Journal of Automated Reasoning manuscript No.
(will be inserted by the editor)

Markov chains and Markov decision processes in Isabelle/HOL

Johannes Hölzl

Received: – / Accepted: –

Abstract This paper presents an extensive formalization of Markov chains (MCs) and Markov decision processes (MDPs), with discrete time and (possibly infinite) discrete state-spaces. The formalization takes a coalgebraic view on the transition systems representing Markov chains and constructs their trace spaces. On these trace spaces properties like fairness, reachability, and stationary distributions are formalized. Similar to Markov chains, MDPs are represented as transition systems with a construction for trace spaces. These trace spaces provide maximal and minimal expectation over all possible non-deterministic decisions. As applications we provide a certifier for finite reachability problems and we relate the denotational semantics and operational semantics of the probabilistic guarded command language (pGCL).

A distinctive feature of our formalization is the order-theoretic and coalgebraic view on our concepts: we view transition systems as coalgebras, we view traces as coinductive streams, we provide iterative computation rules for expectations, and we define many properties on traces as least or greatest fixed points.

Keywords Markov chains · Markov decision processes · probabilistic guarded command language · probabilistic model checking · Isabelle/HOL

1 Introduction

Markov chains (MCs) and Markov decision processes (MDPs) with discrete state space and discrete time are popular probabilistic models of network protocols, algorithms, communication systems or biological systems. In computer science, they are used to analyze probabilistic programs [25], provide semantics for programming languages [23], or analyze queuing and reliability problems [50]. In this paper we present the formalization of Markov chains and Markov decision processes in Isabelle/HOL. The applications of this formalization range from the implementation of a probabilistic model checker on

The author is supported by the DFG project Ni 491/15-1.

Johannes Hölzl
Department of Informatics, Technical University of Munich, 85748 Garching, Germany
E-mail: hoelzl@in.tum.de

MDPs — e.g. extending the CAVA project [18] to probabilistic models —, over using probabilistic models in software verification [36], to the formalization of probabilistic semantics [23].

From a computer science point there are two interesting views on probabilistic programs: (1) a denotational view where the observation is the distribution of the program results including its termination and (2) an operational view where the observation is the distribution of program traces. While (1) requires less mathematical support it also is less expressive. The operational semantics allows us to reason about the interaction with deliberately non-terminating software like operating systems. In this paper we formalize both views while focusing on the operational semantics and later relate it with the denotational semantics. This allows us to introduce an operational view on probabilistic programs, relate it to the denotational view (see Section 5), and also analyze the behaviors of non-terminating programs (see Section 3.6).

1.1 Approach

In this paper we follow a coalgebraic and order-theoretic view on Markov chains. This is different from the traditional way, where properties on Markov chains are expressed using explicit time points t_i and specific paths $x_0, x_1, \ldots, x_i$. For example, the memoryless property states, that for a time point t and for all paths $s_0, \ldots, s_t$:

$$\Pr(X_t = s_t \mid X_{t-1} = s_{t-1}, \ldots, X_0 = s_0) = \Pr(X_t = s_t \mid X_{t-1} = s_{t-1}) \; . \qquad (1)$$

This traditional approach is based on the analysis of stochastic processes (e.g. the sequence of random variables X_t in Eq. (1)), especially when working with continuous time. It is however difficult to derive computation rules over such stochastic processes. An obvious observation of Eq. (1) is that it is a very rigid statement, we can only reason about a single path on our Markov chain. Another observation is that there is an implicit probability space in $\Pr(\ldots)$ and on top of this we have a stochastic process X. In contrast to this, we introduce a coalgebra to model the transition function, construct a probability space, and give a powerful prove rule. Another difference to the traditional approach is: we do not split the stochastic process into a probability space and the stochastic process X, but we directly operate on the law of the stochastic process, i.e. the push-forward measure of X.

In the coalgebraic approach, we first start with a transition system. A $\mathcal{F}$-transition system is represented as a pair $(A, \tau : A \to \mathcal{F}(A))$, where A is the state space and τ describes the transitions per state. And $\mathcal{F}$ is a functor describing the possible structure of the transitions, e.g. $\mathcal{F}(A) = \mathcal{P}(A)$ would be a purely non-deterministic transition system. In our setting we will use two instances for $\mathcal{F}$: $\mathcal{F}(A) = \mathcal{D}(A)$, the probability mass functions (pmfs, also called discrete distributions) on A, or $\mathcal{F}(A) = \mathcal{P}(\mathcal{D}(A))$, sets of pmfs on A. The former are purely probabilistic, the latter combine probabilistic and non-deterministic choice. In our Isabelle/HOL formalization we will write $K :: \alpha \Rightarrow \alpha$ pmf or $K :: \alpha \Rightarrow \alpha$ pmf set for transition systems (K for Markov kernel). The set A is represented by the type α and $\mathcal{F}$ is determined by the return type.

For a Markov chain given by a kernel K, we want to define the expectation of a function on traces in this Markov chain. For this we construct a probability measure $\mathcal{T}_s$ describing the probability that a set of traces is observed in the Markov chain when started in state s. Besides being a probability measure $\mathcal{T}_s$ has the following defining

property: For each measurable function f and state s, we have the following equation for the expectation of f:

$$\int_\omega f(\omega) \ \mathsf{d}\mathcal{T}_s = \int_t \left(\int_\omega f(t{\cdot}\omega) \ \mathsf{d}\mathcal{T}_t \right) \ \mathsf{d}K_s \tag{2}$$

We call this equation the *iteration rule*. It says: the expectation of $f(\omega)$ over all traces ω starting from s is the expectation over all following states t (of the distribution given by K_s) of the expectation of f over all traces when we start in t.

What are the benefits of these approaches? The coalgebraic approach gives us a compositional way to define the transition system of Markov chains. This is done by defining a monad on the type of probability mass functions which allows us to compose the transitions per state from basic distributions. Also after applying Eq. (2) the integral over K is usually nicely decomposed following the construction of K itself. Conversely if we want to prove a statement abstractly over K the Eq. (2) allows us to encode many statements directly in terms of f, while Eq. (1) often requires to bring the statement in the right form. Instead of splitting the trace space up into a probability space and a stochastic process we directly operate on the trace space. This results in slightly less cluttered theorem statements. To reconstruct the traditional statements it is enough to equate the trace space with the push-forward measure of the stochastic process.

The order-theoretic view is expressed by the fact that we specify properties using least and greatest fixed points. For example, the hitting time h_S over an infinite word ω is often defined as the first time t, s.t. $\omega_t \in S$. We define h_S as the least fixed point of the functional F where $F\ h_S\ (s{\cdot}\omega) = (\mathsf{if}\ s \in S\ \mathsf{then}\ 0\ \mathsf{else}\ h_S(\omega) + 1)$. In many cases, the integral of a fixed point is easy to compute by using the iteration rule (and the transfer theorems presented in Section 2.2.1). A good overview of the order-theoretic approach is provided by Monniaux [43], presenting abstract interpretation to analyze MDPs. Again the benefit of this approach is compositionality: many properties are expressible as fixed points and the integral of a fixed point is often proved to be a fixed point again with just one proof rule.

1.2 Contributions

The main contribution of this work is the formalization of Markov chains and Markov decision processes with potentially infinite state spaces. This does not only include the construction of the trace spaces, but consequently developing theorems to compute certain probabilities, e.g. reachability, hitting time or stationary distributions, on countable state spaces. The central theorems for Markov decision processes are the existence of optimal schedulers and the iteration rules for reachability problems. This allows us to give a very simple formalization of the equivalence proof for the operational and denotational semantics of the probabilistic guarded command language (pGCL).

1.3 Structure

In the next section we give an overview of the foundations our formalization is based on, i.e. fixed points on complete lattices, measure theory, and coinductive streams. Starting with Section 3 we formalize Markov chains: construct the trace space and provide

theorems to analyze fixed points on them. Section 3.6 is concerned with the traditional Markov chain analysis of recurrent states and stationary distributions. In Section 4 we formalize MDPs, maximal and minimal expectations on them, and the analysis of reachability problems. In Section 5 we equate the denotational and operational semantics of pGCL. Finally, we give an overview of related work concerned with the analysis of probabilistic systems in interactive theorem provers.

The Isabelle/HOL formalizations of these theories is publicly available in the AFP entry `Markov_Models` [32].

2 Foundations

This section provides an overview of formalized theory in Isabelle/HOL as required for Markov chains and MDPs. That is we introduce the key definitions, theorems, and notations.

The term syntax used in this paper follows Isabelle/HOL, i.e. function application is juxtaposition as in $f\ t$ and $t :: \tau$ means that term t has type τ. Types are built from the base types like bool (booleans), nat (natural numbers), real (real numbers), $\mathsf{enat} = \mathsf{nat} \cup \{\infty\}$ (extended natural numbers), $\mathsf{ereal} = \mathsf{real} \cup \{-\infty, \infty\}$ (extended real numbers), type variables (α, β, etc), via the function type constructor $\alpha \Rightarrow \beta$, and via the set type constructor $\alpha\ \mathsf{set}$. Isabelle/HOL supports type classes providing common operations and specifications for a class of types. Type variables can be annotated with a type class, written as $\tau :: \mathsf{tc}$. By giving a definition for the type class operations and proving the associated specifications we instantiate a type for a type class. Important for us are order type classes like $\mathsf{complete\text{-}lattice}$ or topologies like $\mathsf{topological\text{-}space}$. We define infix syntax for the function space: $A \to B = \{f \mid \forall a \in A.\ f\ a \in B\}$, for the function image: $f\ `\ I = \{f\ x \mid x \in I\}$, for the relational image: $R\ ``\ S = \{y \mid (x, y) \in R \wedge x \in S\}$, and for the indicator function $\mathsf{indicator}\ A\ x = (\mathsf{if}\ x \in A\ \mathsf{then}\ 1\ \mathsf{else}\ 0)$. For types α and β in the order type class, we define for a function $f :: \alpha \Rightarrow \beta$ to be monotone: $\mathsf{mono}\ f \longleftrightarrow (\forall a\ b.\ a \le b \longrightarrow f\ a \le f\ b)$ and anti-monotone: $\mathsf{antimono}\ f \longleftrightarrow (\forall a\ b.\ a \le b \longrightarrow f\ b \le f\ a)$.

2.1 Lattices and Fixed Points

When working with infinite lists, extended real numbers ($\overline{\mathbb{R}}$), or the semantics of a while combinator (e.g. in pGCL), lattice theory and fixed points are helpful abstractions. For an introduction in lattices and orders see the book by Davey and Priestley [14]. Most of Isabelle's fixed point theory was introduced by Paulson [44]. For the theory presented in this paper we mainly added Lemma 2 and the transfer rules (Lemmas 4 and 5).

Isabelle/HOL provides type classes for complete lattices ($\mathsf{complete\text{-}lattice}$) and complete linear orders ($\mathsf{complete\text{-}linorder}$). They extend orders by binary supremum ($x \sqcup y$), binary infimum ($x \sqcap y$), as well as supremum of a set ($\bigsqcup A$) and infimum of a set ($\bigsqcap A$). For the latter two we use the binder syntax: $\bigsqcup_{i \in I} A\ i = \bigsqcup(A`I)$ and $\bigsqcap_{i \in I} A\ i = \bigsqcap(A`I)$. Important complete lattices include bool, enat, ereal, and the function space $\alpha \Rightarrow \beta$ if β is a complete lattice. In the rest of this subsection we assume that all lattices and orders are complete lattices.

On complete lattices, we define the least and greatest fixed points of a function f to be $\mathsf{lfp}\ f = \bigsqcap\{u \mid f\ u \le u\}$ and $\mathsf{gfp}\ f = \bigsqcup\{u \mid u \le f\ u\}$. By the Knaster-Tarski theorem

we know that for each monotone function f the fixed points always exist. Then we get the equations $\mathsf{lfp}\, f = f\,(\mathsf{lfp}\, f)$ and $\mathsf{gfp}\, f = f\,(\mathsf{gfp}\, f)$. And for least and greatest fixed points we have induction and coinduction rules:

Lemma 1 (Fixed point induction and coinduction)

$$\frac{\mathsf{mono}\, f \qquad \forall x \le \mathsf{lfp}\, f.\ P\, x \longrightarrow P\,(f\, x) \qquad \forall S.\ (\forall x \in S.\ P\, x) \longrightarrow P\left(\bigsqcup S\right)}{P\,(\mathsf{lfp}\, f)}$$

$$\frac{\mathsf{mono}\, f \qquad \forall x \ge \mathsf{gfp}\, f.\ P\, x \longrightarrow P\,(f\, x) \qquad \forall S.\ (\forall x \in S.\ P\, x) \longrightarrow P\left(\bigsqcap S\right)}{P\,(\mathsf{gfp}\, f)}$$

Note that the third premise (the admissibility *assumption for P) also implies $P\,\bot$ (resp. $P\,\top$).*

To calculate with fixed points we prove the rolling and the diagonal rules from fixed point calculus [5, 14]:

Lemma 2 (Rolling, iteration, and diagonal rule) *The following fixed point equations hold for least ($\mathsf{fp} = \mathsf{lfp}$) and greatest fixed points ($\mathsf{fp} = \mathsf{gfp}$).*

$$\frac{\mathsf{mono}\, g \qquad \mathsf{mono}\, f}{g\,(\mathsf{fp}\,(\lambda x.\ f\,(g\, x))) = \mathsf{fp}\,(\lambda x.\ g\,(f\, x)))} \qquad \frac{\mathsf{mono}\ f}{\mathsf{fp}\,(f^{n+1}) = \mathsf{fp}\, f}$$

$$\frac{\forall x\, y\, w\, z.\ x \le y \longrightarrow w \le z \longrightarrow f\, x\, w \le f\, y\, z}{\mathsf{fp}\,(\lambda x.\ \mathsf{fp}\,(f\, x)) = \mathsf{fp}\,(\lambda x.\ f\, x\, x)}$$

To use least and greatest fixed points in measure theory, we need countable approximations of them. This is possible if the function f is continuous. We call a function f sup-continuous (inf-continuous) if for each monotone sequence $A :: \mathsf{nat} \Rightarrow \alpha$ the function commutes with supremum: $f\,(\bigsqcup_i A\, i) = \bigsqcup_i f\,(A\, i)$, resp. for a anti-monotone sequence A the function commute with infimum: $f\,(\bigsqcap_i A\, i) = \bigsqcap_i f\,(A\, i)$. Inf-continuous and sup-continuous functions are monotone.

Lemma 3 (Fixed points as infinite iteration) *For sup-continuous (inf-continuous) functions the least (greatest) fixed point is the supremum (infimum) of all iterations:*

$$\frac{\mathsf{sup\text{-}continuous}\, f}{\mathsf{lfp}\, f = \bigsqcup_{i::\mathsf{nat}} f^i\, \bot} \qquad \frac{\mathsf{inf\text{-}continuous}\, f}{\mathsf{gfp}\, f = \bigsqcap_{i::\mathsf{nat}} f^i\, \top}$$

The following transfer rule allows us to transfer fixed points between types. Our theorem is for continuous functions, which is necessary when transferring through the non-negative integral.

Lemma 4 (Transfer rule for continuous fixed points)

$$\frac{\mathsf{sup\text{-}continuous}\, f, g, \textit{and}\ \alpha \qquad \alpha\,\bot = \bot \qquad \alpha \circ f = g \circ \alpha}{\alpha(\mathsf{lfp}\, f) = \mathsf{lfp}\, g} \qquad \frac{\mathsf{inf\text{-}continuous}\, f, g, \textit{and}\ \alpha \qquad \alpha\,\top = \top \qquad \alpha \circ f = g \circ \alpha}{\alpha(\mathsf{gfp}\, f) = \mathsf{gfp}\, g}$$

Proof (by equational reasoning)

$$\alpha(\mathsf{lfp}\ f) = \alpha(\bigsqcup_i f^i\ \bot) = \bigsqcup_i \alpha(f^i\ \bot) = \bigsqcup_i g^i\ (\alpha\ \bot) = \bigsqcup_i g^i\ \bot = \mathsf{lfp}\ g$$

The proof for the greatest fixed point is dual. □

However, these transfer rules require that α is well-behaved on the whole type. In the later application to measures and integrals, the values on which we apply α need to be restricted to measurable sets and non-negative functions. Also α does not necessarily map $\bot$ to $\bot$ (or $\top$ to $\top$), it is only smaller (or greater) than all elements returned by g. For this we introduce the stronger variant, which restricts the elements to which we apply α by a predicate P:

Lemma 5 (Strong transfer rule for continuous fixed points)

$$\frac{\begin{array}{c} P\,\bot \qquad \forall x.\ P\,x \longrightarrow P\,(f\,x) \\ \forall M.\ (\forall i.\ P\,(M\,i)) \longrightarrow P\,(\bigsqcup_i M\,i) \qquad \mathsf{sup\text{-}continuous}\ f\ \textit{and}\ g \\ \forall M.\ \mathsf{mono}\ M \longrightarrow (\forall i.\ P\,(M\,i)) \longrightarrow \alpha\,(\bigsqcup_i M\,i) = (\bigsqcup_i \alpha\,(M\,i)) \\ \alpha\,\bot \le \mathsf{lfp}\ g \qquad \forall x \le \mathsf{lfp}\ f.\ P\,x \longrightarrow \alpha\,(f\,x) = g\,(\alpha\,x) \end{array}}{\alpha(\mathsf{lfp}\,f) = \mathsf{lfp}\,g}$$

The rule for greatest fixed points is dual.

Proof This proof is similar to the proof of Lemma 4. However this time we need to apply anti-symmetry to be able to use $\alpha\ \bot$ as a lower bound. By induction on i we show that $P\ (f^i\ \bot)$ holds, by continuity of P follows $P\ (\mathsf{lfp}\ f)$. Also we show $\alpha(f^i\ \bot) \le \mathsf{lfp}\ g$ by induction on i. Hence with continuity of α and $\mathsf{sup\text{-}continuous}\ f$:

$$\alpha(\mathsf{lfp}\ f) = \bigsqcup_i \alpha(f^i\ \bot) \le \mathsf{lfp}\ g$$

The converse direction is shown by fixed point induction on g. There the induction hypothesis is $S \le \alpha(\mathsf{lfp}\ f)$ and we finish the proof with the equational calculation:

$$g\ S \le g\ (\alpha\ (\mathsf{lfp}\ f)) = \alpha\ (f\ (\mathsf{lfp}\ f)) = \alpha\ (\mathsf{lfp}\ f).$$

□

2.2 Measures and Probability Spaces

The measure and probability theory used in this paper was introduced in Isabelle/HOL by Hölzl and Heller [28], and since then continuously extended [4,16,22,26,30,31,33]. It is generic enough to be used for other applications than discrete stochastic processes, as demonstrated by Gouezel [22] and by Avigad *et al.* [4]. In this section we give a short overview of the existing measure spaces. A more detailed overview of the theory is given in [26, 28].

A measurable space on a set Ω is a σ-algebra on Ω, i.e. a set of sets which contains the empty set and the space Ω, and which is closed under complement and countable union. They are used to specify the domains of measures: measures are undefined on non-measurable sets. All theorems involving measures require that the occurring functions and sets are measurable. This propagates to derived objects, be it probabilities or integrals. A function μ is a measure on a σ-algebra $\mathcal{A}$ on Ω iff it is a non-negative and countably-additive function and maps $\mu\ \emptyset = 0$.

Definition 1 (Measures) Measures are represented by the type α measure, with the following projection functions:

$$\begin{aligned}\mathsf{space} &:: \alpha\ \mathsf{measure} \Rightarrow \alpha\ \mathsf{set}\\ \mathsf{sets} &:: \alpha\ \mathsf{measure} \Rightarrow \alpha\ \mathsf{set}\ \mathsf{set}\\ \mathsf{emeasure} &:: \alpha\ \mathsf{measure} \Rightarrow \alpha\ \mathsf{set} \Rightarrow \mathsf{ereal}\end{aligned}$$

For $M :: \alpha$ measure, the sets sets M form a σ-algebra on space M, and emeasure M is a measure on that σ-algebra. For finite measure values we introduced measure :: α measure $\Rightarrow \alpha$ set $\Rightarrow$ real as abbreviation for measure $M\ A =$ emeasure $M\ A$ (with an implicit coercion from extended reals to reals on the right-hand side), if the measure of A is infinite then measure $M\ A = 0$.

We define that a predicate P holds almost-everywhere on M when the measure of its complement is 0:

$$AE\ x\ \mathsf{in}\ M.\ P\,x \longleftrightarrow \exists N \in \mathsf{sets}\,M.\ \mathsf{emeasure}\ M\ N = 0 \wedge \{x \in \mathsf{space}\,M.\ \neg\ P\,x\} \subseteq N.$$

When M is a probability measure, i.e. emeasure M (space M) $= 1$, we write prob-space M. On such measures, we introduce the notation:

$$\Pr(x\ \mathsf{in}\ M.\ P\,x) = \mathsf{measure}\,M\,\{x \in \mathsf{space}\,M.\ P\,x\}.$$

An important way to construct measurable spaces, is sigma $\Omega\ A$, which is the smallest σ-algebra on the space Ω s.t. all sets in A are measurable. And $\bigsqcup^{\sigma}_{i\in I} M_i$ which is the smallest σ-algebra s.t. for all $i \in I$ the measurable sets of M_i are also measurable.

Measurability can be lifted from sets to functions. Measurable functions are a central notion in measure theory, be it for integration, transformation of measures or as predicates.

Definition 2 (Measurable Functions) A *measurable function* f maps elements of M to elements of N, and maps measurable sets of N to measurable sets of M by its inverse image.

$$\begin{aligned}&\text{-}\ \rightarrow^{\sigma}\ \text{-} \ :: \alpha\ \mathsf{measure} \Rightarrow \beta\ \mathsf{measure} \Rightarrow (\alpha \Rightarrow \beta)\ \mathsf{set}\\ &M \rightarrow^{\sigma} N = \{f \in \mathsf{space}\ M \rightarrow \mathsf{space}\ N \mid\\ &\qquad\qquad \forall A \in \mathsf{sets}\ N.\ \{x \in \mathsf{space}\ M.\ f\,x \in A\} \in \mathsf{sets}\ M\}\end{aligned}$$

For $f \in M \rightarrow^{\sigma} N$, we write that f is N-measurable on M, omitting either N or M when the reader can infer them from the context.

We write vsigma $\Omega\ M\ f$ to be the smallest σ-algebra on Ω, s.t. f is M-measurable. We introduce $\mathcal{B} ::$ bool measure, where all sets (i.e. $\{\{\}, \{\mathsf{True}\}, \{\mathsf{False}\}, \{\mathsf{True}, \mathsf{False}\}\}$) are measurable. Here $P \in M \rightarrow^{\sigma} \mathcal{B}$ is equal to $\{x \in \mathsf{space}\ M \mid P\ x\} \in \mathsf{sets}\ M$. We prefer the former notation as it fits with our other measurability rules.

For a type α in the type class topological-space, we define its Borel space borel $=$ sigma UNIV $\{S :: \alpha\ \mathsf{set} \mid \mathsf{open}\ S\}$. It is the smallest σ-algebra where each open set from the topology is measurable. From this follows immediately that every continuous function f is also measurable, i.e. $f \in$ borel $\rightarrow^{\sigma}$ borel. For many functions used in Isabelle/HOL we provide measurability rules, e.g. assuming f, g, and P are measurable then the functions $f + g$, $f - g$, $-f$, $f * g$, f/g, if P then f else g, min, max, abs, ln etc. are measurable on the corresponding measurability spaces. Functions like f/g

and ln are complicated cases as they are not continuous on the entire type. However, in Isabelle/HOL we have $f/0 = 0$ and ln is arbitrary but constant for non-positive values, i.e. $\ln x = \ln y$ for $x, y \leq 0$. To prove measurability of such a function f we cover the real numbers with a countable set $\mathcal{C}$ of measurable sets and prove that for each element $\Omega \in \mathcal{C}$ the function f is measurable when restricted to Ω.

A class of measurable functions we want to mention explicitly are the least and greatest fixed points. We know that for any sequence of measurable functions F into the extended reals and into the extended natural numbers, the supremum $\bigsqcup_i F\, i$ and the infimum $\bigsqcap_i F\, i$ is measurable. Now by Lemma 3, for a measurable and continuous function its fixed points are also measurable:

Lemma 6 (Measurable least and greatest fixed points into borel) *Fixed points of functions* $F :: (\alpha \Rightarrow \mathsf{ereal}) \Rightarrow (\alpha \Rightarrow \mathsf{ereal})$ *are measurable:*

$$\frac{F \in \Big(M \to^{\sigma} \mathsf{borel}\Big) \to \Big(M \to^{\sigma} \mathsf{borel}\Big)}{\begin{array}{l}\mathsf{sup\text{-}continuous}\ F \longrightarrow \mathsf{lfp}\ F \in M \to^{\sigma} \mathsf{borel} \\ \mathsf{inf\text{-}continuous}\ F \longrightarrow \mathsf{gfp}\ F \in M \to^{\sigma} \mathsf{borel}\end{array}}$$

2.2.1 Integration of Non-negative Functions

Each measure M gives rise to a non-negative Lebesgue integral $\int f\, \mathsf{d}M$ on non-negative borel-measurable function f on M into ereal. For measurable functions we prove the usual theorems about the Lebesgue integral, i.e. that it is monotone and linear and that dominated and monotone convergence holds.

Similar to monotone convergence, we prove a rule for convergence towards 0. However, it is possible to construct a sequence of functions with *infinite* integral where the *infimum* of the sequence is the constant 0-function. To avoid this we assume that starting from some index the sequence has a finite integral. Then monotone convergence towards 0 is easily derived from dominated convergence of the Lebesgue integral.

Lemma 7 (Monotone convergence towards 0**)**

$$\frac{\mathsf{antimono}\ F \qquad \forall i.\ F\, i \in M \to^{\sigma} \mathsf{borel} \qquad \exists i.\ \int F\, i\ \mathsf{d}M < \infty}{\int \bigsqcap_i F\, i\ \mathsf{d}M = \bigsqcap_i \int F\, i\ \mathsf{d}M}$$

For our applications we often integrate over functions defined as fixed points. From monotone convergence follows that the non-negative integral is continuous on measurable functions. Now, the transfer rules of Lemma 4 are not directly applicable. The problems are that (1) $\bot$ is not mapped to $\bot$, (2) the integral is only inf-continuous when the integral is bounded, and (3) the integral is only continuous on measurable functions. Luckily, with Lemma 5, we can derive the transfer rules specific for the non-negative integral:

Lemma 8 (Non-negative integral transfers least fixed points) *The integral of the least fixed point of* $f :: (\alpha \Rightarrow \mathsf{ereal}) \Rightarrow (\alpha \Rightarrow \mathsf{ereal})$ *can be transferred into the least*

fixed point of $g :: (\beta \Rightarrow \mathsf{ereal}) \Rightarrow (\beta \Rightarrow \mathsf{ereal})$*:*

$$\frac{\begin{array}{c}\forall s.\ \mathsf{sets}\,(M\ s) = \mathsf{sets}\,N \qquad \forall F\ s.\ 0 \le g\ F\ s \\ \mathsf{sup\text{-}continuous}\ f, g \qquad f \in \Big(N \to^{\sigma} \mathsf{borel}\Big) \to \Big(N \to^{\sigma} \mathsf{borel}\Big) \\ \forall s.\ \forall F \in N \to^{\sigma} \mathsf{borel}.\ F \le \mathsf{lfp}\ f \longrightarrow \int f\ F\ \mathsf{d}(M\ s) = g\ \Big(\lambda s.\ \int F\ \mathsf{d}(M\ s)\Big)\ s\end{array}}{\int \mathsf{lfp}\ f\ \mathsf{d}(M\ s) = \mathsf{lfp}\ g\ s}$$

Proof This is essentially an instance of Lemma 5, where $\alpha\ F\ s = \int F\ \mathsf{d}(M\ s)$ and $P\ f$ is $f \in N \to^{\sigma} \mathsf{borel}$. α is continuous on functions restricted to P by monotone convergence. □

We get a similar transfer rule for the greatest fixed point. We just need to be sure that the integral of f is always finite, otherwise we cannot apply Lemma 7.

Lemma 9 (Non-negative integral transfers greatest fixed points) *The integral of the greatest fixed point of* $f :: (\alpha \Rightarrow \mathsf{ereal}) \Rightarrow (\alpha \Rightarrow \mathsf{ereal})$ *can be transfered into the greatest fixed point of* $g :: (\beta \Rightarrow \mathsf{ereal}) \Rightarrow (\beta \Rightarrow \mathsf{ereal})$*:*

$$\frac{\begin{array}{c}\forall s.\ \mathsf{sets}\,(M\ s) = \mathsf{sets}\,N \qquad \forall s.\ \mathsf{emeasure}\,(M\ s)\,(\mathsf{space}\,(M\ s)) \neq 0 \\ \mathsf{inf\text{-}continuous}\ f, g \qquad f \in \Big(N \to^{\sigma} \mathsf{borel}\Big) \to \Big(N \to^{\sigma} \mathsf{borel}\Big) \\ \forall s.\ \forall F \in N \to^{\sigma} \mathsf{borel}.\ \int f\ F\ \mathsf{d}(M\ s) < \infty \\ \forall s.\ \forall F \in N \to^{\sigma} \mathsf{borel}.\ \int f\ F\ \mathsf{d}(M\ s) = g\ \Big(\lambda s.\ \int F\ \mathsf{d}(M\ s)\Big)\ s\end{array}}{\int \mathsf{gfp}\ f\ \mathsf{d}(M\ s) = \mathsf{gfp}\ g\ s}$$

Proof This is essentially an instance of the dual of Lemma 5, where $\alpha\ F\ s = \int F\ \mathsf{d}(M\ s)$ and $P\ f$ describes measurable and finitely integrable functions: $f \in N \to^{\sigma} \mathsf{borel}$ and $\forall s.\ \int f\ \mathsf{d}(M\ s) < \infty$. α is continuous on functions restricted to P by monotone convergence towards 0 (Lemma 7). □

The rules for fixed points (i.e. Lemmas 6, 8, and 9) are the main additions for the MDP theory presented in this paper.

2.2.2 Constructing Measure Spaces

Directly constructing measures, i.e. showing countable additivity of a measure, is often difficult. To simplify constructions, Isabelle's measure theory provides the following basic construction mechanisms:

– The *counting measure* on a space Ω returns the cardinality of the measured set: $\mathsf{emeasure}\,(\mathsf{count\text{-}space}\,\Omega)\,A = ||A||$ for a finite set $A \subseteq \Omega$. Here, every subset of Ω is measurable and hence $(\mathsf{count\text{-}space}\,\Omega \to^{\sigma} M) = (\Omega \to \mathsf{space}\ M)$. Integration over the counting space is a generalization of summation: for a non-negative function f with finite support $\{x \in \Omega \mid f\,x \neq 0\}$, we have

$$\int f\ \mathsf{d}(\mathsf{count\text{-}space}\,\Omega) = \sum_{x \in \Omega \wedge f\,x \neq 0} f\,x.$$

– For $f \in M \to^\sigma N$ given, the *distribution* (or push-forward measure) of f is for all $A \in \mathsf{sets}\, N$:

$$\mathsf{emeasure}\,(\mathsf{distr}\, M\, N\, f)\, A = \mathsf{emeasure}\, M\, \{x \in \mathsf{space}\, M \mid f\, x \in A\}$$

The integral of g over the distribution of f equals integral of $(g \circ f)$:

$$\int g\ \mathsf{d}(\mathsf{distr}\, M\, N\, f) = \int g \circ f\ \mathsf{d}M$$

– For a measure M and a measurable set S with a non-zero and finite measure $0 < \mathsf{emeasure}\ M\ S < \infty$, we can define the conditional probability $\mathsf{cprob}\, M\, S$:

$$\mathsf{emeasure}\,(\mathsf{cprob}\, M\, S)\, A = \frac{\mathsf{emeasure}\, M\, (S \cap A)}{\mathsf{emeasure}\, M\, S} \qquad \text{for all measurable sets } A$$

With this we write the conditional probability

$$\Pr(x\ \mathsf{in}\ M.\ P\, x \mid x \in S) = \Pr(x\ \mathsf{in}\ \mathsf{cprob}\, M\, S.\ P\, x).$$

– For an I-indexed family of probability measures M we define the *product measure* $\prod^\sigma_{i \in I}\ M\, i$, the index I is *not* restricted to be countable. On this product the projections $\lambda x.\ x\, j$ into $M\, j$ are independent, i.e. for a finite set $J \subseteq I$ and measurable sets $A\, j$ for $j \in J$:

$$\Pr\left(x\ \mathsf{in}\ \left(\prod^\sigma_{i \in I} M\, i\right).\ \forall j \in J.\ x\, j \in A\, j\right) = \prod_{j \in J} \Pr(x\ \mathsf{in}\ M\, j.\ x \in A\, j)$$

2.3 Giry Monad

An important construction mechanism for probabilistic programming language semantics is the *Giry monad* [20], providing a monadic structure on sub-probability measures. A sub-probability measure M is a measure with weight less or equal to 1, i.e. $\mathsf{emeasure}\ M\ (\mathsf{space}\ M) \leq 1$. For the Giry monad we introduce subprob-algebra, a measurable space on the set of sub-probability measures. Originally, it was formalized in Isabelle/HOL for the semantics of a simple programming language in Eberl *et al.* [16].

For this paper the Giry monad on sub-probability measures is used to introduced the *discrete* Giry monad on probability mass functions in the next subsection (where it is used to define Markov kernels), for the iteration rule of Markov chains (Theorem 1), and for the product equation of Markov chains (Lemma 19). Stating these rules in terms of measures, possible due to the Giry monad, gives less cluttered theorems and a straightforward method to derive the corresponding statements about integrals, measures and the almost everywhere quantifier. As an example we give Lemma 12 (a.k.a. Fubini's theorem) at the end of this section.

Definition 3 (Measurable space of sub-probabilities)

$\mathsf{subprob\text{-}algebra} :: \alpha\ \mathsf{measure} \Rightarrow \alpha\ \mathsf{measure}\ \mathsf{measure}$
$\mathsf{subprob\text{-}algebra}\ K =$
$\bigsqcup^\sigma_{A \in \mathsf{sets}\ K}\ \mathsf{vsigma}\ \{M \mid \mathsf{sets}\ M = \mathsf{sets}\ K \wedge \mathsf{subprob}\ M\}\ \mathsf{borel}\ (\lambda M.\ \mathsf{emeasure}\ M\ A)$

So for all $A \in \mathsf{sets}\ K$ we get $(\lambda M.\ \mathsf{emeasure}\ M\ A) \in \mathsf{subprob\text{-}algebra}\ K \to^\sigma \mathsf{borel}$. And the space of $\mathsf{subprob\text{-}algebra}\ K$ are all sub-probabilities over the measurable sets of K.

For the lifting of a measurable function $f \in M \to^\sigma N$ over a sub-probability M we use $\mathsf{distr}\ M\ N\ f$. The explicit annotation of the measurable space N is necessary as it can not be computed from f or M.

Lemma 10 (Distributions on random variables)

$$\frac{f \in M \to^\sigma N}{(\lambda\mu.\ \mathsf{distr}\ \mu\ N\ f) \in \mathsf{subprob\text{-}algebra}\ M \to^\sigma \mathsf{subprob\text{-}algebra}\ N}$$

We prove that distr is closed under composition and identity functions. Although we can not state this in Isabelle/HOL, this shows that subprob-algebra with distr is an endofunctor on the category of measurable spaces with measurable functions as morphisms. By introducing a bind and return operation we define the Giry monad:

Definition 4 (Giry monad)

$$\mathsf{bind} :: \alpha\ \mathsf{measure} \Rightarrow (\alpha \Rightarrow \beta\ \mathsf{measure}) \Rightarrow \beta\ \mathsf{measure} \quad \text{and}$$
$$\mathsf{return} :: \alpha\ \mathsf{measure} \Rightarrow \alpha \Rightarrow \alpha\ \mathsf{measure}.$$

Similar to distr, the first parameter to return is the measurable space of the returned measure. Note that $\mathsf{sets}\ (\mathsf{bind}\ M\ N)$ is determined by the range of N. This behaviour is different to distr and return which have an explicit parameter determining the measurable sets of the returned measure.

We do not show the concrete definitions of these functions; important are the following specifying properties. As they operate in the category of measurable spaces, we express their behaviour in terms of measurable functions and integrals.

$$\mathsf{return}\ M \in M \to^\sigma \mathsf{subprob\text{-}algebra}\ M$$

$$\frac{f \in M \to^\sigma \mathsf{subprob\text{-}algebra}\ N \qquad (\lambda(x,y).\ g\ x\ y) \in (M \times^\sigma N) \to^\sigma \mathsf{subprob\text{-}algebra}\ L}{(\lambda x.\ \mathsf{bind}\ (f\ x)\ (g\ x)) \in M \to^\sigma \mathsf{subprob\text{-}algebra}\ L}$$

$$\frac{f \in B \to^\sigma \mathsf{borel} \qquad N \in M \to^\sigma \mathsf{subprob\text{-}algebra}\ B}{\int f\ \mathsf{d}(\mathsf{bind}\ M\ N) = \int_x \int f\ \mathsf{d}(N\ x)\ \mathsf{d}M}$$

$$\frac{g \in M \to^\sigma \mathsf{borel} \qquad x \in \mathsf{space}\ M \qquad 0 \le g\ x}{\int g\ \mathsf{d}(\mathsf{return}\ M\ x) = g\ x}$$

From the last two equations we derive similar rules for the behaviour of bind and return under the measure of a set and the almost everywhere quantifier.

From these definitional equations we derive the monad laws.

Lemma 11 (Monad laws)

$$\mathsf{bind}\ M\ (\mathsf{return}\ M) = M$$

$$\frac{f \in M \to^\sigma \mathsf{subprob\text{-}algebra}\ N \qquad x \in \mathsf{space}\ M}{\mathsf{bind}\ (\mathsf{return}\ M\ x)\ f = f\ x}$$

$$\frac{f \in M \to^\sigma \mathsf{subprob\text{-}algebra}\ N \qquad g \in M \to^\sigma \mathsf{subprob\text{-}algebra}\ R}{\mathsf{bind}\ (\mathsf{bind}\ M\ f)\ g = \mathsf{bind}\ M\ (\lambda x.\ \mathsf{bind}\ (f\ x)\ g)}$$

As an application of the Giry monad, let us take a look at Fubini's theorem (i.e. two independent probabilistic choices can be swapped):

Lemma 12 (Fubini's theorem on the Giry monad)

$$\frac{\mathsf{subprob}\ M \qquad \mathsf{subprob}\ N}{\begin{array}{c}\mathsf{bind}\ M\ (\lambda x.\ \mathsf{bind}\ N\ (\lambda y.\ \mathsf{return}\ (M \times N)\ (x, y))) = \\ \mathsf{bind}\ N\ (\lambda y.\ \mathsf{bind}\ M\ (\lambda x.\ \mathsf{return}\ (M \times N)\ (x, y)))\end{array}}$$

The proof is done on the generating set of the product measure $M \times N$, i.e. we need to prove that both measures are equal on all rectangular sets $A \times B$ for $A \in \mathsf{sets}\ M$ and $B \in \mathsf{sets}\ N$.[1] From this theorem we can then derive its form for non-negative integrals, measures of sets and the almost everywhere quantifier using the previously stated definitional equations for bind and return. We use a similar approach for the trace space of Markov chains in Section 3.2.

2.4 Probability Mass Functions

An important subset of measures are the discrete probability measures. In a discrete measure all sets and functions are measurable, and the mass of a set is the sum of the mass of each element in the set. Hence, discrete probability measures are isomorphic to *probability mass functions (pmf)* which assign the probability to single elements.

Hölzl *et al.* [30] introduce the type of probability mass functions as a subtype of measures, allowing to transfer properties about measures to pmfs. This is done by using the lifting and transfer commands by Huffman and Kunčar [34]. The pmf-subtype is introduced using the **typedef**-command:

$$\begin{aligned}\alpha\ \mathsf{pmf} = \{M \mid\ & \mathsf{prob\text{-}space}\ M \wedge \mathsf{sets}\ M = \mathsf{UNIV} \wedge \\ & (\exists S.\ \mathsf{countable}\ S \wedge \mathsf{emeasure}\ M\ S = 1)\}\end{aligned}$$

This generates an injective representation function $\mathsf{measure\text{-}pmf} :: \alpha\ \mathsf{pmf} \Rightarrow \alpha\ \mathsf{measure}$. We declare it as a coercion function, hence omitting it in most cases. In particular, we write $\mathsf{measure}\ p\ A$ for $\mathsf{measure}\ (\mathsf{measure\text{-}pmf}\ p)\ A$. So, the probability mass of a value x is the measure of its singleton set $\{x\}$: We lift the projection function pmf, the support set set-pmf, the functorial operator map-pmf, and the monadic operators bind-pmf and return-pmf. The **lift-definition**-command [34] allows us to specify the behaviour of each of the functions on measures and then lifts them into functions on pmfs by introducing the necessary coercion functions (e.g. measure-pmf).

lift-definition pmf :: α pmf $\Rightarrow \alpha \Rightarrow$ real **is**
  $\lambda M\ x.$ measure M $\{x\}$
lift-definition set-pmf :: α pmf $\Rightarrow \alpha$ set **is**
  $\lambda M.$ $\{x \mid$ measure M $\{x\} \neq 0\}$
lift-definition map-pmf :: $(\alpha \Rightarrow \beta) \Rightarrow \alpha$ pmf $\Rightarrow \beta$ pmf **is**
  $\lambda f\ M.$ distr M (count-space UNIV) f
lift-definition bind-pmf :: $(\alpha \Rightarrow \beta$ pmf$) \Rightarrow \alpha$ pmf $\Rightarrow \beta$ pmf **is**
  bind
lift-definition return-pmf :: $\alpha \Rightarrow \alpha$ pmf **is**
  return (count-space UNIV)

[1] The proof in the Isabelle repository is derived from Fubini's theorem on σ-finite measures. The reason is that Isabelle's Giry monad is only available on sub-probability measures, hence following the presented proof would result in a weaker theorem.

For map-pmf, bind-pmf, and return-pmf we get the Giry monad for pmfs, now the monadic laws do not require measurability assumptions. The type of pmfs forms a bounded natural functor and hence can be used in (co)datatype definitions, for more information see [9, 30].

In the rest of this paper we will use set-pmf as a coercion, e.g. we write $t \in M$ instead of $t \in \mathsf{set\text{-}pmf}\ M$.

We provide the following discrete distributions:

Bernoulli For $0 \leq p \leq 1$ we have
$\mathsf{pmf}\ (\mathsf{bernoulli\text{-}pmf}\ p)\ \mathsf{True} = p$ and $\mathsf{pmf}\ (\mathsf{bernoulli\text{-}pmf}\ p)\ \mathsf{False} = 1 - p$.

Uniform For a non-empty, finite set A we have
$$\mathsf{pmf}\ (\mathsf{pmf\text{-}of\text{-}set}\ A)\ i = \begin{cases} \frac{1}{||A||} & \text{if } x \in A \\ 0 & \text{otherwise} \end{cases}$$

Binomial For n and $0 \leq p \leq 1$ we have
$\mathsf{pmf}\ (\mathsf{binomial\text{-}pmf}\ n\ p)\ k = \binom{n}{k} * p^k * (1-p)^{n-k}$

Geometric For $0 < p \leq 1$ we have
$\mathsf{pmf}\ (\mathsf{geometric\text{-}pmf}\ p)\ k = (1-p)^k * p$

Poisson For $0 < r$ we have
$\mathsf{pmf}\ (\mathsf{poisson\text{-}pmf}\ r)\ k = \frac{r^k}{k!} * \exp^{-r}$

Conditional For a pmf p and a set A with $\mathsf{measure}\ p\ A \neq 0$ we have
$$\mathsf{pmf}\ (\mathsf{cond\text{-}pmf}\ p\ A)\ x = \begin{cases} \frac{\mathsf{pmf}\ p\ x}{\mathsf{measure}\ p\ A} & \text{if } x \in A \\ 0 & \text{otherwise} \end{cases}$$

2.5 Stream Space

When modelling the trace spaces of Markov models we use coinductively defined streams, as formalized by Blanchette *et al.* [10]. This formalization also provides a shallow embedding of linear temporal logic (LTL) formulas. Streams are essentially sequences (i.e. functions of type $\mathsf{nat} \Rightarrow \alpha$), but constructed recursively from a head element and a stream tail. Similarly, Hurd [35] introduces a recursive view on Boolean sequences (equal to bool stream), however without bisimulation and a mechanism to define corecursive functions.

Streams are introduced using Isabelle's **codatatype**-command [9]:

$$\alpha\ \mathsf{stream} = \mathsf{SCons}\ (\mathsf{shd}\colon \alpha)\ (\mathsf{stl}\colon \alpha\ \mathsf{stream})$$

The **codatatype**-command automatically defines the map function $\mathsf{smap} :: (\alpha \Rightarrow \beta) \Rightarrow \alpha\ \mathsf{stream} \Rightarrow \beta\ \mathsf{stream}$, and a set projection $\mathsf{sset} :: \alpha\ \mathsf{stream} \Rightarrow \alpha\ \mathsf{set}$. The function $\mathsf{to\text{-}stream} :: (\mathsf{nat} \Rightarrow \alpha) \Rightarrow \alpha\ \mathsf{stream}$ converts sequences to streams, $\omega\ !!\ n$ selects the n-th element of ω, $\mathsf{sdrop}\ n\ \omega$ drops the first n-elements from a stream ω, and $\mathsf{streams}\ S = \{\omega \mid \mathsf{sset}\ \omega \subseteq S\}$ the set of all streams on S.

Blanchette *et al.* [10] introduce predicate transformers on streams to express LTL formulas. These transformers are composable as the quantifiers are not functions over state predicates, but over stream predicates. We use the usual notation for predicates, i.e. $\Psi \sqcap \Phi$ for conjunction, $\Psi \sqcup \Phi$ for disjunction, and $-\Psi$ for negation. We define $(\Psi\ \mathsf{impl}\ \Phi)\ \omega \longleftrightarrow (\Psi\ \omega \longrightarrow \Phi\ \omega)$, $\mathsf{nxt}\ \Phi\ \omega \longleftrightarrow \Phi\ (\mathsf{stl}\ \omega)$, and $\mathsf{HLD}\ \Phi\ \omega \longleftrightarrow \mathsf{shd}\ \omega \in \Phi$. Fig. 1 lists the predicate transformers (i.e. Always, Eventually, and Strong Until) and counting operators (i.e. Count Occurence and First Occurence). They are defined in

Always

alw	:: $(\alpha$ stream $\Rightarrow$ bool$) \Rightarrow (\alpha$ stream $\Rightarrow$ bool$)$
alw $\Phi\ \omega$	$\overset{\text{gfp}}{=}\ \Phi\ \omega \wedge$ alw Φ (stl ω)

Eventually

ev	:: $(\alpha$ stream $\Rightarrow$ bool$) \Rightarrow (\alpha$ stream $\Rightarrow$ bool$)$
ev $\Phi\ \omega$	$\overset{\text{lfp}}{=}\ \Phi\ \omega \vee$ ev Φ (stl ω)

Strong Until

- suntil -	:: $(\alpha$ stream $\Rightarrow$ bool$) \Rightarrow (\alpha$ stream $\Rightarrow$ bool$) \Rightarrow (\alpha$ stream $\Rightarrow$ bool$)$
(Ψ suntil Φ) ω	$\overset{\text{lfp}}{=}\ \Phi\ \omega \vee \big(\Psi\ \omega \wedge (\Psi$ suntil $\Phi)$ (stl ω)$\big)$

Count Occurrences

scount	:: $(\alpha$ stream $\Rightarrow$ bool$) \Rightarrow (\alpha$ stream $\Rightarrow$ enat$)$
scount $P\ \omega$	$\overset{\text{lfp}}{=}$ if $P\ \omega$ then scount P (stl ω) $+ 1$ else scount P (stl ω)

First Occurrence

sfirst	:: $(\alpha$ stream $\Rightarrow$ bool$) \Rightarrow (\alpha$ stream $\Rightarrow$ enat$)$
sfirst $P\ \omega$	$\overset{\text{lfp}}{=}$ if $P\ \omega$ then 0 else sfirst P (stl ω) $+ 1$

Fig. 1 Stream Operations

terms of greatest fixed points (gfp) and least fixed points (lfp), giving us certain theorems for free, e.g. measurability.

The functions suntil, scount, sfirst, and the measure on stream spaces (introduced in the next section) were explicitly developed for the work presented here.

2.6 Stream Measure

Before we start to use streams in probability theory we need a measurable space and a measure on it. For this we introduce the canonical measure on streams, for which all (co)recursively defined functions are measurable, and for which we can define a measure s.t. all elements at different positions are independently distributed. Such a measure is isomorphic to the infinite product measure $\prod^{\sigma}_{n::\text{nat}} M$, so we define it as its embedding:

Definition 5 (Stream Measure)

stream-space :: α measure $\Rightarrow \alpha$ stream measure
stream-space $M =$
distr $\big(\prod^{\sigma}_{n::\text{nat}} M\big)$ $\big($vsigma (streams (space M)) (op !!) $(\prod^{\sigma}_{n::\text{nat}} M)\big)$ to-stream

By definition, the measurable space of stream-space is the smallest measurable space s.t. !! is measurable. Hence each function which makes !! measurable is also measurable:

Lemma 13 (Basic Measurability on the Stream Space)

$$(\lambda\omega.\ \omega\ !!\ i) \in \text{stream-space}\ M \rightarrow^{\sigma} M \qquad \frac{\forall i.\ (\lambda x.\ f\ x\ !!\ i) \in N \rightarrow^{\sigma} M}{f \in N \rightarrow^{\sigma} \text{stream-space}\ M}$$

The measurability of $\mathsf{shd}, \mathsf{stl}, \mathsf{sdrop}, x \cdot \omega, \mathsf{smap}$, and $\mathsf{to\text{-}stream}$ is easily derived by these two rules. The measurability of the stream operations in Fig. 1 are proved using Lemma 6. Moreover, Lemma 13 allows us to prove the following coinduction rule: a function is stream-measurable if it is in a set F, which is closed under composition with stl and where all functions are measurable under shd.

Lemma 14 (Coinduction on Stream Spaces)

$$\frac{f \in F \qquad \forall f \in F.\ \mathsf{shd} \circ f \in N \to^{\sigma} M \qquad \forall f \in F.\ (\mathsf{stl} \circ f) \in F}{f \in N \to^{\sigma} \mathsf{stream\text{-}space}\ M}$$

For the canonical measure on stream spaces (i.e. the countably infinite product space of M), we get an iteration rule on the integral. This is a helpful rule to compute the integral of a function f defined recursively over the input stream.

Lemma 15 (Iterative Stream Space Measure)

$$\frac{f \in \mathsf{stream\text{-}space}\ M \to^{\sigma} \mathsf{borel}}{\int f\ \mathsf{d}(\mathsf{stream\text{-}space}\ M) = \int_x \int_\omega f\ (x \cdot \omega)\ \mathsf{d}(\mathsf{stream\text{-}space}\ M)\ \mathsf{d}M}$$

3 Markov Chains

In this section we introduced the necessary theories to model and analyze Markov chains. As mentioned in the introduction there are two different views on discrete-time Markov chains:

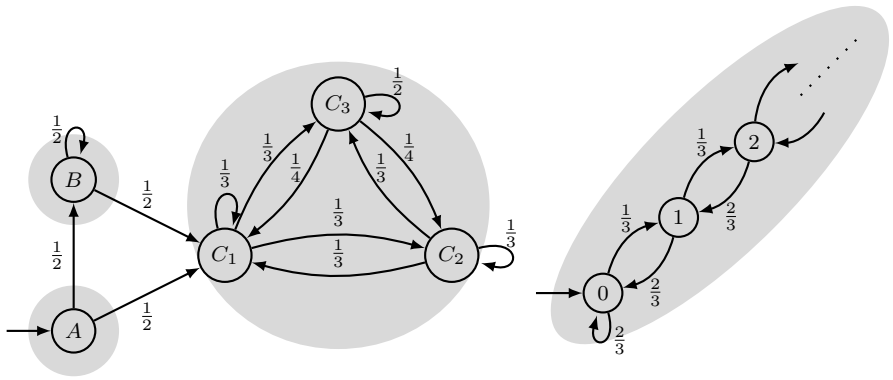
- As a probabilistic transition system, explicitly given as the transitions *per state*. The Markov chains depicted in Fig. 2 can be straightforwardly translated into such transition systems.
- As a discrete stochastic process which is memoryless and time-homogeneous.

We introduce Markov chains as probabilistic transition systems K and for each K we construct a probability space T on its infinite traces. To relate both views, in Section 3.5, we will show that each discrete stochastic process implies a probabilistic transition system K, and vice versa.

Let us take a look at properties of Markov chains we want to analyze. As examples we use the Markov chains depicted in Fig. 2. Fig. 2 (a) is a irregular but finite MC, and Fig. 2 (b) shows a so called birth-death process with an infinite state space. In the later, the states value increases with probability 1/3, or decreases by 2/3 (and in the case of state 0 it stays 0 with probability 2/3). In Section 3.7 we will formalize these Markov chains in Isabelle/HOL.

The formalization presented in this paper is oriented towards questions of the following kind (e.g. about the Markov chains in Fig. 2):

Q1 When we start (b), what is the probability that it comes back to 0?
Q2 Is it possible that (a) never reaches C_3 or that (b) never reaches a state n?
Q3 When we start (b), what is the average time until it reaches 0 again?
Q4 When (a) runs for a long time, is C_3 more likely to occur than C_1?



(a) A finite process (b) An infinite birth-death process

Fig. 2 Two Markov chains

We wish to express these questions in Isabelle/HOL as concise as possible using the stream operators introduced in Fig. 1. To achieve this, we use the theory of Markov chains to express the probability of a set of infinite traces, and integrals on them. We recommend readers unfamiliar with this topic the book by Woess [51], an introductory text book which we use for the analysis part of our formalization.

The theory of Markov chains allows us to reduce some of these questions to an analysis of the graph of non-zero transitions, and to the solution of a linear equation system (in the finite case) or to finding a least fixed point (in the infinite case). Alternatively, we can use generating functions to analyze them. These are also the kind of questions usually answered by probabilistic model checkers on finite models.

3.1 Transition Probabilities

The transition system of a Markov chain specifies for each of its states a distribution of the follower states. Such a transition system is completely described by a function K (the Markov kernel):

$$K :: \sigma \Rightarrow \sigma \ \mathsf{pmf}$$

There is no restriction on the type σ, or on the distributions $K\ s$. An implicit restriction given by the type $\sigma\ \mathsf{pmf}$ is that for each s, the support set of $K\ s$ is countable and non-empty. This has the nice side-effect that in Isabelle/HOL no further assumptions are necessary about K, everything is encoded in the type. In the rest of the section we implicitly assume a $K :: \sigma \Rightarrow \sigma\ \mathsf{pmf}$.

For a state s, we call the set $\mathsf{set\text{-}pmf}\,(K\,s)$, i.e. the follower states with non-zero probability, the *enabled* states in s (for readability, we will often write $t \in K\,s$ instead of $t \in \mathsf{set\text{-}pmf}\ (K\,s)$). The enabled transitions define a directed graph, the vertices are the states of the Markov chain, and (s, t) is an edge when $t \in K\,s$. A state t is accessible from s if it is reachable in finitely many steps:

Definition 6 (Accessible states) The states accessible from s are defined as reflexive and transitive closure of K:

$$\begin{aligned}&\mathsf{acc\text{-}on} && :: \sigma\ \mathsf{set} \Rightarrow \sigma \times \sigma\ \mathsf{set}\\ &\mathsf{acc\text{-}on}\ S && = \{(s,t) \mid t \in K\, s \wedge t \in S\}^*\\ &\mathsf{acc} && = \mathsf{acc\text{-}on}\ \mathsf{UNIV}\end{aligned}$$

Here R^* is the reflexive, transitive closure of a relation R. The $\mathsf{acc\text{-}on}\ S$ relation additionally enforces that we want to stay in the subset S of the state space.

A stream $\omega :: \sigma\ \mathsf{stream}$ of states is enabled from a starting state s when each state in the stream is enabled by the previous one.

Definition 7 (Enabled stream) We define $\mathsf{enabled} :: \sigma \Rightarrow \sigma\ \mathsf{stream} \Rightarrow \mathsf{bool}$ coinductively, over the following rule:

$$\frac{\mathsf{enabled}\, t\, \omega \qquad t \in K\, s}{\mathsf{enabled}\, s\, (t \cdot \omega)}$$

3.2 Probability and Expectation on Markov Chain Traces

Now, the probability that a Markov chain defined by K visits the sequence $s_0, s_1, \ldots, s_n$ of states is the product of $\mathsf{pmf}\,(K\, s_i)\, s_{i+1}$ for i from 0 to $n-1$. The probability for an infinite sequence is 0 (assuming $\mathsf{pmf}\,(K\, s)\, t < 1$ for infinitely many transitions). Fortunately, measure theory allows us to assign a meaningful probability to sets of traces. So, to reason about the infinite behaviour of Markov chains we construct the probability measure on traces (streams of states).

Such infinite traces occur when we reason about fairness, safety, liveness or other limiting behaviour of Markov chains. To handle this, we present in this section an iterative view on the trace space, which matches our coalgebraic view of streams. As mentioned, a Markov chain can also be seen as a stochastic process X, i.e. a probability measure with a sequence of random variables X_t representing the state of the Markov chain at time t. In Section 3.5 we show that the coalgebraic view is equivalent to the stochastic process view. The construction resembles [26], except that we now use pmfs to represent the transition probabilities. Also, the cardinality of the state space is not restricted.

Before we construct the probability measure, we first fix the measurable space of traces S to be the measurable space on streams of states, i.e. the smallest σ-algebra s.t. the head and tail projections are measurable. For this we use the stream space defined in Section 2.5.

$$\begin{aligned}&S :: \sigma\ \mathsf{stream\ measure}\\ &S = \mathsf{stream\text{-}space}\,(\mathsf{count\text{-}space}\,\mathsf{UNIV})\end{aligned}$$

On the measurable space S, we will introduce the trace space $T\, s$ with s as the starting state. That trace space $T\, s$ should give rise to the iteration rule Eq. (2).

How can we construct the probability measure $T\, s$? Our first option is to use the method by Hurd [35] and define the algebra of finite unions of cylinders, i.e. sets starting with the same prefix. We have Caratheodory's extension theorem available, but operating on finite unions of cylinders is very cumbersome. Introductions to the theory of Markov chains, like in Kwiatkowska *et al.* [38] for probabilistic model checking, use a semi-ring of cylinder sets. This already simplifies the existence proof for the

trace space, and in Isabelle/HOL Caratheodory's extension theorem is available on semi-rings too. However we still need to show that the premeasure (i.e. the measure on cylinders) is countably additive, which is quite involved. We use an alternative construction composing the stream measure and the push-forward measure.

First, we introduce the auxiliary constants D (the stream space of decisions $\sigma \Rightarrow \sigma$) and walk. Then we define the trace measure $T\,s$ as the push-forward measure of walk over D. Finally, we characterize $T\,s$ with Theorem 1. After this D and walk are not needed anymore.

$$\begin{aligned} &D :: (\sigma \Rightarrow \sigma)\ \mathsf{stream\ measure} \\ &D = \mathsf{stream\text{-}space}\,(\textstyle\prod^{\sigma}_{s \in \mathsf{UNIV}}\ K\,s) \end{aligned}$$

With this we avoid direct invocation of Caratheodory's extension theorem, hence we also circumvent proving countable additivity[2].

The probability measure D provides us random streams of mappings from state to a following state. We walk along the random stream of mappings, and map the current state to the next state. An invariant we need to maintain is to return an enabled stream. This is necessary, as otherwise the possible values for s in $\mathsf{walk}\ s$ are uncountable. Fortunately, restricting to enabled states does not changed the probabilities as non-enabled states have a probability of 0. To maintain the invariant, we use Hilbert-choice, written $\varepsilon t.\ t \in K\,s$, to guarantee that the returned stream is enabled. We define walk by primitive corecursion on streams:

$$\begin{aligned} &\mathsf{walk} && :: \sigma \Rightarrow (\sigma \Rightarrow \sigma)\ \mathsf{stream} \Rightarrow \sigma\ \mathsf{stream} \\ &\mathsf{shd}\,(\mathsf{walk}\,s\,\omega) && = \mathsf{force\text{-}hd}\ s\ \omega \\ &\mathsf{stl}\,(\mathsf{walk}\,s\,\omega) && = \mathsf{walk}\ (\mathsf{force\text{-}hd}\ s\ \omega)\,(\mathsf{stl}\,\omega) \\ &\quad \textbf{where}\ \mathsf{force\text{-}hd}\ s\ \omega && = \ \mathsf{if}\ \mathsf{shd}\,\omega\,s \in K\,s\ \mathsf{then}\ \mathsf{shd}\,\omega\,s\ \mathsf{else}\ \varepsilon t.\ t \in K\,s \end{aligned}$$

Now, the trace space $T\,s$ is simply a mapping of walk over all probabilistic decisions D:

$$\begin{aligned} &T && :: \sigma \Rightarrow \sigma\ \mathsf{stream\ measure} \\ &T\,s && = \mathsf{distr}\,D\,S\,(\mathsf{walk}\,s) \end{aligned}$$

Now, showing $\mathsf{enabled}\,s\,(\mathsf{walk}\,s\,\omega)$, for all s and ω, is a simple coinductive proof. The existence of an enabled next state is guaranteed by the pmf $K\,s$, as its support set is never empty, i.e. $\mathsf{set\text{-}pmf}\ (K\,s) \neq \emptyset$.

Enabledness is also important to show the measurability of walk. For this we prove the following auxiliary rule:

$$\frac{g \in D \to^{\sigma} \mathsf{count\text{-}space}\,(\mathsf{acc}\ ``\ \{s\})}{(\lambda\omega.\ \mathsf{walk}\ (g\ \omega)\ (\mathsf{sdrop}\,n\,\omega)) \in D \to^{\sigma} S}$$

This is done by coinduction with Lemma 14, where $F\ f$ is instantiated with $\exists n, g \in D \to^{\sigma} \mathsf{count\text{-}space}\,(\mathsf{acc}\ ``\{s\}).f = (\lambda\omega.\ \mathsf{walk}\ (g\,\omega)\ (\mathsf{sdrop}\,n\,\omega))$. In the coinduction step, $g$ is composed with the choice of the next step. The range of $g$ is $\mathsf{acc}\ ``\ \{s\}$, the set of states accessible from s, which is countable. So $g\,\omega$ is replaced by some constant t, then

[2] A more powerful approach is the extension theorem by Ionescu-Tulcea, e.g. as presented in [43, 46], which allows Markov kernels on arbitrary measurable spaces. While we formalized this extension theorem in Isabelle/HOL it was not available for the formalization of Markov chains. Our formalized version of Ionescu-Tulcea is available in Isabelle 2016.

the shd and stl projections of walk are measurable. Finally, by instantiating $g\,\omega = s$ and $n = 0$ we get measurability of walk s:

$$\mathsf{walk}\, s \in D \to^{\sigma} S$$

With this, the trace space $T\,s$ is well defined as a probability measure, due to it being a push-forward measure of a measurable function. Hence we derive the central rules for our Markov chain formalization: the iteration rules of the trace space.

Theorem 1 (The Trace Space is an Iterative Probability Measure) *From measurability it immediately follows that the trace space is a probability measure and almost everywhere enabled:*

$$\mathsf{prob\text{-}space}\,(T\,s) \qquad AE\ \omega\ \mathsf{in}\ T\,s.\ \mathsf{enabled}\,s\,\omega$$

The central rules for T are the iteration rules on T itself (using $\gg\!=$ syntax for the monadic bind), under integration, under measure, and under the almost everywhere quantifier:

$$T\ s\ =\ K\ s \gg\!= \lambda t.\ T\ t \gg\!= \lambda\omega.\ \mathsf{return}\ S\ (t{\cdot}\omega)$$

$$\frac{f \in S \to^{\sigma} \mathsf{borel}}{\int f\ \mathsf{d}(T\,s) = \int_t \int_\omega f\,(t \cdot \omega)\ \mathsf{d}(T\,t)\ \mathsf{d}(K\,s)}$$

$$\frac{P \in S \to^{\sigma} \mathcal{B}}{\Pr\,(\omega\ \mathsf{in}\ T\,s.\ P\,\omega) = \int_t \Pr\,(\omega\ \mathsf{in}\ T\,t.\ P\,(t{\cdot}\omega))\ \ \mathsf{d}(K\,s)}$$

$$\frac{P \in S \to^{\sigma} \mathcal{B}}{AE\ \omega\ \mathsf{in}\ T\,s.\ P\,\omega \longleftrightarrow \forall t \in K\,s.\ AE\ \omega\ \mathsf{in}\ T\,t.\ P\,(t{\cdot}\omega)}$$

We first prove the iteration rule on the Giry monad. Then the rules for the non-negative integral, the measure operator, and the almost everywhere quantifier follow immediately. After deriving these rules, walk and the definition of T is not needed anymore.

We can even show that the iteration rule on measures (i.e. the first equation of Theorem 1) uniquely determines the trace space. We prove that each family M of probability measures on S, which are closed under iteration with K, are equal to T:

Theorem 2 (Iterative construction of T)

$$\frac{\begin{array}{c}\forall s.\ \mathsf{sets}\ (M\ s) = \mathsf{sets}\ S \qquad \forall s.\ \mathsf{prob\text{-}space}\ (M\ s) \\ \forall s.\ M\ s\ =\ K\ s \gg\!= \lambda t.\ M\ t \gg\!= \lambda\omega.\ \mathsf{return}\ S\ (t{\cdot}\omega)\end{array}}{T = M}$$

An important auxiliary lemma we obtain is that a stream predicate P holds almost always, if it holds at each accessible state:

Lemma 16

$$\frac{P \in S \to^{\sigma} \mathcal{B} \qquad \forall t.\ (s,t) \in \mathsf{acc} \longrightarrow AE\ \omega\ \mathsf{in}\ T\,t.\ P\,\omega}{AE\ \omega\ \mathsf{in}\ T\,s.\ \mathsf{alw}\,P\,\omega}$$

3.3 Fairness, Reachability, and Hitting Time

When analyzing LTL formulas on Markov chains, an important questions is: what is the probability that the Markov chain stays in a set of states X until a predicate P holds on the remaining stream. Here, P is a predicate on a stream so it can be an LTL formula again. An important part of model checking is reachability analysis, i.e. the predicate P just states that the first state in the stream is not in X. It is well-known that the solution of a probabilistic reachability problem is a least fixed point. We generalize this result from finite to infinite state spaces and arbitrary predicates P. The second assumption states that on each prefix on which X holds, P holds at most once.

Lemma 17 (Probability of suntil)

$$\frac{P \in S \to^{\sigma} \mathcal{B} \qquad \forall t.\ AE\ \omega\ \mathsf{in}\ T\,t.\ \neg(P \sqcap \mathsf{HLD}\,X \sqcap \mathsf{nxt}\ (\mathsf{HLD}\,X\ \mathsf{suntil}\,P))\ \omega}{\begin{aligned}&\Pr(\omega\ \mathsf{in}\ T\,s.\ (\mathsf{HLD}\ X\ \mathsf{suntil}\ P)\,\omega) = \\ &\quad \mathsf{lfp}\left(\lambda F\,s.\ \Pr(\omega\ \mathsf{in}\ T\,s.\ P\,\omega) + \int_t F\,t * \mathsf{indicator}\,X\,t\ \mathsf{d}(K\,s)\right)\ s\end{aligned}}$$

A specialized version of this theorem tells us, that the probability for P in state t is independent of the reachability of t:

Corollary 1

$$\frac{P \in S \to^{\sigma} \mathcal{B}}{\begin{aligned}&\Pr\left(\omega\ \mathsf{in}\ T\,s.\ \mathsf{HLD}\,(-\{t\})\ \mathsf{suntil}\ \left(\mathsf{HLD}\,\{t\} \sqcap \mathsf{nxt}\,P\right)\omega\right) = \\ &\quad \Pr\left(\omega\ \mathsf{in}\ T\,s.\ \mathsf{ev}\left(\mathsf{HLD}\,\{t\}\right)\omega\right) * \Pr(\omega\ \mathsf{in}\ T\,t.\ P\,\omega)\end{aligned}}$$

A central property of Markov chains is fairness, i.e. when a state t is visited infinitely often, then the transition to another state $t' \in K\ t$ is also taken infinitely often. This holds for all Markov chains!

Definition 8 (Fair Stream)

$$\begin{aligned}&\mathsf{fair} \quad :: \sigma \Rightarrow \sigma \Rightarrow \sigma\ \mathsf{stream} \Rightarrow \mathsf{bool} \\ &\mathsf{fair}\,t\,t' = \mathsf{alw}\left(\mathsf{ev}\left(\mathsf{HLD}\,\{t\}\right)\right)\ \mathsf{impl}\ \mathsf{alw}\left(\mathsf{ev}\left(\mathsf{HLD}\,\{t\} \sqcap \mathsf{nxt}\,(\mathsf{HLD}\,\{t'\})\right)\right)\end{aligned}$$

Theorem 3 (Markov Chains are Fair)

$$t' \in K\,t \longrightarrow AE\ \omega\ \mathsf{in}\ T\,s.\ \mathsf{fair}\,t\,t'\,\omega$$

The proof for this theorem is based on Baier's habilitation thesis [6]. We present our own version, as it shows a nice application of our fixed point approach, and the previously introduced theorems. In [33] another version of the proof was used. With the change to the fixed point approach its size reduced by a third: from 89 lines to 60 lines.

Proof We define N to be the traces with infinitely many ts, but t is never followed by t':

$$N = \mathsf{alw}\ (\mathsf{ev}\ (\mathsf{HLD}\ \{t\})) \sqcap \mathsf{alw}\ (-(\mathsf{HLD}\ \{t\} \sqcap \mathsf{nxt}\ (\mathsf{HLD}\ \{t'\})))$$

Now we prove the following inequality using greatest fixed point coinduction:

$$\Pr(\omega\ \mathsf{in}\ T\,s.\ N\ \omega) \leq \mathsf{gfp}\ (\lambda x.\ 1 \sqcap x * (1 - \mathsf{pmf}\,(K\,t)\,t')) \qquad \text{for all } s \tag{3}$$

As coinduction hypothesis we may assume $\forall s.\ \Pr(\omega \text{ in } T\ s.\ N\ \omega) \leq x$. We do the following calculation:

$$\begin{array}{ll}
& \Pr(\omega \text{ in } T\ s.\ N\ \omega) \\
\leq & \quad \langle \textit{by induction on the first } \mathsf{ev}\ (\mathsf{HLD}\ \{t\}) \textit{ hit by } N \rangle \\
& \Pr\left(\omega \text{ in } T\ s.\ \mathsf{HLD}\ (-\{t\})\ \mathsf{suntil}\ (\mathsf{HLD}\ \{t\} \sqcap \mathsf{nxt}\ (\mathsf{HLD}\ (-\{t'\}) \sqcap \mathsf{nxt}\ N)\ \omega\right) \\
= & \quad \langle \textit{by Corollary 1} \rangle \\
& \Pr(\omega \text{ in } T\ s.\ \mathsf{ev}\ (\mathsf{HLD}\ \{t\})\ \omega)) * \Pr\left(\omega \text{ in } T\ t.\ (\mathsf{HLD}\ (-\{t'\}) \sqcap \mathsf{nxt}\ N)\ \omega\right) \\
\leq & \quad \langle \textit{by Theorem 1 and } \Pr(\ldots) \leq 1 \rangle \\
& 1 * \int_s \Pr(\omega \text{ in } T\ s.\ N\ \omega) * \mathsf{indicator}\ (-\{t'\})\ s\ \mathsf{d}(K\ t) \\
\leq & \quad \langle \textit{by coinduction hypothesis} \rangle \\
& \int_s (1 \sqcap x) * \mathsf{indicator}\ (-\{t'\})\ s\ \mathsf{d}(K\ t) \\
= & \quad \langle \textit{integration over a non-negative constant} \rangle \\
& (1 \sqcap x) * (1 - \mathsf{pmf}\ (K\ t)\ \{t'\})
\end{array}$$

With this we prove Eq. (3). By arithmetic we show that each fixed point of the right-hand side of Eq. (3) is less or equal to 0, and hence the probability for N is 0. Hence N almost never holds: $AE\ \omega \text{ in } T\ s.\ \mathsf{alw}\ (-N)\ \omega$, by Lemma 16. From this follows $AE\ \omega \text{ in } T\ s.\ \mathsf{fair}\ t\ t'\ \omega$. □

A property which follows from fairness is reachability of a set B from s. Suppose that we can always reach B, i.e. no matter where we go there is always an accessible state in B. And assume the set of states accessible from s avoiding B is finite. Then we know that the Markov chain reaches B with probability 1. This is simply proved by contradiction on a path never reaching B. Then by the pigeon hole principle, a state in $-B$ will be reached infinitely often, proving that no state in direction towards B will ever be taken, contradicting fairness.

Lemma 18 (Reachability)

$$\frac{\mathsf{finite}\ (\mathsf{acc\text{-}on}\ (-B)\ ``\ \{s\}) \qquad \forall s.\ (s,t) \in \mathsf{acc\text{-}on}\ (-B) \longrightarrow \exists t' \in B.\ (t,t') \in \mathsf{acc}}{AE\ T\ s \text{ in } \omega.\ \mathsf{ev}\,(\mathsf{HLD}\ B)\,\omega}$$

This lemma tells us that deciding reachability with probability 1 is reducible to computing its bottom SCCs (strongly connected components). This constitutes an important optimization for probabilistic model checking on Markov chains.

When the set B is always reached, we get a similar result for the expectation of the hitting time.

Theorem 4 (Hitting Time Expectation)

$$\frac{s \notin B \qquad \mathsf{finite}\left(\mathsf{acc\text{-}on}\ (-B)\ ``\ \{s\}\right) \qquad AE\ \omega \text{ in } T\ s.\ \mathsf{ev}\,(\mathsf{HLD}\ B)\,\omega}{\int \mathsf{sfirst}\,(\mathsf{HLD}\ B)\ \mathsf{d}(T\ s) < \infty}$$

Similar to Theorem 3 the proof of Theorem 4 was updated from an older version to a version using fixed points. The previous version was used for [33], and the change reduced its size by ≈ 40%: from 181 lines to 106 lines. The theorem itself is an important result for probabilistic model checking when computing rewards: it allows us to decide for which states the rewards are finite, and then compute the finite rewards by a system of linear equations.

3.4 Product Construction

Markov chains have a nice composition property: from two Markov kernels K_1 and K_2, we can construct a product Markov kernel K_p. This will be necessary in the proof of Theorem 11 about stationary distributions.

Definition 9 (Markov Kernel Product)

$$\begin{aligned} K_p &:: (\sigma_1 \times \sigma_2) \Rightarrow (\sigma_1 \times \sigma_2)\ \mathsf{pmf} \\ K_p\ (a, b) &= \mathsf{bind\text{-}pmf}\ (K_1\ a)\ (\lambda a'.\ \mathsf{bind\text{-}pmf}\ (K_2\ b)\ (\lambda b'.\ \mathsf{return\text{-}pmf}\ (a', b'))) \end{aligned}$$

For the kernel K_i, we call the corresponding trace space T_i and the measurable space S_i for $i = 1, 2, p$. For K_p, the behaviour on the first part K_1 of our state is now independent of the second part K_2. This should transfer to the trace space, i.e. the first parts of the traces should be independently distributed of the second parts. We show with Theorem 2 and Lemma 12, that the construction of K_p lifts to T_p.

Lemma 19 (Markov Chain Product)

$$T_p\ (a, b) \quad = \quad T_1\ a \gg\!= \lambda\omega_a.\ T_2\ b \gg\!= \lambda\omega_b.\ \mathsf{return}\ S_p\ (\mathsf{szip}\ (\mathsf{force}\ a\ \omega_a)\ (\mathsf{force}\ b\ \omega_b))$$

Here $\mathsf{szip} :: \alpha\ \mathsf{stream} \Rightarrow \beta\ \mathsf{stream} \Rightarrow (\alpha \times \beta)\ \mathsf{stream}$ is the zip-operation on streams. An annoyance is the occurrence of $\mathsf{force}\ s\ \omega$, which forces the resulting trace to be enabled starting with s. Either it returns ω if it is already enabled, or it selects similar to walk an arbitrary enabled stream. This is necessary to show the measurability of szip.

Now we can show that the projections are independent:

Corollary 2 (Independence of the Markov Chain Product)

$$\frac{P_1 \in S_1 \to^\sigma \mathcal{B} \qquad P_2 \in S_2 \to^\sigma \mathcal{B}}{\begin{aligned} &\Pr(\omega\ \mathsf{in}\ T_p\ (x_1, x_2).\ P_1\ (\mathsf{smap}\ \mathsf{fst}\ \omega) \wedge P_2\ (\mathsf{smap}\ \mathsf{snd}\ \omega)) = \\ &\quad \Pr(\omega\ \mathsf{in}\ T_1\ x_1.\ P_1\ \omega) \cdot \Pr(\omega\ \mathsf{in}\ T_2\ x_2.\ P_2\ \omega) \end{aligned}}$$

3.5 Trace Space as Stochastic Process

One way to represent Markov chains is as a *stochastic process* $X :: \mathsf{nat} \Rightarrow \alpha \Rightarrow \sigma$, i.e. a family of functions $(X_n)_{n \in \mathsf{nat}}$ where X_n is measurable on a probability space M. The space M represents the probabilistic behavior of our entire "world" and with X we observe the behavior of our Markov chain.

Axiomatic Definition as Stochastic Processes We introduce the locale `DTMC`. It assumes a stochastic process X and a probability space M which has the properties of a Markov

chain, i.e. it is memoryless and time-homogeneous.

locale DTMC = prob-space $(M :: \alpha$ measure$)$ +
  fixes $S :: \sigma$ set **and** $X :: \mathsf{nat} \Rightarrow \alpha \Rightarrow \sigma$
  assumes countable S **and** $\forall n.\ AE\ \omega$ in $M.\ X\ n\ \omega \in S$
  assumes $\forall n.\ X\ n \in M \rightarrow^{\sigma}$ count-space UNIV

  assumes $\forall n\ s\ t.$ — The stochastic process X is *memoryless*:
    $\Pr\left(x \text{ in } M.\ \forall n' \leq n.\ X\ n'\ x = t\ n'\right) \neq 0 \longrightarrow$
    $\Pr\left(x \text{ in } M.\ X\ (n+1)\ x = s \mid \forall n' \leq n.\ X\ n'\ x = t\ n'\right) =$
    $\Pr\left(x \text{ in } M.\ X\ (n+1)\ x = s \mid X\ n\ x = t\ n\right)$

  assumes $\forall n\ n'\ s\ t.$ — The stochastic process X is *time-homogeneous*:
    $\Pr\left(x \text{ in } M.\ X\ n\ x = t\right) \neq 0 \wedge \Pr\left(x \text{ in } M.\ X\ n'\ x = t\right) \neq 0 \longrightarrow$
    $\Pr\left(x \text{ in } M.\ X\ (n+1)\ x = s \mid X\ n\ x = t\right) =$
    $\Pr\left(x \text{ in } M.\ X\ (n'+1)\ x = s \mid X\ n'\ x = t\right)$

A DTMC is a set of states S and a time-indexed family of random variables X. The type variable α contains the sample elements of our background probability space M.

The assumptions that the conditions are non-zero are required. Otherwise the Markov chain in Fig. 2 would not fulfill the assumptions of our locale: as $\Pr(X_n = A) = 0$ for all $0 < n$, an unrestricted time-homogeneous property would imply $\frac{1}{2} = \Pr(X_1 = B \mid X_0 = A) = \Pr(X_2 = B \mid X_1 = A) = 0$. It is not necessary to restrict A and B to be elements of S, as the probability to reach elements outside S is zero.

Definition 10 (Trace Space with Initial Distribution) From an initial distribution $I :: \sigma$ pmf and a Markov kernel $K :: \sigma \Rightarrow \sigma$ pmf (and hence also a trace space T), we can construct a Markov process T':

$$\begin{aligned} T' \quad &:: \sigma \text{ pmf} \Rightarrow \sigma \text{ stream measure} \\ T'\ I &= \text{bind}\ I\ (\lambda s.\ \text{distr}\ (T\ s)\ S\ (\lambda \omega.\ s{\cdot}\omega)) \end{aligned}$$

The stochastic process $T'\ I$ fulfills the properties of DTMC:

Theorem 5 ($T'\ I$ is a Markov Chain)

$$\text{DTMC}\ (T'\ I)\ (\text{acc}\ \text{“}\ I)\ (\lambda n\ \omega.\ \omega\ !!\ n)$$

Equivalence We show that both definitions of a Markov chain are equivalent. With Theorem 1 we show the memory-less and the time-homogeneous property. The other direction — to construct a Markov kernel from a stochastic process X fulfilling the Markov chain properties — is more complicated. One problem is, that the Markov kernel is defined at each state, even when it is not reached by X.

Definition 11 (Markov Kernel K from the Stochastic Process X) Given a Markov process DTMC $M\ S\ X$ we construct a Markov kernel K. When X reaches state s at some time n, i.e. $\Pr\left(x \text{ in } M.\ X\ n\ x = s\right) > 0$, then we lift $K\ s$:

lift-definition $K :: \sigma \Rightarrow \sigma$ pmf **is**
  $\lambda s.$ distr (cprob M $\{x \in$ space $M \mid X\ n\ x = s\}$) (count-space UNIV) $(X\ (n+1))$

when no such n exists we simply define $K\ s =$ return-pmf s. The initial distribution I is lifted from $X\ 0$:

lift-definition $I :: \sigma$ pmf **is** distr M (count-space UNIV) $(X\ 0)$

The time-homogeneous property tells us that K is well-defined. From K's definition we show that K is the transition probability (for reachable states):

$$\frac{\Pr\left(x \text{ in } M.\ X\, n\, x = s\right) > 0}{\mathsf{pmf}\ (K\, s)\ t = \Pr\left(x \text{ in } M.\ X\, n\, x = s \mid X\, (n+1)\, x = t\right)}$$

The resulting trace space $T'\ I$ is equal to the distribution of X:

Theorem 6 (The Markov kernel K' has the distribution of X)

$$T'\ I = \mathsf{distr}\ M\ S\ (\lambda x.\ \mathsf{to\text{-}stream}\ (\lambda n.\ X\ n\ x))$$

Theorems 5 and 6 show that for each Markov chain represented as a stochastic process there exists a Markov kernel representing the same Markov chain, and vice versa.

3.6 Classifying Markov Chain States

Now we come to the questions stated at the beginning of Section 3: we want to classify states based on their behaviour on the trace space. But before this, we introduce *communicating* states and *essential* equivalence classes. They are directly defined by the transition system, i.e. expressible with acc. In this section we assume a kernel K and its trace space T.

Definition 12 (Communicating States) The communicating relation is the symmetric subset of the accessibility relation:

$$\begin{aligned}&\mathsf{communicating} &&:: \ (\sigma \times \sigma)\ \mathsf{set}\\ &(x, y) \in \mathsf{communicating} &&\longleftrightarrow (x, y) \in \mathsf{acc} \wedge (y, x) \in \mathsf{acc}\end{aligned}$$

The relation communicating is an equivalence relation as it is symmetric, reflexive, and transitive. Its equivalence classes are called *irreducible*. We write the set of irreducible classes as UNIV/communicating. In Fig. 2 (a) the irreducible classes are $\{A\}$, $\{B\}$, and $\{C_1, C_2, C_3\}$. Fig. 2 (b) has only one irreducible class: $\{0, 1, 2, \ldots\}$. It is a so-called irreducible Markov chain.

An equivalence class is *essential* iff no state outside the class is accessible from a state inside the class. If the Markov chain reaches a state in an essential class, it will stay in this class forever. Essential classes are equivalent to *bottoms* SCCs (strongly connected components) in graph theory.

Definition 13 (Essential Class) An *essential class* C is an irreducible class where all accessible states from C are again in C:

$$\begin{aligned}&\mathsf{essential\text{-}class} &&:: \ \sigma\ \mathsf{set} \Rightarrow \mathsf{bool}\\ &\mathsf{essential\text{-}class}\ C &&\longleftrightarrow C \in \mathsf{UNIV}/\mathsf{communicating} \wedge \left(\forall (x, y) \in \mathsf{acc}.\ x \in C \longrightarrow y \in C\right)\end{aligned}$$

3.6.1 Recurrence

In this section we are interested in *recurrence*, the property of almost surely coming back, and *positive recurrence*, to come back with a finite expectation. While this is often a property of the transition graph, we need to analyze probabilities and even generating functions converging against these probabilities. Hence in this subsection we need convergence and derivatives on the real line. For this we use Isabelle's multivariate analysis library described in [29].

We write $f\ z \xrightarrow{z \to 1^-} x$ to state that the function $f\ z$ converges to x as z approaches 1 from the left, $X\ n \xrightarrow{n \to \infty} x$ to state that the sequence $X\ n$ converges to x, and f has-derivative y (at x) to state that y is the derivative of f at x.

Definition 14 (Recurrent States) A state x is *recurrent* iff it is guaranteed that when we start in x, we come back to x.

$$\begin{aligned}&\mathsf{recurrent} \quad :: \quad \sigma \Rightarrow \mathsf{bool}\\ &\mathsf{recurrent}\ x \longleftrightarrow \Pr(\omega\ \mathsf{in}\ T\ x.\ \mathsf{ev}\ (\mathsf{HLD}\ \{x\})\ \omega) = 1\end{aligned}$$

Q1 can be written as recurrent 0. For **Q3** we are interested in the average time to return to 0. For this we define the *hitting time* of x on the trace ω (this is also called *first passage time*):

Definition 15 (Average Hitting Time)

$$\begin{aligned}&M \quad :: \sigma \Rightarrow \mathsf{ereal}\\ &M\ x = \int \mathsf{sfirst}\ (\mathsf{HLD}\ \{x\})\ \mathsf{d}(T\ x)\end{aligned}$$

With this **Q3** can also be written as $M\ x$. When x is non-recurrent, $M\ x$ is always infinite. But even for a recurrent state it may be infinite. If $M\ x$ is real, the state is called positive recurrent:

Definition 16 (Positive Recurrent States)

$$\begin{aligned}&\mathsf{posrecurrent} \quad :: \quad \sigma \Rightarrow \mathsf{bool}\\ &\mathsf{posrecurrent}\ x \longleftrightarrow \mathsf{recurrent}\ x \wedge M\ x < \infty\end{aligned}$$

Now we provide tools to relate (positive) recurrent states with other quantities, like the probability to hit a state infinitely often or the average number of times of reaching a state.

Definition 17 (Unbound Quantities)

$$\begin{aligned}&G, U' \quad :: \sigma \Rightarrow \sigma \Rightarrow \mathsf{ereal}\\ &G\ x\ y \ = \int \mathsf{scount}\ (\mathsf{HLD}\ \{y\})\ \mathsf{d}(T\ x)\\ &U'\ x\ y = \int \mathsf{sfirst}\ (\mathsf{HLD}\ \{y\})\ \mathsf{d}(T\ x) + 1\end{aligned}$$

The value $G\ x\ y$ (G is called *Green's function* or *Green kernel*) is the expected number of occurrences of y, starting *after* x. The value $U'\ x\ y$ is the expected number of steps until y is reached, starting *after* x (including the last step in which y occurs).

$$\begin{aligned}
&H, U, F :: \sigma \Rightarrow \sigma \Rightarrow \mathsf{real}\\
&H\ x\ y \;= \Pr(\omega \text{ in } T\,x.\ \mathsf{alw}\ (\mathsf{ev}\ (\mathsf{HLD}\ \{y\}))\ \omega)\\
&U\ x\ y \;= \Pr(\omega \text{ in } T\,x.\ \mathsf{ev}\ (\mathsf{HLD}\ \{y\})\ \omega)\\
&F\ x\ y \;= \Pr(\omega \text{ in } T\,x.\ \mathsf{ev}\ (\mathsf{HLD}\ \{y\})\ (x{\cdot}\omega))
\end{aligned}$$

The value $H\ x\ y$ is the probability that y is infinitely often reached when starting in x. The values $U\ x\ y$ and $F\ x\ y$ concern the probability that y is reached when starting in x, with the difference that U starts counting after x itself. Obviously, $\mathsf{recurrent}\ x \longleftrightarrow U\ x\ x = 1$.

With the exception of H and U' the unbound quantities are expressible as infinite sums over time-bounded probabilities:

Definition 18 (Time-bounded Reachability Probabilities)

$$\begin{aligned}
&p, u, f \;:: \sigma \Rightarrow \sigma \Rightarrow \mathsf{nat} \Rightarrow \mathsf{real}\\
&p\ x\ y\ n = \Pr(\omega \text{ in } T\,x.\ (x \cdot \omega)\ !!\ n = y)\\
&u\ x\ y\ n = \Pr(\omega \text{ in } T\,x.\ \mathsf{sfirst}\ (\mathsf{HLD}\ \{y\})\ \omega = (n :: \mathsf{nat}))\\
&f\ x\ y\ n = \Pr(\omega \text{ in } T\,x.\ \mathsf{sfirst}\ (\mathsf{HLD}\ \{y\})\ (x{\cdot}\omega) = (n :: \mathsf{nat}))
\end{aligned}$$

Lemma 20 (Unbound Quantities as Series of Time-bounded Probabilities)

$$G\ x\ y = \textstyle\sum_n p\ x\ y\ n \quad F\ x\ y = \sum_n f\ x\ y\ n \quad U\ x\ y = \sum_n u\ x\ y\ n$$

The sum in $G\ x\ y$ can be infinite. To avoid case distinctions for proofs about G, and to take advantage of mathematical analysis, the theory of Markov chains introduces *generating functions*. For a probability or expectation $P = \sum_n f_n$, its power series is called a generating function $P_g\ z = \sum_n f_n \cdot z^n$. Here z is a real number between 0 and 1. If z approaches 1 from the left, we have $P_g\ z \xrightarrow{z \to 1^-} P$.

Definition 19 (Generating Functions for G, F, U and U')

$$\begin{aligned}
&G_g, F_g, U_g, U'_g :: \sigma \Rightarrow \sigma \Rightarrow \mathsf{real} \Rightarrow \mathsf{real}\\
&G_g\ x\ y\ z \;= \textstyle\sum_n p\ x\ y\ n \cdot z^n\\
&F_g\ x\ y\ z \;= \textstyle\sum_n f\ x\ y\ n \cdot z^n\\
&U_g\ x\ y\ z \;= \textstyle\sum_n u\ x\ y\ n \cdot z^{n+1}\\
&U'_g\ x\ y\ z \;= \textstyle\sum_n u\ x\ y\ n \cdot (n+1) \cdot z^n
\end{aligned}$$

These functions are real valued and, as long as $|z| < 1$, the sums are well-defined. The generating function U'_g is the derivative of U_g.

Lemma 21 (Unbound Quantities as Limits of Generating Functions) *The generating functions tend to their probabilistic variant, if z approaches 1 from the left:*

$$G_g\ x\ y\ z \xrightarrow{z \to 1^-} G\ x\ y \qquad U_g\ x\ y\ z \xrightarrow{z \to 1^-} U\ x\ y \qquad F_g\ x\ y\ z \xrightarrow{z \to 1^-} F\ x\ y$$

For example, we can show that the average number of visits to x is inversely proportional to the probability to *not* reach x:

$$G\ x\ x = 1/(1 - U\ x\ x) \quad .$$

However this equation is undefined when $U\ x\ x = 1$. With generating functions we have a nice tool to relate them also in this case.

Theorem 7 (Relating recurrent and G)

$$\frac{|z| < 1}{G_g\ x\ x\ z = 1/(1 - U_g\ x\ x\ z)}$$

As $G_g\ x\ x\ z \xrightarrow{z \to 1^-} G\ x\ x$ *we also know that:*

$$\frac{1}{1 - U_g\ x\ x\ z} \xrightarrow{z \to 1^-} G\ x\ x$$

So, recurrent x *(i.e.* $U\ x\ x = 1$*) is equal to* $G\ x\ x = \infty$*.*

As $G\ x\ x = \infty \longleftrightarrow G\ y\ y = \infty$ if $(x, y) \in$ communicating, we have:

Corollary 3 (recurrent is Invariant on Irreducible Classes)

$$\frac{(x, y) \in \mathsf{communicating}}{\mathsf{recurrent}\ x \longleftrightarrow \mathsf{recurrent}\ y}$$

Recurrence is not only invariant on irreducible classes, we also know that recurrence of a class implies that it is an essential class. Now we relate H, U and recurrent:

Lemma 22 (Relation between recurrent and H and U)

$$H\ x\ x = \mathsf{if}\ \mathsf{recurrent}\ x\ \mathsf{then}\ 1\ \mathsf{else}\ 0 \qquad H\ x\ y = U\ x\ y \cdot H\ y\ y$$

$$\frac{\mathsf{recurrent}\ x \qquad (x, y) \in \mathsf{acc}}{U\ y\ x = 1 \quad \wedge \quad \mathsf{recurrent}\ y \quad \wedge \quad (x, y) \in \mathsf{communicating}}$$

For recurrent states accessible equals communicating hence they form an essential class:

Theorem 8 (Recurrent Classes are Essential)

$$\frac{\mathsf{recurrent}\ x}{\mathsf{essential\text{-}class}\ \{y \mid (x, y) \in \mathsf{acc}\}}$$

With Theorem 8 and the pigeon hole principle follows also that finite essential classes are recurrent:

Corollary 4 (Finite Essential Classes are Recurrent)

$$\frac{\mathsf{finite}\ C \qquad C \in \mathsf{UNIV}/\mathsf{communicating}}{\mathsf{essential\text{-}class}\ C \longleftrightarrow (\forall x \in C.\ \mathsf{recurrent}\ x)}$$

Lemma 22 also helps us to answer **Q2** (which can be written formally as $1 - U\ A\ C_3 \neq 0$ or $1 - U\ 0\ n \neq 0$), if we can show that C_3 or n are recurrent.

Similar to recurrence, positive recurrence is invariant on irreducible classes. To prove this, we show that the derivative of the generating function U_g is U_g' and then applying l'Hôpital's rule to relate U' and U:

Lemma 23 (Relation between U' and U)

$$\frac{|z| < 1}{(U_g\ x\ x)\ \text{has-derivative}\ (U_g'\ x\ z)\ (\text{at}\ z)} \qquad \frac{\text{recurrent}\ x}{1/U_g'\ x\ x\ z \xrightarrow{z \to 1^-} 1/U'\ x\ x}$$

Hence positive recurrence is invariant on irreducible classes: either all states are positive recurrent or all states are null recurrent.

Corollary 5 (Positive Recurrent is Invariant on Irreducible Classes)

$$\frac{(x, y) \in \text{communicating}}{\text{posrecurrent}\ x \longleftrightarrow \text{posrecurrent}\ y}$$

3.6.2 Stationary Distribution

A stationary distribution μ is a measure with the invariant, that it is closed under multiplication with the transition function τ: $\forall y.\ \mu\ y = (\sum_x \mu\ x \cdot \tau\ x\ y)$. In Isabelle/HOL, we use K, the representation as pmfs, and use the monadic bind:

Definition 20 (Stationary Distribution)

$$\begin{aligned} &\text{stationary-distribution} \quad :: \quad \sigma\ \text{pmf} \Rightarrow \text{bool} \\ &\text{stationary-distribution}\ N \longleftrightarrow N = \text{bind-pmf}\ N\ K \end{aligned}$$

It is often very easy to show that a Markov chain has a stationary distribution N: we only need to show that N is a solution to this equation system.

Example 1 (Stationary Distributions of Fig. 1) The essential class $\{C_1, C_2, C_3\}$ in Fig. 2 (a) has the stationary distribution N with $\text{pmf}\ N\ C_1 = \text{pmf}\ N\ C_2 = 3/10$ and $\text{pmf}\ N\ C_3 = 4/10$. Fig. 2 (b) is an essential Markov chain. Its stationary distribution N is $\text{pmf}\ N\ n = 1/2^{n+1}$.

How do we compute such a distribution? Fortunately, we can compute the distribution by reachability analysis. When the stationary distribution exists, on an essential, positive recurrent class, the inverse of the hitting time equals the stationary distribution.

Definition 21 (Stationary Distribution as Reverse of U')

$$\begin{aligned} &\text{stat} \quad :: \sigma\ \text{set} \Rightarrow \sigma\ \text{measure} \\ &\text{stat}\ C = \text{point-measure UNIV}\ \left(\lambda x.\ \text{indicator}\ C\ x / U'\ x\ x\right) \end{aligned}$$

We define this as a measure, but not a probability mass function, as it will only be a probability measure when it is a stationary distribution.

Lemma 24 (stat is a Sub-probability)

$$\frac{\text{essential-class}\ C \qquad \text{countable}\ C \qquad \forall x \in C.\ \text{posrecurrent}\ x}{\text{emeasure}\ (\text{stat}\ C)\ \text{UNIV} \leq 1}$$

With this we can prove that: a stationary distribution on an essential class is always positive recurrent and that the stationary distribution is equal to stat.

Theorem 9 (Stationary Distribution implies Positive Recurrence)

$$\frac{\mathsf{stationary\text{-}distribution}\ N \qquad N \subseteq C \qquad \mathsf{countable}\ C \qquad \mathsf{essential\text{-}class}\ C}{\forall x \in C.\ \mathsf{posrecurrent}\ x \qquad N = \mathsf{stat}\ C}$$

The last two proofs already require the relations between F and U' and their generating functions. Using them we answer **Q3**; with the stationary distribution we compute $M\ x$.

Another property of the stationary distribution: it is the limit of the time bounded probability $p\ x\ y\ t$ when t goes to infinity. This limit does not always exist: the Markov chain on states A and B with the only transitions $A \to B$ and $B \to A$ has no such limit. To avoid such cases, we introduce the class of aperiodic sets. The aperiodicity guarantees that the probability $p\ x\ y\ t$ does not oscillate in t.

Definition 22 (Aperiodic Classes)

$$\begin{aligned}&\mathsf{aperiodic} \quad :: \quad \sigma\ \mathsf{set} \Rightarrow \mathsf{bool}\\ &\mathsf{aperiodic}\ C \longleftrightarrow C \in \mathsf{UNIV}/\mathsf{communicating} \wedge\\ &\qquad\qquad (\forall x \in C.\ \mathsf{Gcd}\ \{i \mid 0 < i \wedge 0 < p\ x\ x\ i\} = 1)\end{aligned}$$

The function Gcd computes the greatest common divisor of the (possibly infinite) set. To prove aperiodicity it is enough to show for two numbers n and m, with $\mathsf{gcd}\ n\ m = 1$, that $0 < p\ x\ x\ n$ and $0 < p\ x\ x\ m$. However for our analysis we need a different view, we want to know that after a time point I, x is always reached again. We show that these two views are equivalent:

Theorem 10 (Aperiodic Class equals Eventually Non-zero p)

$$\mathsf{aperiodic}\ C \longleftrightarrow C \in \mathsf{UNIV}/\mathsf{communicating}\ \wedge\ (\forall x \in C.\ \exists I.\ \forall i \geq I.\ 0 < p\ x\ x\ i)$$

The proof involves a little number theory: for a set $S \subseteq \mathbb{N}$ closed under addition and containing at least one non-zero element we have $m \cdot \gcd S \in S$ for almost all $m \in \mathbb{N}$. To prove that aperiodic implies the right-hand side we set $S = \{i \mid 0 < i \wedge 0 < p\ x\ x\ i\}$.

Now we can prove our central theorem of this section: For an aperiodic, essential class, the stationary distribution is the limit of the marginal distributions.

Theorem 11 (Stationary Distribution is Asymptotic Distribution)

$$\frac{\begin{array}{c}\mathsf{stationary\text{-}distribution}\ N\\ N \subseteq C \qquad \mathsf{countable}\ C \qquad \mathsf{aperiodic}\ C \qquad \mathsf{essential\text{-}class}\ C\end{array}}{\begin{array}{c}\forall y \in C.\ \displaystyle\int_x \left|p\ x\ y\ t - \mathsf{pmf}\ N\ y\right| \mathsf{d}(\mathsf{count\text{-}space}\ C) \xrightarrow{t\to\infty} 0\\ \forall x, y \in C.\ p\ x\ y\ t \xrightarrow{t\to\infty} \mathsf{pmf}\ N\ y\end{array}}$$

This proof requires the product construction of Markov chains as described in Section 3.4. We construct the product of a Markov chain starting in a state y, and one where the initial distribution is already N. Essential and aperiodic classes lift from the factor Markov chains to the product Markov chain.

With that central theorem we answer our final question **Q4**. The Fig. 2 (b) is aperiodic and has a stationary probability distribution, hence we only need to compare the values of the stationary distribution.

3.7 Modelling Example (a) and (b) in Isabelle/HOL

We want to come back to the example Markov chains shown in Fig. 2, and the questions on them stated in the introduction. With the analysis introduced in the previous sections, we can now state the questions formally:

Q1 When we start (b) does it always come back to 0?
$\Pr(\omega \text{ in } T\,0.\ \mathsf{ev}\,(\mathsf{HLD}\,\{0\})\ \omega) = 1$?

Q2 Is it possible that (a) never reaches C_3 or that (b) never reaches a state n?
$\Pr(\omega \text{ in } T\,A.\ \mathsf{alw}\ (\neg\mathsf{HLD}\,\{C_3\})\ \omega) \neq 0$ or $\Pr(\omega \text{ in } T\,0.\ \mathsf{alw}\ (\neg\mathsf{HLD}\,\{n\})\ \omega) \neq 0$.

Q3 When we start (b) in 0, what is the average time until it reaches 0 again?
$\int \mathsf{sfirst}\ \{0\}\ \mathsf{d}(T\,0) = \ ?$

Q4 When (a) runs for a long time, is C_3 more likely to occur than C_1?
$\lim_{t\to\infty} \Pr(\omega \in T\,A.\ \omega\ !!\ t = C_3) \geq \lim_{t\to\infty} \Pr(\omega \in T\,A.\ \omega\ !!\ t = C_1)$

Now we want to model the examples from Fig. 2 in Isabelle/HOL and compute their stationary distribution.

Example (a) We first define a datatype $\mathsf{state} = A|B|C_1|C_2|C_3$. The Markov chain on it does not have any repeating structure, we construct the Markov kernel by describing the transition function τ as a matrix:

$$\tau\ a\ b = \begin{cases} 1/2 & \text{if } (a,b) \in \{(A,B),(A,C_1),(B,B),(B,C_1),(C_3,C_3)\} \\ 1/3 & \text{if } (a,b) \in \{(C_1,C_1),(C_1,C_2),(C_1,C_3),(C_2,C_1),(C_2,C_2),(C_2,C_3)\} \\ 1/4 & \text{if } (a,b) \in \{(C3,C1),(C3,C2)\} \\ 0 & \text{otherwise} \end{cases}$$

We prove that $\{C_1, C_2, C_3\}$ is essential (by case distinction) and aperiodic (every state has a loop).

Finally, we define the stationary distribution n:

$$\textbf{lift-definition}\ n :: \mathsf{state\ pmf}\ \textbf{is}\ \lambda C_1 \to 0.3 \mid C_2 \to 0.3 \mid C_3 \to 0.4 \mid \ \dashrightarrow 0$$

Again by case distinction and computation by the simplifier we show that n fulfills the equation of a stationary distribution.

Example (b) This example is more interesting, as it is infinite and has a structure in it. Surprisingly its formalization is equivalently short. The Markov kernel is now constructed out of a Bernoulli distribution, i.e. a coin flip with probability 1/3 true.

$$K\ x = \mathsf{map\text{-}pmf}\ (\lambda b.\ \ \mathsf{if}\ \ b\ \ \mathsf{then}\ x+1\ \mathsf{else}\ x-1)\ (\mathsf{bernoulli\text{-}pmf}\ (1/3))$$

Be aware that the difference on natural numbers is saturating, i.e. for $x = 0$ we have $x - 1 = 0$. So $\mathsf{pmf}\ (K\ 0)\ 0 = 2/3$. This transition is also in Fig. 2.

First, we deduce the accessibility structure: everything is from each state accessible: $(i,j) \in \mathsf{acc}$. This is proved by induction on the difference of i and j. Hence the entire essential class is UNIV. As the state 0 has a loop, the entire system is also aperiodic.

Now the stationary distribution is geometric-pmf $(1/2)$. The proof of this statement works nearly automatic, as we have per state only two outgoing edges. So with Theorem 11 we show the limit probability of being in a state j:

$$\lim_n p\ i\ j\ n = \left(\frac{1}{2}\right)^{j+1}$$

Note that the limit probability only depends on j: this is due to the asymmetry at 0. We can compute with Theorem 9 the expected recurrence time (the inverse of pmf (geometric-pmf $(1/2)$) i):

$$U'\ i\ i = 2^{i+1}$$

4 Markov Decision Processes

In this section we introduce MDPs and their general representation in Isabelle/HOL. These definitions and theorems are mostly mechanized (and generalized) versions of definitions and theorems by Baier and Katoen [7], Baier [6], and de Alfaro [2]. MDPs extend Markov chains with non-deterministic choices. Answering the same questions (as presented in Section 3) on MDPs is more difficult, as we do not have a probability space for traces anymore. Instead schedulers are introduced to resolve non-determinism: a scheduler selects based on the history (i.e. the list of visited states) the distribution of the next states. Then a MDP together with a scheduler implies a probability space of traces. Questions like **Q1** to **Q4** are then stated for the maximal or minimal probability/expectation over a specific class of schedulers.

We define MDPs as *non-deterministic probabilistic transition systems*. The non-determinism is modeled by a non-empty set of transition probabilities per state.

Definition 23 (Kernel of a MDP) The transition system of a Markov decision process is specified by a function $K :: \sigma \Rightarrow \sigma$ pmf set (the kernel of the MDP) with the property $\forall x.\ K\ x \neq \emptyset$.

For the rest of this section we assume a MDP specified by its kernel K.

The usual way to resolve the non-determinism is to introduce schedulers and use the Markov chain induced by a scheduler [7] to assign probabilities to sequences. We use a similar concept to resolve the non-determinism, namely *configurations*: A *configuration* describes the current state s, a selected distribution in $K\ s$ and future non-deterministic behaviour by mapping each state to the future configuration.

Definition 24 (Configurations) The type of configurations are modelled as a *non-free* codatatype (i.e. a codatatype with additional equalities):

$$\sigma\ \mathsf{cfg} = \mathsf{Cfg}\ (\mathsf{state}\text{: }\sigma)\ (\mathsf{action}\text{: }\sigma\ \mathsf{pmf})\ (\mathsf{cont}\text{: }\sigma \Rightarrow \sigma\ \mathsf{cfg})$$
$$\mathsf{state}\ (\mathsf{cont}\ c\ s) = s.$$

It is not possible to directly define such a non-free codatatype in Isabelle. Instead we introduce a recursive codatatype with action and continuation and then define a configuration datatype as pair of the recursive codatatype and the current state. And on top of this we derive a corecursion operator and the usual selector functions.

In the literature the non-determinism is resolved by a *scheduler*, a function $\mathsf{sc} :: \sigma\ \mathsf{list} \Rightarrow \sigma\ \mathsf{pmf}$ with $\mathsf{sc}\ (h{\cdot}s) \in K\ s$. The scheduler is used to select the next action depending on the previous behaviour of the MDP. A configuration can be seen as a pair of scheduler sc and non-empty history h: In one direction, each scheduler sc and history h give rise to a configuration $c'\ \mathsf{sc}\ \mathsf{h}$, namely:

$$c'\ \mathsf{sc}\ \mathsf{h} = \mathsf{Cfg}\ (\mathsf{last}\ \mathsf{h})\ (\mathsf{sc}\ \mathsf{h})\ (\lambda s.\ c'\ \mathsf{sc}\ (\mathsf{h} \cdot s)) \quad .$$

In the other direction, a scheduler is constructed from a configuration c' by iterating the continuation of c' over the history h and return state of the resulting configuration.

Definition 25 (Memoryless Configuration) A *memoryless* scheduler decides the next step purely on the current state. We write a configuration with a memoryless scheduler as $\mathsf{memoryless\text{-}on}\ f\ s$, where $f :: \sigma \Rightarrow \sigma\ \mathsf{pmf}$. We define it by the following primitive corecursive equations:

$$\begin{aligned}
&\mathsf{memoryless\text{-}on} && :: (\sigma \Rightarrow \sigma\ \mathsf{pmf}) \Rightarrow \sigma \Rightarrow \sigma\ \mathsf{cfg} \\
&\mathsf{state}\ (\mathsf{memoryless\text{-}on}\ f\ s) && = s \\
&\mathsf{action}\ (\mathsf{memoryless\text{-}on}\ f\ s) && = f\ s \\
&\mathsf{cont}\ (\mathsf{memoryless\text{-}on}\ f\ s)\ t && = \mathsf{memoryless\text{-}on}\ f\ t
\end{aligned}$$

Now, not every configuration is valid in our MDP, only those where the actions are in K. For this we define the set of valid configurations from a state $\mathsf{cfg\text{-}on} :: \sigma \Rightarrow \sigma\ \mathsf{cfg\ set}$ coinductively with the following introduction rule:

$$\frac{\mathsf{state}\ c = s \qquad \mathsf{action}\ c \in K\ s \qquad \forall t \in \mathsf{action}\ c.\ \mathsf{cont}\ c\ t \in \mathsf{cfg\text{-}on}\ t}{c \in \mathsf{cfg\text{-}on}\ s}$$

MDPs do not have a probability space associated with them, instead one is usually interested in the maximal and minimal probability over the valid configurations.

Definition 26 (Markov Kernel of a Configuration) We define $\mathsf{K\text{-}cfg}$ as a Markov chain whose states are configurations:

$$\begin{aligned}
&\mathsf{K\text{-}cfg} && :: \sigma\ \mathsf{cfg} \Rightarrow \sigma\ \mathsf{cfg\ pmf} \\
&\mathsf{K\text{-}cfg}\ c && = \mathsf{map\text{-}pmf}\ (\mathsf{cont}\ c)\ (\mathsf{action}\ c)
\end{aligned}$$

Definition 27 (Trace Space of a Configuration) The Markov chain trace space $\mathsf{T\text{-}cfg}$ is the one induced by the Markov chain kernel $\mathsf{K\text{-}cfg}$. It is used to construct a trace space of states starting from a specific configuration c:

$$\begin{aligned}
&T && :: \sigma\ \mathsf{cfg} \Rightarrow \sigma\ \mathsf{stream\ measure} \\
&T\ c && = \mathsf{distr}\ (\mathsf{T\text{-}cfg}\ c)\ S\ (\mathsf{smap}\ \mathit{state})
\end{aligned}$$

4.1 Extremal Probability and Expectation on MDP Traces

Now, we use the MDP trace space $T\,c$ to define the maximum and minimum expectation and probabilities of the MDP:

Definition 28 (MDP Expectation) The *maximum expectation* $\mathsf{E\text{-}sup}\, s\, f$ and the *minimum expectation* $\mathsf{E\text{-}inf}\, s\, f$ of a Borel-measurable function f starting in a state s are defined as:

$$\begin{aligned}\mathsf{E\text{-}sup}, \mathsf{E\text{-}inf} &:: \sigma \Rightarrow (\sigma\ \mathsf{stream} \Rightarrow \mathsf{ereal}) \Rightarrow \mathsf{ereal}\\ \mathsf{E\text{-}sup}\ s\ f &= \bigsqcup_{c \in \mathsf{cfg\text{-}on}\ s} \int f\ \mathsf{d}(T\ c)\\ \mathsf{E\text{-}inf}\ s\ f &= \bigsqcap_{c \in \mathsf{cfg\text{-}on}\ s} \int f\ \mathsf{d}(T\ c)\end{aligned}$$

Definition 29 (MDP Probability) The *maximal and minimal probability* of a measurable predicate P are defined as:

$$\begin{aligned}\mathsf{P\text{-}sup}, \mathsf{P\text{-}inf} &:: \sigma \Rightarrow (\sigma\ \mathsf{stream} \Rightarrow \mathsf{bool}) \Rightarrow \mathsf{ereal}\\ \mathsf{P\text{-}sup}\, s\, P &= \bigsqcup_{c \in \mathsf{cfg\text{-}on}\ s} \Pr(\omega\ \mathsf{in}\ T\, c.\ P\ \omega)\\ \mathsf{P\text{-}inf}\, s\, P &= \bigsqcap_{c \in \mathsf{cfg\text{-}on}\ s} \Pr(\omega\ \mathsf{in}\ T\, c.\ P\ \omega)\end{aligned}$$

The minimal and maximal expectations do not carry over linearity from integrals: they are not additive anymore. Fortunately, both still support iteration:

Lemma 25 (Iteration for maximal and minimal expectation) *Assume that f is Borel-measurable. Then we get the following iteration rules:*

$$\begin{aligned}\mathsf{E\text{-}sup}\ s\ f &= \bigsqcup_{D \in K\ s} \int_t \mathsf{E\text{-}sup}\ t\ (\lambda\omega.\ f\,(t{\cdot}\omega))\ \mathsf{d}D\\ \mathsf{E\text{-}inf}\ s\ f &= \bigsqcap_{D \in K\ s} \int_t \mathsf{E\text{-}inf}\ t\ (\lambda\omega.\ f\,(t{\cdot}\omega))\ \mathsf{d}D\end{aligned}$$

Similarly to Lemma 8, we want to show that the minimal and maximal expectation transfer certain least fixed points. For the maximal expectation this is quite obvious, as the supremum of a least fixed point commutes with the supremum over all configurations. The following lemma is not as general as possible, as the functional reads in each iteration only the head of the stream. This is enough for our current applications, and it is symmetric with the lemma for minimal expectation.

Lemma 26 (E-sup Transfers Least Fixed Points)

$$\frac{\begin{array}{c}g \in (\mathsf{count\text{-}space}\ \mathsf{UNIV} \times^{\sigma} S) \to^{\sigma} \mathsf{borel} \qquad \forall s.\ \mathsf{sup\text{-}continuous}\ (g\ s)\\ \forall s, c \in \mathsf{cfg\text{-}on}\ s, f \in S \to^{\sigma} \mathsf{borel}.\ \int_{\omega} g\ s\ (f\ \omega)\ \mathsf{d}(T\ c) = g\ s\ (\int f\ \mathsf{d}(T\ c))\end{array}}{\mathsf{E\text{-}sup}\ s\ (\mathsf{lfp}\ (\lambda F\ (s{\cdot}\omega).\ g\ s\ \omega)) = \mathsf{lfp}\left(\lambda f\ s.\ \bigsqcup_{D \in K\ s} \int_t g\ t\ (f\ t)\ \mathsf{d}D\right) s}$$

Proving such a transfer lemma for the minimal expectation is more complicated, Baier [6] shows how it is done for reachability probability when the set of distributions per state are finite (i.e. $K\, s$ is finite for all s). That proof nicely generalizes from the reachability probability to expectations of least fixed points. Note the assumption $\forall s.\ \mathsf{finite}\ (K\ s)$: it is not clear to the author how to remove it or even weaken it.

Lemma 27 (E-inf Transfers Least Fixed Points)

$$\frac{g \in (\mathsf{count\text{-}space}\ \mathsf{UNIV} \times^{\sigma} S) \to^{\sigma} \mathsf{borel} \quad \forall s.\ \mathsf{sup\text{-}continuous}\ (g\ s) \quad \forall s.\ \mathsf{finite}\ (K\ s) \quad \forall s.\ \forall c \in \mathsf{cfg\text{-}on}\ s.\ \forall f \in S \to^{\sigma} \mathsf{borel}.\ \int_{\omega} g\ s\ (f\ \omega)\ \mathsf{d}(T\ c) = g\ s\ (\int f\ \mathsf{d}(T\ c))}{\mathsf{E\text{-}inf}\ s\ (\mathsf{lfp}\ (\lambda F\ (s{\cdot}\omega).\ g\ s\ \omega)) = \mathsf{lfp}\left(\lambda f\ s.\ \prod_{D \in K\ s} \int_t g\ t\ (f\ t)\ \mathsf{d}D\right) s}$$

4.2 Reachability Probabilities on MDPs

A central property of model checking is the reachability probability, i.e. to compute the probability of S_1 suntil S_2 for sets of states S_1 and S_2. In this section we will analyze finite reachability probabilities, i.e. on a finite MDP. Hence in this and in the next section we assume a *finite* MDP, i.e. there is a non-empty finite set S, under which K is closed: $\left(\bigcup_{x \in S} \bigcup_{D \in K\ x} \mathsf{set\text{-}pmf}\ D\right) \subseteq S$ and $K\ x$ is finite for all x in S.

Definition 30 A *reachability problem* is defined by the transition states $S_1 \subseteq S$ and the target states $S_2 \subseteq S$. We assume non-overlapping S_1 and S_2: $S_1 \cap S_2 = \emptyset$. We write $S_1\ U\ S_2$ for the sequences which stay in S_1 until they reach S_2 after a finite time: $S_1\ U\ S_2 = (\mathsf{HLD}\ S_1)\ \mathsf{suntil}\ (\mathsf{HLD}\ S_2)$.

For model checking a characterization as a least fixpoint is required and an optimal, memoryless scheduler is needed. We get these by the following two theorems:

Theorem 12 (Least Fixed Point Characterization) *The maximum reachability probability* $\mathsf{P\text{-}sup}\ s\ (S_1\ U\ S_2)$*, and the minimum reachability probability* $\mathsf{P\text{-}inf}\ s\ (S_1\ U\ S_2)$*, is characterized by the least fixpoint of* F^+*, and* F^-*, respectively:*

$$\mathsf{P\text{-}sup}\ s\ (S_1\ U\ S_2) = \mathsf{lfp}\ F^+\ s \quad \textbf{\textit{and}} \quad \mathsf{P\text{-}inf}\ s\ (S_1\ U\ S_2) = \mathsf{lfp}\ F^-\ s \quad \textbf{\textit{where}}$$

$$F^+\ v\ s = \mathsf{if}\ s \in S_2\ \mathsf{then}\ 1\ \mathsf{else\ if}\ s \in S_1\ \mathsf{then}\ \coprod_{D \in K\ s} \int v\ \mathsf{d}D\ \mathsf{else}\ 0$$

$$F^-\ v\ s = \mathsf{if}\ s \in S_2\ \mathsf{then}\ 1\ \mathsf{else\ if}\ s \in S_1\ \mathsf{then}\ \prod_{D \in K\ s} \int v\ \mathsf{d}D\ \mathsf{else}\ 0$$

For our formalization in Isabelle/HOL we used the proof by Baier [6].

Theorem 13 (Existence of an Optimal Scheduler) *There exists a* memoryless scheduler $\mathsf{sc} \in \Pi_{s \in S} K\ s$*, for which the reachability probability is maximal:*

$$\mathsf{P\text{-}sup}\ s\ (S_1\ U\ S_2) = \int_t \mathsf{P\text{-}sup}\ t\ (S_1\ U\ S_2)\ \mathsf{d}(sc\ s) \qquad \textit{for } s \in S_1$$

The proof in Isabelle/HOL is based on the proof by de Alfaro [2, Theorem 3.10].

In the rest of this section we want to state rules to prove upper and lower bounds of reachability probabilities. With a fixed memoryless scheduler sc we get a fully probabilistic system $\tau\ s\ t = \mathsf{pmf}\ (\mathsf{sc}\ s)\ t$. On such a system, we assume a set $N \subseteq S$ which is always reachable in S, i.e. there exists always an enabled path to N. The set N is usually the set of states for which we know that reachability is 0 or 1. We use the following theorem to reduce an inequality on two solution vectors (i.e. functions from state to probability) to the inequality only on N. Then, when validating upper bounds, we instantiate l with the maximum probability and u with the upper bound. When validating lower bounds, we instantiate u with the minimum probability and l with the lower bound.

Theorem 14 (Monotonicity of Solutions of Reachability Problems) *We have a fixed scheduler sc, which induces a Markov chain graph (hence also* acc*). Then for a lower bound l and an upper bound u it is enough to compare them only on N:*

$$\frac{\begin{array}{c}\forall s \in S \setminus N.\ \int u\ \mathsf{d}(sc\ s) \leq u\ s \qquad \forall s \in S \setminus N.\ l\ s \leq \int l\ \mathsf{d}(sc\ s) \\ \forall s \in N.\ l\ s \leq u\ s \qquad \forall s \in S.\ \exists t \in N.\ (s,t) \in \mathsf{acc}\end{array}}{\forall s \in S.\ l\ s \leq u\ s}$$

This theorem is based on Baier and Katoen [7, Theorem 10.19], which is weaker as it only shows equality instead of inequality. However, it is straightforward to adapt their proof to our theorem. With this, Theorem 12, and the scheduler obtained by Theorem 13, we validate upper bounds as follows:

Theorem 15 (Bound for Maximal Probability)

$$\frac{\begin{array}{c}\forall s \in S - S_1 - S_2.\ u\,s = 0 \qquad \forall s \in S_2.\ u\,s = 1 \qquad \forall s \in S_1, D \in K\,s.\ \int u\ \mathsf{d}D \leq u\ s \\ \forall s \in S_1.\ \exists t \in S_2.\ (s,t) \in \{(s,t) \mid s \in S_1 \wedge t \in \bigcup (K\,s)\}^*\end{array}}{\forall s \in S.\ \mathsf{P\text{-}sup}\ s\ (S_1\ U\ S_2) \leq u\,s}$$

The theorem for minimal probability is similar. However, for each state and each *scheduler* we need to reach S_2.

Theorem 16 (Bound for Minimal Probability)

$$\frac{\begin{array}{c}\mathsf{wf}\ R \\ \forall s \in S - S_1 - S_2.\ l\,s = 0 \qquad \forall s \in S_2.\ l\,s = 1 \qquad \forall s \in S_1, D \in K\,s.\ l\ t \leq \int l\ \mathsf{d}D \\ \forall s \in S_1, D \in K\,s.\ l\,s \neq 0 \longrightarrow \exists t \in D.\ t \in S_2 \vee (t \in S_1 \wedge l\,t \neq 0 \wedge (t,s) \in R)\end{array}}{\forall s \in S.\ l\ s \leq \mathsf{P\text{-}inf}\ s\ (S_1\ U\ S_2)}$$

The well-founded relation R guarantees that no matter which action a scheduler chooses we can probabilistically choose a state with a smaller number of steps to reach a target state. These two theorems allow us to validate bounds for the maximal and minimal probability without knowing an optimal memoryless scheduler.

4.3 Certifying Model Checking Results

From Theorem 15 and 16 we derive a certification algorithm, presented in Fig. 3. This algorithm will allow us to certify upper (lower) bounds of the maximal (minimal) probability of a reachability problem.

The certification algorithm gets as input the Markov decision process, the reachability problem, and the certificates for the maximum and minimum probabilities as input values. They are represented as follows: The states and actions are represented as natural numbers. Each entry in distr is indexed by state and contains a list of distributions indexed by action. The distributions are represented as an association list from state to rational number. So the certificate represents the MDP interpreted as S and K and the RP interpreted as S_1, and S_2. See the input and the interpretation in Fig. 3. The algorithm needs to certify that the solutions sol^+ and sol^- fulfill

$$\forall s \in S.\ \mathsf{sol}^-\ s \leq \mathsf{P\text{-}inf}\ s\ (S_1\ U\ S_2) \wedge \mathsf{P\text{-}sup}\ s\ (S_1\ U\ S_2) \leq \mathsf{sol}^+\ s\ . \tag{4}$$

Input

n	:: nat
distr	:: (nat × rat) list list array
$\mathsf{st}_1, \mathsf{st}_2$	:: bool array
$\mathsf{sol}^+, \mathsf{sol}^-$	:: rat array
wit^+	:: (nat × nat × nat) array
wit^-	:: (nat list × nat) array

Interpretation

$S = \{0, \dots, n-1\}$
$K\ s = \{\mathsf{distr}[s][i] \mid i < \mathsf{distr}[s]\}$
$S_1 = \{i < n \mid \mathsf{st}_1[i]\}, S_2 = \{i < n \mid \mathsf{st}_2[i]\}$

Implementation

$$\begin{aligned}
&\mathsf{valid\text{-}RP} && = 0 < n = |\mathsf{distrs}| = |\mathsf{st}_1| = |\mathsf{st}_2| \wedge \\
&&& \quad \forall i < n.\ \neg(\mathsf{st}_1[i] \wedge \mathsf{st}_2[i]) \wedge \mathsf{distrs}[i] \neq [] \wedge \forall d \in \mathsf{distrs}[i].\ \mathsf{valid\text{-}D}\ d \\
&\mathsf{valid\text{-}D}\ d && = \mathsf{distinct}\ (\mathsf{map}\ \mathsf{fst}\ d) \wedge \textstyle\sum(\mathsf{map}\ \mathsf{snd}\ d) = 1 \wedge \forall (j, x) \in d.\ 0 \leq x \wedge j < n \\
&\mathsf{valid\text{-}C}\ s\ w\ (\bowtie)\ c && = |s| = |w| = n \wedge \\
&&& \quad \forall i < n.\ \mathsf{if}\ \mathsf{st}_2[i]\ \mathsf{then}\ s[i] = 1\ \mathsf{else}\ \mathsf{if}\ \neg\mathsf{st}_1[i]\ \mathsf{then}\ s[i] = 0\ \mathsf{else} \\
&&& \qquad \Big(\forall d \in \mathsf{distrs}[i].\ \mathsf{spmult}\ d\ s \bowtie s[i]\Big) \wedge \\
&&& \qquad 0 \leq s[i] \wedge \Big(0 < s[i] \longrightarrow c\ \mathsf{distr}[i]\ w[i]\Big) \\
&\mathsf{chk}^+\ D\ ((j, a), o) && = j < n \wedge a < |D| \wedge \mathsf{snd}\ \mathsf{wit}^+[j] < o \wedge D[a]@j \neq 0 \wedge 0 < \mathsf{sol}^+[j] \\
&\mathsf{chk}^-\ D\ (J, o) && = \forall j \in J \| d \in D.\ j < n \wedge \mathsf{snd}\ \mathsf{wit}^-[j] < o \wedge d@j \neq 0 \wedge 0 < \mathsf{sol}^-[j] \\
&\mathsf{valid} && = \mathsf{valid\text{-}RP} \wedge \mathsf{valid\text{-}C}\ \mathsf{sol}^+\ \mathsf{wit}^+\ (\leq)\ \mathsf{chk}^+ \wedge \mathsf{valid\text{-}C}\ \mathsf{sol}^-\ \mathsf{wit}^-\ (\geq)\ \mathsf{chk}^-
\end{aligned}$$

Fig. 3 The MDP reachability checker

For the logic, arrays $A :: \alpha$ array and lists $xs :: \alpha$ list are isomorphic. For an association list $xs :: (\alpha \times \beta)$ list, the lookup function $xs@a$ returns b for the first occurrence of (a, b) in xs. We write $|A|$ for the size of an array A and $A[i]$ for its i-th element. Functions in Isabelle/HOL are always total, so when an array is accessed above its size or a lookup on an association list fails they return a fixed but unknown element.

The assumptions in Theorem 15 require us to always guarantee to reach S_2 for a non-zero solution. For Theorem 16 a well-founded relation is required on the state set, also to guarantee that S_2 is reached. To validate these assumptions we use the witnesses wit^+ and wit^-. The second part of the witnesses is the number of steps necessary to reach S_2. If we follow the action and state information stored in the first part of wit^+ we always reach S_2. For wit^-, the action is chosen by the scheduler, so wit^+ stores for each action a state to go into the direction of S_2.

Before we validate the solutions, we first check that the MDP and the RP are correctly represented. For each association list in distrs, valid-D checks that the lookup table only contains unique keys, and represents a probability distribution.

The validation function valid-C is parameterized in the solution s, the witness w, the order $\bowtie$, and the witness checker c. For each state and for each action it checks that the solution is an upper or lower bound, depending on the order $\bowtie$. The function spmult $d\ s$ implements the sparse vector-vector multiplication, i.e. spmult $d\ s = \sum_{i<n} d@i \cdot s[i]$. For each state with a positive solution, we check that the witness points towards states in S_2, this check is done by c.

For the upper and lower bound validation, we only need to define the witness checking. The function c is called for each state, and gets a distribution for each action and the witness information. For the upper bound, the witness is an enabled state j

when the action a is chosen. It should have a smaller number of steps to reach S_2. This guarantees that we always have an enabled path to S_2.

For the lower bound, the first part of the witness is a list of enabled states J. For each action d there is an enabled state j with a smaller number of steps to reach S_2. This guarantees that for each strategy we find a path to S_2 with non-zero probability. We use the parallel list quantifier $\forall j \in J \| d \in D.\ P\ j\ d$ defined as $|J| = |D| \wedge \forall i < |J|.\ P\ J[i]\ D[i]$.

We show soundness using Theorem 15 for the upper bound, and Theorem 16 for the lower bound. For Theorem 16 we define $s\ R\ t$ as $\mathsf{snd\ wit}^-[s] > \mathsf{snd\ wit}^-[t]$. For Theorem 15 we can construct the necessary path to S_2 for each $s \in S_1$ by following the first part of the witnesses.

Corollary 6 (Soundness) *When* valid *returns true, then Eq. (4) holds.*

The runtime complexity to validate a certificate is linear in the size of the problem, i.e. it is $\mathcal{O}(M)$ where M is the number of enabled edges (s, α, t) with $0 < \tau\ s\ \alpha\ t$. The certificate does not contain the optimal schedulers. It is only necessary to provide one scheduler which certifies a lower (or upper) bound. We hope that the approach presented by Haddad and Monmege [24] can be used to produce such bounds.

5 Probabilistic Guarded Command Language (pGCL)

The *probabilistic Guarded Command Language (pGCL)* is a simple imperative language allowing probabilistic and non-deterministic choice. An extensive discussion is given by McIver and Morgan [42]. However, they miss the relation of the expectation transformer semantics to an operational semantics. Such a correspondence proof is given by Gretz *et al.* [23]. Here the reward of a run is computed using the program state at termination. The correspondence proof in [23] computes the minimal expectation in the MDP by summing the probabilities of all outcomes. In this section we prove the same statement in Isabelle/HOL by transferring a least fixed point through the minimal expectation. We do not use the pGCL formalization by Cock [13] as he uses a shallow embedding, while we need a deep embedding of pGCL programs to construct the corresponding MDP. However, the work described in this section could be used to connect the formalization of MDPs and the work on verification condition generators for pGCL by Cock.

Definition 31 (pGCL Syntax) We define the syntax of pGCL programs as a datatype over a state space σ, with state transformers $\sigma \Rightarrow \sigma$:

$$\begin{aligned}\sigma\ \mathsf{pgcl} = {} & \mathsf{Skip} \mid \mathsf{Abort} \mid \mathsf{Assign}\ (\sigma \Rightarrow \sigma) \\ \mid {} & \mathsf{Seq}\ (\sigma\ \mathsf{pgcl})\ (\sigma\ \mathsf{pgcl}) \\ \mid {} & \mathsf{Par}\ (\sigma\ \mathsf{pgcl})\ (\sigma\ \mathsf{pgcl}) \\ \mid {} & \mathsf{If}\ (\sigma \Rightarrow \mathsf{bool})\ (\sigma\ \mathsf{pgcl})\ (\sigma\ \mathsf{pgcl}) \\ \mid {} & \mathsf{Prob}\ (\mathsf{bool\ pmf})\ (\sigma\ \mathsf{pgcl})\ (\sigma\ \mathsf{pgcl}) \\ \mid {} & \mathsf{While}\ (\sigma \Rightarrow \mathsf{bool})\ (\sigma\ \mathsf{pgcl})\ (\sigma\ \mathsf{pgcl})\end{aligned}$$

Instead of using real numbers in the interval $[0; 1]$ we use the isomorphic type bool pmf. The type of states σ does not have any structure, we directly embed the operations in our programs, c.f. the state update in Assign, or the predicates in If and While.

Definition 32 (pGCL Denotational Semantics as Expectation Transformer) We specify the weakest pre-expectation transformer wp on the complete lattice of $\sigma \Rightarrow$ ereal where σ is the type of states. We need to restrict the expectations to be non-negative, hence Abort does not return bottom but the constant 0, and the fixed point in While needs to be clamped by 0.

$$
\begin{aligned}
&\mathsf{wp} && :: \sigma\ \mathsf{pgcl} \Rightarrow (\sigma \Rightarrow \mathsf{ereal}) \Rightarrow (\sigma \Rightarrow \mathsf{ereal})\\
&\mathsf{wp}\ \mathsf{Skip}\ f && = f\\
&\mathsf{wp}\ \mathsf{Abort}\ f && = (\lambda s.\ 0)\\
&\mathsf{wp}\ (\mathsf{Assign}\ u)\ f && = f \circ u\\
&\mathsf{wp}\ (\mathsf{Seq}\ c_1\ c_2)\ f && = \mathsf{wp}\ c_1\ (\mathsf{wp}\ c_2\ f)\\
&\mathsf{wp}\ (\mathsf{If}\ b\ c_1\ c_2)\ f && = (\lambda s.\ \mathsf{if}\ b\ s\ \mathsf{then}\ \mathsf{wp}\ c_1\ f\ s\ \mathsf{else}\ \mathsf{wp}\ c_2\ f\ s)\\
&\mathsf{wp}\ (\mathsf{Par}\ c_1\ c_2)\ f && = \mathsf{wp}\ c_1\ f \sqcap \mathsf{wp}\ c_2\ f\\
&\mathsf{wp}\ (\mathsf{Prob}\ p\ c_1\ c_2)\ f && = (\lambda s.\ \mathsf{pmf}\ p\ \mathsf{True} * \mathsf{wp}\ c_1\ f\ s + \mathsf{pmf}\ p\ \mathsf{False} * \mathsf{wp}\ c_2\ f\ s)\\
&\mathsf{wp}\ (\mathsf{While}\ b\ c)\ f && = \mathsf{lfp}\ (\lambda X\ s.\ \mathsf{if}\ b\ s\ \mathsf{then}\ \mathsf{wp}\ c\ ((\lambda s.\ 0) \sqcup X)\ s\ \mathsf{else}\ f\ s)
\end{aligned}
$$

As small-step semantics we give a Markov decision process on σ pgcl $\times\ \sigma$, i.e. the product of program points and states. The MDP will provide us with a minimal expectation on traces on these states.

Definition 33 (pGCL Small-step Semantics as Markov Decision Process) The step-function computes the transition from a configuration consisting of the current program and state to the possible following configurations.

$$
\begin{aligned}
&\ll\text{-},\text{-}\gg && :: \sigma\ \mathsf{pgcl} \Rightarrow \sigma \Rightarrow (\sigma\ \mathsf{pgcl} \times \sigma)\ \mathsf{pmf\ set}\\
&\ll c, s \gg && = \{\mathsf{return\text{-}pmf}\ (c, s)\}\\
&\mathsf{step} && :: (\sigma\ \mathsf{pgcl} \times \sigma) \Rightarrow (\sigma\ \mathsf{pgcl} \times \sigma)\ \mathsf{pmf\ set}\\
&\mathsf{step}\ (\mathsf{Skip}, s) && = \ll \mathsf{Skip}, s \gg\\
&\mathsf{step}\ (\mathsf{Abort}, s) && = \ll \mathsf{Abort}, s \gg\\
&\mathsf{step}\ (\mathsf{Assign}\ u, s) && = \ll \mathsf{Skip}, u\ s \gg\\
&\mathsf{step}\ (\mathsf{Seq}\ c_1\ c_2, s) && =\\
&\quad (\mathsf{map\text{-}pmf}\ (\lambda(c_1', s').\ (\mathsf{if}\ c_1' = \mathsf{Skip}\ \mathsf{then}\ c_2\ \mathsf{else}\ \mathsf{Seq}\ c_1'\ c_2, s'))) \text{`}\ \mathsf{step}\ (c_1, s)\\
&\mathsf{step}\ (\mathsf{If}\ b\ c_1\ c_2, s) && = \mathsf{if}\ b\ s\ \mathsf{then}\ \mathsf{step}\ (c_1, s)\ \mathsf{else}\ \mathsf{step}\ (c_2, s))\\
&\mathsf{step}\ (\mathsf{Par}\ c_1\ c_2, s) && = \ll c_1, s \gg \cup \ll c_2, s \gg\\
&\mathsf{step}\ (\mathsf{Prob}\ p\ c_1\ c_2, s) && = \{\mathsf{map\text{-}pmf}\ (\lambda b.\ (\mathsf{if}\ b\ \mathsf{then}\ c_1\ \mathsf{else}\ c_2, s))\ p\}\\
&\mathsf{step}\ (\mathsf{While}\ b\ c, s) && = \mathsf{if}\ b\ s\ \mathsf{then} \ll \mathsf{Seq}\ c\ (\mathsf{While}\ b\ c), s \gg \mathsf{else} \ll \mathsf{Skip}, s \gg
\end{aligned}
$$

This function defines the kernel of our MDP, hence we get a function E-inf representing the minimal expectation. The minimal expected outcome of a pGCL program is now defined as its reward on this MDP.

Definition 34 (Reward of a Run of a pGCL Program) The reward of a run is the value of the observation function f when the program terminates, i.e. when the current command is Skip.

$$
\begin{aligned}
&\mathsf{r} && :: (\sigma \Rightarrow \mathsf{ereal}) \Rightarrow (\sigma\ \mathsf{pgcl} \times \sigma)\ \mathsf{stream} \Rightarrow \mathsf{ereal}\\
&\mathsf{r}\ f && = \mathsf{lfp}\ (\lambda F\ ((c, s) \cdot \omega).\ 0 \sqcup \mathsf{if}\ c = \mathsf{Skip}\ \mathsf{then}\ f\ s\ \mathsf{else}\ F\ \omega)
\end{aligned}
$$

The supremum with 0 is necessary to get a reward of 0 for non-terminating runs. Also, with the supremum we guarantee that r is always non-negative.

For this reward function we can easily derive that the functional is non-negative, monotone, continuous, and measurable, hence also r is non-negative, measurable, and monotone. From this we derive that r nicely transfers through $\mathsf{E\text{-}inf}$ by instantiating Lemma 27:

$$\mathsf{E\text{-}inf}\, s\, (\mathsf{r}\, f) = \mathsf{lfp} \left(\lambda F\, x.\ \prod_{D \in \mathsf{step}\ x} \int_{(c,s)} \mathsf{if}\ c = \mathsf{Skip}\ \mathsf{then}\ f\, s\ \mathsf{else}\ F\, (c, s)\ \mathrm{d}D \right) s \quad (5)$$

First we prove the following equations for Skip and Seq. These equations will be reused multiple times in the final correspondence proof.

Lemma 28 (E-inf at a Skip state)

$$\frac{0 \leq f\ s}{\mathsf{E\text{-}inf}\ (\mathsf{Skip}, s)\ (\mathsf{r}\ f) = f\ s}$$

Lemma 29 (E-inf at a Seq state)

$$\frac{\forall s.\ 0 \leq f\ s}{\mathsf{E\text{-}inf}\ (\mathsf{Seq}\, c_1\, c_2, s)\ (\mathsf{r}\ f) = \mathsf{E\text{-}inf}\ (c_1, s)\ (\mathsf{r}\ (\lambda s'.\ \mathsf{E\text{-}inf}\ (c_2, s')\ (\mathsf{r}\ f)))}$$

Proof (by anti-symmetry) We use the representation of $\mathsf{E\text{-}inf}$ over r as least fixed point and perform induction in both directions. What remains is applying one unfolding of the least fixed point and using Lemma 28. □

The minimal expectation $\mathsf{E\text{-}inf}$ over r is well-defined and we prove our final correspondence theorem, equating the denotational (weakest pre-expectation transformer) to the operational (MDP) semantics:

Theorem 17 (Correspondence) *Assume a non-negative observation function $f :: \sigma \Rightarrow \mathsf{ereal}$. Then the minimal expectation on the MDP equals the transformation by the weakest pre-expectation transformer on f:*

$$\mathsf{E\text{-}inf}\ (c, s)\ (\mathsf{r}\ f) = \mathsf{wp}\ c\ f\ s$$

Proof (by structural induction on c, generalized in s and f) All cases besides While are simple rewrites either directly with Eq. (5) or by Lemmas 28 and 29. For the While-case we prove that $\mathsf{E\text{-}inf}$ over While is itself a least fixed point over $\mathsf{E\text{-}inf}$:

$$\mathsf{E\text{-}inf}\, (\mathsf{While}\, g\, c, s)\, (\mathsf{r}\, f) = \mathsf{lfp}\, (\lambda F\, s.\ \mathsf{if}\ g\, s\ \mathsf{then}\ \mathsf{E\text{-}inf}\, (c, s)\, (r\, (0 \sqcup F))\ \mathsf{else}\ f\, s)\, s \quad (6)$$

That the least fixed point at the right is less or equal to $\mathsf{E\text{-}inf}\, (\mathsf{While}\, g\, c, s)\, (\mathsf{r}\, f)$ is shown by fixed point induction: either $g\, s$ holds, then we have a sequential application and finish the proof with Lemma 29, or $g\, s$ does not hold, and the behavior is equal to Skip.

The other direction is more involved: we use the following functional w to unroll the while-loop as a single least fixed point, then we massage it into the right form.

$$\mathsf{w}\ F\ x = \prod_{D \in \mathsf{step}\, x} \int_{(d,s)} \begin{Bmatrix} F\ (d, s) & \text{if } d \neq \mathsf{Skip} \\ F\ (c, s) & \text{if } d = \mathsf{Skip} \wedge g\, s \\ f\ s & \text{if } d = \mathsf{Skip} \wedge \neg g\, s \end{Bmatrix} \mathrm{d}D$$

Then we prove the following rule, by induction on $\mathsf{E\text{-}inf}$ and generalizing over s, t and d to abstract the position in the while-loop unrolling:

$$\frac{(t = \mathsf{While}\ g\ c \wedge d = c \wedge g\ s) \vee t = \mathsf{Seq}\ d\ (\mathsf{While}\ g\ c)}{\mathsf{E\text{-}inf}\ (t, s)\ (r\ f) \leq \mathsf{lfp}\ \mathsf{w}\ (d, s)} \tag{7}$$

We finish the proof with the following calculation:

$$\begin{aligned}
&\mathsf{E\text{-}inf}\ (\mathsf{While}\ g\ c, s)\ (r\ f) \\
&\quad = && \langle \textit{by Eq. (5) and computing one iteration} \rangle \\
&\mathsf{if}\ g\,s\ \mathsf{then}\ \mathsf{E\text{-}inf}\ (\mathsf{Seq}\,c\,(\mathsf{While}\,g\,c), s)\,(r\,f)\ \mathsf{else}\ f\,s \\
&\quad \leq && \langle \textit{by Eq. (7) where } t = \mathsf{While}\,g\,c,\ d = c \textit{ and } g\,s. \rangle \\
&\mathsf{if}\ g\,s\ \mathsf{then}\ \mathsf{lfp}\ \mathsf{w}\,(c, s)\ \mathsf{else}\ f\,s \\
&\quad \leq && \langle \textit{by def. of } \mathsf{w},\ \textit{Eq. (5), and diagonal rule of Lemma 2.} \rangle \\
&\mathsf{if}\ g\,s\ \mathsf{then}\ \mathsf{lfp}\,(\lambda F\,x.\ \mathsf{E\text{-}inf}\,x\,(r\,(\lambda s.\ \mathsf{if}\ g\,s\ \mathsf{then}\ 0 \sqcup F\,(c, s)\ \mathsf{else}\ f\,s))\,(c, s)\ \mathsf{else}\ f\,s \\
&\quad = && \langle \textit{by rolling rule of Lemma 2.} \rangle \\
&\mathsf{lfp}\,(\lambda F\,s.\ \mathsf{if}\ g\,s\ \mathsf{then}\ \mathsf{E\text{-}inf}\,(c, s)\,(r\,(0 \sqcup F))\ \mathsf{else}\ f\,s)\,s
\end{aligned}$$

□

In this proof the least fixed point transformations are especially helpful. The diagonal rule of Lemma 2, i.e. $\mathsf{lfp}\ (\lambda x.\ \mathsf{lfp}\ (f\ x)) = \mathsf{lfp}\ (\lambda x.\ f\ x\ x)$, is the central part of our proof, to unroll the entire While-loop.

With the correspondence theorem we get now two different views on the behavior of pGCL programs.

6 Related Work

The first formalizations of measure and probability theory in a HOL theorem prover was Hurd's probabilistic monad [35]. He formalizes the foundations of measure theory and used it to construct a probability space on $\mathsf{nat} \Rightarrow \mathsf{bool}$. He verifies programs using a random number generator as functions reading bits from traces in this probability space. To construct a specific probability distribution, Hurd's approach requires the construction of a random variable reading bits from the random monad. In our approach we have the option to directly use the probability mass function to get a certain probability distribution in a probabilistic program.

Liu *et al.* [40] classify finite-state Markov chains in HOL4 based on the probability monad of [35]. They do not construct the trace space of Markov chains, and only give an axiomatic definition of finite-state Markov chains as stochastic processes. Their transition functions are modeled as $\sigma \Rightarrow \sigma \Rightarrow \mathsf{real}$ for which they do not provide a compositional way to construct them. They show for aperiodic Markov chains that the limit of the marginal probability is a stationary distribution. Their proof is only for finite-state Markov chains, hence there is no construction of the product Markov chain and generally the proofs are less involved.

The simpler approach, which does not require measure theory, is to employ expectation transformers or the Giry monad on distributions with countable support. Hurd *et al.* [36] formalizes an expectation transformer approach which even allows demonic (non-deterministic) choice. He introduces pGCL programs as explicit datatypes in HOL and provides a verification condition generator (VCG). This work was extended by Celiku and McIver [12] which introduces a VCG to prove upper bounds of

the expected running time of pGCL programs. A similar formalization was done by Cock [13] in Isabelle/HOL, providing a VCG on the shallow embedding of pGCL programs. Audebaud and Paulin-Mohring [3] develop a shallow embedding of probabilistic programs in Coq by formalizing the Giry monad on discrete distributions. Their usage of sub-probability mass functions allows them to define a least fixed point on discrete distributions. This is not directly possible in our formalization, as there is no general order on probability mass functions, only on sub-probability mass functions. There are further formalizations of probability mass functions in Coq. Affeldt, Hagiwara and Sénizergues [1] use them to formalize information theory. Petcher and Morrisett [45] as well as Rand and Zdancewic [48] introduce probabilistic programs generating finite distributions. In Isabelle/HOL, Lochbihler [41] formalizes probabilistic programs interacting with oracles. For this he introduces sub-probability distributions similar to [3] and builds on top of them generative probabilistic values modeling the interaction with an oracle. The formalizations in [1, 45, 48] are restricted to finite probability distributions. Besides Liu *et al.* [40], none of these approaches relate transition systems with trace spaces, either they are only concerned with the trace space [35], or only with the transition system [3, 13, 36, 41].

The work presented in this paper is founded on Isabelle/HOL's measure and probability theory presented in [26, 31, 33]. The previous work constructs finite-state Markov chains to verify probabilistic model checking. We extend this to countable (possibly infinite) discrete state spaces, and further analysis concepts such as (positive) recurrence, the period of a state and stationary distributions. Our new approach to Markov chains subsumes the old formalization in [31, 33]. The theories in the AFP [32] were ported to our new formalization. Popescu *et al.* [47] use the work presented in this paper to model programs with probabilistic choice and parallel composition as infinite-state Markov chains. The Giry monad was introduced by Eberl *et al.* [16], the type of probability mass functions was introduced by Hölzl *et al.* [30].

An earlier formalization of Lebesgue integration in Isabelle/HOL was done by Richter [49]. A formalization of Markov kernels for Isabelle/HOL was already done by Berg [8], unfortunately the author was not aware of this work until recently. Berg's formalization uses type classes to assign σ-algebras to types. This provided better automation but is less flexible: Isabelle only allows one type class instance (i.e. only one σ-algebra) per type. For example this does not allow product σ-algebras on the function space with explicit index sets.

On a non-formalized view, the work by Monniaux [43] is very similar to the mathematical MDP analysis presented in this paper. While the formalization happened without knowing the work by Monniaux, the developed Isabelle theories are very similar: both (1) use fixed points to specify properties on the trace space, (2) compute integrals of these properties by transfer theorems, and (3) finally equate an expectation semantics to semantics on the trace space.

A fully automatic approach to analyzing Markov chains exists with probabilistic model checkers, like PRISM [39] or MRMC [37]. Here we have the usual dichotomy between model checking and theorem proving, where model checking works fully automatic, but only for a fixed Markov chain and for fixed properties, e.g. reachability and average cost. Some approaches to go beyond fixed Markov chains are the use of probabilistic push-down automaton (or equivalent, recursive Markov chains) [17, 19] or allowing to specify the transition probabilities of Markov chains and MDPs as polynomials [15].

7 Discussion and Conclusion

We chose an unusual approach to the formalization of Markov chains and MDPs in Isabelle/HOL, compared to mathematical textbooks. The abstract concepts of Markov chains and MDPs itself are not changed, but we stick to an order theoretic and coalgebraic view on the trace space of Markov chains (e.g. Theorem 1 in Section 3) or define important properties as fixed points (e.g. Section 2.5). We rely on a well-developed theory of lattices, fixed points (e.g. Section 2.1) and coinductive streams (e.g. Section 2.5), allowing us to provide generic theorems and powerful induction principles. The formalization is ~ 7.600 lines of theory files, not containing the material in Section 2.

With our measure theoretic tools it is easy to show that Eq. (1) is equivalent to Eq. (2) (assuming time-homogeneity); we do so in Section 3.5. However, for our purpose using Eq. (2) produces shorter and more appealing proofs:

- Typical example of probabilities or expectations we want to compute are the probability until a certain set of states is reached or the expected hitting time for a set of states. These are easily computed by solving a fixed point equation, as mentioned in the previous paragraph, and the iteration rule given in Eq. (2).
- The iteration rule allows us to work with expectations. While an iteration rule can be derived from Eq. (1) the other direction is easier: just instantiate f with the indicator function, the rest follows by induction.
- In Eq. (1) we can only reason about one prefix $s_0, s_1, \ldots, s_t$. It is possible, but not always easy to derive the probability for more complex sets of traces. In Eq. (2) we reason about the expectation of a function f, which subsumes the probability for a prefix.

Similar to Eq. (2), we prove iteration rules for MDPs, here on the maximal and minimal probability; see Section 4.1.

While many properties known from model checking are nicely described as fixed points, properties known from classical Markov chain analysis are different. Properties like stationary distributions, recurrence or time bounded probabilities are not definable as fixed point over the trace space. Despite that difference, we can still use our Markov chain framework to formalize such properties. As mentioned in the previous paragraph our approach subsumes the usual formalization of Markov chains.

In our experience the usage of least and greatest fixed points leads to *short* and *natural* proofs, i.e. natural in the sense that many involved definitions and proofs are about fixed points and often involve theorems generic on complete lattices. Concrete examples that the resulting proofs are shorter are found in the proofs for Theorem 3 and Theorem 4. The previous proofs where done without using and manipulating fixed points. After rewriting the proofs using fixed points the size of the proofs where reduced by around one third, c.f. Section 3.3. Another example is the formalization of Theorem 17 to relate the denotational and operational semantics on PGCL in Section 5. The entire formalization requires only ≈ 260 lines of theory in Isabelle/HOL. And it is natural as (1) all operations and proofs working on expectation of infinite traces (or infinite traces itself) could be reduced to fixed points and (2) after proving Eq. (5) most parts of the proof are massaging fixed points with general fixed point theorems.

For us it was important to show the equivalence between the stochastic process and the transition matrix interpretation of Markov chains. The measure space constructions in Section 3.2 made this possible. With this equivalence we not only allow the application of our theory to Markov chains defined as stochastic processes, but give

a stronger guarantee that our formalization is generically applicable. As an unfortunate counter-example, the formalization by Liu *et al.* [40] does not fulfill this guarantee. In their formalization of the time-homogeneous property they do not include the assumption that the states are reachable. Hence the Markov chains they handle are restricted to Markov chains where the probability to reach a state is *always* non-zero, i.e. $\Pr(X\ t = x) \neq 0$ for all t.

One restriction of our formalization is that the trace space $T\,x$ assumes a starting state x. In Section 3.5 we introduce also T' parametric over an initial distribution I of starting states. This is a simple combination of I and T using the bind operator of the Giry monad.

We have shown that the formalization of Markov decision processes can be used (1) to easily construct models — e.g. in the pGCL proof — and (2) to analyze properties on the MDP — the certification of reachability results. Here our generalized view on Markov kernels with arbitrary discrete state spaces caters for a concise construction of MDPs. There is no need to restrict the state space to be countable.

As future work, we plan to extend and generalize Markov chains to continuous-time Markov chains with discrete state space. In the course of this we extended also Markov chains from discrete state spaces to state spaces on arbitrary measure spaces. The rough diamond presented in [27] extends the pGCL semantics to provide semantics for expected running time. This may enable the analysis of cryptographic protocols were the operational semantics usually provides a clearer cost model. It should be also possible to connect to other formalizations of probabilistic programs, i.e. provide operational semantics for Cock's pGCL formalization [13] and Lochbihler's generative probabilistic values [41].

Acknowledgments.

The author wants to thank Tobias Nipkow, Dmitriy Traytel, and Fabian Immler and the anonymous reviewers for suggesting many textual improvements.

References

1. Affeldt, R., Hagiwara, M., Sénizergues, J.: Formalization of Shannon's theorems. J. Autom. Reasoning **53**(1), 63–103 (2014)
2. de Alfaro, L.: Formal verification of probabilistic systems. Ph.D. thesis, Stanford University (1997). Technical report STAN-CS-TR-98-1601
3. Audebaud, P., Paulin-Mohring, C.: Proofs of randomized algorithms in Coq. Science of Computer Programming **74**(8), 568–589 (2009). Special Issue on Mathematics of Program Construction (MPC 2006)
4. Avigad, J., Hölzl, J., Serafin, L.: A formally verified proof of the Central Limit Theorem. CoRR **abs/1405.7012** (2014). URL `http://arxiv.org/abs/1405.7012`
5. Backhouse, R.C.: Galois connections and fixed point calculus. In: R.C. Backhouse, R.L. Crole, J. Gibbons (eds.) Algebraic and Coalgebraic Methods in the Mathematics of Program Construction, *LNCS*, vol. 2297, pp. 89–148 (2000)
6. Baier, C.: On the algorithmic verification of probabilistic systems. Habilitation, Universität Mannheim (1998)
7. Baier, C., Katoen, J.P.: Principles of Model Checking. The MIT Press (2008)
8. Berg, M.: Formal verification of cryptographic security proofs. Ph.D. thesis, Saarland University (2013)
9. Blanchette, J.C., Hölzl, J., Lochbihler, A., Panny, L., Popescu, A., Traytel, D.: Truly modular (co)datatypes for Isabelle/HOL. In: G. Klein, R. Gamboa (eds.) Interactive Theorem Proving (ITP 2014), *LNCS*, vol. 8558, pp. 93–110. Springer (2014)

10. Blanchette, J.C., Popescu, A., Traytel, D.: Unified classical logic completeness. In: S. Demri, D. Kapur, C. Weidenbach (eds.) Automated Reasoning (IJCAR 2014), *LNCS*, vol. 8562, pp. 46–60. Springer (2014)
11. Blazy, S., Paulin-Mohring, C., Pichardie, D. (eds.): Interactive Theorem Proving (ITP 2013), *LNCS*, vol. 7998. Springer (2013)
12. Celiku, O., McIver, A.: Cost-based analysis of probabilistic programs mechanised in HOL. Nord. J. Comput. **11**(2), 102–128 (2004)
13. Cock, D.: Verifying probabilistic correctness in Isabelle with pGCL. In: F. Cassez, R. Huuck, G. Klein, B. Schlich (eds.) Systems Software Verification (SSV 2012), *EPTCS*, vol. 102, pp. 167–178 (2012)
14. Davey, B.A., Priestley, H.A.: Introduction to Lattices and Order, second edn. Cambridge University Press (2002)
15. Daws, C.: Symbolic and parametric model checking of discrete-time markov chains. In: Z. Liu, K. Araki (eds.) Theoretical Aspects of Computing (ICTAC 2004), *LNCS*, vol. 3407, pp. 280–294 (2004)
16. Eberl, M., Hölzl, J., Nipkow, T.: A verified compiler for probability density functions. In: European Symposium on Programming (ESOP 2015), LNCS (2015)
17. Esparza, J., Kučera, A., Mayr, R.: Model checking probabilistic pushdown automata. In: Logic in Computer Science (LICS 2004), pp. 12–21 (2004)
18. Esparza, J., Lammich, P., Neumann, R., Nipkow, T., Schimpf, A., Smaus, J.: A fully verified executable LTL model checker. In: N. Sharygina, H. Veith (eds.) Computer Aided Verification (CAV 2013), *LNCS*, vol. 8044, pp. 463–478. Springer (2013)
19. Etessami, K., Yannakakis, M.: Recursive markov chains, stochastic grammars, and monotone systems of nonlinear equations. Journal of the ACM **56**(1), 1–66 (2009)
20. Giry, M.: A categorical approach to probability theory. In: Categorical Aspects of Topology and Analysis, *Lecture Notes in Mathematics*, vol. 915, pp. 68–85 (1982)
21. Gonthier, G., Norrish, M. (eds.): CPP 2013, *LNCS*, vol. 8307. Springer (2013)
22. Gouezel, S.: Ergodic theory. The Archive of Formal Proofs (2015). `https://www.isa-afp.org/entries/Ergodic_Theory.shtml`, (Formal proof development)
23. Gretz, F., Katoen, J., McIver, A.: Operational versus weakest pre-expectation semantics for the probabilistic guarded command language. Performance Evaluation **73**, 110–132 (2014)
24. Haddad, S., Monmege, B.: Reachability in mdps: Refining convergence of value iteration. In: Reachability Problems (RP 2014), *LNCS*, vol. 8762, pp. 125–137. Springer (2014)
25. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Tech. Rep. SICS/R90013, Swedish Institute of Computer Science (1994)
26. Hölzl, J.: Construction and stochastic applications of measure spaces in higher-order logic. Ph.D. thesis, Technische Universität München (2013)
27. Hölzl, J.: Formalising semantics for expected running time of probabilistic programs. In: C.J. Blanchette, S. Merz (eds.) Interactive Theorem Proving (ITP 2016), *LNCS*, vol. 9807, pp. 475–482. Springer (2016)
28. Hölzl, J., Heller, A.: Three chapters of measure theory in Isabelle/HOL. In: M.C.J.D. van Eekelen, H. Geuvers, J. Schmaltz, F. Wiedijk (eds.) Interactive Theorem Proving (ITP 2011), *LNCS*, vol. 6898, pp. 135–151. Springer (2011)
29. Hölzl, J., Immler, F., Huffman, B.: Type classes and filters for mathematical analysis in Isabelle/HOL. In: Blazy et al. [11], pp. 279–294
30. Hölzl, J., Lochbihler, A., Traytel, D.: A formalized hierarchy of probabilistic system types (proof pearl). In: C. Urban, X. Zhang (eds.) Interactive Theorem Proving (ITP 2015), *LNCS*, vol. 9236, pp. 203–220 (2015)
31. Hölzl, J., Nipkow, T.: Interactive verification of Markov chains: Two distributed protocol case studies. In: U. Fahrenberg, A. Legay, C. Thrane (eds.) Quantities in Formal Methods (QFM 2012), *EPTCS*, vol. 103. arXiv (2012)
32. Hölzl, J., Nipkow, T.: Markov models. The Archive of Formal Proofs (2012). `https://www.isa-afp.org/entries/Markov_Models.shtml`, (Formal proof development)
33. Hölzl, J., Nipkow, T.: Verifying pCTL model checking. In: C. Flanagan, B. König (eds.) Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2012), *LNCS*, vol. 7214, pp. 347–361 (2012)
34. Huffman, B., Kunčar, O.: Lifting and transfer: A modular design for quotients in Isabelle/HOL. In: Gonthier and Norrish [21], pp. 131–146
35. Hurd, J.: Formal verification of probabilistic algorithms. Ph.D. thesis, University of Cambridge (2002)

36. Hurd, J., McIver, A., Morgan, C.: Probabilistic guarded commands mechanized in HOL. Theoretical Computer Science **346**(1), 96–112 (2005)
37. Katoen, J.P., Zapreev, I.S., Hahn, E.M., Hermanns, H., Jansen, D.N.: The ins and outs of the probabilistic model checker MRMC. Performance Evaluation **68**, 90–104 (2011)
38. Kwiatkowska, M., Norman, G., Parker, D.: Stochastic model checking. In: M. Bernardo, J. Hillston (eds.) Formal Methods for the Design of Computer, Communication and Software Systems: Performance Evaluation (SFM 2007), *LNCS*, vol. 4486, pp. 220–270 (2007)
39. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: Computer Aided Verification (CAV 2011), *LNCS*, vol. 6806, pp. 585–591 (2011)
40. Liu, L., Hasan, O., Aravantinos, V., Tahar, S.: Formal reasoning about classified Markov chains in HOL. In: Blazy et al. [11], pp. 295–310
41. Lochbihler, A.: Probabilistic functions and cryptographic oracles in higher order logic. In: ESOP, *LNCS*, vol. 9632, pp. 503–531. Springer (2016)
42. McIver, A., Morgan, C.: Abstraction, Refinement And Proof For Probabilistic Systems. Monographs in Computer Science. Springer (2004)
43. Monniaux, D.: Abstract interpretation of programs as Markov decision processes. Science of Computer Programming **58**(1-2), 179–205 (2005). Special Issue on the Static Analysis Symposium (SAS 2003)
44. Paulson, L.C.: A fixedpoint approach to (co)inductive and (co)datatype definitions. In: G.D. Plotkin, C. Stirling, M. Tofte (eds.) Proof, Language, and Interaction, Essays in Honour of Robin Milner, pp. 187–212. The MIT Press (2000)
45. Petcher, A., Morrisett, G.: The foundational cryptography framework. In: POST, *LNCS*, vol. 9036, pp. 53–72. Springer (2015)
46. Pollard, D.: A Users's Guide to Measure Theoretic Probability. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press (2002)
47. Popescu, A., Hölzl, J., Nipkow, T.: Formalizing probabilistic noninterference. In: Gonthier and Norrish [21], pp. 259–275
48. Rand, R., Zdancewic, S.: VPHL: A verified partial-correctness logic for probabilistic programs. ENTCS **319**, 351–367 (2015). DOI 10.1016/j.entcs.2015.12.021
49. Richter, S.: Formalizing integration theory with an application to probabilistic algorithms. In: TPHOLs, *LNCS*, vol. 3223, pp. 271–286. Springer (2004)
50. Trivedi, K.S.: Probability & Statistics with Reliability, Queuing, and Computer Science Applications. Prentice-Hall (1982)
51. Woess, W.: Denumerable Markov Chains. European Mathematical Society (2009)