi_k$ are still present, because positions orthogonal to π_1 remain in $C[t_v\delta]$. Moreover, all orthogonal occurrences of t in $t_v\delta$ are orthogonal with all occurrences at a position in $\{\pi_2, \ldots, \pi_k\}$ because they are below π_1. Hence, we get $\mathrm{maxOO}(t, t_{\mathsf{e}_s(v)}) = \mathrm{maxOO}(t, C[t_v\delta]) = \mathrm{maxOO}(t, t_v\delta) + k - 1 \geq \mathrm{maxOO}(t, t_v) + k - 1$ as desired. $\square$

Lemma 21 (Detecting Pattern Terms). *For a term t and a substitution σ, $\langle t, \sigma \rangle$ is a pattern term iff some cycle of $G_{\sigma,t}$ contains a variable x such that $x\sigma \notin \mathcal{V}$.*

Proof. We first show the "if" direction, i.e., we prove that the condition of Lemma 21 is sufficient. To this end, we first show that $x, x\sigma, x\sigma^2, \ldots$ are all pairwise different.

Since x is on a cycle of $G_{\sigma,t}$, there exists a minimal $k \in \mathbb{N}_{>0}$ such that $x \in \mathcal{V}(x\sigma^k)$.

If $k = 1$, then we have $x \in \mathcal{V}(x\sigma)$, i.e., $x\sigma = C[x]$ for a non-empty context $C \neq \square$. Thus, all $x\sigma^m = C^m[x]$ are pairwise different for $m \geq 0$.

Otherwise, if $k > 1$, first note that $x, x\sigma, \ldots, x\sigma^{k-1}$ are pairwise different. The reason is that otherwise, there would be $0 \leq i < j \leq k-1$ with $x\sigma^i = x\sigma^j$. But this would imply $x\sigma^{i+k-j} = x\sigma^{j+k-j} = x\sigma^k$, which is a contradiction because $x \in \mathcal{V}(x\sigma^k)$, but $x \notin \mathcal{V}(x\sigma^{i+k-j})$ due to the minimality of k.

Next, note that $x\sigma^k, x\sigma^{k+1}, \ldots, x\sigma^{2 \cdot k - 1}$ are also pairwise different. The reason is that $x\sigma^k = C[x]_\pi$ for a non-empty context $C \neq \square$. Hence, the subterms of $x\sigma^k, x\sigma^{k+1}, \ldots, x\sigma^{2 \cdot k-1}$ at position π are $x, x\sigma, \ldots, x\sigma^{k-1}$ which are pairwise different by the observation above. Moreover, all terms $x\sigma^k, x\sigma^{k+1}, \ldots, x\sigma^{2 \cdot k-1}$ are pairwise different from the terms $x, x\sigma, \ldots, x\sigma^{k-1}$ due to the additional non-empty contexts. By repeating this reasoning, we can infer that all terms $x, x\sigma, x\sigma^2, \ldots$ are pairwise different.

Now we need to show that all terms $t, t\sigma, t\sigma^2, \ldots$ are pairwise different. By the definition of $G_{\sigma,t}$, t contains a variable y at a position τ where y has a path to x in

$G_{\sigma,t}$. Let j be the length of the shortest path from y to x, i.e., $j \geq 0$ is the minimal number such that $x \in \mathcal{V}(y\sigma^j)$. Similar to the argumentation above, this implies that $y, y\sigma, \ldots, y\sigma^j$ are pairwise different. By considering only the subterm of t at position τ, this also means that $t, t\sigma, \ldots, t\sigma^j$ are pairwise different. As $t\sigma^j$ contains the variable x at some position, by considering the subterm at this position and the argumentation above, we obtain that all terms $t, t\sigma, \ldots, t\sigma^j, t\sigma^{j+1}, \ldots$ are pairwise different.

Now we show the "only if" direction, i.e., we prove that the condition of Lemma 21 is necessary. We assume the contrary, i.e., assume that for all variables x in cycles of $G_{\sigma,t}$ we have $x\sigma \in \mathcal{V}$. This means that for every such variable x there exists a $k_x > 0$ such that $x\sigma^{k_x} = x$.

Moreover, $x\sigma \in \mathcal{V}$ implies that every variable in a cycle of $G_{\sigma,t}$ only has one outgoing edge. Therefore, $t\sigma^{|\operatorname{dom}(\sigma)|}$ only contains variables on cycles of $G_{\sigma,t}$. Let $k = k_{x_1} \cdot \ldots \cdot k_{x_n}$ where $x_1, \ldots, x_n$ are all variables on cycles of $G_{\sigma,t}$. Thus, for all $m \in \mathbb{N}$ we have $t\sigma^{|\operatorname{dom}(\sigma)|} = t\sigma^{|\operatorname{dom}(\sigma)|+m\cdot k}$, which shows that $\langle t, \sigma\rangle$ is not a pattern term. □

Theorem 27 (Embedding RWs via Patterns). *Let $\mathcal{P}$ be a PTRS, $\langle t, \sigma\rangle$ be a pattern term, and let $\mathfrak{T}$ be a $\mathcal{P}$-pattern tree for $\langle t, \sigma\rangle$ with $\mathrm{h}(\mathfrak{T}) > 0$. If we have $\sum_{v \in \mathrm{Leaf}(\mathfrak{T})} p_v \cdot \mathrm{maxOPO}(t, \sigma, t_v) \underset{(-)}{\geq} 1$, then $\mathcal{P}$ is not* (P)AST.

Proof. Compared to the proof of Thm. 16, we define the random walk μ via $\mathrm{maxOPO}(t, \sigma, t_v)$ instead of $\mathrm{maxOO}(t, t_v)$. So we now have to prove the inequation $\mathrm{maxOPO}(t, \sigma, t_{\mathsf{e}_s(v)}) \geq \mathrm{maxOPO}(t, \sigma, t_v) + k - 1$, where $\mathrm{maxOPO}(t, \sigma, s) = k > 0$.

Let $\{\pi_1, \ldots, \pi_h\} \in \mathrm{NO}(t, s)$ be a witness for h orthogonal occurrences in s where the sum of multiplicities is k. The term s has the form $s = C[t\sigma^{m_{\pi_1}}\delta]$ for some context C with a hole at position π_1 and some substitution δ, where $m_{\pi_1} > 0$. Let $\mathfrak{T}_s$ be the tree resulting from starting with $s = C[t\sigma^{m_{\pi_1}}\delta]$ at the root and by rewriting $t\sigma^{m_{\pi_1}}\delta$ at position π_1 according to the rules used in the tree $\mathfrak{T}$. For every $v \in \mathrm{Leaf}(\mathfrak{T})$ let $\mathsf{e}_s(v) \in \mathrm{Leaf}(\mathfrak{T}_s)$ be the corresponding leaf in $\mathfrak{T}_s$ which we get by following the same path in $\mathfrak{T}_s$ as the path to the leaf v in $\mathfrak{T}$. We have $t_{\mathsf{e}_s(v)} = C[t_v\sigma^{m_{\pi_1}-1}\delta] = C[t_{v'}\delta]$ for the corresponding leaf v' in the tree $\mathfrak{T}_{m_{\pi_1}}$ that starts with $t\sigma^{m_{\pi_1}}$ and uses the same rules as in $\mathfrak{T}$ at the same positions.

All the occurrences of $\langle t, \sigma\rangle$ at the positions $\pi_2, \ldots, \pi_h$ with multiplicities $m_{\pi_2}, \ldots, m_{\pi_h}$ are still present, because they are all pairwise orthogonal. Let $\{\chi_1, \ldots, \chi_p\}$ be a witness for p orthogonal occurrences in t_v where the sum of multiplicities is $\mathrm{maxOPO}(t, \sigma, t_v)$. By the requirements on pattern trees, we have $p \geq 1$. Due to commutation, for every occurrence $\langle t, \sigma\rangle \blacktriangleleft_{\chi_j}^{m_{\chi_j}} t_v$ for a leaf $v \in \mathrm{Leaf}(\mathfrak{T})$ and every $\kappa_{\chi_j,v}$ such that $t\sigma^{m_{\chi_j}}\kappa_{\chi_j,v} = t_v|_{\chi_j}$ we have a (unique) occurrence $\langle t, \sigma\rangle \blacktriangleleft_{\chi_j}^{m_{\chi_j}+m_{\pi_1}-1} t_{v'}$ in the corresponding leaf $v' \in \mathfrak{T}_{m_{\pi_1}}$. To see this, note that

$$t_{v'}|_{\chi_j} = t_v|_{\chi_j}\sigma^{m_{\pi_1}-1} = t\sigma^{m_{\chi_j}}\kappa_{\chi_j,v}\sigma^{m_{\pi_1}-1} = t\sigma^{m_{\chi_j}+m_{\pi_1}-1}\kappa_{\chi_j,v}.$$

We get

$$\begin{aligned}
&\mathrm{maxOPO}(t, \sigma, t_{v'})\\
=&\max\{m_S \mid S \in \mathrm{NO}(t, s), \forall \pi_1, \pi_2 \in S \text{ with } \pi_1 \neq \pi_2 : \pi_1 \perp \pi_2\}\\
=&\max\{\sum_{\chi \in S} m_\chi \mid S \in \mathrm{NO}(t, s), \forall \pi_1, \pi_2 \in S \text{ with } \pi_1 \neq \pi_2 : \pi_1 \perp \pi_2\}\\
&\Downarrow \quad \big(\text{since } \{\chi_1, \ldots, \chi_p\} \text{ is a set of orthogonal occurrences}\big)\\
\geq & \sum_{\chi \in \{\chi_1, \ldots, \chi_p\}} m_\chi\\
&\Downarrow \quad \big(\text{by the commutation property of } \mathfrak{T} \text{ as described above}\big)\\
= & \sum_{\chi \in \{\chi_1, \ldots, \chi_p\}} (m_\chi + m_{\pi_1} - 1)\\
=&\Big(\sum_{\chi \in \{\chi_1, \ldots, \chi_p\}} m_\chi\Big) + p \cdot (m_{\pi_1} - 1)\\
&\Downarrow \quad \big(\text{since } p \geq 1 \text{ and } m_{\pi_1} \geq 1\big)\\
\geq&\Big(\sum_{\chi \in \{\chi_1, \ldots, \chi_p\}} m_\chi\Big) + m_{\pi_1} - 1\\
&\Downarrow \quad \big(\text{since } \{\chi_1, \ldots, \chi_p\} \text{ is a witness of the maximum}\big)\\
=&\max\{m_S \mid S \in \mathrm{NO}(t, s), \forall \pi_1, \pi_2 \in S \text{ with } \pi_1 \neq \pi_2 : \pi_1 \perp \pi_2\} + m_{\pi_1} - 1\\
=&\mathrm{maxOPO}(t, \sigma, t_v) + m_{\pi_1} - 1.
\end{aligned}$$

Overall, because of $t_{\mathsf{e}_s(v)} = C[t_{v'}\delta]$ we get

$$\begin{aligned}
&\mathrm{maxOPO}(t, \sigma, t_{\mathsf{e}_s(v)})\\
=&\mathrm{maxOPO}(t, \sigma, C[t_{v'}\delta])\\
&\Downarrow \quad \big(\text{removing } \delta \text{ can only decrease the number of occurrences}\big)\\
\geq&\mathrm{maxOPO}(t, \sigma, C[t_{v'}])\\
&\Downarrow \quad \big(\text{since the occurrences } \pi_2, \ldots, \pi_h \text{ remain and are orthogonal}\big)\\
\geq&\mathrm{maxOPO}(t, \sigma, t_{v'}) + m_{\pi_2} + \ldots + m_{\pi_h}\\
&\Downarrow \quad \big(\text{by the previous inequation}\big)\\
\geq&\mathrm{maxOPO}(t, \sigma, t_v) + m_{\pi_1} - 1 + m_{\pi_2} + \ldots + m_{\pi_h}\\
&\Downarrow \quad \big(m_{\pi_1} + m_{\pi_2} + \ldots + m_{\pi_h} = k\big)\\
=&\mathrm{maxOPO}(t, \sigma, t_v) + k - 1
\end{aligned}$$

□