Deductive Verification of Infinite-State Stochastic Systems using Martingales

by

Aleksandar Nevenov Chakarov

B.A., Bard College, 2010

M.S., University of Colorado Boulder, 2012

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

2016

This thesis entitled:
Deductive Verification of Infinite-State Stochastic Systems using Martingales
written by Aleksandar Nevenov Chakarov
has been approved for the Department of Computer Science

_____

Prof. Sriram Sankaranarayanan

_____

Prof. Rafael Frongillo

Date _____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Chakarov, Aleksandar Nevenov (Ph.D., Computer Science)

Deductive Verification of Infinite-State Stochastic Systems using Martingales

Thesis directed by Prof. Sriram Sankaranarayanan

The focus of this dissertation is the analysis of and verification of discrete time stochastic systems using martingales. Martingale theory yields a powerful set of tools that have recently been used to prove quantitative properties of stochastic systems such as stochastic safety. In this thesis, we focus on the analysis of qualitative trace properties of stochastic systems such as almost sure reachability and termination, persistence and recurrence. An almost sure reachability property $\Diamond(T)$ states that with probability 1 the executions of the system reach a target set of states T. A qualitative persistence property $\Diamond\Box(T)$ specifies that almost all executions of the stochastic system eventually reach the target set T and stay there forever. Likewise, a recurrence property $\Box\Diamond(T)$ specifies that a target set of states T is visited infinitely often by almost all executions of the stochastic system.

For each type of property, we present deductive reasoning techniques in the form of proof rules that rely on finding an appropriate certificate function to establish almost sure reachability, persistence and recurrence properties of infinite-state, discrete time polynomial stochastic systems. Next, we extend known efficient constraint-based and abstract interpretation-based invariant synthesis techniques to deduce the necessary supermartingale expressions to partly mechanize such proofs. We demonstrate that martingale certificates can serve as expectation invariants and generalize this idea to sets of mutually inductive expectation invariants.

Finally, we explore the connection between the properties of our martingale certificates and existing concentration of measure results to establish probability bounds on the quantitative version of these properties.

Dedication

To my parents Nazmi and Tanya,

my sister Rozi, my aunt Maria,

and my grandmother Roza.

Thank you for being my constant source of support!

Acknowledgements

I would like to thank, first and foremost, my PhD Advisor Prof. Sriram Sankaranarayanan for always been the source of energy and support that a graduate student needs to complete this journey! Sriram, your passion and flair for always choosing the most intellectually stimulating problems made the trip here possible. I am very lucky and grateful to be your student!

Second, I would like to thank the NSF for providing the funding and my collaborators for the research power to complete the work in this dissertation.

Third, I would like to thank the committee members Prof. Fabio Somenzi, Prof. Matthew Hammer, Dr. Aditya Nori, Prof. Rafael Frongillo, Prof. Bor-Yuh Evan Chang and Prof. Pavol Černý for their guidance throughout the program.

Next, I would like to thank my friends Alex, Maria and Julie for being my source of sanity and extra nudge right when I needed it the most! Alex, I wouldn't have finished this without you!

I would like to thank the former and current members of the CU Programming Languages and Verification (CUPLV) and Verification of Cyber Physical Systems groups for guiding me towards success as a PhD Student. Through your help I learned a lot!

Finally, I would like to thank the CS Graduate Program Advisor Rajshree Shrestha for helping me navigate through the maze!

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

Probabilistic or stochastic systems are a modeling formalism used to describe complex processes that arise in a multitude of areas: medical and financial decision making [150], stochastic modeling [138] (climate change [90] and earthquake prediction models [155]), stochastic optimal control [14], computer performance modeling and reliability [99], stochastic optimization [92], sensor fusion algorithms [27], randomized algorithms [129], population and epidemiological modeling [45], biochemical reactions and systems biology [31]. These systems all operate in the presence or under the influence of some source of **uncertainty** that can be modeled **quantitatively** or **estimated empirically**. Often the sources of uncertainty occur naturally due to physical limitations such as imperfections in sensing and actuating equipment, incomplete information as in the case of individual agents trading on the stock market, emerging collective behavior (the global economy); or are intentionally introduced [129] to simulate the effects and evolutions of stochastic processes, to overcome a difficulty of selecting an optimal strategy and break symmetries [92], to account for noise in input data [16, 131], or to leverage performance [124] and privacy [62] gains at the expense of accuracy.

Analysis and **verification** of such systems with respect to a set of target properties employs formal mathematical tools to describe models, estimate probabilities or to **prove** that a property holds during the operation of such a system. This is a challenging but important problem with applications to areas such as risk assessment, safety and quality control.

Mathematical models based on Markov processes [142] have emerged as the preferred com-

putational models for these systems. The main characteristic of Markov processes is that the distribution over the next states is a function of the current system state, and does not explicitly reference the history of states visited by the process since the beginning. Another characteristic of Markov models is that they make a strong distinction between stochastic and **nondeterministic** sources of uncertainty. Stochastic choice is a random decision among alternatives that is governed by a probability distribution and, therefore, enjoys a number of statistical properties. In contrast, nondeterministic choice need not obey results in probability theory such as the law of large numbers, the central limit theorem [40], or the concentration of measure phenomenon [60] but is instead used to represent rational cooperative or adversarial behavior.

Depending on the choice of abstraction of the **state space**, or the set of values that the model variables range over, **timing behavior** and presence of **nondeterminism** (unquantifiable uncertainty) different variations of Markov processes have been formalized:

- **Discrete Time Markov Chain (DTMC)** is a stochastic model in which execution (time) proceeds in discrete steps. Nondeterminism is precluded. State space is discrete and usually finite [110, 66] but (countably) infinite state DTMCs exist [61].

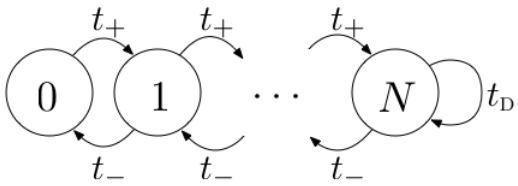
- **Continuous Time Markov Chain (CTMC)** is a stochastic model in which state space is discrete, time flows continuously, and execution remains within a state until a discrete jump is taken at some specific time instance.

  Example 1.0.1. *Suppose we want to model a server application with finite buffer N that accepts and processes requests based on empirical data stored in the server logs. By looking at the sequence of events (arrivals, completions), we can model the server application as a DTMC in which each state s, $s \in \{0, \ldots, N\}$, represents the number of outstanding requests. At each time step a new request arrives with* ***transition probability*** *t_+ or an existing request is processed with transition probability t_- where probabilities of all outgoing transitions from a state sum to 1.*

  Alternatively, by looking at the timestamps of incoming requests and time taken to process

a request, we can build a CTMC exponential distribution model that best approximates the empirical data. Using a CTMC model one can accurately model the flow of time as requests come in at some ***transition rate*** t_+ *and are handled at rate* t_-*.*

Both Markov chain models can be visualized with the following state transition diagram:



where $t_{\mathrm{D}} = t_+$ *denotes the probability (rate) of dropping a request due to a full buffer. Some important questions one may ask when analyzing the system is:* ***What is a safe upper bound on the probability the server drops a request?*** *Or,* ***if the system is currently in state*** K***, for some*** $0 \leq K \leq N$***, for the next 1000 time steps what fraction of the time does the system spend in expectation in suboptimal conditions (outstanding requests below*** $0.1N$ ***or above*** $0.9N$***)?*** *This is a sample of the problems that both performance analysis and verification of probabilistic systems address.*

- **Markov Decision Process** (MDP) is a stochastic model with discrete state space in which time flows discretely and nondeterministic choice is allowed. This model is often used in multi-agent settings to give game-theoretic (or turn-based) semantics [142].

A unifying approach to model these systems is to write them down in the form of probabilistic programs [81]. A large number of probabilistic programming languages/formalisms have been proposed including: BUGS [77], IBAL [136], STAN [30], Church [79], Figaro [137], Tabular [80], R2 [134], and others [2]. We present most of our examples in the form of simple probabilistic programs written in imperative style. These examples are easy to translate into many of the above languages.

In this thesis we focus on the analysis of discrete time, infinite-state stochastic systems. The evolution (or **dynamics**) of such systems is governed by polynomial difference equations over the

system variables and well-defined noise terms. We analyze traces or trajectories of such systems with focus on **reachability** and **termination**, as well as repeated reachability (**persistence** and **recurrence**) properties. We phrase these properties as (**probabilistic**) **assertions** over the system variables and often write as formulas in probabilistic branching time logic (PCTL).

The contributions of this thesis are in providing a *deductive* verification approach to reachability and repeated reachability properties. We describe approaches to handle both *qualitative* (probability 1) and *quantitative* (or, estimation) probabilistic assertions. We present two modifications of existing invariant generation techniques to find functions that certify the correctness of the claims of probabilistic assertions. Finally, we demonstrate how our technique defines a class of functions that is *mutually invariant* in expectation.

The core of our approaches is in identifying functions over the system states that act as **martingales** over the sample runs of stochastic processes. Martingales are a class of particularly well-behaved stochastic processes with constraints on their expected values and strong convergence properties. The strengths of using martingales are many: (*i*) they allow us to reason symbolically about distributions over system states; (*ii*) they act as invariants in expectation; (*iii*) they allow us to prove progress towards and convergence to a set of system states with probability 1; and, (*iv*) they allow us to leverage strong theoretical results on concentration of measure probability bounds of probabilistic assertions.

Organization. The rest of this thesis is structured as follows. Chapter 2 presents the necessary background information on probability theory, stochastic processes with focus on martingales, the properties we analyze and the concentration of measure phenomenon. Chapter 3 presents an overview of relevant analysis efforts by the formal methods and verification, model checking, and program analysis communities. Chapter 4 presents a deductive proof technique that employs martingale program expressions to establish qualitative, or almost sure, reachability and repeated reachability properties. The soundness of the deductive proof rules is proved using convergence properties of martingales. Chapter 5 presents a technique based on solving linear and semidefinite constraints that generates supermartingale expressions. Chapter 6 demonstrates how martingale

expressions can be used to prove probability bounds on the quantitative version of the properties of interest. Chapter 7 extends the idea that a supermartingale expression constitutes an expectation invariant to a *set* of program expressions that act as mutually inductive expectation invariants. We show sets of such program expressions can be inferred automatically by defining an abstract interpreter that operates over the moments of the reachable distributions of states. Chapter 8 presents open problems and directions for future research before concluding.

Chapter 2

Preliminaries

2.1 Probability Theory Basics.

Let $\Omega = \{\omega_1, \omega_2, \ldots\}$ be a (potentially infinite) set of **outcomes**. A subset of outcomes $E \subseteq \Omega$ is called an **event** and we denote by $\mathscr{E} \subseteq \mathcal{P}(\Omega)$ the set of **all events**. A collection $\mathcal{E}$ of subsets of Ω (events) is called a **σ-algebra on Ω** if $\mathcal{E}$:

(1) contains $\emptyset$ and Ω: $\emptyset, \Omega \in \mathcal{E}$;

(2) is closed under complementation: $E_i \in \mathcal{E} \implies E_i^C \in \mathcal{E}$, where $E_i^C \triangleq \Omega \setminus E_i$;

(3) is closed under countable unions: let $E_i \in \mathcal{E}$ for all $n \in \mathbb{N}$, then $\bigcup_{n=0}^{\infty} E_i \in \mathcal{E}$.

From items (2) and (3) it follows that $\mathcal{E}$ is closed under countable intersections. A σ-algebra of particular interest to us is the standard Borel σ-algebra.

Example 2.1.1 (Borel σ-algebra)**.** *Let $\mathbb{R}^n$ denote the standard n-dimensional Euclidean space. Let $d : \mathbb{R} \times \mathbb{R} \to [0, \infty)$ denote the standard Euclidean distance function between any two points x, y in $\mathbb{R}^n$ defined as: $d(x, y) \triangleq \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$, where x_i is the i-th coordinate of x.*

Let x be a point in $\mathbb{R}^n$ and $r > 0$ then the open ball of radius r centered at x is the set $B_r \triangleq \{y \in \mathbb{R}^n : d(y, x) < r\}$. The collection $\mathcal{B}(R^n)$ of all open balls of $\mathbb{R}^n$ when closed under union, intersection and complementation forms a σ-algebra. $\mathcal{B}(R^n)$ is called the standard Borel σ-algebra and any set $B \in \mathcal{B}(R^n)$ is called a Borel set.

A set of outcomes Ω and a σ-algebra $\mathcal{E}$ over Ω form a **measurable space** $(\Omega, \mathcal{E})$ — that is a collection of "well-behaved" events on which we can impose a measure. A function $\mu : \mathcal{E} \to \mathbb{R}$ is called a **measure** if:

- $\mu(E) \geq \mu(\emptyset) = 0 \quad$ for all $E \in \mathcal{E}$;
- if $\{E_i\}_{i=1}^{\infty}$ is a countable sequence of disjoint events in $\mathcal{E}$ (i.e., for any E_i, E_j in the sequence, $E_i \cap E_j = \emptyset$), then $\mu\left(\bigcup_{i=1}^{\infty} E_i\right) = \sum_{i=1}^{\infty} \mu(E_i)$.

A measure is therefore a nonnegative, countably additive function that assigns to each event its corresponding measure. A measure μ is a **probability measure** (or **probability distribution**) if $\mu(\Omega) = 1$. The **set of support** of μ is $Supp(\mu) \triangleq \{E \in \mathcal{E} : \mu(E) > 0\}$. A measurable space $(\Omega, \mathcal{E})$ equipped with a probability measure $P : \mathcal{E} \to [0, 1]$ forms a **probability space** $(\Omega, \mathcal{E}, P)$.

Let $(\Omega_1, \mathcal{E}_1)$, $(\Omega_2, \mathcal{E}_2)$ be measurable spaces, a function $f : \Omega_1 \to \Omega_2$ is called **measurable** if the preimage of every event $E \in \mathcal{E}_2$ belongs to $\mathcal{E}_1$:

$$f^{-1}(E) \triangleq \{\omega_1 \in \mathcal{E}_1 \mid f(\omega_1) \in E\} \in \mathcal{E}_1 \quad \text{for all } E \in \mathcal{E}_2.$$

The function f is also called a **measurable map** from $(\Omega_1, \mathcal{E}_1)$ to $(\Omega_2, \mathcal{E}_2)$.

Random Variables. Let $(\Omega, \mathcal{E}, P)$ be a probability space and $(S, \mathcal{S})$ be a measurable space. If X is a measurable map from $(\Omega, \mathcal{E})$ to $(S, \mathcal{S})$, then X is called an S-valued **random variable**. If X can take on only countably many values then X is a **discrete** random variable; if X can take on uncountably many values then X is a **continuous** random variable.

Example 2.1.2. *If $(S, \mathcal{S}) = (\mathbb{R}, \mathcal{B})$ then X is a real-valued (continuous) random variable. If X is generalized to higher dimensions, that is, $(S, \mathcal{S}) = (\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$ then X is called a **random vector**.*

For random variable X, we abuse notion and denote the event $X \in B$ for some $B \in \mathcal{B}$ to mean $\{\omega : X(\omega) \in B\}$; additionally, $X \leq x$ denotes $\{\omega : X(\omega) \leq x\}$ (similarly for $X = x$, etc.). We refer to $F_X(x) = P(X \leq x)$ as the **(cumulative) distribution function** of X. If F_X has the form:

$$F_X(x) = \int_{-\infty}^{x} f_X(u)\, du$$

then we refer to $f_X(x)$ as the **probability density** function of X.

Given a set $E \in \mathcal{E}$ an important example of a random variable is the **indicator function** $\mathbb{1}_E(\omega) : \Omega \to \{0, 1\}$:

$$\mathbb{1}_E(\omega) \triangleq \begin{cases} 1, & \text{if } \omega \in E \\ 0, & \text{otherwise} \end{cases}$$

The continuous random variable X above **induces** a probability distribution

$$\mu_X(B) = P(\{\omega : X(\omega) \in B\})$$

over the Borel sets of $\mathbb{R}^n$.

A random variable X from $(\Omega, \mathcal{E})$ to $(S, \mathcal{S})$ also induces a σ-algebra when all events B in $\mathcal{S}$ are considered as a range for the mapping and closed under union and complementation. It is called the σ-algebra **generated by** X, denoted $\sigma(X)$, and defined as the collection of events:

$$\sigma(X) = \{\{\omega : X(\omega) \in B\} : B \in \mathcal{S}\}.$$

Expected Value. Let X be a non-negative random variable on $(\Omega, \mathcal{E}, P)$. The **expected value** (or the **first moment**) of X is $\mathbb{E}X \triangleq \int X dP$ is always well-defined but may be ∞. In general, the i-th moment of X is $m_i(X) \triangleq \int X^i dP$, for all $i \in \mathbb{N}$. If X can take on negative values, we define $\mathbb{E}X = \mathbb{E}X^+ - \mathbb{E}X^-$ where $X^+ \triangleq \max\{x, 0\}$ is the positive part of X and $X^- \triangleq \min\{-x, 0\}$. The expectation $\mathbb{E}X$ is then said to **exist** if either $\mathbb{E}X^+ < \infty$ or $\mathbb{E}X^- < \infty$.

Theorem 2.1.1 (Jensen's Inequality)**.** Let φ be a **convex** function, that is,

$$\lambda\varphi(x) + (1 - \lambda)\varphi(y) \geq \varphi(\lambda x + (1 - \lambda)y)$$

for all $\lambda \in (0, 1)$, $x, y \in \mathbb{R}$. Then

$$\varphi(\mathbb{E}X) \leq \mathbb{E}(\varphi(X))$$

provided that $\mathbb{E}|X| < \infty$ and $\mathbb{E}|\varphi(X)| < \infty$.

Linearity of Expectation. Let X, Y be two random variables from $(\Omega, \mathcal{E}, P)$ to $(S, \mathcal{S})$ and let $a, b, c \in \mathbb{R}$. We refer to the property

$$\mathbb{E}(aX + bY + c) = a\mathbb{E}X + b\mathbb{E}Y + c$$

as **linearity of expectation** provided the individual expectations $\mathbb{E}X$, $\mathbb{E}Y$ exist and are finite.

For two random variables X, Y from $(\Omega, \mathcal{E}, P)$ to $(S, \mathcal{S})$, we define the **conditional expectation** to be the random variable $\mathbb{E}(X \,|\, Y \in B) : \Omega \to S$:

$$\mathbb{E}(X \,|\, Y \in B) \triangleq \int_B X f_Y(y)\, dy \quad \text{for } B \in S,$$

where $f_Y(y)$ is the probability density of Y.

Independence. Given a probability space $(\Omega, \mathcal{E}, P)$, two events $A, B \in \mathcal{E}$ are **independent** if $P(A \cap B) = P(A)P(B)$. Two random variables X, Y are independent if for all $E_i, E_j \in \mathcal{E}$,

$$P(X \in E_i \wedge Y \in E_j) = P(X \in E_i)\, P(Y \in E_j),$$

or equivalently, if events $A \triangleq \{X \in E_i\}$ and $B \triangleq \{Y \in E_j\}$ are independent.

Finally, an important result [61] that allows the construction of countable probability spaces by means of induction.

Theorem 2.1.2 (Kolmogorov's Extension Theorem)**.** Suppose we are given probability measures μ_n on $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$ that are **consistent**, that is,

$$\mu_{n+1}((a_1, b_1] \times \cdots \times (a_n, b_n] \times \mathbb{R}) = \mu_n((a_1, b_1] \times \cdots \times (a_n, b_n])$$

Then there is a unique probability measure P on $(\mathbb{R}^{\mathbb{N}}, \mathcal{B}(\mathbb{R}^{\mathbb{N}}))$ with

$$P(\omega : \omega_i \in (a_i, b_i], 1 \leq i \leq n) = \mu_n((a_1, b_1] \times \cdots \times (a_n, b_n]).$$

2.2 Stochastic Processes, Martingales and Markov Processes.

Let $(\Omega, \mathcal{E}, P)$ be a probability space, $(S, \mathcal{S})$ be a measurable space, and T be a totally ordered set. A **stochastic process** $\mathcal{X} = \{X_t : t \in T\}$ is a collection of S-valued random variables **indexed**

by T ("time"). If T is a discrete set, i.e., $T = \{0, 1, 2, \ldots\}$ then we refer to $\mathcal{X}$ as a discrete time stochastic process; if T is continuous, i.e., $T = [0, \infty)$ then we refer to $\mathcal{X}$ as a continuous time stochastic process; if $(S, \mathcal{S}) = (\mathbb{R}, \mathcal{B})$ we refer to $\mathcal{X}$ as a real-valued stochastic process.

Remark 1. In this thesis we focus on real-valued discrete time stochastic processes.

A set $\mathcal{F} \subseteq \mathcal{E}$ is called a sub-σ-algebra if $\mathcal{F}$ is a σ-algebra. A family of increasing sub-σ-algebras $\mathcal{F}_0 \subseteq \cdots \subseteq \mathcal{F}_n \subseteq \mathcal{F}_{n+1} \subseteq \cdots \subseteq \mathcal{E}$, $n \geq 0$ is called a **filtration**. A stochastic process $\mathcal{X}$ is said to be **adapted** to filtration $\mathcal{F}_n$ if $X_n \in \mathcal{F}_n$ for all $n \geq 0$. One can think of $\mathcal{F}_n$ as containing the information about all events that occur up to time n. That is $\mathcal{F}_n = \sigma(X_0, \ldots, X_n)$ is the information available at time n. Thus, a filtration captures the information about the evolution of the adapted stochastic process.

A stochastic process $\mathcal{X}$ has the **Markov property** if for every $A \in \mathcal{S}$, $n \in \mathbb{N}$,

$$P(X_n \in A \,|\, \mathcal{F}_{n-1}) = P(X_n \in A \,|\, X_{n-1} = x_{n-1}),$$

i.e., the stochastic choice of X_n depends only on the outcome of X_{n-1} but not the history of $X_0, \ldots, X_{n-2}$. Such a process is therefore called a "memoryless" or Markov process.

A random variable τ taking on values $\{0, 1, \ldots\} \cup \{\infty\}$ is a **stopping time** if for every $n < \infty$, event $\{\tau = n\} \in \mathcal{F}_n$. The name comes from the fact that the decision whether to "stop" a process at some time n has to be made by only knowing the information up to time n that is by knowing only $\mathcal{F}_n$ but not $\mathcal{F}_{n+1}, \mathcal{F}_{n+2}, \ldots$, or $\mathcal{F}$.

Example 2.2.1. *Given a stochastic process $\mathcal{X}$ and an event $A \in \mathcal{S}$, an important example of a stopping time is the **hitting time** of A:*

$$\tau_A \triangleq \inf\{n : X_n \in A\}, \qquad \textit{i.e., the first time } \mathcal{X} \textit{ enters region } A.$$

This is a stopping time since the event:

$$\{\tau_A = n\} = \{\omega \,:\, X_0(\omega) \in A^C, \ldots, X_{n-1}(\omega) \in A^C, X_n(\omega) \in A\} \in \mathcal{F}_n.$$

An example of a "non-stopping time" (a non-example) is time at which $\mathcal{X}$ achieves its maximum value:

$$\tau_{bad} \triangleq \inf\{N : \forall n \geq N,\, X_n \leq X_N\}$$

because one needs to know the complete evolution of $\mathcal{X}$ to determine the maximum.

Let $\mathcal{X} : \{X_n\}_{n=0}^{\infty}$ be a stochastic process and τ be a stopping time, then $\mathcal{X}^{\tau}$ denotes the process $\mathcal{X}$ **stopped at** τ, that is,

$$\mathcal{X}^{\tau} \triangleq \{X_{\min(n,\tau)}\}_{n=1}^{\infty}.$$

2.2.1 Martingales

Martingales are a type of stochastic process that is fundamental to this thesis.

Definition 2.2.1 (Martingales)**.** *Let $\mathcal{F}_n$ be a filtration. Let $\mathcal{M} = \{M_0, M_1, \ldots\}$ be a discrete time stochastic process adapted to $\mathcal{F}_n$. The process $\mathcal{M}$ is called a* ***supermartingale*** *if:*

$$\mathbb{E}(M_{n+1} \,|\, \mathcal{F}_n) \leq M_n \quad \textit{for all } n \geq 0$$

The process $\mathcal{M}$ is called a ***submartingale*** *if:*

$$\mathbb{E}(M_{n+1} \,|\, \mathcal{F}_n) \geq M_n \quad \textit{for all } n \geq 0$$

If $\mathcal{M}$ is both a supermartingale and a submartingale, that is,

$$\mathbb{E}(M_{n+1} \,|\, \mathcal{F}_n) = M_n \quad \textit{for all } n \geq 0$$

then $\mathcal{M}$ is a ***martingale****.*

Martingales represent stochastic processes whose individual runs $\mathcal{M}(\omega)$ for some $\omega \in \Omega$ appear to be random fluctuations under the influence of stochastic noise. However, their collective behavior captured as "snapshots" of the conditional expected value $\mathbb{E}(M_{n+1}(\omega) \,|\, M_n(\omega) = m_n)$ is invariant on every step $n \geq 0$ and, therefore, is invariant over time: $\mathbb{E}(M_{n+1} \,|\, M_0) = M_0$.

Example 2.2.2. *Let $\{X_n\}_{n=0}^{\infty}$ be a stochastic process with $X_0 = 0$ defined for all $n \geq 1$ as follows:*

$$X_{n+1}(\omega) = \begin{cases} X_n - 1, & \text{with probability } p, p \in (0,1) \\ X_n + 1, & \text{with probability } (1-p) \end{cases}$$

Then

$$\mathbb{E}(X_{n+1} \,|\, \mathcal{F}_n) = p(X_n - 1) + (1-p)(X_n + 1) = X_n + (1 - 2p).$$

Notice that when $p < \frac{1}{2}$ then X_n is a submartingale $\mathbb{E}(X_{n+1} \,|\, \mathcal{F}_n) \geq X_n$; when $p = \frac{1}{2}$ then X_n is a martingale $\mathbb{E}(X_{n+1} \,|\, \mathcal{F}_n) = X_n$; and finally, when $p > \frac{1}{2}$ then X_n is a supermartingale $\mathbb{E}(X_{n+1} \,|\, \mathcal{F}_n) \leq X_n$.

Similarly, supermartingales represent stochastic processes whose expected value decreases (does not increase) over time.

Theorem 2.2.1 (Martingale Convergence Theorem [61, Theorem 5.2.8])**.** Let $\mathcal{X} : \{X_n\}_{n=0}^{\infty}$ is a submartingale with $\sup \mathbb{E}X_n^+ < \infty$, then X_n converges a.s. to a limit X with $\mathbb{E}|X| < \infty$.

A special case of the Martingale Convergence Theorem [61, Theorem 5.2.9] that is better suited to our framework is the following.

Theorem 2.2.2 (Supermartingale Convergence Theorem)**.** Let $\mathcal{X} : \{X_i\}_{i=0}^{\infty}$ is a non-negative supermartingale, then as $n \to \infty$, $X_n \to X$ a.s. and $\mathbb{E}X \leq \mathbb{E}X_0$.

In the case of Theorem 2.2.2, the limit is the random variable $X_\infty = \liminf_{n\to\infty} X_n$ which need not be part of the set of support of $\mathcal{X}$ for convergence result to hold.

Example 2.2.3. *Consider the stochastic process defined by $X_0 = 1$ and $X_n = U_n X_{n-1}$ where U_n is a uniform real random variable on the range $(0,1)$. Then $\mathcal{X} : \{X_n\}_{n=0}^{\infty}$ is a supermartingale since $\mathbb{E}(X_{n+1} \,|\, X_n) = \mathbb{E}(U_n X_n \,|\, X_n) = \mathbb{E}(U_n) X_n = \frac{1}{2} X_n \leq X_n$. Moreover, X_n is positive for every $n \in \mathbb{N}$ and therefore converges a.s. to $X_\infty = 0$.*

Additive and Multiplicative Supermartingales.

Definition 2.2.2. *Let $\mathcal{M} = \{M_i\}_{i=0}^{\infty}$ be a supermartingale. Then $\mathcal{M}$ is called ε-**additive** if and only if*

$$\exists \varepsilon > 0, \quad \mathbb{E}(M_{n+1}(\omega) \,|\, \mathcal{F}_n) \leq M_n(\omega) - \varepsilon, \quad \textit{for all } n \geq 0.$$

*The non-negative supermartingale $\mathcal{M} : \{M_n(\omega) \geq 0\}_{n=0}^{\infty}$ is called α-**multiplicative** if and only if*

$$\exists \alpha \in (0,1), \quad \mathbb{E}(M_{n+1}(\omega) \,|\, \mathcal{F}_n) \leq \alpha M_n(\omega), \quad \textit{for all } n \geq 0.$$

Lemma 2.2.1. Let $\mathcal{M} = \{M_i\}_{i=0}^{\infty}$ be a nonnegative α-multiplicative supermartingale for some $\alpha \in (0,1)$. Then $\widehat{\mathcal{M}} = \{\widehat{M_i} : \frac{M_i}{\alpha^i}\}_{i=0}^{\infty}$ is a nonnegative supermartingale.

Proof. Without loss of generality, let $\omega \in \Omega$ and $\widehat{M_n}(\omega) = \widehat{m}_n$ for some $\widehat{m}_n$. We need to show that $\mathbb{E}(\widehat{M_{n+1}} \,|\, \widehat{M_n} = \widehat{m}_n) \leq \widehat{m}_n$.

Notice $\widehat{M_j}$ is a deterministic function of M_j for all $j \geq 0$. Therefore,

$$\begin{aligned}
\mathbb{E}(\widehat{M_{n+1}} | \widehat{M_n} = \widehat{m}_n) &= \mathbb{E}\left(\frac{M_{n+1}}{\alpha^{n+1}} \,\middle|\, \frac{M_n}{\alpha^n} = \widehat{m}_n\right), \text{ (by definition)} \\
&= \left(\frac{1}{\alpha}\right) \frac{\mathbb{E}(M_{n+1} | M_n = \alpha^n \widehat{m}_n)}{\alpha^n}, \text{ (by linearity of expectation)} \\
&\leq \left(\frac{1}{\alpha}\right) \frac{\alpha M_n}{\alpha^n} \;\; = \;\; \widehat{m}_n.
\end{aligned}$$

□

Theorem 2.2.3. Let $\mathcal{M} = \{M_i\}_{i=0}^{\infty}$ be nonnegative α-multiplicative supermartingale for some $\alpha \in (0,1)$. Then $\mathcal{M}$ converges almost surely (samplewise) to 0.

Proof. From Lemma 2.2.1, we conclude that $\widehat{M} : \{\frac{M_i}{\alpha^i}\}_{i=0}^{\infty}$ is a non-negative supermartingale. Applying the standard **supermartingale convergence theorem** (Theorem 2.2.2), we note that every nonnegative supermartingale converges samplewise almost surely. Therefore, for any sample ω, the sequence $\widehat{M_0}(\omega), \widehat{M_1}(\omega), \ldots$, converges to some finite value $\widehat{M^*}(\omega)$. Next, notice that the sequence $1, \alpha, \alpha^2, \ldots$, converges to zero. Moreover, the product of two convergent sequences is also convergent. Therefore, the sequence $\widehat{M_i}(\omega) \times \alpha^i \equiv M_i(\omega)$ converges to the product $\widehat{M^*} \times 0 = 0$. This proves $\mathcal{M}$ converges almost surely to 0. □

2.2.2 Discrete Time Stochastic Systems.

Stochastic processes provide a general framework that captures a rich class of processes that operate under uncertainty. In order to make explicit the laws that govern the evolution of the infinite-state stochastic systems we are interested in, we develop a model of infinite-state discrete time stochastic transition systems (DTSTS).

Let $(\Omega, \mathcal{E}, P)$ be a probability space. Let $(S, \mathcal{S})$ be a measure space and let $\mathcal{X}$ be an S-valued stochastic process. We refer to S as the **state space** on which $\mathcal{X}$ evolves. For the probabilistic transition systems we define next, we distinguish between the **state (or program) variables** $X = \{x_1, \ldots, x_n\}$ and the **random variables** $R = \{r_1, \ldots, r_m\}$. We use the notation $\mathbf{x}, \mathbf{r}$ to denote a valuation of the state, respectively, the random variables, and, $\mathbf{x}, \mathbf{x}'$ to denote the current, and respectively, the next state of the process, that is: $X_n = \mathbf{x}$ and $X_{n+1} = \mathbf{x}'$ for some $n \geq 0$. The evolution of the stochastic systems studied here is given in the form of a piecewise polynomial stochastic difference equation $\mathbf{x}' := F(\mathbf{x}, \mathbf{r})$ and an initial distribution $\mathcal{D}_0$, where the initial state $\mathbf{x}_0$ is drawn according to initial distribution $\mathcal{D}_0$.

We make this definition precise:

Definition 2.2.3 (DTSS)**.** *A **discrete-time stochastic system** (DTSS)* Π *is defined as the tuple* $\langle S, \mathcal{R}, \mathcal{F}, \mathcal{D}_0 \rangle$ *with the following components:*

(1) a state space S and an associated σ-algebra $\mathcal{S}$ on it,

(2) a probability space $\mathcal{R} : \langle R, \mathcal{F}_R, P_R \rangle$ from which random samples $\mathbf{r}$ are drawn,

(3) an update function $\mathcal{F} : S \times R \to S$, wherein $\mathcal{F}(\mathbf{x}, \mathbf{r})$ denotes the next state obtained from a state $\mathbf{x} \in S$ and random sample $\mathbf{r} \in R$,

(4) an initial probability distribution $\mathcal{D}_0$ over S.

In Definition 2.2.3 we make two assumptions that carry over to extensions of the DTSS model:

(1) **No Nondeterminism** - We define the update mapping $\mathcal{F}$ to be a function.

(2) **Independence of Samples** - The formulation above naturally assumes that the samples of the random variable $\mathbf{r}_i$ are drawn independent of the current state $\mathbf{x}_i$ and from previous samples $\mathbf{r}_0, \ldots, \mathbf{r}_{i-1}$.

Example 2.2.4 (Strange Random Walk)**.** *Let $\mathcal{Y} = \{Y_i\}_{i=1}^{\infty}$ be a real-valued stochastic process with Y_0 distributed uniformly over $[0, 1]$. For all $n \geq 0$, define:*

$$Y_{n+1} = \begin{cases} Y_n^2, & \text{with probability } \frac{1}{2}, \\ 2Y_n - Y_n^2, & \text{with probability } \frac{1}{2}. \end{cases}$$

The corresponding discrete time stochastic system is $\Pi : \langle \mathbb{R}, \mathcal{R}, \mathcal{F}, \mathcal{D}_0 \rangle$, where $\mathcal{R}$ is the probability space for the uniform distribution $\mathcal{U}(0, 1)$, the initial probability distribution $\mathcal{D}_0$ is $\mathcal{U}(0, 1)$, and update function is $\mathcal{F} : (\mathbf{x}, \mathbf{r}) \mapsto \mathbb{1}_{\{\mathbf{r}_b \leq 1/2\}}(\omega) \times (\mathbf{x}^2) + \mathbb{1}_{\{\mathbf{r}_b > 1/2\}}(\omega) \times (2\mathbf{x} - \mathbf{x}^2)$.

Trace Semantics. Let $\Pi : \langle S, \mathcal{R}, \mathcal{F}, \mathcal{D}_0 \rangle$ be a stochastic system as in Definition 2.2.3, with state space $S \subseteq \mathbb{R}^n$ and the Borel σ-algebra $\mathcal{S}$ over S. The sample set $\Omega : S \times R^{\omega}$ consists of tuples $\langle \mathbf{x}_0, \mathbf{r}_0, \mathbf{r}_1, \cdots, \mathbf{r}_n, \cdots \rangle$. Here $\mathbf{x}_0$ is the initial state sampled from distribution $\mathcal{D}_0$, and $\mathbf{r}_0, \mathbf{r}_1, \ldots, \mathbf{r}_n, \ldots$ denote successive draws from the sample set R of the probability space $\mathcal{R}$.

We define a discrete-time interpretation of the stochastic system to be the stochastic process π, i.e. a countable sequence of random variables $\{\pi_i\}_{i=0}^{\infty}$, where each $\pi_i : S \times R \to S$ maps a state $\mathbf{x}_i$ and random variables $\mathbf{r}_i$ to the next state $\mathbf{x}_{i+1}$. As a whole π maps samples $\omega \in \Omega$ to traces (or sample executions) of the form:

$$\pi(\omega) : \ \mathbf{x}_0 \xrightarrow{\mathbf{r}_1} \mathbf{x}_1 \xrightarrow{\mathbf{r}_2} \cdots \xrightarrow{\mathbf{r}_n} \mathbf{x}_n \cdots$$

where each state update is an application of $\mathcal{F}$. Therefore, each random variable $\pi_i \in \pi$ is measurable w.r.t. the product σ-algebra $\mathcal{S} \times \mathcal{F}_R$, and induces a probability measure $\mu_{i+1} : \mathcal{S} \to [0, 1]$ over the next states.

For any $n \geq 0$ and measurable sets $S_1, \ldots, S_{n+1} \in \mathcal{S}$, define

$$\bar{\mu}_{n+1}(S_1 \times \cdots \times S_{n+1}) \triangleq \mu_1(S_1)\mu_2(S_2) \cdots \mu_n(S_n)\mu_{n+1}(S_{n+1}).$$

Then $\bar{\mu}_{n+1}$ is a probability measure. Moreover, $\bar{\mu}_{n+1}$ is a **consistent** probability measure since

$$\bar{\mu}_{n+1}(S_1 \times \cdots \times S_n \times S) = \mu_1(S_1)\mu_2(S_2)\cdots\mu_n(S_n).$$

Kolmogorov's Extension Theorem (Theorem 2.1.2) yields the following unique probability measure P over $(S^{\mathbb{N}}, \mathcal{S}^{\mathbb{N}})$:

$$P(\omega : \omega_i \in S_i, 1 \leq i \leq n) \triangleq \mu_n(S_1 \times \cdots \times S_n).$$

Let $\pi(\omega)$ be the sample path defined by sample ω. The probability of path $\pi(\omega)$ is then defined as $Pr(\pi(\omega)) = P(\omega)$. We refer to this measure as the **path probability** measure.

The stochastic process π is the **sample-based** or **operational semantics** of the system Π.

Denotational Semantics. The **denotational semantics** $[\![\Pi]\!]$ of the stochastic system $\Pi : \langle S, \mathcal{R}, \mathcal{F}, \mathcal{D}_0 \rangle$ can be thought of as a linear operator transforming distributions of system states:

$$\mathcal{D}_0 \xrightarrow{[\![\Pi]\!]} \mathcal{D}_1 \xrightarrow{[\![\Pi]\!]} \cdots \xrightarrow{[\![\Pi]\!]} \mathcal{D}_n \xrightarrow{[\![\Pi]\!]} \cdots$$

where for all $n \geq 0$, $\mathcal{D}_n$ denotes the distribution of system states after the system performs n steps of execution.

Formally, for $n \geq 0$, distribution $\mathcal{D}_n$ is defined by: (i) the probability space $\mathcal{P}_n : \langle \Omega_n, \mathcal{S}_n, \mu_n \rangle$ with sample set $\Omega_n : S \times R^n$, σ-algebra $\mathcal{S}_n$ over Ω_n giving the set of events, probability measure μ_n; and, (ii) the $\mathcal{S}_n$-measurable random variable π_n mapping samples $\omega \in \Omega_n$ to the next system state. The distribution over random variables $\mathbf{r}$ is defined by the probability space $\mathcal{R} : \langle R, \mathcal{F}_R, P \rangle$ and the $D_R : R \to \mathbb{R}^m$ mapping samples to the values of random variables $\mathbf{r}$.

Let $\mathcal{P}_n \otimes \mathcal{R}$ be the product space with sample set $\Omega_{n \otimes R} : \Omega_n \times R$, product σ-algebra $\mathcal{F}_{n \otimes R}$ generated by sets $S_n \times R_j$ where $S_n \in \mathcal{S}_n$, $R_j \in \mathcal{F}_R$, and the probability measure

$$\mu_{n \otimes R}(S_n \times R_j) = \mu_n(S_n)P(R_j) \quad \text{for } S_n \times R_j \text{ in } \mathcal{F}_{n \times R}.$$

The next state of the system is given by the random variable $\pi_{n+1} : \Omega_{n \otimes R} \to \Sigma$ defined as:

$$\pi_{n+1}(\omega_n, \omega_R) : \{\mathbf{x}' \,|\, \pi_n(\omega_n) = \mathbf{x},\, D_r(\omega_R) = \mathbf{r},\, \mathbf{x}' = \mathcal{F}(\mathbf{x}, \mathbf{r})\}.$$

Distribution $\mathcal{D}_{n+1}$ is then the probability measure induced by π_{n+1} on the probability space $\mathcal{P}_n \otimes \mathcal{R}$.

The denotational semantics $[\![\Pi]\!]$ can be thought of as the collecting semantics of Π.

Pre-Expectations. Key to the analysis is the notion of **pre-expectation**. The definitions below are inspired by [105, 121] and are related to drift operators of Markov processes [122]. We formalize the notion of pre-expectations over general stochastic systems.

Consider a stochastic system $\Pi : \langle S, \mathcal{R}, \mathcal{F}, \mathcal{D}_0 \rangle$, and a function $h : S \to \mathbb{R}$ over the state-space. The **pre-expectation** of h w.r.t to $\mathcal{F}$ is another function $\hat{h} : S \to \mathbb{R}$ such that for any state $\mathbf{x} \in S$, $\hat{h}(\mathbf{x})$ yields the expected value of $h(\mathbf{x}')$, where the expectation is taken over all states $\mathbf{x}'$ reached in one step from $\mathbf{x}$, that is,

$$\hat{h}(\mathbf{x}) : \ \mathbb{E}_R \left(h(\mathcal{F}(\mathbf{x}, \mathbf{r})) \right).$$

In general, a pre-expectation can be difficult to compute for a stochastic system, even if $h(\mathbf{x})$ is of a simple form, for example, polynomial.

Example 2.2.5. *Consider the DTSS* Π *corresponding to the Strange Random Walk of Example 2.2.4 and let* $h(\mathbf{x}) = \mathbf{x}$ *be a function. The pre-expectation of* h *is computed as follows:*

$$\hat{h}(\mathbf{x}) = \mathbb{E}_R[\mathbb{1}_{\{\mathbf{r}_b \leq 1/2\}}(\omega)(\mathbf{x}^2) + \mathbb{1}_{\{\mathbf{r}_b > 1/2\}}(\omega)(2\mathbf{x} - \mathbf{x}^2) = \frac{1}{2}(\mathbf{x}^2) + \frac{1}{2}(2\mathbf{x} - \mathbf{x}^2) = \mathbf{x}.$$

This means for DTSS Π *function* h *is invariant under the pre-expectation transformation (i.e.,* $\hat{h} = h$*). This notion of invariance is closely related to the notion of stochastic invariance that martingales provide and is the subject of Chapter 4.*

2.3 Probabilistic Properties and Concentration of Measure.

In this section we describe some of the properties of interest for the class of stochastic systems described in the previous section.

Let Π be a discrete time stochastic system over some state-space S with an associated σ-algebra $\mathcal{S}$. Let $\Omega : S \times R^{\omega}$ be the set of outcomes and Pr the path probability that maps a measurable set $T \subseteq \Omega$ to its probability $Pr(T)$. Let π be a function that maps each sample $\omega \in \Omega$ to the corresponding sample path of the system $\pi(\omega) : \langle \mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_m, \ldots \rangle$. Likewise, let π_m map each sample $\omega \in \Omega$ to the state encountered at time m, i.e., $\pi_m(\omega) : \mathbf{x}_m$.

Almost Sure Reachability. For a predicate φ over the system states, the **reachability** property $\Diamond\varphi$ is a collection of sample paths

$$[\![\Diamond\varphi]\!] \triangleq \{\omega \in \Omega \,|\, \exists\, n \geq 0, \pi_n(\omega) \models \varphi\}$$

that eventually (or within some finite time) satisfy φ. This is measurable as it is the countable union of measurable sets. The probability of the reachability property $\Diamond\varphi$ is denoted $\Pr(\Diamond\varphi)$. We say a reachability property $\Diamond\varphi$ holds almost surely (a.s.) iff $Pr(\Diamond\varphi) = 1$.

Almost Sure Termination: Given a region $\top \subseteq S$ of designated **final** (or **terminating** states) and the membership predicate (also known as the characteristic function)

$$\varphi_\top(\mathbf{x}) \triangleq \begin{cases} 1, & \text{if } \mathbf{x} \in \top \\ 0, & \text{otherwise} \end{cases}$$

the almost sure termination problem asks whether $Pr(\Diamond\varphi_\top) = 1$. Henceforth, we identify membership predicates φ_T with the set of states $T \subseteq S$ they characterize.

Almost Sure Repeated Reachability. For a predicate φ over the system states, the **persistence** property $\Diamond\Box\varphi$ is the collection of sample paths

$$[\![\Diamond\Box\varphi]\!] \triangleq \{\omega \in \Omega \mid \exists n \geq 0, \forall m \geq n, \pi_m(\omega) \models \varphi\}$$

whose execution reaches φ in some finite number of steps m and remains in φ forever. The probability of the persistence property $\Diamond\Box\varphi$ is denoted $\Pr(\Diamond\Box\varphi)$. We say the persistence property $\Diamond\Box\varphi$ holds almost surely if and only if $\Pr(\Diamond\Box\varphi) = 1$.

Similarly, a **recurrence** property $\Box\Diamond\varphi$ is the collection of sample paths

$$[\![\Box\Diamond\varphi]\!] \triangleq \{\omega \in \Omega \mid \forall n \geq 0,\ \exists\ m \geq n, \pi_m(\omega) \models \varphi\}$$

along which execution returns infinitely often to φ. We say that the recurrence property $\Box\Diamond\varphi$ holds almost surely if and only if $\Pr(\Box\Diamond\varphi) = 1$.

Qualitative Properties: The three properties stated above are also known as qualitative (in contrast with quantitative properties we describe next) reachability, persistence and recurrence

properties and can be stated in the probabilistic temporal logic PCTL* respectively as the formulas $\mathbb{P}_{=1}(\Diamond\varphi)$, $\mathbb{P}_{=1}(\Diamond\Box\varphi)$ and $\mathbb{P}_{=1}(\Box\Diamond\varphi)$.

Probabilisitic Assertions. In addition to qualitative properties that hold either with probability 0 or with probability 1, there is another important class of **quantitative** probabilistic properties in which probability can range between $[0, 1]$.

Given the stochastic system Π above and a state predicate φ, a probabilistic assertion is of the form: $P(\varphi) \leq \gamma$, where $\gamma \in [0, 1]$ is a probability bound. This formula encodes two types of problems:

(1) A verification query - Given a constant γ does the probability of φ exceed the probability bound?

(2) An estimation query - Can you estimate a tight upper (or lower) probability bound γ?

Quantitative properties have been extensively studied in the literature. Here we mention these for two reasons. First, to point out that we have also studied these in Sankaranarayanan et al. [148] and Bouissou et al. [23] but that work is not part of this thesis. Second, to demonstrate how the theory of martingales we develop here naturally enables the application of concentration of measure results [60] that can answer the estimation query.

Concentration of Measure. Concentration of measure is the phenomenon whereby a large portion of the probability measure associated with a random variable X concentrates within a "narrow" range around the expected value $\mathbb{E}X$. This phenomenon has been formalized in a series of increasingly tight probability bound inequalities: Markov's Inequality, Chebyshev's Inequality, etc. We focus on the result that leverages martingales [60] and its extensions [64], [13, Chapter 3].

Theorem 2.3.1 (Azuma-Hoeffding Theorem)**.** Let $\mathcal{M} : \{M_n\}_{n=0}^{\infty}$ be a bounded supermartingale, i.e., $a_n \leq M_n - M_{n-1} \leq b_n$ for all $n > 0$. Then for all $n \in \mathbb{N}$ and $t \in \mathbb{R}$ such that $t \geq 0$,

$$P(M_n - M_0 \geq t) \leq \exp\left(\frac{-t^2}{2\sum_{k=1}^{n}(b_k - a_k)^2}\right).$$

Moreover, if $\mathcal{M}$ is a martingale the symmetric bound holds as well:

$$P(M_n - M_0 \leq -t) \leq \exp\left(\frac{-t^2}{2\sum_{k=1}^{n}(b_k - a_k)^2}\right).$$

Combining both bounds, we conclude that for a martingale $\mathcal{M}$ we obtain

$$P(|M_n - M_0| \geq t) \leq 2\exp\left(\frac{-t^2}{2\sum_{k=1}^{n}(b_k - a_k)^2}\right).$$

Azuma-Hoeffding bound is a **concentration of measure inequality**.

Example 2.3.1. *Consider the supermartingale process* $\{X_n\}_{n=0}^{\infty}$. *Notice that for all* $n \geq 1$, $a \leq X_n - X_{n-1} \leq b$ *with* $a = -1$ *and* $b = 1$. *Fix* $t = 30, n = 50$, *then the Azuma-Hoeffding bound is*

$$P(M_n - M_0 \geq t) = P(M_{50} \geq 30) \leq \exp\left(\frac{-t^2}{2n(b-a)^2}\right) \leq \exp\left(\frac{-(30)^2}{2(50)(2)^2}\right) = 0.1054.$$

Chapter 3

Overview

3.1 Probabilistic Programs

Probabilistic programs provide a common framework for modeling stochastic processes. Probabilistic programs are imperative (or functional) programs augmented with two new features: (*i*) the ability to generate (or **sample**) random values from built-in standard distribution primitives such as `rand()`, `Bernoulli(p)`, `Uniform(lb,ub)`, `Gaussian(mu,sigma)`, etc.); (*ii*) the ability to filter (or **condition**) the execution of the program based on Boolean predicates over the program variables. For example, consider the following simple probabilistic program:

```
1  int x = Uniform(-5,3);
2  int y = Uniform(-3,5);
3  int count = 0;
4  while (x + y <= 10) {
5     if flip(3/4){
6        x = x + Uniform(0,2);
7        y = y + 2;
8     }
9     count++;
10 }
```

The program defines program variables `x`, `y`, `count` and initializes `x` and `y` according to draws from uniform distributions over the corresponding real intervals. On every iteration of the loop the program checks that the Boolean predicate ($x + y <= 10$) is satisfied and independent of any previous

random draws "flips" a biased coin and with probability $3/4$ chooses to increment $\mathtt{x}, \mathtt{y}$; whereas, with complementary probability $1 - (3/4) = 1/4$ it leaves the values of the program variables unchanged. Variable `count` tracks the number of iterations of the loop.

This probabilistic program models (or, equivalently, defines) a discrete time stochastic process Π as in Definition 2.2.3 whose sample executions $\pi(\omega)$ and distributions $\mathcal{D}_n$ over the possible values of program variables $\mathtt{x}, \mathtt{y}, \mathtt{count}$ we visualize in Figure 3.1. We focus on probabilistic programs with loops.

Semantics. The first formal semantics of imperative probabilistic programs is defined in a seminal work by Kozen [108]. He defines two types of semantics: an operational one based on measurable functions over measurable spaces that a machine equipped with an infinite stack of random choices operates; and a second, equivalent one, based on continuous linear operators on a Banach space of measures. The former represents the view that probabilistic programs are essentially deterministic functions given that all random choices have been resolved a priori. This corresponds to drawing the sample $\omega \in \Omega$ that "drives" a stochastic process as defined in Section 2.2.2 and it is the basis for the operational semantics we build our polynomial stochastic transition system (PSTS) model upon (see Section 4.1). The first semantics has inspired most of the subsequent weakest precondition type analysis approaches (see Section 3.4, **weakest precondition**) whereas the second one motivated distribution transformers and abstract interpretation efforts based on Markov decision process (MDP) (see Section 3.4, **abstract interpretation**).

Other notable works to the development of semantics of probabilistic programs are the Giry monad [78, 133], probabilistic powerdomain of Jones [101] that provides formal grounds to interpret nondeterminism in probabilistic programs as **sets** of possible probability distributions, probabilistic predicate dynamic logic [109], probabilistic concurrent constraint programming [85, 57], and others.

Recent work by Gordon et al. [81] focuses on conditioning and inference as semantic elements central to probabilistic programming. Gretz et al. [84, 83] establish an equivalence of operational semantics based on MDP with rewards and the weakest precondition (pre-expectation) calculus of McIver and Morgan [121] for probabilistic programs over discrete distributions. Jansen et al. [100]

provide a formal semantic definition of conditioning within the same context.

Properties, Program Analysis and Verification. From the standpoint of formal methods and program analysis probabilistic programs present three major challenges:

(1) They model physical processes that manipulate continuous quantities that are often modeled as real-valued program variables. The stochastic systems that such programs define are *infinite state*.

(2) Due to the uncertainty inherent in these programs, we are forced to maintain both a *set* S of all possible states the system could occupy and a probability distribution *function* μ_S over S to capture the likelihood of each state. As indicated in Figure 4.6 both S and μ_S vary along the executions of a probabilistic program. This poses an efficient *representation* and **propagation** challenges for most automated analysis tools.

(3) Program variables are random variables. This means that predicates over program variables are also random variables which in turn induces *stochastic control flow* of the programs.

The stochastic nature of probabilistic programs makes formal reasoning about the properties of these programs challenging but also intriguing. Fundamental trace properties such as termination, reachability and invariance take on a stochastic interpretation. For example, the question: *Does every possible execution of the motivating probabilistic program reach a target state* ($\texttt{x} + \texttt{y} > \texttt{10}$ *for termination*)*?*, becomes *Does the set of all terminating executions have probability* 1*?*. Properties that hold with probability 1 are called **qualitative** properties.

A new class of **quantitative** properties emerges that seeks to provide quantitive bounds on the probability of events. Examples of such properties are: *What is the probability that* $\texttt{x} > 5$ *after* $n = 25$ *iterations of the loop? What is the expected value of* `y` *upon program termination? What is the expected number of loop iterations before terminating?* Both types of properties can be expressed as **probabilistic assertions** (see Section 2.3); however, care must be taken to ensure such properties are well-defined. In order to formally define stochastic trace properties probabilistic extensions to modal temporal logics such as probabilistic branching time logic PCTL [87, 10] have

been developed. Each PCTL property is no longer satisfied or falsified like its nonprobabilistic analogue but it is done so **with some probability**.

Given a probabilistic program P and a probabilistic assertion φ, the **verification problem** is to determine whether the probabilistic assertion holds. Determining the probability with which a probabilistic assertion holds (or an upper and lower bound on it) is called **probabilistic inference**.

3.2 Explicit State Verification

Explicit state verification encompasses a set of techniques that either represents or reasons about the probability of individual program states. Because probabilistic programs provide infinite state verification problems these techniques necessarily have to provide either aggregate *statistical* guarantees, or provide sound means of extrapolating from finitely many observed states to the infinitely many unobserved ones.

Simulation Based Techniques. Simulation approaches such as Monte-Carlo techniques [145] provide a quick and efficient way to empirically observe individual behaviors of a probabilistic program by simply executing it. The strength of these approaches lies in the fact that in the presence of complete dynamic information it is easy to sample and evaluate even highly non-linear, discontinuous expressions over parametric distributions. While highly accurate, these approaches lack formal guarantees for coverage and convergence. The outcomes of such experiments are then analyzed **in aggregate** using statistical analysis techniques and results incorporate a confidence estimate of quality or reproducibility of the inferred facts [99].

Because of the efficiency of Monte-Carlo simulations, many of the probabilistic symbolic execution techniques we present next use it to guide the executions of the analysis. Statistical explicit state model checking also relies on it to speed up estimation of probability bounds on probabilistic assertions. Finally, some volume computation and Bayesian inference techniques use it to numerically integrate probability distributions within a bounded range.

We use Monte-Carlo simulations throughout this dissertation to visualize the behavior of the probabilistic programs in the form of individual sample traces and intermediate distribution over

the values of program variables. We briefly explore statistical results in the context of quantitative properties studied in Chapter 6.

Probabilistic Symbolic Execution. Probabilistic symbolic executions [76, 148, 69, 42, 116, 70, 68, 19, 20, 18] uses standard symbolic execution [107] to *symbolically* represent all program states whose execution follows the same program path via symbolic path constraints. Monte-Carlo simulation in combination with statistical hypothesis testing [148] or exhaustive enumeration of program paths up to a bounded depth has been used to collect a sufficient set of program path constraints. Probabilistic program slicing [93] has been used to reduce the number of relevant paths. Finally a probability computation is performed for each path to determine the path probability as well as the probability of satisfying a probabilistic assertion along that path. Approaches to probability computation rely either on model counting or volume computation. Model counting is performed using boolean satisfiability (SAT) or satisfiability modulo theory (SMT) solvers [73, 38] that count the number of satisfying assignments for a path formula. Volume computation relies on precise tools like LattE integrale [55] that use improved versions of Barvinok's algorithm [12] to compute the number of discrete lattice points within a convex polyhedron (weighted by the probability of each state), on approximate numerical integration techniques [148, 19], or on provably correct probabilistic samplers [94] to compute the probability of a path formula involving real-valued variable constraints.

The estimation problem then reports the sum of the probabilities of the assertion along the observed individual paths; whereas the verification problem considers both this sum and the total probability of the unseen paths as the complement of the path probability of all observed paths.

Probabilistic Model Checking. Probabilistic model checking [10] leverages results from probability theory on finite state Markov chains and Markov decision processes [142] to extend standard (non-probabilistic) model checking [44] to reason over finite state *probabilistic* transition models. The ability to symbolically represent finite state distributions using multi-terminal bounded decision diagrams (MTBDDS) [74] in combination with symbolic model checking [28] are behind the wide success of probabilistic symbolic model checkers such as PRISM [111] (see [1] for

a survey of applications).

PRISM takes in a finite state transition system in the form of a discrete or continuous time Markov chain (DTMC, CTMC [110]), or a Markov Decision Process (MDP) [71] and a probabilistic assertion φ written as a probabilistic branching time logic PCTL [87] formula. The property is then verified to hold along the nodes and paths of the model graph as with standard (non-probabilistic) model checking. The probability computation is built up in a bottom up fashion in the structure of the formula φ. The probability bound derived for the initial state of the system is reported as the probability of the assertion.

Statistical model checking [43] reduces the effort of computing the exact probability of an assertion by providing high *confidence* approximantion on the probability that the assertion holds. This is done by integrating the efficiency of sampling with the statistical guarantees that hypothesis testing provides as one observes a small but sufficient number of samples [157]. This eliminates the need for exhaustive computation over the full transition system model.

The downside of the model checking approach is that it is mostly restricted to finite state probabilistic models. This means that all stochastic processes with polynomial dynamics we consider in this dissertation cannot be handled directly and would require some form of discretization and finitezation procedure. This could be accomplished by extending probabilistic counter-example guided abstraction refinement framework of Hermanns et al. [91] to infinite state probabilistic programs with polynomial guards and updates we consider here.

For the most recent advances in probabilistic model checking see the survey by Katoen [104].

3.3 Probabilistic Abstract Interpretation

Abstract Interpretation [51] provides a framework for safely overapproximating the executions of a program P. Abstraction allows for a finite representation (called an **abstract element**) of potentially infinite number of concrete states of a system by carefully selecting what information to capture by the elements of the **abstract domain** (and what to "abstracted away"). The goal of abstract interpretation is to infer invariants that overapproximate the set of all possible

behaviors of P providing alternate "abstract" semantics for P that manipulate the finite representations (abstract elements) instead. The strength of abstract interpretation lies in the fact that it automatically performs abstraction, loop invariant generation, and, therefore, resolves program termination. However, the choice of a good abstract domain is essential to the success of this analysis approach.

Abstract domains for probabilistic programs were first considered by Monniaux [126], by enriching standard abstract domains such as intervals and polyhedra with bounds on the measure concentrated in each element of the domain. Unfortunately, this approaches suffers from the fact that the domain does not track *how* the measure is distributed within each abstract element. This is due to the fact that join (and widening) operations overapproximate the set of reachable states and measure needs to extend even over the unreachable states artifact of the join (or widening). The amount of imprecision that accumulates over the course of the abstract interpretation analysis often leads to a failure case when the probability of a query is estimated to be anywhere in $[0, 1]$. A refinement of this idea was presented by Smith [149] where the abstract domain is selected such that the measure over the elements follows a truncated Gaussian distribution. This analysis unfortunately also proved to be ineffective when reasoning about correlated program variables.

Refinements of Monniaux's original approach appears in the work of Mardziel et al [119] that in addition tracks the number of discrete points within each polyhedral abstract element as well as bounds on the measure associated with any discrete point. This allows the analysis to track distributions more precisely but is prohibitively expensive computationally because after every step of program execution the analysis relies on the LattE integrale [55] to compute the number of discrete points in a polyhedron. Bouissou et al. [22, 23] partly alleviate the problem of correlated program variables by extending abstract domains with probabilistic affine forms that symbolically track limited dependence information between program expressions.

Monniaux also presents a backward abstract interpretation scheme to compute the probability of an observable assertion at the program output, and characterize the output distribution [128]. This approach goes along the lines of Kozen's second semantics [108] and focuses on measurable

functions. The backwards approach treats the program as a measurable function, and the backward abstract interpretation follows the natural definition of the output distribution through the inverse mapping [40]. However, the approach requires a user generated query or a systematic gridding of the output states to define the distribution.

Di Pierro et al [56] present a non-standard approach to probabilistic abstract interpretation based on the notion of distance between probability measure functions. Abstraction is a non-invertable operator between Hilbert spaces of probability measures. However, instead of defining a concretization based on overapproximation of all possible concrete states the Moore-Penrose pseudo-inverse operator is used to provide a single "closest" concretized measure over a potentially different set of support.

Cousot and Monerau [53] present a systematic and general abstract semantics for probabilistic programs that views the abstract probabilistic semantics obtained by separately considering abstractions of the program semantics, the probability (event) space, and a "law abstraction". The abstraction law is a function mapping abstract states to the distribution over the set of possible abstract next states obtained from a single step of program execution. Their approach conveniently captures existing techniques as instances of their framework, while providing new ways of abstracting probabilistic program semantics.

Summary. In general probabilistic abstract interpretation analyses, that overapproximate distributions over the reachable states tend to be ill-suited for analysis of probabilistic programs with approaches by Bouissou et al. [22, 23] and, despite computationally expensive, Mardziel et al. [119] being most practical.

Therefore, the work of this dissertation takes a different approach: we define an abstract domain over the moments of probability distributions. Based on our current understanding, the approach of using martingale expressions to reason about probabilistic programs fits into their framework by viewing expectation invariants of Chapter 7 as representing sets of distributions (with constraints over their first moments); and interpreting the proposed transfer functions as **law abstractions** that characterize next state distributions.

3.4 Deductive Verification Approaches

Rather than keep track of the distributions along possible executions of the system, deductive verification approaches derive facts about the executions of the program as a whole without explicitly tracking or approximating the distribution over the program states. These facts can be in the form of polynomial functions over the program states that encode functional relations between the values of program variables at a given program point. Boolean predicates involving such functions can serve as program invariants, prove progress measures and prove global properties for the program.

Probabilistic Weakest Precondition. McIver and Morgan were among the first to consider deductive approaches for infinite state probabilistic programs with discrete distributions [121]. Their work provides a weakest precondition (Floyd-Hoare style) calculus that transforms **quantitative invariants** over the states. Quantitative invariants are invariants over the *expected* value of program expressions. At the heart of the calculus is the quantitative interpretation of the program semantics: a probabilistic program P transforms the expected values of program expressions $\mathtt{e}$ as it executes. This transformation can be captures as a weakest "pre-expectation" predicate transformer $wp(P, \mathtt{e})$ that for an expression $\mathtt{e}$ over the post states of the program produces an expression $\mathtt{e}'$ whose expected value over the initial states matches that of $\mathtt{e}$ over the post states. Therefore, answering a probabilistic query φ at the end of the program execution is equivalent to answering a query about the expected value of expression $\widehat{\varphi} = wp(P, \varphi)$ that is the pre-expectation of φ. Katoen et al. [105] provide a constraint-based framework for automatically inferring linear program expressions that are remain invariant under the pre-expectation operator: $\mathtt{e} = wp(P, \mathtt{e})$. This framework was later implemented as the tool PRINSYS [82] that relies on computer algebra systems (CAS) to perform quantifier elimination during the process of generating candidate expressions. We discuss this tool in more detail when we compare the results of our implementation in Chapters 5 and 7.

Unfortunately, the works of McIver and Morgan and Katoen et al. focus on probabilistic programs in which the stochastic inputs are restricted to discrete distributions. We naturally lift

this restriction and consider a richer class of distributions including Gaussian, Poisson, Uniform or Exponential random variables. Our setup in Chapter 5 can use any distributions whose expectations (and some higher moments) exist, and are available. Furthermore, our technique synthesizes invariants that are polynomial expressions involving the program variables. In Section 4.2 we prove that the quantitative invariants in our polynomial stochastic transition systems correspond to the well-known concept of martingales and supermartingales from probability theory [156].

Finally, McIver and Morgan treat demonic non-deterministic as well as stochastic inputs. Any nondeterminism is resolved as the minimum over all nondeterministic choices of the pre-expectations. We use this idea in Section 4.5 to extend our deductive approach using martingales to polynomial stochastic transition systems with nondeterminism.

Martingales and the Concentration of Measure Phenomenon. The approach and semantics we adopt in this thesis are similar to those of McIver and Morgan [121] and Gretz et al. [84, 83, 100] (see Sections 4.1, 4.5) in that distributions are represented **implicitly** by only tracking information about properties of reachable distributions such as moment inequalities of program expressions relevant to the probabilistic assertions. We make this point precise in Chapters 4, 6, 7.

We extend these deductive approaches to richer classes of probabilistic programs by supporting continuous distributions. However, at the same time we syntactically restrict conditional statements in our probabilistic programs by requiring that guards are either only over program variables or random draws but not both (such guards can be expressed by first storing the random choices as program variables). Observe statements [81, 93, 146] can be encoded as blocking loops (see [93] for details). We adopt a simple probabilistic transition system model to reflect the control flow graph of such programs (standard algorithmic construction of this process can be found in Manna and Pnuelli [118] and Chatterjee et al. [35]).

Probabilistic assertions (queries) [148, 76, 146, 42] normally asked over terminating behaviors of programs (or over the **output distribution**) are simplified by decoupling termination reasoning and reasoning about the probabilistic bounds. This is done by trivially extending all runs of the probabilistic transition systems to infinite runs through *stuttering* (repeating infinitely the first

terminating state along an execution). This is equivalent to considering the *stopped* version of the corresponding stochastic process.

In the context of a probabilistic program P, when a program expression $\mathtt{e}$ is evaluated along the sample traces of the probabilistic program, $\mathtt{e}$ induces a stochastic process. Any program expression $\mathtt{e}$ that satisfies the pre-expectation constraint $\mathtt{e} \leq \mathtt{pre}\mathbb{E}(P, \mathtt{e})$, induces a stochastic process that is a supermartingale (we prove this result in Section 4.2). This establishes a formal link between the sample runs of a probabilistic program and the convergence properties of martingales.

Martingale theory in connection with concentration of measure inequalities has been employed before to establish performance guarantees in randomized algorithms [129, 60]. Bournez [24] presents a variations of Foster's Theorem [72] to prove almost sure termination of probabilistic programs by manually providing a suitable supermartingale expression.

In Chapter 4 we present an alternative proof that supermartingale expressions can be used to prove almost sure termination of probabilistic programs. We phrase this result in the form of a deductive proof rules with sufficient conditions on the supermartingale expression to prove general almost sure reachability properties for a target set T. These conditions are involve only the initial distribution of the program, the target set T of interest and the pre-expectation of the certificate expression; however, they are independent of any intermediate probability distribution $\mathcal{D}_n$. We then show how by massaging the constraints we can provide similar proof rules to establish qualitative repeated reachability properties such as persistence $\Diamond\Box T$ or recurrence $\Box\Diamond T$ for a target set of states T.

In Chapter 5 we provide the necessary steps to encode the conditions of these proof rules as linear and semidefinite programming constraints over the unknown coefficients of program expressions. Solving these constraints allows us to infer supermartingale certificates. In Chapter 6 we present how these supermartingale certificates allow us to leverage results from probability theory (known as concentration of measure inequalities for large deviations of supermartingales) to provide upper bounds on the probability of extremely unlikely events known as "rare events".

In Chapter 7 we show that by taking linear combinations of martingale and supermartingale

expressions we can derive new expectation invariants. This gives us the means to lift the notion of martingale and supermartingales expressions to sets of mutually inductive expectation invariants. A set $E = \{\mathtt{e}_1, \ldots, \mathtt{e}_k\}$ of program expressions $\mathtt{e}_1, \ldots, \mathtt{e}_k$ is called an inductive expectation invariant if, when considered as a set of facts, E is expressive enough to prove that the expected value $\mathbb{E}_{\mathcal{D}_n}(\mathtt{e}_n)$ of each individual expression does not change sign across any iteration of the probabilistic loop. This provides us with a basis for a novel abstract interpretation procedure to automatically infer new supermartingale certificates.

Recent Extensions. The work on reachability properties of Chapter 4 was published in Chakarov et al. [32]. It first illustrated how an almost sure termination argument for probabilistic programs can be mechanized by reducing it to convergence properties of *ranking* supermartingale program expressions. This idea was later extended by Ferrer et al. [67] to a sound and complete proof technique for a.s. termination of a class of *affine* probabilistic programs. The algorithmic steps and their complexity were subsequently analyzed by Chatterjee et al. [36] and further extended to the class of *polynomial* probabilistic programs and ranking supermartingales in Chatterjee [35].

The proof rules for repeated reachability of Section 4.4.1, 4.4.4 of *polynomial* stochastic systems were published in Chakarov et al. [34]. Simultaneously, a deductive framework to cover all of PCTL* was presented by Dimitrova et al. [58]. However, assertions and inductive invariant annotations for proof rules in their framework need to be supplied manually.

The problem of generating polynomial expectation invariants was first done by Chen et al. [37] via simple Lagrange interpolation. We presented the use of Putinar's representation result (Chapter 5) in conjunction with semidefinite programming in Chakarov et al. [34]. The use of alternative representations for polynomial supermartingale generation was considered in [35].

Finally, quantitative reachability and termination the Bernstein's inequality was considered by Chatterjee et al. [36]. Kaminski et al. [102] consider the complexity and provide a predicate transformer calculus [103] to infer the expected termination time of a class of probabilistic programs. More general quantitative properties via concentration of measure results were considered by Bouissou et al. [23].

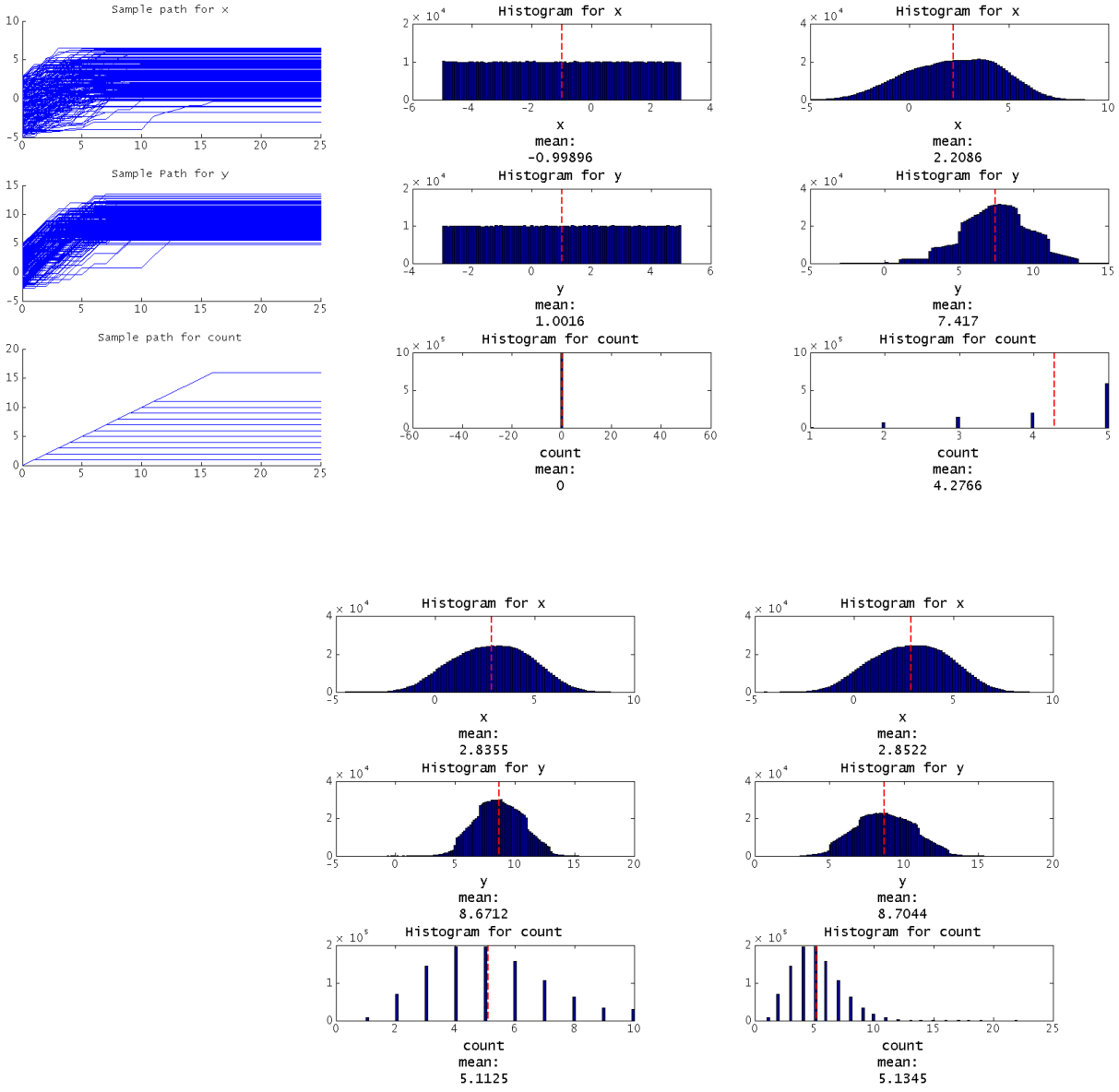


Figure 3.1: In top left, a number of sample executions of the motivating probabilistic program. Then from top to bottom, left to right, frequency histograms for the distributions $\mathcal{D}_n$ of values for the program variables at the loop head after $n = 0, 5, 10, 25$ iterations of the loop.

Chapter 4

Qualitative Program Analysis with Martingales: Almost Sure Termination, Persistence and Recurrence.

In this chapter we provide proof rules for almost sure reachability (termination), persistence, and recurrence properties using results from supermartingale theory.

We begin the chapter by presenting an extension of our discrete time stochastic systems to location based **polynomial stochastic transition systems**. In Section 4.2 we present **supermartingale expressions** that relate the properties of the system to convergence properties of martingales. We then state convergence results that establish the qualitative properties of a system in the form of proof rules and prove their soundness. Finally, we discuss some of the related work on qualitative properties and discuss some recent extensions of the work presented here.

Motivating Example. Consider the probabilistic program in Figure 4.1 that initializes the program variables $(\mathtt{x}, \mathtt{y})$ to a random integer drawn from a joint uniform distribution $\mathcal{D}_0$ over the set of support $[-5, 3] \times [-3, 5]$ (lines 1-2). We have also instrumented the program with an explicit loop counter variable `count`. On every iteration of the loop the program checks that the loop guard holds and independent of any previous random draws "flips" a biased coin (line 5) and with probability $3/4$ chooses to increment $\mathtt{x}, \mathtt{y}$ (lines 5-6); with complementary probability $1 - (3/4) = 1/4$ it leaves the values of the program variables unchanged.

An important question about the program is: **Does every execution of the program terminate (i.e., reach the termination location in line 11)?** The answer is **no**. For any $n \in \mathbb{N}$, the probability of an execution not terminating after n loop iterations (i.e., $\mathtt{count} = n$)

```
1  int x = unifRand(-5,3);
2  int y = unifRand(-3,5);
3  int count = 0;
4  while (x + y <= 10) {
5     if flip(3/4){ //r_flip
6        x = x + unifRand(0,2);//r1
7        y = y + 2;
8     }
9     count++;
10 }
11 //loop termination
```

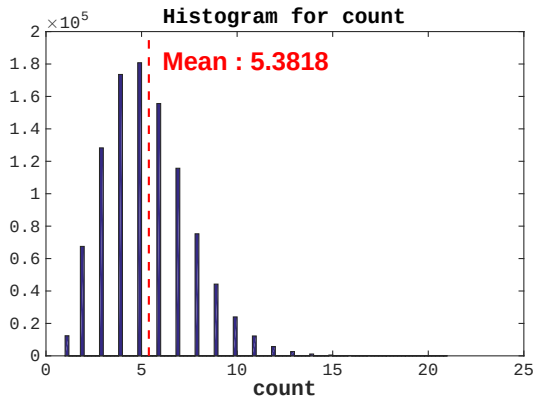


Figure 4.1: (LEFT) An example of a simple probabilistic program; (RIGHT) Histogram of the value of `count` for 10^6 simulations of the program.

is strictly greater than $\left(\frac{1}{4}\right)^n > 0$. Furthermore, the only non-terminating execution has to always take the "else" branch of the conditional flip in order to never make progress towards termination condition $\mathtt{x} + \mathtt{y} > 10$. Such an execution violates the semantics of the probabilistic choice in line 5 and in fact has probability $\lim_{n\to\infty}\left(\frac{1}{4}\right)^n = 0$.

To account for the probabilistic nature of the program and its properties, we reformulate the problem of termination to almost sure (a.s.) termination: **Does the set of executions that reach a termination location have probability 1?** The answer for the program in Figure 4.1 is **yes**. Using the theory of supermartingale convergence that is the subject of this chapter we can provide a formal proof that with probability 1 every run of the program terminates. Similar to the analysis of non-probabilistic programs, the key step of the proof is identifying the program expression $10 - \mathtt{x} - \mathtt{y}$ is a **supermartingale ranking function**.

4.1 Probabilistic Transition Systems

In this chapter, we focus on **polynomial** stochastic systems, which are instances of DTSS with piecewise polynomial update functions.

Let $X = \{x_1, \ldots, x_n\}$ denote a set of system (program) variables and let $\mathbf{x}$ denote a vector representation of their valuation. Let $R = \{r_1, \ldots, r_m\}$ denote a set of random (stochastic) variables with corresponding vector $\mathbf{r}$.

Definition 4.1.1. *A **polynomial stochastic transition system** Π is a tuple $\langle X, \mathcal{R}, L, \mathcal{T}, \ell_0, \mathcal{D}_0 \rangle$ with the following components:*

(1) a state space $X \subseteq \mathbb{R}^n$ is a semi-algebraic set (i.e., X is the solution of finitely many polynomial inequalities) with an associated σ-algebra $\mathcal{X} \subseteq \mathcal{B}(\mathbb{R}^n)$ on it,

(2) a probability space $\mathcal{R} : \langle R, \mathcal{F}_R, P_R \rangle$ for the stochastic inputs written collectively as $\mathbf{r} : (\mathbf{r}_c, \mathbf{r}_b)$, where $\mathbf{r}_c$ denotes the (possibly multivariate) continuous random variable and $\mathbf{r}_b$ denotes the (discrete) random variables that take on finitely many values,

*(3) a finite set of locations $L = \{\ell_0, \ldots, \ell_F\}$ with distinguished **initial** ℓ_0 and **final** location ℓ_F,*

(4) an initial probability distribution $\mathcal{D}_0$ over $L \times X$ with from which an initial state $(\ell_0, \mathbf{x}_0)$ is drawn,

(5) a finite set of transitions $\mathcal{T} = \{\tau_1, \ldots, \tau_p\}$. Each transition $\tau \in \mathcal{T}$ is a tuple $\langle \ell, \varphi_\tau, f_\tau \rangle$ that consists of:

*(a) A **source location** $\ell_\tau \in L$,*

*(b) A **guard** predicate $\varphi_\tau(\mathbf{x})$ that is a conjunction of polynomial inequalities over $\mathbf{x}$,*

*(c) An **update function** $f_\tau(\mathbf{x}, \mathbf{r}) : X \times R \to L \times X$ that is a piecewise polynomial function of the form:*

$$f_\tau(\mathbf{x}, \mathbf{r}) : \begin{cases} (m_1, g_{\tau,1}(\mathbf{x}, \mathbf{r}_c)), & \text{if } \psi_{\tau,1}(\mathbf{r}_b) \\ \vdots & \\ (m_j, g_{\tau,j}(\mathbf{x}, \mathbf{r}_c)), & \text{if } \psi_{\tau,j}(\mathbf{r}_b)\,, \end{cases}$$

*where $m_1, \ldots, m_j \in L$ are **target locations**, $g_{\tau,1}, \ldots, g_{\tau,j}$ are **fork update** (multivariate polynomials over $\mathbf{x}, \mathbf{r}_c$), and $\psi_{\tau,1}(\mathbf{r}_b), \ldots, \psi_{\tau,j}(\mathbf{r}_b)$ represent mutually exclusive*

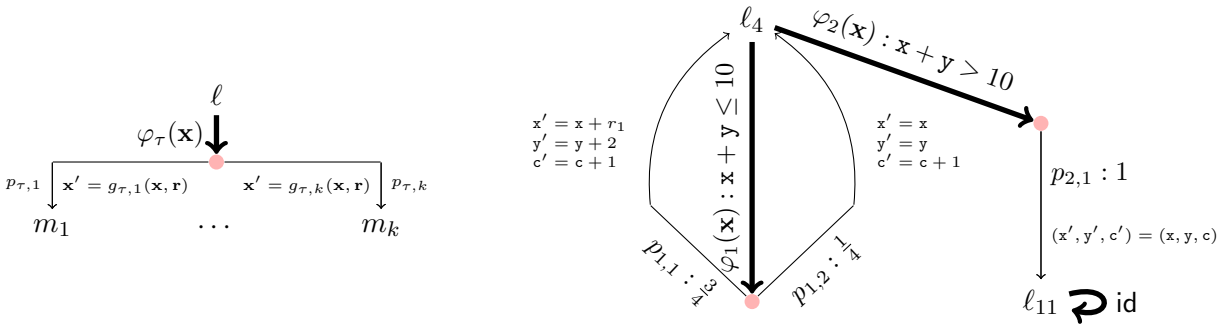


Figure 4.2: (LEFT) Shows the structure of a generic PTS transition with $k \geq 1$ forks. Each fork has a probability $p_{\tau,j}$, fork update $g_{\tau,j}$ and target location m_j. (RIGHT) Shows the PSTS for program in Figure 4.1 with two transitions and a self-loop id (`count` is abbreviated `c`).

and exhaustive predicates over the discrete random variables $\mathbf{r}_b$*. We refer to each tuple* $(m_i, g_{\tau,i}(\mathbf{x}, \mathbf{r}_c))$ *as a* ***fork*** *that is* ***guarded*** *by* $\psi_{\tau,i}$ *with corresponding* ***fork probability*** $p_{\tau,i} : \text{Prob}(\psi_{\tau,i}(\mathbf{r}_b))$.

Example 4.1.1. *Consider the probabilistic program of Figure 4.1. The corresponding PSTS is as follows:*

- $X = \{\mathtt{x}, \mathtt{y}, \mathtt{count}\}$,
- $R = \{r_1, r_{flip}\}$ *where* $\mathbf{r}_c : r_1 \sim Uniform(0, 2)$, *and* $\mathbf{r}_b : r_{flip} \sim Bernoulli(3/4)$. *We assume the values of* r_1 *and* r_{flip} *are drawn independently and let* $\mathcal{D}_R$ *denote their joint probability distribution.*
- $L = \{\ell_4, \ell_{11}\}$ *where* ℓ_4 *is the initial location and* ℓ_{11} *is the final (terminating) location that corresponds to line 11.*
- $\mathcal{T} = \{\tau_1, \tau_2, \mathit{id}\}$ *where*

$$\tau_1 : \langle \ell_4, \varphi_1(\mathbf{x}) : \mathtt{x} + \mathtt{y} \leq 10, f_{\tau_1} \rangle$$

with

$$f_{\tau_1}(\mathbf{x}, \mathbf{r}) : \begin{cases} (\ell_4, g_{\tau,1}(\mathbf{x}, \mathbf{r}_c)), & \text{if } \psi_{\tau,1}(\mathbf{r}_b) : r_{flip} = 1 \\ (\ell_4, g_{\tau,2}(\mathbf{x}, \mathbf{r}_c)), & \text{if } \psi_{\tau,2}(\mathbf{r}_b) : r_{flip} = 0 \end{cases}$$

with fork updates $g_{\tau_1,1}(\mathbf{x}, \mathbf{r}_c)$, $g_{\tau_1,2}(\mathbf{x}, \mathbf{r}_c)$ *as depicted;*

$$\tau_2 : \langle \ell_4, \varphi_2(\mathbf{x}) : \mathrm{x} + \mathrm{y} > 10, f_{\tau_2} \rangle$$

with

$$f_{\tau_2}(\mathbf{x}, \mathbf{r}) : (\ell_{11}, id), \qquad \textit{if} \;\; \psi_{\tau,1}(\mathbf{r}_b) : \textit{true};$$

and a special self-loop transition id *with* $\varphi_{\mathsf{id}} : \textit{true}$ *for guard and the identity function* $id(\mathbf{x}, \mathbf{r}) = \mathbf{x}$ *for update function.*

For a polynomial stochastic transition system to represent a stochastic system over X as in Definition 2.2.3, we require that the transitions together form a function over the state-space X:

(1) The guards are pairwise **mutually exclusive**: $\varphi_{\tau_i} \wedge \varphi_{\tau_j}$ is unsatisfiable for all $i \neq j$.

(2) The guards are **mutually exhaustive**: $\bigvee_{j=1}^{k} \varphi_{\tau_j} \equiv \textit{true}$.[1]

Mutual exhaustiveness is not strictly necessary as long as the disjunction of the transition guards covers the set of reachable program (system) states.

Conditions (1) and (2) together imply that the set of transition guards forms a partition of the state-space X; therefore, system update function $\mathcal{F}_{\mathcal{T}} \triangleq \varphi_{\tau_i} \mapsto f_{\tau_i}$ is a piecewise polynomial transition function that casts any polynomial transition system into a discrete time stochastic system.

Execution. A **state** of a polynomial stochastic transition system is a tuple $\langle \ell, \mathbf{x} \rangle$ of a location $\ell \in L$ and a valuation $\mathbf{x}$ of the system variables X. Before taking a transition $\tau : \langle \ell, \varphi_\tau, f_\tau \rangle$ from a state $s : \langle \ell, \mathbf{x} \rangle$ a check is performed to determine whether τ is **enabled**, i.e., $\mathbf{x} \models \varphi$. If so, the result of executing τ from s is a probability distribution over the post states obtained by carrying out the following steps:

(1) Draw a vector of random variables $\mathbf{r} : \langle r_1, \ldots, r_m \rangle$ from the joint distribution $\mathcal{D}_R$.

(2) Choose a fork $\langle m_j, g_{\tau,j} \rangle$ for $j \in [1, k]$ based on the $\psi_j(\mathbf{r}_b)$.

[1] For simplicity, we assume fork guards $\psi_{\tau,i}$, $i \in [1, j]$ satisfy the same constraints.

(3) Update the state of the system to $s' : \langle m_j, \mathbf{x}' = g_{\tau,j}(\mathbf{x}, \mathbf{r}_c)\rangle$. We call $m_j \in L$ the post-location and $\mathbf{x}'$ the post-state of the system variables.

Let $\textsc{PostDistrib}(s, \tau)$ denote the distribution over the post-states $\langle \ell', \mathbf{x}'\rangle$ that results from taking the enabled transition τ from the state s. Because $\mathcal{F}_{\mathcal{T}}$ is a function, exactly one transition is enabled from each state s of the polynomial stochastic transition system. Therefore, we abbreviate the unique post-distribution as $\textsc{PostDistrib}(s)$. For the full details please refer to the **operational semantics** defined in Section 2.2.2 (for every $n \geq 0$, if $s \sim \mathcal{D}_n$, then $\textsc{PostDistrib}(s)$ is the consistent product extension using Kolmogorov's Extension Theorem).

Definition 4.1.2. *Let Π be a polynomial stochastic transition system. A **sample execution** (a **run**, or a **trace**) of the system is the countably infinite sequence of states:*

$$\rho : \langle \ell_0, \mathbf{x}_0\rangle \xrightarrow{\tau_1} \langle \ell_1, \mathbf{x}_1\rangle \xrightarrow{\tau_2} \cdots \xrightarrow{\tau_n} \langle \ell_n, \mathbf{x}_n\rangle \cdots$$

An alternative notation that emphasizes the stochastic variables driving the system is:

$$\rho(\omega) \triangleq \langle \ell_0, \mathbf{x}_0\rangle \xrightarrow{f_{\tau_1}(\mathbf{x}_0, \omega_1)} \langle \ell_1, \mathbf{x}_1\rangle \xrightarrow{f_{\tau_2}(\mathbf{x}_1, \omega_2)} \cdots \xrightarrow{f_{\tau_n}(\mathbf{x}_{n-1}, \omega_n)} \langle \ell_n, \mathbf{x}_n\rangle \cdots$$

such that:

- *$\langle \ell_0, \mathbf{x}_0\rangle$ is the initial state drawn from the initial distribution $\mathcal{D}_0$, i.e., $\langle \ell_0, \mathbf{x}_0\rangle \sim \mathcal{D}_0(\omega_0)$;*
- *for every $j \geq 0$, state $s_j : \langle \ell_j, \mathbf{x}_j\rangle$ satisfies transition guard $\varphi_{\tau_{j+1}}$, i.e., $\mathbf{x}_j \models \varphi_{\tau_{j+1}}$;*
- *each state $s_{j+1} : \langle \ell_{j+1}, \mathbf{x}_{j+1}\rangle$ is a sample from $\textsc{PostDistrib}(s_j)$, i.e., $\mathcal{D}_{j+1} = \textsc{PostDistrib}(s_j)$ and $s_{j+1} \sim \mathcal{D}_{j+1}(\omega_j)$.*

Definition 4.1.3. *Let Π be a polynomial stochastic transition system. Let*

$$\rho(\omega) : \langle \ell_0, \mathbf{x}_0\rangle \xrightarrow{f_{\tau_1}(\mathbf{x}_0, \omega_1)} \langle \ell_1, \mathbf{x}_1\rangle \xrightarrow{f_{\tau_2}(\mathbf{x}_1, \omega_2)} \cdots \xrightarrow{f_{\tau_n}(\mathbf{x}_{n-1}, \omega_n)} \langle \ell_n, \mathbf{x}_n\rangle \cdots$$

*be a sample execution, the corresponding **sample path** $\pi_\rho(\omega)$ is the projection of $\rho(\omega)$ onto L:*

$$\pi_\rho(\omega) \triangleq \ell_0 \xrightarrow{\tau_1} \ell_1 \xrightarrow{\tau_2} \cdots \xrightarrow{\tau_n} \ell_n \cdots$$

We say that a sample trace $\rho(\omega)$ of Π **terminates** if it eventually reaches a state $(\ell_F, \mathbf{x})$ where ℓ_F is the final (terminating) location of Π and $\mathbf{x}$ is any system state. Equivalently, a sample run $\rho(\omega)$ of Π is **terminating** if its corresponding sample path $\pi_\rho(\omega)$ eventually reaches ℓ_F. We call such a path **terminating**.

Example 4.1.2. *Continuing with the PSTS of Example 4.1.1, we present two sample traces:*

$$\rho_1(\omega) : \langle \ell_4, (\mathtt{x}:0, \mathtt{y}:0, \mathtt{count}:0)\rangle \xrightarrow{\tau_1} \langle \ell_4, (1,2,1)\rangle \xrightarrow{\tau_1} \langle \ell_4, (1,4,2)\rangle \xrightarrow{\tau_1} \langle \ell_4, (3,6,3)\rangle$$
$$\xrightarrow{\tau_1} \langle \ell_4, (3,6,4)\rangle \xrightarrow{\tau_1} \langle \ell_4, (3,8,5)\rangle \xrightarrow{\tau_2} \langle \ell_{11}, (3,8,5)\rangle \xrightarrow{\mathit{id}} \langle \ell_{11}, (3,8,5)\rangle \xrightarrow{\mathit{id}} \cdots$$
$$\rho_\perp(\omega) : \langle \ell_4, (\mathtt{x}:0, \mathtt{y}:0, \mathtt{count}:0)\rangle \xrightarrow{\tau_1} \langle \ell_4, (0,0,1)\rangle \xrightarrow{\tau_1} \langle \ell_4, (0,0,2)\rangle \xrightarrow{\tau_1} \cdots$$

where $\rho_1(\omega)$ represents a terminating run identified by the self-loop id, and $\rho_\perp(\omega)$ represents a non-terminating run that takes transition τ_1 forever. The corresponding sample paths are:

$$\pi_{\rho_1}(\omega) : \ell_4 \xrightarrow{\tau_1} \ell_4 \xrightarrow{\tau_1} \ell_4 \xrightarrow{\tau_1} \ell_4 \xrightarrow{\tau_1} \ell_4 \xrightarrow{\tau_1} \ell_4 \xrightarrow{\tau_2} \ell_{11} \xrightarrow{\mathit{id}} \ell_{11} \xrightarrow{\mathit{id}} \cdots$$
$$\pi_{\rho_\perp}(\omega) : \ell_4 \xrightarrow{\tau_1} \ell_4 \xrightarrow{\tau_1} \cdots$$

where $\pi_{\rho_1}(\omega)$ is terminating and $\pi_{\rho_\perp}(\omega)$ is non-terminating.

Finally, Figure 4.1 (RIGHT) *presents a histogram of 10^6 sample traces of the value of* `count` *that reaches line 11.*

4.2 Martingale and Supermartingale Expressions

This section provides the formal connection between polynomial stochastic transition systems and (super-) martingale stochastic processes and their convergence properties.

Let $\Pi : \langle X, \mathcal{R}, L, \mathcal{T}, \ell_0, \mathcal{D}_0\rangle$ be a polynomial system. A function $h[X]$ over the program variables of Π is called a (polynomial) **program expression** if $h : X \to \mathbb{R}$ is a (polynomial) function from system states to real numbers. A **flow-sensitive expression map** is a function V that maps each location $\ell \in L$ to a program expression $V_\ell[X]$.

4.2.1 Pre-Expectations

Inspired by the work of McIver and Morgan [121, 105] and related to the drift operator of Markov processes [122] we define the notion of a pre-expectation operator $\texttt{pre}\mathbb{E}$. Operator $\texttt{pre}\mathbb{E}$ transforms the program expressions of a PSTS over the post-state variables $\texttt{e}(\mathbf{x}')$ into expressions of equal expected value over the current states obtained after one step of execution.

Definition 4.2.1 (Pre-Expectation across a Transition)**.** *Let $\Pi : \langle X, \mathcal{R}, L, \mathcal{T}, \ell_0, \mathcal{D}_0 \rangle$ be a polynomial stochastic transition system. Let $\texttt{e}[X]$ be a program expression and V be a flow-sensitive expression map over the variables of Π. Let $\tau \in \mathcal{T}$ be an enabled transition (i.e., $\tau : \langle \ell_\tau, \varphi_\tau, f_\tau \rangle$ and $\mathbf{x} \models \varphi$) with forks $(m_1, g_{\tau,1}), \ldots, (m_j, g_{\tau,j})$ and corresponding fork probabilities $p_{\tau,1}, \ldots, p_{\tau,j}$. The **pre-expectation of** $\texttt{e}[X]$ **across transition** τ is a function $\texttt{pre}\mathbb{E}(\texttt{e}, \tau) : X \to \mathbb{R}$ defined as follows:*

$$\forall \mathbf{x} \in X, \quad \texttt{pre}\mathbb{E}(\texttt{e}, \tau)(\mathbf{x}) \triangleq \mathbb{E}_R[\texttt{e}(f_\tau(\mathbf{x}, \mathbf{r}))|\mathbf{x}] = \sum_{i=1}^{j} p_{\tau,i} \cdot \mathbb{E}_{R_c}[\texttt{e}\left(g_{\tau,i}(\mathbf{x}, \mathbf{r}_c)\right)],$$

*where $\mathbb{E}_R[\cdot|\mathbf{x}]$ denotes the conditional expectation over the random choices $\mathbf{r} = (\mathbf{r}_b, \mathbf{r}_c)$ given the system state $\mathbf{x}$, and $\mathbb{E}_{R_c}[h(\mathbf{r}_c)]$ is the expected value of h with respect to $\mathbf{r}_c$. The **pre-expectation of** V **across transition** τ is the program expression:*

$$\texttt{pre}\mathbb{E}(V, \tau) \triangleq \sum_{i=1}^{j} p_{\tau,i} \cdot \mathbb{E}_R[V_{m_j}\left(g_{\tau,i}(\mathbf{x}, \mathbf{r}_c)\right)].$$

Intuition. Notice this definition is equivalent to defining the pre-expectation of $\texttt{e}[X]$ across τ as the weighted average of $\texttt{e}$ over all pre-states $\langle \ell_\tau, \mathbf{x} \rangle$, which under the update of a fork $(m_i, g_i(\mathbf{x}, \mathbf{r}_c))$ result in the post-states $\langle \ell', \mathbf{x}' \rangle = \langle m_i, g_i(\mathbf{x}, \mathbf{r}_c) \rangle$, multiplied by the probability $p_{\tau,i}$ of selecting the fork. Similarly, the pre-expectation of V across τ is the weighted average of the probability $p_{\tau,i}$ of selecting a fork $(m_i, g_{\tau,i})$ multiplied by the expected value of the target location expression V_{m_i} when the system variables are updated according to fork update $g_{\tau,i}(\mathbf{x}, \mathbf{r})$.

Given the piecewise polynomial definition of the system update function $\mathcal{F}_\mathcal{T}$ above, we can define the pre-expectation transformation over the entire stochastic system Π.

Definition 4.2.2. *The **pre-expectation** of a program expression* $\mathtt{e}[X] : X \to \mathbb{R}$ *with respect to a polynomial stochastic transition system* $\Pi : \langle X, \mathcal{R}, L, \{\tau_1, ..., \tau_n\}, \ell_0, \mathcal{D}_0 \rangle$ *is a function* $\mathtt{pre}\mathbb{E}(\mathtt{e}, \Pi) : X \to \mathbb{R}$ *defined by*

$$\mathtt{pre}\mathbb{E}(\mathtt{e}, \Pi)(\mathbf{x}) \triangleq \mathbb{E}_R\left[\mathtt{e}(\mathcal{F}_{\mathcal{T}}(\mathbf{x}, \mathbf{r}))|\mathbf{x}\right] = \sum_{i=1}^{n} \mathbb{1}(\varphi_i(\mathbf{x})) \cdot \mathtt{pre}\mathbb{E}(\mathtt{e}, \tau_i)(\mathbf{x}), \quad \textit{for all } \mathbf{x} \in X,$$

where φ_i *is the guard of transition* τ_i*, and* $\mathbb{1}(\varphi)$ *is the indicator function of predicate* φ*.*

Next, we define the notion of martingale and supermartingale (program) expressions for a polynomial stochastic transition system.

Definition 4.2.3. *Let* $\Pi : \langle X, \mathcal{R}, L, \mathcal{T}, \ell_0, \mathcal{D}_0 \rangle$ *be a polynomial stochastic transition system. Let* $\mathtt{e}[X]$ *be a program expression over the program variables of* Π*. The expression* $\mathtt{e}[X]$ *is a **supermartingale expression** for* Π *if and only if*

$$\forall \mathbf{x} \in X, \quad \mathtt{pre}\mathbb{E}(\mathtt{e}, \Pi)(\mathbf{x}) \leq e(\mathbf{x}) \quad (i.e., \mathbb{E}_R\left[\mathtt{e}(\mathcal{F}_{\mathcal{T}}(\mathbf{x}, \mathbf{r}))|\mathbf{x}\right] \leq \mathbf{x}).$$

Expression $\mathtt{e}[X]$ *is a **martingale expression** for* Π *if and only if*

$$\forall \mathbf{x} \in X, \quad \mathtt{pre}\mathbb{E}(\mathtt{e}, \Pi)(\mathbf{x}) = e(\mathbf{x}) \quad (i.e., \mathbb{E}_R\left[\mathtt{e}(\mathcal{F}_{\mathcal{T}}(\mathbf{x}, \mathbf{r}))|\mathbf{x}\right] = \mathbf{x}).$$

Expression $\mathtt{e}[X]$ *is an* ε***-additive*** *supermartingale expression for some constant* $\varepsilon > 0$ *iff*

$$\exists \forall \mathbf{x} \in X, \quad \mathtt{pre}\mathbb{E}(\mathtt{e}, \Pi)(\mathbf{x}) \leq \mathtt{e}(\mathbf{x}) - \varepsilon.$$

Expression $\mathtt{e}[X]$ *is an* α***-multiplicative*** *supermartingale expression for some constant* $\alpha \in (0, 1)$ *if and only if*

$$\exists \forall \mathbf{x} \in X, \quad \mathtt{pre}\mathbb{E}(\mathtt{e}, \Pi)(\mathbf{x}) \leq \alpha \mathtt{e}(\mathbf{x}).$$

The following lemma holds respectively for any supermartingale, martingale, ε-additive and α-multiplicative supermartingale expression $\mathtt{e}[X]$. It shows that these supermartingale expressions define corresponding supermartingale processes over the sample traces of the PSTS.

Lemma 4.2.1. A martingale expression $\mathtt{e}[X]$ defines a martingale stochastic process $\{\mathtt{e}_i \triangleq \mathtt{e}(\mathbf{x}_i)\}_{i=0}^{\infty}$.

Proof. Immediate from Definition 4.2.3. □

Example 4.2.1. *Consider our motivating example PSTS of Example 4.1.1 and the program expression* $\mathtt{e}_1 : \mathtt{10} - \mathtt{x} - \mathtt{y}$ *and suppose* $\varphi_{\tau_1}(\mathbf{x})$ *holds, then:*

$$\begin{aligned}
\mathtt{pre}\mathbb{E}(\mathtt{e}_1, \Pi) = \mathtt{pre}\mathbb{E}(\mathtt{e}_1, \tau_1) &= \mathbb{E}_R[\mathtt{10} - \mathtt{x}' - \mathtt{y}' \mid \mathbf{x} = (x, y, count)] \\
&= \left(\frac{3}{4}\right)(10 - \mathbb{E}_R(x + r_1) - (y + 2)) + \left(\frac{1}{4}\right)(10 - x - y) \\
&= \left(\frac{3}{4}\right)(10 - (x + 1) - (y + 2)) + \left(\frac{1}{4}\right)(10 - x - y) \\
&= (10 - x - y) - \frac{9}{4} \leq 10 - x - y = \mathtt{e}_1(\mathbf{x});
\end{aligned}$$

therefore, $\mathtt{e}_1$ *is a supermartingale expression under* τ_1 *(also under* τ_2 *and, trivially,* id*) but an* ε*-additive supermartingale expression (*$\varepsilon = \frac{9}{4}$*) whenever* $\varphi_\tau(\mathbf{x})$ *holds.*

Consider program expression $\mathtt{e}_2 : \mathtt{4x} + \mathtt{4y} - \mathtt{9count}$*:*

$$\begin{aligned}
\mathtt{pre}\mathbb{E}(\mathtt{e}_2, \Pi) = \mathtt{pre}\mathbb{E}(\mathtt{e}_2, \tau_1) &= \mathbb{E}_R[\mathtt{4x}' + \mathtt{4y}' - \mathtt{9count}' \mid \mathbf{x} = (x, y, count)] \\
&= \left(\frac{3}{4}\right)(4(x + 1) + 4(y + 2) - 9(count + 1)) + \left(\frac{1}{4}\right)(4x + 4y - 9(count + 1)) \\
&= (4x + 4x - 9count) = \mathtt{e}_2(\mathbf{x}).
\end{aligned}$$

This means $\mathtt{e}_2$ *is a martingale expression.*

Often, it is not possible to obtain a single expression that is a martingale, (ε-additive, or α-multiplicative) supermartingale expression for the whole PSTS. However, we can extend the notion of martingale expressions to flow-sensitive expression maps that provide the flexibility to label individual locations with different program expressions.

Definition 4.2.4. *The* ***pre-expectation*** *of a flow-sensitive expression map* V *w.r.t a polynomial stochastic transition system* $\Pi : \langle X, \mathcal{R}, L, \{\tau_1, \ldots, \tau_n\}, \ell_0, \mathcal{D}_0 \rangle$ *is:*

$$\mathtt{pre}\mathbb{E}(V, \Pi)(\mathbf{x}) \triangleq \sum_{i=1}^{n} \mathbb{1}(\varphi_i(\mathbf{x})) \cdot \mathtt{pre}\mathbb{E}(V, \tau_i)(\mathbf{x}), \quad \textit{for all } \mathbf{x} \in X.$$

Definition 4.2.5. *Let* $\Pi : \langle X, \mathcal{R}, L, \mathcal{T}, \ell_0, \mathcal{D}_0 \rangle$ *be a PSTS. A flow-sensitive expression map* V *of* Π *is a* ***supermartingale expression map*** *if and only if for every transition* $\tau : \langle \ell, \varphi_\tau, f_\tau \rangle$ *:*

$$\forall \mathbf{x} \in [\![\varphi_\tau(\mathbf{x})]\!], \quad \texttt{pre}\mathbb{E}(V, \tau)(\mathbf{x}) \leq V_\ell(\mathbf{x}),$$

that is, for any state $\langle \ell, \mathbf{x} \rangle$ *in which* τ *is enabled the pre-expectation of* V *across* τ *is no more than the value of the program expression* V_ℓ *in that (source) state.*

In the next two sections we present the qualitative reachability, persistence and recurrence properties of interest and how to construct the required supermartingale expression maps. We also prove that supermartingale expression maps V induce supermartingale processes $\{V(\rho(\omega)\}_{n=0}^{\infty}$ when evaluated along the runs $\rho(\omega)$ of a PSTS.

Drift Operator. Related to the pre-expectation operator is the notion of a drift operator.

Definition 4.2.6 (Drift Operator)**.** *Let* Π *be a polynomial stochastic transition system and* h *be a function over the state of* Π*. The* ***drift*** *of* h *w.r.t.* Π *is the function* $\mathcal{D}_\Pi h :\ \texttt{pre}\mathbb{E}(h, \Pi) - h$*.*

Wherever the system Π is clear from the context, we use $\mathcal{D}h$ to denote the drift operator $\mathcal{D}_\Pi$ applied to the function h.

4.3 Almost Sure Reachability $\Diamond T$

In this section, we provide two sets of sufficient conditions to prove that the executions of a PSTS Π almost surely reach a set of target states $T = \{\langle \ell_T, \mathbf{x} \rangle \in L \times X \mid \ell_T \in L_T,\ \mathbf{x} \models \varphi_T\}$ for some $L_T \subseteq L$ and some predicate φ_T over the system states. The approach is to identify supermartingale expression maps as certificates that act as ranking supermartingales along the sample executions of the polynomial stochastic transition system. We then use the ranking supermartingale convergence properties to prove that the PSTS converges sample-wise almost surely to the target set T. The conditions in these proof rules could be seen as constraints on the certificate functions to guarantee the desired property.

Almost Sure Termination. Let $\Pi : \langle X, \mathcal{R}, L, \mathcal{T}, \ell_0, \mathcal{D}_0 \rangle$ be a PSTS with final location ℓ_F. Let $L_T = \{\ell_F\}$ be the target location and $\varphi_T(\mathbf{x}) : \mathit{true}$ be the target predicate. We refer to Π as almost surely terminating if and only if the qualitative property $[\![\Diamond T]\!]$ holds.

Definition 4.3.1 (Ranking Super Martingale)**.** *A non-negative supermartingale $\mathcal{M} : \{M_n\}_{n=0}^{\infty}$ is called **ranking** if and only if it has one of the following properties:*

(1) $\mathcal{M}$ is an c-additive supermartingale: by Definition 2.2.2,

$$\exists\, \varepsilon > 0, \quad \mathbb{E}(M_{n+1}(\omega) \,|\, \mathcal{F}_n) \leq M_n(\omega) - \varepsilon, \qquad \mathit{or}$$

(2) $\mathcal{M}$ is an α-multiplicative supermartingale: by Definition 2.2.2,

$$\exists\, \alpha \in (0,1), \quad \mathbb{E}(M_{n+1}(\omega) \,|\, \mathcal{F}_n) \leq \alpha M_n(\omega).$$

It is trivial to convert a lower-bounded c-additive supermartingale $\mathcal{M}$ (i.e., for all $n \geq 0$, $M_n \geq K$ for some $K < 0$) to a ranking supermartingale by adding a constant $-K$ term to every random variable M_n.

Proof Rules of Reachability. We provide proof rules for establishing almost sure reachability properties that rely on Lyapunov-like certificate functions. One set of proof rules employs α-multiplicative supermartingale certificates and we call this geometric (REACH-GEOM); whereas the other employs ε-additive supermartingale certificates and we call this additive (REACH-ADD).

Let $S \triangleq L \times X$ be the set of states and let $s \in S$ denote a state (i.e., $s : \langle \ell, \mathbf{x} \rangle$ for some $\ell, \mathbf{x}$).

REACH-GEOM**: Geometric rule for reachability**

$(p1)$	$(\forall\, s \in S)\ V(s) \geq 0,$	Positive semidefinite V.
$(p2)$	$(\exists\, \varepsilon > 0)\ (\forall\, s \in S \setminus T)\ V(s) \geq \varepsilon,$	Lower bound outside T.
$(p3)$	$(\exists\, \alpha \in (0,1))\ (\forall\, s \in S \setminus T)\ \mathcal{D}V(s) \leq (\alpha - 1)V(s),$	Drift condition outside T.
	$\Diamond T$ almost surely.	

REACH-ADD: **Additive rule for reachability**

$(p1)$	$(\forall\ s \in S)\ V(s) \geq 0,$	Positive semidefinite V.
$(p2)$	$(\exists\ \varepsilon > 0)\ (\forall\ s \in S \setminus T)\ V(s) \geq \varepsilon,$	Lower bound outside T.
$(p4)$	$(\exists\ c > 0)\ (\forall\ s \in S \setminus T)\ \mathcal{D}V(s) \leq -c,$	Drift condition outside T.
	$\Diamond T$ almost surely.	

Both REACH-GEOM and REACH-ADD state that a polynomial stochastic system Π satisfies $\Diamond T$ almost surely if there exists a nonnegative certificate function V (condition $(p1)$) whose value outside T is lower bounded by some $\varepsilon > 0$ (condition $(p2)$). Moreover, the drift conditions ensure that in expected value V in the next decreases by some fixed non-zero quantity outside T (an additive constant in $(p4)$, or, a multiplicative factor in $(p3)$). These conditions together guarantee that V is a supermartingale whose drift condition outside T forces its value to decrease along almost all sample paths and eventually reach a value ε at which point the sample path "enters" T.

Soundness. We establish the soundness of the proof rules in a series of theorems. For convenience we recall Theorem 4.3.1.

Theorem 4.3.1. Let $\mathcal{M} = \{M_i\}_{i=0}^{\infty}$ be nonnegative α-multiplicative supermartingale for some $\alpha \in (0, 1)$. Then $\mathcal{M}$ converges almost surely (samplewise) to 0.

This result can be applied directly to stochastic transition systems. Let $m(\ell, \mathbf{x})$ be a nonnegative α-multiplicative supermartingale expression (map) for a transition system Π. For a sample trace $\rho(\omega) : \{\langle \ell_i, \mathbf{x}_i \rangle\}_{i=0}^{\infty}$ of Π, we say that the stochastic process $m(\ell_i, \mathbf{x}_i)$ converges almost surely to 0 along all sample traces. Therefore, it almost surely crosses the level set $m(\ell_i, \mathbf{x}_i) = \varepsilon$ of (p2).

Lemma 4.3.1. Let $m(\ell, \mathbf{x})$ be a nonnegative α-multiplicative supermartingale expression map for a polynomial stochastic transition system Π. Then $m(\ell, \mathbf{x})$ converges almost surely to 0, i.e.,

$$\mathrm{P}\big(\{\omega \in \Omega \,\big|\, \exists i \in \mathbb{N},\ \rho(\omega_i) = \langle \ell_i, \mathbf{x}_i \rangle \in T\}\big) = 1, \quad \text{for any } \varepsilon > 0.$$

Proof. The result follows directly from the almost sure convergence of $m(\ell, \mathbf{x})$ to zero on sample paths of Π and the choice of ε. □

Definition 4.3.2 (Multiplicative Super Martingale Ranking Function)**.** *A multiplicative supermartingale ranking function (SMRF) V is a supermartingale expression map that satisfies the following:*

- *for all $\ell \notin L_T$, for all $\mathbf{x} \in X$, $\quad V(\ell)[\mathbf{x}] \geq \varepsilon$;*
- *for all $\ell_T \in L_T$, for all $\mathbf{x} \in \varphi_T$, $\quad V(\ell_T)[\mathbf{x}] \in [0, \varepsilon)$;*
- *there exists $\alpha \in (0,1)$ s.t. for each transition $\tau : \langle \ell, \varphi, f_\tau \rangle$ with $\ell \notin L_T$ or $\varphi \not\subseteq \varphi_T$,*

$$(\forall \mathbf{x} \in X)\, \varphi(\mathbf{x}) \implies \mathtt{pre}\mathbb{E}(V, \tau) \leq \alpha V(\ell)[\mathbf{x}].$$

In order to simplify notation, from here on we write $V(\ell, \mathbf{x})$ to mean $V(\ell)[\mathbf{x}]$.

Theorem 4.3.2 (Soundness of REACH-GEOM)**.** A polynomial stochastic system Π satisfies the almost sure reachability property $\Diamond T$ if it has a multiplicative supermartingale ranking function V.

Proof. Follows directly from Definition 4.3.2 and Lemma 4.3.1 for the stochastic process: $\{V(\rho_n(\omega))\}_{n=0}^{\infty}$ where $\rho_n(\omega) = \langle \ell_n, \mathbf{x}_n \rangle \in S$. □

Example 4.3.1. *Consider a stochastic system Π with a single system variable x over $\mathbb{R}$, a single location ℓ, and a single self-looping transition with update: $x' := 0.1(1+w)x$, where w is a standard Gaussian random variable. We show that $\Diamond(T : |x| \leq 0.1)$ holds almost surely.*

Consider the function $V(\ell, x) = x^2$, which is nonnegative on X (condition $(p1)$) and $V(\ell, x) \geq 0.01$, for all $x \in X \setminus T$ (condition $(p2)$). Moreover, for all $x \in X$,

$$\mathtt{pre}\mathbb{E}(V, \Pi) = \mathbb{E}_w(V(\ell, x_{n+1}) | x_n) = 0.02 x_n^2, \text{ so } \mathcal{D}V(\ell, x) \leq -0.98 V(\ell, x).$$

Hence, $V(\ell, x)$ defines a 0.02-multiplicative supermartingale expression (condition $(p3)$).

Applying REACH-GEOM*, we conclude that $\Diamond(-0.1 \leq x \leq 0.1)$ holds a.s.*

Note 1. Note that the multiplicative certificate $V(\ell, x)$ in the example above works with REACH-ADD because the bound ε defines a minimum decrease constant $c = -0.98 \cdot 0.01 = -0.0098$ in condition (p4) when the system is outside the region T.

Lemma 4.3.2. Every REACH-GEOM certificate V is also a certificate for REACH-ADD and if X is a compact set then the converse holds.

Proof. Correspondence of conditions $(p1), (p2)$ is immediate.

($\Rightarrow$): We need to show that if V satisfies $(p3)$ then V satisfies $(p4)$ for some $c > 0$.

Fix $\alpha \in (0, 1)$ and let $c = (1 - \alpha)\varepsilon$. Then $c > 0$ and so:

$$\begin{aligned}
\texttt{pre}\mathbb{E}(V, \Pi) &\leq \alpha V \leq V - (1 - \alpha)V \\
&\leq V - (1 - \alpha)\varepsilon, \qquad \text{since for all } \langle \ell, \mathbf{x} \rangle \notin T,\ V(\ell, \mathbf{x}) \geq \varepsilon, \\
&\leq V - c.
\end{aligned}$$

($\Leftarrow$): In general the converse does not hold as $\lim_{x \to \infty} \frac{V(\ell,\mathbf{x})-c}{V(\ell,\mathbf{x})} = 1$. Therefore, no such α exists in general. However, over a compact (closed and bounded) X we can establish the converse.

Assume V satisfies $(p4)$ on compact state-space X we want to show V satisfies $(p3)$ for some $\alpha \in (0, 1)$. Since X is compact, and V is continuous (V is polynomial), then by the Extreme Value Theorem we know V achieves a maximum $V_{\max} > 0$. Let $\alpha = 1 - \frac{c}{V_{\max}}$, then $\alpha \in (0, 1)$.

$$\begin{aligned}
&\texttt{pre}\mathbb{E}(V, \Pi) \leq V - c, \text{ then} \\
&\frac{\texttt{pre}\mathbb{E}(V, \Pi)}{V} \leq \frac{V - c}{V}, \text{ since for all } \langle \ell, \mathbf{x} \rangle \notin T,\ V(\ell, \mathbf{x}) \geq \varepsilon > 0, \\
&\qquad = 1 - \frac{c}{V} \leq 1 - \frac{c}{V_{max}}, \text{ by definition of } V_{\max}, \\
&\texttt{pre}\mathbb{E}(V, \Pi) \leq \left(1 - \frac{c}{V_{\max}}\right) V = \alpha V.
\end{aligned}$$

□

The constraint of X being a compact set provides a sufficient condition to establish the converse that can be checked directly by observing X. In Section 4.4.3 we propose a sufficient condition on the **dynamics**: **bounded increase** of $\mathcal{F}_{\mathcal{T}}$ to establish certificate equivalence.

4.3.1 Direct Proof of Soundness of REACH-ADD

In this section, we provide a technique for proving that a PSTS Π with a final location ℓ_F is almost surely terminating under REACH-ADD.

Let $\mathcal{M} : \{M_n\}_{n=0}^{\infty}$ be a non-negative c-additive supermartingale for some $c > 0$. Let $\varepsilon > 0$ be a level set of $\mathcal{M}$ as in REACH-ADD. A sample path of $\mathcal{M}$ is said to cross the ε-level set if for some $n \geq 1$, $M_n(\omega) \leq \varepsilon$.

Theorem 4.3.3. A nonnegative c-additive supermartingale a.s. crosses the ε-levelset for any $\varepsilon > 0$.

Proof. Let $T = \inf_{n \geq 0} M_n \leq \varepsilon$ be the random variable that represents the stopping time then $\mathcal{M}$ crosses the ε-level set. The **stopped process** denoted $M_{\min(n,T)}$ (or M_n^T) has sample paths $m_0, \ldots, m_t, m_t, m_t, \ldots$. Note that $M_{\min(n,T)} \geq 0$ over all sample paths.

Define a stochastic process $Y_n = M_{\min(n,T)} + c\min(n, T)$, for all $n \geq 0$. In other words, for each sample path $y_n = m_n + cn$ if $n \leq T$, and $y_n = m_t + ct$ 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 $m_{\min(n,T)}$. Hence, the sample paths of Y_n are one-to-one correspondent with those of M_n^T.

Lemma 4.3.3. The stochastic process $\{Y_n\}_{n=0}^{\infty}$ is a supermartingale (relative to M_n^T) and $Y_n \geq 0$.

Proof. The non-negativity of Y_n for all n follows from the fact that $M_n^T \geq 0$ for all n and $c > 0$. Next, we show that Y_n is a s.m. For any sample path,

$$\mathbb{E}(Y_{n+1} \,|\, Y_n, M_n, T) = \mathbb{E}(M_{n+1} \,|\, Y_n, M_n) + c\min(n+1, T).$$

We identify two cases: when (a) $n + 1 \leq T$, or (b) $n + 1 > T$.

Case (a): $\mathbb{E}(Y_{n+1} \,|\, Y_n, M_n, T) = \mathbb{E}(M_{n+1} \,|\, M_n) + (n+1)c \leq M_n - c + (n+1)c \leq Y_n$.

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

In both cases, we conclude $\mathbb{E}(Y_{n+1}|M_n, Y_n, T) \leq Y_n$. □

Recall that under the Supermartingale Convergence Theorem 2.2.2, a nonnegative super martingale converges (samplewise) almost surely.

Therefore, with probability 1, a sample path $y_0, \ldots, y_n, \ldots$ converges to a finite value $Y(\omega) = \tilde{y}$. From the Supermartingale Convergence Theorem 2.2.2, it also follows that $\mathbb{E}(Y) \leq \mathbb{E}(Y_0)$. Moreover, $M_0 = Y_0$ and in Lemma 4.3.3 we showed $M_n \leq \mathbb{E}Y_n$ for all $n \geq 0$.

Lemma 4.3.4. For any convergent sample path $y_0, \ldots, y_n, \ldots$, the corresponding sample path $m_0, \ldots, m_n, \ldots$ of $\mathcal{M}$ eventually crosses the ε-level set.

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 the sake of contradiction, assume the sample path of $\mathcal{M}$ doesn't cross the ε-level set. Therefore, the stopping time $T = \infty$. This means, $m_n = y_n - nc$ for all $n \geq 0$. Choosing $\alpha = \varepsilon$, for any $n > N$, $m_n \leq \tilde{y} + \alpha - n\varepsilon \leq \tilde{y} - (n-1)\varepsilon$. Therefore, for $n > 1 + \frac{\tilde{y}}{c}$, we conclude that $m_n \leq \varepsilon$. This contradicts our assumption that $T = \infty$. □

To complete the proof of Theorem 4.3.3, we observe that (a) a sample path $y_0, \ldots, y_n, \ldots$ of $\{Y_n\}$ converges almost surely since $\{Y_n\}$ is a nonnegative s.m.; (b) for each convergent sample path the corresponding (unique) path $m_0, \ldots, m_n, \ldots$ crosses the ε-level set; therefore, (c) any sample path of $\{M_n\}$ crosses the ε-level set almost surely. □

Definition 4.3.3 (Additive Super Martingale Ranking Function)**.** *An additive supermartingale ranking function (SMRF) η is a supermartingale expression map that satisfies the following:*

- *$V(\ell) \geq \varepsilon$ for all $\ell \notin L_T$, and for all $\ell_T \in L_T$, $V(\ell_T) \in [0, \varepsilon]$;*
- *there exists a constant $c > 0$ s.t. for each transition $\tau : \langle \ell, \varphi, f_\tau \rangle$ with $\ell \notin L_T$ or $\varphi \not\subseteq \varphi_T$,*

$$(\forall \mathbf{x} \in X)\, \varphi(\mathbf{x}) \implies \texttt{pre}\mathbb{E}(V, \tau) \leq V(\ell)[\mathbf{x}] - c.$$

The SMRF definition above is a generalization of similar rules over discrete spaces, including the probabilistic variant rule [121] and Foster's theorem [72, 24, 25].

Theorem 4.3.4 (Soundness of REACH-ADD)**.** If a PSTS Π has an additive supermartingale ranking function V then every sample execution of Π reaches the target set T (terminates) **almost surely**.

For any sample execution of Π, we define the process $\{M_n\}$ where $M_n = V(\ell_n, \mathbf{x}_n)$ for all $n \geq 0$. It follows that $\{M_n\}$ is a ranking supermartingale. The rest follows from Theorem 4.3.3.

Example 4.3.2. *Consider the PSTS in Example 4.1.1 and Figure 4.2, the additive SMRF $V(\ell_4)$:* $\texttt{14} - \texttt{x} - \texttt{y}$ *and $V(\ell_{11})$:* $\texttt{x} + \texttt{y} - \texttt{10}$ *proves almost sure termination.*

4.4 Almost Sure Persistence and Recurrence

We now turn our attention to proving almost sure persistence $\Diamond\Box T$ and recurrence $\Box\Diamond T$ properties of polynomial stochastic systems. Like REACH-GEOM and REACH-ADD all proof rules involve finding a suitable "certificate" in the form of a stochastic analogue of a Lyapunov-like function over the state-space $S : L \times X$. The soundness of our approach (presented in Section 4.4.2) relies on certificate functions behaving as supermartingale expressions over the state variables of the stochastic system.

We begin by motivating these properties with an example from the literature and a caveat on the proper use of SMRF from the previous section in establishing persistence properties.

Motivating Example: Room Temperature Control. In [4], Abate et al. present a room temperature control problem subject to stochastic temperature perturbations. Suppose that there are two adjacent rooms, whose temperatures change according to the following stochastic difference equation:

$$x_i' := x_i + b_i(x_0 - x_i) + a \cdot \sum_{i \neq j}(x_j - x_i) + c_i\left(1 - \sigma\left(x_i\right)\right) + \nu_i, \text{ for } i \in \{1, 2\},$$

where $a = 0.0625$, $b_1 = 0.0375$, $b_2 = 0.025$ are respectively the inter-room and external heat convection constants, $x_0 = 6°\text{C}$ is the outdoor temperature, $c_1 = 0.65$, $c_2 = 0.6$ are the heat units supplied to the two rooms by the heater, and ν_1, ν_2 are i.i.d. stochastic random variables. The behavior of the heater is governed by the controller unit term σ. We focus on the evolution of the room temperatures within the range $[6, 33]^2$.

Abate et al. construct a (nonlinear) sigmoidal controller $\sigma(t)$ that keeps the temperatures within a comfortable range $S : [17, 22] \times [16, 23]$ and focus on bounding the probability that the system leaves S (i.e., **stochastic safety**) within finitely many steps under the influence of Gaussian noise. Figure 4.3 shows 100 sample executions of the system when the controller is approximated with a degree-7 polynomial:

$$\sigma(t) : 29.2 - 13.42t + 2.55t^2 - 0.26t^3 + 0.015t^4 - 5.13 \times 10^{-4}t^5 + 9.23 \times 10^{-6}t^t - 6.87 \times 10^{-8}t^7$$

under two different types of random noise: uniform $\mathcal{U}$ on a given range and normal $\mathcal{N}$.

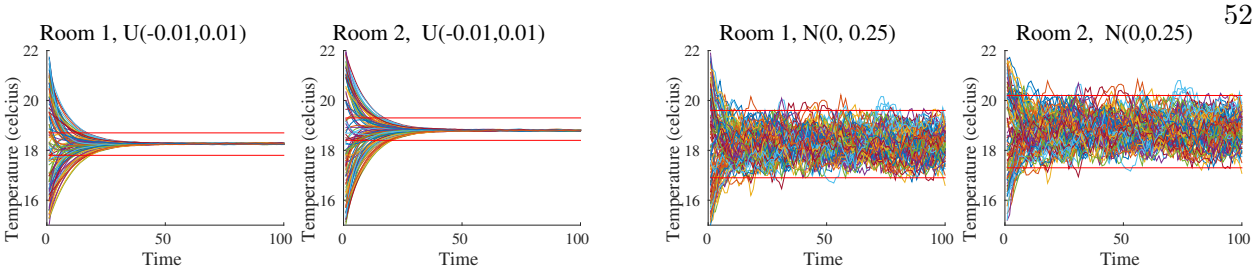


Figure 4.3: 100 simulations of the two-room controller system, with initial temperatures x_1, x_2 uniformly drawn from $[15, 22]^2$, under two different types of stochastic noise: (LEFT) $\nu_i \sim \mathcal{U}(-0.01, 0.01)$, with the red horizontal lines indicating the intervals $[17.8, 18.7]$ (for room 1) and $[18.4, 19.3]$ (for room 2); (RIGHT) $\nu_i \sim \mathcal{N}(0, 0.25)$, with the red horizontal lines indicating the intervals $[16.9, 19.6]$ (for room 1) and $[17.3, 20.2]$ (for room 2).

Controller σ was originally designed to keep the system in the comfortable range S with finite-time (100 min) guarantees in mind. We prove that under mild stochastic disturbances (LEFT) the system satisfies the **almost sure persistence** property $\Diamond\Box S$, i.e., with probability 1 the system eventually enters S and stays there forever. This is demonstrated by the proof rule PERSIST-GEOM of Section 4.4.1 and the certificate $V(x_1, x_2) : (x_1 - 18.3)^2 + (x_2 - 18.8)^2$. When the level of stochastic disturbance is increased (RIGHT) the almost sure persistence property no longer holds. This is consistent with the results in [4]; however, a weaker, almost sure **recurrence** property: $\Box\Diamond(16.9 \leq x_1 \leq 19.6 \wedge 17.3 \leq x_2 \leq 20.2)$ holds, i.e., with probability 1 the system visits the region infinitely often. This is demonstrated by proof rule REC of Section 4.4.4 using the same certificate function V.

Caveat: Additive SMRF for a.s. $\Diamond\Box(T)$. Given an additive supermartingale ranking function m, let $M_\kappa : \{\langle \ell, \mathbf{x}\rangle \in L \times X |\ m(\ell, \mathbf{x}) \leq \kappa\}$ denote the κ-sublevel set of m, for $\kappa \in \mathbb{R}$. For any κ such that $M_\kappa \neq \emptyset$, we can prove $\Diamond(M_\kappa)$ holds a.s. Does it follow immediately that $\Diamond\Box(M_\kappa)$ also holds? If we were to use our intuition from the deterministic case, it seems clear that $m(\ell, \mathbf{x})$ decreases continuously forever by at least some $\varepsilon > 0$, and therefore, it seems intuitive that $\Diamond\Box(m(\ell, \mathbf{x}) \leq \kappa)$ should hold. Surprisingly (or not), the following counterexample shows that in general, we **cannot** conclude a property $\Diamond\Box(M_\kappa)$ holds almost surely.

Example 4.4.1 (MOONWALK). *The* MOONWALK *system consists of a random walk over the state-space $X : \mathbb{Z}_{\leq 0}$ of the nonpositive integers:*

$$x_{n+1} := \begin{cases} x_n - 1 & \textit{with prob.} \quad p(x_n), \\ 0 & \textit{with prob.} \quad 1 - p(x_n), \end{cases}$$

wherein $p(x) : \frac{x-0.5}{x-1} = 1 - \frac{0.5}{1-x}$*, for* $x < 0$*, and* $p(0) = 1$*. In other words, the random walk either chooses to decrease* x *by* 1 *with probability* $p(x)$ *or jumps to* 0 *with probability* $1 - p(x)$*. Since there is a single location* $L = \{\ell\}$ *we omit it from the discussion. The initial state follows a negative Poisson distribution.*

The function $m(x) : x$ *is an additive supermartingale expression for the* MoonWalk *system:*

$$\mathbb{E}(m(x_{n+1})|x_n < 0) = \tfrac{x_n-0.5}{x_n-1}(x_n - 1) + 0 \cdot \left(\tfrac{-0.5}{x_n-1}\right) = x_n - 0.5.$$

For $x_n = 0$*, the pre-expectation is* $x_n - 1$*.*

Therefore, $m[x] : x$ *is an additive supermartingale expression. Yet the sublevel sets of the function* m *cannot be used for establishing persistence properties, because of the following result.*

Lemma 4.4.1. For any $\kappa < 0$, the probability that a sample path of MoonWalk satisfies $\Diamond\Box(x \leq \kappa)$ is 0.

Proof. The proof relies on events $\mathsf{Tail}(\eta, L)$ that are defined as the event that the system has an initial prefix of length $L > 0$ reaching $x \leq \kappa$, and the rest of the execution satisfies $x \leq \kappa$ forever. First, we prove that for any κ, $\mathbb{P}(\mathsf{Tail}(\kappa, L)) = 0$ for all L. Note that $\mathsf{Tail}(\eta, L) \subseteq \mathsf{Tail}(\eta, L+1)$, and

$$\mathbb{P}(\mathsf{Tail}(\eta, L)) \leq \prod_{j=-\eta}^{\infty} p(-j) = \prod_{j=-\eta}^{\infty} \left(1 - \frac{0.5}{j+1}\right) = 0\,.$$

The last product equals zero since the sum $\sum_{j=-\eta}^{\infty} \frac{0.5}{j+1}$ diverges. (See Lemma 4.4.2 below.) Event $\Diamond\Box(x \leq \kappa)$ can be seen as $\bigcup_{L=1}^{\infty} \mathsf{Tail}(\kappa, L)$. Since each $\mathsf{Tail}(\kappa, L)$ almost never happens (has zero measure), it follows that $\Diamond\Box(x \leq \kappa)$ almost never happens. □

For completeness, we provide a simple proof to the claim $\prod_{j=-T}^{\infty} \left(1 - \frac{0.5}{j+1}\right) = 0$*, which appeared in the proof of Lemma 4.4.1.*

Lemma 4.4.2. For any sequence $\{a_i\}_{i \geq 0} \subset (0, 1)$, $\prod_{i=0}^{\infty}(1-a_i) \in [0, 1]$. Moreover, $\prod_{i=0}^{\infty}(1-a_i) = 0$ if and only if the infinite sum $\sum_{i=0}^{\infty} a_i$ diverges.

Proof. Let $s_k = \prod_{i=0}^{k}(1-a_i)$ for all $k \geq 0$. Then $\{s_k\}_k$ is a nonincreasing sequence in $(0,1)$, implying that $\{s_k\}_k$ always converges, i.e., $\prod_{i=0}^{\infty}(1-a_i) \in [0,1]$. The equivalence can then be shown using the inequality $\frac{-t}{1-t} < \log(1-t) < -t$, which holds for any $t \in (0,1)$ and the inequality $\log(1-t) > -\frac{3}{2}t$ which holds for any $t \in (0, 0.5)$.

If $\prod_{i=0}^{\infty}(1-a_i) > 0$, then

$$-\sum_{i=0}^{\infty} a_i > \sum_{i=0}^{\infty} \log(1-a_i) = \log\left(\prod_{i=0}^{\infty}(1-a_i)\right) \in \mathbb{R},$$

implying that $\sum_{i=0}^{\infty} a_i \geq 0$ converges.

Conversely, suppose that $\prod_{i=0}^{\infty}(1-a_i) = 0$; we show that $\sum_{i=0}^{\infty} a_i$ diverges. Without loss of generality suppose that $\lim_i a_i = 0$. Let $\bar{K}$ be such that $a_k \leq 0.5$ for all $k \geq \bar{K}$. Then for any $L \in \mathbb{R}$, there exists an integer $K > \bar{K}$ such that $\sum_{i=\bar{K}+1}^{K} a_i \geq L$. In fact, pick $K > \bar{K}$ satisfying $s_K \leq \exp\left(-\frac{3}{2}L\right) s_{\bar{K}}$ (which is possible since $\lim_k s_k = 0$). Then

$$\begin{aligned} -\frac{3}{2}\sum_{i=\bar{K}+1}^{K} a_i < & \sum_{i=\bar{K}+1}^{K} \log(1-a_i) \\ = & \log(s_K) - \log(s_{\bar{K}}) \leq -\tfrac{3}{2}L, \end{aligned}$$

i.e., $\sum_{i=\bar{K}+1}^{K} a_i \geq L$. This together with the nonnegativity of a_i's shows that the sum $\sum_{i=0}^{\infty} a_i$ diverges. □

Using an additive supermartingale expression m to prove tail invariance properties of the form $\Diamond\Box(m(\mathbf{x}) \leq \kappa)$ requires additional assumptions on the expression m. One such condition is the bounded increase (Definition 4.4.1 ahead) which is assumed in Theorem 4.4.3, in order to establish the soundness of proof rules for persistence properties using additive supermartingale ranking functions.

4.4.1 Proof Rules for Persistence

We provide a series of proof rules for proving persistence properties. The relation between these rules is also examined.

PERSIST-GEOM: **Geometric rule for persistence**

$(p1)$	$(\forall\ s \in S)\ V(s) \geq 0,$	Positive semidefinite V.
$(p2)$	$(\exists\ \varepsilon > 0)\ (\forall\ s \in S \setminus T)\ V(s) \geq \varepsilon,$	Lower bound outside T.
$(p3)$	$(\exists\ \alpha \in (0,1))\ (\forall\ s \in S \setminus T)\ \mathcal{D}V(s) \leq (\alpha - 1)V(s),$	Drift condition outside T.
$(p5)$	$(\forall\ s \in T)\ \mathcal{D}V(s) \leq 0,$	Drift condition inside T.
	$\Diamond\Box(T)$ almost surely.	

PERSIST-ADD: **Additive rule for persistence**

$(p1)$	$(\forall\ s \in S)\ V(s) \geq 0,$	Positive semidefinite V.
$(p2)$	$(\exists\ \varepsilon > 0)\ (\forall\ s \in S \setminus T)\ V(s) \geq \varepsilon,$	Lower bound outside T.
$(p4)$	$(\exists\ c < 0)\ (\forall\ s \in S \setminus T)\ \mathcal{D}V(s) \leq -c,$	Drift condition outside T.
$(p5)$	$(\forall\ s \in S)\ \mathcal{D}V(s) \leq 0,$	Drift condition inside T.
	$\Diamond\Box(T)$ almost surely.	

Both PERSIST-GEOM and PERSIST-ADD state that a polynomial stochastic system Π satisfies $\Diamond\Box T$ almost surely if there exists a nonnegative certificate function V (condition $(p1)$) whose value outside T is lower bounded by some $\varepsilon > 0$ (condition $(p2)$). Moreover, the drift conditions ensure that in expected value V in the next step does not increase inside T (condition $(p5)$), and decreases by some fixed non-zero quantity outside T (an additive constant in $(p4)$, or, a multiplicative factor in $(p3)$). Intuitively, these conditions together guarantee that V is a supermartingale whose drift condition outside T forces its value to decrease along almost all sample paths and eventually reach a value ε at which point the sample path is "forced" to enter T and persists forever.

Example 4.4.2. *Consider the stochastic system* Π *of Example 4.3.1. We show that using the same certificate* $V(\ell, x) = x^2$ *the almost sure persistence property* $\Diamond\Box(T : |x| \leq 0.1)$ *holds.*

We already established that $V(\ell, x)$ *is nonnegative on* X *(condition* $(p1)$*), and* $V(\ell, x) \geq 0.01$ *for all* $x \in X \setminus T$ *(condition* $(p2)$*). Also, for all* $x \in X$*,*

$$\texttt{pre}\mathbb{E}(V, \Pi) = \mathbb{E}_w(V(\ell, x_{n+1})|x_n) = 0.02x_n^2, \text{ so } \mathcal{D}V(\ell, x) \leq -0.98V(\ell, x).$$

Hence, $V(\ell, x)$ defines a 0.02-multiplicative supermartingale expression (conditions $(p3)$, $(p4)$).

Applying PERSIST-GEOM*, we conclude that $\Diamond\Box(-0.1 \leq x \leq 0.1)$ holds a.s.*

Example 4.4.3. *For the polynomial stochastic system of Example 2.2.4 over the state-space $X = [0, 1]$, a single location ℓ and a self-looping transtion τ, we establish the almost sure persistence property $\Diamond\Box(x \leq 0.05 \vee x \geq 0.95)$.*

Consider the certificate function $V(\ell, x) = x(1 - x)$. For all $x \in X$, $V(\ell, x) \geq 0$, and for all $x \in X \setminus T = (0.05, 0.95)$, $V(\ell, x) \geq 0.0475$ (conditions $(p1)$, $(p2)$). Next, note that $\texttt{pre}\mathbb{E}(V, \Pi) = \texttt{pre}\mathbb{E}(V, \tau) = x(1 - x)(1 - x + x^2)$, and $\mathcal{D}V(x) = x(1 - x)(x^2 - x)$. It is easy to check that for all $x \in (0.05, 0.95)$, $\mathcal{D}V(x) \leq -0.00225625$, and for all $x \in [0, 0.05] \cup [0.95, 1]$, $\mathcal{D}V(x) \leq 0$ ($(p4)$, $(p5)$).

Applying PERSIST-ADD*, we conclude that $\Diamond\Box(x \leq 0.05 \vee x \geq 0.95)$ holds a.s.*

Note 2. Similar to the certificates for REACH-ADD and REACH-GEOM, in both of the above examples we could interchange certificates $V(\ell, x)$ to be used with either PERSIST-GEOM or PERSIST-ADD: $\mathcal{D}V(x) \leq -0.98x^2 \leq -0.0098$, for all $x \in X \setminus T$, in the first case; and $\mathcal{D}V(x) \leq x(1 - x)(x^2 - x) \leq -0.0475V(x)$ in the second. We expand on this point in Section 4.4.3.

Finally, if the set of locations L is a singleton we omit it from the description of Π.

Strong Persistence. We strengthened proof rules REACH-GEOM and REACH-ADD by requiring a drift condition to hold inside the target region T. As a result we are able to prove stronger properties. Rules PERSIST-GEOM and PERSIST-ADD present sufficient conditions under which certificates V prove that $\Diamond\Box T$ holds almost surely. Unfortunately, the difference in constraints inside and outside T may force V to be a high degree polynomial (or a piecewise polynomial function). To simplify constraints and make the search for certificates for almost sure persistence properties tractable we propose a stronger version of proof rules for persistence of the form: $(\forall\ \varepsilon > 0)\ \Diamond\Box(V(s) \leq \varepsilon)$.

SPERSIST-GEOM**: Geometric rule for strong persistence**

$(p1)$	$(\forall\ s \in S)\ V(s) \geq 0$,	Positive semidefinite V.
$(p6)$	$(\exists\ \alpha \in (0, 1))\ (\forall\ s \in S)\ \mathcal{D}V(s) \leq (\alpha - 1)V(s)$,	Drift condition.
	$(\forall\ \varepsilon > 0)\ \Diamond\Box(V(s) \leq \varepsilon)$ almost surely.	

SPERSIST-GEOM: **Geometric rule for strong persistence (cont)**

Similarly, we provide an additive version of strong persistence rule. We say a function $V(\ell, \mathbf{x})$ has **bounded increase** over Π iff there is a constant $C > 0$ such that, for every possible next state $\langle \ell', \mathbf{x}' \rangle$ reached from $\langle \ell, \mathbf{x} \rangle$, $|V(\ell', \mathbf{x}') - V(\ell, \mathbf{x})| \leq C$.

SPERSIST-ADD: **Additive rule for strong persistence**

$(p7)$	$(\exists\ c < 0)\ (\forall\ s \in S)\ \mathcal{D}V(s) \leq -c,$	Drift condition.
$(p8)$	$(\forall\ s \in S)\ V(s)$ has bounded increase,	(See above, and Def. 4.4.1 on page 60).
	$(\forall\ K)\ \Diamond\Box(V(s) \leq K)$ almost surely.	

SPERIST-GEOM presents a stronger, yet simpler to state and encode, version of the drift requirement dictated by PERSIST-GEOM (viz. inside the region T). Similarly, SPERSIST-ADD does not insist on V being positive definite but only V decreasing in expectation by $-c$ everywhere. The benefit of the stronger formulations of the persistence rule is that each level set of the Lyapunov-like certificate V acts as a **tail invariant**: a set S that almost all traces of the stochastic system reach and asymptotically confine to.

4.4.2 Soundness of Persistence Rules

The soundness of SPERSIST-GEOM and PERSIST-GEOM follow closely the proof of soundness of REACH-GEOM and rely on the convergence of α-multiplicative supermartingales.

Lemma 4.4.3. Let $\Pi : \langle X, \mathcal{R}, L, \mathcal{T}, \ell_0, \mathcal{D}_0 \rangle$ be a polynomial stochastic transition system. Let ρ map each sample $\omega \in \Omega$ to the corresponding sample path of Π, i.e., $\rho(\omega) : \langle \ell_0, \mathbf{x}_0 \rangle, \langle \ell_1, \mathbf{x}_1 \rangle, \ldots$ and let $\rho_m(\omega) : \langle \ell_m, \mathbf{x}_m \rangle$. If V is an α-multiplicative supermartingale expression of Π for some $\alpha \in (0, 1)$, then $\{V(\rho_i(\omega))\}_{i=0}^{\infty}$ is an α-multiplicative supermartingale.

Proof. Fix any $v \in \mathbb{R}$ such that $\exists\, \omega \in \Omega$ satisfying $V(\rho_n(\omega)) = v$. For any $\langle \ell, x \rangle \in L \times X$ satisfying $V(\ell, x) = v$, $\mathbb{E}(V(\rho_{n+1}(\omega)) \,|\, \rho_n(\omega) = \langle \ell, x \rangle) \leq \alpha v$, by definition of α-multiplicative supermartingale expressions. Hence, $\mathbb{E}(V(\rho_{n+1}(\omega)) \,|\, V(\rho_n(\omega)) = v) \leq \alpha v$, i.e, $\{V(\rho_n(\omega))\}_{n=0}^{\infty}$ is an α-multiplicative supermartingale. □

Now we prove the soundness of SPERSIST-GEOM.

Theorem 4.4.1 (Soundness of SPERSIST-GEOM)**.** A polynomial stochastic system Π satisfies the almost sure persistence property $(\forall\, \varepsilon > 0)\ \Diamond\Box(V(\ell, \mathbf{x}) \leq \varepsilon)$ if V is a function satisfying conditions $(p1), (p6)$ of SPERSIST-GEOM.

Proof. Fix any $\varepsilon > 0$. By Lemma 4.4.3, $\{V(\rho_n(\omega))\}_{n=0}^{\infty}$ is a nonnegative α-multiplicative supermartingale, satisfying the condition of Theorem 4.3.1. Hence,

$$\Pr(\{\omega \in \Omega \,|\, (\exists\, n_0)\ (\forall\, n \geq n_0)\ V(\rho_n(\omega)) \leq \epsilon\}) \geq$$
$$\Pr(\{\omega \in \Omega \,|\, \lim_{n \to \infty} V(\rho_n(\omega)) = 0\}) = 1,$$

i.e., $\Diamond\Box(V(\mathbf{x}) \leq \varepsilon)$ holds a.s. □

The proof of the soundness of PERSIST-GEOM is similar.

Theorem 4.4.2 (Soundness of PERSIST-GEOM)**.** A polynomial stochastic system Π satisfies the almost sure persistence property $\Diamond\Box(T)$ if there exists a function V that satisfies conditions $(p1)$-$(p3)$,$(p5)$ of PERSIST-GEOM.

Proof. Note that $\Diamond\Box(T)$ holds a.s. if and only if

$$\Pr(\{\omega \in \Omega \,|\, (\forall\, n_0)\ (\exists\, n \geq n_0)\ \rho_n(\omega) \notin T\}) = 0.$$

Since V satisfies conditions $(p1)$-$(p4)$, it suffices to show that $\Pr(\{\omega \in \Omega \,|\, \varphi(\omega)\}) = 0$, where φ is the predicate:

$$\varphi(\omega) : (\forall\, n_0)\ (\exists\, n \geq n_0)\ V(\rho_n(\omega)) \geq \varepsilon\ \wedge\ \texttt{pre}\mathbb{E}V(\rho_n(\omega)) \leq \alpha V(\rho_n(\omega)).$$

Let $\omega \in \Omega$ be such that $\varphi(\omega)$ is true. Then there is an infinite sequence of indices $\{n_\ell\}_{\ell=1}^{\infty}$ such that

$$V(\rho_{n_\ell}(\omega)) \geq \epsilon > 0 \ \wedge \ \texttt{pre}\mathbb{E}V(\rho_{n_\ell}(\omega)) \leq \alpha V(\rho_{n_\ell}(\omega)).$$

Define the stochastic process $\{M_n\}_n$ by

$$M_n : \frac{V(\rho_n(\omega))}{\alpha^\ell}, \quad \text{where } n_\ell < n \leq n_{\ell+1}.$$

We show that $\{M_n\}_n$ is a nonnegative supermartingale; following similar arguments in the proof of Theorem 4.3.1, the nonnegative supermartingale $\{V(\rho_n(\omega))\}_n$ converges to 0 a.s. This implies that $\varphi(\omega)$, which contains the requirement that $V(\rho_n(\omega)) \geq \varepsilon$ infinitely often, holds with probability zero. In fact, following the same arguments in Lemma 4.4.3, for any $v \in V(\rho_n(\Omega))$,

$$\mathbb{E}(V(\rho_{n+1}(\omega)) \,|\, V(\rho_n(\omega)) = v) \leq \begin{cases} \alpha v, & \text{if } n = n_\ell \text{ for some } \ell, \\ v, & \text{otherwise.} \end{cases}$$

When $n = n_\ell$ for some ℓ, for all $m \in \mathbb{R}$,

$$\begin{aligned} \mathbb{E}(M_{n_\ell+1} | M_{n_\ell} = m) &= \mathbb{E}\left(\frac{V(\rho_{n_\ell+1}(\omega))}{\alpha^{\ell+1}} \;\middle|\; \frac{V(\rho_{n_\ell}(\omega))}{\alpha^\ell} = m \right) \\ &\leq \frac{1}{\alpha^{\ell+1}} \cdot (\alpha \cdot \alpha^\ell m) = m. \end{aligned}$$

When $n_\ell < n < n_{\ell+1}$ for some ℓ (so $n+1 \leq n_\ell + 1$), for all $m \in \mathbb{R}$,

$$\begin{aligned} \mathbb{E}(M_{n+1} | M_n = m) &= \mathbb{E}\left(\frac{V(\rho_{n+1}(\omega))}{\alpha^\ell} \;\middle|\; \frac{V(\rho_n(\omega))}{\alpha^\ell} = m \right) \\ &\leq \frac{1}{\alpha^\ell} \cdot (\alpha^\ell m) = m. \end{aligned}$$

Hence $\{M_n\}_n$ is a nonnegative supermartingale. By the Supermartingale Convergence Theorem 2.2.2, M_n convergence samplewise almost surely. Since $\alpha \in (0,1)$ and $\{n_\ell\}_\ell$ is an infinite sequence, $\{V(\rho_n(\omega))\}_n$ converges to 0 a.s. Hence,

$$\begin{aligned} &\Pr(\{\omega \in \Omega \,|\, \varphi(\omega)\}) \\ \leq \ &\Pr(\{\omega \in \Omega \,|\, (\forall\, n_0)\ (\exists\, n \geq n_0)\ V(\rho_n(\omega)) \geq \varepsilon \ \wedge \ \{M_n\}_n \text{ is a nonneg. supermartingale}\}), \end{aligned}$$

which is 0, implying that $\Diamond\Box(T)$ holds a.s. $\square$

Necessity. The two necessary conditions on the multiplicative supermartingale $m(\ell, \mathbf{x})$ are $\alpha \in (0, 1)$ and $m(\ell, \mathbf{x})$ nonnegative. We illustrate their necessity through the following example.

Example 4.4.4. *Consider a stochastic transition system with a single variable x defined over the state space $[0, \infty)$ with two self-loop transitions τ_1 and τ_2. Transition τ_1 has a guard $x \geq 1$ and does not alter the value of x. Transition τ_2 has a guard $x < 1$ and chooses between $x' := 2x$ or $x' := \frac{x}{2}$ with equal probabilities. That is:*

$$x' := \begin{cases} x & \text{if } x \geq 1, \text{ and} \\ 2x & \text{if } x < 1, \text{with prob.} \frac{1}{2} \\ \frac{x}{2} & \text{if } x < 1, \text{with prob.} \frac{1}{2} \end{cases}$$

The initial value of x is exponentially distributed over $[0, \infty)$. Note that $m(x) : x$ is a nonnegative α-multiplicative supermartingale only when $\alpha = 1$. Clearly, m does not converge almost surely to zero.

Consider another transition system involving $x \in \mathbb{R}$ and a single transition with two forks: $x' := x$ with probability $\frac{2}{3}$, and $x' := -x$ with probability $\frac{1}{3}$. Clearly, $m(x) : x$ is a $\frac{1}{3}$-multiplicative supermartingale. However, m is not nonnegative over the state-space $\mathbb{R}$, so m does not prove any persistence property. Indeed, $\Diamond\Box(x \leq 1)$ is not a tail invariant for the system.

4.4.3 Relations Between Proof Rules

In Section 4.3 we proved the equivalence of certificates for reachability properties (Lemma 4.3.2). In Note 2 we alluded to the fact that certificates for rule PERSIST-GEOM can equivalently be used for rule PERSIST-ADD to prove persistence properties of polynomial stochastic systems. The following result establishes the relationship between geometric and multiplicative proof rules.

Definition 4.4.1 (Bounded Increase)**.** *An expression $m[\mathbf{x}]$ has **bounded increase** for a stochastic transitions system Π iff there exists $M > 0$ so that for all possible states $\mathbf{x} \in X$ and all possible next states $\mathbf{x}'$ reachable from $\mathbf{x}$, $|m(\mathbf{x}') - m(\mathbf{x})| \leq M$. A flow-sensitive expression map $V(\ell, \mathbf{x})$ has **bounded increase** iff there exists $M_V > 0$ s.t. for all $\langle \ell, \mathbf{x} \rangle \in S$, $|V(\ell', \mathbf{x}') - V(\ell, \mathbf{x})| \leq M_V$.*

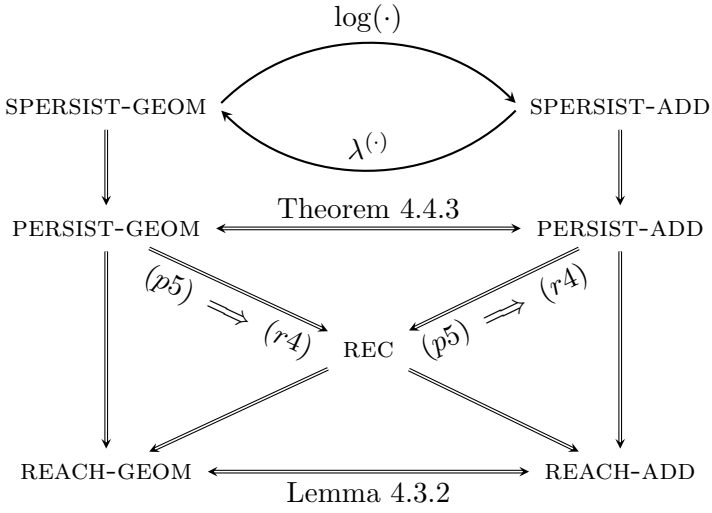


Figure 4.4: Diagram of the relations between persistence rules: double arrows denote a logical implication between rules, single arrows denote conversion of certificates via the labelling function.

We give an example of bounded increase expressions, which do not have to be bounded functions: whether a particular expression $m[\mathbf{x}]$ has bounded increase on a system depends as much on the system itself as on the growth of m.

Example 4.4.5. *Consider a stochastic system over $\mathbb{R}$, in which $x_{n+1} := x_n - 1 + w_n$, with w_n a uniform random variable over $[-1, 1]$. Then the function $m(x) : x$ has bounded increase property.*

If each w_n is a Gaussian random variable, then $m(x)$ does not satisfy the bounded increase property. Restricting the set of support of distribution w_n to a compact set (by truncation), however, allows x to satisfy the bounded increase property again.

Returning to the MoonWalk system in Example 4.4.1, the additive supermartingale $m(x) : x$, whose sublevel sets do not prove any tail invariance property (due to Lemma 4.4.1), does not satisfy the bounded increase property since it is possible to move from $x = -j$, for any $j > 0$, to $x = 0$ with nonzero probability.

From Additive to Multiplicative Supermartingales. In order to get the key points of the proof across we present a version using a supermartingale expression certificate. This proof extends to supermartingale ranking functions where $m(s) = V(\ell, \mathbf{x})$ and the argument is lifted over

the statespace $S = L \times X$.

Let Π be polynomial stochastic systems with state-space X. Let $\mathcal{X}$ denote the Borel measurable sets of X. Then $(X, \mathcal{X})$ is the Borel measurable space over system states of Π.

Definition 4.4.2. *A **Markov kernel** P over $(X, \mathcal{X})$ is a mapping*

$$P(\mathbf{x}, S) = Pr\{\mathbf{x}_{n+1} \in S \mid \mathbf{x}_n = \mathbf{x}\}, \quad \mathbf{x} \in X, S \in \mathcal{X},$$

with the following two properties:

- *for each $\mathbf{x} \in X$, function p mapping $S \mapsto P(\mathbf{x}, S)$ is a probability measure;*
- *for each $S \in \mathcal{S}$, function $\mathbf{x} \mapsto P(\mathbf{x}, S)$ is measurable.*

The kernel P is called **Markov** (or memoryless) because when making the choice for $\mathbf{x}_{n+1}$ it is oblivious to the entire history of the stochastic process $\{\mathbf{x}_i\}$ except for its current value $\mathbf{x}_n$. P is an extension of the probability transition matrix over the infinite state-space X that for each state $\mathbf{x}$, P induces a probability measure $p(\mathbf{x}'|\mathbf{x})$ over the possible next states $\mathbf{x}'$ starting from the current state $\mathbf{x}$.

Given a transition system Π, it corresponds to a Markov process over $(X, \mathcal{X})$ with a Markov kernel operator $P(\mathbf{x}, S)$ is defined as follows: let $\hat{h}_S : \texttt{pre}\mathbb{E}(\mathbb{1}(S), \Pi)$ be the pre-expectation operator applied to the indicator function over the set S according to the system Π. It follows that $P(\mathbf{x}, S) = \hat{h}_S(\mathbf{x})$.

Likewise, given a system specified as a Markov kernel P defined through a measure $p(\mathbf{x}'|\mathbf{x})$, the pre-expectation operator $\hat{h} : \texttt{pre}\mathbb{E}(h)$ is defined as the integral:

$$\texttt{pre}\mathbb{E}(h)[\mathbf{x}] : \int_X h(\mathbf{x}') dp(\mathbf{x}'|\mathbf{x})$$

Theorem 4.4.3. Let $m(\mathbf{x})$ be an ε-additive supermartingale expression that has bounded increase in Π. Then there exist positive constants $\lambda > 1$ and $\alpha < 1$ such that $\lambda^{m(\mathbf{x})}$ is an α-multiplicative supermartingale expression. Moreover, let $\kappa \in \mathbb{R}$ be such that $\{\mathbf{x} \in X \mid m(\mathbf{x}) \leq \kappa\}$ is nonempty. Then the system Π satisfies the tail invariance property $m(\mathbf{x}) \leq \kappa$ almost surely.

Proof. Let m be the additive supermartingale expression. Then there exists $\varepsilon > 0$ such that

$$\mathbb{E}(m(\mathbf{x}') \mid m(\mathbf{x})) = \int_X m(\mathbf{x}')dp(\mathbf{x}'|\mathbf{x}) \leq m(\mathbf{x}) - \varepsilon.$$

The expression m is bounded; therefore, there exists $M \geq 0$ such that $\forall \mathbf{x}, \mathbf{x}' \in X$, $(m(\mathbf{x}') - m(\mathbf{x})) \leq M$. We want to show that there exist $\lambda > 1$, $\alpha \in (0, 1)$ such that

$$\int_X \lambda^{m(\mathbf{x}')} p(\mathbf{x}'|\mathbf{x})d\mathbf{x}' \leq \alpha\lambda^{m(\mathbf{x})}, \quad \forall \mathbf{x} \in X.$$

This condition is equivalent to showing that for all $\mathbf{x} \in X$,

$$\int_X \lambda^{m(\mathbf{x}')-m(\mathbf{x})} p(\mathbf{x}'|\mathbf{x})d\mathbf{x}' \leq \alpha.$$

Let $\hat{m}$ denote $m(\mathbf{x}')$ and m denote $m(\mathbf{x})$. Let $f(\mathbf{x}, \lambda) = \int_X \lambda^{\hat{m}-m} dp(\mathbf{x}'|\mathbf{x})$. It is easy to see that $\forall \mathbf{x} \in X$ and for all $\lambda > 1$, $f(\mathbf{x}, \lambda) > 0$. We know $f(\mathbf{x}, \lambda)$ is continuous and it is easy to check that $f(\mathbf{x}, 1) = 1$. Moreover,

$$\begin{aligned}\frac{\partial}{\partial\lambda} f(\mathbf{x}, \lambda) &= \int_X \frac{\partial}{\partial\lambda} \lambda^{m(\mathbf{x}')-m(\mathbf{x})} dp(\mathbf{x}'|\mathbf{x}) \\ &= \int_X (\hat{m} - m)\lambda^{\hat{m}-m-1} dp(\mathbf{x}'|\mathbf{x})\,.\end{aligned}$$

Therefore, at $\lambda = 1$, we obtain

$$\frac{\partial f}{\partial\lambda} = \int_X (\hat{m} - m)dp(\mathbf{x}'|\mathbf{x}) \leq -\varepsilon\,.$$

Next, we obtain

$$\begin{aligned}\frac{\partial^2}{\partial\lambda^2} f(\mathbf{x}, \lambda) &= \int_X (\hat{m} - m)(\hat{m} - m - 1)\lambda^{\hat{m}-m-2} dp(\mathbf{x}'|\mathbf{x}) \\ &\leq M(M-1)\lambda^{M-2},\end{aligned}$$

Now consider an interval $\lambda \in [1, 1+\Delta]$ for which, $M(M-1)\lambda^{(M-2)} \leq K$. Without loss of generality, we can also choose a K such that $K \geq \epsilon^2$.

The Taylor expansion of $f(\mathbf{x}, \lambda)$ around $\lambda = 1 + \delta$ and $\delta \in [0, \Delta]$:

$$\begin{aligned}f(\mathbf{x}, 1+\delta) &= f(\mathbf{x}, 1) + \delta\tfrac{\partial}{\partial\lambda} f(\mathbf{x}, 1) + \tfrac{\delta^2}{2}\tfrac{\partial^2}{\partial\lambda^2} f(\mathbf{x}, \theta) \\ &\leq 1 - \epsilon\delta + \tfrac{K}{2}\delta^2\end{aligned}$$

The last inequality arises from the facts that $\frac{\partial f}{\partial \lambda}|_{\lambda=1} \leq -\varepsilon$ and $\frac{\partial^2 f}{\partial \lambda^2} \leq K$ for $\lambda \in [1, 1+\Delta]$.

Choosing $\delta = \frac{\epsilon}{K}$, we obtain,

$$f(\mathbf{x}, 1+\delta) \leq 1 - \frac{\epsilon^2}{K} + \frac{\epsilon^2}{2K} = 1 - \frac{\epsilon^2}{2K}$$

$$f\left(\mathbf{x}, 1+\frac{\epsilon}{K}\right) \in (0, 1-\frac{\epsilon^2}{2K})$$

Let us choose $\lambda = 1 + \frac{\epsilon}{K}$ and $\alpha = 1 - \frac{\epsilon^2}{2K}$. We now note that for all $\mathbf{x} \in X$,

$$\int_X \lambda^{m(\mathbf{x}')} dp(\mathbf{x}'|\mathbf{x}) \leq \alpha \lambda^{m(\mathbf{x})}.$$

Finally, the tail invariance property $\Diamond\Box(m(\mathbf{x}) \leq \kappa)$ follows from Theorem 4.3.1. □

As a corollary we can state the following result:

Theorem 4.4.4 (Soundness of PERSIST-ADD)**.** Let $m(\mathbf{x})$ be a bounded increase, ε-additive supermartingale expression. Let T be any number such that $\{\mathbf{x} \in X \mid m(\mathbf{x}) \leq T\}$ is nonempty. Then the system Π satisfies the tail invariance property $m(\mathbf{x}) \leq T$ almost surely.

In fact, it is possible to prove the converse of Theorem 4.4.3.

Lemma 4.4.4. Let $m(\mathbf{x})$ be a positive α-supermartingale expression for $\alpha \in (0,1)$. For $\varepsilon = \log(\alpha)$, the expression $m_{\log}(\mathbf{x}) = \log(m(\mathbf{x}))$ is an ε-additive supermartingale expression.

Proof. The result follows immediately by an application of Jensen's Inequality to the concave, monotone function $\log(\cdot)$. □

This shows that any positive α-multiplicative supermartingale expression used to prove a tail invariant property has an equivalent additive supermartingale (more precisely, SMRF) formulation. The following example demonstrates, however, that sometimes the α-supermartingale formulation may be more natural to employ.

Example 4.4.6. *Figure 4.5 shows a nonlinear Markov jump system with two modes q_1, q_2 and two state variables $\mathbf{x} : (x, y)$ that evolve according to the mode-dependent difference equations. The system jumps between modes with equal probability.*

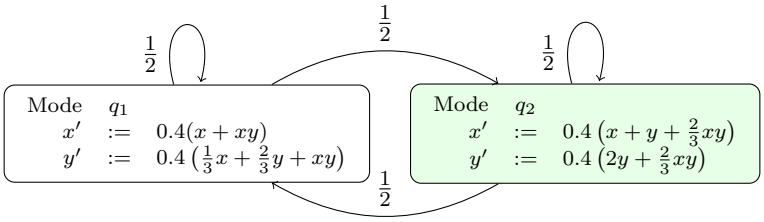


Figure 4.5: A two-state nonlinear Markov jump system with two modes of evolution: q_1 and q_2. Evolution in each mode is defined by the corresponding difference equations. Transition between modes occurs with equal probabilities.

Observe that 0 is an equilibrium and $X : [-0.5, 0.5]^2$ *is an invariant of the system, i.e., all sample paths starting in* X*, stay in the set forever. Figure 4.6(a) shows the sample paths that start inside* X *converge towards* 0*. We establish that the persistence property* $\Diamond\Box(|x| \leq 0.1 \wedge |y| \leq 0.1)$ *holds almost surely over all executions of the system, by synthesizing the nonnegative certificate function* $V(\mathbf{x}) : 2.3x^2 + 4.15xy + 3.7y^2$*. After one time step, the expected value of* V *is at most* $\frac{1}{2}$*-fraction of its original value, i.e.,* $(\forall \mathbf{x} \in X)\ \mathbb{E}(V(\mathbf{x}')|\mathbf{x}) \leq \frac{1}{2}V(\mathbf{x})$*. Figure 4.6(b) plots the function* V *over the sample paths, showing its convergence. We use the certificate* $V(\mathbf{x})$ *in* Persist-Geom *(page 55) to establish the required property.*

Outside X*, the system appears unstable as shown in Figure 4.6(c), yet the behaviors approach* $x = y$ *asymptotically. Using the certificate* $\hat{V}(\mathbf{x}) : (x - y)^2$ *in* spersist-geom *(page 56) we can prove that* $(\forall\ \varepsilon > 0)\ \Diamond\Box(|x - y| \leq \varepsilon)$*.*

Relations Between Proof Rules. Figure 4.4 summarizes the relationship between the different proof rules for establishing persistence properties of a polynomial stochastic system. We first establish the equivalence of certificates for rules persist-geom and persist-add as alluded to in Note 2.

Lemma 4.4.5. Every persist-geom certificate V is also a certificate for persist-add and vice-versa.

Proof. The proof follows directly from the definition of $\mathcal{D}$ and the lower bound ε in condition $(p2)$. □

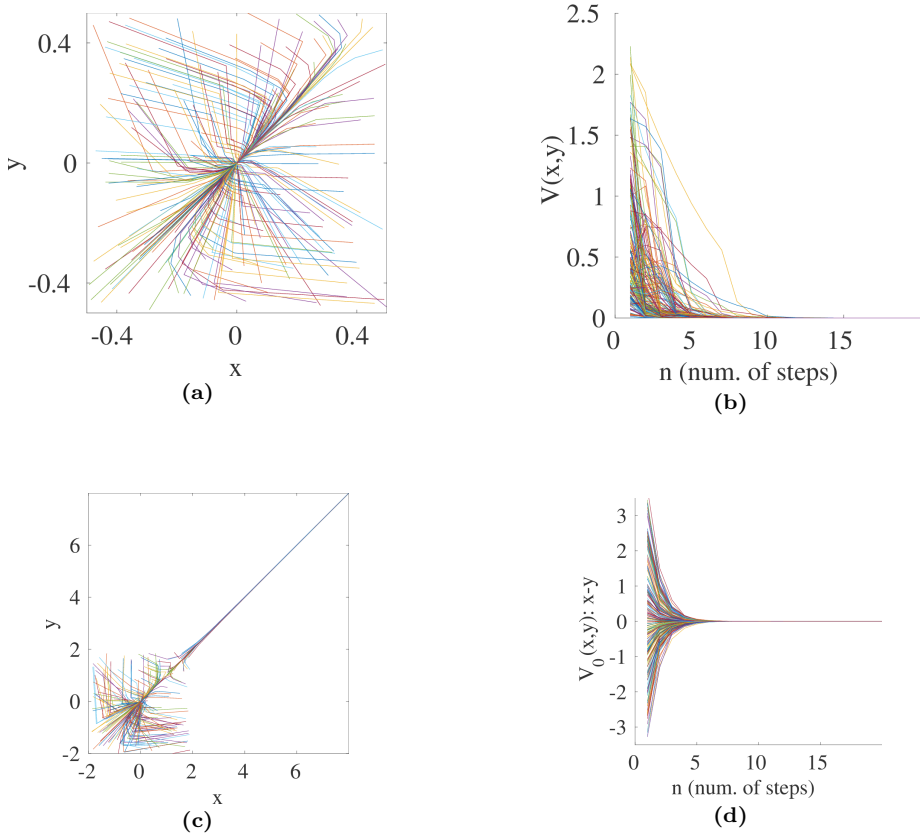


Figure 4.6: Sample paths of the Markov jump system described in Figure 4.5.

Under some technical conditions, it is possible to prove the converse of Theorem 4.4.3. This shows that any positive α-multiplicative supermartingale expression used to prove a tail invariant property has an equivalent additive supermartingale (more precisely, SMRF) formulation and vice versa.

Incompleteness. We demonstrate that although sound, our approach is incomplete. The existence of a nonnegative α-multiplicative supermartingale expression or a SMRF of bounded increase is sufficient but not a necessary condition for the system to almost surely satisfy a tail invariant property.

Example 4.4.7 (Incompleteness)**.** *Consider the* MoonWalk *system with modified probability* $p(x) : 1 - \frac{0.5}{(x-1)^2}$*, for* $x < 0$*, and* $p(0) = 1$*. The probability of the event* $\{x > \kappa\}$ *is* $\sum_{j=-\kappa}^{\infty} \frac{0.5}{(j+1)^2}$*, which converges. By the Borel-Cantelli Lemma [61, 2.3.1],* $\Pr(x > \kappa \text{ i.o.}) = 0$ *holds, i.e., the tail invariant* $\Diamond\Box(x < \kappa)$ *holds; however, the system does not have the bounded increase property.*

4.4.4 Proof Rule for Recurrence

We now focus on proof rules for proving the almost sure recurrence property: $\square\lozenge(T)$, i.e., T is visited infinitely often by almost all sample paths. The proof rule is almost identical to a related rule that establishes "positive recurrence" in Markov chains [122].

REC: Rule for Recurrence

$(r1)$	$(\forall\ s \in S)\ V(s) \geq 0,$	Positive semidefinite V.
$(r2)$	$(\exists\ \varepsilon > 0)\ (\forall\ s \in S \setminus T)\ V(s) \geq \varepsilon,$	Lower bound outside T.
$(r3)$	$(\exists\ c > 0)\ (\forall\ s \in S \setminus T)\ \mathcal{D}V(s) \leq -c,$	Drift condition outside T.
$(r4)$	$(\exists\ H)\ (\forall\ s \in T)\ \mathcal{D}V(s) \leq H,$	Bounded Drift inside T.

$\square\lozenge(T)$ almost surely

Condition $(r4)$ guarantees that Π does not make unbounded jumps so that $\{V(\rho_n(\omega))\}_{n=0}^{\infty}$ reaches (REACH-ADD: $(r1)$-$(r3)$) almost surely within finite time to the target set T.

4.5 PSTS with Bounded Nondeterminism

In this section we relax the "No Nondeterminism" restriction in our transition model with a bounded form of nondeterminism (see Ferrer et al. [67], Chatterjee et al. [36, 35] for similar extensions). Specifically, we introduce a new set of **nondeterministic** variables $U = \{u_1, \ldots, u_q\}$ disjoint from X and R that range over a bounded domain $\mathcal{U}$. Transition guards φ_τ are predicates over the system X and the nondeterministic variables U. Transition update functions are polynomial functions of the joint state $\mathbf{x}, \mathbf{r}, \mathbf{u}$ of the system, stochastic, and nondeterministic variables.

We introduce the nondeterministic variables U with the goal to remove the **mutual exclusion** ($i \neq j \implies \varphi_i(\mathbf{x}) \wedge \varphi_j(\mathbf{x}) = false$) restriction on the transition guards of a PSTS. This means that for a given state $\langle \ell, \mathbf{x} \rangle \in L \times X$ more than one transition is potentially enabled. Thus, a single step of execution of the model could result in a **set** of distributions $\{\mathcal{D}'_1, \ldots \mathcal{D}'_p\}$ over the post states (one for each enabled transition). By lifting transition guards to be **mutually exclusive**

predicates over $\mathbf{x}, \mathbf{u}$ instead, we allow for the nondeterministic choice of which enabled transition is selected on every step to be resolved by a **transition scheduler**.

Definition 4.5.1 (Transition Scheduler)**.** *Let $\Pi : \langle X, \mathcal{R}, \mathcal{U}, L, \mathcal{T}, \ell_0, \mathcal{D}_0 \rangle$ be a polynomial stochastic transition system over a (filtered) probability space $(\Omega, \mathcal{E}, \{\mathcal{F}_n\}_{n=0}^{\infty}, P)$. Let $\mathcal{T} = \{\tau_1, \dots \tau_p\}$ the set of transitions and suppose mutual exclusion is not necessarily enforced for the transition guards, i.e., for transitions $\tau_i : \langle \ell, \varphi_i, f_i \rangle$, $\tau_j : \langle \ell, \varphi_j, f_j \rangle$, $\exists \mathbf{u}_i, \mathbf{u}_j \in \mathcal{U}$, $\varphi_i(\mathbf{x}, \mathbf{u}_i) \wedge \varphi_j(\mathbf{x}, \mathbf{u}_j)$ is satisfiable.*

*A **transition scheduler** $\phi : X \times \mathbb{N} \to \mathcal{U}$ is an $\mathcal{F}_{n-1}$-measurable function such that for any state $\rho_n(\omega) = \langle \ell_n, \mathbf{x}_n \rangle$:*

$$\phi(\mathbf{x}_n, n+1) = \mathbf{u}_{n+1} \implies (\mathbf{x}_n, \mathbf{u}_{n+1}) \models \varphi_j \wedge \ell_n = \ell_j,$$

i.e., the scheduler ϕ resolves all nondeterministic choice $\mathbf{u}$ and thereby selects (schedules) a single enabled transition $\tau_j : \langle \ell_j, \varphi_j, f_j \rangle \in \mathcal{T}$. We denote by Φ the set of all transition schedulers.

Given a transition scheduler $\phi \in \Phi$, the **execution under** ϕ of the nondeterministic PSTS proceeds on every step n by first selecting an enabled transition τ_j from state $\langle \ell_{n-1}, \mathbf{x}_{n-1} \rangle$ according to $\mathbf{u}_n = \phi(\mathbf{x}_{n-1}, n)$, then updating location and system variables via the transition update $f_{\tau_j}(\mathbf{x}_{n-1}, \mathbf{r}_n, \mathbf{u}_n)$ as before to corresponding post-state values $\langle \ell', \mathbf{x}' \rangle$. Since a unique transition $\tau_{\phi(\mathbf{x}_n, n+1)}$ is scheduled on every step this induces a deterministic PSTS Π^{ϕ} with piecewise system update function $\mathcal{F}_{\mathcal{T}}^{\phi}$. We require that once the choice $\mathbf{u}_n$ is made, that all moments of distribution $\mathcal{D}_n$ exist and are finite.

We refer to Π^{ϕ} as the **induced** PSTS and denote its sample executions as:

$$\rho^{\phi}(\omega) \triangleq \langle \ell_0, \mathbf{x}_0 \rangle \xrightarrow{\tau_{\phi(\mathbf{x}_0, 1)}} \langle \ell_1, \mathbf{x}_1 \rangle \xrightarrow{\tau_{\phi(\mathbf{x}_1, 2)}} \cdots \xrightarrow{\tau_{\phi(\mathbf{x}_{n-1}, n)}} \langle \ell_n, \mathbf{x}_n \rangle \cdots .$$

A property P of the nondeterministic PSTS Π is said to hold almost surely if and only if P holds almost surely for Π^{ϕ}, for all transition schedulers $\phi \in \Phi$.

Notice the post-distribution of Π under ϕ is again $\textsc{PostDistrib}^{\phi}(s) = \textsc{PostDistrib}(s, \tau_{\phi})$. Similarly, Definitions 4.2.2, 4.2.3, 4.2.4 can be lifted to pre-expectations of Π under ϕ by defining:

$$\texttt{pre}\mathbb{E}(\texttt{e}, \Pi^{\phi}) = \sum_{i=1}^{n} \mathbb{1}(\varphi_i^{\phi}(\mathbf{x}, \mathbf{u})) \cdot \texttt{pre}\mathbb{E}(\texttt{e}, \tau_i)(\mathbf{x}, \mathbf{u}), \quad \text{for all } \mathbf{x} \in X$$

where $\mathbb{1}(\varphi_i^\phi) = 1$ if ϕ selects enabled transition i, and 0, otherwise. And define the operator

$$\text{maxpre}\mathbb{E} = \sup_{\phi} \texttt{pre}\mathbb{E}(\texttt{e}, \Pi^\phi)$$

to be the the scheduler that selects that maximum pre-expectation among all possible schedulers.

Soundness of Proof Rules. We make the following observation: all certificate functions V used in the soundness proofs are defined over the sample executions of the underlying PSTS. Without loss of generality, fix a transition scheduler ϕ. Notice that in Definition 4.2.5 we imposed a conservative requirement that a supermartingale expression map (and therefore, a supermartingale ranking function) decreases in expectation across every transition τ over all states in which τ is enabled (and not only those states that are reachable!). This means that convergence properties of $\{V_n(\omega) : V(\rho_n^\phi(\omega))\}_{n=0}^\infty$ hold over all possible choices ϕ can make (i.e., regardless of the specific choices that scheduler ϕ makes). This means that our proof rules present sufficient conditions to guarantee soundness under maxpre$\mathbb{E}$ and, therefore, under all possible transition schedulers $\phi \in \Phi$.

We conclude that a property P of the nondeterministic PSTS holds almost surely if there exists a supermartingale ranking function certificate V satisfying the conditions of the corresponding proof rule REACH-ADD, REACH-GEOM, PERSIST-ADD, PERSIST-GEOM, or REC with maxpre$\mathbb{E}$ substituted in for $\texttt{pre}\mathbb{E}$.

Chapter 5

Synthesizing Linear and Polynomial Proof Certificates

In Chapter 4 we presented a deductive analysis approach that establishes sufficient conditions to prove qualitative properties for a polynomial stochastic transition system. These conditions are presented in the form of proof rules whose key aspect is identifying a suitable polynomial certificate function that behaves as a ranking supermartingale over the runs of the PSTS.

In the previous chapter we provided the specific constraints that each certificate function needs to satisfy in order for the corresponding proof rule to be sound. In this chapter we demonstrate how two well-studied constraint-based approaches can be applied in the context of polynomial stochastic transition systems to automatically generate polynomial certificates. The first approach closely follows [46, 47, 147] and relies on solving systems of **linear** inequalities over the program expressions of the PSTS. It produces **linear** certificate functions. The second follows [135, 6, 17] and leverages efficient semidefinite programming techniques to solve classes of polynomial inequalities and produce polynomial certificates.

Example 5.0.1 (The tortoise and the hare)**.** *Consider the program shown in Figure 5.1. It shows a program that manipulates two real-valued variables* `h` *and* `t`*. Initially, the value of* `t` *is set to* 30 *and* `h` *to* 0*. The loop iterates as long as* $\mathtt{h} \leq \mathtt{t}$*. Each loop iteration increments* `t` *by* 1 *while the value for* `h` *may remain unchanged with probability* $\frac{1}{2}$*, or increase by a uniform random variable over the interval* $[0, 10]$*. Does this program terminate almost surely? Using the techniques of Chapter 4 we establish almost-sure termination using a* ***supermartingale*** *ranking expression* $\mathtt{t} - \mathtt{h}$*. It is initially positive and whenever its value is non-positive, the loop termination condition is achieved. Finally,*

```
real h = 0; // h is hare
real t = 30; // t is tortoise
while (h <= t){
  if flip (1/2) {
      h = h + unifRand(0,10);
  }
  t = t + 1;
} // terminate a.s.?
```

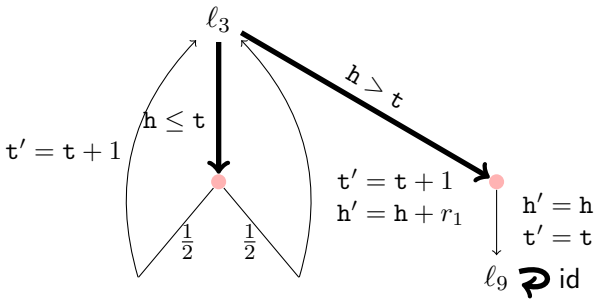


Figure 5.1: (LEFT) A simple probabilistic program inspired by the tortoise and the hare fable [5]. (RIGHT) The corresponding PSTS for the program on the left showing the two transitions and self-loop id.

for each iteration of the loop, the value of this expression decreases ***in expectation*** *by at least* 1.5. *Therefore, by rule* REACH-ADD *we guarantee this loop terminates almost surely.*

In this chapter, we describe a procedure that automatically generates this expression. Furthermore, we also derive the martingale expression $2.5\mathtt{t} - \mathtt{h}$ *which will be of interest in Chapter 6.*

5.1 Preliminaries

A **linear constraint** φ over X is an inequality of the form $c_1x_1 + \cdots c_nx_n + c_0 \geq 0$ for some coefficients $c_0, \ldots, c_n \in \mathbb{R}$ and real-valued variables $x_1, \ldots, x_n$. We write coefficients $c_1, \ldots, c_n$ in vector form as $\mathbf{c}$ and denote with $\bar{\mathbf{c}}$ the vector $\mathbf{c}$ extended with c_0. The linear inequality above is then: $\varphi(\bar{\mathbf{c}}, \mathbf{x}) : \mathbf{c}^T\mathbf{x} + c_0 \geq 0$. A linear inequality $\varphi(\bar{\mathbf{c}}, \mathbf{x})$ is called **homogeneous** if $c_0 = 0$ and **conic** if each $c_i \geq 0$. An **convex constraint** is a conic constraint if in addition to each c_i being nonnegative, $\sum_{i=0}^{n} c_i = 1$. A **linear (convex) assertion** $\Phi(X)$ over X is a conjunction $\bigwedge_{i=1}^{k} \varphi_i(\bar{\mathbf{c}}_i, \mathbf{x})$ of linear (convex) constraints over X.

A **polyhedron** P is the set of points satisfying a linear assertion. The set of all points that satisfy a homogeneous convex assertion forms a **convex (polyhedral) cone**.

Theorem 5.1.1 (Farkas' Lemma [65])**.** Let S be a system of linear inequalities over X of the form:

$$S: \left[\begin{array}{ccccccc} a_{11}x_1 & + & \cdots & + & a_{1n}x_n & + & b_1 \geq 0 \\ \vdots & & & & \vdots & & \vdots \\ a_{m1}x_1 & + & \cdots & + & a_{mn}x_n & + & b_m \geq 0 \end{array}\right]$$

for some $m \in \mathbb{N}$.

Suppose S is satisfiable, then it entails a linear inequality

$$\psi(\mathbf{c}, \mathbf{x}) : c_1x_1 + \cdots + c_nx_n + c_0 \geq 0$$

if and only if there exist **multipliers** $\lambda_0, \lambda_1, \ldots, \lambda_m \in \mathbb{R}^+$ such that

$$c_1 = \sum_{i=1}^{m} \lambda_i a_{i1}, \quad \cdots, \quad c_n = \sum_{i=1}^{m} \lambda_i a_{in}, \quad c_0 = \left(\sum_{i=1}^{m} \lambda_i b_i\right) + \lambda_0.$$

Moreover, S is unsatisfiable if and only if $\psi_{false} : 0 \geq 1$ can be derived as a conic combination of the linear inequalities of S.

Farkas' Lemma provides a complete method for solving a system S of linear inequalities, that is, either there exist multipliers $\lambda : \lambda_0, \ldots, \lambda_m$ with the desired properties to show $\psi(\mathbf{c}, \mathbf{x}) \geq 0$, or there exist multipliers $\bar{\lambda} : \bar{\lambda}_0, \ldots, \bar{\lambda}_m$ that can prove the unsatisfiability of S via ψ_{false}. An extension to Farkas' Lemma that also handles strict inequalities is referred to as Motzkin Transposition Theorem [130] and both can be found in standard textbooks [41, 26].

Polynomials. Let $\mathbb{R}[X]$ denote the set of all multivariate polynomials on X. A **polynomial constraint** over X is an inequality of the form $p(\mathbf{x}) \geq 0$ where $p(\mathbf{x}) \in \mathbb{R}[X]$ is a non-negative polynomial on $x_1, \ldots, x_n$. A **polynomial assertion** $\varphi(X) : \bigwedge_{i=1}^{k} p_i(\mathbf{x}) \geq 0$ is a conjunction of polynomial inequalities over X. The set of points satisfying a polynomial assertion

$$S(\bar{p}) = \{\mathbf{x} \in X \,|\, p_1(\mathbf{x}) \geq 0 \wedge \ldots \wedge p_k(\mathbf{x}) \geq 0\}$$

is called a **semi-algebraic** set.

A polynomial $p(\mathbf{x})$ is a **sum-of-squares** (SOS) polynomial iff it can be represented as

$$p = p_1^2 + \ldots + p_k^2, \qquad \text{for some } k \in \mathbb{N},$$

and some polynomials $p_1, \ldots, p_k$ over $\mathbf{x}$.

While proving p is a nonnegative polynomial can be done by finding *some* SOS representation of p, this can be expensive because it involves an search over all possible representative polynomials $p_1, \ldots, p_k$. To make the problem tractable, we often fix a basis of representative polynomials $g_1(\mathbf{x}), \ldots, g_k(\mathbf{x})$ and prove that a polynomial p is nonnegative over the semi-algebraic set

$$S(\bar{g}) \triangleq \{\mathbf{x} \in \mathbb{R}^n \,|\, g_1(\mathbf{x}) \geq 0, \ldots, g_m(\mathbf{x}) \geq 0\}.$$

Definition 5.1.1 (Putinar Representation)**.** *A polynomial $p \in \mathbb{R}[X]$ has a **Putinar representation** over $S(\bar{g})$ if and only if $p = \sigma_0 + \sum_{i=1}^{m} \sigma_i \cdot g_i$, for σ_i sum of square polynomials.*

We then leverage a result analogous to Farkas' Lemma but over **polynomial** assertions called Putinar's Positivstellensatz [143].

Theorem 5.1.2 (Putinar's Positivstellensatz)**.** Let $S(\bar{g}) \triangleq \{\mathbf{x} \in \mathbb{R}^n \,|\, g_1(\mathbf{x}) \geq 0, \ldots, g_m(\mathbf{x}) \geq 0\}$ be a compact semi-algebraic set. Let q be a polynomial of the form

$$q = \lambda_0 + \sum_{i=1}^{m} \lambda_i g_i$$

for some SOS polynomials $\lambda_0, \lambda_1, \ldots, \lambda_m$. If the semi-algebraic set $\{\mathbf{x} \,|\, q(\mathbf{x}) \geq 0\}$ is compact, then every $p(\mathbf{x}) > 0$ on $S(\bar{g})$ has a Putinar repreresentation:

$$p = \sigma_0 + \sum_{i=1}^{m} \sigma_i g_i,$$

for SOS polynomials $\sigma_0, \sigma_1, \ldots, \sigma_m$.

We refer to $\sigma_0, \sigma_1, \ldots, \sigma_m$ as the **certificates** that establish the non-negativity of p. The search for certificates of p can be formulated as:

$$(\forall \mathbf{x} \in S)\, \lambda_0 + \sum_{i=1}^{m} \lambda_i g_i \geq 0 \implies \sigma_0 + \sum_{i=1}^{m} \sigma_i g_i > 0.$$

This means that $p(\mathbf{x}) > 0$, for all $\mathbf{x} \in T$, if

$$(\forall \mathbf{x} \in S(\bar{g}))\ (\sigma_0 - \lambda_0) + \sum_{i=1}^{m} \sigma_i g_i - \sum_{i=1}^{m} \lambda_i g_i \geq 0.$$

This leads to the following sufficient condition $p - q \in SOS$:

$$(\forall \mathbf{x} \in S(\bar{g}))\ \bar{\sigma}_0 + \sum_{i=1}^{m} (\bar{\sigma}_i) g_i \geq 0.$$

where we require that all $\bar{\sigma}_i = \sigma_i - \lambda_i \in SOS$. In practice, we fix a degree $d \in \mathbb{N}$ and consider all SOS $\bar{\sigma}_i$ up to degree d. We refer to this as the truncated **quadratic module** of degree d and use it in Section 5.3 to encode constraints.

Finally, given a PSTS Π, a **polyhedral (polynomial) invariant** I of Π maps every location ℓ to a linear (polynomial) assertion that overapproximates the set of states $\langle \ell, \mathbf{x} \rangle$ reachable from $\langle \ell_0, \mathbf{x}_0 \rangle$. A common technique to obtain polynomial invariants is to use standard (non-probabilistic) abstract interpretation [51].

5.2 Linear Supermartingale Certificate Generation

Our goal is to automatically discover supermartingale expression maps and super martingale ranking functions (SMRF). Our approach builds upon previous work by Colón et al. and Sankaranarayanan et al. for constraint-based invariant and ranking function discovery for standard (non-probabilistic) non-deterministic transition systems [46, 47, 147]. We restrict our approach to **affine PSTS** which restrict the guard φ of any transition $\tau : (\ell, \varphi, f_\tau)$, to be **polyhedral** (i.e., a linear assertion over X), and update function f_τ to be a piece-wise affine function:

$$\text{for each fork } (m_i, g_i) \text{ of } f_\tau, \quad g_i(\mathbf{x}, \mathbf{r}) : A_i \mathbf{x} + B_i \mathbf{r} + \mathbf{a}_i, \tag{5.1}$$

for some real $n \times n$ matrices A_i, $n \times m$ matrices B_i, and real vectors $\mathbf{a}_i$.

A **template expression** is a bilinear form $c_0 + \sum_{i=1}^{n} c_i x_i$ with unknowns $c_0, \ldots, c_n$. We may also consider a template expression map η that maps each location ℓ_j to a template expression $\eta(\ell_j) : c_{j0} + \sum_{i=1}^{n} c_{ji} x_i$. We collectively represent the unknown coefficients as a vector $\mathbf{c}$. We encode the conditions for a template expression (map) corresponding to our objective: martingales, supermartingales or supermartingale ranking functions. Solving the resulting constraints directly yields (super) martingales.

Example 5.2.1. *Consider the PSTS in example 5.0.1 (see Figure 4.2). We wish to discover a supermartingale expression using the template* $c_1\mathtt{h} + c_2\mathtt{t} + c_0$ *at locations* ℓ_3, ℓ_9.

Encoding Supermartingales. In this section section we discuss how the conditions on the pre-expectations of a linear supermartingale expression (map) can be encoded in a system of linear inequalities using Farkas Lemma. Let $\tau : (\ell, \varphi, f_\tau)$ be a transition with k forks $f_\tau : (m_1, g_1(\mathbf{x}, \mathbf{r})), \ldots, (m_k, g_k(\mathbf{x}, \mathbf{r}))$. Let η be a template expression map. We wish to enforce that η is a supermartingale (see Definitions 4.2.1):

$$(\forall\ \mathbf{x} \in X) \quad (\varphi[\mathbf{x}]) \implies \mathtt{pre}\mathbb{E}(\eta, \tau) \leq \eta(\ell)[x].$$

That is,

$$(\forall\ \mathbf{x} \in X) \quad (\varphi[\mathbf{x}]) \implies \sum_{i=1}^{k} p_i \cdot \mathbb{E}_R\left[\eta_\ell(g_i(\mathbf{x}, \mathbf{r}))\right] \leq \eta(\ell)[\mathbf{x}]. \tag{5.2}$$

Recall that in (5.1) we require each fork update $g_i(\mathbf{x}, \mathbf{r})$ to be an affine function. Each $\eta(m_i, g_i(\mathbf{x}, \mathbf{r}))$ can thus be expressed as $\eta(m_i, g_i(\mathbf{x}, \mathbf{r})) = \mathbf{c}^T A_i \mathbf{x} + \mathbf{c}^T B_i \mathbf{r} + \mathbf{c}^T \mathbf{a}_i$.

Example 5.2.2. *Returning back to Example 5.2.1, we wish to encode pre-expectation condition for the transition* $\tau : (\ell_3, (\mathtt{h} \leq \mathtt{t}), f_1, f_2)$, *where* $f_1 : (\ell_3, \mathtt{h}' \mapsto \mathtt{h}, \mathtt{t}' \mapsto \mathtt{t} + 1)$ *with probability* $p_1 : \frac{1}{2}$, *and* $f_2 : (\ell_3, \mathtt{h}' \mapsto \mathtt{h} + r_1, \mathtt{t}' \mapsto \mathtt{t} + 1)$, *with probability* $\frac{1}{2}$. *We encode the pre-expectation condition for* τ*:*

$$(\forall \mathtt{h}, \mathtt{t})\,(\mathtt{h} \leq \mathtt{t}) \implies \begin{bmatrix} \frac{1}{2}\mathbb{E}_{r_1}(c_1\mathtt{h} + c_2(\mathtt{t}+1) + c_0) + \\ \frac{1}{2}\mathbb{E}_{r_1}(c_1(\mathtt{h} + r_1) + c_2(\mathtt{t}+1) + c_0) \end{bmatrix} \leq c_1\mathtt{h} + c_2\mathtt{t} + c_0\,.$$

We note that $\mathbb{E}_{r_1}(r_1) = 5$ *(See Figure 5.1). By linearity of expectation, we obtain*

$$(\forall \mathtt{h}, \mathtt{t})\,(\mathtt{h} \leq \mathtt{t}) \implies c_1\mathtt{h} + c_2\mathtt{t} + c_2 + \frac{1}{2}c_1\mathbb{E}_{r_1}(r_1) \leq c_1\mathtt{h} + c_2\mathtt{t}\,.$$

Simplifying, we obtain $(\forall \mathtt{h}, \mathtt{t})\ (\mathtt{h} \leq \mathtt{t}) \implies c_2 + \frac{5}{2}c_1 \leq 0$. *Here, the RHS is independent of the variables* $\mathtt{h}, \mathtt{t}$. *Moreover, any constant term* c_0 *is preserved by the* $\mathtt{pre}\mathbb{E}$ *operator and cancels out. Therefore, we obtain* $c_2 + \frac{5}{2}c_1 \leq 0$.

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

$$\mathbb{E}_R\left[\eta_{m_j}(g_j(\mathbf{x}, \mathbf{r}))\right] = \mathbf{c}^T A\mathbf{x} + \mathbf{c}^T B\mu + \mathbf{c}^T\mathbf{a}\,. \tag{5.3}$$

To encode the supermartingale property for transition τ, we use Farkas Lemma to encode the implication

$$(\forall\, \mathbf{x})\,(\varphi[\mathbf{x}]) \implies \underbrace{\mathtt{pre}\mathbb{E}(\eta,\tau)}_{\text{template expression}} \leq \underbrace{\eta(\ell)[\mathbf{x}]}_{\text{template expression}} \tag{5.4}$$

Let φ be satisfiable and represented in the constraint form as $A\mathbf{x} \leq \mathbf{b}$. Below is a suitable reformulation of the Farkas Lemma.

Theorem 5.2.1 (Farkas Lemma)**.** The linear constraint $A\mathbf{x} \leq \mathbf{b} \implies \mathbf{c}^T\mathbf{x} \leq d$ is **valid** if and only if its **alternative** is satisfiable $A^T\lambda = \mathbf{c} \ \wedge\ \mathbf{b}^T\lambda \geq d \ \wedge\ \lambda \geq 0$.

Encoding the entailment in Equation (5.4) using Farkas Lemma ensures that the resulting constraints are linear inequalities.

Example 5.2.3. *Continuing with Example 5.2.2, the transition* id *yields the constraint* true*. Therefore, the only constraint is* $c_2 + \frac{5}{2}c_1 \leq 0$*. Solving, we obtain the line* $(c_1 : 1, c_2 : \frac{-5}{2})$*, yielding the martingale* $2\mathtt{h} - 5\mathtt{t}$*, while the ray* $(c_1 : -1, c_2 : 0)$ *yields the supermartingale expression* $-\mathtt{h}$*. Other supermartingales such as* $\mathtt{t} - \mathtt{h}$ *can obtained as linear combinations.*

Finding Additive Supermartingale Ranking Functions. The process of discovering SMRFs is similar, but requires extra constraints to reflect the additional conditions SMRFs satisfy. An abstract interpretation pass can be used to yield helpful invariants by treating the random variables and forks as nondeterministic choices. Given an affine PSTS Π, a polyhedral invariant $I(\ell)$ of Π and a target set $T \subseteq S$ we encode a SMRF as follows:

(1) To encode the non-negativity, we use the invariants at each location $\ell \neq \ell_T$, $I(\ell) \models \eta(\ell) \geq \varepsilon$. For location ℓ_T, we encode $I(\ell_T) \models 0 \leq \eta(\ell_T) < \varepsilon$. The latter condition requires the Motzkin transposition theorem, a generalization of Farkas lemma that deals with strict inequalities [105]. Here ε is treated as an unknown constant, whose value is also inferred as part of the process.

(2) The drift (or adequate decrease) condition is almost identical to that for supermartingales.

However, we introduce an unknown $c > 0$ and require that for each transition $\tau : \langle \ell_\tau, \varphi_\tau, f_\tau \rangle$:

$$I(\ell_\tau) \wedge \varphi_\tau \models \texttt{pre}\mathbb{E}(\eta, \tau) \leq \eta(\ell) - c.$$

Example 5.2.4. *Revisiting Example 5.2.2, we perform an abstract interpretation to obtain the facts* $I(\ell_3) : 0 \leq \texttt{h} \leq \texttt{t} + 9 \wedge \texttt{h} \leq 9\texttt{t} - 270 \wedge \texttt{t} \geq 30$ *and* $I(\ell_9) : \texttt{h} > \texttt{t} \wedge \texttt{h} \leq \texttt{t} + 9$. *We use the template* $\eta(\ell_3) : c_{3,1}\texttt{h} + c_{3,2}\texttt{t} + d_3$ *and* $\eta(\ell_9) : c_{9,1}\texttt{h} + c_{9,2}\texttt{t} + d_9$. *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* $c = \frac{3}{2}$ *and* $\varepsilon = 9$. *This yields the SMRF* $\eta(\ell_3) : \texttt{t} - \texttt{h} + 18$, $\eta(\ell_9) : \texttt{t} - \texttt{h} + 9$. *When the value of* η *drops below* $\varepsilon = 9$ *the PSTS is in the target location* ℓ_9.

Note 3. Notice that $\texttt{pre}\mathbb{E}$ operator preserves positive multiplicative factors and additive terms: if $\eta(\ell) = \mathbf{c}_\ell^T \mathbf{x}$ is a template expression then according to Equations 5.2, 5.3:

$$\texttt{pre}\mathbb{E}(\eta, \tau) = \sum_{i=1}^{k} p_i \cdot [\mathbf{c}_\ell^T A_i \mathbf{x} + \mathbf{c}_\ell^T B_i \mu + \mathbf{c}_\ell^T \mathbf{a}_i] = \mathbf{c}^T \cdot \sum_{i=1}^{k} p_i \cdot [A_i \mathbf{x} + B_i \mu + \mathbf{a}_i],$$

and so for any real $\lambda > 0$,

$$\texttt{pre}\mathbb{E}(\lambda \cdot \eta, \tau) - \lambda\eta(\ell) = \lambda(\texttt{pre}\mathbb{E}(\eta, \tau) - \eta(\ell)).$$

This means that scaling each program expression by some multiple λ_c scales the linear constraints so that the adequate decrease equals 1 ($c = 1$). In Example 5.2.4 we could take $\lambda_c = \frac{2}{3}$ and $\eta(\ell_3) = \frac{2}{3}\texttt{t} - \frac{2}{3}\texttt{h} + 12$, $\eta(\ell_9) = \frac{2}{3}\texttt{t} - \frac{2}{3}\texttt{h} + 6$ to obtain an SMRF with $\varepsilon = 6$. Alternatively, we could scale η by $\lambda_K = \frac{1}{K} = \frac{1}{9}$ to guarantee a bound $\varepsilon = 1$, where

$$K \triangleq \max_{\langle \ell', \mathbf{x}' \rangle} |\eta(\ell', \mathbf{x}') - \eta(\ell, \mathbf{x})|$$

is the maximum possible increase of η over the post-states of the system.

Finding Multiplicative Supermartingale Ranking Functions. Let us consider the supermartingale drift condition

$$\texttt{pre}\mathbb{E}(\eta, \tau) \leq \eta(\ell) \iff \texttt{pre}\mathbb{E}(\eta, \tau) - \eta(\ell) \leq 0$$

and α-multiplicative drift condition

$$\texttt{pre}\mathbb{E}(\eta,\tau) - \alpha\eta(\ell) \leq 0, \quad \text{for some } \alpha \in (0,1).$$

The latter one we can write as

$$\mathbf{c}^T \cdot \sum_{i=1}^{k} p_i \left(B_i\mu + \mathbf{a}_i\right) \leq \mathbf{c}^T \left(\alpha I - \sum_{i=1}^{k} p_i A_i\right) \mathbf{x}.$$

Unfortunately, in general the drift condition presents a bilinear constraint: we need to find an $\alpha \in (0,1)$ and a vector of coefficients $\mathbf{c}$ whose product $\alpha\mathbf{c}^T\mathbf{x}$ appears in a constraint. This may require a policy iteration scheme whereby we first find a vector $\mathbf{c}$ such that for some α (say, initially $\alpha = 1$) the constraint holds then attempt to minimize α until a vector of coefficients $\mathbf{c}$ and $\alpha \in (0,1)$ that satisfies all constraints is found.

Affine PSTS and linear certificates. One may think that the class of affine PSTS is simple (given the polynomial dynamics we presented in Chapter 4) and that in general linear SMRF certificates may be sufficient to establish the almost sure reachability and repeated reachability properties.

Unfortunately, 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--;
}
```

Using recurrence properties of symmetric random walks, one can show that the program above terminates almost surely. Yet, no linear SMRF can found since `x` is martingale and does not show adequate decrease. In fact the expected time to termination is infinite, i.e., for the stopping time $T : \{x = 0\}$, $\mathbb{E}T = \infty$. If the flip probability is changed to $0.5 - \delta$, for any $\delta \in (0, 0.5)$, then the expression `x` becomes a δ-additive SMRF for the program above.

5.3 Polynomial Supermartingale Certificate Generation

Given a polynomial stochastic system Π, a polynomial invariant $I(\ell)$ of Π and a semi-algebraic target set $T \subseteq S$, the problem of finding polynomial certificates V that prove reachability, persistence or recurrence properties is, in general, intractable. For instance, proving strong persistence is equivalent to solving a polynomial optimization problem that under arbitrary number of variables X and degrees of polynomials constraints is known to be NP-hard.

In order to address the complexity of the problem, we follow the lead of many of successful research techniques and impose several restrictions on the proof rules so that finding solutions become more tractable. One of the most influential approaches to generating non-negative polynomials under constraints is to reduce the problem to sum-of-squares (SOS) optimization for which efficient techniques exist (see e.g. [6, 17] and the references therein). To illustrate, we consider SPERSIST-GEOM and PERSIST-GEOM; the formulations for the other proof rules are analogous.

Certificates for Persistence. Recall that for proving strong persistence properties via the geometric rule SPERSIST-GEOM, we need to find a function V such that

$$(p1) \quad (\forall\, s \in S)\ V(s) \geq 0 \qquad \text{Positive definite V.}$$

$$(p6) \quad (\exists \alpha \in (0,1))\ (\forall\, s \in S)\ \mathcal{D}V(s) \leq -\alpha V(s) \quad \text{Drift condition.}$$

We impose the following restrictions to make the feasibility problem tractable.

Restriction 1: We require that V is a **polynomial template** of a fixed maximal degree d_V:

$$V(\mathbf{c}, \mathbf{x}) = \sum_{i=1}^{m} c_i m_i(\mathbf{x}),$$

where $c_i \in \mathbb{R}$ denote unknown coefficients and $m_i(\mathbf{x})$ denote some fixed set of monomials with maximum degree d_V. Let d_{f_τ} denote the degree of the polynomial update function in each transition τ of Π and let $d_f = \max_{\tau \in \mathcal{T}} d_{f_\tau}$. This means that $\mathcal{D}V$ is also a polynomial template with degree $d_V d_f$. Moreover, as we saw in Section 5.2, $\mathcal{D}V$ can be expressed in terms of the unknown coefficients of V and the moments μ of the random variable $\mathbf{r}_c$.

Example 5.3.1. *Consider the PSTS of Example 4.4.6 and fix a degree-2 monomial basis* $\bar{m} = \{m_1 : \mathtt{x}^2,\ m_2 : \mathtt{xy},\ m_3 : \mathtt{y}^2\}$. *The degree-2 polynomial template expression* V *over the monomial*

basis $\bar{m}$ *is*

$$V(\mathbf{c}, \mathbf{x}) = c_0\mathtt{x}^2 + c_1\mathtt{xy} + c_2\mathtt{y}^2.$$

The pre-expectation $\mathtt{pre}\mathbb{E}(V, f)$*, and therefore,* $\mathcal{D}V$ *is a degree* $d_V d_f = 2 \cdot 2$ *template over the monomials* $\{m_1 : \mathtt{x}^2, m_2 : \mathtt{xy}, m_3 : \mathtt{y}^2, m_4 : \mathtt{x}^2\mathtt{y}, m_5 : \mathtt{xy}^2, m_6 : \mathtt{x}^2\mathtt{y}^2\}$.

Restriction 2: We replace the nonnegativity constraints by the more restrictive **sum-of-squares** (SOS) constraints, i.e., we require that both V and $-\mathcal{D}V$ be sums of squares of some unknown polynomial functions. We also require that V is positive definite, which is a common regularity condition assumed in semidefinite optimization and allows us to find an $\alpha \in (0, 1)$ such that the condition $(p6)$ in SPERSIST-GEOM holds.

$(p1)$	$V \in SOS$, $V(\cdot, \mathbf{x}) > 0$, for $\mathbf{x} \neq 0$	Positive definite SOS V
$(p6)$	$-\mathcal{D}V \in SOS$	SOS drift condition.

Under these two restrictions, the generally intractable feasibility problem from SPERSIST-GEOM is equivalent to a linear semidefinite feasibility problem: a polynomial being a sum of squares (of polynomial functions) is equivalent to its vector of coefficients being the image of an unknown positive semidefinite matrix under a predetermined linear transformation. (For more details on SOS relaxation techniques for solving polynomial feasibility/optimization problems, see e.g. [6, 17].)

For proving persistence properties with respect to a nonempty set T via the geometric rule PERSIST-GEOM, we need to find a function V such that conditions $(p1)$-$(p3)$,$(p5)$ hold.

$(p1)$	$(\forall\ s \in S)\ V(s) \geq 0$,	Positive semidefinite V.
$(p2)$	$(\exists\ \varepsilon > 0)\ (\forall\ s \in S \setminus T)\ V(s) \geq \varepsilon$,	Lower bound outside T.
$(p3)$	$(\exists\ \alpha \in (0,1))\ (\forall\ s \in S \setminus T)\ \mathcal{D}V(s) \leq -\alpha V(s)$,	Drift condition outside T.
$(p5)$	$(\forall\ s \in T)\ \mathcal{D}V(s) \leq 0$,	Drift condition inside T.

We require that V is a polynomial of degree at most d and that $T = \{\mathbf{x} \,|\, g_1(\mathbf{x}) \geq 0 \wedge \cdots \wedge g_\ell(\mathbf{x}) \geq 0\}$ for some polynomials $g_1, \ldots, g_\ell$. Then we replace those constraints pertaining to the elements in T, by truncated **quadratic module** membership. For instance, we replace the condition $(p5)$ by

the tractable constraint:

$$\mathcal{D}V = \sigma_0 + \sigma_1 g_1 + \cdots + \sigma_\ell g_\ell, \quad \sigma_0, \sigma_1, \ldots, \sigma_\ell \text{ SOS of degree at most } \tilde{d}, \text{ for some } d \in \mathbb{N}.$$

The tractability of such a constraint is due to the fact that the polynomials σ_i being SOS can be phrased as semidefinite feasibility constraints. Similar treatment can be applied on those constraints pertaining to the elements in $X \setminus T$, which is also a semialgebraic set. Finally, constraints $(p3)$, $(p5)$ are encoded as polynomial (SOS) constraints via Putinar's Positivstellensatz.

5.4 Synthesis Results

In this section we present the results of two implementations: one for affine PSTS that produce linear supermartingale and supermartingale certificate functions with focus on proving reachability properties, and a second one, for polynomial stochastic systems and polynomial supermartingale certificates for proving persistence and recurrence properties.

5.4.1 Linear Certificates

We implemented the ideas of Section 5.2 in a constraint generation framework that reads in the description of a PSTS and generates constraints for supermartingale expression maps. The tool uses the Parma Polyhedra Library [9] to generate all possible solutions to these constraints in terms of martingale and supermartingale expressions. In Chapter 7 we present an abstract interpretation based approach that can also automatically infer invariants using a numerical polyhedral abstract domain.

Example 5.4.1 (Robot Dead Reckoning)**.** *Dead reckoning is an approach for position estimation starting from a known* ***fix*** *at some time* $t = 0$*. Figure 5.2* (LEFT) *shows a model for robot navigation that involves estimating the actual position* $(\mathtt{x}, \mathtt{y})$ *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* $\mathtt{dxc}, \mathtt{dyc}$ *capture the commanded direction, whereas the actual directions are slightly*

```
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 }
```

Figure 5.2: (LEFT) Probabilistic program model for dead reckoning and (RIGHT) Modeling a betting strategy for Roulette.

off by a random value. Our goal is to estimate how the position $\texttt{x},\texttt{y}$ *deviates from the actual position* $\texttt{estX},\texttt{estY}$*. Our analysis shows that expressions* $\texttt{x} - \texttt{estX}$ *and* $\texttt{y} - \texttt{estY}$ *are martingales at the loop head and, therefore, stay close together. We provide a quantitative result to this problem in Chapter 6.*

Example 5.4.2 (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 5.2 (*RIGHT*). The model captures the various outcome combinations* $(\{Bank\} \uplus \{Red, Black\} \times \{1,2,3\})$*, including the one where the bank wins outright with probability* $\frac{1}{37}$*. Our analysis discovers the martingale expression* $15 \times \texttt{i} - 74 \times \texttt{money}$ *which can be used to bound the probability of the* $\texttt{money}$ *exceeding a certain quantity after* n *rounds. We generate the SMRF:* $-\texttt{money}$*. Thus, the program terminates almost surely in the gambler's* ***ruin****.*

Table 5.1 shows a summary of the evaluation of our approach over a set of 11 linear PSTS benchmarks. A description of the remaining benchmarks as well as the inferred properties are

ID	Description	$\lvert X\rvert$	$\lvert R\rvert$	$\lvert L\rvert$	$\lvert T\rvert$	#M	#S
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	discr. inverted pendulum stoch. disturb.	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

Table 5.1: Results on a set of benchmark programs. #**M** is the number of non-trivial martingales discovered and # **S** is the number of non-trivial supermartingales. All timings are under 0.1 seconds on Macbook Air laptop with 8 GB RAM, running Mac OS X 10.8.3.

provided in Appendix A.1.

5.4.2 Polynomial Certificates

There are many standard semidefinite optimization solvers (see e.g., Mittelmann [125]) and SOS optimization front-ends (such as SOSTOOLS [3], Yalmip [115], and others) available for solving the SOS optimization problems outlined in Section 5.3. We present some simple examples on the use of SPERSIT-GEOM rules for proving persistence. In each example, an α-multiplicative supermartingale expression is obtained using SDPT3-4.0 [153] on MATLAB R2014b, taking less than 10 seconds on a Linux machine with Intel(R) Core(TM) i7-4650U CPU @ 1.70GHz.

Example 5.4.3. *(Rimless wheel model [29, 120, 151]) A rimless wheel with 8 equally spaced inelastic spokes of length L rolls down a hill with stochastic slope angle γ. Let ω_n be the angular velocity at the n-th impact (which occurs when the stance leg is vertical). In [29, 151], the dynamics of the rimless wheel is described as:*

$$x_{n+1} := \cos^2\theta\left(x_n + \tfrac{2g}{L}\left(1-\cos\left(\tfrac{\theta}{2}+\gamma\right)\right)\right) - \tfrac{2g}{L}\left(1-\cos\left(\tfrac{\theta}{2}-\gamma\right)\right),$$

where $x_n = \omega_n^2$, g is the gravitational constant, $\theta = 45^\circ$ is the angle between two consecutive spokes and $\gamma \sim \mathcal{N}(8,1)$ (in degrees). We approximate the functions $\xi \mapsto \cos(\frac{\theta}{2} \pm \xi)$ over the interval $[5, 11]$

by degree 2 polynomials, and find that the angular velocity in the approximated stochastic system goes to 0 almost surely when $L = 2g$.

Notice that the certificate function $V(x) = 0.00085x^3 + x^4$ *satisfies conditions* $(p1)$, $(p6)$ *with* $X : [0, \infty)$ *and* $\alpha = 0.95$*: V and $-\mathcal{D}V$ are nonnegative on X and $\mathcal{D}V(x) \leq -0.05V(x)$ for all $x \geq 0$. Hence, V is a α-multiplicative supermartingale for this system over X, and $\Diamond\Box(V(x) \leq \varepsilon)$ holds a.s. for any $\varepsilon > 0$. In other words, despite the randomness in the slope of the terrain, the rolling rimless wheel (with very long spokes) would eventually become stationary almost surely.*

Example 5.4.4. *(Room temperature control [4]) In the two-room temperature control example from Section 4.4, we are interested in the evolution of the room temperatures within the range* $X = [6, 33]^2$. *Consider the nonnegative function* $V(x_1, x_2) : (x_1 - 18.3)^2 + (x_2 - 18.8)^2$. *When the noise follows the uniform distribution* $\mathcal{U}(-0.01, 0.01)$, *$\mathcal{D}V$ is nonpositive on X, and* $V(x_1, x_2) \geq 0.09$ *and* $\mathcal{D}V(x_1, x_2) \leq -0.01V(x_1, x_2)$ *for all* $x \in X \setminus [17.8, 18.7] \times [18.4, 19.3]$. *Hence conditions* $(p1)$-$(p4)$ *hold, implying the persistence property* $\Diamond\Box(17.8 \leq x_1 \leq 18.7 \wedge 18.4 \leq x_2 \leq 19.3)$.

In the case of Gaussian noise $\mathcal{N}(0, 0.25)$, $\mathcal{D}V(x_1, x_2) \leq 0.25$ *for all* $(x_1, x_2) \in [16.9, 19.6] \times [17.3, 20.2]$, *and* $V(x_1, x_2) \geq 0.8$ *and* $\mathcal{D}V(x_1, x_2) \leq -6 \times 10^{-5}$ *for all* $(x_1, x_2) \in X \setminus [16.9, 19.6] \times [17.3, 20.2]$.

Hence conditions $(r1)$-$(r4)$ *hold, implying the almost sure recurrence property* $\Box\Diamond(16.9 \leq x_1 \leq 19.6 \wedge 17.3 \leq x_2 \leq 20.2)$.

We list some additional examples in which one of the systems found in Appendix A.2 is proved to satisfying a persistence or recurrence property via the proof rules from Section 4.4. The details of these examples and the properties we establish are found in the Appendix.

Stochastic Systems	Noise u_j (i.i.d.)	Supermartingale $V(x,y)$
$x' := x + \frac{1}{2}y + u_1,$ $y' := \frac{1}{2}x + y - u_2$ (over $X = \mathbb{R}^2$)	$\mathcal{N}(-1,1)$	$\max(x-y,0)$ (for proving recurrence)
$x' := 0.5(x+y) + 0.4u_1\sqrt{x^2+y^2},$ $y' := 0.5(x-y) + 0.4u_2\sqrt{x^2+y^2},$ (over $X = \mathbb{R}^2$)	$\mathcal{N}(0,1)$	$x^2 + y^2$ (0.82-multi.)
$x' := 0.75y^4 + 0.1u_1,$ $y' := 0.75x^4 + 0.1u_2,$ (over $X = \{(x,y) \mid x^2+y^2 \leq 1\}$)	$\mathcal{U}(-1,1)$	$0.78x^2 + 1.23xy + 0.78y^2$ (0.75-multi.)
$x' := 0.1(y(3x^2+2y^2-0.5) + u_1\sqrt{x^2+y^2}),$ $y' := 0.1(y(2x^2+4xy+3y^2-0.5) + u_2\sqrt{x^2+y^2}),$ (over $X = \{(x,y) \mid x^2+y^2 \leq 1\}$)	$\mathcal{U}(-\sqrt{3},\sqrt{3})$	$1.55x^2 + 2.36xy + 1.34y^2$ (0.5-multi.)

Table 5.2: The results of our implementation on the PSTS in Section A.2. Stochastic Systems summarizes the update dynamics of each system. Noise presents the type of stochastic noise: $\mathcal{N}(\mu,\sigma)$ - Gaussian noise with mean μ and standard deviation σ, $\mathcal{U}(a,b)$ - Uniform noise over the interval $[a,b]$. Supermartingale presents the inferred certificate $V(\mathbf{x})$.

Chapter 6

Quantitative Analysis

In Chapter 4 we presented a set of deductive proof rules for establishing qualitative (a.s.) reachability and repeated reachability properties. The soundness of the rules as well as their application requires finding suitable supermartingale ranking functions. In Chapter 5 we proposed two constraint-based generation techniques to automatically find supermartingale certificate functions. Chapter 7 present another automated technique based on abstract interpretation to generate a set of mutually inductive supermartingale expressions. The focus of the current chapter is the **quantitative** analysis of stochastic properties that martingales and supermartingales enable.

6.1 Concentration of Measure

Concentration of measure (COM) is a phenomenon in probability theory stating that under suitable conditions given a random variable X whose expected value $\mathbb{E}X$ exists, most of the probability measure μ_X of X is concentrated within a narrow range around $\mathbb{E}X$.

To best illustrate the phenomenon consider the simple example below.

Example 6.1.1 (CLT)**.** *Figure 6.1 presents a simple probabilistic program that aggregates the sum of* 500 *independent samples drawn from a real uniform distribution over the range* $[0, 1]$*. For this program a standard non-probabilistic static analysis [51, 123] may infer the following loop invariant:* $0 \leq \mathtt{x} \leq 501 \land \mathtt{i} = \mathtt{N}$ *at line* ℓ_3 *at the time when the loop is exited. This invariant (shaded pink in Figure 6.1*(RIGHT)*) accurately captures the set of* ***all*** *possible reachable states of the program. However, the value of* $\mathtt{x}$ *is clearly tightly clustered around* $\mathbb{E}(\mathtt{x}) = 250$*. In fact, in none of the* 500

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

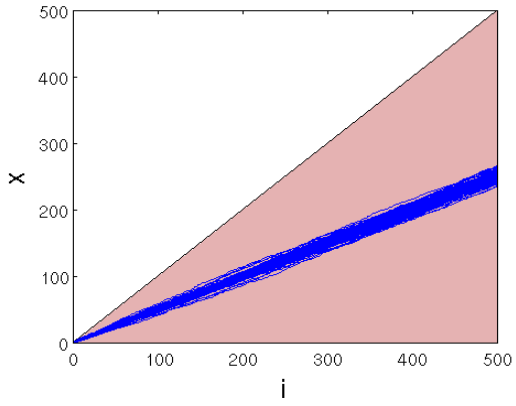


Figure 6.1: (LEFT) A simple probabilistic program that accumulates 500 iid uniform random draws on $[0, 1]$; (RIGHT) A graph of 10000 sample executions of the sum `X` versus the number of draws `i` of the probabilistic program (in blue); a standard non-probabilistic loop invariant for the same program (in pink).

simulations is the final value of $\mathtt{x} \notin [200, 300]$. *Using the approach we describe in this chapter, we can* ***guarantee*** $P(\mathtt{x} \in [200, 300]) \geq 0.84$.

An alternative perspective on Concentration of measure (COM) is that it provides bounds on the confidence of a prediction. In the context of Example 6.1.1, suppose one wants to make a guess about the value of `x` upon termination. Presented with the evidence in Figure 6.1, one may reasonably choose the expected value $\mu_{\mathtt{x}} = \mathbb{E}\mathtt{x}$ of the random quantity (in fact, this is the most likely outcome), or, to maximize the chance of their success, predict a value very close to $\mu_{\mathtt{x}}$, say, $[\mu_{\mathtt{x}} - 50, \mu_{\mathtt{x}} + 50]$.

One may then want to establish some guarantees on the quality of the prediction: for example, with confidence $\delta \in (0, 1)$ the predicted value is within $n = 50$ units around the expected value. That is, were the experiment repeated a large number of times, the prediction would be correct in δ fraction of the experiments. This means that the probability of deviating more than n units away from the expected value (our prediction) is upper-bounded by δ. In this sense COM provides a formal framework for relating the size of deviations and the level sets of possible probability distributions.

Concentration of Measure (COM) Inequalities. Concentration of measure is a well-

studied phenomenon [112, 39, 60, 21, 13] with a wealth of results that under suitable conditions provide an upper bound on the probability of deviations from the expected value of a random quantity. Most results are in the form of inequalities such as

$$P(X - \mathbb{E}X \geq t) \leq K(t), \quad \text{for a random variable } X\text{; or,}$$
$$P(X_n - X_0 \geq t) \leq K(t), \quad \text{for a stochastic process } \{X_n\}_{n=0}^{\infty},$$

where the probability bound K often decreases exponentially in t. The bound K also depends on the properties of the random variable including the existence of its finite moments.

Below are two of the earliest and well-known concentration of measure inequalities.

- (Markov Inequality) Let X be a nonnegative random variable, then

$$P(X \geq a) \leq \frac{\mathbb{E}(X)}{a}, \quad \text{for any } a > 0.$$

- (Chebyshev Inequality) Let X be a random variable, then for any $k > 0$,

$$P(|X - \mathbb{E}(X)| \geq k \cdot \sigma_X) \leq \frac{\sigma_X^2}{k^2},$$

  where $\sigma_X^2 \triangleq \mathbb{E}[(X - \mathbb{E}(X))^2]$ is the **variance** of X, and $\sigma_X \triangleq \sqrt{\sigma_x^2}$ is the standard deviation.

- (Chebyshev Higher-Moments) Chebyshev's Inequality above has been extended to higher moments as well. Let X be a random variable, then for any $k > 0$, $n \geq 2$,

$$P\left(|X - \mathbb{E}(X)| \geq k(m_n)^{\frac{1}{n}}\right) \leq \frac{1}{k^n},$$

  where $m_n \triangleq \mathbb{E}(|X - \mathbb{E}(X)|^n)$ is the n-th central moment of X.

The following example motivates our interest into COM for reachability properties.

Example 6.1.2. *Consider the empirical result of our motivating example from Chapter 4 reproduced for ease of reference in Figure 6.2. The histogram on the* (RIGHT) *presents* 10^6 *simulations of the probabilistic program in Figure 6.2* (LEFT)*. The mean value for* `count` *is* $\mathbb{E}(\hat{count}) = 5.38218$ *and its maximum value is* 21*. Although* `count` *spans a wide range* $[1, 21]$*, empirical estimates show*

```
1  int x = unifRand(-5,3);
2  int y = unifRand(-3,5);
3  int count = 0;
4  while (x + y <= 10) {
5     if flip(3/4){ //r_flip
6        x = x + unifRand(0,2);
7        y = y + 2;
8     }
9     count++;
10 }
11 //loop termination
```

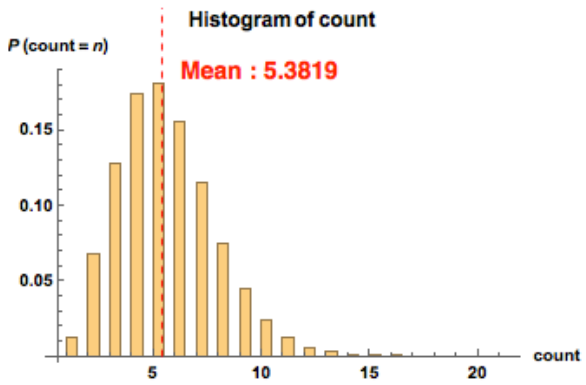


Figure 6.2: (LEFT) An example of a simple probabilistic program; (RIGHT) Histogram of the value of `count` for 10^6 simulations of the program.

that $\mathbb{P}(\texttt{count} \geq 13) = 0.004595$. *In fact, using statistical inference we can conclude that at the 99% confidence level the same probability lies in the interval* $0.008379 \pm 2.3 \times 10^{-4}$. *This means that over* 0.99 *of the probability measure associated with* `count` *at the end of the program is concentrated in the range* $[1, 12]$.

Applying Markov's Inequality we can prove that

$$\mathbb{P}(|count - \mathbb{E}(\hat{count})| \geq 6.6181) \leq \frac{\mathbb{E}(\hat{count})}{6.6181} = 0.8132.$$

This example presents an apparent disconnect between the strength of what is provable using concentration of measure inequalities and statistical results obtained in practice. Approaches that rely on providing statistical guarantees have gained much popularity because of their ease of use and the fact that they have precise dynamic (runtime) information available: all moments $m_0, m_1, \ldots$ of the distributions are known (or at least their empirical estimates: $\hat{m}_0, \hat{m}_1, \ldots$).

The imprecision of COM inequalities is due to the fact they were developed to be highly applicable and usually rely only on the first two moments as available information (in Example 6.1.2, $\mathbb{E}(\hat{count})$ only). Had variance information ($\hat{m}_2 : \sigma^2 = 5.03697$) been available Chebyshev's inequality can be used to prove a stronger result:

$$\mathbb{P}(|count - \mathbb{E}(\hat{count})| \geq 6.6182) \leq \frac{\sigma^2}{(6.6182)^2} = 0.119.$$

While information about the expected value of program variables may be easy to compute statically,

variance information may not be.

6.2 Martingales and COM: Azuma-Hoeffding Inequalities

Fortunately, strong COM results have been proved for **large deviations** of martingales.

Theorem 6.2.1 (Hoeffding Inequality [60])**.** Let $\mathcal{X}$ be a bounded martingale such that $a_i \leq X_i - X_{i-1} \leq b_i$, for all $i \geq 0$. Then,

$$Pr(X_n > X_0 + t), Pr(X_n < X_0 - t) \leq \exp\left(-\frac{t^2}{2\sum_{i=1}^{n}(b_i - a_i)^2}\right).$$

Theorem 6.2.2 (Azuma Inequality [60])**.** Let $\mathcal{X}$ be a bounded martingale such that $-c_i \leq X_i - X_{i-1} \leq c_i$, for all $i \geq 0$. Then,

$$Pr(X_n > X_0 + t), Pr(X_n < X_0 - t) \leq \exp\left(-\frac{t^2}{2\sum_{i=1}^{n} c_i^2}\right).$$

Notice that if we let $c_i = \max(|a_i|, |b_i|)$ then $-c_i \leq a_i \leq b_i \leq c_i$. This means that in the asymmetric case when $a_i \neq -b_i$, the Hoeffding bounds could always be "relaxed" to the Azuma bounds and that they are strictly better otherwise. This is why these inequalities often go together under the name Azuma-Hoeffding inequality.

Example 6.2.1. *Returning to the example above, we are interested in the probability of a large deviation of* `count` *such as* $\mathbb{P}(count \geq 21)$ *over the runs of length up to 21.*[1] *Facts at our disposal are* $\texttt{4x} + \texttt{4y} - \texttt{9count}$ *is a martingale expression and so is* $M[\texttt{x}] : \texttt{x} + \texttt{y} - \texttt{2.25count}$. *For the martingale* $\{M_n\}_{n=0}^{\infty}$ *induced over the runs of the program we know* $M_0 \geq -8$ *and that along any iteration* i *of the loop,* $a \leq M_i - M_{i-1} \leq b$ *with* $a = -2.25$, $b = 1.75$. *We are interested in probability of a deviation of* `count` $\mathbb{P}(\texttt{count}_n \geq 21)$ *for* $n = 21$. *This implies that at iteration* n, $x + y \leq 10$ *still holds.*

We outline below how the probability of the tail event $P(\texttt{count}_n \geq 21)$ *can be upper bounded*

[1] Technically, this analysis is performed over the stopped version of the stochastic process $\mathcal{M}$ that freezes the value of `count` whenever $x + y > 10$ is satisfied for the first time.

by the deviation of a martingale using the Azuma-Hoeffding inequalities:

$$\begin{aligned}
\texttt{count}_n \geq 21 \quad &\iff \quad -2.25 \cdot \texttt{count}_n \leq -47.25, && (\textit{arithmetic}) \\
\texttt{x}_n + \texttt{y}_n - 2.25 \cdot \texttt{count}_n &\leq \texttt{x}_n + \texttt{y}_n - 47.25, && (\textit{arithmetic}) \\
M_n &\leq 10 - 47.25, && (\textit{using } x + y \leq 10) \\
M_n - M_0 &\leq 8 - 37.25 = -29.25.
\end{aligned}$$

This means that if we let $t = 29.25$*,* $n = 21$*, and* $c_i = -a_i$ *for all* $i \geq 0$*,*

$$\mathbb{P}(M_n \leq M_0 - t) \leq \exp\left(-\frac{t^2}{2\sum_{i=1}^{n} c_i^2}\right)$$

is equivalent to

$$\mathbb{P}(\texttt{count}_n \geq 21) \leq \exp\left(-\frac{(10 + M_0 - \frac{9}{4}count)^2}{2(n)(c_i)^2}\right) = \exp\left(-\frac{29.25^2}{2(21)(2.25)^2}\right) = 0.01788.$$

6.2.1 Square Integrable Martingales*

In this section we present some recent results from Bercu et al. [13] that strengthen Azuma-Hoeffding type inequalities by requiring a finite second moment. For the purposes of this section **only**, we assume that the second moment of any martingale expressions $\texttt{e}$ exist and are finite $\mathbb{E}[\texttt{e}_n^2] < \infty$. We defer discussion on how to establish until after the results.

Definition 6.2.1. *A martingale* $\{M_n\}_{n=0}^{\infty}$ *is* ***square integrable*** *if for all* $n \geq 0$*,* $\mathbb{E}[M_n^2] < \infty$*. The* ***increasing process*** *of* $\{M_n\}$ *is defined by* $\langle M \rangle_0 = 0$ *and for all* $n \geq 1$*,*

$$\langle M \rangle_n \triangleq \sum_{k=1}^{n} \mathbb{E}[(M_k - M_{k-1})^2 | \mathcal{F}_{k-1}]$$

and we let $\Delta M_k = M_k - M_{k-1}$ *and the* ***variance process***

$$V_k = \langle M \rangle_k - \langle M \rangle_{k-1} = \mathbb{E}[(M_n - M_{n-1})^2 \,|\, \mathcal{F}_{n-1}].$$

Theorem 6.2.3 (Theorem [13, 3.1])**.** Let $\{M_n\}$ be a square integrable martingale with $M_k - M_{k-1} \leq B_k$ and let $\{V_n\}$ be the corresponding variance process. For any positive t,

$$P(M_n \geq t) \leq \exp\left(-\frac{t^2}{\|\mathscr{A}_n\|_\infty}\right)$$

where

$$\mathscr{A}_n \triangleq \sum_{k=1}^{n} {B_k}^2 \Phi\left(\frac{V_k}{B_k^2}\right) \quad \text{and} \quad \Phi(v) = \begin{cases} \frac{1-v^2}{|\log(v)|} & \text{if } v < 1, \\ 2v & \text{if } v \geq 1. \end{cases}$$

Theorem 6.2.4 (Theorem [13, 3.4])**.** Let $\{M_n\}$ be a square integrable martingale with $|M_k - M_{k-1}| \leq B_k$ a.s. and let $\langle M \rangle_n$ be the corresponding increasing process. For any positive t,

$$P(M_n \geq t) \leq \exp\left(-\frac{3t^2}{\|5\langle M\rangle_n + \mathscr{B}_n\|_\infty}\right) \leq \exp\left(-\frac{t^2}{2\|\mathscr{B}_n\|_\infty}\right)$$

where

$$\mathscr{B}_n \triangleq \sum_{k=1}^{n} {B_k}^2.$$

Notice that the last term above represents the Azuma-Hoeffding inequality and, therefore, presents an improvement over Theorem 6.2.2 in the case when M_n is bounded symmetrically.

Theorem 6.2.5 (Theorem [13, 3.6])**.** Let $\{M_n\}$ be a square integrable martingale with $A_k \leq M_k - M_{k-1} \leq B_k$, for all $k \geq 0$, for some negative random variables A_k and nonnegative random variables B_k. Let $\langle M \rangle_n$ be the corresponding increasing process. For any positive t,

$$P(M_n \geq t) \leq \exp\left(-\frac{3t^2}{\|2\langle M\rangle_n + \mathscr{D}_n\|_\infty}\right)$$

where

$$\mathscr{D}_n \triangleq \sum_{k=1}^{n} (B_k - A_k)^2.$$

The result in Theorem 6.2.5 is shown to be an improvement over the Hoeffding bound.

Remark 2. For a martingale expression $M_n[X]$, it may be highly non-trivial to (statically) establish whether $\mathbb{E}[M_n^2] < \infty$ holds. In Section 6.2.1 we assumed that moments of martingale expressions exist and are finite to demonstrate and underscore the strength of the applicable COM results. In practice most examples of stochastic processes in the literature with $\mathbb{E}[M_n] < \infty$ but $\mathbb{E}[M_n^2] = \infty$ rely on: (i) stochastic dynamics that perform unbounded jumps; (ii) carefully crafted probability distributions on every step that depend on the current state of the system. To alleviate (i), one may consider expressions that satisfy the bounded increase, or general compactness, properties.

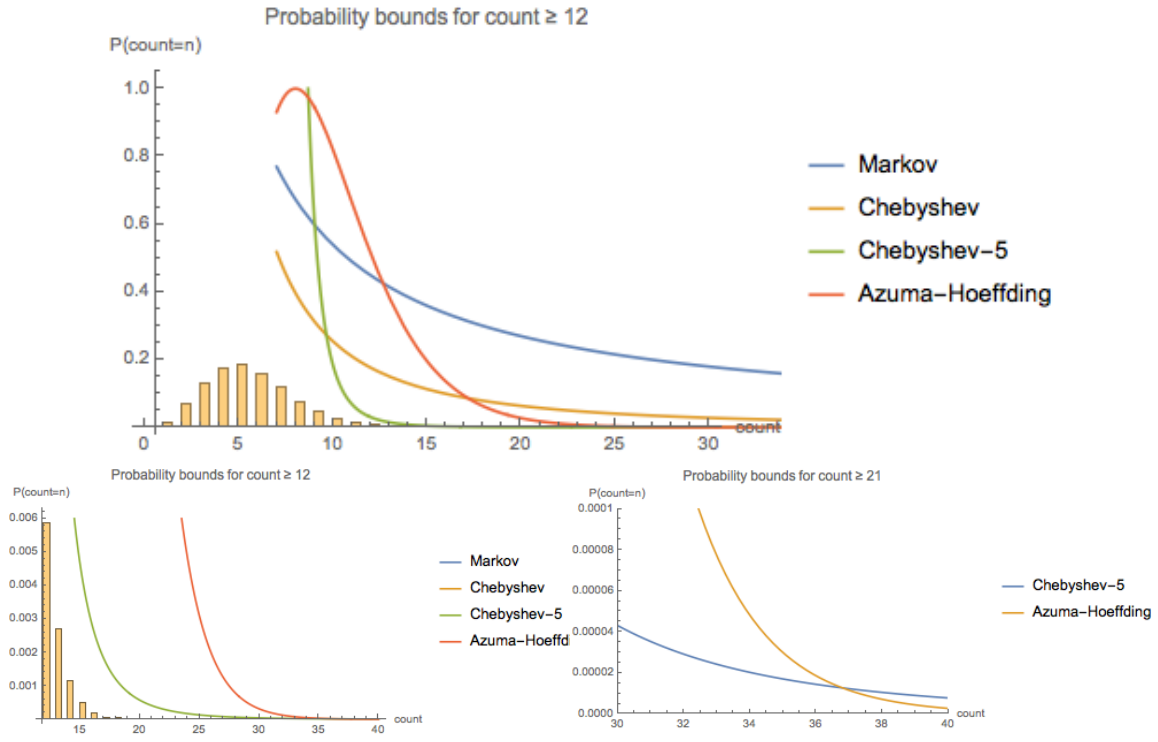


Figure 6.3: (TOP) A histogram of the probability of $P(\texttt{count} = n)$ at the end of execution and a comparison of the probability bounds: `Markov` - Markov Inequality; `Chebyshev` - Chebyshev Inequality; `Chebyshev-5` - Chebyshev Fifth-Moment Inequality using $\hat{m}_5$; `expAzumaHoeffding` - Azuma-Hoeffding Inequality of Theorem 6.2.1. (LEFT) Rescaled version to directly compare `Azuma-Hoeffding` and `Chebyshev-5`. (RIGHT) Rescaled version to directly compare `Azuma-Hoeffding` and `Chebyshev-5` on a large deviation $P(\texttt{count}_n) \geq 30$.

To address (*ii*), one may consider the class of PSTS without demonic nondeterminism and with constant fork probabilities. We speculate that such a restriction of the operational model may constitute a class of stochastic systems for which sufficient conditions to establish square integrability exist. The importance of the COM results, however, requires a careful formal treatment of this topic. We reserve as future work finding sufficient conditions to establish square integrability.

We compare the tightness of the Azuma-Hoeffding inequality to the Markov and Chebyshev ones when applied to Example 6.2.1.

Example 6.2.2. *Figure 6.3 present a comparison of the Azuma-Hoeffding bounds and the standard bounds that rely on the empirical estimates of the moments. Notice that Chebyshev Higher-Moment inequality with estimate $\hat{m}_5$ provides a very tight upper bound on the empirical probability of* $\texttt{count} \geq T$ *for* $T \geq 11$*. This is to be expected as in incorporates strong runtime results. However, the bound*

that this inequality provides decreases only polynomially in the size of the deviation $|X - \mathbb{E}(X)|$. *Eventually, all exponential bounds overtake this inequality as demonstrated in Figure 6.3* (LEFT, RIGHT) *for* `Azuma-Hoeffding`.

Conclusion. In general, we conclude that concentration of measure inequalities provide a formal framework for bounding the probability of rare events in the form of large deviations. For such events statistical methods based on empirical estimates and confidence intervals (i.e., hypothesis testing) fail to provide satisfactory results. Martingale probability bound inequalities such as the Azuma-Hoeffding inequality (and those based on square integrable martingales) significantly improve upon general COM results by incorporating the properties of the underlying stochastic processes.

6.3 Expected Values and Concentration Results of Hitting Times

In Chapter 4 we proved that additive and multiplicative supermartingales almost surely converge. We carefully used these convergence results to prove almost sure reachability of a target region T using the notion of stopping (or hitting) times. In this section we formalize the notion of the expected time to fulfill such a property.

Let Π be a PSTS and T be a target set of states. Let V be the corresponding c-additive supermartingale certificate according to rule REACH-ADD, for some $c > 0$, and $\varepsilon > 0$ as the target level set for reaching T. Let $\{V_n\}_{n=0}^{\infty}$ be the c-additive supermartingale over the executions of Π with $\mathbb{E}(V_0) < \infty$. Let $\tau_\varepsilon \triangleq \{\min n \in \mathbb{N} \,|\, V_n \leq \varepsilon\}$ be the hitting time of T.

In the proof of Theorem 4.3.3 we showed that $\tau_\varepsilon < \infty$ a.s.; hence, $\mathbb{E}(\tau_\varepsilon) < \infty$. Our goal here is to derive an upper bound on the expectation.

Lemma 6.3.1. Let V, ε be the certificate and level set bound according to REACH-ADD. Let τ_ε be the hitting time of T, then

$$\mathbb{E}(\tau_\varepsilon) \leq \frac{\mathbb{E}(V_0) - \mathbb{E}(\widetilde{V})}{c},$$

where $\widetilde{V}$ is the limit random variable $V_n \to V$ a.s.

Proof. First, we prove by induction that for all $n \geq 1$, $\mathbb{E}(V_n) \leq \mathbb{E}(V_0) - c\sum_{i=0}^{n} P(V_i \geq \varepsilon)$.

$$\begin{aligned}
\mathbb{E}(V_{n+1}) &= \mathbb{E}\left(\mathbb{E}(V_{n+1} \mid \mathcal{F}_n)\right), && \text{by the law of total expectation} \\
&\leq \mathbb{E}\left(\mathbb{E}(V_n) - c\mathbb{1}_{\{V_n \geq \varepsilon\}}\right), && \text{by definition, drift outside (p4)} \\
&= \mathbb{E}(V_n) - c\mathbb{E}(\mathbb{1}_{\{V_n \geq \varepsilon\}}), && \text{by linearity of expectation} \\
&\leq \mathbb{E}(V_0) - c \cdot \textstyle\sum_{i=0}^{n-1} P(V_i \geq \varepsilon) - cP(V_n \geq \varepsilon), && \text{by the IH} \\
&\leq \mathbb{E}(V_0) - c \cdot \textstyle\sum_{i=0}^{n} P(V_i \geq \varepsilon).
\end{aligned}$$

Next, observe that by Theorem 2.2.2, we know that $V_n \to \widetilde{V}$ almost surely. This means that

$$\lim_{n\to\infty} P(V_n \geq \varepsilon) = 0 \text{ and } \tau_\varepsilon < \infty \text{ almost surely.}$$

Then

$$\begin{aligned}
\mathbb{E}(\tau_\varepsilon) = \sum_{i=0}^{\infty} P(\tau_\varepsilon \geq i) &= \sum_{i=0}^{\infty} P(V_i \geq \varepsilon) \\
&= \left(\frac{1}{c}\right)\left(c \cdot \sum_{i=0}^{\infty} P(V_i \geq \varepsilon)\right) \\
&\leq \left(\frac{1}{c}\right)\left(\mathbb{E}(V_0) - \mathbb{E}(M)\right) \\
&\leq \frac{\mathbb{E}(V_0) - \mathbb{E}(\widetilde{V})}{c}.
\end{aligned}$$

□

This result is similar to the ones in Ferrer et al. [67, Lemma 5.5] (where the proof first appeared) and Chatterjee et al. [36, Proposition 1].

Example 6.3.1. *Consider the "The tortoise and hare" program of Example 5.0.1. Using the constraint synthesis technique of Chapter 6 we were able to infer the* ***super martingale*** *expression* $\mathtt{t} - \mathtt{h} + 9$. *Notice that initially* $\mathtt{t} = 30, \mathtt{h} = 0$*; therefore,* $(\mathtt{t} - \mathtt{h} + 9)_0 = 39$. *On every step the value of this expression decreases in expectation by* $c = 1.5$. *It satisfies the conditions of* REACH-ADD *with* $\varepsilon = 9$ *and so the system terminates almost-surely in expected at most* $\mathbb{E}(\tau_\varepsilon) \leq \frac{39-9}{1.5} = 20$ *steps.*

Let V be the α-multiplicative supermartingale certificate according to rule REACH-GEOM, for some $\alpha \in (0, 1)$, and $\varepsilon > 0$ as the target level set for reaching T. Let $\{V_n\}_{n=0}^{\infty}$ be the α-multiplicative

supermartingale over the executions of Π with $\mathbb{E}(V_0) < \infty$. Let $\tau_\varepsilon \triangleq \{\min n \in \mathbb{N} \,|\, V_n \leq \varepsilon\}$ be the hitting time of T.

Lemma 6.3.2. Let V, ε be the certificate and levelset bound according to REACH-GEOM. Let τ_ε be the hitting time of T, then

$$\mathbb{E}(\tau_\varepsilon) \leq \frac{\mathbb{E}(V_0)}{c},$$

where $c = \min(1-\alpha, \alpha)$.

Proof. First, we prove by induction that for all $n \geq 1$,

$$\mathbb{E}(V_n) \leq \mathbb{E}(V_0) - c\sum_{i=0}^{n-1} P(V_i \geq \varepsilon),$$

where $c = \min(1-\alpha, \alpha)$.

Base Case: Let $n = 0$, then $\mathbb{E}(V_0) \leq \mathbb{E}(V_0)$.

Inductive Step:

$$\begin{aligned}
\mathbb{E}(V_{n+1}) &= \mathbb{E}(\mathbb{E}(V_{n+1} \,|\, \mathcal{F}_n)), && \text{by the law of total expectation} \\
&\leq \mathbb{E}\left(\alpha V_n\right), && \text{by definition} \\
&= \mathbb{E}\left(V_n - (1-\alpha)V_n\right), && \text{arithmetic} \\
&\leq \mathbb{E}(V_n) - \beta\mathbb{E}(V_n), && \text{by lin. of expectation; } \beta = \min(1-\alpha, \alpha) \\
&= \mathbb{E}(V_n) - \beta\mathbb{E}(V_n \mathbb{1}_{\{V_n \leq \varepsilon\}} + V_n \mathbb{1}_{\{V_n \geq \varepsilon\}}), && \text{split } S \text{ into } S \setminus T \text{ and } T,\ V_n \geq 0 \\
&\leq \mathbb{E}(V_n) - \beta\mathbb{E}(\varepsilon \mathbb{1}_{\{V_n \geq \varepsilon\}}), && \text{arithmetic}, \beta V_n \mathbb{1}_{\{V_n \geq c\}} \geq 0 \\
&= \mathbb{E}(V_n) - \beta\varepsilon\mathbb{E}(\mathbb{1}_{\{V_n \geq \varepsilon\}}), && \text{by linearity of expectation, } c = \beta\varepsilon \\
&\leq \mathbb{E}(V_0) - c\textstyle\sum_{i=0}^{n-1} P(V_i \geq \varepsilon) - cP(V_n \geq \varepsilon), && \text{by the IH} \\
&\leq \mathbb{E}(V_0) - c\textstyle\sum_{i=0}^{n} P(V_i \geq \varepsilon).
\end{aligned}$$

Next, by Theorem 4.3.1 we know $X_n \to 0$ almost surely. This means that

$$\lim_{n\to\infty} P(V_n \geq \varepsilon) = 0 \text{ and } \tau_\varepsilon < \infty \text{ almost surely.}$$

Finally, since $\mathit{expect}(V_n) = \sum_{i=0}^{\infty} \mathbb{1}_{\{V_n \geq \varepsilon\}} \to 0$ as $n \to 0$.

$$\mathbb{E}(\tau_\varepsilon) = \sum_{i=0}^{\infty} P(\tau_\varepsilon \geq i) = \sum_{i=0}^{\infty} P(V_i \geq \varepsilon) \leq \frac{\mathbb{E}(V_0)}{c}.$$

□

6.4 Additional Applications

CLT. For the program in Example 6.1.1 we inferred $2\mathtt{x} - \mathtt{i}$ is a martingale. Its change at any step is bounded by ± 1. The initial value of the expression is: $(2\mathtt{x} - \mathtt{i})_0 = 0$. We also know that and $\mathtt{i}_{500} = 500$. Therefore, if we choose $t = 50$, we conclude that after $N = 500$ steps,

$$P(|(2\mathtt{x} - \mathtt{i})_{500} - (2\mathtt{x} - \mathtt{i})_0| \geq 50) \leq 2\exp\left(-\frac{2500}{2 \times 500 \times 1}\right) \leq 0.16.$$

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 idea that allows us to infer bounds tight bounds is to consider deviations "proportional" to the number of steps: $|M_n - M_0| \geq a\sqrt{n}$, for constants $a > 0$ and $n \geq 0$. For $n > 0$, we conclude the bounds

$$P(|(2\mathtt{x} - \mathtt{i})_n - (2\mathtt{x} - \mathtt{i})_0| \geq a\sqrt{n}) \leq 2\exp(-\frac{a^2 n}{2n}) \leq 2\exp(-\frac{a^2}{2}).$$

For $a = 3$, the upper bound is 0.0223.

Robot Dead Reckoning. In Section 5.4.1 we found that expressions $\mathtt{x} - \mathtt{estX}$ and $\mathtt{y} - \mathtt{estY}$ are martingales at the loop head of the program in Figure 5.2 (LEFT). In each case the absolute change in the martingales is bounded by 0.05. Given the initial difference of 0 between the values, we infer using Azuma-Hoeffding theorem that $P(|\mathtt{x} - \mathtt{estX}| \geq 3) \leq 1.5 \times 10^{-3}$. In contrast, a worst-case analysis concludes that $|\mathtt{x} - \mathtt{estX}| \leq 0.05 * 500 \leq 25$. The analysis for $\mathtt{y} - \mathtt{estY}$ yields identical results.

Chapter 7

Inductive Expectation Invariants

In Chapter 4 we showed that supermartingale expressions induce supermartingale stochastic processes when evaluated along the sample runs of a polynomial stochastic transition system. In Chapter 5 we then presented a constraint based procedure to infer supermartingale expressions.

In the current we present a generalization of the idea that martingale expressions act as **expectation invariants**. An expectation invariant is a program expression `e` whose expectation *at any given iteration of the loop* exists and is always nonnegative.

First, we formally define this notion and prove using mathematical induction that a given program expression is invariant. This allows us to formulate an induction based proof rule to lift the notion of expectation invariants to a set of expressions that together acts as an inductive invariant over the expected values of expressions. We present an abstract interpretation based procedure that mechanizes the generation of conic inductive expectation invariants.

7.1 Expectation Invariants

Expectation invariants are invariant inequalities on the expected value of program expressions. Therefore, one could view the set of possible state distributions $\mathcal{D}_i$ at step i as the concrete domain over which our analysis operates to produce the abstract facts in the form of **expectation invariants** over these distributions. We formalize the notion of expectation invariants and derive a fixed point characterization of expectation invariants in the next section.

7.1.1 Definitions and Examples

In this chapter we focus on probabilistic programs with loops that can be converted into a PSTS with a single location representing the loop head and force all transitions $\mathcal{T}$ to be self-loops. This simplifies our transitions to $\tau : \langle \mathbf{g}, f_\tau \rangle$ where $\mathbf{g}$ is the transition guard (previously φ_τ) and $f_\tau : \{f_1, \ldots, f_k\}$ is the transition update function with forks $f_1, \ldots, f_k$. An execution step starts at and returns to the loop head; therefore, $\mathcal{D}_n$ is now the distribution (at the n-th step) over the reachable states in X.

We refer to such PSTS as **probabilistic loops** $\mathcal{P} : \langle \mathcal{T}, \mathcal{D}_0, n \rangle$, where n is the value of the loop counter. When $\mathcal{P}$ is clear from the context we write $\texttt{pre}\mathbb{E}(\texttt{e})$ to denote the pre-expectation expression corresponding to e and if we want to emphasize the distribution over which the expected value is taken we write $\texttt{pre}\mathbb{E}(\mathcal{D}_n, \texttt{e})$.

Example 7.1.1. *Below is our motivating example probabilistic program with the corresponding probabilistic loop. To eliminate the final location, we introduce a* ***stuttering*** *transition which preserves the values of the program variables when the loop guard* $(x + y > 10)$ *is violated.*

```
real x := rand(-5,3)
real y := rand(-3,5)
int count := 0
while (x+y <= 10)
    if flip(3/4)
      x := x + rand(0,2)
      y := y + 2
    count++
```

```
real x := rand(-5,3)
real y := rand(-3,5)
int count := 0
while (forever)
  if (x + y <= 10)
    if flip(3/4)
       x := x + rand(0,2)
       y := y + 2
    count++
  else // if (x + y > 10)
    // Preserve x,y,count
```

τ_1 (loop body)

$$\mathbf{g}_1 \;:\; (x + y \leq 10)$$

$$f_{\tau_1} \;:\; \begin{cases} f_1 : \begin{bmatrix} \mathtt{x}' & \mapsto & \mathtt{x} + r_1, \\ \mathtt{y}' & \mapsto & \mathtt{y} + 2, \\ \mathtt{count}' & \mapsto & \mathtt{count} + 1, \end{bmatrix} & \text{w.p. } \frac{3}{4} \\ f_2 : \begin{bmatrix} \mathtt{x}' & \mapsto & \mathtt{x}, \\ \mathtt{y}' & \mapsto & \mathtt{y}, \\ \mathtt{count}' & \mapsto & \mathtt{count} + 1, \end{bmatrix} & \text{w.p. } \frac{1}{4} \end{cases}$$

τ_2 (stuttering)

$$\mathbf{g}_2 \;:\; (x + y > 10)$$

$$f_{\tau_2} \;:\; \begin{cases} \mathtt{x}' & \mapsto & \mathtt{x}, \\ \mathtt{y}' & \mapsto & \mathtt{y}, \\ \mathtt{count}' & \mapsto & \mathtt{count}, \end{cases}$$

Let $\mathcal{P} : \langle \mathcal{T}, \mathcal{D}_0, n \rangle$ be a probabilistic loop and let $\langle \mathbf{x}_0, 0 \rangle$ be the initial state of the system. From Section 4.1 we know that $\mathbf{x}_0$ is drawn from an initial distribution $\mathcal{D}_0$ and that any n-step sample execution of $\mathcal{P}$ defines a sample trajectory through the distributions of reachable states $\mathcal{D}_0, \ldots, \mathcal{D}_n$ at step i for any $0 \leq i \leq n$. We then define the **expectation** of a program expression $\mathtt{e}$ **at time step** n as $\mathbb{E}(\mathtt{e} \,|\, n) \triangleq \mathbb{E}_{\mathcal{D}_n}(\mathtt{e})$.

Notation: We denote the expectation of an expression $\mathtt{e}$ over the program variables at the j^{th} step as $\mathbb{E}(\mathtt{e} \,|\, n = j)$ or equivalently $\mathbb{E}_{\mathcal{D}_j}(\mathtt{e})$. Unless otherwise mentioned, $\mathtt{e}$ will denote an affine (or linear) expression over the program variables.

Definition 7.1.1 (Expectation Invariants)**.** *An expression* $\mathtt{e}$ *over the program variables* X *is called an* ***expectation invariant*** *(EI) if and only if for all* $n \geq 0$, $\mathbb{E}(\mathtt{e} \,|\, n) \geq 0$.

Thus, expectation invariants are program expressions whose expectations over the initial distribution are nonnegative, and under **any** number $n \geq 0$ of iterations of the probabilistic loop remain non-negative.

Example 7.1.2. *Consider the program from Example 7.1.1, and the expression* $\mathtt{y} - \mathtt{x}$*. Initially,* $\mathbb{E}(\mathtt{y} - \mathtt{x} \,|\, 0) = \mathbb{E}_{\mathcal{D}_0}(\mathtt{y} - \mathtt{x}) = 1 - (-1) = 2 \geq 0$*. We can show that* $\mathbb{E}(\mathtt{y} - \mathtt{x} \,|\, i) = \mathbb{E}(\mathtt{y} \,|\, i) - \mathbb{E}(\mathtt{x} \,|\, i) \geq 0$ *at any step* i*. Therefore,* $\mathtt{y} - \mathtt{x}$ *is an expectation invariant.*

7.1.2 Martingales and Expectation Invariants

Expectation invariants as given by Definition 7.1.1 are closely related to the concept of (super-, sub-) **martingales**. In fact, martingales naturally yield expectation invariants.

Lemma 7.1.1. For every supermartingale expression $\mathtt{e}$ the expression $e_0 - \mathtt{e}$ is an expectation invariant, wherein $e_0 = \mathbb{E}(\mathtt{e} \,|\, n = 0)$.

Proof. First, we observe that for all $\mathbf{x}$, $\mathtt{pre}\mathbb{E}(\mathtt{e}) \leq \mathtt{e}$. Therefore, assuming that the expectations on both sides exist, we have for all $n \geq 0$, $\mathtt{pre}\mathbb{E}(\mathcal{D}_n, \mathtt{e}) \leq \mathbb{E}_{\mathcal{D}_n}(\mathtt{e})$.

We now prove that $\mathbb{E}(e_0 - \mathtt{e} \,|\, n) \geq 0$ for all $n \in \mathbb{N}$ by induction. Clearly, for $n = 0$, the statement holds. Furthermore, assume that $\mathbb{E}(e_0 - \mathtt{e} \,|\, n = j)$ holds, we obtain

$$\begin{aligned}\mathbb{E}(e_0 - \mathtt{e} \,|\, n = j+1) &= \mathbb{E}(e_0 - \mathtt{pre}\mathbb{E}(\mathtt{e}) \,|\, n = j) \\ &= e_0 - \mathbb{E}(\mathtt{pre}\mathbb{E}(\mathtt{e}) \,|\, n = j) \\ &\geq e_0 - \mathbb{E}(\mathtt{e} \,|\, n = j) \geq 0.\end{aligned}$$

□

However, expectation invariants can arise without martingales, as shown by the following simple example that repeatedly swaps two variables $\mathtt{x}, \mathtt{y}$: Notice that expressions $\mathtt{x}$ and $\mathtt{y}$ are

```
real x := rand(0, 5)
real y := rand(4, 7)
while (true)
    x := y + unifRand(-1,1)
    y := x + unifRand(-2,2)
```

expectation invariant. However, they are not martingales. In fact, to prove that $\mathbb{E}(\mathtt{x}) \geq 0$, at any step, we require that $\mathbb{E}(\mathtt{y}) \geq 0$ at the previous step.

Therefore, the notion of expectation invariants subsumes that of martingales as defined here. Drawing analogies to the familiar case of Floyd-Hoare invariants, martingales correspond to assertions which are invariant by themselves, whereas expectation invariants are analogous to the general case of **mutually inductive invariants** [117].

7.1.2.1 Proving Expectation Invariance

We now focus on the question of proving that a given expression $\mathtt{e}$ over the program variables is an expectation invariant. This requires constructing (approximations) to the distribution $\mathcal{D}_n$ for each n, or alternatively, an argument based on mathematical induction. We first observe an important property of each $\mathcal{D}_n$.

Definition 7.1.2 (Admissible Distribution)**.** *We say that a distribution $\mathcal{D}$ over the state-space $\mathcal{X}$ is **admissible** if all moments exist.*[1] *In other words, for any polynomial $p(\mathbf{x})$ over the program variables, $\mathbb{E}_{\mathcal{D}}(p(\mathbf{x}))$ exists, and is finite.*

Lemma 7.1.2. For any polynomial expression $\mathtt{e}(\mathbf{x})$ over the program variables, and any $n \in \mathbb{N}$, $\mathbb{E}_{\mathcal{D}_n}(\mathtt{e})$ exists (and is finite).

Let us assume that any program $\mathcal{P}$ which we attempt to analyze is such that

(1) $\mathcal{D}_0$, the initial state distribution, is admissible;

(2) For each transition τ, the distribution of the random variables $\mathcal{D}_R$ is admissible.

Under these assumptions, we invoke Lemma 7.1.2 to conclude that $\mathcal{D}_n$ is admissible for each $n \geq 0$. However, rather than construct $\mathcal{D}_n$ explicitly for each n (which can be impractical), we formulate the principle of inductive expectation invariants. Consider expressions $E = \{\mathtt{e}_1, \ldots, \mathtt{e}_m\}$ wherein each $\mathtt{e}_i$ is a linear (or polynomial) expression involving the program variables.

Definition 7.1.3 (Inductive Expectation Invariants)**.** *The set E of expressions forms an inductive expectation invariant of the program $\mathcal{P}$ if and only if for each $\mathtt{e}_j$, $j \in [1, m]$,*

(1) $\mathbb{E}_{\mathcal{D}_0}(\mathtt{e}_j) \geq 0$, i.e., the expectation at the initial step is non-negative.

(2) For every admissible distribution $\mathcal{D}$ over the state-space $\mathcal{X}$,

$$(\mathbb{E}_{\mathcal{D}}(\mathtt{e}_1) \geq 0 \ \wedge \ \cdots \ \wedge \ \mathbb{E}_{\mathcal{D}}(\mathtt{e}_m) \geq 0) \ \models \ \mathtt{pre}\mathbb{E}(\mathcal{D}, \mathtt{e}_j) \geq 0\,. \tag{7.1}$$

The inductive expectation invariant principle stated above follows the standard Floyd-Hoare approach of "abstracting away" the distribution at the n^{th} step by the inductive invariant itself, and using these to show that the invariant continues to hold for one more step. Furthermore, it abstracts away from a specific $\mathcal{D}_n$ to any admissible distribution $\mathcal{D}$.

[1] While the existence of only the first moment suffices, our experiments demonstrate that our current synthesis approach can be extended to polynomial expectation invariants.

Theorem 7.1.1. Let $E : \{\mathtt{e}_1, \ldots, \mathtt{e}_m\}$ be inductive expectation invariants (Definition 7.1.3), it follows that each $\mathtt{e}_j \in E$ is an expectation invariant of the program:

$$\forall n \in \mathbb{N}, \quad \mathbb{E}(\mathtt{e}_j \,|\, n) \geq 0$$

Proof. The proof uses the important fact that each distribution $\mathcal{D}_n$ is admissible. We prove by simultaneous induction that

$$\bigwedge_{j=1}^{m} \mathbb{E}(\mathtt{e}_j \,|\, n) \geq 0\,.$$

Base-Case: The base case for $n = 0$ follows from item (1) of Definition 7.1.3.

Induction Step: Let us assume that the required statement holds for n and attempt to show for $n + 1$. Using Equation (7.1), and the admissibility of $\mathcal{D}_n$, we note that for each $j \in [1, m]$,

$$(\mathbb{E}_{\mathcal{D}_n}(\mathtt{e}_1) \geq 0 \ \wedge \ \cdots \ \wedge \ \mathbb{E}_{\mathcal{D}_n}(\mathtt{e}_m) \geq 0) \ \models \ \mathtt{pre}\mathbb{E}(\mathcal{D}_n, \mathtt{e}_j) \geq 0$$

Therefore, we conclude that $\mathtt{pre}\mathbb{E}(\mathcal{D}_n, \mathtt{e}_j) \geq 0$. Since, $\mathbb{E}_{\mathcal{D}_{n+1}}(\mathtt{e}_j) = \mathtt{pre}\mathbb{E}(\mathcal{D}_n, \mathtt{e}_j)$, we conclude that $\mathbb{E}_{\mathcal{D}_{n+1}}(\mathtt{e}_j) \geq 0$ for each j. Thus, the induction step is proven. □

However, Definition 7.1.3 is quite unwieldy, in practice, since the quantification over **all possible admissible distributions** $\mathcal{D}$ over the state space $\mathcal{X}$ is a higher order quantifier (over probability spaces and measurable functions). Rather than reason with this quantifier, we will use the following facts about expectations to formulate a new principle:

Theorem 7.1.2 (Facts About Expectations over Admissible Distributions)**.** The following hold over all possible admissible distributions $\mathcal{D}$ over a σ-algebra $\mathcal{X}$, linear assertion φ, and linear (or polynomial expressions) $\mathtt{e}, \mathtt{e}_1, \ldots, \mathtt{e}_k$:

(1) Linearity of expectation: $\mathbb{E}_{\mathcal{D}}(\lambda_1 \mathtt{e}_1 + \ldots + \lambda_k \mathtt{e}_k) = \lambda_1 \mathbb{E}_{\mathcal{D}}(\mathtt{e}_1) + \cdots + \lambda_k \mathbb{E}_{\mathcal{D}}(\mathtt{e}_k)$, for $\lambda_i \in \mathbb{R}$.

(2) If $\varphi \models \mathtt{e} \geq 0$ then $\mathbb{E}_{\mathcal{D}}(\mathbb{1}_{\varphi} \times \mathtt{e}) \geq 0$, provided $[\![\varphi]\!]$ is measurable. Specifically,

$$\mathbb{E}_{\mathcal{D}}(\mathbb{1}_{\mathtt{e} \geq 0} \times \mathtt{e}) \geq 0.$$

(3) $\mathbb{E}_{\mathcal{D}}(\mathbb{1}_{\varphi}\mathtt{e} + \mathbb{1}_{\neg\varphi}\mathtt{e}) = \mathbb{E}_{\mathcal{D}}(\mathtt{e})$, provided $[\![\varphi]\!]$ is measurable.

Using these facts as "axioms", we attempt to reformulate the key step 2 of Definition 7.1.3 as a simple quantified statement in (first-order) linear arithmetic. Consider, once again, the key statement of the principle (7.1). The central idea of our approach is to express the pre-expectation $\texttt{pre}\mathbb{E}(\texttt{e}_j)$ for each $\texttt{e}_j \in E$ as

$$\texttt{pre}\mathbb{E}(\texttt{e}_j) = \sum_{i=1}^{m} \lambda_{j,i}\texttt{e}_i + \sum_{p} \mu_{j,p}(\mathbb{1}_{\varphi_p} \times g_p)\,, \tag{7.2}$$

wherein $\lambda_{j,i} \geq 0$ and $\mu_{j,p} \geq 0$ are real-valued multipliers, g_p are linear expressions over the program variables and φ_p are assertions such that $\varphi_p \models g_p \geq 0$. The origin of the expressions g_p and assertions φ_p will be made clear, shortly. Let us fix a finite set of expressions $E = \{\texttt{e}_1, \ldots, \texttt{e}_m\}$.

Lemma 7.1.3. Suppose for all $\texttt{e}_i \in E$, the principle (7.2) holds:

$$\texttt{pre}\mathbb{E}(\texttt{e}_j) = \sum_{i=1}^{m} \lambda_{j,i}\texttt{e}_i + \sum_{p} \mu_{j,p}(\mathbb{1}_{\varphi_p} \times g_p)\,,$$

for some $\lambda_{j,i} \geq 0, \mu_{j,p} \geq 0$ and $\varphi_p \models g_p \geq 0$, then E satisfies the original induction principle (7.1):

$$\underline{\text{For all admissible } \mathcal{D},}\ (\mathbb{E}_{\mathcal{D}}(\texttt{e}_1) \geq 0\ \wedge\ \cdots\ \wedge\ \mathbb{E}_{\mathcal{D}}(\texttt{e}_m) \geq 0)\ \models\ \mathbb{E}_{\mathcal{D}}(\texttt{pre}\mathbb{E}(\texttt{e}_j)) \geq 0\,.$$

Proof. Let E be such that for each $\texttt{e}_i \in E$, we satisfy (7.2) as below:

$$\texttt{pre}\mathbb{E}(\texttt{e}_j) = \sum_{i=1}^{m} \lambda_{j,i}\texttt{e}_i + \sum_{p} \mu_{j,p}(\mathbb{1}_{\varphi_p} \times g_p)\,,$$

for some $\lambda_{j,i} \geq 0, \mu_{j,p} \geq 0$ and $\varphi_p \models g_p \geq 0$.

Let $\mathcal{D}$ be any admissible distribution such that $\bigwedge_{j=1}^{m} \mathbb{E}_{\mathcal{D}}(\texttt{e}_j) \geq 0$. Using, linearity of expectation, we note that

$$\mathbb{E}_{\mathcal{D}}\left(\sum_{i=1}^{m} \lambda_{j,i}\texttt{e}_i\right) = \sum_{i=1}^{m} \lambda_{j,i} \underbrace{\mathbb{E}_{\mathcal{D}}(\texttt{e}_i)}_{\geq 0} \geq 0\,. \tag{7.3}$$

Similarly, applying Theorem 7.1.2, we note that $\mathbb{E}_{\mathcal{D}}(\mathbb{1}_{\varphi_p} \times g_p) \geq 0$.

$$\mathbb{E}_{\mathcal{D}}\left(\sum_{p} \mu_{j,p}\ (\mathbb{1}_{\varphi_p} \times g_p)\right) = \sum_{p} \mu_{j,p} \underbrace{\mathbb{E}_{\mathcal{D}}\left(\mathbb{1}_{\varphi_p} \times g_p\right)}_{\geq 0} \geq 0 \tag{7.4}$$

Combining, (7.3) and (7.4), we note that

$$\begin{aligned}
\texttt{pre}\mathbb{E}(\mathcal{D}, \texttt{e}_j) &= \mathbb{E}_{\mathcal{D}}\left(\textstyle\sum_{i=1}^{m} \lambda_{j,i}\texttt{e}_i + \sum_p \mu_{j,p}\ (\mathbb{1}_{\varphi_p} \times g_p)\right) && \text{From Stmt. of Thm.} \\
&= \mathbb{E}_{\mathcal{D}}\left(\textstyle\sum_{i=1}^{m} \lambda_{j,i}\texttt{e}_i\right) + \mathbb{E}_{\mathcal{D}}\left(\sum_p \mu_{j,p}\ (\mathbb{1}_{\varphi_p} \times g_p)\right) && \\
&\geq 0 && \text{Applying (7.3) and (7.4)}
\end{aligned}$$

□

7.2 Conic Inductive Expectation Invariants

We now formalize this intuitive notion of inductive invariants using the concept of **conic inductive expectation invariants**. Let $\mathcal{P}$ be a program with transitions $\mathcal{T}$. Let $\mathbf{g}_i$ be a linear assertion representing the guard of the transition τ_i. We express $\mathbf{g}_i$ as $\bigwedge_{j=1}^{n_i} g_{i,j} \geq 0$, wherein $g_{i,j}$ are affine program expressions. Let $\mathbf{g_i} : (g_{i,1}\ \ldots\ g_{i,n_i})^T$ be a vector representing $\mathbf{g}_i$. Likewise, let $E = \{\texttt{e}_1, \ldots, \texttt{e}_m\}$ be a finite set of expressions, we denote the vector of expressions as $\mathbf{e} :\ (\texttt{e}_1, \ldots, \texttt{e}_m)^T$.

Definition 7.2.1 (Conic Inductive Expectation Invariants)**.** *The finite set E is a **conic inductive invariant** of the program $\mathcal{P}$ if and only if for each $\texttt{e}_j \in E$,*

(1) Initial Condition: $\mathbb{E}_{\mathcal{D}_0}(\texttt{e}_j) \geq 0$ over the initial distribution $\mathcal{D}_0$;

(2) Induction Step: There exists a vector of multipliers $\lambda_j \geq 0$, such that for each transition $\tau_l : (\mathbf{g}_l, \mathcal{F}_l)$, expression $\texttt{pre}\mathbb{E}_{\tau_l}(\texttt{e}_j)$ can be expressed as a conic combination of expressions in E and the expressions in $\mathbf{g}_l$:

$$\underline{\textit{For each } \texttt{e}_j}\ (\exists\ \lambda_j \geq 0)\ (\forall\ \tau_l \in \mathcal{T})\ (\exists\ \mu_l \geq 0)\ \texttt{pre}\mathbb{E}_{\tau_l}(\texttt{e}_j) = \lambda_j^T\ \mathbf{e} + \mu_l^T \mathbf{g}_l\,. \tag{7.5}$$

In particular, we note that the order of quantification in Equation (7.5) is quite important. We note for a given expression $\texttt{e}_j$ the multipliers λ_j must stay the same across all the transitions $\tau_l \in \mathcal{T}$. This will ensure the applicability of the linearity of expectation.

Example 7.2.1. *The set $E = \{e_1 :\ \texttt{y} - 2\texttt{x},\ e_2 :\ 2\texttt{x} - \texttt{y} + 3,\ e_3 :\ 4\texttt{x} - 3\texttt{count} + 4,\ e_4 : -2\texttt{x}+\texttt{y}-3,\ e_5 :\ -4\texttt{x}+3\texttt{count}-4\}$ is a conic inductive invariant for the program in Example 7.1.1.*

Consider $\mathtt{e}_1 : \mathtt{y} - 2\mathtt{x}$. *We have*

$$\mathtt{pre}\mathbb{E}_{\tau_1}(\mathtt{e}_1) : \mathbb{E}_{r_1}\left(\frac{3}{4}(\mathtt{y} + 2 - 2\mathtt{x} - 2r_1) + \frac{1}{4}(\mathtt{y} - 2\mathtt{x})\right) = \mathtt{y} - 2\mathtt{x}.$$

Likewise, $\mathtt{pre}\mathbb{E}_{\tau_2}(\mathtt{e}_1) : \mathtt{e}_1$, *since* τ_2 *is a stuttering transition.*

Therefore, setting $\lambda : (1\ 0\ 0\ 0\ 0)^T$, *we obtain* $\mathtt{pre}\mathbb{E}(\mathtt{e}_1) : \ \lambda^T\mathbf{e} + \mathbf{0} \times \mathbb{1}_{x+y\leq 10}$.

Changing the order of quantification in Equation 7.5 makes the rule unsound. In particular, we will address the need to maintain the multipliers λ_j the same across all transitions. Consider a variant of the Equation (7.5), as below:

$$\underline{\text{For each } \mathtt{e}_j}\ (\forall\ \tau_l \in \mathcal{T})\ (\exists\ \lambda_{\mathbf{j}} \geq \mathbf{0})\ (\exists\ \mu_l \geq 0)\ \mathtt{pre}\mathbb{E}_{\tau_l}(\mathtt{e}_j) = \lambda_j^T\ \mathbf{e} + \mu_l^T \mathbf{g}_l\,. \tag{7.6}$$

Such a rule **seems** like a natural encoding of the implication:

$$\bigwedge_{j=1}^{m} \mathtt{e}_j \geq 0\ \wedge\ \bigwedge_{k=1}^{q} g_{l,k} \geq 0\ \models\ \mathtt{pre}\mathbb{E}_{\tau_l}(\mathtt{e}_j) \geq 0\,.$$

The following example demonstrates the **unsoundness** of the rule (7.6).

Example 7.2.2. *Consider the program below:*

```
real x := unifRand(-1,1)
while (true)
      if ( x <= 0)
         x := 2 * x;
      else
         x := x / 2;
```

$$\begin{array}{rcl} X & : & \{\mathtt{x}\} \\ \top & : & \{\tau_1, \tau_2\} \\ \tau_1 & : & \begin{cases} \mathbf{g}_1 : & x < 0 \\ F_1(x) : & 2x \end{cases} \\ \tau_2 & : & \begin{cases} \mathbf{g}_2 : & x \geq 0 \\ F_2(x) : & 0.5x \end{cases} \\ \mathcal{D}_0 & : & \text{Uniform}[-1, 1] \end{array}$$

First, we observe that $\mathbb{E}_{\mathcal{D}_0}(\mathtt{x}) = 0$. *This gives us two IEI candidates* $\mathtt{x}$ *and* $-\mathtt{x}$. *Using the rule in Equation 7.6 we obtain:*

- *For transition* τ_1, *we have*

$$\mathtt{pre}\mathbb{E}_{\tau_1}(\mathtt{x}) = 2 \times (\mathtt{x}), \quad \textit{and} \quad \mathtt{pre}\mathbb{E}_{\tau_1}(-\mathtt{x}) = 2 \times (-\mathtt{x});$$

- *For transition τ_2, we have*

$$\texttt{pre}\mathbb{E}_{\tau_2}(\texttt{x}) = 0.5 \times (\texttt{x}), \quad \textit{and} \quad \texttt{pre}\mathbb{E}_{\tau_2}(-\texttt{x}) = 0.5 \times (-\texttt{x}).$$

This means that according to rule (7.6), we can conclude that $\mathbb{E}(\texttt{x} \,|\, n) \geq 0$ and $\mathbb{E}(-\texttt{x} \,|\, n) \geq 0$, and so $\mathbb{E}(\texttt{x} \,|\, n) = 0$ for all $n \geq 0$. This is clearly false since any negative initial value of $\texttt{x}$ only ever execute τ_1 and grows unbounded!

The correct version of the rule (7.5), is able to correctly prove the invariance of $-\texttt{x}$ and disprove $\texttt{x}$.

Lemma 7.2.1. Let $E : \{\texttt{e}_1, \ldots, \texttt{e}_m\}$ be a conic inductive invariant for a program $\mathcal{P}$ as given by Definition 7.2.1. It follows that each $\texttt{e}_j$ satisfies Equation (7.2):

$$\texttt{pre}\mathbb{E}(\texttt{e}_i) = \sum_{j=1}^{n} \lambda_{j,i}\texttt{e}_j + \sum_{p} \mu_{i,p}(\mathbb{1}_{\varphi_p} \times g_p)\,,$$

for $\lambda_{j,i}, \mu_{i,p} \geq 0$ and $\varphi_p \models g_p \geq 0$.

Proof. We note that for each $\texttt{e}_i$, the expression:

$$\texttt{pre}\mathbb{E}(\texttt{e}_i) : \sum_{\tau \in \top} \mathbb{1}_{g_\tau} \times \texttt{pre}\mathbb{E}_\tau(\texttt{e}_i)\,. \tag{7.7}$$

From (7.2), there exists λ such that for each transition τ,

$$\texttt{pre}\mathbb{E}_\tau(\texttt{e}_i) = \lambda^T \mathbf{e} + \mu^T \mathbf{g}_\tau\,.$$

Here the guard assertion for τ is given by $\mathbf{g}_\tau \geq 0$. Substituting this into Equation (7.7) yields,

$$\begin{aligned}
\texttt{pre}\mathbb{E}(\texttt{e}_i) &= \textstyle\sum_{\tau \in \top} \mathbb{1}_{\mathbf{g}_\tau \geq 0} \times \left(\lambda^T \mathbf{e} + \mu_\tau^T \mathbf{g}_\tau\right) \\
&= \textstyle\sum_{\tau \in \top} \mathbb{1}_{\mathbf{g}_\tau \geq 0} \times \left(\lambda^T \mathbf{e}\right) + \sum_{\tau \in \top} \mathbb{1}_{\mathbf{g}_\tau \geq 0} \times \left(\mu_\tau^T \mathbf{g}_\tau\right) \\
&= \lambda^T \textstyle\sum_{\tau \in \top} \mathbb{1}_{\mathbf{g}_\tau \geq 0} \times \mathbf{e} + \sum_{\tau \in \top} \mathbb{1}_{\mathbf{g}_\tau \geq 0} \times \left(\mu_\tau^T \mathbf{g}_\tau\right).
\end{aligned}$$

In particular, we note that having a common set of multipliers λ across transitions allows us to rewrite the summation $\sum_{\tau \in \top} \mathbb{1}_{\mathbf{g}_\tau \geq 0} \times \left(\lambda^T \mathbf{e}\right)$ as $\lambda^T \sum_{\tau \in \top} \mathbb{1}_{\mathbf{g}_\tau \geq 0} \times \mathbf{e}$. Next, since the transition

guards are mutually exclusive and exhaustive, it follows that $\sum_{\tau\in\top} \mathbb{1}_{\mathbf{g}_\tau \geq 0} \times \mathbf{e} = \mathbf{e}$. Therefore, we write

$$\texttt{pre}\mathbb{E}(\mathsf{e}_i) = \lambda^T \mathbf{e} + \sum_{\tau\in\top} \mathbb{1}_{\mathbf{g}_\tau \geq 0} \times \mu_\tau^T \mathbf{g}_\tau \,.$$

This concludes the proof.

□

Theorem 7.2.1. Let E be a conic inductive invariant for a program $\mathcal{P}$ as given by Definition 7.2.1. It follows that each $\mathsf{e}_j \in E$ is an expectation invariant of the program.

Proof. Proof simply combines Lemma 7.2.1 with Lemma 7.1.3. □

7.2.1 Pre-Expectation Closed Cones

Thus far, we have presented inductive expectation invariants as a finite set of expressions $E = \{\mathsf{e}_1, \ldots, \mathsf{e}_m\}$, satisfying the conditions in Definition 7.1.3 or 7.2.1. We transfer our notion from a finite set of expressions to a finitely generated cone of these in preparation for our fixed point characterization in the next section.

Definition 7.2.2 (Cones)**.** *Let $E = \{\mathsf{e}_1, \ldots, \mathsf{e}_k\}$ be a finite set of program expressions over the program variables $\mathbf{x}$. The set of **conic combinations (the cone)** of E is defined as*

$$\mathsf{Cone}(E) = \left\{ \lambda_0 + \sum_{i=1}^{k} \lambda_i \mathsf{e}_i \;\mid 0 \leq \lambda_i, 0 \leq i \leq k \right\}.$$

*Expressions e_i are called the **generators** of the cone.*

Given a non-empty linear assertion assertion $\varphi : \bigwedge_{i=1}^{k} \mathsf{e}_i \geq 0$, it is well-known that $\varphi \models \mathsf{e} \geq 0$ iff $\mathsf{e} \in \mathsf{Cone}(\mathsf{e}_1, \ldots, \mathsf{e}_k)$. Likewise, let E be an inductive expectation invariant. It follows that any $\mathsf{e} \in \mathsf{Cone}(E)$ is an expectation invariant of the program $\mathcal{P}$.

Example 7.2.3. *Revisiting Example 7.2.1, we consider the conic combination:*

$$4(-2\mathtt{x} + \mathtt{y} - 3) + 3(4\mathtt{x} - 3\mathtt{count} + 4) = 4\mathtt{x} + 4\mathtt{y} - 9\mathtt{count}$$

As a result, we conclude that $\mathbb{E}_{\mathcal{D}_n}(4\mathtt{x} + 4\mathtt{y} - 9\mathtt{count}) \geq 0$ *at each step* $n \geq 0$.

Analyzing the program by replacing the probabilistic statements with non-deterministic choice, and performing polyhedral abstract interpretation yields the invariant $\mathtt{x} + \mathtt{y} \leq 14$ *[52]. This allows us to bound the set of support for* $\mathcal{D}_n$*, and also allows us to conclude that* $\mathbb{E}_{\mathcal{D}_n}(14 - \mathtt{x} - \mathtt{y}) \geq 0$. *Combining these facts, we obtain,*

$$\mathbb{E}_{\mathcal{D}_n}(56 - 9\mathtt{count}) \geq 0, \text{ or equivalently, } \mathbb{E}_{\mathcal{D}_n}(\mathtt{count}) \leq \frac{56}{9}\,.$$

Conic Representations: A finitely generated cone $\mathsf{Cone}(\mathtt{e}_1, \ldots, \mathtt{e}_k)$ of affine expressions $\mathtt{e}_1, \ldots, \mathtt{e}_k$ is represented using a polyhedral cone. Specifically, let $\mathtt{e}: \; c_0 + \mathbf{c}^T\mathbf{x}$ be any element of the cone. The polyhedral cone representation uses variables $(c_0, \mathbf{c})$. Such a polyhedron can be represented using the constraint representation as:

$$C: \; P \begin{pmatrix} \mathbf{c} \\ c_0 \end{pmatrix} \leq 0$$

or as a set of generators given by the coefficient vectors of the expressions $\mathtt{e}_1, \ldots, \mathtt{e}_k$.

Example 7.2.4. *Consider the cone generated by expressions*

$$\mathtt{e}_1: \; -2x + y - 3, \; \mathtt{e}_2: \; 4x - 3z + 4\,.$$

Any element of the cone can be written as $c_0 + c_1x + c_2y + c_3z$ *wherein the constraints:*

$$(\exists\ \lambda_1, \lambda_2 \geq 0) \quad \begin{bmatrix} c_0 = -3\lambda_1 + 4\lambda_2 \ \wedge \ c_1 = -2\lambda_1 \\ c_2 = \lambda_1 \ \wedge \ c_3 = -3\lambda_2 \end{bmatrix}$$

Alternatively, we may express the cone with a vertex $(c_0, c_1, c_2, c_3): \; (0, 0, 0, 0)$ *and rays:*

$$\left\{ \begin{pmatrix} -3 \\ -2 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 4 \\ 4 \\ 0 \\ -3 \end{pmatrix} \right\}$$

7.2.2 Expectation Invariants as Fixed Points

In this section, we show that the notion of conic invariants as presented in Definition 7.2.1 can be expressed as a (pre-) fixed point of a monotone operator over finitely generated cones representing sets of expressions. This naturally allows us to use abstract interpretation starting from the cone representing all expressions ($\top$) and performing a downward Kleene iteration until convergence. We use a (dualized) widening operator to ensure fast convergence to fixed point in finitely many iterations.

Let $\mathcal{P}$ be a program over variables $\mathbf{x}$ with transitions $\mathcal{T} : \{\tau_1, \ldots, \tau_k\}$ and initial distribution $\mathcal{D}_0$. For simplicity, we describe our approach to generate affine expressions of the form $c_0 + \mathbf{c}^T\mathbf{x}$ for $c_0 \in \mathbb{R}, \mathbf{c} \in \mathbb{R}^n$. Let $\mathbb{A}(\mathbf{x})$ represent the set of all affine expressions over $\mathbf{x}$.

Polyhedral Cones of Expectation Invariant Candidates: Our approach uses finitely generated cones $I : \mathsf{Cone}(E)$ where $E = \{\mathsf{e}_1, \ldots, \mathsf{e}_m\}$ is a finite set of affine expressions over $\mathbf{x}$. Each element $\mathsf{e} \in I$ represents a **candidate expectation invariant**. Once a (pre-) fixed point is found by our technique, we obtain a cone $I^* : \mathsf{Cone}(E^*)$, wherein E^* will be shown to be a conic inductive invariant according to Definition 7.2.1.

A finitely generated cone of affine expressions $I : \mathsf{Cone}(E)$ is represented by a polyhedral cone of its coefficients $C(I) : \{(c_0, \mathbf{c}) \mid c_0 + \mathbf{c}^T\mathbf{x} \in I\}$. The generators of $C(I)$ are coefficient vectors $(c_{0,i}, \mathbf{c}_i)$ representing the expression $\mathsf{e}_i : c_{0,i} + \mathbf{c}_i^T\mathbf{x}$.

Our analysis operates on the lattice of polyhedral cone representations, Cones, ordered by the set theoretic inclusion operator $\subseteq$. This is, in fact, dual to the polyhedral domain, originally proposed by Cousot & Halbwachs [52].

Initial Cone: For simplicity, we will assume that $\mathcal{D}_0$ is specified to us, and we are able to compute $E_{\mathcal{D}_0}(\mathbf{x})$ precisely for each program variable. The initial cone I_0 is given by

$$I_0 : \mathsf{Cone}\left(\{x_1 - \mathbb{E}_{\mathcal{D}_0}(x_1), \mathbb{E}_{\mathcal{D}_0}(x_1) - x_1, \cdots, \mathbb{E}_{\mathcal{D}_0}(x_n) - x_n, x_n - \mathbb{E}_{\mathcal{D}_0}(x_n)\}\right) .$$

Such a cone represents the invariant candidates $x_i = \mathbb{E}_{\mathcal{D}_0}(x_i)$. The representation of the initial cone is given by the set of $2n$ rays of the form $[\mathbb{E}_{\mathcal{D}_0}(x_i)\ 0\ \cdots\ 0\ \pm 1\ 0\ \cdots\ 0]$.

Pre-Expectation Operators: We now describe the parts of the monotone operator over finitely generated cones. Let $E = \{\mathtt{e}_1, \ldots, \mathtt{e}_m\}$ be a set of expressions. Let $\tau : \langle \mathbf{g}, f_\tau \rangle$ be a transition, wherein $\mathbf{g} : \bigwedge_{l=1}^{p} g_l \geq 0$. We first present a pre-expectation operator over cones, lifting the notation $\mathtt{pre}\mathbb{E}_\tau$ from expressions to cones of such:

Definition 7.2.3 (Pre-Expectation Operator)**.** *The* ***pre-expectation*** *of a cone* $I : \mathsf{Cone}(E)$ ***w.r.t a transition*** τ *is defined as:*

$$\mathtt{pre}\mathbb{E}_\tau(I) = \{(\mathtt{e}, \lambda) \in \mathbb{A}(x) \times \mathbb{R}^m \mid \lambda \geq 0 \ \wedge \ \exists\, \mu \geq 0 \ (\mathtt{pre}\mathbb{E}_\tau(\mathtt{e}) \ \equiv \ \sum_{j=1}^{m} \lambda_j \mathtt{e}_j + \sum_{i=1}^{p} \mu_i \mathbf{g}_i) \ \}.$$

The refinement $\mathtt{pre}\mathbb{E}_\tau(I)$ of a cone contains all affine program expressions whose pre-expectation belongs to the conic hull of I and the cone generated by the guard assertion. For technical reasons, we attach to each expression a certificate λ that shows its membership back in the cone. This can be seen as a way to ensure the proper order of quantification in Definition 7.2.1.

Given a polyhedron $C(I)$ representing I, we can show that $C(\mathtt{pre}\mathbb{E}_\tau(I))$ is a polyhedral cone over the variables $(c_0, \mathbf{c})$ representing the expression coefficients and λ for the multipliers.

Lemma 7.2.2. For a given cone C, the pre-expectation operator across a transition $\mathtt{pre}\mathbb{E}_\tau(C)$ is also a cone.

PreExpectation of Cones: First, we define the lifting of $\mathtt{pre}\mathbb{E}_\tau(I)$ for a single cone of expressions I. Let τ be given by the guard set $\bigwedge_{i=1}^{l} \mathbf{g}_i^T \mathbf{x} + h_i \geq 0$, and update with forks $f_1, \ldots, f_k$ wherein $f_i : A_i\mathbf{x} + B_i\mathbf{r} + \mathbf{a}_i$ is taken with probability p_i. Consider a generic next state affine expression: $\mathtt{e} : \ c_0 + \mathbf{c}^T\mathbf{x}'$. We write

$$\mathtt{pre}\mathbb{E}_\tau(\mathtt{e}) : \ c_0 + \mathbf{c}^T(p_1A_1 + \cdots + p_kA_k)\mathbf{x} + \mathbf{c}^T\mathbb{E}_{\mathcal{D}_R}(p_1B_1\mathbf{r} + \cdots + p_kB_k\mathbf{r}) + \mathbf{c}^T(p_1\mathbf{a}_1 + \cdots + p_k\mathbf{a}_k)\,.$$

Simplifying, we write

$$\mathtt{pre}\mathbb{E}_\tau(\mathtt{e}) : \ (c_0 + \alpha^T\mathbf{c}) + \mathbf{c}^T\mathcal{B}\mathbf{x}\,.$$

Let I be the cone generated by the expressions $\{\mathtt{e}_1, \ldots, \mathtt{e}_k\}$ wherein $\mathtt{e}_j : \ d_{0,j} + \mathbf{d}_j^T\mathbf{x}$. The generators of the cone are given by the rays $(d_{0,j}, \mathbf{d}_j)$ for $j = 1, \ldots, k$. We compute an augmented

representation of the cone I given by the constraints:

$$P_I(d_0, \mathbf{d}, \lambda, \mu) : \ \lambda \geq 0 \ \wedge \ \mu \geq 0 \ \wedge \ d_0 = \sum_{j=1}^{k} \lambda_j d_{0,j} + \sum_{i=1}^{l} \mu_i h_i \ \wedge \ \mathbf{d} = \sum_{j=1}^{k} \lambda_j \mathbf{d}_j + \sum_{i=1}^{l} \mu_i \mathbf{g}_i \, .$$

The polyhedron P_I represents all combinations of expressions $(d_0 + \mathbf{d}^T\mathbf{x})$ that are derived by a conic combination of expressions $\mathtt{e}_1, \ldots, \mathtt{e}_k$ through multipliers $\lambda_1, \ldots, \lambda_k \geq 0$ and guard inequality expressions $\mathbf{g}_1, \ldots, \mathbf{g}_l$ through multipliers $\mu_1, \ldots, \mu_l \geq 0$. The cone $\mathtt{pre}\mathbb{E}_\tau(I)$ is given as

$$\mathtt{pre}\mathbb{E}_\tau(I) : \ (\exists \ \mu) \ P_I(c_0 + \alpha^T\mathbf{c}, \mathcal{B}^T\mathbf{c}, \lambda, \mu) \, .$$

Note that $\mathtt{pre}\mathbb{E}_\tau(I)$ is a polyhedron over variables $(c_0, \mathbf{c})$ representing an expression $\mathtt{e} : \ c_0 + \mathbf{c}^T\mathbf{x}$ and multipliers $\lambda \in \mathbb{R}^k$.

Example 7.2.5.

Next, we define a pre-expectation operator across all transitions:

$$\mathtt{pre}\mathbb{E}(I) = \{\mathtt{e} \in \mathbb{A}(x) \mid (\exists \ \lambda \geq 0) \ (\mathtt{e}, \lambda) \in \bigcap_{j=1}^{k} \mathtt{pre}\mathbb{E}_{\tau_j}(I)\}$$

An expression $\mathtt{e}$ belongs to $\mathtt{pre}\mathbb{E}(I)$ if for some $\lambda \geq 0$, $(e, \lambda) \in \mathtt{pre}\mathbb{E}_{\tau_j}(I)$ for each transition $\tau_j \in \mathcal{T}$.

Given a cone $C(I)$, we first compute the cones $C(\hat{I}_1), \ldots, C(\hat{I}_k)$ representing the pre-expectations across transitions $\tau_1, \ldots, \tau_k$, respectively. Next, we compute $C(I') : \ (\exists \ \lambda) \bigcap_{j=1}^{k} C(\hat{I}_j)$, representing $I' : \mathtt{pre}\mathbb{E}(I)$, by intersecting the cones $C(\hat{I}_j)$ and projecting the dimensions corresponding to λ.

We define the operator $\mathcal{G}$ over cones as $\mathcal{G}(I) : I_0 \cap \mathtt{pre}\mathbb{E}(I)$, where I_0 is the initial cone.

Theorem 7.2.2. The operator $\mathcal{G}$ satisfies the following properties:

(1) $\mathcal{G}$ is a monotone operator over the lattice Cones ordered by set-theoretic inclusion.

(2) A finite set of affine expressions E is a conic inductive invariant (Definition 7.2.1) if and only if $I : \mathsf{Cone}(E)$ is a pre-fixed point of $\mathcal{G}$, i.e, $I \subseteq \mathcal{G}(I)$.

7.2.3 Proof of Theorem 7.2.2

The details of the proof are quite intricate, so we build the proof of Theorem 7.2.2 in a bottom up fashion.

Let $\mathcal{P} : \langle \mathcal{T}, \mathcal{D}_0, n \rangle$ be a probabilistic loop with $\mathcal{T} = \{\tau_1, \ldots, \tau_m\}$, where each transition is of the form $\tau_i : \langle \mathbf{g}_i, \mathcal{F}_i \rangle$. We begin by showing that $\texttt{pre}\mathbb{E}_\tau$ is a monotone operator.

Lemma 7.2.3. Let $I_1 = \mathit{Cone}(e_1, \ldots, e_m)$ and $I_2 = \mathit{Cone}(h_1, \ldots, h_k)$ be two finitely generated cones such that $I_1 \subseteq I_2$ and let $\tau_i \in \mathcal{T}$, then $\texttt{pre}\mathbb{E}_{\tau_i}(I_1) \subseteq \texttt{pre}\mathbb{E}_{\tau_i}(I_2)$.

Proof. Let $(\mathtt{e}, \lambda) \in \texttt{pre}\mathbb{E}_{\tau_i}(I_1)$ for some $\mathtt{e} \in \mathbb{A}(x)$ and $\lambda \geq 0$. By Definition 7.2.2, there exists $\mu_i \geq 0$ such that $\texttt{pre}\mathbb{E}_{\tau_i}(\mathtt{e}) = \lambda^T (e_1 \cdots e_j)^T + \mu_i^T \mathbf{g}_i$.

Since $I_1 \subseteq I_2$ then for every generator e_j of I_1 there exist non-negative coefficients $\lambda_{j1}, \ldots, \lambda_{jk}$ such that $e_j = \lambda_j^T (h_1 \cdots h_k)$. Therefore, we can define the **change of basis transformation** matrix $\Lambda = \begin{bmatrix} \lambda_{11} & \cdots & \lambda_{1k} \\ \vdots & \ddots & \vdots \\ \lambda_{m1} & \cdots & \lambda_{mk} \end{bmatrix}$ such that $\begin{bmatrix} e_1 \\ \vdots \\ e_m \end{bmatrix} = \Lambda \begin{bmatrix} h_1 \\ \vdots \\ h_k \end{bmatrix}$. Notice that Λ is independent of τ_i; moreover, Λ is a non-negative matrix.

This means that $\texttt{pre}\mathbb{E}_{\tau_i}(\mathtt{e}) = \lambda^T (e_1 \cdots e_m)^T + \mu_i^T \mathbf{g}_i = \lambda^T \Lambda (h_1 \cdots h_k)^T + \mu_i^T \mathbf{g}_i$. Therefore, there exists a non-negative $\lambda' \equiv \lambda^T \Lambda$ such that $(\mathtt{e}, \lambda')$ belongs to I_2. □

A consequence of Lemma 7.2.3 we see that for every $(\mathtt{e}, \lambda)$ pair in I_1, there exists a unique $(\mathtt{e}, \lambda')$ pair in I_2 regardless of the guard $\mathbf{g}_i$ (and therefore, transition τ_i). The following result follows immediately.

Lemma 7.2.4 (Monotonicity of $\texttt{pre}\mathbb{E}$)**.** Let $I_1 = \mathit{Cone}(e_1, \ldots, e_m)$ and $I_2 = \mathit{Cone}(h_1, \ldots, h_k)$ such that $I_1 \subseteq I_2$, then $\texttt{pre}\mathbb{E}(I_1) \subseteq \texttt{pre}\mathbb{E}(I_2)$.

Proof. If $\mathbf{e} \in \texttt{pre}\mathbb{E}(I_1)$ then there exists a signature $\lambda \geq 0$ such that $(\mathtt{e}, \lambda) \in \bigcap_{\tau \in \mathcal{T}} \texttt{pre}\mathbb{E}_\tau(I_1)$. Define $\lambda' \equiv \lambda^T \Lambda$. Then $(\mathtt{e}, \lambda')$ belongs to $\bigcap_{\tau \in \mathcal{T}} \texttt{pre}\mathbb{E}_\tau(I_2)$. Therefore, $\mathtt{e}$ belongs to $\texttt{pre}\mathbb{E}(I_2)$. This completes the proof of monotonicity. □

Lemma 7.2.5. Let E be a conic inductive invariant, then $Cone(E) \subseteq I_0$.

Proof. WLOG, let $\mathtt{e}_i \in E$, then by construction, $\mathtt{e}_i$ is a generator of $Cone(E)$. By Definition 7.2.1, we know that $\mathbb{E}_{\mathcal{D}_0}(\mathtt{e}_i) = k$, for some $k \geq 0$. On the other hand, $\mathtt{e}_i$ is a linear expression, so $\mathtt{e}_i : c_0 + \mathbf{c}^T\mathbf{x}$ for some $c_0 \geq 0$, $\mathbf{c} \geq 0$.

$$I_0 \equiv Cone(\{1, x_1 - \mathbb{E}_{\mathcal{D}_0}(x_1), \mathbb{E}_{\mathcal{D}_0}(x_1) - x_1, \cdots, x_n - \mathbb{E}_{\mathcal{D}_0}(x_n), \mathbb{E}_{\mathcal{D}_0}(x_n) - x_n\}).$$

For every $j \geq 1$, define $(\lambda_{2j-1}, \lambda_{2j}) \equiv (c_j, 0)$ if $c_j \geq 0$ and $(0, -c_j)$ otherwise. Finally, define $\lambda_0 \equiv k - \sum c_j$. It then follows that $\mathtt{e}_i = \lambda^T \mathbf{i_0}$ with $\lambda = (\lambda_0, \lambda_1, \cdots, \lambda_{2n}) \geq 0$.

Therefore, $\mathtt{e}_i \in I_0$. □

Lemma 7.2.6. Let E be a conic inductive invariant and $I = Cone(E)$, then $I \subseteq \mathtt{pre}\mathbb{E}(I)$.

Proof. Let $\mathtt{e} \in I$, then by Definition 7.2.1, there exists a certificate $\lambda \geq 0$ that satisfies the requirements of Definition 7.2.2, simultaneously, for every transition.

Therefore, $\mathtt{e} \in \mathtt{pre}\mathbb{E}(I)$. □

Theorem 1 (Theorem 7.2.2)**.** *The operator $\mathcal{G}$ satisfies the following properties:*

(1) $\mathcal{G}$ is a monotone operator over the lattice Cones *ordered by set-theoretic inclusion.*

(2) A finite set of affine expressions E is a conic inductive invariant (Def. 7.2.1) if and only if $I : \mathsf{Cone}(E)$ is a pre-fixed point of $\mathcal{G}$, i.e, $I \subseteq \mathcal{G}(I)$.

Proof. **Part 1:** Let $I_1 = Cone(e_1, \ldots, e_m)$ and $I_2 = Cone(h_1, \ldots, h_k)$ such that $I_1 \subseteq I_2$. Expanding the definition of $\mathcal{G}$ and applying Lemma 7.2.4, $\mathcal{G}(I_1) = I_0 \cap \mathtt{pre}\mathbb{E}(I_1) \subseteq I_0 \cap \mathtt{pre}\mathbb{E}(I_2) = \mathcal{G}(I_2)$. This completes the proof that if $I_1 \subseteq I_2$ then $\mathcal{G}(I_1) \subseteq \mathcal{G}(I_2)$. **Part 2** ($\Rightarrow$): Let E be a conic inductive expectation invariant, then by Lemma 7.2.5, $I = Cone(E)$ is a subset of I_0. By Lemma 7.2.6, we know $I \subseteq \mathtt{pre}\mathbb{E}(I)$. Therefore, by definition of $\mathcal{G}$, I is a pre-fixed point of $\mathcal{G}$.

($\Leftarrow$): Let $I = Cone(E)$ for some set of expressions E such that $I \subseteq \mathcal{G}(I)$. Then $I \subseteq I_0 \cap \mathtt{pre}\mathbb{E}(I)$. Since $I \subseteq \mathtt{pre}\mathbb{E}(I)$ then there exists a certificate $\lambda \geq 0$ common for all transitions τ; this satisfies condition (2) of Definition 7.2.1. Now let $\mathtt{e} : c_0 + \mathbf{c}^T\mathbf{x}$ be a linear expression in I. Since

$I \subseteq I_0$, then $\mathtt{e} : \lambda'_0 + \sum_{i=1}^{m}[\lambda'_{2i-1}(x_i - \mathbb{E}_{\mathcal{D}_0}(x_i)) + \lambda'_{2i}(\mathbb{E}_{\mathcal{D}_0}(x_i) - x_i)] = \lambda'_0 + \sum_{i=1}^{m}[(\lambda'_{2i-1} - \lambda'_{2i})x_i + (\lambda'_{2i} - \lambda'_{2i-1})\mathbb{E}_{\mathcal{D}_0}(x_i)]$ for some non-negative scalars λ'_j. Define $\kappa_i \equiv \lambda'_{2i-1} - \lambda'_{2i}$. The expectation $\mathbb{E}_{\mathcal{D}_0}(\mathtt{e}) = \mathbb{E}_{\mathcal{D}_0}\left(\lambda'_0 + \sum_{i=1}^{m}[\kappa_i x_i - \kappa_i \mathbb{E}_{\mathcal{D}_0}(x_i)]\right) \geq 0$. Therefore, $\mathtt{e}$ satisfies Definition 7.2.1(1). Thus, E is a conic inductive expectation invariant. □

7.2.4 Iteration over Polyhedral Cones

Our goal is to compute the greatest fixed point of $\mathcal{G}$ representing the largest cone of expressions whose generators satisfy Definition 7.2.1. We implement this by a downward Kleene iteration until we obtain a pre-fixed point, which in the ideal case is also the greatest fixed point of $\mathcal{G}$.

$$(J_0 :\ \mathbb{A}(x)) \supseteq (J_1 :\ \mathcal{G}(J_0)) \ \supseteq \cdots (J_{k+1} :\ \mathcal{G}(J_k)) \cdots \text{ until convergence: } \ J_i \subseteq J_{i+1}\,.$$

However, the domain Cones has infinite descending chains and is not a complete lattice. Therefore, the greatest fixed point cannot necessarily be found in finitely many steps by the Kleene iteration. We resort to a **dual widening** operator $\widetilde{\nabla}$ to force convergence of the downward iteration.

Definition 7.2.4 (Dual Widening)**.** *Let I_1, I_2 be two successive cone iterates, satisfying $I_1 \supseteq I_2$. The operator $\widetilde{\nabla}(I_1, I_2)$ is a **dual widening operator** if:*

- $\widetilde{\nabla}(I_1, I_2) \subseteq I_1$, $\widetilde{\nabla}(I_1, I_2) \subseteq I_2$;
- *For every infinite descending sequence $J_0 \supseteq \mathcal{G}(J_0) \supseteq \mathcal{G}^2(J_0) \supseteq \cdots$, the **widened sequence** $J'_0 = J_0$, $J'_n = J'_{n-1} \widetilde{\nabla} J_n$ converges in finitely many steps.*

A common strategy to compute an approximation of the greatest fixed point when using dual widening is to delay widening for a fixed number K of iterations.

Example 7.2.6. *Consider a simulation of a peg performing an unbounded random walk in two dimensions (*`x`*,* `y`*). Starting at the origin, at every step the peg chooses uniformly at random a direction {N, E, S, W} and a random step size $r_1 \sim U[0, 2]$. The program* 2D-walk *tracks the steps (*`count`*) and the Manhattan distance (*`dist`*) to the origin.*

The following table summarizes the result of the expectation invariant analysis:

Cone	*Generators*	*Constraints*	*Cone*	*Generators*	*Constraints*
I_0	1, $-$***count***, `count`, `x`, $-$`x`, `y`, $-$`y`, `dist`, $-$`dist`,	$c_0 \geq 0$	I_4	1, $\mathbf{4} -$ ***count***, `count`, `x`, $-$`x`, `y`, $-$`y`, `dist`, $-$`dist`	$c_0 + 4c_4 \geq 0$, $c_0 \geq 0$
I_1	1, $\mathbf{1} -$ ***count***, `count`, `x`, $-$`x`, `y`, $-$`y`, `dist`, $-$`dist`	$c_0 + c_4 \geq 0$, $c_0 \geq 0$	I_5	1, $\mathbf{5} -$ ***count***, `count`, `x`, $-$`x`, `y`, $-$`y`, `dist`, $-$`dist`	$c_0 + 4c_4 \geq 0$, $c_0 \geq 0$
I_2	1, $\mathbf{2} -$ ***count***, `count`, `x`, $-$`x`, `y`, $-$`y`, `dist`, $-$`dist`	$c_0 + 2c_4 \geq 0$, $c_0 \geq 0$	$\vdots$	$\vdots$	$\vdots$
I_3	1, $\mathbf{3} -$ ***count***, `count`, `x`, $-$`x`, `y`, $-$`y`, `dist`, $-$`dist`	$c_0 + 3c_4 \geq 0$, $c_0 \geq 0$	I_∞	1, `count`, `x`, $-$`x`, `y`, $-$`y`, `dist`, $-$`dist`	$c_4 \geq 0$, $c_0 \geq 0$

The table shows the value of expression `count` *is unbounded from above. To force convergence, we employ dual widening after a predefined number ($K = 5$) of iterations.*

Definition 7.2.5 (Standard Dual Widening)**.** *Let $I_1 = Cone(g_1, \ldots, g_k)$ and $I_2 = Cone(h_1, \ldots, h_l)$ be two finitely generated cones such that $I_1 \supseteq I_2$. The dual widening operator $I_1 \widetilde{\nabla} I_2$ is defined as $I = Cone(g_i \mid g_i \in I_2)$. Cone I is the cone generated by the generators of I_1 that are* ***subsumed*** *by I_2.*

Example 7.2.7. *Returning to Example 7.2.6, we consider cone iterates I_4, I_5. In this case generator subsumption reduces to a simple containment check. Since generator $\mathbf{4} -$* ***count*** *is not subsumed in I_5, we arrive at $I'_5 \equiv I_4 \widetilde{\nabla} I_5 = \widetilde{I^*} = I_\infty$.*

Note 4. Alternatively, one can define dual widening as a widening operator [86, 8] over the dual polyhedron that the generators of I_1, I_2 give rise to. On the set of affine PSTS loop benchmarks our dual widening approach and those based on [86] and [8] produce identical fixed points where the difference in timings is not statistically significant.

7.3 Experimental Results

We present the experimental results of our prototype implementation that relies on PPL [8] for manipulating the polyhedral representations of cones. Table 7.1 presents the summary of the experiments we conducted on a set of probabilistic benchmarks. In Applendix A.1, we present a description of these models and the expectation invariants obtained.

Table 7.1: Summary of results: $|X|$ is the number of program variables; $|\mathcal{T}|$ - transitions; # - iterations to convergence; $\widetilde{\nabla}$ - use of dual widening. Lines L (Rays R) is the number of resultant inductive expectation equalities (inequalities) as the fixpoint generators. Time t is taken on a MacBook Pro (2.4 GHz) laptop with 8 GB RAM, running MacOS X 10.9.1 (where $\varepsilon = 0.05$ sec).

Name	Description	$\lvert X\rvert$	$\lvert\mathcal{T}\rvert$	Iters		Fixpt		t
				#	$\widetilde{\nabla}$	L	R	
MOT-EXAMPLE	Motivating Example 7.1.1	3	2	2	No	2	1	$\leq \varepsilon$
MOT-EX-INV	Example 7.1.1 with loop inv.	3	2	2	No	2	2	0.10
MOT-EX-POLY	Ex. 7.1.1 poly constr. (deg ≤ 2)	9	2	2	No	5	2	0.18
2D-WALK	Random walk in 2 dimensions	4	4	7	Yes	3	1	$\leq \varepsilon$
AGGREGATE-RV	Accumulate RVs	3	2	2	No	2	0	$\leq \varepsilon$
HARE-TURTLE	Stochastic Hare-Turtle race	3	2	2	No	1	1	$\leq \varepsilon$
COUPON5	Coupon Collector's Problem (n = 5)	2	5	2	No	1	2	$\leq \varepsilon$
FAIR-COIN-BIASED	Simulate biased coin with fair coin	3	2	3	No	1	1	$\leq \varepsilon$
HAWK-DOVE-FAIR	Stochastic 2-player game (collab.)	6	2	2	No	4	1	$\leq \varepsilon$
HAWK-DOVE-BIAS	Stochastic 2-player game (exploit)	6	2	2	No	3	1	$\leq \varepsilon$
FAULTY-INCR	Faulty incrementor	2	2	7	Yes	1	1	$\leq \varepsilon$

In all experiments we emphasize precision over computational effort. All examples except MOT-EX-LOOP-INV and MOT-EX-POLY run in under $\varepsilon = 0.05$ seconds, so we choose not to report these timing. Accordingly, dual widening $\widetilde{\nabla}$ delay was set sufficiently large at $K = 5$ to only force finite convergence but not to speed up computation. Nevertheless, the iterations converge quite fast and in many cases without the use of widening. Programs 2D-WALK and FAULTY-INCR require the widening ($\widetilde{\nabla}$) operator to ensure convergence. In all cases, **line** generators of the final pre-fixed point yield expectation invariants like $\mathbb{E}(\mathtt{e}) = 0$ and **rays** yield the invariants $\mathbb{E}(\mathtt{e}) \geq 0$.

Comparison with PRINSYS. PRINSYS [82] implements the constraint-based quantitative invariant synthesis approach developed by Katoen et al. [105]. The tool uses a manually supplied template with unknown coefficients. The REDUCE computer algebra system is used to perform quantifier elimination and simplify the constraints. We applied PRINSYS with a linear template expression $\sum_j c_j x_j$ for all state variables x_j in the program. Our comparison was carried out over the 6 benchmark examples distributed with the tool. The comparison checked whether PRINSYS could discover quantitative invariants discovered by our approach. Table 7.2 presents a summary of the comparison.

From a total set of 26 inductive expectation invariants our tool generates, PRINSYS could generate 3 of them. Notice that we had to manually provide some additional initial invariants which PRINSYS was able to trivially, yet correctly prove invariant (denoted by asterisk in last column). Overall, we observe that mutual inductive expectation invariants investigated in this paper provide interesting, significant facts about the probabilistic loops in the PRINSYS benchmarks.

Next, we attempted to check whether PRINSYS can discover additional linear quantitative invariants not discovered by our approach due to the incompleteness of widening. Unfortunately, this check turned out inconclusive at the time of the experiment. The existing PRINSYS implementation automatically generates and simplifies nonlinear constraints on the template coefficients. However, the process of deriving an actual quantitative invariant requires manually extracting solutions from a set of nonlinear inequalities. Our manual efforts failed to find new invariants unique to the PRINSYS tool, but the overall comparison remains incomplete since we could not arguably find all solutions manually.

Finally, it is important to observe that PRINSYS can generate invariants for templates that include indicator functions, while our technique curently does not. Similarly, PRINSYS handles nondeterminism in the programs, while we do not.

Table 7.2: Summary of comparison results: IEI - invariants generated by our tool; Iters, Time - number of iterations, time for our tool to converge; PRINSYS - was PRINSYS able to infer this quantitative invariant.

Name	IEI	Iters	Time	PRINSYS
BIASED-COIN	$2 - 2\mathtt{b} - \mathtt{count} = 0$	6	0.0877	No
	$3 - 4\mathtt{x} \geq 0$			No
BINOMIAL-UPDATE (M=20)	$4\mathtt{x} - 3\mathtt{n} = 0$	21	0.1337	No
	$\mathtt{x} \geq 0$			No
COWBOYS	$1 - 7\mathtt{trn} - \mathtt{cont} \geq 0$	4	0.1146	No
	$6 - 2\mathtt{trn} - 6\mathtt{cont} - 5\mathtt{cnt} = 0$			No
FAIR-COIN	$\mathtt{x} - \mathtt{y} = 0$	6	0.0844	Yes
	$3 - 4\mathtt{x} \geq 0$			No
	$-4\mathtt{x} + 3\mathtt{count} \geq 0$			No
GEOMETRIC	$\mathtt{x} \geq 0$	29	0.1988	No
	$3\mathtt{x} - \mathtt{flip} = 0$			No
	$4\mathtt{x} - \mathtt{count} = 0$			No
	$\mathtt{count} \geq 0$			Yes*
UNLIMITED-MARTINGALE	$\mathtt{rounds} \geq 0$	13	0.2	Yes*
	$32\mathtt{c} + \mathtt{rounds} \leq 1600$			No
	$\mathtt{c} + \mathtt{b} = 51$			No
	(10 additional inequalities)			No

Chapter 8

Future Work and Conclusion

In this dissertation we presented a deductive verification approach for qualitative (almost sure) and quantitative reachability and repeated reachability properties of discrete time, infinite-state polynomial stochastic systems. The central theme and main contribution is the identification of martingale processes adapted to the stochastic system that aid our analysis efforts.

Martingale theory provides two types of theoretical results that are fundamental to our verification approach:

- Martingales are stochastic processes that act as invariants in expected value across any number of steps. They satisfy strong convergence properties that we leverage to prove progress towards a target set of states.

- Martingales satisfy strong concentration of measure properties. There are tight probability inequality bounds on the probability of large deviations of martingales from their initial value.

We use the former type of results to construct the deductive proof framework for our qualitative analysis. The soundness of this framework is founded upon the convergence results of these properties. The latter type of results we use in our quantitative analysis of reachability properties. We demonstrated how the use of Azuma-Hoeffding type rules can be automated to provide tight *guaranteed* upper bounds for probabilistic assertions (and **rare events**).

Inspired by interaction between martingales and the pre-expectation operator, we built static

(program) analysis techniques to automatically infer supermartingale expressions and more general expectation invariants. Using mathematical induction and symbolic reasoning about distributions over the reachable states of a system, we extended the concept of expectation invariants to a sets of conic inductive expectation invariants that in the future can be used in more sophisticated verification tools.

Future Work. There are several future research directions that are worth exploring:

- Martingales have been used in financial applications to improve upon simulation techniques (empirical martingale simulations [59]). However, to the best of our knowledge martingales have not been used in simulation-based approaches by either the program or cyber-physical system verification communities.

- A dual point to the above is how to infer martingales through simulation or by analyzing fixed empirical datasets. This type of work has been explored in mathematical finance by Bibby et al. [15].

- The research area of concentration of measure is an active one. Martingales have gained increasing attention over the past two decades. It is important to understand the applications of more advanced COM results to verification problems. For example, square integrability and total variation measures provide even stronger concentration results [13].

- Martingales have been applied to the manual analysis of models others than the Markov models presented here. Autoregressive regressive processes and finite memory filters seem to be likely within the scope of verification techniques based on martingales.

- In Chapter 7 we saw that our simple abstract interpretation framework was also able to derive polynomial inductive expectation invariants. Additionally, there exist constraint based polynomial supermartingale generation techniques as we saw in Chapter 5.3 and also [151, 35]. It is an interesting direction to try and extend such approaches to polynomial inductive expectation invariants (pIEI).

- Finally, extending the deductive and certificate generation techniques to continuous-time stochastic systems [151] and stochastic hybrid systems is a challenging but promising future direction.

Bibliography

[1] Prism. http://www.prismmodelchecker.org/. Accessed 2016-07-04.

[2] Probabilistic-programming.org. http://probabilistic-programming.org. Accessed 2016-07-06.

[3] G. Valmorbida S. Prajna P. Seiler A. Papachristodoulou, J. Anderson and P. A. Parrilo. SOSTOOLS: Sum of squares optimization toolbox for MATLAB. http://arxiv.org/abs/1310.4716, 2013. Available from http://www.eng.ox.ac.uk/control/sostools, http://www.cds.caltech.edu/sostools and http://www.mit.edu/~parrilo/sostools.

[4] Alessandro Abate, Joost-Pieter Katoen, John Lygeros, and Maria Prandini. Approximate model checking of stochastic hybrid systems. European Journal of Control, 16(6):624–641, 2010.

[5] By Aesop. Aesop's fables. BookRix, 2015.

[6] Miguel F. Anjos and Jean B. Lasserre, editors. Handbook on semidefinite, conic and polynomial optimization. International Series in Operations Research & Management Science, 166. Springer, New York, 2012.

[7] Philippe Audebaud and Christine Paulin-Mohring. Proofs of randomized algorithms in coq. Science of Computer Programming, 74(8):568–589, 2009.

[8] Roberto Bagnara, Patricia M Hill, Elisa Ricci, and Enea Zaffanella. Precise widening operators for convex polyhedra. In Static Analysis, pages 337–354. Springer, 2003.

[9] Roberto Bagnara, Patricia M Hill, and Enea 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):3–21, 2008.

[10] Christel Baier, Joost-Pieter Katoen, et al. Principles of model checking, volume 26202649. MIT press Cambridge, 2008.

[11] Gilles Barthe, Thomas Espitau, Luis María Ferrer Fioriti, and Justin Hsu. Synthesizing probabilistic invariants via doob's decomposition. arXiv preprint arXiv:1605.02765, 2016.

[12] Alexander I Barvinok. A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed. Mathematics of Operations Research, 19(4):769–779, 1994.

[13] Bernard Bercu, Bernard Delyon, and Emmanuel Rio. Concentration inequalities for sums and martingales. Springer, 2015.

[14] Dimitri P.. Bertsekas and Steven Eugene Shreve. Stochastic optimal control: The discrete time case. Athena Scientific, 1996.

[15] Bo Martin Bibby and Michael Sørensen. Martingale estimation functions for discretely observed diffusion processes. Bernoulli, pages 17–39, 1995.

[16] Christopher M Bishop. Pattern recognition. Machine Learning, 128, 2006.

[17] Grigoriy Blekherman, Pablo A. Parrilo, and Rekha R. Thomas, editors. Semidefinite optimization and convex algebraic geometry, volume 13 of MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA; Mathematical Optimization Society, Philadelphia, PA, 2013.

[18] Mateus Borges, Antonio Filieri, Marcelo D'Amorim, and Corina S Păsăreanu. Iterative distribution-aware sampling for probabilistic symbolic execution. In Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, pages 866–877. ACM, 2015.

[19] Mateus Borges, Antonio Filieri, Marcelo d'Amorim, Corina S Păsăreanu, and Willem Visser. Compositional solution space quantification for probabilistic software analysis. In ACM SIGPLAN Notices, volume 49, pages 123–132. ACM, 2014.

[20] James Bornholt, Todd Mytkowicz, and Kathryn S McKinley. Uncertain¡t¿: A first-order type for uncertain data. ACM SIGPLAN Notices, 49(4):51–66, 2014.

[21] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. Concentration inequalities: A nonasymptotic theory of independence. Oxford university press, 2013.

[22] Olivier Bouissou, Eric Goubault, Jean Goubault-Larrecq, and Sylvie Putot. A generalization of p-boxes to affine arithmetic. Computing, 94(2-4):189–201, 2012.

[23] Olivier Bouissou, Eric Goubault, Sylvie Putot, Aleksandar Chakarov, and Sriram Sankaranarayanan. Uncertainty propagation using probabilistic affine forms and concentration of measure inequalities. In Tools and Algorithms for the Construction and Analysis of Systems, pages 225–243. Springer, 2016.

[24] Olivier Bournez and Florent Garnier. Proving positive almost-sure termination. In Term Rewriting and Applications, pages 323–337. Springer, 2005.

[25] Olivier Bournez and Florent Garnier. Proving positive almost sure termination under strategies. In Term Rewriting and Applications, pages 357–371. Springer, 2006.

[26] Stephen Boyd and Lieven Vandenberghe. Convex optimization. Cambridge university press, 2004.

[27] Richard R Brooks and Sundararaja S Iyengar. Multi-sensor fusion: fundamentals and applications with software. Prentice-Hall, Inc., 1998.

[28] Jerry R Burch, Edmund M Clarke, Kenneth L McMillan, David L Dill, and Lain-Jinn Hwang. Symbolic model checking: 1020 states and beyond. Information and computation, 98(2):142–170, 1992.

[29] Katie Byl and Russ Tedrake. Metastable walking machines. The International Journal of Robotics Research, 28(8):1040–1064, 2009.

[30] Bob Carpenter, Andrew Gelman, Matt Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Michael A Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. J Stat Softw, 2016.

[31] M. Ceska, F. Dannenberg, M. Kwiatkowska, and N. Paoletti. Precise parameter synthesis for stochastic biochemical systems. In Proc. 12th International Conference on Computational Methods in Systems Biology (CMSB'14), LNCS/LNBI. Springer, 2014. To appear.

[32] Aleksandar Chakarov and Sriram Sankaranarayanan. Probabilistic program analysis with martingales. In Computer Aided Verification, pages 511–526. Springer, 2013.

[33] Aleksandar Chakarov and Sriram Sankaranarayanan. Expectation invariants for probabilistic program loops as fixed points. In Static Analysis, pages 85–100. Springer, 2014.

[34] Aleksandar Chakarov, Yuen-Lam Voronin, and Sriram Sankaranarayanan. Deductive proofs of almost sure persistence and recurrence properties. In Tools and Algorithms for the Construction and Analysis of Systems, pages 260–279. Springer, 2016.

[35] Krishnendu Chatterjee, Hongfei Fu, and Amir Kafshdar Goharshady. Termination analysis of probabilistic programs through positivstellensatz's. arXiv preprint arXiv:1604.07169, 2016.

[36] Krishnendu Chatterjee, Hongfei Fu, Petr Novotnỳ, and Rouzbeh Hasheminezhad. Algorithmic analysis of qualitative and quantitative termination problems for affine probabilistic programs. In Proceedings of the 43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, pages 327–342. ACM, 2016.

[37] Yu-Fang Chen, Chih-Duo Hong, Bow-Yaw Wang, and Lijun Zhang. Counterexample-guided polynomial loop invariant generation by lagrange interpolation. In Computer Aided Verification, pages 658–674. Springer, 2015.

[38] Dmitry Chistikov, Rayna Dimitrova, and Rupak Majumdar. Approximate counting in smt and value estimation for probabilistic programs. In International Conference on Tools and Algorithms for the Construction and Analysis of Systems, pages 320–334. Springer, 2015.

[39] Fan RK Chung and Linyuan Lu. Complex graphs and networks, volume 107. American mathematical society Providence, 2006.

[40] Kai Lai Chung. A course in probability theory. 1974.

[41] Vasek Chvatal. Linear programming. Macmillan, 1983.

[42] Guillaume Claret, Sriram K Rajamani, Aditya V Nori, Andrew D Gordon, and Johannes Borgström. Bayesian inference using data flow analysis. In Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering, pages 92–102. ACM, 2013.

[43] Edmund Clarke, Alexandre Donzé, and Axel Legay. Statistical model checking of mixed-analog circuits with an application to a third order δ- σ modulator. In Haifa Verification Conference, pages 149–163. Springer, 2008.

[44] Edmund M Clarke, Orna Grumberg, and Doron Peled. Model checking. MIT press, 1999.

[45] David Clayton, Michael Hills, and A Pickles. Statistical models in epidemiology, volume 161. IEA, 1993.

[46] Michael A Colón, Sriram Sankaranarayanan, and Henny B Sipma. Linear invariant generation using non-linear constraint solving. In Computer Aided Verification, pages 420–432. Springer, 2003.

[47] Michael A Colóon and Henny B Sipma. Synthesis of linear ranking functions. In Tools and Algorithms for the Construction and Analysis of Systems, pages 67–81. Springer, 2001.

[48] Oswaldo Luiz Valle Costa, Marcelo Dutra Fragoso, and Ricardo Paulino Marques. Discrete-time Markov jump linear systems. Springer Science & Business Media, 2006.

[49] Costas Courcoubetis and Mihalis Yannakakis. Verifying temporal properties of finite-state probabilistic programs. In Foundations of Computer Science, 1988., 29th Annual Symposium on, pages 338–345. IEEE, 1988.

[50] Costas Courcoubetis and Mihalis Yannakakis. The complexity of probabilistic verification. Journal of the ACM (JACM), 42(4):857–907, 1995.

[51] Patrick Cousot and Radhia Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In Proceedings of the 4th ACM SIGACT-SIGPLAN symposium on Principles of programming languages, pages 238–252. ACM, 1977.

[52] Patrick Cousot and Nicholas Halbwachs. Automatic discovery of linear restraints among the variables of a program. In POPL'78, pages 84–97, January 1978.

[53] Patrick Cousot and Michael Monerau. Probabilistic abstract interpretation. In European Symposium on Programming, pages 169–193. Springer, 2012.

[54] Luca De Alfaro. Formal verification of probabilistic systems. PhD thesis, Citeseer, 1997.

[55] Jesús A De Loera, B Dutra, Matthias Köppe, S Moreinis, G Pinto, and J Wu. Software for exact integration of polynomials over polyhedra. Computational Geometry, 46(3):232–252, 2013.

[56] Alessandra Di Pierro, Chris Hankin, and Herbert Wiklicky. Probabilistic λ-calculus and quantitative program analysis. Journal of Logic and Computation, 15(2):159–179, 2005.

[57] Alessandra Di Pierro and Herbert Wiklicky. Probabilistic concurrent constraint programming: Towards a fully abstract model. In International Symposium on Mathematical Foundations of Computer Science, pages 446–455. Springer, 1998.

[58] Rayna Dimitrova, Luis Marıa Ferrer Fioriti, Holger Hermanns, and Rupak Majumdar. Probabilistic ctl: The deductive way. In Tools and Algorithms for the Construction and Analysis of Systems, pages 280–296. Springer, 2016.

[59] Jin-Chuan Duan and Jean-Guy Simonato. Empirical martingale simulation for asset prices. Management Science, 44(9):1218–1233, 1998.

[60] Devdatt P Dubhashi and Alessandro Panconesi. Concentration of measure for the analysis of randomized algorithms. Cambridge University Press, 2009.

[61] Rick Durrett. Probability: theory and examples. Cambridge university press, 2010.

[62] Cynthia Dwork. Differential privacy. In Automata, languages and programming, pages 1–12. Springer, 2006.

[63] Javier Esparza, Andreas Gaiser, and Stefan Kiefer. Proving termination of probabilistic programs using patterns. In Computer Aided Verification, pages 123–138. Springer, 2012.

[64] Xiequan Fan, Ion Grama, and Quansheng Liu. Hoeffdings inequality for supermartingales. Stochastic Processes and their Applications, 122(10):3545–3559, 2012.

[65] Gy Farkas. A fourier-féle mechanikai elv alkalmazásai (hungarian). Mathematikai és Természettudományi Ertesito, 12:457–472, 1894.

[66] William Feller. An Introduction to Probability Theory and Its Applications, Vol. 1, 3rd Edition. Wiley, 3rd edition, January 1968.

[67] Luis María Ferrer Fioriti and Holger Hermanns. Probabilistic termination: Soundness, completeness, and compositionality. In ACM SIGPLAN Notices, volume 50, pages 489–501. ACM, 2015.

[68] Antonio Filieri, Marcelo F Frias, Corina S Păsăreanu, and Willem Visser. Model counting for complex data structures. In Model Checking Software, pages 222–241. Springer, 2015.

[69] Antonio Filieri, Corina S Păsăreanu, and Willem Visser. Reliability analysis in symbolic pathfinder. In Proceedings of the 2013 International Conference on Software Engineering, pages 622–631. IEEE Press, 2013.

[70] Antonio Filieri, Corina S Păsăreanu, Willem Visser, and Jaco Geldenhuys. Statistical symbolic execution with informed sampling. In Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering, pages 437–448. ACM, 2014.

[71] V. Forejt, M. Kwiatkowska, G. Norman, and D. Parker. Automated verification techniques for probabilistic systems. In M. Bernardo and V. Issarny, editors, Formal Methods for Eternal Networked Software Systems (SFM'11), volume 6659 of LNCS, pages 53–113. Springer, 2011.

[72] FG Foster. On the stochastic matrices associated with certain queuing processes. The Annals of Mathematical Statistics, pages 355–360, 1953.

[73] Matthew Fredrikson and Somesh Jha. Satisfiability modulo counting: A new approach for analyzing privacy properties. In Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), page 42. ACM, 2014.

[74] Masahiro Fujita, Patrick C. McGeer, and JC-Y Yang. Multi-terminal binary decision diagrams: An efficient data structure for matrix representation. Formal methods in system design, 10(2-3):149–169, 1997.

[75] Andreas Gaiser. Verification of Reachability Properties and Termination for Probabilistic Systems. PhD thesis, Universität München, 2013.

[76] Jaco Geldenhuys, Matthew B Dwyer, and Willem Visser. Probabilistic symbolic execution. In Proceedings of the 2012 International Symposium on Software Testing and Analysis, pages 166–176. ACM, 2012.

[77] Wally R Gilks, Andrew Thomas, and David J Spiegelhalter. A language and program for complex bayesian modelling. The Statistician, pages 169–177, 1994.

[78] Michele Giry. A categorical approach to probability theory. In Categorical aspects of topology and analysis, pages 68–85. Springer, 1982.

[79] Noah D Goodman, Vikash K Mansinghka, Daniel M Roy, Keith Bonawitz, and Joshua B Tenenbaum. Church: A language for generative models. In In UAI, 2008.

[80] Andrew D Gordon, Thore Graepel, Nicolas Rolland, Claudio Russo, Johannes Borgstrom, and John Guiver. Tabular: a schema-driven probabilistic programming language. In ACM SIGPLAN Notices, volume 49, pages 321–334. ACM, 2014.

[81] Andrew D Gordon, Thomas A Henzinger, Aditya V Nori, and Sriram K Rajamani. Probabilistic programming. In Proceedings of the on Future of Software Engineering, pages 167–181. ACM, 2014.

[82] Friedrich Gretz, Joost-Pieter Katoen, and Annabelle McIver. Prinsyson a quest for probabilistic loop invariants. In Quantitative Evaluation of Systems, pages 193–208. Springer, 2013.

[83] Friedrich Gretz, Joost-Pieter Katoen, and Annabelle McIver. Operational versus weakest pre-expectation semantics for the probabilistic guarded command language. Performance Evaluation, 73:110–132, 2014.

[84] Friedrich Gretz, Joost-Pieter Katoen, Annabelle McIver, et al. Operational versus weakest precondition semantics for the probabilistic guarded command language. In QEST, pages 168–177, 2012.

[85] Vineet Gupta, Radha Jagadeesan, and Vijay Saraswat. Probabilistic concurrent constraint programming. In International Conference on Concurrency Theory, pages 243–257. Springer, 1997.

[86] Nicholas Halbwachs. Détermination automatique de relations linéaires vérifiées par les variables d'un programme. PhD thesis, Institut National Polytechnique de Grenoble-INPG, 1979.

[87] Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. Formal aspects of computing, 6(5):512–535, 1994.

[88] Sergiu Hart and Micha Sharir. Concurrent probabilistic programs, or: How to schedule if you must. SIAM Journal on Computing, 14(4):991–1012, 1985 (preprint 1982).

[89] Sergiu Hart, Micha Sharir, and Amir Pnueli. Termination of probabilistic concurrent program. ACM Transactions on Programming Languages and Systems (TOPLAS), 5(3):356–380, 1983.

[90] Klaus Hasselmann. Stochastic climate models part i. theory. Tellus, 28(6):473–485, 1976.

[91] Holger Hermanns, Björn Wachter, and Lijun Zhang. Probabilistic cegar. In International Conference on Computer Aided Verification, pages 162–175. Springer, 2008.

[92] Daniel P Heyman and Matthew J Sobel. Stochastic Models in Operations Research: Stochastic Optimization, volume 2. Courier Corporation, 2003.

[93] Chung-Kil Hur, Aditya V Nori, Sriram K Rajamani, and Selva Samuel. Slicing probabilistic programs. In ACM SIGPLAN Notices, volume 49, pages 133–144. ACM, 2014.

[94] Chung-Kil Hur, Aditya V Nori, Sriram K Rajamani, and Selva Samuel. A provably correct sampler for probabilistic programs. In LIPIcs-Leibniz International Proceedings in Informatics, volume 45. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2015.

[95] Joe Hurd. Formal verification of probabilistic algorithms. PhD thesis, University of Cambridge, 2001.

[96] Joe Hurd. A formal approach to probabilistic termination. In Theorem Proving in Higher Order Logics, pages 230–245. Springer, 2002.

[97] Joe Hurd, Annabelle McIver, and Carroll Morgan. Probabilistic guarded commands mechanized in hol. Theoretical Computer Science, 346(1):96–112, 2005.

[98] Alon Itai and Michael Rodeh. The lord of the ring or probabilistic methods for breaking symmetry in distributive networks. Technion-Israel Institute of Technology. Department of Computer Science, 1982.

[99] Raj Jain. The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling. John Wiley & Sons, 1990.

[100] Nils Jansen, Benjamin Lucien Kaminski, Joost-Pieter Katoen, Federico Olmedo, Friedrich Gretz, and Annabelle McIver. Conditioning in probabilistic programming. Electronic Notes in Theoretical Computer Science, 319:199–216, 2015.

[101] Claire Jones. Probabilistic non-determinism. 1990.

[102] Benjamin Lucien Kaminski and Joost-Pieter Katoen. On the hardness of almost–sure termination. In International Symposium on Mathematical Foundations of Computer Science, pages 307–318. Springer, 2015.

[103] Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, and Federico Olmedo. Weakest precondition reasoning for expected run–times of probabilistic programs. In European Symposium on Programming Languages and Systems, pages 364–389. Springer, 2016.

[104] Joost-Pieter Katoen. The probabilistic model checking landscape. 2016.

[105] Joost-Pieter Katoen, Annabelle K McIver, Larissa A Meinicke, and Carroll C Morgan. Linear-invariant generation for probabilistic programs. In Static Analysis, pages 390–406. Springer, 2010.

[106] Mark Kattenbelt. Automated Quantitative Software Verification. Oxford University, 2010.

[107] James C King. Symbolic execution and program testing. Communications of the ACM, 19(7):385–394, 1976.

[108] Dexter Kozen. Semantics of probabilistic programs. Journal of Computer and System Sciences, 22(3):328–350, 1981.

[109] Dexter Kozen. A probabilistic pdl. In Proceedings of the fifteenth annual ACM symposium on Theory of computing, pages 291–297. ACM, 1983.

[110] M. Kwiatkowska, G. Norman, and D. Parker. Stochastic model checking. In M. Bernardo and J. Hillston, editors, Formal Methods for the Design of Computer, Communication and Software Systems: Performance Evaluation (SFM'07), volume 4486 of LNCS (Tutorial Volume), pages 220–270. Springer, 2007.

[111] Marta Kwiatkowska, Gethin Norman, and David Parker. Prism 4.0: Verification of probabilistic real-time systems. In Computer Aided Verification, pages 585–591. Springer, 2011.

[112] Michel Ledoux. The concentration of measure phenomenon. Number 89. American Mathematical Soc., 2005.

[113] Daniel Lehmann, Amir Pnueli, and Jonathan Stavi. Impartiality, justice and fairness: The ethics of concurrent termination. Springer, 1981.

[114] Daniel Lehmann and Michael O Rabin. On the advantages of free choice: a symmetric and fully distributed solution to the dining philosophers problem. In Proceedings of the 8th ACM SIGPLAN-SIGACT symposium on Principles of programming languages, pages 133–138. ACM, 1981.

[115] Johan Lofberg. Yalmip: A toolbox for modeling and optimization in matlab. In Computer Aided Control Systems Design, 2004 IEEE International Symposium on, pages 284–289. IEEE, 2004.

[116] Kasper Luckow, Corina S Păsăreanu, Matthew B Dwyer, Antonio Filieri, and Willem Visser. Exact and approximate probabilistic symbolic execution for nondeterministic programs. In Proceedings of the 29th ACM/IEEE international conference on Automated software engineering, pages 575–586. ACM, 2014.

[117] Zohar Manna and Amir Pnueli. Temporal Verification of Reactive Systems: Safety. Springer, New York, 1995.

[118] Zohar Manna and Amir Pnueli. Temporal verification of reactive systems: safety. Springer Science & Business Media, 2012.

[119] Piotr Mardziel, Stephen Magill, Michael Hicks, and Mudhakar Srivatsa. Dynamic enforcement of knowledge-based security policies using probabilistic abstract interpretation. Journal of Computer Security, 21(4):463–532, 2013.

[120] Tad McGeer. Passive dynamic walking. The International Journal of Robotics Research, 9(2):62–82, 1990.

[121] Annabelle McIver and Charles Carroll Morgan. Abstraction, refinement and proof for probabilistic systems. Springer Science & Business Media, 2006.

[122] Sean P Meyn and Richard L Tweedie. Markov chains and stochastic stability. Springer Science & Business Media, 2012.

[123] Antoine Miné. A new numerical abstract domain based on difference-bound matrices. In Programs as Data Objects, pages 155–172. Springer, 2001.

[124] Sasa Misailovic, Daniel M Roy, and Martin C Rinard. Probabilistically accurate program transformations. In Static Analysis, pages 316–333. Springer, 2011.

[125] Hans D. Mittelmann. Decision tree for optimization software. `http://plato.asu.edu/sub/nlores.html#semidef`. Accessed: 2016-06-26.

[126] David Monniaux. Abstract interpretation of probabilistic semantics. In International Static Analysis Symposium, pages 322–339. Springer, 2000.

[127] David Monniaux. An abstract monte-carlo method for the analysis of probabilistic programs. In ACM SIGPLAN Notices, volume 36, pages 93–101. ACM, 2001.

[128] David Monniaux. Backwards abstract interpretation of probabilistic programs. In European Symposium on Programming, pages 367–382. Springer, 2001.

[129] Rajeev Motwani and Prabhakar Raghavan. Randomized algorithms. ACM Computing Surveys (CSUR), 28(1):33–37, 1996.

[130] Theodore Samuel Motzkin. Beiträge zur Theorie der linearen Ungleichungen. Azriel Press, 1936.

[131] Kevin P Murphy. Machine learning: a probabilistic perspective. MIT press, 2012.

[132] Jiawang Nie and Markus Schweighofer. On the complexity of putinar's positivstellensatz. Journal of Complexity, 23(1):135–150, 2007.

[133] nLab. Giry monad. `https://ncatlab.org/nlab/show/Giry+monad`. Accessed 2016-07-04.

[134] Aditya V Nori, Chung-Kil Hur, Sriram K Rajamani, and Selva Samuel. R2: An efficient mcmc sampler for probabilistic programs. In AAAI, pages 2476–2482, 2014.

[135] Pablo A Parrilo. Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization. PhD thesis, Citeseer, 2000.

[136] Avi Pfeffer. The design and implementation of ibal: A general-purpose probabilistic language. Introduction to statistical relational learning, page 399, 2007.

[137] Avi Pfeffer. Figaro: An object-oriented probabilistic programming language. Charles River Analytics Technical Report, 137, 2009.

[138] Mark Pinsky and Samuel Karlin. An introduction to stochastic modeling. Academic press, 2010.

[139] Amir Pnueli. On the extremely fair treatment of probabilistic algorithms. In Proceedings of the fifteenth annual ACM symposium on Theory of computing, pages 278–290. ACM, 1983.

[140] Amir Pnueli and Lenore Zuck. Parameterized verification by probabilistic abstraction. In Foundations of Software Science and Computation Structures, pages 87–102. Springer, 2003.

[141] Amir Pnueli and Lenore D Zuck. Probabilistic verification. Information and computation, 103(1):1–29, 1993.

[142] Martin L Puterman. Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons, 2014.

[143] Mihai Putinar. Positive polynomials on compact semi-algebraic sets. Indiana University Mathematics Journal, 42(3):969–984, 1993.

[144] Hadi Ravanbakhsh and Sriram Sankaranarayanan. Counter-example guided synthesis of control lyapunov functions for switched systems. In Decision and Control (CDC), 2015 IEEE 54rd Annual Conference on. IEEE, 2015.

[145] Reuven Y Rubinstein and Dirk P Kroese. Simulation and the Monte Carlo method, volume 707. John Wiley & Sons, 2011.

[146] Adrian Sampson, Pavel Panchekha, Todd Mytkowicz, Kathryn S McKinley, Dan Grossman, and Luis Ceze. Expressing and verifying probabilistic assertions. ACM SIGPLAN Notices, 49(6):112–122, 2014.

[147] Sriram Sankaranarayanan. Mathematical analysis of programs. PhD thesis, Stanford University, 2005.

[148] Sriram Sankaranarayanan, Aleksandar Chakarov, and Sumit Gulwani. Static analysis for probabilistic programs: inferring whole program properties from finitely many paths. ACM SIGPLAN Notices, 48(6):447–458, 2013.

[149] Michael JA Smith. Probabilistic abstract interpretation of imperative programs using truncated normal distributions. Electronic Notes in Theoretical Computer Science, 220(3):43–59, 2008.

[150] Frank A Sonnenberg and J Robert Beck. Markov models in medical decision making a practical guide. Medical decision making, 13(4):322–338, 1993.

[151] Jacob Steinhardt and Russ Tedrake. Finite-time regional verification of stochastic non-linear systems. The International Journal of Robotics Research, 31(7):901–923, 2012.

[152] Ilya Tkachev and Alessandro Abate. Characterization and computation of infinite-horizon specifications over markov processes. Theoretical Computer Science, 515:1–18, 2014.

[153] RH Tütüncü, KC Toh, and MJ Todd. Sdpt3a matlab software package for semidefinite-quadratic-linear programming, version 3.0. Web page http://www. math. nus. edu. sg/˜ mattohkc/sdpt3. html, 2001.

[154] Moshe Y Vardi. Automatic verification of probabilistic concurrent finite state programs. In Foundations of Computer Science, 1985., 26th Annual Symposium on, pages 327–338. IEEE, 1985.

[155] David Vere-Jones. Stochastic models for earthquake occurrence. Journal of the Royal Statistical Society. Series B (Methodological), pages 1–62, 1970.

[156] D Williams. Probability theory with martingales, 1997.

[157] Håkan LS Younes and Reid G Simmons. Statistical probabilistic model checking with a focus on time-bounded properties. Information and Computation, 204(9):1368–1409, 2006.

Appendix A

Benchmark Polynomial Stochastic Transition Systems

A.1 Affine Stochastic Transition Systems

Track: The TRACK benchmark describes a program that tracks a target value with feedback. Each step attempts to guess the target value (set to 113) within a range (+/-5) and obtains a feedback that is equal to the difference between the current and the target. However, when the system tries to update the current value, the updated value has some noise added to it. Once the target is close enough in value, the process stops.

Our approach infers the martingale `curValue` and the super martingale `count`. The ranking function $-$`count` is inferred as a SMRF. Another interesting set of super martingale maps infers the expression `curValue` $-$ 108 at the loop head and 0 at the loop exit. Likewise, we infer the s.m. expression 118 $-$ `curValue` at the loop head and 0 at the loop exit. Thus, we establish that `curValue` is within the range $[108, 118]$ with a high probability at loop exit.

2D Random Walk: This is a 2-dimensional random walk (see Fig. A.2) on an infinite $N \times N$ lattice. Each step is taken in either of the four directions with uniform probability. Variable `dist` tracks the Manhattan distance to the origin.

Our tool finds the following martingales: $\texttt{x}, \texttt{y}, \texttt{dist}$ and the following super martingale, that serves as the ranking function $-$`count`.

Suppose we are interested in computing the probability that after n number of steps the probability that distance is greater than $100\sqrt{n}$. We know that the variable `dist` is a bounded martingale that changes value by at most $[-1, 1]$ at each step: $|\texttt{dist}_{i+1} - \texttt{dist}_i| \leq 1$. Therefore,

applying Azuma's theorem,

$$P(|\texttt{dist}_n - \texttt{dist}_0| > 100\sqrt{n}) \leq \exp^{\frac{-10000n}{2\sum_{i=0}^{n} 1}} \sim 3.369694 * 10^{(-2172)}\,.$$

Coupon Collectors: There are $n = 5$ coupon and we try to collect them all. We continue obtaining coupons until we collect all n coupons. Figure A.3 shows the model. Obtaining a coupon we don't already have can be modeled by a geometric distribution with decreasing probability on each iteration. Our tool finds the supermartingale: $5 - i$, which guarantees termination.

Additionally, we infer many interesting martingale maps. One such map annotates the first loop with $12\texttt{count} + 65$, the second with $12\texttt{count} + 50$, the third with $12\texttt{count} + 30$, the fourth with $12\texttt{count}$ and the last loop head with the expression 0.

Fair-Bias Coin: This program simulates the outcomes of a coin whose expected value is $\frac{n}{2}$ heads out of n flips using a biased coin and a single flip of a fair coin to initiate the sequence. Although no termination criterion is specified, our tool infers the following 2 supermartingales: $2\texttt{count} - 6\texttt{heads}$ and $6\texttt{heads} - 4\texttt{count}$. Using Azuma's theorem, we can bound the range of `heads` as a function of count.

Inverted Pendulum Controller Figure A.5 shows a discretized model of a closed loop inverted pendulum controller that consists of an inverted pendulum on a movable cart. The controller maintains the position of the cart close to the origin and the angle of the pendulum close to vertical. We model disturbances that directly affect the cart velocity and its angular velocity. Our approach does not infer any non-trivial super martingale on this example other than the super martingale $\texttt{MAX} - \texttt{count}$. Abstract interpreters do not produce invariants that can help the synthesis process.

Packing Variable Weight Objects Figure A.6 shows a model that captures the packing of objects in a carton. The objects can be of varying weights ("NORMAL", "HEAVY", or "LIGHT"). Each iteration tracks how many heavy and light objects are in the carton. It tries to balance the light and the heavy objects against each other. Our analysis produces many interesting martingale and super-martingale expressions. Examples include the super martingale expressions

$40\texttt{totalWeight} - 57\texttt{nMedium} - 46\texttt{nHeavy}$, and $2\texttt{nHeavy} - \texttt{nMedium}$. Our approach also infers the martingale $20\texttt{totalWeight} - 17\texttt{nPacked} - 3\texttt{nMedium} - 6\texttt{nHeavy}$, and $20\texttt{totalWeight} - 20\texttt{nMedium} - 23\texttt{nHeavy} - 17\texttt{nLight}$. These martingales are useful since they characterize the composition of a carton's weight in terms of the heavy, light and normal objects.

Convoy2 Figure A.7 shows a discrete model that tracks a convoy of cars in terms of their positions x_i, velocities v_i and accelerations a_i for $i = 1, 2$. The leader's acceleration changes randomly while the follower simply increases/decreases their acceleration based on the leader.

Our approach inferred the martingale a_1. But was unable to infer other non-trivial properties. We suspect that non-linear martingales may exist for this example.

A.2 Polynomial Stochastic Transition Systems

In this section, we illustrate our approach to proving recurrence and persistence properties through motivating examples.

We show an example of proving recurrence property using a certificate function.

Example A.2.1. *Consider the following stochastic process over* (x, y)*:*

$$x' = x + \tfrac{1}{2}y + w_1,\ y' = \tfrac{1}{2}x + y - w_2$$

with normal random variables w_1, w_2 *with mean* -1 *and variance* 1*. Suppose, we wish to show the property* $\Box\Diamond(x - y <= 0)$*.*

For this system, we use the rule Rec *in page 67 using the function*

$$V(x, y) : \begin{cases} x - y & x - y \geq 0 \\ 0 & \textit{otherwise} \end{cases}$$

It is easy to check that the premises are satisfied by the system, and therefore, the system indeed satisfies the required recurrence property.

Example A.2.2. *Consider a Markov jump linear system (MJLS) model similar to the example in Section 4.4.3 with two modes and two state variables* $\mathbf{x} : (x, y)$*. The system jumps between two*

modes according to a Markov chain whose transitions are independent of the state $\mathbf{x}$*. In each mode, the system evolves discretely according to a matrix* A_1 *in mode* m_1 *or* A_2 *in* m_2 *where*

$$A_1 = \begin{bmatrix} 0.0 & 2.0 \\ 0.1 & 0.0 \end{bmatrix}, \qquad A_2 = \begin{bmatrix} 0.0 & 0.1 \\ 2.0 & 0.0 \end{bmatrix}.$$

The stochastic stability of MJLS has been well studied in the control theory literature [48]. According to the standard notion of mean square stability (MSS), this stochastic system is unstable, since its spectral radius is $r_\sigma = 2.005 > 1$*. This means the traces of this system need not converge in* L_2 *norm.*

On the other hand, our approach shows that the tail invariance property $\Diamond\Box(-\epsilon \leq xy \leq \epsilon)$ *holds almost surely for any given* $\epsilon > 0$*. In other words, the system's behaviors almost surely converge to any given "narrow strip" around the* x *and the* y *axes. The tail invariance property is established by the synthesis of a non-negative* α*-supermartingale expression* $m(x, y) : x^2y^2$ *via SOS relaxation, with* $\alpha = 0.04$*. In fact, letting* $\mathbf{x}_n$ *be the state at the* n*-th step for* $n \geq 0$*, we have*

$$\begin{aligned} \mathbb{E}(m(\mathbf{x}_{n+1})|\mathbf{x}_n) &= 0.5(2y_n)^2(0.1x_n)^2 + 0.5(0.1y_n)^2(2x_n)^2 \\ &= 0.04x_n^2y_n^2 = 0.04m(\mathbf{x}_n). \end{aligned}$$

Hence this proves the tail invariance property, shows that $\Diamond\Box(-\epsilon \leq xy \leq \epsilon)$ *holds almost surely for any given* $\epsilon > 0$*. In other words, that the system's behaviors almost surely converge to any given "narrow strip" around the* x *and the* y *axes.*

Example A.2.3. *Consider the following polynomial stochastic system over* $\mathbb{R}^2$*:*

$$\begin{aligned} x_{n+1} = &\quad 0.5x_n + 0.5y_n + 0.4w_{1,n}\sqrt{x_n^2 + y_n^2}, \\ y_{n+1} = &\quad 0.5x_n - 0.5y_n + 0.4w_{2,n}\sqrt{x_n^2 + y_n^2}, \end{aligned}$$

where $w_{1,n}, w_{2,n} \sim \mathcal{N}(0, 1)$ *are i.i.d. Our approach synthesizes a nonnegative* α*-multiplicative supermartingale* $m(x, y) : x^2 + y^2$*, where* $\alpha = 0.82$*. In fact,*

$$\mathbb{E}(m(x', y') \,|\, x, y) = \ 0.25((x + y)^2 + (x - y)^2) + 0.32(x^2 + y^2) = \ 0.82\, m(x, y).$$

Hence $\Diamond\Box(x^2 + y^2 \leq \epsilon)$ *holds almost surely for any* $\epsilon > 0$*. In particular, the system almost surely converges to the origin (which is an equilibrium).*

Example A.2.4. *Consider the polynomial stochastic system:*

$$\begin{aligned} x_{n+1} = &\quad 0.75y_n^4 + 0.1u_{1,n}, \\ y_{n+1} = &\quad 0.75x_n^4 + 0.1u_{2,n}, \end{aligned}$$

where $u_{1,n}, u_{2,n}$ are i.i.d. uniform random variables on $[-1, 1]$. It is easy to check that

$$X = \{(x, y) \in \mathbb{R}^2 \,|\, x^2 + y^2 \leq 1\}$$

is an invariant for the system.

The function $m(x, y) : 0.78x^2 + 1.23xy + 0.78y^2$ is a nonnegative α-multiplicative supermartingale over X with $\alpha = 0.75$. Hence $\Diamond\Box(m(x, y) \leq \epsilon)$ holds almost surely for any $\epsilon > 0$. Note that m defines a (weighted) norm on $\mathbb{R}^2$; the persistence property implies that any sample path of the given system converges almost surely to the origin.

Example A.2.5. *Consider the following stochastic system, a modified example from [152]:*

$$\begin{aligned} x_{n+1} := &\quad 0.1y_n(3x_n^2 + 2y_n^2 - 0.5) + 0.1u_{1,n}\sqrt{x_n^2 + y_n^2}, \\ y_{n+1} := &\quad 0.1y_n(2x_n^2 + 4x_n y_n + 3y_n^2 - 0.5) + 0.1u_{2,n}\sqrt{x_n^2 + y_n^2}, \end{aligned}$$

where $u_{1,n}, u_{2,n}$ are i.i.d. uniform random variables on $[-\sqrt{3}, \sqrt{3}]$. It is easy to see that 0 is the only equilibrium and that $X = \{(x, y) \in \mathbb{R}^2 \,|\, x^2 + y^2 \leq 1\}$ is an invariant.

The function $m(x, y) : 1.55x^2 + 2.36xy + 1.34y^2$ is a nonnegative α-multiplicative supermartingale over X with $\alpha = 0.5$. Hence $m(x, y)$ converges almost surely to 0. Note that m defines a (weighted) norm on $\mathbb{R}^2$, showing that any sample path of the given system converges almost surely to the origin.

```
real curVal := unifRand(0,500);  // current value is drawn random
real tgtVal := 113;  // target is fixed for convenience
int count := 0; // Count of number of iterations
// Main iteration loop
while(count <= N || tgtVal - curVal <= 5 || tgtVal-curVal >= -5){
  // Difference b/w current and target is the feedback we obtain
  delta = (tgtVal - curVal);
  // Try and move the pointer by delta units, but
  // we add an uniform error between -20 and 20 units.
  d = (delta + unifRand(-20;20));
  // update our current value.
  curVal = (curVal + d);
  // if we threaten to go beyond MAX, truncate it to MAX
  if (curVal > MAX) then
      curVal = MAX
   end;
  // If we threaten to go below 0, truncate it to 1
  if (curVal < 1) then
      curVal = 1
  end;
  // Increment the counter.
  count = (count + 1);
}
```

Figure A.1: The “value tracking” benchmark program.

```
real x,y, dist, count
while (count <= N) {
   c = choice (1:1/4, 2:1/4, 3:1/4, 4:1/4);
   if (x >= 0){
     if (y >= 0) {
       switch (c){
          1: x,dist := (x + r1, dist + r1)
          2: y,dist := (y + r1, dist + r1)
          3: x,dist := (x - r1, dist - r1)
          4: y,dist := (y - r1, dist - r1)
        }
     } else{
         switch (c){
          1: x,dist := (x + r1, dist + r1)
          2: y,dist := (y + r1, dist - r1)
          3: x,dist := (x - r1, dist - r1)
          4: y,dist := (y - r1, dist + r1)
        }
     }
   } else {
     if (y >= 0) {
       switch (c){
          1: x,dist := (x - r1, dist + r1)
          2: y,dist := (y + r1, dist + r1)
          3: x,dist := (x + r1, dist - r1)
          4: y,dist := (y - r1, dist - r1)
        }
     } else{
         switch (c){
          1: x,dist := (x - r1, dist + r1)
          2: y,dist := (y + r1, dist - r1)
          3: x,dist := (x + r1, dist - r1)
          4: y,dist := (y - r1, dist + r1)
        }
     }

   }
   count := count + 1;
}
```

Figure A.2: 2D random walk example.

```
int count,i
i = 1;
count  = 0;
while ( i >= 1 &  i <= 2) {
    count := count +1 ;
    if (flip(4/5)){
        i := i + 1 ;
        break;
    }
}

while ( i >= 2 &  i <= 3) {
    count := count +1 ;
    if (flip(3/5)){
        i := i + 1 ;
        break;
    }
}

while ( i >= 3 &  i <= 4) {
    count := count +1 ;
    if (flip(2/5)){
        i := i + 1 ;
        break;
    }
}

while ( i >= 4 &  i <= 5) {
    count := count +1 ;
    if (flip(1/5)){
        i := i + 1 ;
        break;
    }
}
// Collected all coupons
```

Figure A.3: Coupon collector problem for $n = 5$ coupons.

```
int count, old, heads
// We need a fair coin for a seed
count := 1
if (flip(1/2)){
  old := 1
  heads := 1
} else {
  old := 0
  heads := 0
}

while (true){
  if (old <= 0){
    if (flip(2/3)){
      heads := heads +1
      old := 1
    } else {
      old := 0
    }
  } else {
     if (flip(1/3)){
       old := 1
       heads := heads +1
     } else {
       old := 0
     }
  }
  count = count +1;
}
```

Figure A.4: Fair coin from a biased coin.

```
MAX = 10;
cartPos := unifRand(-5.0,-3.0);
cartVelo := unifRand(2.0,3.0);
pAng := unifRand(1.0,5.0);
pAngDer := unifRand(-0.5,0.5);
count := 0;
while (count <= MAX ){
 count = count +1;
 cartPos = cartPos + 0.01 * cartVelo ;
 cartVelo = 0.02*cartPos + 1.03*cartVelo - 0.3*pAng - 0.06*pAngDer
        + unifRand(-1.0;1.0);
 pAng = pAng + 0.01 * pAngDer;
 pAngDer = 0.04*cartPos + 0.07*cartVelo - 0.51*pAng + 0.85*pAngDer
        + unifRand(-0.8;0.8)

}
```

Figure A.5: Inverted pendulum controller (Discretized) under disturbance.

```
nPerCarton := 10;
nLight := 0;
nHeavy := 0;
nMedium := 0;
nPacked := 0;
totalWeight := 0
while (nPacked < nPerCarton) {
   obj := choose (NORM: 1/2, HEAVY: 1/4, LIGHT: 1/4 );
   switch (obj) {
     NORM:
        nMedium := nMedium+1;
        nPacked := nPacked +1
        totalWeight := totalWeight + 0.9 + unifRand(0,0.2);
     HEAVY:
        totalWeight := totalWeight + 1.1 + unifRand(0,0.1);
        if (nHeavy <= nLight) {
                nHeavy := nHeavy+1
                nPacked := nPacked +1
        }
      LIGHT:
         totalWeight := totalWeight + 0.8 + unifRand(0,0.1);
         if (nLight <= nHeavy){
                nLight := nLight+1;
                nPacked := nPacked +1
         }
   }
   count := count +1
}
```

Figure A.6: Packing objects of different weights in a carton.

```
real x1, v1, a1
real x2, v2, a3

while (true){
  x1 := x1 + v1 + 1/2 * a1
  v1 := v1 + a1
  a1 := a1 + unifRand(-2,2)
  x2 := x2 + v2 + 1/2 * a2
  v2 := v2 + a2
  if (x1 - x2 <= 8 & a2 >= -4){
      a2 := a2 -1;
   }
  if (x1 - x2 >= 12 & a2 <= 4){
     a2 := a2 + 1;
  }
}
```

Figure A.7: Convoy of cars with stochastic leader.