Probabilistic Program Analysis with Martingales

Aleksandar Chakarov[1] and Sriram Sankaranarayanan[1]

University of Colorado, Boulder, CO.
`firstname.lastname@colorado.edu`

Abstract. We present techniques for the analysis of infinite state probabilistic programs to synthesize probabilistic invariants and prove almost-sure termination. Our analysis is based on the notion of (super) martingales from probability theory. First, we define the concept of (super) martingales for loops in probabilistic programs. Next, we present the use of concentration of measure inequalities to bound the values of martingales with high probability. This directly allows us to infer probabilistic bounds on assertions involving the program variables. Next, we present the notion of a super martingale ranking function (SMRF) to prove almost sure termination of probabilistic programs. Finally, we extend constraint-based techniques to synthesize martingales and super-martingale ranking functions for probabilistic programs. We present some applications of our approach to reason about invariance and termination of small but complex probabilistic programs.

1 Introduction

Probabilistic programs are obtained by enriching standard imperative programming languages with *random value generators* that yield sequences of (pseudo) random samples from some probability distribution. These programs commonly occur in a variety of situations including randomized algorithms [24,12], network protocols, probabilistic robotics, and monte-carlo simulations. The analysis of probabilistic programs is a rich area that has received a lot of attention in the past, yielding tools such as PRISM [18] that implement a wide variety of approaches ranging from symbolic [2] to statistical model checking [33,6].

Our goal in this paper is to investigate a deductive approach to infinite state probabilistic programs, exploring techniques for synthesizing *invariance* and *termination* proofs in a probabilistic setting. Our approach is an extension of McIver and Morgan's *quantitative invariants* [21,20]. Whereas the earlier approach is limited to *discrete probabilistic choices* (eg., Bernoulli trials), our extension handles probabilistic programs with integer and real-valued random variables according to a range of distributions including uniform, Gaussian and Poisson. We make use of the concepts of *martingales* and *super martingales* from probability theory to enable the synthesis of probabilistic invariants and almost sure termination proofs. A martingale expression for a program is one whose *expected value* after the $(n+1)^{th}$ loop iteration is equal to its sample value at the n^{th} iteration. We use Azuma-Hoeffding theorem to derive probabilistic bounds on the value of a martingale expression. Next, we present the concept of a *super martingale ranking function* (SMRF) to prove almost sure termination of loops that manipulate real-valued variables. Finally, we extend constraint-based program analysis techniques originally proposed by Colón et al. to yield techniques for inferring martingales and super-martingales using a template (super) martingale expression with unknown coefficients [8,7]. Our approach uses Farkas lemma to encode the conditions for being a (super) martingale as a system of linear inequality constraints over the unknown coefficients. However, our approach does not require non-linear constraint solving unlike earlier constraint-based approaches [7]. The constraint-based synthesis of linear (super) martingales yields *linear* inequality constraints. We present a preliminary evaluation of our approach over many small but complex benchmark examples using a prototype martingale generator.

The contributions of this paper are (a) we extend *quantitative invariants* approach of McIver and Morgan [20] to a wider class of probabilistic programs through martingale and super-martingale program expressions using concentration of measure inequalities (Azuma-Hoeffding theorem) to generate probabilistic assertions. (b) We define super martingale ranking functions (SMRFs) to prove almost sure termination of probabilistic programs. (c) We present constraint-based techniques to generate (super) martingale expressions. Some of the main limitations of the current work are: (a) currently, our approach applies to purely stochastic programs. Extensions to programs with demonic non-determinism will be considered in the future. (b) The martingale synthesis approach focuses on linear expressions and systems. Extensions to non-linear programs and expressions will be considered in the future. (c) Finally, our approach for proving almost sure termination is *sound* but incomplete over the reals. We identify some of the sources of incompleteness that arise especially for probabilistic programs.

Supplementary Materials An extended version of this paper that includes many of the omitted technical details along with a prototype implementation of the techniques presented here will be made available on-line at our project page [1].

1.1 Motivating Examples

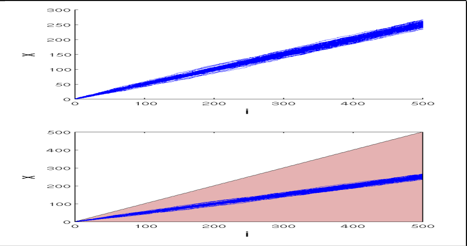
We motivate our approach on two simple examples of probabilistic programs.

Example 1. Figure 1 shows a program that accumulates the sum of samples drawn from a uniform random distribution between $[0, 1]$. Static analysis techniques can infer the invariant $0 \leq \mathrm{x} \leq 501 \ \wedge \ \mathrm{i} = \mathrm{N}$ at loop exit [10,22]. However, the value of x remains *tightly clustered* around 250, as seen in Fig. 1. Using the approaches in this paper, we can *statically* conclude the probabilistic assertion $\mathtt{Pr}(x \in [200, 300]) \geq 0.84$. Unlike statistical model checkers, our symbolic technique provides guaranteed probability bounds as opposed to high confidence bounds [33]. For the example above, the behavior of x can also be deduced using the properties of sums of uniform random variables. However, our approach using martingale theory is quite general: the martingale and the inferred bounds hold even if the call to `unifRand` is substituted by (say) a truncated Gaussian distribution with mean $\frac{1}{2}$.

Example 2 (Almost Sure Termination). Consider the program shown in Figure 1. It shows a program that manipulates two real-valued variables h and t. Initially, the value

[1] `http://systems.cs.colorado.edu/research/cyberphysical/probabilistic-program-analysis`

```
real x = 0;
real N = 500;
for ( i=0; i < N; ++i )
  x = x + unifRand(0,1);
// Prob(x \in [200,300]) ?
```



```
real h, t;
// h is hare and t is tortoise
h = 0; t = 30;
while ( h <= t ){
  if (flip (0.5) )
      h = h + unifRand(0,10);
  t = t +1;
} // almost sure terminate?
```

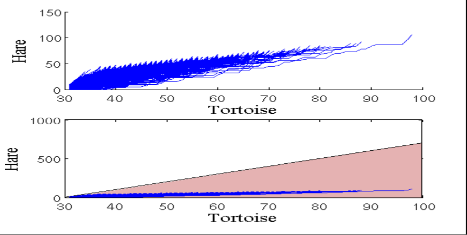


Fig. 1: (**Top, Left**) Program that sums up random variables. (**Top,Right**) sample paths of the program with value of i in x-axis and y-axis showing x. Loop invariants assuming (demonic) non-deterministic semantics for `unifRand` are shown on the right. (**Bottom,Left**) A simple probabilistic program with a loop, (**Bottom,Right**) Sample executions of the program with x axis representing t and y axis representing h. The plot on the bottom also contrasts the worst-case invariants inferred on h, t.

of t is set to 30 and h to 0. The loop iterates as long as h $\leq$ t. Does the program terminate? In the worst case, the answer is no. It is possible that the coin flips avoid incrementing h or when it gets incremented, the uniform random value drawn lies in the interval $[0, 1]$. However, the techniques of this paper establish almost-sure termination using a *super martingale* expression t $-$ h. Such an expression behaves like a ranking function: It is initially positive and whenever its value is non-positive, the loop termination condition is achieved. Finally, for each iteration of the loop, the value of this expression decreases *in expectation* by at least 1.5. Therefore, the techniques presented in this paper infer the almost sure termination of this loop. Furthermore, we also conclude the martingale expression 2.5t$-$h. We use Azuma-Hoeffding theorem to conclude that the value of this expression is tightly clustered around its initial value 75.

2 Probabilistic Transition Systems

In this section, we present a simple transition system model for probabilistic programs. Our model is inspired by the probabilistic guarded command language (PGCL) proposed by McIver et al. [21]. Unlike pGCL, our model allows non-discrete real-valued random variables with arbitrary distributions (Gaussian, Uniform, Exponential etc.). However, we do not allow (demonic) non-determinism. Let $X = \{x_1, \ldots, x_n\}$ be a set

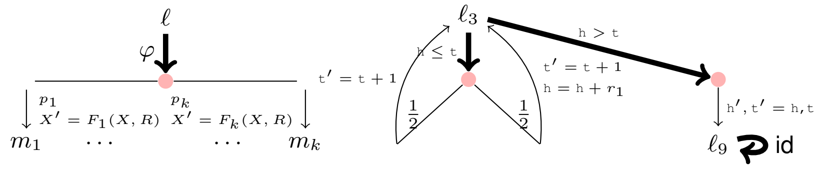


Fig. 2: (LEFT) Structure of a generic PTS transition with $k \geq 1$ forks. Each fork has a probability p_j, an update function F_j and a destination m_j. (MIDDLE) PTS for program in Figure 1 showing two transitions.

of real-valued *program variables* and $R = \{r_1, \ldots, r_m\}$ be a set of real-valued *random variables*. The random variables are assumed to have a joint distribution $\mathcal{D}$.

Definition 1 (Probabilistic Transition System). *A Probabilistic Transition System (PTS) Π is defined by a tuple $\langle X, R, L, \mathcal{T}, \ell_0, \boldsymbol{x}_0 \rangle$ such that*

1. *X, R represent the program and random variables, respectively.*
2. *L represents a finite set of* locations. *$\ell_0 \in L$ represents the* initial location, *and $\boldsymbol{x}_0$ represents the* initial values *for the program variables.*
3. *$\mathcal{T} = \{\tau_1, \ldots, \tau_p\}$ represents a finite set of* transitions. *Each transition $\tau_j \in \mathcal{T}$ is a tuple $\langle \ell, \varphi, f_1, \ldots, f_k \rangle$ consisting of (see Fig 2):*
   (a) Source location $\ell \in L$, and guard assertion φ over X,
   (b) Forks *$\{f_1, \ldots, f_k\}$, where each fork $f_j : (p_j, F_j, m_j)$ is defined by a fork probability $p_j \in (0, 1]$, a (continuous) update function $F_j(X, R)$ and a destination $m_j \in L$. The sum of the fork probabilities is $\sum_{j=1}^{k} p_j = 1$.*

No Demonic Restriction: We assume that all PTSs satisfy the *no demonic restriction*:

1. For each location ℓ, if τ_1 and τ_2 are any two different outgoing transitions at ℓ, then their guards φ_1 and φ_2 are *mutually exclusive*: $\varphi_1 \wedge \varphi_2 \equiv \mathit{false}$.
2. Let $\varphi_1, \ldots, \varphi_p$ be the guards of all the outgoing transitions at location ℓ. Their disjunction is *mutually exhaustive*: $\varphi_1 \vee \varphi_2 \vee \cdots \vee \varphi_p \equiv \mathit{true}$.

Mutual exhaustiveness is not strictly necessary. However, it simplifies our operational semantics considerably. The definition of a PTS seems quite involved at a first sight. We illustrate how probabilistic programs can be translated into PTS by translating the program in Example 2, as shown in Figure 2. The self loop on location ℓ_9 labeled id indicates a transition with guard *true* and a single fork with probability 1 that applies the identity function on the state variables. This transition is added to conform to the no demonic restriction. The semantics for a PTS Π are now described formally starting from the notion of a state and the semantics of each transition.

A *state* of the PTS is a tuple $(\ell, \boldsymbol{x})$ with location $\ell \in L$ and valuation $\boldsymbol{x}$ for the system variables X. We consider the effect of executing a single transition $\tau : \langle \ell, \varphi, f_1, \ldots, f_k \rangle$ on a state $s : (\ell, \boldsymbol{x})$. We assume that τ is *enabled* on s, i.e., $\boldsymbol{x} \models \varphi$. The result of executing τ on s is a *probability distribution* over the post states, obtained by carrying out the following steps:

1. Choose a fork f_j for $j \in [1, k]$ with probability p_j, and a vector of random variables $\boldsymbol{r} : (r_1, \ldots, r_m)$ drawn according to the joint distribution $\mathcal{D}$.
2. Update the states by computing the function $\boldsymbol{x}' = F_j(\boldsymbol{x}, \boldsymbol{r})$. The post-location is updated to $m_j \in L$.

Let $\textsc{Post-Distrib}(s, \tau)$ represent the distribution of the post states $(\ell', \boldsymbol{x}')$ starting from a fixed state $s : (\ell, \boldsymbol{x})$ for an enabled transition τ. A formal definition is provided in our extended version. Due to the no demonic restriction, exactly one transition τ is enabled for each state s. Therefore, the distribution $\textsc{Post-Distrib}(s, \tau)$ can simply be written as $\textsc{Post-Distrib}(s)$ since τ is uniquely defined given s. Operationally, a PTS is a Markov chain, where each *sample execution* is a countable sequence of states.

Definition 2 (Sample Executions). *Let Π be a transition system. A sample execution σ of Π is a countably infinite sequence of states $\sigma :\ (\ell_0, \boldsymbol{x}_0) \xrightarrow{\tau_1} (\ell_1, \boldsymbol{x}_1) \xrightarrow{\tau_2} \cdots \xrightarrow{\tau_n} (\ell_n, \boldsymbol{x}_n)\ \cdots$, such that (a) $(\ell_0, \boldsymbol{x}_0)$ is the unique initial state. (b) The state $s_j : (\ell_j, \boldsymbol{x}_j)$ for $j \geq 0$ satisfies the guard for the transition τ_{j+1}. Note that by the no demonic restriction, τ_{j+1} is uniquely defined for each s_j. (c) Each state $s_{j+1} :\ (\ell_{j+1}, \boldsymbol{x}_{j+1})$ is a sample from* $\textsc{Post-Distrib}(s_j)$.

Figure 1 plots the sample paths for two probabilistic transition systems.

Almost-Sure Termination: Let Π be a PTS with a special *final location* ℓ_F. We assume that the only outgoing transition at the final location is the *identity* transition id, as defined earlier. A sample execution σ of Π *terminates* if it eventually reaches a state $(\ell_F, \boldsymbol{x})$, and thus, continues to cycle in the same state. Let $\pi : \ell_0 \xrightarrow{\tau_1} \ell_1 \xrightarrow{\tau_2} \cdots \ell_F$ be a syntactic path through the PTS. It can be shown that (a) for each finite syntactic path there is a well-defined probability $\mu(\pi) \in [0, 1]$ that characterizes the probability that a sample path traverses the sequence of locations in π, and furthermore (b) the overall probability of termination can be obtained as the sum of the probabilities of all finite syntactic paths π_j that lead from ℓ_0 to ℓ_F.

Definition 3. *A PTS is said to terminate almost surely iff the sum of probabilities of the terminating syntactic paths is* 1.

A measure theoretic definition of almost sure termination is provided in our extended technical report.

Pre-Expectation: We now define the useful concept of pre-expectation of an expression e over program variables across a transition τ, extending the definition for discrete probabilities from McIver & Morgan [20]. Let $s : (\ell, \boldsymbol{x})$ be a state and τ be the enabled transition on s. We define the pre-expectation $\mathbb{E}(\mathsf{e}'|s)$ as the expected value of the expression e' over the $\textsc{Post-Distrib}(s)$, *as an expression involving the current state variables of the program.* Formally, let $\tau : (\ell, \varphi, f_1, \ldots, f_k)$ have $k \geq 1$ forks each of the form $f_j : (p_j, F_j, m_j)$. We define $\mathbb{E}_\tau(\mathsf{e}'|s) = \sum_{j=1}^{k} p_j \mathbb{E}_R(\textsc{pre}(\mathsf{e}', F_j))$, where $\textsc{pre}(\mathsf{e}', F_j)$ represents the expression $\mathsf{e}'[\boldsymbol{x}' \mapsto F_j(\boldsymbol{x}, \boldsymbol{r})]$ obtained by substituting the post-state variables $\boldsymbol{x}'$ by $F_j(\boldsymbol{x}, \boldsymbol{r})$, and $\mathbb{E}_R(g)$ represents the expectation of the expression g over the distribution $\mathcal{D}$. We clarify this further using an example.

Example 3. Going back to Example 2, we wish to compute the pre-expectation of the expression $5 \cdot \mathrm{t} - 2 \cdot \mathrm{h}$ across the transition $\tau_1 : (\ell_3, (\mathrm{h} \leq \mathrm{t}), f_1, f_2)$ with forks $f_1 : (\frac{1}{2}, F_1 : \lambda\, (\mathrm{h}, \mathrm{t}).\ (\mathrm{h}, \mathrm{t}+1), \ell_3)$ and $f_2 : (\frac{1}{2}, F_2 : \lambda(\mathrm{h}, \mathrm{t}).\ (\mathrm{h} + r_1, \mathrm{t}+1), \ell_3)$ (see Fig. 2). The precondition across F_1 yields $\textsc{pre}(5 \cdot \mathrm{t}' - 2 \cdot \mathrm{h}', F_1) = 5 \cdot \mathrm{t} - 2 \cdot \mathrm{h} + 5$. The precondition across F_2 yields $\textsc{pre}(5 \cdot \mathrm{t}' - 2 \cdot \mathrm{h}', F_2) = 5 \cdot \mathrm{t} - 2 \cdot \mathrm{h} + 5 - 2r_1$. The overall pre-expectation is given by

$$\mathbb{E}_\tau(5 \cdot \mathrm{t}' - 2 \cdot \mathrm{h}' | (\ell_3, \mathrm{h}, \mathrm{t})) = \begin{pmatrix} \frac{1}{2}(\mathbb{E}_{r_1}(5 \cdot \mathrm{t} - 2 \cdot \mathrm{h} + 5)) + & (* \leftarrow f_1 *) \\ \frac{1}{2}(\mathbb{E}_{r_1}(5 \cdot \mathrm{t} - 2 \cdot \mathrm{h} + 5 - 2r_1)) & (* \leftarrow f_2 *) \end{pmatrix}.$$

Simplifying, we obtain $\mathbb{E}(5 \cdot \mathrm{t}' - 2 \cdot \mathrm{h}' | (\ell_3, \mathrm{h}, \mathrm{t})) = 5 \cdot \mathrm{t} - 2 \cdot \mathrm{h} + 5 - \mathbb{E}_{r_1}(r_1) = 5 \cdot \mathrm{t} - 2 \cdot \mathrm{h} + 5 - 5 = 5 \cdot \mathrm{t} - 2 \cdot \mathrm{h}$. In other words, the expected value of the expression $5 \cdot \mathrm{t} - 2 \cdot \mathrm{h}$ in the post-state across transition τ is equal to its value in the current state. Such an expression is called a *martingale* [32].

The rest of this paper will expand on the significance of martingale expressions.

3 Martingales and Supermartingale Expressions

A discrete-time stochastic process $\{M_n\}$ is a countable sequence of random variables $M_0, M_1, M_2, \ldots$ where M_n is distributed based on the samples drawn from $M_0, \ldots, M_{n-1}$. By convention, M_n denotes the random variable and m_n its sample.

Definition 4 (Martingales and Super Martingales). *A process $\{M_n\}$ is a* martingale *iff for each $n > 0$, $\mathbb{E}(M_n | m_{n-1}, \ldots, m_0) = m_{n-1}$. In other words, at each step the expected value at the next step is equal to the current value. Likewise $\{M_n\}$ is a* super-martingale *iff for each $n > 0$, $\mathbb{E}(M_n | m_{n-1}, \ldots, m_0) \leq m_{n-1}$.*

If $\{M_n\}$ is a martingale then it is also a super-martingale. Furthermore, the process $\{-M_n\}$ obtained by negating M_n is also a super-martingale. The study of martingales is fundamental in probability theory [32] with numerous applications to the analysis of randomized algorithms [12].

Martingale Expressions: Let Π be a PTS with locations L, variables X and random variables R. We assume, for convenience, that all variables in X and random values are real-valued. We define martingale expressions for a given PTS Π. Next, we explore properties of martingales and super martingales, linking them to those of the PTS.

Definition 5 (Martingale Expressions). *An expression $\mathbf{e}[X]$ over program variables X is a* martingale *for the PTS Π iff for every transition $\tau : (\ell, \varphi, f_1, \ldots, f_k)$ in Π and for every state $s : (\ell, \boldsymbol{x})$ for which τ is enabled, the pre-expectation of $\mathbf{e}$ equals its current state value: $\forall\, \boldsymbol{x}.\ \varphi[\boldsymbol{x}] \ \Rightarrow\ \mathbb{E}_\tau(\mathbf{e}' | \ell, \boldsymbol{x}) = \mathbf{e}$. Likewise, an expression is a* super-martingale *iff for each transition τ, $\forall\, \boldsymbol{x}.\ \varphi[\boldsymbol{x}] \ \Rightarrow\ \mathbb{E}_\tau(\mathbf{e}' | \ell, \boldsymbol{x}) \leq \mathbf{e}$.*

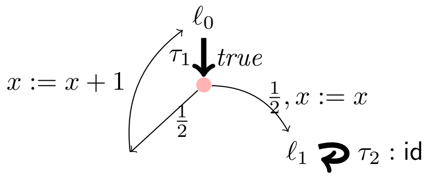
In other words, the stochastic process $\{\mathbf{e}_n\}$ obtained by evaluating the expression $\mathbf{e}$ on a sample execution of Π must be a (super) martingale. Note that in the terminology of McIver & Morgan, martingale expressions correspond to *exact invariants* [21].

Example 4. We noted that for the expression $e :\ 5\cdot\mathrm{t} - 2\cdot\mathrm{h}$, and for any τ_1 enabled state $(\ell_3, \mathrm{h}, \mathrm{t})$, we have that $\mathbb{E}(5\cdot\mathrm{t}' - 2\cdot\mathrm{h}'|\mathrm{h},\mathrm{t}) = 5\cdot\mathrm{t} - 2\cdot\mathrm{h}$. The remaining transitions in the PTS (Cf.Fig. 2) also preserve the expression e. Therefore, e is a martingale. Likewise, we can show that the expression $f :\ -\mathrm{h}$ is a super-martingale.

Often, it is not possible to obtain a single expression that is a martingale for the program as a whole. However, it is natural to obtain a more complex function of the state that maps to different program expressions at different locations.

Example 5. Consider a simple program and its corresponding PTS, as shown below. The program increments a variable x as long as a coin flip turns heads.

```
int x := 0;
while (flip (0.5))
   x ++;
// end
```



We can show that no linear expression over x can be a martingale. Any such expression must be of the form $\mathbf{e} :\ cx$ (we do not need a constant term in our analysis) for some coefficient c. We note that the pre-expectation w.r.t τ_1 is $\mathbb{E}_{\tau_1}(cx'|x) = \frac{1}{2}(c(x+1)) + \frac{1}{2}(cx) = cx + \frac{c}{2}$. This yields the constraint $c = 0$ if the expression is to be a martingale. However, consider a *flow-sensitive* map of the state (ℓ, x) given by $f(\ell, x) = \begin{cases} x & \text{if } \ell = \ell_0 \\ x - 1 & \text{if } \ell = \ell_1 \end{cases}$. We can conclude that $f(\ell, x)$ is a martingale, since its pre-expectation for any transition equals its current value.

Often, labeling different locations in the program with different martingale expressions is quite advantageous.

Definition 6 (Flow-Sensitive Expression Maps). *A flow-sensitive* expression map η *maps each location $\ell \in L$ to a (polynomial) expression $\eta(\ell)$ over X.*

An expression map η is a function that maps a state $s : (\ell, \boldsymbol{x})$ to a real-value by computing $\eta(\ell)[\boldsymbol{x}]$. Let η be an expression map and $\tau : (\ell, \varphi, f_1, \ldots, f_k)$ be a transition with forks $f_1, \ldots, f_k$, where $f_j : (p_j, F_j, m_j)$. The *pre-expectation* of η w.r.t. τ is given by $\mathbb{E}_\tau(\eta'|\ell, \boldsymbol{x}) = \sum_{j=1}^{k} p_j \mathbb{E}_R(\mathrm{PRE}(\eta(m_j), F_j))$.

Example 6. For the program and map η in Ex. 5, we compute the pre-expectation of η w.r.t $\tau_1 : (\ell_0, \mathit{true}, f_1, f_2)$ where $f_1 : (\frac{1}{2}, \lambda x.x+1, \ell_0)$ and $f_2 : (\frac{1}{2}, \mathsf{id}, \ell_1)$.

$$\mathbb{E}_{\tau_1}(\eta'|\ell_0, x) = \frac{1}{2}(\underbrace{\quad x+1 \quad}_{\mathrm{PRE}(\eta(\ell_0), \lambda x.x+1)}) + \frac{1}{2}(\underbrace{\quad x-1 \quad}_{\mathrm{PRE}(\eta(\ell_1), \lambda x.x)}) = x (= \eta(\ell_0))\,.$$

Definition 7 (Martingales and Super Martingale Expression Maps). *An expression map η is a martingale for a PTS Π iff for every transition $\tau : (\ell, \varphi, f_1, \ldots, f_k)$, we have $\forall\, \boldsymbol{x}.\ \varphi[\boldsymbol{x}] \ \Rightarrow\ \mathbb{E}_\tau(\eta'|\ell, \boldsymbol{x}) = \eta(\ell)[\boldsymbol{x}]$.*

Likewise, the map is a super-martingale iff for every transition τ, $\forall\, \boldsymbol{x}.\ \varphi[\boldsymbol{x}] \ \Rightarrow \mathbb{E}_\tau(\eta'|\ell, \boldsymbol{x}) \leq \eta(\ell)[\boldsymbol{x}]$.

Example 7. The map η in Example 6 is a martingale under the condition for transition τ_2. The only other transition is trivial.

3.1 From Martingales to Probabilistic Assertions

We now present some of the key properties of martingales that can be used to make a link from (super) martingale expression maps to probabilistic assertions.

Theorem 1 (Azuma-Hoeffding Theorem). *Let $\{M_n\}$ be a super martingale such that $|m_n - m_{n-1}| < c$ over all sample paths for constant c. Then for all $n \in \mathbb{N}$ and $t \in \mathbb{R}$ such that $t \geq 0$, it follows that $\mathtt{Pr}(M_n - M_0 \geq t) \leq \exp\left(\frac{-t^2}{2nc^2}\right)$. Moreover, if $\{M_n\}$ is a martingale the symmetric bound holds as well: $\mathtt{Pr}(M_n - M_0 \leq -t) \leq \exp\left(\frac{-t^2}{2nc^2}\right)$. Combining both bounds, we conclude that for a martingale $\{M_n\}$ we obtain $\mathtt{Pr}(|M_n - M_0| \geq t) \leq 2\exp\left(\frac{-t^2}{2nc^2}\right)$.*

Azuma-Hoeffding bound is a *concentration of measure inequality*. For a martingale, it bounds the probability of a large deviation on either side of its starting value. For a super-martingale, the inequality bounds the probability of a large deviation *above* the starting value. Both bounds are useful in proving probabilistic assertions.

We note the condition of *bounded change* on the martingale. This has to be established on the side for each transition in the program, and the bound c calculated (using optimization techniques) before the inequality can be applied.

Example 8. The Azuma-Hoeffding bound applies to the program in Ex. 1 (Section 1). We now observe that the expression $2\mathtt{x} - \mathtt{i}$ is a martingale of the loop. Its change at any step is bounded by ± 1. Further, the initial value of the expression is 0. Therefore, choosing $t = 50$, we conclude that after $N = 500$ steps, $\mathtt{Pr}(|(2\mathtt{x} - \mathtt{i})_{500} - (2\mathtt{x} - \mathtt{i})_0| \geq 50) \leq 2\exp\left(-\frac{2500}{2\times 500\times 1}\right) \leq 0.16$. We note that $(2\mathtt{x} - \mathtt{i})_0 = 0$ and $\mathtt{i}_{500} = 500$. Simplifying, with probability at least 0.84, we conclude that $x \in [200, 300]$ after 500 steps.

Since the bounds depend on the number of steps n taken from the start, they are easiest to apply when n is fixed or bounded in an interval. Another common idea is to infer bounds $|M_n - M_0| \geq a\sqrt{n}$ for constant $a > 0$ and $n \geq 0$.

Example 9. Continuing with Ex. 8, for $n > 0$, we conclude the bounds $\mathtt{Pr}(|(2\mathtt{x} - \mathtt{i})_n - (2\mathtt{x} - \mathtt{i})_0| \geq a\sqrt{n}) \leq 2\exp(-\frac{a^2 n}{2n}) \leq 2\exp(-\frac{a^2}{2})$. For $a = 3$, the upper bound is 0.0223.

4 Almost Sure Termination

In this section, we provide a technique for proving that a PTS Π with a final location ℓ_F is almost surely terminating (see definition of almost sure termination in Section 2).

Definition 8 (Ranking Super Martingale). *A super martingale (s.m.) $\{M_n\}$ is* ranking *iff it has the following properties:*

1. *There exists $\epsilon > 0$ such that for all sample paths, $\mathbb{E}(M_{n+1}|m_n) \leq m_n - \epsilon$.*
2. *For all $n \geq 0$, $M_n \geq -K$ for some constant $K > 0$.*

Let $\{M_n\}$ be a ranking s.m. with positive initial condition $m_0 > 0$. A sample path eventually *becomes negative* if $m_n \leq 0$ for some $n \geq 1$.

Theorem 2. *A ranking super martingale with a positive initial condition almost surely becomes negative.*

Proof. The *stopping time* for a sample path is defined as $t = \inf_{n \geq 0} m_n \leq 0$. We use T as a random variable for the stopping time. The *stopped process* is denoted $M_{\min(n,T)}$ (or M_n^T) has sample paths $m_0, \ldots, m_t, m_t, m_t, \ldots$. Note that $M_{\min(n,T)} \geq -K$ over all sample paths.

Next, we define a process $Y_n = M_{\min(n,T)} + \epsilon \min(n, T)$. In other words, for each sample path $y_n = m_n + \epsilon n$ if $n \leq t$, and $y_n = m_t + \epsilon t$ if $n > t$. Note that given a sample path prefix $m_0, \ldots, m_n$ of M_n^T we can compute y_n. Therefore, Y_n is *adapted* to $M_{\min(n,T)}$. Likewise, given $y_0, \ldots, y_n$ we can compute $m_{\min(n,t)}$. Therefore, sample paths of Y_n are one-to-one correspondent with those of M_n^T.

Lemma 1. *$\{Y_n\}$ is a super martingale (relative to M_n^T) and $Y_n \geq -K$.*

Proof. $Y_n \geq -K$ for all n follows from the fact that $M_n^T \geq -K$ for all n. Next, we show that Y_n is a s.m. For any sample path, $\mathbb{E}(Y_{n+1}|y_n, m_n) = \mathbb{E}(M_{n+1}|y_n, m_n) + \min(n+1, t)\epsilon$. We split two cases (a) $n + 1 \leq t$ or (b) $n + 1 > t$.

Case (a): $\mathbb{E}(Y_{n+1}|y_n, m_n) = \mathbb{E}(M_{n+1}|m_n) + (n+1)\epsilon \leq m_n - \epsilon + (n+1)\epsilon \leq y_n$.

Case (b): $y_n = m_t + t\epsilon$. We have $\mathbb{E}(Y_{n+1}|m_n, y_n) = m_t + t\epsilon = y_n$.

In either case, we conclude $\mathbb{E}(Y_{n+1}|m_n, y_n) \leq y_n$.

We note the well-known s.m. convergence theorem.

Theorem 3 (Super Martingale Convergence Theorem). *A lower-bounded super martingale converges (samplewise) almost surely.*

Therefore, with probability 1, a sample path $y_0, \ldots, y_n, \ldots$ converges to a value $\tilde{y}$.

Lemma 2. *For any convergent sample path $y_0, \ldots, y_n, \ldots$, the corresponding $\{M_n\}$ sample path $m_0, \ldots, m_n, \ldots$ eventually becomes negative.*

Proof. Convergence of y_n to $\tilde{y}$ implies for any $\alpha > 0$, there exists N such that $\forall n \geq N, |y_n - \tilde{y}| \leq \alpha$. For contradiction, assume the $\{M_n\}$ sample path has stopping time $t = \infty$. Therefore, $m_n = y_n - n\epsilon$ for all $n \geq 0$. Choosing $\alpha = \epsilon$, for any $n > N$, $m_n \leq \tilde{y} + \alpha - n\epsilon \leq \tilde{y} - (n-1)\epsilon$. Therefore, for $n > 1 + \frac{\tilde{y}}{\epsilon}$, we conclude that $m_n \leq 0$. This contradicts our assumption that $t = \infty$.

To complete the proof, we observe that (a) a sample path $y_0, \ldots, y_n, \ldots$ converges almost surely since $\{Y_n\}$ is a lower bounded s.m.; (b) for each convergent sample path the corresponding (unique) path $m_0, \ldots, m_n, \ldots$ becomes negative; and therefore (c) any $\{M_n\}$ sample path becomes negative almost surely.

Definition 9 (Super Martingale Ranking Function). *A super martingale ranking function (SMRF) η is a s.m. expression map that satisfies the following:*

- *$\eta(\ell) \geq 0$ for all $\ell \neq \ell_F$, and $\eta(\ell_F) \in [-K, 0)$ for some lower bound K.*
- *There exists a constant $\epsilon > 0$ s.t. for each transition τ (other than the self-loop* id *around ℓ_F) with guard φ, $(\forall\, \boldsymbol{x})\ \varphi[\boldsymbol{x}] \ \Rightarrow\ \mathbb{E}_\tau(\eta' | \ell, \boldsymbol{x}) \leq \eta(\ell)[\boldsymbol{x}] - \epsilon$.*

The SMRF definition above is a generalization of similar rules over discrete spaces, including the probabilistic variant rule [20] and Foster's theorem [14,4].

Theorem 4. *If a PTS Π has a super martingale ranking function η then every sample execution of Π terminates* almost surely.

For any sample execution of Π, we define the process $\{M_n\}$ where $m_n = \eta(s_n)$. It follows that $\{M_n\}$ is a ranking super martingale. The rest follows from Theorem 2

Example 10. For the PTS in Example 1, a.s. termination is established by the SMRF $\eta(\ell_0):\ N - i,$ and $\eta(\ell_1):-1$. Consider the PTS in Example 2 and Figure 2, the SMRF $\eta(\ell_3):\mathsf{t}-\mathsf{h}+9$ and $\eta(\ell_9):\mathsf{t}-\mathsf{h}$ proves a.s. termination. The PTS in Ex. 5 has a SMRF $\eta(\ell_2):\ 1$ and $\eta(\ell_4):-1$.

Unlike standard ranking functions, we do not obtain completeness. Consider a purely symmetric random walk:

```
int x := 10; while (x >= 0) { if (flip(0.5)) x++; else x --; }
```

We can show using recurrence properties of symmetric random walks, that the program above terminates almost surely. Yet, no SMRF can be found since the martingale x does not show adequate decrease. However, if the flip probability is changed to $0.5 - \delta$, then the variable x is a SMRF for the program.

5 Discovering (Super) Martingales

Next, we turn our attention to the discovery of (super) martingale expression maps and super martingale ranking functions (SMRF). Our approach builds upon previous work by Colón et al. for constraint-based invariant and ranking function discovery for standard non-deterministic transition systems [7,8]. We restrict our approach to *affine PTS* wherein each transition $\tau : (\ell, \varphi, f_1, \ldots, f_k)$, the guard φ is *polyhedral* (conjunctions of linear inequalities) and the update function F_i for each f_i is affine, $F_i(X, R) : A_i\boldsymbol{x} + B_i\boldsymbol{r} + \boldsymbol{a_i}$.

A *template expression* is a bilinear form $d + \sum_{i=1}^n c_i x_i$ with unknowns $c_1, \ldots, c_n, d$. We may also consider a template expression map η that maps each location ℓ_j to a template expression $\eta(\ell_j) : d_j + \sum_{i=1}^n c_{ji} x_i$. We collectively represent the unknown coefficients as a vector $\boldsymbol{c}$. We encode the conditions for a template expression (map) corresponding to our objective: (super) martingales or super martingale ranking functions. Solving the resulting constraints directly yields (super) martingales.

Example 11. Consider the PTS in example 2 (see Fig. 2). We wish to discover a s.m. using the template $c_1\mathrm{h} + c_2\mathrm{t}$ at locations ℓ_3, ℓ_9.

Encoding Super Martingales We discuss how the conditions on the pre-expectations of s.m. can be encoded using Farkas Lemma. Let $\tau : (\ell, \varphi, f_1, \ldots, f_k)$ be a transition with k forks. Let η be a template expression map. We wish to enforce that η is a s.m. $(\forall\ \boldsymbol{x})\ (\varphi[\boldsymbol{x}]) \ \Rightarrow\ \mathbb{E}_\tau(\eta'|\ell, \boldsymbol{x}) \le \eta(\ell)[\boldsymbol{x}]$. Recall that $\mathbb{E}_\tau(\eta'|\ell, \boldsymbol{x}) = \sum_{j=1}^{k} p_j \mathbb{E}_R(\mathrm{PRE}(\eta(m_j), F_j))$. Each $\mathrm{PRE}(\eta(m_j), F_j)$ can be expressed as $\mathrm{PRE}(\eta(m_j), F_j) = \boldsymbol{c}^T A\boldsymbol{x} + \boldsymbol{c}^T B\boldsymbol{r} + \boldsymbol{c}^T\boldsymbol{a}$.

Example 12. Returning back to Ex. 11, we wish to encode pre-expectation condition for the transition $\tau : (\ell_3, (\mathrm{h} \le \mathrm{t}), f_1, f_2)$, where $f_1 : (\frac{1}{2}, (\lambda(\mathrm{h},\mathrm{t}).\ \mathrm{h}, \mathrm{t}+1), \ell_3)$ and $f_2 : (\frac{1}{2}, (\lambda(\mathrm{h},\mathrm{t}).\ \mathrm{h}+r_1, \mathrm{t}+1), \ell_3)$. We encode the pre-expectation condition for τ:

$$(\forall \mathrm{h},\mathrm{t})\ (\mathrm{h} \le \mathrm{t}) \ \Rightarrow\ \begin{bmatrix} \frac{1}{2}\mathbb{E}_{r_1}(c_1\mathrm{h} + c_2(\mathrm{t}+1)) + \\ \frac{1}{2}\mathbb{E}_{r_1}(c_1(\mathrm{h}+r_1) + c_2(\mathrm{t}+1)) \end{bmatrix} \le c_1\mathrm{h} + c_2\mathrm{t}\,.$$

We note that $\mathbb{E}_{r_1}(r_1) = 5$ (See Figure 1). By linearity of expectation, we obtain

$$(\forall \mathrm{h},\mathrm{t})\ (\mathrm{h} \le \mathrm{t}) \ \Rightarrow\ c_1\mathrm{h} + c_2\mathrm{t} + c_2 + \frac{1}{2}c_1\mathbb{E}_{r_1}(r_1) \le c_1\mathrm{h} + c_2\mathrm{t}\,.$$

Simplifying, we obtain $(\forall \mathrm{h},\mathrm{t})\ (\mathrm{h} \le \mathrm{t}) \ \Rightarrow\ c_2 + \frac{5}{2}c_1 \le 0$. Here, the RHS is independent of the variables h, t. Therefore, we obtain $c_2 + \frac{5}{2}c_1 \le 0$.

Let $\boldsymbol{\mu}$ represent the vector of mean values where $\boldsymbol{\mu}_j = \mathbb{E}_R(r_j)$. Therefore,

$$\mathbb{E}_R(\mathrm{PRE}(\eta(m_j), F_j)) = \boldsymbol{c}^T A\boldsymbol{x} + \boldsymbol{c}^T B\boldsymbol{\mu} + \boldsymbol{a}^T\boldsymbol{c}\,.$$

To encode the s.m. property for τ, we use Farkas Lemma to encode the implication

$$(\forall\ \boldsymbol{x})\ (\varphi[\boldsymbol{x}]) \ \Rightarrow\ \underbrace{\mathbb{E}_\tau(\eta'|\ell, \boldsymbol{x})}_{\text{template expression}} \ \le\ \underbrace{\eta(\ell)[\boldsymbol{x}]}_{\text{template expression}} \tag{1}$$

Let φ be satisfiable and represented in the constraint form as $A\boldsymbol{x} \le \boldsymbol{b}$.

Theorem 5 (Farkas Lemma). *The linear constraint $A\boldsymbol{x} \le \boldsymbol{b} \Rightarrow \boldsymbol{c}^T\boldsymbol{x} \le d$ is* valid *iff its* alternative *is satisfiable $A^T\boldsymbol{\lambda} = \boldsymbol{c}\ \wedge\ \boldsymbol{b}^T\boldsymbol{\lambda} \ge d\ \wedge\ \boldsymbol{\lambda} \ge 0$.*

Encoding the entailment in Eq. (1) using Farkas' Lemma ensures that the resulting constraints are linear inequalities.

Example 13. Continuing with Ex. 12, the transition id yields the constraint *true*. Therefore, the only constraint is $c_2 + \frac{5}{2}c_1 \le 0$. Solving, we obtain the line $(c_1 : 1, c_2 : \frac{-5}{2})$, yielding the martingale $\mathrm{h} - \frac{5}{2}\mathrm{t}$, while the ray $(c_1 : -1, c_2 : 0)$ yields the s.m. $-\mathrm{h}$. Other s.m. such as $\mathrm{t} - \mathrm{h}$ are obtained as linear combinations.

Finding Super Martingale Ranking Functions The process of discovering SM-RFs is quite similar, but requires extra constraints. An abstract interpretation pass can be used to yield helpful invariants by treating the random variables and forks as non-deterministic choices. Let $I(\ell)$ be a polyhedral invariant inferred at the location ℓ.

```
real x,y, estX, estY := 0,0,0,0
real dx, dy, dxc, dyc := 0,0,0,0
int i, N := 0,500
for i = 0 to N {
  cmd := choice(N:0.1,S:0.1,
    E:0.1,W:0.1,NE:0.1,SE:0.1,
    NW:0.1,SW:0.1,Stay:0.2)
  switch (cmd) {
    N: dxc,dyc := 0, rand(1,2)
    S: dxc, dyc := 0, -rand(1,2)
    Stay: dxc,dyc := 0,0
    E: dxc,dyc := rand(1,2), 0
    ...
  }
  dx:= dxc+rand(-.05,.05)
  dy:= dyc+rand(-.05,.05)
  x := x + dx
  y := y + dy
  estX := estX + dxc
  estY := estY + dyc }
```

```
int i := 0;
real money := 10,  bet
while (money >= 10 ) {
  bet := rand(5,10)
  money := money - bet
  if (flip(36/37)) // bank lost
  if flip(1/3)  //  col. 1
     if flip(1/2)
        money := money + 1.6*bet // Red
     else  money := money + 1.2*bet // Black
   elseif flip(1/2) //  col. 2
      if flip(1/3)
         money := money + 1.6*bet; // Red
       else  money := money + 1.2*bet // Black
    else  //  col. 3
       if flip(2/3)
         money := money + 0.4*bet // Red
  i := i + 1 }
```

Fig. 3: (**Left**) Probabilistic program model for dead reckoning and (**Right**) Modeling a betting strategy for Roulette.

1. To encode the non-negativity, we use the invariants at each location $\ell \neq \ell_F$, $I(\ell) \models \eta(\ell) \geq 0$. For location ℓ_F, we encode $I(\ell_F) \models -K \leq \eta(\ell_F) < 0$. The latter condition requires the Motzkin's transposition theorem, a generalization of Farkas' lemma that deals with strict inequalities [17]. Here K is treated as an unknown constant, whose value is also inferred as part of the process.
2. The adequate decrease condition is almost identical to that for s.m. However, we introduce an unknown $\epsilon > 0$ and require that $\mathbb{E}_\tau(\eta'|\ell, \boldsymbol{x}) \leq \eta(\ell) - \epsilon$.

Example 14. Revisiting Ex. 12, we perform an abstract interpretation to obtain the facts $I(\ell_3):\ 0 \leq \mathrm{h} \leq \mathrm{t}+9\ \wedge\ \mathrm{h} \leq 9\mathrm{t}-270 \wedge\ \mathrm{t} \geq 30$ and $I(\ell_9):\ \mathrm{h} > \mathrm{t}\ \wedge\ \mathrm{h} \leq \mathrm{t}+9$. We use the template $\eta(\ell_3) : c_{3,1}\mathrm{h} + c_{3,2}\mathrm{t} + d_3$ and $\eta(\ell_9) : c_{9,1}\mathrm{h} + c_{9,2}\mathrm{t} + d_9$. We obtain the result $c_{3,1} = c_{9,1} = -1, c_{3,2} = c_{9,2} = 1$ and $d_3 = 10, d_9 = 0$, with $\epsilon = \frac{3}{2}$ and $K = -9$. This yields the SMRF $\eta(\ell_3) : \mathrm{t} - \mathrm{h} + 9,\ \eta(\ell_9) : \mathrm{t} - \mathrm{h}$.

6 Evaluation

We have implemented the ideas presented thus far using a constraint generation framework that reads in the description of a PTS and generates constraints for super-martingale expression maps. Our tool uses the Parma Polyhedra Library [1] to generate all possible solutions to these constraints in terms of martingale and super-martingale expressions. Currently, our implementation does not directly communicate with an abstract interpreter for deriving useful invariants. In some of the examples presented in this section, such invariants are computed using a numerical domain polyhedral abstract interpreter and added manually to the PTS description.

Robot Dead Reckoning: Dead reckoning is an approach for position estimation starting from a known *fix* at some time $t = 0$. Figure 3 (**left**) shows a model for robot

Table 1: Results on a set of benchmark programs. **#M:** # of non-trivial martingales discovered and # **S.M.** # of non-trivial super martingales. All timings are under 0.1 seconds on Macbook air laptop with 8 GB RAM, running MAC OSX 10.8.3,

ID	Description	$\lvert X\rvert$	$\lvert R\rvert$	$\lvert L\rvert$	$\lvert T\rvert$	#M	#S.M.
ROULETTE	betting strategy for roulette	3	1	1	1	1	1
TRACK	Target tracking with feedback	3	5	3	9	1	3
2DWALK	Random walk on $\mathbb{R}^2$	4	1	1	4	3	1
COUPON5	coupon collectors with $n = 5$ coupons	2	0	5	4	4	8
FAIRBIASCOIN	simulating a fair coin by biased coin	3	0	2	3	0	2
QUEUE	queue with random arrivals/service	3	0	1	2	1	2
CART	steering a cart on a rough surface	5	4	6	12	2	4
INVPEND	discrete inverted pendulum under stochastic disturbance	5	6	3	3	0	0
PACK	packing variable weight objects in cartons	6	2	3	5	3	4
CONVOY2	leader following over a convoy of cars	6	1	2	4	1	0
DRECKON	dead reckoning model	10	4	3	3	4	1

navigation that involves estimating the actual position (x,y) of the robot as it is commanded to make various moves. Each step involves a choice of direction chosen from the set of compass directions $\{N, W, E, S, NE, NW, SW, SE\}$ each with probability 0.1 or a "Stay" command with probability 0.2. The variables dxc,dyc capture the commanded direction, whereas the actual directions are slightly off by a random value. Our goal is to estimate how the position x,y deviates from the actual position estX,estY.

Our analysis shows that the expressions $\mathrm{x} - \mathrm{estX}$ and $\mathrm{y} - \mathrm{estY}$ are martingales at the loop head. The absolute change in these martingales are bounded by 0.05. Given the initial difference of 0 between the values, we infer using Azuma-Hoeffding theorem that $\mathtt{Pr}(|\mathrm{x} - \mathrm{estX}| \geq 3) \leq 1.5 \times 10^{-3}$. In contrast, a worst-case analysis concludes that $|\mathrm{x} - \mathrm{estX}| \leq 0.05 * 500 \leq 25$. The analysis for $\mathrm{y} - \mathrm{estY}$ yields identical results.

Roulette: For our next example, we analyze a betting strategy for a game of Roulette. The game involves betting money on a *color* (red or black) and a column (1,2 or 3). At each step, the player chooses an amount to bet randomly between 5 and 10 dollars. We skip a detailed description of the betting strategy and simply model the effect of the strategy as a probabilistic program, as shown in Figure 3 (**right**). The model captures the various outcome combinations ($\{Bank\} \uplus \{\mathrm{Red}, \mathrm{Black}\} \times \{1, 2, 3\}$), including the one where the bank wins outright with probability $\frac{1}{37}$. Our analysis discovers the martingale expression $15 \times \mathrm{i} - 74 \times \mathrm{money}$ which can be used to bound the probability of the money exceeding a certain quantity after n rounds. We generate the SMRF: $-\mathrm{money}$. Thus, the program terminates almost surely in the gambler's *ruin*.

Table 1 shows an evaluation of our approach over a set of linear PTS benchmarks. A description of the benchmarks and the inferred properties are provided in our extended technical report available on-line [2].

[2] `http://systems.cs.colorado.edu/research/cyberphysical/probabilistic-program-analysis`

7 Related Work

Probabilistic programs can be quite hard to reason about. A large volume of related work has addressed techniques for formally specifying and verifying properties of probabilistic systems. Statistical approaches rely on simulations [33,6,19], providing high confidence guarantees. On the other hand, symbolic techniques including BDD-based approaches [2], probabilistic CEGAR [16], deductive approaches [21] and abstract interpretation techniques [23,11,3] attempt to establish guaranteed probability bounds on temporal properties of programs.

Martingale theory has been employed to establish guarantees for randomized algorithms [24,12]. In particular, the *method of bounded differences* is a popular approach that establishes a martingale and uses Azuma's inequality to place bounds on its values. The contribution of this paper lies in a partial mechanization of the process of discovering martingales and the termination analysis of programs using super-martingale ranking functions.

Our work is closely related to the *quantitative invariants* proposed by McIver and Morgan, and summarized in their monograph [20]. Informally, quantitative invariants involve program expressions whose pre-expectations are at least their current value. These are used to establish pre-/post-annotations for programs, and in some cases they lead to an *almost sure termination* proof principle. In the framework of this paper, a quantitative invariant roughly corresponds to the negation of a super-martingale (also known as a sub-martingale). There are many differences between McIver & Morgan's approach and that of this paper. Chiefly, our approach considers real-/integer-valued random variables with a large variety of probability distributions, whereas the probabilistic distributions studied by McIver & Morgan are restricted to discrete distributions over a finite set of choices. The use of concentration of measure inequalities and the presentation of almost sure termination proofs using martingale theory are unique to this work. On the other hand, our work does not consider (demonic) non-determinism. Furthermore, we do not integrate martingales and super-martingales into a deductive proof system for proving properties of expectations. Part of the reason for this lies in the difficulty of establishing that expectations of program variables are well-defined for a given program. Many simple examples fail to yield well-defined expectations. We plan to study these issues further and achieve a more complete deductive verification framework as part of our future work.

The constraint-based analysis of (non-probabilistic) programs has been studied for invariance and termination proofs by many authors, including Colón et al. [8,7], Bradley et al. [5], Cousot [9], Podelski et al. [28] and Gulwani et al. [15]. Extensions to polynomial invariants were considered by Sankaranarayanan et al. [31], Carbonell et al. [30], Müller-Olm et al. [25] and Platzer et al. [26]. Recent work of Katoen et al. uses a constraint-based invariant synthesis method to derive quantitative invariants, generalizing earlier approaches [17]. Our work in this paper was inspired, in part, by this generalization. Katoen et al. derive quantitative invariants that involve a combination of characteristic functions over linear assertions and program expressions. As a result, their approach yields nonlinear (bilinear) constraints of the same form as those obtained by earlier work by Colón et al. [7]. Our approach focuses on linear (super) martingales and therefore, we obtain linear inequality constraints.

The almost sure termination of probabilistic programs was studied for finite state and “weakly” infinite programs by Esparza et al. using *patterns* [13]. Their approach attempts to find the existence of a sequence of discrete probabilistic choices that will lead to termination from any reachable state. As such, finite length patterns do not exist for general infinite state systems studied here. Other approaches to almost sure termination including that of Morgan [20] and Bournez et al. [4] use the *probabilistic variant rule*, or equivalently *Foster’s theorem*, a well-known result in (discrete) Markov chain theory [14]. In fact, these principles turn out to be specializations of the SMRF principle presented in this paper.

(Super) martingale theory is widely used in stochastic calculus to reason about continuous time systems. Recently, Platzer considers an extension of differential logic to stochastic hybrid systems [27]. Martingale theory is used as a basis for proving properties in this setting. However, Platzer’s work does not deal directly with martingale expressions over state variables or the generation of such expressions. The techniques presented in this paper are dependent on discrete time martingale theory. We plan to extend our use of martingale theory to reason about stochastic hybrid systems. Another future direction will consider the use of (super) martingales to infer relational abstractions of probabilistic systems [29].

Acknowledgments

Thanks to Prof. Manuel Lladser for useful pointers to martingale theory and the anonymous reviewers for detailed comments. This work was supported by the US National Science Foundation (NSF) under award numbers CNS-0953941 and CNS-1016994.

References

1. R. Bagnara, P. M. Hill, and E. Zaffanella. The Parma Polyhedra Library: Toward a complete set of numerical abstractions for the analysis and verification of hardware and software systems. *Science of Computer Programming*, 72(1–2):3–21, 2008.
2. C. Baier, E. Clarke, V. Hartonas-Garmhausen, M. Kwiatkowska, and M. Ryan. Symbolic model checking for probabilistic processes. In *Proc.ICALP’97*, volume 1256 of *LNCS*, pages 430–440. Springer, 1997.
3. O. Bouissou, E. Goubault, J. Goubault-Larrecq, and S. Putot. A generalization of p-boxes to affine arithmetic. *Computing*, 2012.
4. O. Bournez and F. Garnier. Proving positive almost-sure termination. In *RTA*, volume 3467 of *LNCS*, pages 323–337. Springer, 2005.
5. A. R. Bradley, H. B. Sipma, and Z. Manna. Termination of polynomial programs. In *VMCAI 2005*, volume 3385 of *LNCS*, January 2005.
6. E. Clarke, A. Donze, and A. Legay. Statistical model checking of analog mixed-signal circuits with an application to a third order $\delta - \sigma$ modulator. In *Hardware and Software: Verification and Testing*, volume 5394/2009 of *LNCS*, pages 149–163, 2009.
7. M. Colón, S. Sankaranarayanan, and H. Sipma. Linear invariant generation using non-linear constraint solving. In *CAV*, volume 2725, pages 420–433, July 2003.
8. M. Colón and H. Sipma. Synthesis of linear ranking functions. In *TACAS*, volume 2031, pages 67–81. Springer, April 2001.

9. P. Cousot. Proving program invariance and termination by parametric abstraction, lagrangian relaxation and semidefinite programming. In *VMCAI*, volume 3385 of *LNCS*, pages 1–24. Springer, 2005.
10. P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among the variables of a program. In *POPL'78*, pages 84–97, Jan. 1978.
11. P. Cousot and M. Monerau. Probabilistic abstract interpretation. In *ESOP*, volume 7211 of *LNCS*, pages 169–193. Springer, 2012.
12. D. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.
13. J. Esparza, A. Gaiser, and S. Kiefer. Proving termination of probabilistic programs using patterns. In *CAV*, volume 7358 of *LNCS*, pages 123–138. Springer, 2012.
14. F. Foster. On the stochastic matrices associated with certain queuing processes. *Annals Mathematical Statistics*, 24(3):355–360, 1953.
15. S. Gulwani, S. Srivastava, and R. Venkatesan. Program analysis as constraint solving. In *PLDI*, pages 281–292. ACM, 2008.
16. H. Hermanns, B. Wachter, and L. Zhang. Probabilistic CEGAR. In *CAV*, volume 5123 of *LNCS*, pages 162–175. Springer, 2008.
17. J.-P. Katoen, A. McIver, L. Meinicke, and C. Morgan. Linear-invariant generation for probabilistic programs. In *SAS*, volume 6337 of *LNCS*, page 390406. Springer, 2010.
18. M. Kwiatkowska, G. Norman, and D. Parker. PRISM: probabilistic model checking for performance and reliability analysis. *SIGMETRICS Perf. Eval. Review*, 36(4):40–45, 2009.
19. R. Lassaigne and S. Peyronnet. Probabilistic verification and approximation. *Annals of Pure and Applied Logic*, 152(1-3):122 – 131, 2008.
20. A. McIver and C. Morgan. *Abstraction, Refinement and Proof for Probabilistic Systems*. Monographs in Computer Science. Springer, 2004.
21. A. McIver and C. Morgan. Developing and reasoning about probabilistic programs in pGCL. In *PSSE*, volume 3167 of *LNCS*, pages 123–155. Springer, 2006.
22. A. Miné. A new numerical abstract domain based on difference-bound matrices. In *PADO II*, volume 2053, pages 155–172, May 2001.
23. D. Monniaux. Abstract interpretation of programs as markov decision processes. *Sci. Comput. Program.*, 58(1-2):179–205, 2005.
24. R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.
25. M. Müller-Olm and H. Seidl. Precise interprocedural analysis through linear algebra. In *POPL*, pages 330–341. ACM, 2004.
26. A. Platzer. Differential dynamic logic for hybrid systems. *J. Autom. Reasoning*, 41(2):143–189, 2008.
27. A. Platzer. Stochastic differential dynamic logic for stochastic hybrid programs. In *CADE*, volume 6803 of *LNCS*, pages 446–460. Springer, 2011.
28. A. Podelski and A. Rybalchenko. A complete method for the synthesis of linear ranking functions. In *VMCAI*, volume 2937 of *LNCS*, pages 239–251. Springer, 2004.
29. A. Podelski and A. Rybalchenko. Transition invariants. In *LICS*, pages 32–41. IEEE Computer Society, 2004.
30. E. Rodriguez-Carbonell and D. Kapur. Automatic generation of polynomial loop invariants: Algebraic foundations. In *Proc. ISSAC*, Spain, 2004.
31. S. Sankaranarayanan, H. Sipma, and Z. Manna. Non-linear loop invariant generation using Gröbner bases. In *POPL*, pages 318–330. ACM Press, 2004.
32. D. Williams. *Probability with Martingales (Cambridge Mathematical Textbooks)*. Cambridge University Press, Feb 1991.
33. H. L. S. Younes and R. G. Simmons. Statistical probabilitistic model checking with a focus on time-bounded properties. *Information & Computation*, 204(9):1368–1409, 2006.