Step-Indexed Logical Relations for Probability

Aleš Bizjak and Lars Birkedal

Aarhus University
{abizjak,birkedal}@cs.au.dk

Abstract. It is well-known that constructing models of higher-order probabilistic programming languages is challenging. We show how to construct step-indexed logical relations for a probabilistic extension of a higher-order programming language with impredicative polymorphism and recursive types. We show that the resulting logical relation is sound and complete with respect to the contextual preorder and, moreover, that it is convenient for reasoning about concrete program equivalences. Finally, we extend the language with dynamically allocated references and show how to extend the logical relation to this language. We show that the resulting relation remains useful for reasoning about examples involving both state and probabilistic choice.

1 Introduction

It is well known that it is challenging to develop techniques for reasoning about programs written in probabilistic higher-order programming languages. A probabilistic program evaluates to a distribution of values, as opposed to a set of values in the case of nondeterminism or a single value in the case of deterministic computation. Probability distributions form a monad. This observation has been used as a basis for several denotational domain-theoretic models of probabilistic languages and also as a guide for designing probabilistic languages with monadic types [11, 17, 16]. Game semantics has also been used to give denotational models of probabilistic programming languages [6, 8] and a fully abstract model using coherence spaces for PCF extended with probabilistic choice was recently presented [9].

The majority of models of probabilistic programming languages have been developed using denotational semantics. However, Johann et.al. [10] developed operationally-based logical relations for a polymorphic programming language with effects. Two of the effects they considered were probabilistic choice and *global* ground store. However, as pointed out by the authors [10], extending their construction to local store and, in particular, higher-order local store, is likely to be problematic. Also, recently, operationally-based bisimulation techniques have been extended to call-by-value [4] and call-by-name [5] probabilistic extensions of PCF. The operational semantics of probabilistic higher-order programming languages has been extensively investigated in [12].

Step-indexed logical relations [1, 2] have proved to be a successful method for proving contextual approximation and equivalence for programming languages with a wide range of features, including computational effects.

In this paper we show how to extend the method of step-indexed logical relations to reason about contextual approximation and equivalence of probabilistic higher-order programs. To define the logical relation we employ biorthogonality [13, 15], together with step-indexing. Biorthogonality is used to ensure completeness of the logical relation with respect to contextual equivalence, but it also makes it possible to keep the value relations simple, see Figure 1. Moreover, the definition using biorthogonality makes it possible to "externalize" the reasoning in many cases when proving example equivalences. By this we mean that the reasoning reduces to algebraic manipulations of probabilities. This way, the quantitative aspects do not complicate the reasoning much, compared to the usual reasoning with step-indexed logical relations for deterministic languages or languages with nondeterministic choice. To define the biorthogonal lifting of value relations we use two notions of observation; the termination probability and its stratified version approximating it. We define these and prove the required properties in Section 3.

We develop our step-indexed logical relations for the call-by-value language $\mathbf{F}^{\mu,\oplus}$. This is system $\mathbf{F}$ with recursive types, extended with a single probabilistic choice primitive `rand`. The primitive `rand` takes a natural number n and reduces with uniform probability to one of $1, 2, \ldots, n$. Thus `rand` n represents the uniform probability distribution on the set $\{1, 2, \ldots, n\}$. We choose to add `rand` instead of just a single coin flip primitive to make the examples easier to write.

To show that the model is useful we use it to prove some example equivalences in Section 5. We show two examples based on parametricity. In the first example, we characterize elements of the universal type $\forall\alpha.\alpha \to \alpha$. In a deterministic language, and even in a language with nondeterministic choice, the only interesting element of this type is the identity function. However, since in a probabilistic language we not only observe the end result, but also the likelihood with which it is returned, it turns out that there are many more elements. Concretely, we show that the elements of the type $\forall\alpha.\alpha \to \alpha$ that are of the form $\Lambda.v$, for a value v, correspond precisely to *left-computable* real numbers in the interval $[0, 1]$. In the second example we show a free theorem involving functions on lists. We show additional equivalences in the Appendix, including the correctness of von Neumann's procedure for generating a fair sequence of coin tosses from an unfair coin, and equivalences from the recent papers using bisimulations [4, 5].

We add dynamically allocated references to the language and extend the logical relation to the new language in Section 6. For simplicity we only sketch how to extend the construction with first-order state. This already shows that an extension with general references can be done in the usual way for step-indexed logical relations. We conclude the section by proving a representation independence result involving both state and probabilistic choice.

2 The language $\mathbf{F}^{\mu,\oplus}$

The language is a standard pure functional language with recursive, universal and existential types with an additional choice primitive `rand`. The base types

include the type of natural numbers $\mathtt{nat}$ with some primitive operations. The grammar of terms e is

$$\begin{aligned} e ::= {} & x \mid \langle\rangle \mid \underline{n} \mid \langle e_1, e_2\rangle \mid \lambda x.e \mid \mathtt{inl}\ e \mid \mathtt{inr}\ e \mid \Lambda.e \mid \mathtt{pack}\, e \mid \mathtt{proj}_i\, e \mid e_1\, e_2 \mid \\ & \mid \mathtt{match}\,(e, x_1.e_1, x_2.e_2) \mid e[] \mid \mathtt{unpack}\ e_1\ \mathtt{as}\ x\ \mathtt{in}\ e_2 \mid \mathtt{unfold}\, e \mid \mathtt{fold}\, e \mid \\ & \mid \mathtt{rand}\, e \mid \mathtt{if}_1\ e\ \mathtt{then}\ e_1\ \mathtt{else}\ e_2 \mid \mathtt{P}\, e \mid \mathtt{S}\, e \end{aligned}$$

We write $\underline{n}$ for the numeral representing the natural number n and $\mathtt{S}$ and $\mathtt{P}$ are the successor and predecessor functions, respectively. For convenience, numerals start at $\underline{1}$. Given a numeral $\underline{n}$, the term $\mathtt{rand}\,\underline{n}$ evaluates to one of the numerals $\underline{1}, \ldots, \underline{n}$ with uniform probability. There are no types in the syntax of terms, e.g., instead of $\Lambda\alpha.e$ and $e\,\tau$ we have $\Lambda.e$ and $e[]$. This is for convenience only.

We write $\alpha, \beta, \ldots$ for *type variables* and $x, y, \ldots$ for *term variables*. The notation $\tau[\vec{\tau}/\vec{\alpha}]$ denotes the simultaneous capture-avoiding substitution of types $\vec{\tau}$ for the free type variables $\vec{\alpha}$ in the type τ; $e[\vec{v}/\vec{x}]$ denotes simultaneous capture-avoiding substitution of values $\vec{v}$ for the free term variables $\vec{x}$ in the term e.

We write **Stk** for the set of evaluation contexts given by the call-by-value reduction strategy. Given two evaluation contexts E, E' we define their composition $E \circ E'$ by induction on E in the natural way. Given an evaluation context E and expression e we write $E[e]$ for the term obtained by plugging e into E. For any two evaluation contexts E and E' and a term e we have $E[E'[e]] = (E \circ E')[e]$.

For a type variable context Δ, the judgment $\Delta \vdash \tau$ expresses that the free type variables in τ are included in Δ. The typing judgments are entirely standard with the addition of the typing of $\mathtt{rand}$ which is given by the rule

$$\frac{\Delta \mid \Gamma \vdash e : \mathtt{nat}}{\Delta \mid \Gamma \vdash \mathtt{rand}\ e : \mathtt{nat}}.$$

The complete set of typing rules are in the Appendix. We write $\mathfrak{T}(\Delta)$ for the set of types well-formed in context Δ, and $\mathfrak{T}$ for the set of *closed* types τ. We write $\mathbf{Val}\,(\tau)$ and $\mathbf{Tm}\,(\tau)$ for the sets of *closed* values and terms of type τ, respectively. We write **Val** and **Tm** for the set of *all*[1] *closed* values and closed terms, respectively. $\mathbf{Stk}\,(\tau)$ denotes the set of τ-accepting evaluation contexts, i.e., evaluation contexts E, such that given any closed term e of type τ, $E[e]$ is a typeable term. **Stk** denotes the set of all evaluation contexts.

For a typing context $\Gamma = x_1{:}\tau_1, \ldots, x_n{:}\tau_n$ with $\tau_1, \ldots, \tau_n \in \mathfrak{T}$, let

$$\mathbf{Subst}(\Gamma) = \{\gamma \in \mathbf{Val}^{\vec{x}} \mid \forall 1 \leq i \leq n.\ \gamma(x_i) \in \mathbf{Val}\,(\tau_i)\}$$

denote the set of type-respecting value substitutions. In particular, if $\Delta \mid \Gamma \vdash e : \tau$ then $\varnothing \mid \varnothing \vdash e\gamma : \tau\delta$ for any $\delta \in \mathfrak{T}^{\Delta}$ and $\gamma \in \mathbf{Subst}(\Gamma\delta)$, and the type system satisfies standard properties of progress and preservation and a canonical forms lemma.

The operational semantics of the language is a standard call-by-value semantics but weighted with $p \in [0, 1]$ which denotes the likelihood of that reduction.

[1] In particular, we do not require them to be typeable.

We write $\overset{p}{\rightsquigarrow}$ for the one-step reduction relation. All the usual β reductions have weight equal to 1 and the reduction from $\mathtt{rand}\,\underline{n}$ is

$$\mathtt{rand}\,\underline{n} \overset{\frac{1}{n}}{\rightsquigarrow} \underline{k} \qquad \text{for } k \in \{1, 2, \ldots, n\}.$$

The rest of the rules are given in Figure **??** in the Appendix. The operational semantics thus gives rise to a Markov chain with closed terms as states.

3 Observations and biorthogonality

We will use biorthogonality to define the logical relation. This section provides the necessary observation predicates used in the definition of the biorthogonal lifting of value relations to expression relations. Because of the use of biorthogonality the value relations (see Figure 1) remain as simple as for a language without probabilistic choice. The new quantitative aspects only appear in the definition of the biorthogonal lifting ($\top\top$-closure) defined in Section 4.1. Two kinds of observations are used. The probability of termination, $\mathbf{Pr}\,(e \Downarrow)$, which is the actual probability that e terminates, and its approximation, the *stratified* termination probability $\mathbf{Pr}\left(e \Downarrow^k\right)$, where $k \in \mathbb{N}$ denotes, intuitively, the number of computation steps. The stratified termination probability provides the link between steps in the operational semantics and the indexing in the definition of the interpretation of types.

The probability of termination, $\mathbf{Pr}\,(\cdot \Downarrow)$, is a function of type $\mathbf{Tm} \to \mathcal{I}$ where $\mathcal{I}$ is the unit interval $[0, 1]$. Since $\mathcal{I}$ is a pointed ω-cpo for the usual order, so is the space of all functions $\mathbf{Tm} \to \mathcal{I}$ with pointwise ordering. We define $\mathbf{Pr}\,(\cdot \Downarrow)$ as a fixed point of a continuous function Φ on this ω-cpo: Let $\mathcal{F} = \mathbf{Tm} \to \mathcal{I}$ and define $\Phi : \mathcal{F} \to \mathcal{F}$ as

$$\Phi(f)(e) = \begin{cases} 1 & \text{if } e \in \mathbf{Val} \\ \displaystyle\sum_{e \overset{p}{\rightsquigarrow} e'} p \cdot f\,(e') & \text{otherwise} \end{cases}$$

Note that if e is stuck then $\Phi(f)(e) = 0$ since the empty sum is 0.

The function Φ is monotone and preserves suprema of ω-chains. The proof is straightforward and can be found in the Appendix. Thus Φ has a least fixed point in $\mathcal{F}$ and we denote this fixed point by $\mathbf{Pr}\,(\cdot \Downarrow)$, i.e., $\mathbf{Pr}\,(e \Downarrow) = \sup_{n \in \omega} \Phi^n(\bot)(e)$.

To define the stratified observations we need the notion of a path. Given terms e and e' a path π from e to e', written $\pi : e \rightsquigarrow^* e'$, is a sequence $e \overset{p_1}{\rightsquigarrow} e_1 \overset{p_2}{\rightsquigarrow} e_2 \overset{p_3}{\rightsquigarrow} \cdots \overset{p_n}{\rightsquigarrow} e'$. The *weight* of π is the product of the weights of reductions in π. We write $\mathfrak{R}$ for the set of all paths and $\cdot$ for their concatenation (when defined). For a non-empty path $\pi \in \mathfrak{R}$ we write $\ell\,(\pi)$ for its last expression.

We call reductions of the form $\mathtt{unfold}\,(\mathtt{fold}\,v) \overset{1}{\rightsquigarrow} v$ *unfold-fold* reductions and reductions of the form $\mathtt{rand}\,\underline{n} \overset{\frac{1}{n}}{\rightsquigarrow} \underline{k}$ *choice* reductions. If *none* of the reductions in a path π is a choice reduction we call π *choice-free* and similarly if none of the reductions in π is an unfold-fold reductions we call π *unfold-fold free*.

We define the following types of multi-step reductions which we use in the definition of the logical relation.

- $e \overset{\text{cf}}{\Longrightarrow} e'$ if there is a *choice-free* path from e to e'
- $e \overset{\text{uff}}{\Longrightarrow} e'$ if there is an *unfold-fold* free path from e to e'.
- $e \overset{\text{cuff}}{\Longrightarrow} e'$ if $e \overset{\text{cf}}{\Longrightarrow} e'$ and $e \overset{\text{uff}}{\Longrightarrow} e'$.

The following useful lemma states that all but choice reductions preserve the probability of termination. As a consequence, we will see that all but choice reductions preserve equivalence.

Lemma 3.1. *Let* $e, e' \in \mathbf{Tm}$ *and* $e \overset{cf}{\Longrightarrow} e'$. *Then* $\mathbf{Pr}(e \Downarrow) = \mathbf{Pr}(e' \Downarrow)$.

The proof proceeds on the length of the reduction path with the strengthened induction hypothesis stating that the probabilities of termination of all elements on the path are the same. To define the stratified probability of termination that approximates $\mathbf{Pr}(\cdot \Downarrow)$ we need an auxiliary notion.

Definition 3.2. *For a closed expression* $e \in \mathbf{Tm}$ *we define* $\mathbf{Red}(e)$ *as the (unique) set of paths containing* exactly one *unfold-fold or choice reduction and* ending *with such a reduction. More precisely, we define the function* $\mathbf{Red} : \mathbf{Tm} \to \mathcal{P}(\mathfrak{R})$ *as the least function satisfying*

$$\mathbf{Red}(e) = \begin{cases} \{e \overset{1}{\rightsquigarrow} e'\} & \text{if } e = E[\mathtt{unfold}\,(\mathtt{fold}\,v)] \\ \{e \overset{p}{\rightsquigarrow} E[\underline{k}] \mid p = \frac{1}{n}, k \in \{1, 2, \ldots, n\}\} & \text{if } e = E[\mathtt{rand}\,\underline{n}] \\ \left\{(e \overset{1}{\rightsquigarrow} e') \cdot \pi \mid \pi \in \mathbf{Red}(e')\right\} & \text{if } e \overset{1}{\rightsquigarrow} e' \text{ and } e \overset{cuff}{\Longrightarrow} e' \\ \emptyset & \text{otherwise} \end{cases}$$

The power set $\mathcal{P}(\mathfrak{R})$ *is ordered by inclusion and the map defining* $\mathbf{Red}(\cdot)$ *is monotone, so the least fixed point exists.*

Using $\mathbf{Red}(\cdot)$ we define a monotone map $\Psi : \mathcal{F} \to \mathcal{F}$ that preserves ω-chains.

$$\Psi(f)(e) = \begin{cases} 1 & \text{if } \exists v \in \mathbf{Val}, e \overset{\text{cuff}}{\Longrightarrow} v \\ \displaystyle\sum_{\pi \in \mathbf{Red}(e)} \mathcal{W}(\pi) \cdot f(\ell(\pi)) & \text{otherwise} \end{cases}$$

and then define $\mathbf{Pr}\left(e \Downarrow^k\right) = \Psi^k(\bot)(e)$. The intended meaning of $\mathbf{Pr}\left(e \Downarrow^k\right)$ is the probability that e terminates within k unfold-fold and choice reductions. Since Ψ is monotone we have that $\mathbf{Pr}\left(e \Downarrow^k\right) \leq \mathbf{Pr}\left(e \Downarrow^{k+1}\right)$ for any k and e.

The following lemma is the reason for counting only certain reductions. It allows us to stay at the same step-index even when taking steps in the operational semantics. As a consequence we will get a more extensional logical relation. The proof is by case analysis and can be found in the Appendix.

Lemma 3.3. *Let* $e, e' \in \mathbf{Tm}$. *If* $e \overset{cuff}{\Longrightarrow} e'$ *then for all* k, $\mathbf{Pr}\left(e \Downarrow^k\right) = \mathbf{Pr}\left(e' \Downarrow^k\right)$.

The following is immediate from the definition of the chain $\left\{\mathbf{Pr}\left(e \Downarrow^k\right)\right\}_{k=0}^{\infty}$ and the fact that $\mathtt{rand}\,\underline{n}$ reduces with uniform probability.

Lemma 3.4. *Let e be a closed term. If $e \overset{1}{\rightsquigarrow} e'$ and the reduction is an unfold-fold reduction then $\mathbf{Pr}\left(e \Downarrow^{k+1}\right) = \mathbf{Pr}\left(e' \Downarrow^k\right)$. If the reduction from e is a choice reduction, then $\mathbf{Pr}\left(e \Downarrow^{k+1}\right) = \frac{1}{|\mathbf{Red}(e)|}\sum_{e' \in \mathbf{Red}(e)} \mathbf{Pr}\left(e' \Downarrow^k\right)$.*

The following proposition is needed to prove adequacy of the logical relation with respect to contextual equivalence. It is analogous to the property used to prove adequacy of step-indexed logical relations for deterministic and nondeterministic languages. Consider the case of may-equivalence. To prove adequacy in this case (cf. [3, Theorem 4.8]) we use the fact that if e may-terminates, then there is a natural number n such that e terminates in n steps. This property does not hold in the probabilistic case, but the property analogous to it still holds and is sufficient to prove adequacy.

Proposition 3.5. *For each $e \in \mathbf{Tm}$ we have $\mathbf{Pr}\left(e \Downarrow\right) \leq \sup_{k\in\omega}\left(\mathbf{Pr}\left(e \Downarrow^k\right)\right)$.*

Proof. We only give a sketch; the full proof can be found in the Appendix. We use Scott induction on the set $\mathcal{S} = \left\{f \in \mathcal{F} \;\middle|\; \forall e, f(e) \leq \sup_{k\in\omega}\left(\mathbf{Pr}\left(e \Downarrow^k\right)\right)\right\}$. It is easy to see that $\mathcal{S}$ is closed under limits of ω-chains and that $\bot \in \mathcal{S}$ so we only need to show that $\mathcal{S}$ is closed under Φ. We can do this by considering the kinds of reductions from e when considering $\Phi(f)(e)$ for $f \in \mathcal{S}$.

4 Logical, CIU and contextual approximation relations

The contextual and CIU approximations are defined in a way analogous to the one for deterministic programming languages. We require some auxiliary notions. A *type-indexed relation* $\mathcal{R}$ is a set of tuples $(\Delta, \Gamma, e, e', \tau)$ such that $\Delta \vdash \Gamma$ and $\Delta \vdash \tau$ and $\Delta \mid \Gamma \vdash e : \tau$ and $\Delta \mid \Gamma \vdash e' : \tau$. We write $\Delta \mid \Gamma \vdash e \;\mathcal{R}\; e' : \tau$ for $(\Delta, \Gamma, e, e', \tau) \in \mathcal{R}$.

Definition 4.1 (Precongruence). *A type-indexed relation $\mathcal{R}$ is* reflexive *if $\Delta \mid \Gamma \vdash e : \tau$ implies $\Delta \mid \Gamma \vdash e \;\mathcal{R}\; e : \tau$. It is* transitive *if $\Delta \mid \Gamma \vdash e \;\mathcal{R}\; e' : \tau$ and $\Delta \mid \Gamma \vdash e' \;\mathcal{R}\; e'' : \tau$ implies $\Delta \mid \Gamma \vdash e \;\mathcal{R}\; e'' : \tau$. It is* compatible *if it is closed under the term forming rules, e.g.,*[2]

$$\frac{\Delta \mid \Gamma, x{:}\tau_1 \vdash e \;\mathcal{R}\; e' : \tau_2}{\Delta \mid \Gamma \vdash \lambda x.e \;\mathcal{R}\; \lambda x.e' : \tau_1 \to \tau_2} \qquad \frac{\Delta \mid \Gamma \vdash e \;\mathcal{R}\; e' : \mathtt{nat}}{\Delta \mid \Gamma \vdash \mathtt{rand}\, e \;\mathcal{R}\; \mathtt{rand}\, e' : \mathtt{nat}}$$

A precongruence *is a reflexive, transitive and compatible type-indexed relation.*

The compatibility rules guarantee that a compatible relation is sufficiently big, i.e., at least reflexive. In contrast, the notion of adequacy, which relates the operational semantics with the relation, guarantees that it is not too big. In the deterministic case, a relation $\mathcal{R}$ is adequate if when $e \;\mathcal{R}\; e'$ are two related closed terms, then if e terminates so does e'. Here we need to compare probabilities of termination instead, since these are our observations.

[2] We only show a few rules, the rest are analogous and can be found in the Appendix.

Definition 4.2. *A type-indexed relation $\mathcal{R}$ is* adequate *if for all e, e' such that $\varnothing \mid \varnothing \vdash e \mathcal{R} e' : \tau$ we have $\mathbf{Pr}(e \Downarrow) \leq \mathbf{Pr}(e' \Downarrow)$.*

The *contextual approximation relation*, written $\Delta \mid \Gamma \vdash e \lesssim^{ctx} e' : \tau$, is defined as the *largest adequate precongruence* and the *CIU approximation relation*, written $\Delta \mid \Gamma \vdash e \lesssim^{CIU} e' : \tau$, is defined using evaluation contexts in the usual way, e.g. [14], using $\mathbf{Pr}(\cdot \Downarrow)$ for observations. The fact that the largest adequate precongruence exists is proved as in [14].

Logical relation We now define the step-indexed logical relation. We present the construction in the elementary way with explicit indexing instead of using a logic with guarded recursion as in [7] to remain self-contained.

Interpretations of types will be defined as decreasing sequences of relations on *typeable* values. For *closed types* τ and σ we define the sets $\mathbf{VRel}(\tau, \sigma)$, $\mathbf{SRel}(\tau, \sigma)$ and $\mathbf{TRel}(\tau, \sigma)$ to be the sets of decreasing sequences of relations on typeable values, evaluation contexts and expressions respectively. The types τ and σ denote the types of the left-hand side and the right-hand side respectively, i.e. if $(v, u) \in \varphi(n)$ for $\varphi \in \mathbf{VRel}(\tau, \sigma)$ then v has type τ and u has type σ. The order relation $\leq$ on these sets is defined pointwise, e.g. for $\varphi, \psi \in \mathbf{VRel}(\tau, \sigma)$ we write $\varphi \leq \psi$ if $\forall n \in \omega^{op}, \varphi(n) \subseteq \psi(n)$. We implicitly use the inclusion from $\mathbf{VRel}(\tau, \sigma)$ to $\mathbf{TRel}(\tau, \sigma)$. The reason for having relations on values and terms of different types on the left and right-hand sides is so we are able to prove parametricity properties in Section 5.

We define maps $\cdot^{\top}_{\tau,\sigma} : \mathbf{VRel}(\tau, \sigma) \to \mathbf{SRel}(\tau, \sigma)$ and $\cdot^{\bot}_{\tau,\sigma} : \mathbf{SRel}(\tau, \sigma) \to \mathbf{TRel}(\tau, \sigma)$. We usually omit the type indices when they can be inferred from the context. The maps are defined as follows

$$r^{\top}_{\tau,\sigma}(n) = \left\{(E, E') \;\middle|\; \forall k \leq n, \forall (v, v') \in r(k), \mathbf{Pr}\left(E[v] \Downarrow^k\right) \leq \mathbf{Pr}\left(E'[v'] \Downarrow\right)\right\}$$

and $r^{\bot}_{\tau,\sigma}(n) = \left\{(e, e') \;\middle|\; \forall k \leq n, \forall (E, E') \in r(k), \mathbf{Pr}\left(E[e] \Downarrow^k\right) \leq \mathbf{Pr}\left(E'[e'] \Downarrow\right)\right\}$. We write $r^{\top\top} = r^{\top\bot}$ for their composition from $\mathbf{VRel}(\tau, \sigma)$ to $\mathbf{TRel}(\tau, \sigma)$.

The following lemma establishes basic properties of the functions $\cdot^{\top}$ and $\cdot^{\top\top}$. The function $\cdot^{\top}$ is order-reversing and $\cdot^{\top\top}$ is order-preserving and inflationary.

Lemma 4.3. *Let τ, σ be closed types and $r, s \in \mathbf{VRel}(\tau, \sigma)$. Then $r \leq r^{\top\top}$ and if $r \leq s$ then $s^{\top} \leq r^{\top}$ and $r^{\top\top} \leq s^{\top\top}$.*

For a type-variable context Δ we define $\mathbf{VRel}(\Delta)$ using $\mathbf{VRel}(\cdot, \cdot)$

$$\mathbf{VRel}(\Delta) = \left\{(\varphi_1, \varphi_2, \varphi_r) \;\middle|\; \varphi_1, \varphi_2 \in \mathfrak{T}^{\Delta}, \forall \alpha \in \Delta, \varphi_r(\alpha) \in \mathbf{VRel}(\varphi_1(\alpha), \varphi_2(\alpha))\right\}$$

where the first two components give syntactic types for the left and right hand sides of the relation and the third component is a relation between those types.

The interpretation of types, $[\![\cdot \vdash \cdot]\!]$ is by induction on the judgement $\Delta \vdash \tau$. Given a judgment $\Delta \vdash \tau$ and $\varphi \in \mathbf{VRel}(\Delta)$ we have $[\![\Delta \vdash \tau]\!](\varphi) \in \mathbf{VRel}(\varphi_1(\tau), \varphi_2(\tau))$ where the φ_1 and φ_2 are the first two components of φ and $\varphi_1(\tau)$ denotes substitution of types in φ_1 for free type variables in τ. Moreover $[\![\cdot]\!]$ is *non-expansive*

in the sense that $[\![\Delta \vdash \tau]\!](\varphi)(n)$ can depend only on the values of $\varphi_r(\alpha)(k)$ for $k \leq n$. The interpretation of types is defined in Figure 1. Note that the value relations are as simple as for a language without probabilistic choice. The crucial difference is hidden in the $\top\top$-closure of value relations.

$$
\begin{aligned}
[\![\Delta \vdash \mathtt{nat}]\!](\varphi)(n) &= \{(\underline{k}, \underline{k}) \mid k \in \mathbb{N}, k > 0\} \\
[\![\Delta \vdash \tau \to \sigma]\!](\varphi)(n) &= \{(\lambda x.e, \lambda y.e') \mid \forall j \leq n, \forall (v, v') \in [\![\Delta \vdash \tau]\!](\varphi)(j), \\
&\qquad ((\lambda x.e)\, v, (\lambda y.e')\, v') \in [\![\Delta \vdash \sigma]\!](\varphi)^{\top\top}(j)\} \\
[\![\Delta \vdash \forall \alpha.\tau]\!](\varphi)(n) &= \{(\Lambda.e, \Lambda.e') \mid \forall \sigma, \sigma' \in \mathfrak{T}, \forall r \in \mathbf{VRel}(\sigma, \sigma'), \\
&\qquad (e, e') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi[\alpha \mapsto r])^{\top\top}(n)\} \\
[\![\Delta \vdash \exists \alpha.\tau]\!](\varphi)(n) &= \{(\mathtt{pack}\, v, \mathtt{pack}\, v') \mid \exists \sigma, \sigma' \in \mathfrak{T}, \exists r \in \mathbf{VRel}(\sigma, \sigma'), \\
&\qquad (v, v') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi[\alpha \mapsto r])(n)\} \\
[\![\Delta \vdash \mu \alpha.\tau]\!](\varphi)(0) &= \mathbf{Val}(\varphi_1(\mu\alpha.\tau)) \times \mathbf{Val}(\varphi_2(\mu\alpha.\tau)) \\
[\![\Delta \vdash \mu \alpha.\tau]\!](\varphi)(n+1) &= \{(\mathtt{fold}\, v, \mathtt{fold}\, v') \mid \\
&\qquad (v, v') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi[\alpha \mapsto [\![\Delta \vdash \mu\alpha.\tau]\!](\varphi)])(n)\}
\end{aligned}
$$

Fig. 1. Interpretation of types. The cases for sum and product types are in Appendix.

Context extension lemmas To prove soundness and completeness we need lemmas stating how extending evaluation contexts preserves relatedness. We only show the case for `rand`. The rest are similarly simple.

Lemma 4.4. *Let $n \in \mathbb{N}$. If $(E, E') \in [\![\Delta \vdash \mathtt{nat}]\!](\varphi)^{\top}(n)$ are related evaluation contexts then $(E \circ (\mathtt{rand}\, []), E' \circ (\mathtt{rand}\, [])) \in [\![\Delta \vdash \mathtt{nat}]\!](\varphi)^{\top}(n)$.*

Proof. Let $n \in \mathbb{N}$ and $(v, v') \in [\![\Delta \vdash \tau]\!](\varphi)(n)$. By construction we have $v = v' = \underline{m}$ for some $m \in \mathbb{N}$, $m \geq 1$. Let $k \leq n$. If $k = 0$ the result is immediate, so assume $k = \ell + 1$. Using Lemma 4 we have $\mathbf{Pr}\left(E[\mathtt{rand}\, \underline{m}] \Downarrow^k\right) = \frac{1}{m} \sum_{i=1}^{m} \mathbf{Pr}\left(E[\underline{i}] \Downarrow^\ell\right)$ and using the assumption $(E, E') \in [\![\Delta \vdash \mathtt{nat}]\!](\varphi)^{\top}(n)$, the fact that $k \leq n$ and monotonicity in the step-index the latter term is less than $\frac{1}{m} \sum_{i=1}^{m} \mathbf{Pr}\left(E'[\underline{i}] \Downarrow\right)$ which by definition of $\mathbf{Pr}(\cdot \Downarrow)$ is equal to $\mathbf{Pr}\left(E'[\mathtt{rand}\, \underline{m}] \Downarrow\right)$.

We define the logical approximation relation for open terms given the interpretations of types in Figure 1. We define $\Delta \mid \Gamma \vdash e \lesssim^{log} e' : \tau$ to mean

$$\forall n \in \mathbb{N}, \forall \varphi \in \mathbf{VRel}(\Delta), \forall (\gamma, \gamma') \in [\![\Delta \vdash \Gamma]\!](\varphi)(n), (e\gamma, e'\gamma) \in [\![\Delta \vdash \tau]\!]\, \varphi^{\top\top}(n)$$

Here $[\![\Delta \vdash \Gamma]\!]$ is the obvious extension of interpretation of types to interpretation of contexts which relates substitutions, mapping variables to values. We then have the following fundamental property of the logical relation.

Proposition 4.5. *The logical approximation relation $\lesssim^{log}$ is compatible. In particular it is reflexive.*

Proof. The proof is a simple consequence of the context extension lemmas. We show the case for `rand`. We have to show that $\Delta \mid \Gamma \vdash e \lesssim^{log} e' : \mathtt{nat}$ implies $\Delta \mid \Gamma \vdash \mathtt{rand}\, e \lesssim^{log} \mathtt{rand}\, e' : \mathtt{nat}$. Let $n \in \mathbb{N}$, $\varphi \in \mathbf{VRel}(\Delta)$ and $(\gamma, \gamma') \in$

$[\![\Delta \vdash \Gamma]\!](\varphi)(n)$. Let $f = e\gamma$ and $f' = e'\gamma'$. Then our assumption gives us $(f, f') \in [\![\Delta \vdash \mathtt{nat}]\!](\varphi)^{\top\top}(n)$ and we are to show $(\mathtt{rand}\ f, \mathtt{rand}\ f') \in [\![\Delta \vdash \mathtt{nat}]\!](\varphi)^{\top\top}(n)$. Let $j \leq n$ and $(E, E') \in [\![\Delta \vdash \mathtt{nat}]\!](\varphi)^{\top}(j)$. Then from Lemma 6 we have $(E \circ (\mathtt{rand}\ []), E' \circ (\mathtt{rand}\ [])) \in [\![\Delta \vdash \mathtt{nat}]\!](\varphi)^{\top}(n)$ which suffices by the fact that $(f, f') \in [\![\Delta \vdash \mathtt{nat}]\!](\varphi)^{\top\top}(n)$.

We now want to relate logical, CIU and contextual approximation relations. We start by showing adequacy of the logical relation.

Corollary 4.6. *Logical approximation relation $\lesssim^{log}$ is adequate.*

Proof. Assume $\varnothing \mid \varnothing \vdash e \lesssim^{log} e' : \tau$. We are to show that $\mathbf{Pr}(e \Downarrow) \leq \mathbf{Pr}(e' \Downarrow)$. Straight from the definition we have $\forall n \in \mathbb{N}, (e, e') \in [\![\varnothing \vdash \tau]\!]^{\top\top}(n)$. The empty evaluation context is always related to itself (at any type). This implies $\forall n \in \mathbb{N}, \mathbf{Pr}(e \Downarrow^n) \leq \mathbf{Pr}(e' \Downarrow)$ which further implies (since the right-hand side is independent of n) that $\sup_{n\in\omega}(\mathbf{Pr}(e \Downarrow^n)) \leq \mathbf{Pr}(e' \Downarrow)$. Using Proposition 1 we thus have $\mathbf{Pr}(e \Downarrow) \leq \sup_{n\in\omega}(\mathbf{Pr}(e \Downarrow^n)) \leq \mathbf{Pr}(e' \Downarrow)$ concluding the proof.

As a simple corollary, we have that for well-typed expressions, the supremum of stratified observations is equal to the actual observation.

Corollary 4.7. *If $e \in \mathbf{Tm}$ is well-typed then $\sup_{n\in\omega}(\mathbf{Pr}(e \Downarrow^n)) = \mathbf{Pr}(e \Downarrow)$.*

We now have that the logical relation is adequate and compatible. This does not immediately imply that it is contained in the contextual approximation relation, since we do not know that it is transitive. However we have the following lemma where by transitive closure we mean that for each Δ, Γ and τ we take the transitive closure of the relation $\{(e, e') \mid \Delta \mid \Gamma \vdash e \lesssim^{log} e' : \tau\}$. This is another type-indexed relation.

Lemma 4.8. *The transitive closure of $\lesssim^{log}$ is compatible and adequate.*

Proof. Transitive closure of an adequate relation is adequate. Similarly the transitive closure of a compatible and *reflexive* relation (in the sense of Definition 3) is again compatible (and reflexive).

And we arrive at the the main theorem.

Theorem 4.9 (CIU theorem). *The relations $\lesssim^{log}$, $\lesssim^{CIU}$ and $\lesssim^{ctx}$ coincide.*

Proof. It is standard (e.g. [14]) that $\lesssim^{ctx}$ is included in $\lesssim^{CIU}$. We show that the logical approximation relation is contained in the CIU approximation relation in the standard way for biorthogonal step-indexed logical relations. To see that $\lesssim^{log}$ is included in $\lesssim^{ctx}$ we have by Lemma 7 that the transitive closure of $\lesssim^{log}$ is an adequate precongruence, thus included in $\lesssim^{ctx}$. And $\lesssim^{log}$ is included in the transitive closure of $\lesssim^{log}$. We have thus completed the cycle of inclusions.

Using the logical relation and Theorem 1 we can prove the following extenstionality properties. The proofs are standard and can be found in the Appendix.

Lemma 4.10 (Functional extensionality for values). *Suppose* $\tau, \sigma \in \mathfrak{T}(\Delta)$ *and let* f *and* f' *be two* values *of type* $\tau \to \sigma$ *in context* $\Delta \mid \Gamma$*. If for all* $u \in \mathbf{Val}(\tau)$ *we have* $\Delta \mid \Gamma \vdash f\, u \lesssim^{ctx} f'\, u : \sigma$ *then* $\Delta \mid \Gamma \vdash f \lesssim^{ctx} f' : \tau \to \sigma$*.*

The extensionality for *expressions*, as opposed to only *values*, of function type does not hold in general due to the presence of choice reductions. See Remark **??** for an example. We also have extensionality for values of the universal type.

Lemma 4.11 (Extensionality for the universal type). *Let* $\tau \in \mathfrak{T}(\Delta, \alpha)$ *be a type. Let* f, f' *be two* values *of type* $\forall\alpha.\tau$ *in context* $\Delta \mid \Gamma$*. If for all closed types* σ *we have* $\Delta \mid \Gamma \vdash f[] \lesssim^{ctx} f'[] : \tau[\sigma/\alpha]$ *then* $\Delta \mid \Gamma \vdash f \lesssim^{ctx} f' : \forall\alpha.\tau$*.*

5 Examples

We now use our logical relation to prove some example equivalences. We show two examples involving polymorphism. In the Appendix we show additional examples. In particular we show the correctness of von Neumann's procedure for generating a fair sequence of coin tosses from an unfair coin. That example in particular shows how the use of biorthogonality allows us to "externalize" the reasoning to arithmetic manipulations.

We first define $\mathtt{fix} : \forall\alpha,\beta.((\alpha\to\beta)\to(\alpha\to\beta)) \to (\alpha\to\beta)$ be the term $\Lambda.\Lambda.\lambda f.\lambda z.\delta_f(\mathtt{fold}\,\delta_f)\,z$ where δ_f is the term $\lambda y.\mathtt{let}\ y' = \mathtt{unfold}\, y\ \mathtt{in}\ f\,(\lambda x.y'\,y\,x)$. This is a call-by-value fixed-point combinator. We also write $e_1 \oplus e_2$ for the term $\mathtt{if}_1\ \mathtt{rand}\,\underline{2}\ \mathtt{then}\ e_1\ \mathtt{else}\ e_2$. Note that the choice is made before evaluating e_i's.

We characterize inhabitants of a polymorphic type and show a free theorem. For the former, we need to know which real numbers can be probabilities of termination of programs. Recall that a real number r is *left-computable* if there exists a *computable* increasing (not necessarily strictly) sequence $\{q_n\}_{n\in\omega}$ of rational numbers such that $r = \sup_{n\in\omega} q_n$. In Appendix **??** we prove.

Proposition 5.1. *For any expression* e, $\mathbf{Pr}(e \Downarrow)$ *is a left-computable real number and for any left-computable real number* r *in the interval* $[0,1]$ *there is a closed term* e_r *of type* $\mathbf{1}\to\mathbf{1}$ *such that* $\mathbf{Pr}(e_r\,\langle\rangle \Downarrow) = r$*.*

Inhabitants of the type $\forall\alpha.\alpha\to\alpha$ In this section we use further syntactic sugar for sequencing. When $e, e' \in \mathbf{Tm}$ are closed terms we write $e; e'$ for $(\lambda\_.e')\,e$, i.e. first run e, ignore the result and then run e'. We will need the property that for all terms $e, e' \in \mathbf{Tm}$, $\mathbf{Pr}(e; e' \Downarrow) = \mathbf{Pr}(e \Downarrow)\cdot\mathbf{Pr}(e' \Downarrow)$. The proof is by Scott induction and can be found in the Appendix.

Using Proposition 3 we have for each left-computable real r in the interval $[0,1]$ an inhabitant t_r of the type $\forall\alpha.\alpha\to\alpha$ given by $\Lambda.\lambda x.e_r\,\langle\rangle; x$.

We now show that these are the only inhabitants of $\forall\alpha.\alpha\to\alpha$ of the form $\Lambda.\lambda x.e$. Given such an inhabitant let $r = \mathbf{Pr}(e[\langle\rangle/x] \Downarrow)$. We know from Proposition 3 that r is left-computable.

Given a value v of type τ and $n \in \mathbb{N}$ we define relations $R(n) = \{(\langle\rangle, v)\}$ and $S(n) = \{(v, \langle\rangle)\}$. Note that the relations are independent of n, i.e. R and

S are constant relations. By reflexivity of the logical relation and the relational actions of types we have

$$\forall n, (e[\langle\rangle/x], e[v/x]) \in R^{\top\top}(n) \qquad \text{and} \qquad \forall n, (e[v/x], e[\langle\rangle/x]) \in S^{\top\top}(n) \quad (1)$$

from which we conclude that $\mathbf{Pr}\,(e[\langle\rangle/x] \Downarrow) = \mathbf{Pr}\,(e[v/x] \Downarrow)$. We now show that v and $e[v/x]$ are CIU-equivalent. Let $E \in \mathbf{Stk}\,(\tau)$ be an evaluation context. Let $q = \mathbf{Pr}\,(E[v] \Downarrow)$. Define the evaluation context $E' = -; e_q\,\langle\rangle$. Then $(E, E') \in S^{\top}(n)$ for all n which then means, using (1) and Proposition 1, that $\mathbf{Pr}\,(E[e[v/x]] \Downarrow) \leq \mathbf{Pr}\,(E'[e[\langle\rangle/x]] \Downarrow)$. We then have

$$\mathbf{Pr}\,(E'[e[\langle\rangle/x]] \Downarrow) = \mathbf{Pr}\,(e[\langle\rangle/x] \Downarrow) \cdot \mathbf{Pr}\,(e_q\,\langle\rangle \Downarrow) = r \cdot \mathbf{Pr}\,(E[v] \Downarrow)$$

and so $\mathbf{Pr}\,(E[e[v/x]] \Downarrow) \leq r \cdot \mathbf{Pr}\,(E[v] \Downarrow)$.

Similarly we have $(E', E) \in R^{\top}(n)$ for all n which implies $\mathbf{Pr}\,(E[e[v/x]] \Downarrow) \geq \mathbf{Pr}\,(E'[e[\langle\rangle/x]] \Downarrow)$. We also have $\mathbf{Pr}\,(E'[e[\langle\rangle/x]] \Downarrow) = r \cdot \mathbf{Pr}\,(E[v] \Downarrow)$.

So we have proved $\mathbf{Pr}\,(E[e[v/x]] \Downarrow) = r \cdot \mathbf{Pr}\,(E[v] \Downarrow) = \mathbf{Pr}\,(e[v/x] \Downarrow) \cdot \mathbf{Pr}\,(E[v] \Downarrow)$. It is easy to show by Scott induction, that $\mathbf{Pr}\,(E[t_r[]\,v] \Downarrow) = \mathbf{Pr}\,(e_r\,\langle\rangle \Downarrow) \cdot \mathbf{Pr}\,(E[v] \Downarrow)$. We have thus shown that for any value v, the terms $e[v/x]$ and $\mathbf{Pr}\,(t_r[]\,v \Downarrow)$ are CIU-equivalent. Using Theorem 1 and Lemmas 9 and 8 we can thus conclude that the terms $\forall\alpha.\lambda x.e$ and t_r are contextually equivalent.

Remark 5.2. Unfortunately we cannot so easily characterize general values of the type $\forall\alpha.\alpha \to \alpha$, that is, those not of the form $\Lambda.v$ for a value v. Consider the term $\Lambda.t_{\frac{1}{2}} \oplus t_{\frac{1}{3}}$. It is a straightforward calculation that for any evaluation context E and value v, $\mathbf{Pr}\left(E\left[\left(t_{\frac{1}{2}} \oplus t_{\frac{1}{3}}\right)\,v\right] \Downarrow\right) = \frac{5}{12}\mathbf{Pr}\,(E[v] \Downarrow) = \mathbf{Pr}\left(E\left[t_{\frac{5}{12}}\,v\right] \Downarrow\right)$ thus if $\Lambda.t_{\frac{1}{2}} \oplus t_{\frac{1}{3}}$ is equivalent to any $\Lambda.t_r$ it must be $\Lambda.t_{\frac{5}{12}}$.

Let E be the evaluation context $E = \texttt{let}\ f\ = -[]\ \texttt{in let}\ x\ = f\,\langle\rangle\ \texttt{in}\ f\,\langle\rangle$. We compute $\mathbf{Pr}\left(E\left[\Lambda.t_{\frac{1}{2}} \oplus t_{\frac{1}{3}}\right] \Downarrow\right) = \frac{13}{72}$ and $\mathbf{Pr}\left(E\left[\Lambda.t_{\frac{5}{12}}\right] \Downarrow\right) = \frac{25}{144}$ showing that $\Lambda.t_{\frac{1}{2}} \oplus t_{\frac{1}{3}}$ is *not* equivalent to $\Lambda.t_{\frac{5}{12}}$.

This example also shows that extensionality for *expressions*, as opposed to *values*, of function type does not hold.

A free theorem We now show a free theorem for functions on lists. Let τ be a type and α not free in τ. We write $[\tau]$ for the type of lists $\mu\alpha.(\mathbf{1} + \tau \times \alpha)$, $\texttt{nil}$ for the empty list and $\texttt{cons} : \forall\alpha.\alpha \to [\alpha] \to [\alpha]$ for the other constructor $\texttt{cons} = \Lambda.\lambda x.\lambda xs.\texttt{fold}\,(\texttt{inr}\,\langle x, xs\rangle)$. The function $\texttt{map}$ of type $\forall\alpha.\forall\beta.(\alpha \to \beta) \to [\alpha] \to [\beta]$ is the function applying the given function to all elements of the list in order.

Additionally, we define composition of terms $f \circ g$ as the term $\lambda x.f(g(x))$ (for x not free in f and g).

We will now show that any term m of type $\forall\alpha.\forall\beta.(\alpha \to \beta) \to [\alpha] \to [\beta]$ that is equivalent to a term of the form $\Lambda.\Lambda.\lambda x.e$ satisfies $m[][]\,(f \circ g) =^{ctx} m[][]f \circ \texttt{map}[][]\,g$ for all *values* f and all *deterministic and terminating* g. By this we mean that for each value v in the domain of g, there exists a *value* u in the codomain of g, such

that $g\,v =^{ctx} u$. For instance, if g reduces without using choice reductions and is terminating, then g is deterministic. There are other functions that are also deterministic and terminating, though, for instance $\lambda x.\langle\rangle \oplus \langle\rangle$. In the Appendix we show that these restrictions are not superfluous.

So let m be a closed term of type $\forall\alpha.\forall\beta.(\alpha \to \beta) \to [\alpha] \to [\beta]$ and suppose further that m is equivalent to a term of the form $\Lambda.\Lambda.\lambda x.e$. Let $\tau, \sigma, \rho \in \mathfrak{T}$ be closed types and $f \in \mathbf{Val}\,(\sigma \to \rho)$ and $g \in \mathbf{Tm}\,(\tau \to \sigma)$ be a deterministic and terminating function. Then

$$\varnothing \mid \varnothing \vdash m[][](f \circ g) =^{ctx} m[][]f \circ \mathtt{map}[][]g : [\tau] \to [\rho].$$

We prove two approximations separately, starting with $\lesssim^{ctx}$. We use Theorem 1 multiple times. We have $\alpha, \beta \mid \varnothing \vdash m[][] : (\alpha \to \beta) \to [\alpha] \to [\beta]$. Let $R = \lambda n.\{(v,u) \mid g\,v =^{ctx} u\}$ be a member of $\mathbf{VRel}\,(\tau,\sigma)$ and $S \in \mathbf{VRel}\,(\rho,\rho)$ be the constant identity relation on $\mathbf{Val}\,(\rho)$. Let φ be a function mapping α to R and β to S. From Proposition 2 we have $(m[][], m[][]) \in [\![(\alpha \to \beta) \to [\alpha] \to [\beta]]\!]\,(\varphi)^{\top\top}(n)$ for all $n \in \mathbb{N}$.

We first claim that $(f \circ g, f) \in [\![\alpha \to \beta]\!]\,(\varphi)(n)$ for all $n \in \mathbb{N}$. Since f is a value and has a type, it must be of the form $\lambda x.e$ for some x and e. Take $j \in \mathbb{N}$, related values $(v,u) \in r(j)$, $k \le j$ and $(E,E') \in S^{\top}(k)$ two related evaluation contexts. We then have $\mathbf{Pr}\,(E'[f\,u] \Downarrow) = \mathbf{Pr}\,(E'[f(g\,v)] \Downarrow)$ by Theorem 1 and the definition of relation R. Using the results about $\mathbf{Pr}\left(\cdot \Downarrow^k\right)$ and $\mathbf{Pr}\,(\cdot \Downarrow)$ proved in Section **??** in the Appendix this gives us

$$\mathbf{Pr}\left(E[f(g(v))] \Downarrow^k\right) \le \sum_{\pi : f(g(v)) \rightsquigarrow^* w} \mathcal{W}\,(\pi)\,\mathbf{Pr}\left(E[w] \Downarrow^k\right) \le \sum_{\pi : f(g(v)) \rightsquigarrow^* w} \mathcal{W}\,(\pi)\,\mathbf{Pr}\,(E'[w] \Downarrow)$$

and the last term is equal to $\mathbf{Pr}\,(E'[f(g\,v)] \Downarrow)$ which is equal to $\mathbf{Pr}\,(E'[f\,u] \Downarrow)$.

From this we can conclude $(m[][]\,(f \circ g), m[][]\,f) \in [\![[\alpha] \to [\beta]]\!]\,(\varphi)^{\top\top}(n)$ for all $n \in \mathbb{N}$. Note that we have *not yet* used the fact that g is deterministic and terminating. We do so now.

Let xs be a list of elements of type τ. Then it is easy to derive, by induction on the length of the list, using the assumption on g, that there exists a list ys of elements of type σ, such that $\mathtt{map}[][]\,g\,xs =^{ctx} ys$ and it is similarly easy to show that this list ys satisfies $(xs, ys) \in [\![[\alpha]]\!]\,(\varphi)(n)$ for all n.

This gives us $(m[][]\,(f \circ g)\,xs, m[][]\,f\,ys) \in [\![[\beta]]\!]\,(\varphi)^{\top\top}(n)$ for all $n \in \mathbb{N}$. Since the relation S is the identity relation, it is easy to see that for all evaluation contexts E of a suitable type, $(E,E) \in S^{\top}(n)$ for all n, which gives

$$m[][]\,(f \circ g)\,xs \lesssim^{CIU} m[][]\,f\,ys =^{ctx} m[][]\,f\,(\mathtt{map}[][]\,g\,xs) =^{ctx} (m[][]\,f \circ \mathtt{map}[][]\,g)\,xs$$

where the last equality is an instance of the fact that all except the choice reductions preserve equivalence, which is very easy to see by showing that CIU-equivalence is preserved by such reductions and then using Theorem 1.

We now conclude by using the fact that m is (equivalent to) a term of the form $\Lambda.\Lambda.\lambda x.e$ and use Lemma 8 to conclude $m[][]\,(f \circ g) \lesssim^{ctx} m[][]\,f \circ \mathtt{map}[][]\,g$.

For the other direction, we proceed analogously. The relation for β remains the identity relation, and the relation for R for α is $\{(v,u) \mid v =^{ctx} g\,u\}$.

6 Extension to references

We now sketch the extension of $\mathbf{F}^{\mu,\oplus}$ to include dynamically allocated references. For simplicity we add ground store only, so we do not have to solve a domain equation giving us the space of semantic types and worlds. We show an equivalence using state and probabilistic choice which shows that the addition of references to the language is orthogonal to the addition of probabilistic choice and, in particular, that the extension with *higher-order* dynamically allocated references can be done as in earlier work on step-indexed logical relations.

We extend the language by adding the type $\texttt{ref}\,\texttt{nat}$ and extend the grammar of terms with $\ell \mid \texttt{ref}\, e \mid e_1 := e_2 \mid !e$ with ℓ being locations.

To model allocation we need to index the interpretation of types by worlds. To keep things simple a world $w \in \mathcal{W}$ is partial bijection f on locations together with, for each pair of locations $(\ell_1, \ell_2) \in f$, a relation R on numerals. We write $(\ell_1, \ell_2, R) \in w$ when the partial bijection in w relates ℓ_1 and ℓ_2 and R is the relation assigned to the pair (ℓ_1, ℓ_2). Technically, worlds are relations of type $\texttt{Loc}^2 \times \mathcal{P}\left(\{\underline{n} \mid n \in \mathbb{N}\}\right)$ satisfying the conditions described above.

The operational semantics has to be extended to include heaps, which are modeled as finite maps from locations to numerals. Given a world, we define a set of heaps that satisfy the world as

$$(h_1, h_2) \in \lfloor w \rfloor \leftrightarrow \forall (\ell_1, \ell_2, R) \in w, (h_1(\ell_1), h_2(\ell_2)) \in R$$

The interpretation of types is then extended to include worlds. The denotation of a type is now an element of $\mathcal{W} \overset{mon}{\to} \mathbf{VRel}\,(\cdot, \cdot)$ where the order on $\mathcal{W}$ is inclusion. Let $\mathbf{WRel}\,(\tau, \tau') = \mathcal{W} \overset{mon}{\to} \mathbf{VRel}\,(\tau, \tau')$. The interpretation of the reference type is $[\![\Delta \vdash \texttt{ref}\,\texttt{nat}]\!]\,(\varphi)(n) = \lambda w. \left\{(\ell_1, \ell_2) \;\middle|\; (\ell_1, \ell_2, =) \in w\right\}$ where $=$ is the equality relation on numerals.

The rest of the interpretation stays the same, apart from some quantification over "future worlds" in the function case to maintain monotonicity. We also need to change the definition of the $\top\top$-closure to use the world satisfaction relation. For $r \in \mathbf{WRel}\,(\tau, \tau')$ we define an indexed relation (indexed by worlds) $r^{\top}$. The relation $r^{\top}$ at world w and step-index n, written $r^{\top}(w)(n)$, is

$$\left\{(E, E') \;\middle|\; \begin{array}{c} \forall w' \geq w, \forall k \leq n, \forall (h_1, h_2) \in \lfloor w' \rfloor, \forall v_1, v_2 \in r(w')(k), \\ \mathbf{Pr}\left(\langle h_1, E[v_1]\rangle \Downarrow^k\right) \leq \mathbf{Pr}\left(\langle h_2, E[v_2]\rangle \Downarrow\right) \end{array}\right\}$$

and analogously for $\cdot^{\perp}$.

We now sketch a proof that two modules, each implementing a counter by using a single internal location, are contextually equivalent. The increment method is special. When called, it chooses, uniformly, whether to increment the counter or not. The two modules differ in the way they increment the counter. One module increments the counter by 1, the other by 2. Concretely, we show that the two counters $\texttt{pack}\ (\lambda - .\texttt{ref}\,\underline{1}, \lambda x.!x, \lambda x.\langle\rangle \oplus (x := \texttt{S}\,!x))$ and $\texttt{pack}\ (\lambda - .\texttt{ref}\,\underline{2}, \lambda x.!x\ \texttt{div}\ \underline{2}, \lambda x.\langle\rangle \oplus (x := \texttt{S}\,(\texttt{S}\,!x)))$ are contextually equivalent at type $\exists\alpha.(\mathbf{1} \to \alpha) \times (\alpha \to \texttt{nat}) \times (\alpha \to \mathbf{1})$. We have used $\texttt{div}$ for the division function on numerals which can easily be implemented as a deterministic terminating function.

The interpretation of existentials $[\![\Delta \vdash \exists\alpha.\tau]\!](\varphi)(n)$ now maps the world w to

$$\left\{(\mathtt{pack}\, v, \mathtt{pack}\, v') \;\middle|\; \begin{array}{l}\exists\sigma,\sigma' \in \mathfrak{T}, \exists r \in \mathbf{WRel}(\sigma,\sigma'), \\ (v,v') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi[\alpha \mapsto r])(w)(n)\end{array}\right\}$$

To prove the counters are contextually equivalent we show them directly related in the value relation. We choose the types σ and σ' to be $\mathtt{ref}\,\mathtt{nat}$ and the relation r to be $\lambda w.\left\{(\ell_1,\ell_2) \;\middle|\; \left(\ell_1,\ell_2,\left\{(\underline{n},\underline{2\cdot n}) \;\middle|\; n \in \mathbb{N}\right\}\right) \in w\right\}$. We now need to check all three functions to be related at the value relation.

First, the allocation functions. We only show one approximation, the other is completely analogous. Concretely, we show that for any $n \in \mathbb{N}$ and any world $w \in \mathcal{W}$ we have $(\lambda - .\mathtt{ref}\ \underline{1}, \lambda - .\mathtt{ref}\ \underline{2}) \in [\![\mathbf{1} \to \alpha]\!](r)(w)(n)$. Let $n \in \mathbb{N}$ and $w \in \mathcal{W}$. Take $w' \geq w$ and related arguments v, v' at type $\mathbf{1}$. We know by construction that $v = v' = \langle\rangle$ so we have to show that $(\mathtt{ref}\ \underline{1}, \mathtt{ref}\ \underline{2}) \in [\![\alpha]\!](r)^{\top\top}(w')(n)$.

Let $w'' \geq w'$ and $j \leq n$ and take two related evaluation contexts (E, E') at $[\![\alpha]\!](r)^{\top}(w'')(j)$ and $(h,h') \in \lfloor w'' \rfloor$. Let $\ell^{(\prime)} \notin \mathrm{dom}\left(h^{(\prime)}\right)$. We have

$$\mathbf{Pr}\left(\langle h, E[\mathtt{ref}\ \underline{1}]\rangle \Downarrow^j\right) = \mathbf{Pr}\left(\langle h[\ell \mapsto \underline{1}], E[\ell]\rangle \Downarrow^j\right)$$

and $\mathbf{Pr}\left(\langle h', E'[\mathtt{ref}\ \underline{2}]\rangle \Downarrow\right) = \mathbf{Pr}\left(\langle h'[\ell' \mapsto \underline{2}], E'[\ell']\rangle \Downarrow\right)$.

Let w''' be w'' extended with (ℓ, ℓ', r). Then the extended heaps are in $\lfloor w''' \rfloor$ and $w''' \geq w''$. Thus E and E' are also related at w''' by monotonicity. Similarly we can prove that $(\ell, \ell') \in [\![\alpha]\!](r)(j)(w''')$. This then allows us to conclude $\mathbf{Pr}\left(\langle h[\ell \mapsto \underline{1}], E[\ell]\rangle \Downarrow^j\right) \leq \mathbf{Pr}\left(\langle h'[\ell' \mapsto \underline{2}], E'[\ell']\rangle \Downarrow\right)$ which concludes the proof.

Lookup is simple to show. The more interesting case is the update, which chooses to increase or not.

Let $n \in \mathbb{N}$ and $w \in \mathcal{W}$. Let ℓ and ℓ' be related at $[\![\alpha]\!](r)(w)(n)$. We need to show that $(\langle\rangle \oplus (\ell := \mathtt{S}\,!\ell), \langle\rangle \oplus (\ell' := \mathtt{S}\,(\mathtt{S}\,!\ell'))) \in [\![\mathbf{1}]\!](r)^{\top\top}(w)(n)$. Take $w' \geq w$, $j \leq n$ and $(h,h') \in \lfloor w' \rfloor$. Take related evaluation contexts E and E' at w' and j. We have the following two identities

$$\mathbf{Pr}\left(\langle h, E[\langle\rangle \oplus (\ell := \mathtt{S}\,!\ell)]\rangle \Downarrow^j\right) = \frac{1}{2}\mathbf{Pr}\left(\langle h, E[\langle\rangle]\rangle \Downarrow^j\right) + \frac{1}{2}\mathbf{Pr}\left(\langle h, E[\ell := \mathtt{S}\,!\ell]\rangle \Downarrow^j\right)$$

$$\mathbf{Pr}\left(\langle h', E'[\langle\rangle \oplus (\ell' := \mathtt{S}\,\mathtt{S}\,!\ell')]\rangle \Downarrow\right) = \frac{1}{2}\mathbf{Pr}\left(\langle h', E'[\langle\rangle]\rangle \Downarrow\right) + \frac{1}{2}\mathbf{Pr}\left(\langle h', E'[\ell' := \mathtt{S}\,\mathtt{S}\,!\ell']\rangle \Downarrow\right)$$

Since ℓ and ℓ' are related at $[\![\alpha]\!](r)(w)(n)$ and $w' \geq w$ and $(h,h') \in \lfloor w' \rfloor$ we know that $h(\ell) = \underline{m}$ and $h'(\ell') = \underline{2\cdot m}$ for some $m \in \mathbb{N}$.

Thus $\mathbf{Pr}\left(\langle h, E[\ell := \mathtt{S}\,!\ell]\rangle \Downarrow^j\right) = \mathbf{Pr}\left(\langle h_1, E[\langle\rangle]\rangle \Downarrow^j\right)$ where $h_1 = h[\ell \mapsto \underline{m+1}]$. Similarly we have $\mathbf{Pr}\left(\langle h', E'[\ell' := \mathtt{S}\,\mathtt{S}\,!\ell']\rangle \Downarrow\right) = \mathbf{Pr}\left(\langle h_2, E'[\langle\rangle]\rangle \Downarrow\right)$ where $h_2 = h'\left[\ell' \mapsto \underline{2\cdot(m+1)}\right]$. The fact that h_1 and h_2 are still related concludes the proof.

The above proof shows that reasoning about examples involving state and choice is possible and that the two features are largely orthogonal.

7 Conclusion

We have constructed a step-indexed logical relation for a higher-order language with probabilistic choice. In contrast to earlier work, our language also features

impredicative polymorphism and recursive types. We also show how to extend our logical relation to a language with dynamically allocated local state. In future work, we will explore whether the step-indexed technique can be used for developing models of program logics for probabilistic computation that support reasoning about more properties than just contextual equivalence. We are also interested in extensions to include primitives for continuous probability distributions.

References

1. Ahmed, A.: Step-indexed syntactic logical relations for recursive and quantified types. In: Proceedings of ESOP (2006)
2. Appel, A.W., McAllester, D.: An indexed model of recursive types for foundational proof-carrying code. ACM Transactions on Programming Languages and Systems 23(5) (2001)
3. Birkedal, L., Bizjak, A., Schwinghammer, J.: Step-indexed relational reasoning for countable nondeterminism. Logical Methods in Computer Science 9(4) (2013)
4. Crubillé, R., Lago, U.D.: On probabilistic applicative bisimulation and call-by-value -calculi. In: Proceedings of ESOP (2014)
5. Dal Lago, U., Sangiorgi, D., Alberti, M.: On coinductive equivalences for higher-order probabilistic functional programs. In: Proceedings of POPL (2014)
6. Danos, V., Harmer, R.S.: Probabilistic game semantics. ACM Transactions on Computational Logic 3(3) (2002)
7. Dreyer, D., Ahmed, A., Birkedal, L.: Logical step-indexed logical relations. Logical Methods in Computer Science 7(2) (2011)
8. Ehrhard, T., Pagani, M., Tasson, C.: The computational meaning of probabilistic coherence spaces. In: Proceedings of LICS (2011)
9. Ehrhard, T., Tasson, C., Pagani, M.: Probabilistic coherence spaces are fully abstract for probabilistic PCF. In: Proceedings of POPL (2014)
10. Johann, P., Simpson, A., Voigtländer, J.: A generic operational metatheory for algebraic effects. In: Proceedings of LICS (2010)
11. Jones, C., Plotkin, G.: A probabilistic powerdomain of evaluations. In: Proceedings of LICS (1989)
12. Lago, U.D., Zorzi, M.: Probabilistic operational semantics for the lambda calculus. RAIRO - Theoretical Informatics and Applications 46 (2012)
13. Pitts, A.M.: Parametric polymorphism and operational equivalence. Mathematical Structures in Computer Science 10(3) (2000)
14. Pitts, A.M.: Typed operational reasoning. In: Pierce, B.C. (ed.) Advanced Topics in Types and Programming Languages, chap. 7. MIT Press (2005)
15. Pitts, A.M.: Step-indexed biorthogonality: a tutorial example. In: Ahmed, A., Benton, N., Birkedal, L., Hofmann, M. (eds.) Modelling, Controlling and Reasoning About State. No. 10351 in Dagstuhl Seminar Proceedings (2010)
16. Ramsey, N., Pfeffer, A.: Stochastic lambda calculus and monads of probability distributions. In: Proceedings of POPL (2002)
17. Saheb-Djahromi, N.: Cpo's of measures for nondeterminism. Theoretical Computer Science 12(1) (1980)

Appendix

A Language definitions and properties

$$
\begin{aligned}
\tau &::= \alpha \mid \mathbf{1} \mid \mathtt{nat} \mid \tau_1 \times \tau_2 \mid \tau_1 + \tau_2 \mid \tau_1 \to \tau_2 \mid \mu\alpha.\tau \mid \forall\alpha.\tau \mid \exists\alpha.\tau \\
v &::= x \mid \langle\rangle \mid \underline{n} \mid \langle v_1, v_2\rangle \mid \lambda x.e \mid \mathtt{inl}\ v \mid \mathtt{inr}\ v \mid \Lambda.e \mid \mathtt{pack}\, v \\
e &::= x \mid \langle\rangle \mid \underline{n} \mid \langle e_1, e_2\rangle \mid \lambda x.e \mid \mathtt{inl}\ e \mid \mathtt{inr}\ e \mid \Lambda.e \mid \mathtt{pack}\, e \\
&\quad \mid \mathtt{proj}_i\, e \mid e_1\, e_2 \mid \mathtt{match}\,(e, x_1.e_1, x_2.e_2) \mid e[] \\
&\quad \mid \mathtt{unpack}\ e_1\ \mathtt{as}\ x\ \mathtt{in}\ e_2 \mid \mathtt{unfold}\, e \mid \mathtt{fold}\, e \mid \mathtt{rand}\, e \\
&\quad \mid \mathtt{if}_1\ e\ \mathtt{then}\ e_1\ \mathtt{else}\ e_2 \mid \mathtt{P}\, e \mid \mathtt{S}\, e \\
E &::= - \mid \langle E, e\rangle \mid \langle v, E\rangle \mid \mathtt{inl}\ E \mid \mathtt{inr}\ E \mid \mathtt{pack}\, E \\
&\quad \mid \mathtt{proj}_i\, E \mid E\, e \mid v\, E \mid \mathtt{match}\,(E, x_1.e_1, x_2.e_2) \mid E[] \\
&\quad \mid \mathtt{unpack}\ E\ \mathtt{as}\ x\ \mathtt{in}\ e \mid \mathtt{unfold}\, E \mid \mathtt{fold}\, E \\
&\quad \mid \mathtt{if}_1\ E\ \mathtt{then}\ e_1\ \mathtt{else}\ e_2 \mid \mathtt{rand}\, E \mid \mathtt{P}\, E \mid \mathtt{S}\, E
\end{aligned}
$$

Fig. 2. Types, terms and evaluation contexts. $\underline{n}$ are numerals of type $\mathtt{nat}$.

$$
\frac{\alpha \in \Delta}{\Delta \vdash \alpha} \qquad \Delta \vdash \mathbf{1} \qquad \Delta \vdash \mathtt{nat} \qquad \frac{\Delta \vdash \tau_1 \qquad \Delta \vdash \tau_2}{\Delta \vdash \tau_1 \times \tau_2} \qquad \frac{\Delta \vdash \tau_1 \qquad \Delta \vdash \tau_2}{\Delta \vdash \tau_1 + \tau_2}
$$

$$
\frac{\Delta \vdash \tau_1 \qquad \Delta \vdash \tau_2}{\Delta \vdash \tau_1 \to \tau_2} \qquad \frac{\Delta, \alpha \vdash \tau}{\Delta \vdash \exists\alpha.\tau} \qquad \frac{\Delta, \alpha \vdash \tau}{\Delta \vdash \forall\alpha.\tau} \qquad \frac{\Delta, \alpha \vdash \tau}{\Delta \vdash \mu\alpha.\tau}
$$

Fig. 3. Well-formed types. The judgment $\Delta \vdash \tau$ expresses $ftv(\tau) \subseteq \Delta$.

The following lemma uses definitions from Section 3.

Lemma A.1. *Φ is monotone and preserves suprema of ω-chains.*

Proof. Since the order in $\mathcal{F}$ is pointwise and multiplication and addition are monotone it is easy to see that Φ is monotone.

To show that it is continuous let $\{f_n\}_{n\in\omega}$ be an ω-chain in $\mathcal{F}$. If e is a value the result is immediate. Otherwise we have

$$
\Phi\left(\sup_{n\in\omega} f_n\right)(e) = \sum_{e \overset{p}{\rightsquigarrow} e'} p \cdot \left(\sup_{n\in\omega} f_n\right)(e')
$$

and since suprema in $\mathcal{F}$ are computed pointwise we have

$$= \sum_{e \overset{p}{\rightsquigarrow} e'} p \cdot \sup_{n \in \omega} \left(f_n(e') \right)$$

Using the fact that sum and product are continuous and that the sum in the definition of Φ is finite we get

$$\Phi \left(\sup_{n \in \omega} f_n \right) (e) = \sup_{n \in \omega} \left(\sum_{e \overset{p}{\rightsquigarrow} e'} p \cdot f_n(e') \right)$$
$$= \sup_{n \in \omega} \Phi \left(f_n \right) (e) = \left(\sup_{n \in \omega} \Phi(f_n) \right) (e)$$

Example A.2. Let us compute probabilities of termination of some example programs.

- If $v \in \mathbf{Val}$ then by definition $\mathbf{Pr} \left(v \Downarrow \right) = 1$.
- If $e \in \mathbf{Tm} \setminus \mathbf{Val}$ is stuck then $\mathbf{Pr} \left(e \Downarrow \right) = 0$ by definition.
- Suppose there exists a cycle $e \overset{1}{\rightsquigarrow} e_1 \overset{1}{\rightsquigarrow} e_2 \overset{1}{\rightsquigarrow} \cdots \overset{1}{\rightsquigarrow} e_n \overset{1}{\rightsquigarrow} e$. Then $\mathbf{Pr} \left(e \Downarrow \right) = \mathbf{Pr} \left(e_1 \Downarrow \right) = \cdots = \mathbf{Pr} \left(e_n \Downarrow \right) = 0$.
  It follows from the assumption that none of e_k are values and since the sum of outgoing weights is at most 1 we have that for each e_k and e all other weights must be 0. We thus get that $\mathbf{Pr} \left(e \Downarrow \right) = \mathbf{Pr} \left(e_1 \Downarrow \right) = \cdots = \mathbf{Pr} \left(e_n \Downarrow \right)$ by simply unfolding the fixed point n-times. To show that they are all 0 we use Scott induction. Define

  $$\mathcal{S} = \{ f \in \mathcal{F} | f(e) = f(e_1) = f(e_2) = \ldots = f(e_n) = 0 \} .$$

  Clearly $\mathcal{S}$ is an admissible subset of $\mathcal{F}$ and $\bot \in \mathcal{S}$. Using the above existence of the cycle of reductions it is easy to show that $\mathcal{S} \subseteq \Phi \left[\mathcal{S} \right]$. Hence by the principle of Scott induction we have $\mathbf{Pr} \left(\cdot \Downarrow \right) \in \mathcal{S}$ and thus $\mathbf{Pr} \left(e \Downarrow \right) = \mathbf{Pr} \left(e_1 \Downarrow \right) = \ldots = \mathbf{Pr} \left(e_n \Downarrow \right) = 0$.

This example also shows that we do really want the least fixed point of Φ, since this allows us to use Scott-induction and prove that diverging terms have zero probability of termination.

Remark A.3. It is perhaps instructive to consider the relationship to the termination predicate when we do not have weights on reductions. In such a case we can consider two extremes, may- and must-termination predicates. These can be considered to be maps $\mathbf{Tm} \to \mathbf{2}$ where $\mathbf{2}$ is the boolean lattice $0 \leq 1$. Let $\mathcal{B} = \mathbf{Tm} \to \mathbf{2}$. Since $\mathbf{2}$ is a complete lattice so is $\mathcal{B}$. In particular it is a pointed ω-cpo. We can define may-termination as the least fixed point of $\Psi : \mathcal{B} \to \mathcal{B}$ defined as

$$\Psi(f)(e) = \begin{cases} 1 & \text{if } e \in \mathbf{Val} \\ \max\limits_{e \rightsquigarrow e'} f(e') & \text{otherwise} \end{cases} .$$

Observe again that if e is stuck then $\Psi(f)(e) = 0$ since the maximum of an empty set is the least element by definition.

Must-termination is slightly different. We need a special case for stuck terms.

$$\Psi'(f)(e) = \begin{cases} 1 & \text{if } e \in \mathbf{Val} \\ \min\limits_{e \rightsquigarrow e'} f(e') & \exists e' \in \mathbf{Tm}, p \in \mathcal{I}, e \overset{p}{\rightsquigarrow} e' \\ 0 & \text{otherwise} \end{cases}$$

Let $\downarrow$ be the least fixed point of Ψ and $\Downarrow$ the least fixed point of Ψ'. An additional property that holds for $\downarrow$ and $\Downarrow$, because of the fact that $\mathbf{2}$ is discrete, is that for a given e, if $e\downarrow = 1$ then there is a natural number n, such that $\Psi^n(\bot)(e) = 1$, i.e. if it terminates we can observe this in finite time. This is because if an increasing sequence in $\mathbf{2}$ has supremum 1, then the sequence must be constant 1 from some point onward.

In contrast, if $\mathbf{Pr}\,(e \Downarrow) = 1$ it is not necessarily the case that there is a natural number n with $\Phi^n(\bot)(e) = 1$ because it might be the case that 1 is only reached in the limit.

The next lemma uses the abbreviation ; defined in Section 5.

Lemma A.4. *For all terms* $e, e' \in \mathbf{Tm}$, $\mathbf{Pr}\,(e; e' \Downarrow) = \mathbf{Pr}\,(e \Downarrow) \cdot \mathbf{Pr}\,(e' \Downarrow)$.

Proof. We prove two approximations separately, both of them by Scott induction.

$\leq$ Consider the set

$$\mathcal{S} = \left\{ f \in \mathcal{F} \;\middle|\; \begin{array}{l} f \leq \mathbf{Pr}\,(\cdot \Downarrow) \wedge \forall e, e' \in \mathbf{Tm}, \\ f(e; e') \leq \mathbf{Pr}\,(e \Downarrow) \cdot \mathbf{Pr}\,(e' \Downarrow) \end{array} \right\}.$$

It is easy to see that $\mathcal{S}$ contains $\bot$ and is closed under ω-chains, so we only need to show that it is preserved by Φ. The first condition is trivial to check since $\mathbf{Pr}\,(\cdot \Downarrow)$ is a fixed point of Φ. Let $f \in \mathcal{F}$ and $e, e' \in \mathbf{Tm}$. If $e \in \mathbf{Val}$ then $\Phi(f)(e; e') = f(e')$ on account of one β-reduction. By assumption $f(e') \leq \mathbf{Pr}\,(e' \Downarrow)$ and by definition we have $\mathbf{Pr}\,(e \Downarrow) = 1$.
If e is not a value we have $\Phi(f)(e; e') = \sum_{e \overset{p}{\rightsquigarrow} e''} p \cdot f\,(e''; e') \leq \sum_{e \overset{p}{\rightsquigarrow} e''} p \cdot \mathbf{Pr}\,(e'' \Downarrow) \cdot \mathbf{Pr}\,(e' \Downarrow) = \mathbf{Pr}\,(e \Downarrow) \cdot \mathbf{Pr}\,(e' \Downarrow)$.
Thus we can conclude by Scott induction that $\mathbf{Pr}\,(\cdot \Downarrow) \in \mathcal{S}$.

$\geq$ For this direction we consider the set

$$\mathcal{S} = \left\{ f \in \mathcal{F} \;\middle|\; \begin{array}{l} \forall E \in \mathbf{Stk}, e \in \mathbf{Tm}, v \in \mathbf{Val}, \\ \mathbf{Pr}\,(E[e] \Downarrow) \geq f(e) \cdot \mathbf{Pr}\,(E[v] \Downarrow) \end{array} \right\}.$$

It is easy to see that it is admissible and closed under Φ. Hence $\mathbf{Pr}\,(\cdot \Downarrow) \in \mathcal{S}$. Thus we have, taking $E = -; e'$ and any value v, that $\mathbf{Pr}\,(e \Downarrow) \cdot \mathbf{Pr}\,(v; e' \Downarrow) \leq \mathbf{Pr}\,(e; e' \Downarrow)$ and it is easy to see that $\mathbf{Pr}\,(v; e' \Downarrow) = \mathbf{Pr}\,(e' \Downarrow)$.

Lemma A.5. *Let* $e, e' \in \mathbf{Tm}$. *If* $e \overset{cuff}{\Longrightarrow} e'$ *then for all* k, $\mathbf{Pr}\left(e \Downarrow^k\right) = \mathbf{Pr}\left(e' \Downarrow^k\right)$.

Proof. When k is 0 the result is immediate. So assume $k > 0$. We need to distinguish two cases.

- If there exists $v' \in \mathbf{Val}$ such that $e' \overset{\text{cuff}}{\Longrightarrow} v'$ then we also have $e \overset{\text{cuff}}{\Longrightarrow} v'$ and we are done.
- If not, then we need to inspect the definition of $\mathbf{Red}(e)$ and $\mathbf{Red}(e')$. It is easy to see that any path $\pi \in \mathbf{Red}(e')$ corresponds to a unique path $\pi' \cdot \pi$ in $\mathbf{Red}(e)$. It is similarly easy to see that $\mathcal{W}(\pi) = \mathcal{W}(\pi' \cdot \pi)$ and that $\ell(\pi) = \ell(\pi' \cdot \pi)$. Thus we have that $\mathbf{Pr}\left(e \Downarrow^k\right) = \mathbf{Pr}\left(e' \Downarrow^k\right)$.

Proposition A.6. *For each $e \in \mathbf{Tm}$ we have $\mathbf{Pr}(e \Downarrow) \leq \sup_{k\in\omega}\left(\mathbf{Pr}\left(e \Downarrow^k\right)\right)$.*

Proof. We use Scott induction. Let $\mathcal{S}$ be the set

$$\mathcal{S} = \left\{ f \in \mathcal{F} \;\middle|\; \forall e, f(e) \leq \sup_{k\in\omega}\left(\mathbf{Pr}\left(e \Downarrow^k\right)\right) \right\}$$

It is easy to see that $\mathcal{S}$ is closed under limits of ω-chains and that $\bot \in \mathcal{S}$ so we only need to show that $\mathcal{S}$ is closed under Φ. Let $f \in \mathcal{S}$ and e an expression. We have

$$\Phi(f)(e) = \begin{cases} 1 & \text{if } e \in \mathbf{Val} \\ \displaystyle\sum_{e \overset{p}{\rightsquigarrow} e'} p \cdot f(e') & \text{otherwise} \end{cases}$$

and we consider 4 cases.

- $e \in \mathbf{Val}$. We always have $e \overset{\text{cuff}}{\Longrightarrow} e$ and so we have that for any $k > 0$, $\mathbf{Pr}\left(e \Downarrow^k\right) = 1$ which is the top element.
- $e \overset{p}{\rightsquigarrow} e'$ and the reduction is not unfold-fold or choice. Then we use Lemma 3 to get $\mathbf{Pr}\left(e \Downarrow^k\right) = \mathbf{Pr}\left(e' \Downarrow^k\right)$ for all k. Similarly we have that $\Phi(f)(e) = f(e')$ from the definition of Φ. Thus we can use the assumption that $f \in \mathcal{S}$.
- $e \overset{1}{\rightsquigarrow} e'$ and the reduction is unfold-fold. This follows directly from the definition of $\mathbf{Red}(\cdot)$, Ψ and the assumption that $f \in \mathcal{S}$.
- The reduction from e is a choice reduction. Suppose e reduces to $e_1, e_2, \ldots, e_n$. Then we know from the operational semantics that the weights are all $\frac{1}{n}$. We get

$$\Phi(f)(e) = \sum_{i=1}^{n} \frac{1}{n} f(e_i) \qquad \text{and} \qquad \mathbf{Pr}\left(e \Downarrow^{k+1}\right) = \sum_{i=1}^{n} \frac{1}{n} \mathbf{Pr}\left(e_i \Downarrow^k\right). \quad (2)$$

  Using the fact that $\mathbf{Pr}\left(e_i \Downarrow^k\right)$ is an increasing chain in k for each e_i we have

$$\sup_{k\in\omega}\left(\mathbf{Pr}\left(e \Downarrow^k\right)\right) = \sum_{i=1}^{n} \frac{1}{n} \sup_{k\in\omega}\left(\mathbf{Pr}\left(e_i \Downarrow^k\right)\right) \quad (3)$$

  By assumption $f(e_i) \leq \sup_{k\in\omega}\left(\mathbf{Pr}\left(e_i \Downarrow^k\right)\right)$ for all $i \in \{1, 2, \ldots, n\}$ which concludes the proof using (**??**) and (**??**).

Interpretation of types and the logical relation

Lemma A.7. *The interpretation of types in Figure 1 is well defined. In particular the interpretation of types is non-expansive.*

The substitution lemma is crucial for proving compatibility of existential and universal types. The proof is by induction.

Lemma A.8 (Substitution). *For any well-formed types $\Delta, \alpha \vdash \tau$ and $\Delta \vdash \sigma$ and any φ we have $[\![\Delta \vdash \tau[\sigma/\alpha]]\!](\varphi) = [\![\Delta, \alpha \vdash \tau]\!](\varphi[\alpha \mapsto [\![\Delta \vdash \sigma]\!](\varphi)])$.*

We state and prove additional context extension lemmas. The other cases are similar.

Lemma A.9. *Let $n \in \mathbb{N}$. If $(v, v') \in [\![\Delta \vdash \tau_1 \to \tau_2]\!](\varphi)(n)$ and $(E, E') \in [\![\Delta \vdash \tau_2]\!](\varphi)^{\top}(n)$ then $(E \circ (v\,[]), E' \circ (v'\,[])) \in [\![\Delta \vdash \tau_1]\!](\varphi)^{\top}(n)$.*

This follows directly from the definition of the interpretation of types.

Corollary A.10. *Let $n \in \mathbb{N}$. If $(e, e') \in [\![\Delta \vdash \tau_1]\!](\varphi)^{\top\top}(n)$ and $(E, E') \in [\![\Delta \vdash \tau_2]\!](\varphi)^{\top}(n)$ then*

$$(E \circ ([]\,e), E' \circ ([]\,e')) \in [\![\Delta \vdash \tau_1 \to \tau_2]\!](\varphi)^{\top}(n).$$

Proof. Let $n \in \mathbb{N}$. Take $(v, v') \in [\![\Delta \vdash \tau_1 \to \tau_2]\!](\varphi)(n)$. By Lemma **??** and monotonicity we have for all $k \leq n$, $(E \circ (v\,[]), E' \circ (v'\,[])) \in [\![\Delta \vdash \tau_1]\!](\varphi)^{\top}(k)$ and by the assumption that $(e, e') \in [\![\Delta \vdash \tau_1]\!](\varphi)^{\top\top}(n)$ we have

$$\mathbf{Pr}\left(E[v\,e] \Downarrow^k\right) \leq \mathbf{Pr}\left(E'[v'\,e'] \Downarrow\right)$$

concluding the proof.

Lemma A.11. *Let $n \in \mathbb{N}$. If $(E, E') \in [\![\Delta \vdash \tau[\mu\alpha.\tau/\alpha]]\!](\varphi)^{\top}(n)$ then*

$$(E \circ (\mathtt{unfold}\,[]), E' \circ (\mathtt{unfold}\,[])) \in [\![\Delta \vdash \mu\alpha.\tau]\!](\varphi)^{\top}(n).$$

Proof. Let $n \in \mathbb{N}$. We consider two cases.

- $n = m + 1$
  Take $(\mathtt{fold}\,v, \mathtt{fold}\,v') \in [\![\Delta \vdash \mu\alpha.\tau]\!](\varphi)(n)$. By definition

  $$(v, v') \in [\![\Delta \vdash \tau[\mu\alpha.\tau/\alpha]]\!](\varphi)(m).$$

  Let $k \leq n$. If $k = 0$ the condition is trivially true (since $\mathbf{Pr}\left(E[\mathtt{unfold}\,\mathtt{fold}\,v] \Downarrow^k\right) = 0$) so assume $k = \ell + 1$. Note that crucially $\ell \leq m$. Using Lemma 5, Lemma 4 and Lemma 2 we have

  $$\begin{aligned}\mathbf{Pr}\left(E[\mathtt{unfold}\,(\mathtt{fold}\,v)] \Downarrow^k\right) &= \mathbf{Pr}\left(E[v] \Downarrow^\ell\right)\\ &\leq \mathbf{Pr}\left(E'[v'] \Downarrow\right)\\ &= \mathbf{Pr}\left(E'[\mathtt{unfold}\,(\mathtt{fold}\,v')] \Downarrow\right)\end{aligned}$$

  concluding the proof.

– $n = 0$. This case is trivial, since $\mathbf{Pr}\left(e \Downarrow^0\right) = 0$ for any e.

Lemma A.12. *Let $n \in \mathbb{N}$. If $(E, E') \in [\![\Delta \vdash \mu\alpha.\tau]\!]\,(\varphi)^{\top}(n)$ then*

$$(E \circ (\mathtt{fold}\,[]), E' \circ (\mathtt{fold}\,[])) \in [\![\Delta \vdash \tau[\mu\alpha.\tau/\alpha]]\!]\,(\varphi)^{\top}(n).$$

Proof. Easily follows from the fact that if (v, v') are related at the unfolded type then $(\mathtt{fold}\,v, \mathtt{fold}\,v')$ are related at the folded type (using weakening to get to the same stage).

To relate the logical relation to contextual and CIU approximations we first have that the composition of logical and CIU approximations is included in the logical approximation relation.

Corollary A.13. *If $\Delta \mid \Gamma \vdash e \lesssim^{log} e' : \tau$ and $\Delta \mid \Gamma \vdash e' \lesssim^{CIU} e'' : \tau$ then $\Delta \mid \Gamma \vdash e \lesssim^{log} e'' : \tau$.*

This follows directly from the definition. This corollary in turn implies, together with Proposition 2 and the fact that all compatible relations are in particular reflexive, that CIU approximation relation is contained in the logical relation.

Corollary A.14. *If $\Delta \mid \Gamma \vdash e \lesssim^{CIU} e' : \tau$ then $\Delta \mid \Gamma \vdash e \lesssim^{log} e' : \tau$.*

Finally we have adequacy of the logical relation.

Corollary A.15. *Logical approximation relation $\lesssim^{log}$ is adequate.*

Proof. Assume $\varnothing \mid \varnothing \vdash e \lesssim^{log} e' : \tau$. We are to show that $\mathbf{Pr}\,(e \Downarrow) \leq \mathbf{Pr}\,(e' \Downarrow)$. Straight from the definition we have $\forall n \in \mathbb{N}, (e, e') \in [\![\varnothing \vdash \tau]\!]^{\top\top}(n)$. The empty evaluation context is always related to itself (at any type). This implies $\forall n \in \mathbb{N}, \mathbf{Pr}\,(e \Downarrow^n) \leq \mathbf{Pr}\,(e' \Downarrow)$ which further implies (since the right-hand side is independent of n) that $\sup_{n\in\omega}(\mathbf{Pr}\,(e \Downarrow^n)) \leq \mathbf{Pr}\,(e' \Downarrow)$. Using Proposition 1 we thus have $\mathbf{Pr}\,(e \Downarrow) \leq \sup_{n\in\omega}(\mathbf{Pr}\,(e \Downarrow^n)) \leq \mathbf{Pr}\,(e' \Downarrow)$ concluding the proof.

Lemma A.16 (Functional extensionality for values). *Suppose $\tau, \sigma \in \mathfrak{T}(\Delta)$ and let $\lambda x.e$ and $\lambda x'.e'$ be two values of type $\tau \to \sigma$ in context $\Delta \mid \Gamma$. If for all $u \in \mathbf{Val}\,(\tau)$ we have $\Delta \mid \Gamma \vdash (\lambda x.e)\ u \lesssim^{ctx} (\lambda x'.e')\ u : \sigma$ then*

$$\Delta \mid \Gamma \vdash \lambda x.e \lesssim^{ctx} \lambda x'.e' : \tau \to \sigma\ .$$

Proof. We use Theorem 1 several times and show $\lambda x.e$ and $\lambda x'.e'$ are logically related. Let $n \in \mathbb{N}$, $\varphi \in \mathbf{VRel}\,(\Delta)$ and $(\gamma, \gamma') \in [\![\Delta \vdash \Gamma]\!]\,(\varphi)(n)$. Let $v = \lambda x.e\gamma$ and $v' = \lambda x'.e'\gamma'$. We are to show $(v, v') \in [\![\Delta \vdash \tau \to \sigma]\!]\,(\varphi)^{\top\top}(n)$ and to do this we show directly $(v, v') \in [\![\Delta \vdash \tau \to \sigma]\!]\,(\varphi)(n)$.

Let $j \leq n$, $(u, u') \in [\![\tau]\!]\,(\varphi)(n)$, $k \leq j$ and $(E, E') \in [\![\sigma]\!]\,(\varphi)^{\top}(k)$. We have to show $\mathbf{Pr}\left(E[v\,u] \Downarrow^k\right) \leq \mathbf{Pr}\,(E'[v'\,u'] \Downarrow)$. From Proposition 2 we have that $(v, v) \in [\![\tau \to \sigma]\!]\,(\varphi)^{\top\top}(n)$ and so $\mathbf{Pr}\left(E[v\,u] \Downarrow^k\right) \leq \mathbf{Pr}\,(E'[v\,u'] \Downarrow)$. From the assumption of the lemma we have that $v\,u' \lesssim^{CIU} v'\,u'$ which concludes the proof.

Lemma A.17 (Extensionality for the universal type). *Let $\tau \in \mathfrak{T}(\Delta, \alpha)$ be a type. Let $\Lambda.e, \Lambda.e'$ be two terms of type $\forall\alpha.\tau$ in context $\Delta \mid \Gamma$. If for all closed types $\sigma \in \mathfrak{T}$ we have*

$$\Delta \mid \Gamma \vdash e \lesssim^{ctx} e' : \tau[\sigma/\alpha]$$

then $\Delta \mid \Gamma \vdash \Lambda.e \lesssim^{ctx} \Lambda.e' : \forall\alpha.\tau$.

Proof. We again use Theorem 1 multiple times. Let $n \in \mathbb{N}$, $\varphi \in \mathbf{VRel}\,(\Delta)$ and $(\gamma, \gamma') \in [\![\Delta \vdash \Gamma]\!]\,(\varphi)(n)$. Let $v = \Lambda.e\gamma$ and $v' = \Lambda.e'\gamma'$. We show directly that $(v, v') \in [\![\Delta \vdash \forall\alpha.\tau]\!]\,(\varphi)(n)$.

So take $\sigma, \sigma' \in \mathfrak{T}$ and $r \in \mathbf{VRel}\,(\sigma, \sigma')$ and we need to show $(e\gamma, e'\gamma') \in [\![\Delta, \alpha]\!]\,(\varphi\,[\alpha \mapsto r])^{\top\top}(n)$. Let $k \leq n$ and (E, E') related at k. We have to show $\mathbf{Pr}\left(E[e\gamma] \Downarrow^k\right) \leq \mathbf{Pr}\,(E'[e'\gamma'] \Downarrow)$. From Proposition 2 we have

$$(e\gamma, e\gamma') \in [\![\Delta, \alpha]\!]\,(\varphi\,[\alpha \mapsto r])^{\top\top}(n)$$

and so $\mathbf{Pr}\left(E[e\gamma] \Downarrow^k\right) \leq \mathbf{Pr}\,(E'[e\gamma'] \Downarrow)$. Let $\vec{\sigma}$ be the types for the right hand side in φ. Then $E' \in \mathbf{Stk}\,(\tau[\vec{\sigma}, \sigma'/\Delta, \alpha])$. Using the assumption of the lemma we get that $e\gamma' \lesssim^{CIU} e'\gamma'$ at the type $\tau[\vec{\sigma}, \sigma'/\Delta, \alpha]$ which immediately implies that $\mathbf{Pr}\,(E'[e\gamma'] \Downarrow) \leq \mathbf{Pr}\,(E'[e'\gamma'] \Downarrow)$ concluding the proof.

B The probability of termination

We prove the claims from Section 5 about the termination probability.

Proposition B.1. *For any expression e, $\mathbf{Pr}\,(e \Downarrow)$ is a left-computable real number.*

Proof. We first prove by induction that for any n, $\Phi^n(\bot)$ restricts to a map $\mathbf{Tm} \to [0,1] \cap \mathbb{Q}$. The proof is simple since the function $\bot$ clearly maps into rationals and for the inductive step we use the fact that the sums in the definition of Φ are always finite, and the rational numbers are closed under finite sums.

To conclude the proof we have by definition that $\mathbf{Pr}\,(e \Downarrow) = \sup_{n\in\omega} \Phi^n(\bot)(e)$ and we have just shown that all the numbers $\Phi^n(\bot)(e)$ are rational. Moreover the sequence $\{\Phi^n(\bot)(e)\}_{n\in\mathbb{N}}$ is computable, since for a given n we only need to check all the reductions from e of length at most n to determine the value of $\Phi^n(\bot)(e)$ and the reduction relation $\overset{p}{\rightsquigarrow}$ is naturally computable.

Example B.2. To see that the probability of termination can also be non-computable we informally describe a program whose probability of termination would allow us to solve the halting problem were it computable.

The program we construct is recursively defined as $T = \mathtt{fix}[][], \varphi$ where

$$\varphi = \lambda f.\lambda x.t\,x \oplus (\Omega \oplus f\,(\mathtt{succ}\ x))$$

where $t\,x$ is a program that runs the x-th Turing machine on the empty input and does not use any choice reductions. Thus $\mathbf{Pr}\,(t\,x \Downarrow) \in \{0, 1\}$. It is well known

that the empty string acceptance problem is undecidable. Note that we put Ω in the program to ensure that every second digit in binary will be 0. It is an easy computation to show that

$$\mathbf{Pr}\,(T\,\underline{1}\Downarrow) = \sum_{n=0}^{\infty} \frac{1}{2^{2n+1}} p_{n+1}$$

where $p_n = 1$ if the n-th Turing machine terminates on the empty input and 0 otherwise. If $\mathbf{Pr}\,(T\,\underline{1}\Downarrow)$ were computable we could decide whether a given Turing machine accepts the empty string by computing its index n and then computing the first $2n$ digits of $\mathbf{Pr}\,(T\,\underline{1}\Downarrow)$.

We will now generalize the last example and show that any left-computable real arises as the probability of termination of a program. Technically, we show that given a term of the language that computes an increasing bounded sequence of rationals (represented as pairs of naturals) we can define a program that terminates with probability the supremum of the sequence. We then use the fact that our language $\mathbf{F}^{\mu,\oplus}$ is Turing complete to claim that any computable sequence of rationals can be represented as such a term of $\mathbf{F}^{\mu,\oplus}$.

Proposition B.3. *For every left-computable real in* $[0,1]$ *there is a program* e_r *of type* $\mathbf{1} \to \mathbf{1}$ *such that* $\mathbf{Pr}\,(e_r\,\langle\rangle\Downarrow) = r$.

Proof. So let $r : \mathtt{nat} \to \mathtt{nat} \times \mathtt{nat}$ compute an increasing sequence of rationals in the interval $[0,1]$. Additionally assume that for all $n \in \mathbb{N}$.

$$r\,\underline{n} \overset{\text{cf}}{\Longrightarrow} \langle \underline{k_n}, \underline{\ell_n} \rangle$$

for some $k_n, \ell_n \in \mathbb{N}$. That is, r does not use choice reductions. This is not an essential limitation, but simplifies the argument which we are about to give.

First we define a recursive function e of type $e : (\mathtt{nat} \to \mathtt{nat} \times \mathtt{nat}) \to \mathbf{1}$ as $e = \mathtt{fix}[][]\,\varphi$ where

$$\begin{aligned}\varphi = \lambda f.\lambda r.\mathtt{let}\ (\underline{k},\underline{\ell})\ &=\ r\,\underline{1}\ \mathtt{in}\\ &\mathtt{let}\ y\ =\ \mathtt{rand}\,\underline{\ell}\ \mathtt{in}\\ &\mathtt{if}\ y \leq \underline{k}\ \mathtt{then}\ \langle\rangle\ \mathtt{else}\ f\,r'\end{aligned}$$

and

$$r' = \lambda z.\frac{r\,(\mathtt{succ}\,z) - (\underline{k},\underline{\ell})}{1 - (\underline{k},\underline{\ell})}$$

and subtraction and division is implemented in the obvious way. Note that the condition in φ ensures that $(\underline{k},\underline{\ell})$ does not represent the rational number 1 and therefore division would make sense. But technically, since we implement rationals with pairs of naturals no exception can occur and we just represent the pair with the second component being $\underline{0}$.

Let f and r be values of the appropriate type. We have

$$\mathbf{Pr}\left(\varphi\, f\, r \Downarrow^{m+1}\right) \leq \frac{k_1}{\ell_1}\mathbf{Pr}\left(\langle\rangle \Downarrow^{m}\right) + \frac{\ell_1 - k_1}{\ell_1} \cdot \mathbf{Pr}\left(f\, r' \Downarrow^{m}\right)$$

where $r\,\underline{1} \stackrel{\text{cf}}{\Longrightarrow} (\underline{k}_1, \underline{\ell}_1)$. The inequality comes from the fact that applying r might take some unfold-fold reductions. Iterating this we get

$$\mathbf{Pr}\left(e\, r \Downarrow^{m+1+2n}\right) \leq \frac{k_n}{\ell_n} + \frac{\ell_n - k_n}{\ell_n} \cdot \mathbf{Pr}\left(e\, r^{(n)} \Downarrow^{m+1}\right)$$

where $r\,\underline{n} \stackrel{\text{cf}}{\Longrightarrow} (\underline{k}_n, \underline{\ell}_n)$ and

$$r^{(n)} = \lambda z. \frac{r\left(\mathtt{succ}^n\, z\right) - (\underline{k}_n, \underline{\ell}_n)}{1 - (\underline{k}_n, \underline{\ell}_n)}$$

is the n-th iteration of the $'$ used on r in φ.

It is easy to see that $\mathbf{Pr}\left(e\, r^{(n)} \Downarrow^{1}\right) = 0$ since it takes at least one unfold-fold and one choice reduction to terminate. Thus picking $m = 1$ we have $\mathbf{Pr}\left(e\, r \Downarrow^{2+2n}\right) = \frac{k_n}{\ell_n}$ and thus

$$\sup_{n\in\omega} \mathbf{Pr}\left(e\, r \Downarrow^{n}\right) \leq \sup_{n\in\omega} \frac{k_n}{\ell_n}$$

Using the same reasoning as above we also have

$$\mathbf{Pr}\left(e\, r \Downarrow\right) \geq \frac{k_n}{\ell_n} + \frac{\ell_n - k_n}{\ell_n} \cdot \mathbf{Pr}\left(e\, r^{(n)} \Downarrow\right) \geq \frac{k_n}{\ell_n}$$

which shows (using Proposition 1) that

$$\sup_{n\in\omega} \frac{k_n}{\ell_n} \leq \mathbf{Pr}\left(e\, r \Downarrow\right) \leq \sup_{n\in\omega} \mathbf{Pr}\left(e\, r \Downarrow^{n}\right) \leq \sup_{n\in\omega} \frac{k_n}{\ell_n}$$

and so

$$\sup_{n\in\omega} \frac{k_n}{\ell_n} = \mathbf{Pr}\left(e\, r \Downarrow\right).$$

C Distributions

We now define distributions and prove some of their properties and properties of the probability of termination which are used in the examples.

By a distribution we mean a subprobability measure on the discrete space **Val** of values. Let

$$\mathbf{Dist} = \{f : \mathbf{Val} \to [0,1] | \sum_{v\in\mathbf{Val}} f(v) \leq 1\}$$

be the space of subprobability measures on **Val**. To be precise, $f \in \mathbf{Dist}$ are not measures, but given any f we can define a subprobability measure $\mu_f(A) = \sum_{v \in A} f(v)$ and given any subprobability measure μ, we can define $f_\mu \in \mathbf{Dist}$ as the Radon-Nikodym derivative with respect to the counting measure. Or in more prosaic terms $f_\mu(v) = \mu(\{v\})$. It is easy to see that these two operations are mutually inverse and since $f \in \mathbf{Dist}$ are easier to work with we choose this presentation.

Lemma C.1. Dist *ordered pointwise is a pointed ω-cpo.*

Proof. The bottom element is the everywhere 0 function. Let $\{f_n\}_{n \in \omega}$ be an ω-chain. Define the limit function f as the pointwise supremum

$$f(v) = \sup_{n \in \omega} f_n(v).$$

Clearly all pointwise suprema exist and f is the least upper bound, provided we can show that $f \in \mathbf{Dist}$. To show this last fact we need to show

$$\sum_{v \in \mathbf{Val}} \sup_{n \in \omega} f_n(v) \leq 1.$$

but this is a simple consequence of Fatou's lemma since from the assumption that $\{f_n\}_{n \in \omega}$ we have $\sup_{n \in \omega} f_n(v) = \lim_{n \to \infty} f_n(v) = \liminf_{n \to \infty} f_n(v)$ and so by Fatou's lemma (relative to the counting measure on **Val**) we have

$$\sum_{v \in \mathbf{Val}} \sup_{n \in \omega} f_n(v) \leq \liminf_{n \to \infty} \left(\sum_{v \in \mathbf{Val}} f_n(v) \right) \leq \liminf_{n \to \infty} 1 = 1.$$

Now define $\Xi : (\mathbf{Tm} \to \mathbf{Dist}) \to (\mathbf{Tm} \to \mathbf{Dist})$ as follows

$$\Xi(\varphi)(e) = \begin{cases} \delta_e & \text{if } e \in \mathbf{Val} \\ \displaystyle\sum_{e \overset{p}{\rightsquigarrow} e'} p \cdot \varphi(e') & \text{otherwise} \end{cases}$$

where δ_e is (the density function of) the Dirac measure at point e. Since **Dist** is an ω-cpo so is $\mathbf{Tm} \to \mathbf{Dist}$ ordered pointwise. It is easy to see that in this ordering Ξ is monotone and continuous and so by Kleene's fixed point theorem it has a least fixed point reached in ω iterations. Let $\mathcal{D} = \sup_{n \in \omega} (\Xi^n(\bot))$ be this fixed point.

Lemma C.2. *Let $e \in \mathbf{Tm}$ and $v \in \mathbf{Val}$. If $\mathcal{D}(e)(v) > 0$ then there exists a path π from e to v, i.e. e steps to v.*

Proof. We use Scott induction. Define

$$\mathcal{S} = \left\{ f : \mathbf{Tm} \to \mathbf{Dist} \;\middle|\; \forall e, v, f(e)(v) > 0 \to \exists \pi, \pi : e \rightsquigarrow^* v \right\}$$

The set $\mathcal{S}$ contains $\bot$. To see that it is closed under ω-chains observe that if $(\sup_{n\in\omega} f_n)(e)(v) > 0$ then there must be $n \in \omega$, such that $f_n(e)(v) > 0$ so we may use the path from e to v that we know exists from the assumption that $f_n \in \mathcal{S}$.

It is similarly easy to see that given $f \in \mathcal{S}$ we have $\Xi(f) \in \mathcal{S}$. Thus we have that $\mathcal{D} \in \mathcal{S}$ concluding the proof.

Lemma C.3. *For any expression $e \in \mathbf{Tm}$ we have*

$$\sum_{v\in\mathbf{Val}} \mathcal{D}(e)(v) = \mathbf{Pr}(e \Downarrow)$$

Proof. First we show by induction on n that all the finite approximations of $\mathbf{Pr}(e \Downarrow)$ and $\mathcal{D}(e)$ agree.

- The base case is trivial since by definition

$$\sum_{v\in\mathbf{Val}} \Xi^0(\bot)(e)(v) = 0 = \Phi^0(\bot)(e)$$

- For the inductive case we consider two cases. If $e \in \mathbf{Val}$ then both sides are 1. In the other case we have

$$\begin{aligned}\sum_{v\in\mathbf{Val}} \Xi^{n+1}(\bot)(e)(v) &= \sum_{v\in\mathbf{Val}} \left(\sum_{e\overset{p}{\rightsquigarrow}e'} p \cdot \Xi^n(e')\right)(v) \\ &= \sum_{v\in\mathbf{Val}} \left(\sum_{e\overset{p}{\rightsquigarrow}e'} p \cdot \Xi^n(e')(v)\right)\end{aligned}$$

by Tonelli's theorem we can we can interchange the sums to get

$$\begin{aligned}&= \sum_{e\overset{p}{\rightsquigarrow}e'} \left(p \sum_{v\in\mathbf{Val}} \Xi^n(e')(v)\right) \\ &= \sum_{e\overset{p}{\rightsquigarrow}e'} p \cdot \Phi^n(\bot)(e') = \Phi^{n+1}(\bot)(e)\end{aligned}$$

Thus we have that for all n,

$$\sum_{v\in\mathbf{Val}} \Xi^n(\bot)(e)(v) = \Phi^n(\bot)(e)$$

and so

$$\sup_{n\in\omega} \left(\sum_{v\in\mathbf{Val}} \Xi^n(\bot)(e)(v)\right) = \sup_{n\in\omega} (\Phi^n(\bot)(e)) = \mathbf{Pr}(e \Downarrow)$$

By the dominated convergence theorem we can exchange the sup (which is the limit) and the sum on the left to get

$$\begin{aligned}\sup_{n\in\omega}\left(\sum_{v\in\mathbf{Val}}\Xi^n(\bot)(e)(v)\right) &= \sum_{v\in\mathbf{Val}}\sup_{n\in\omega}\left(\Xi^n(\bot)(e)(v)\right)\\ &= \sum_{v\in\mathbf{Val}}\mathcal{D}(e)(v)\end{aligned}$$

as required.

Proposition C.4 (Monadic bind for distributions). *Let $e \in \mathbf{Tm}$ and E an evaluation context of appropriate type.*

$$\mathcal{D}\left(E[e]\right) = \sum_{v\in\mathbf{Val}}\mathcal{D}(e)(v)\cdot\mathcal{D}\left(E[v]\right).$$

Proof. It is easy to show by induction on ℓ that

$$\forall e\in\mathbf{Tm}, \Xi^{\ell}\left(\bot\right)\left(E[e]\right) = \sum_{v\in\mathbf{Val}}\sum_{\substack{\pi:e\rightsquigarrow^* v\\ \mathbf{len}(\pi)\le\ell}}\mathcal{W}\left(\pi\right)\cdot\Xi^{\ell-\mathbf{len}(\pi)}\left(E[v]\right) \tag{4}$$

(using the fact that the length of the empty path is 0 and its weight 1).

Similarly it is easy to show by induction on ℓ that

$$\forall e\in\mathbf{Tm}, \Xi^{\ell+1}\left(\bot\right)\left(e\right)\left(v\right) = \sum_{\substack{\pi:e\rightsquigarrow^* v\\ \mathbf{len}(\pi)\le\ell}}\mathcal{W}\left(\pi\right) \tag{5}$$

which immediately implies

$$\forall e\in\mathbf{Tm}, \mathcal{D}(e)(v) = \sum_{\pi:e\rightsquigarrow^* v}\mathcal{W}\left(\pi\right) \tag{6}$$

Using these we have

$$\mathcal{D}(E[e]) = \sup_{\ell\in\omega}\sum_{v\in\mathbf{Val}}\sum_{\substack{\pi:e\rightsquigarrow^* v\\ \mathbf{len}(\pi)\le\ell}}\mathcal{W}\left(\pi\right)\cdot\Xi^{\ell-\mathbf{len}(\pi)}\left(E[v]\right)$$

and since for each v the sequence $\displaystyle\sum_{\substack{\pi:e\rightsquigarrow^* v\\ \mathbf{len}(\pi)\le\ell}}\mathcal{W}\left(\pi\right)\cdot\Xi^{\ell-\mathbf{len}(\pi)}\left(E[v]\right)$ is increasing with ℓ we have

$$\begin{aligned}&= \sum_{v\in\mathbf{Val}}\sup_{\ell\in\omega}\sum_{\substack{\pi:e\rightsquigarrow^* v\\ \mathbf{len}(\pi)\le\ell}}\mathcal{W}\left(\pi\right)\cdot\Xi^{\ell-\mathbf{len}(\pi)}\left(E[v]\right)\\ &= \sum_{v\in\mathbf{Val}}\sum_{\pi:e\rightsquigarrow^* v}\mathcal{W}\left(\pi\right)\cdot\mathcal{D}\left(E[v]\right)\\ &= \sum_{v\in\mathbf{Val}}\mathcal{D}\left(E[v]\right)\sum_{\pi:e\rightsquigarrow^* v}\mathcal{W}\left(\pi\right)\\ &= \sum_{v\in\mathbf{Val}}\mathcal{D}(e)(v)\cdot\mathcal{D}\left(E[v]\right)\end{aligned}$$

Corollary C.5. *Let $e \in \mathbf{Tm}$ be typeable and E an evaluation context of appropriate type. Then $\mathbf{Pr}(E[e] \Downarrow) = \sum_{\pi : e \rightsquigarrow^* v} \mathcal{W}(\pi) \cdot \mathbf{Pr}(E[v] \Downarrow)$.*

Corollary C.6. *For any term e and evaluation context E the equality*

$$\mathbf{Pr}(E[e] \Downarrow) = \sum_{v \in \mathbf{Val}} \mathcal{D}(e)(v) \cdot \mathbf{Pr}(E[v] \Downarrow)$$

holds.

Corollary C.7. *Let $e \in \mathbf{Tm}$ and E an evaluation context. Suppose $\mathcal{D}(e) = p \cdot \delta_v$ for some $v \in \mathbf{Val}$ and $p \in [0, 1]$. Then $\mathbf{Pr}(E[e] \Downarrow) = p \cdot \mathbf{Pr}(E[v] \Downarrow)$.*

Proof. Use Proposition **??** and Lemma **??**.

Proposition C.8. *For any evaluation context E and term e and any $k \in \mathbb{N}$,*

$$\mathbf{Pr}\left(E[e] \Downarrow^k\right) \leq \sum_{\pi : e \rightsquigarrow^* v} \mathcal{W}(\pi) \cdot \mathbf{Pr}\left(E[v] \Downarrow^k\right)$$

The proof proceeds by induction on k.

D Further examples

In this section we show further equivalences which did not fit into the paper proper due to space restrictions.

Fair coin from an unfair one Given an unfair coin, that is, a coin that comes up heads with probability p and tails with probability $1 - p$, where $0 < p < 1$ we can derive an infinite sequence of fair coin tosses using the procedure proposed by von Neumann. The procedure follows from the observation that if we toss an unfair coin twice, the likelihood of getting (H, T) is the same as the likelihood of getting (T, H). So the procedure works as follows

- Toss the coin twice
- If the result is (H, T) or (T, H) return the result of the first toss
- Else repeat the process

We only consider rational p in this section (for a computable p we could proceed similarly, but the details would be more involved, since the function which returns 1 with probability p and 0 with probability $1 - p$ is a bit more challenging to write).

Let $1 \leq k < n$ be two natural numbers and $p = \frac{k}{n}$. Below we define $e_p : \mathbf{1} \to \mathbf{2}$ to be the term implementing the von Neumann procedure for generating fair coin tosses from an unfair coin t_p which returns `true` with probability p and `false` with proability $1 - p$. We will show that e_p is contextually equivalent to $\lambda x.\texttt{true} \oplus \texttt{false}$. We define e_p as

$$e_p = \texttt{fix}[][]\varphi$$

where

$$\begin{aligned}
\mathbf{2} &= \mathbf{1} + \mathbf{1} \\
\texttt{true} &= \texttt{inl}\,\langle\rangle \\
\texttt{false} &= \texttt{inr}\,\langle\rangle \\
e \equiv e' &= \texttt{match}\,(e, _.e', _.\texttt{match}\,(e', _.\texttt{false}, _.\texttt{true})) \\
\texttt{if}\ e\ \texttt{then}\ e_1\ \texttt{else}\ e_2 &= \texttt{match}\,(e, _.e_1, _.e_2) \\
t_p &= \lambda\langle\rangle.\texttt{let}\ y\ = \texttt{rand}\,\underline{n}\ \texttt{in}\ (y \leq \underline{k})
\end{aligned}$$

and

$$\begin{aligned}
\varphi = \lambda f.\lambda\langle\rangle.\texttt{let}\ x\ &=\ t_p\,\langle\rangle\ \texttt{in} \\
\texttt{let}\ y\ &=\ t_p\,\langle\rangle\ \texttt{in} \\
\texttt{if}\ x \equiv y\ \texttt{then}\ f\,\langle\rangle\ \texttt{else}\ x&.
\end{aligned}$$

By a simple calculation using the operational semantics we can see that given any evaluation context E, we have $\mathbf{Pr}\,(E[t_p\,\langle\rangle] \Downarrow) = \frac{k}{n}\mathbf{Pr}\,(E[\texttt{true}] \Downarrow) + \frac{n-k}{n}\mathbf{Pr}\,(E[\texttt{false}] \Downarrow)$. Given any value f of type $(\mathbf{1} \to \mathbf{2})$ and any evaluation context E with the hole of type $\mathbf{2}$ we compute that $\mathbf{Pr}\,(E[\varphi\,f\,\langle\rangle] \Downarrow)$ is equal to $\frac{k^2+(n-k)^2}{n^2}\mathbf{Pr}\,(E[f\,\langle\rangle] \Downarrow) + 2 \cdot \frac{k\cdot(n-k)}{n^2}\mathbf{Pr}\,(E[\texttt{true} \oplus \texttt{false}] \Downarrow)$. Finally for e_p and any evaluation context E with hole of type $\mathbf{2}$ we have

$$\begin{aligned}
\mathbf{Pr}\,(E[e_p\,\langle\rangle] \Downarrow) = \mathbf{Pr}\,(\varphi\,e_p\,\langle\rangle \Downarrow) = \frac{k^2 + (n-k)^2}{n^2}&\mathbf{Pr}\,(E[e_p\,\langle\rangle] \Downarrow) \\
+ 2 \cdot \frac{k \cdot (n-k)}{n^2}&\mathbf{Pr}\,(E[\texttt{true} \oplus \texttt{false}] \Downarrow)\,.
\end{aligned}$$

from which we have by simple algebraic manipulation that $\mathbf{Pr}\,(E[e_p\,\langle\rangle] \Downarrow) = \mathbf{Pr}\,(E[\texttt{true} \oplus \texttt{false}] \Downarrow)$.

It is now straightforward to show $\varnothing \mid \varnothing \vdash e_p \cong^{log} \lambda\langle\rangle.\texttt{true} \oplus \texttt{false} : \mathbf{1} \to \mathbf{2}$ since both e_p and $\lambda\langle\rangle.\texttt{true} \oplus \texttt{false}$ are values, so we can show them related in the value relation. The proof uses reflexivity of $\cong^{log}$.

Alternatively, we could have used Theorem 1 and showed directly that $e_p\,\langle\rangle$ and $\texttt{true} \oplus \texttt{false}$ are CIU-equivalent and then used extensionality for values to conclude the proof.

A hesitant identity function We consider the identity function e that does not return immediately, but instead when applied to a value v flips a coin whether to return v or call itself recursively with the same argument. We show that this function is contextually equivalent to the identity function $\lambda x.x$. The reason for this is, intuitively, that even though e when applied may diverge, the probability of it doing so is 0.

Example D.1. Let $e = \texttt{fix}[][]\,(\lambda f.\lambda x.(x \oplus f\,x)) : \alpha \to \alpha$. We have

$$\alpha \mid \varnothing \vdash e \lesssim^{log} \lambda x.x : \alpha \to \alpha$$

and

$$\alpha \mid \varnothing \vdash \lambda x.x \lesssim^{log} e : \alpha \to \alpha.$$

Proof. We prove the two approximations separately. Let $\varphi \in \mathbf{VRel}(\alpha)$, $n \in \mathbb{N}$. Since e and $\lambda x.x$ are values we show them directly related in the value relation. In both cases let $\varphi = \lambda f.\lambda x.(x \oplus f\, x)$ and $h = \lambda z.\delta_\varphi\,(\mathtt{fold}\,\delta_\varphi)\,z$.

- By definition of the interpretation of function types we have to show, given $k \leq n$ and $(v, v') \in \varphi_r(\alpha)(k)$, that $(e\,v, (\lambda x.x)\,v') \in \varphi_r(\alpha)^{\top\top}(k)$.

  It is straightforward to see that $e\,v \overset{\text{cf}}{\Longrightarrow} \varphi\,e\,v$ using exactly one unfold-fold reduction.

  Now let (E, E') be related at k. We proceed by induction and show that for every $\ell \leq k$, $\mathbf{Pr}\left(E[e\,v] \Downarrow^{\ell}\right) \leq \mathbf{Pr}\left(E'[v'] \Downarrow\right)$ which suffices by Lemma 2. When $\ell = 0$ there is nothing to prove. So let $\ell = \ell' + 1$.

  $$\mathbf{Pr}\left(E[e\,v] \Downarrow^{\ell}\right) = \mathbf{Pr}\left(\varphi\,e\,v \Downarrow^{\ell'}\right) = \mathbf{Pr}\left(E[v \oplus e\,v] \Downarrow^{\ell'}\right).$$

  If $\ell' = 0$ we are trivially done. So suppose $\ell' = \ell'' + 1$ to get using Lemma 4

  $$\mathbf{Pr}\left(E[v \oplus e\,v] \Downarrow^{\ell'}\right) = \frac{1}{2}\mathbf{Pr}\left(E[v] \Downarrow^{\ell''}\right) + \frac{1}{2}\mathbf{Pr}\left(e\,v \Downarrow^{\ell''}\right)$$

  Using the fact that $\ell'' \leq k$ and monotonicity we have

  $$\mathbf{Pr}\left(E[v] \Downarrow^{\ell''}\right) \leq \mathbf{Pr}\left(E'[v'] \Downarrow\right).$$

  Using the induction hypothesis we have

  $$\mathbf{Pr}\left(e\,v \Downarrow^{\ell''}\right) \leq \mathbf{Pr}\left(E'[v'] \Downarrow\right)$$

  which together conclude the proof.
- Again by definition of the interpretation of function types we have to show, given $k \leq n$ and $(v, v') \in \varphi_r(\alpha)(k)$, that $((\lambda x.x)\,v', e\,v) \in \varphi_r(\alpha)^{\top\top}(k)$.

  Again we have that $e\,v' \overset{\text{cf}}{\Longrightarrow} \varphi\,e\,v'$ using exactly one unfold-fold reduction. Let $\ell \leq k$ and (E, E') related at ℓ. Using Lemma 2 and the fact that $\mathbf{Pr}(\cdot \Downarrow)$ is a fixed point of Φ we have

  $$\begin{aligned}\mathbf{Pr}\left(E'[e\,v'] \Downarrow\right) &= \mathbf{Pr}\left(E'[\varphi\,e\,v'] \Downarrow\right) \\ &= \frac{1}{2}\mathbf{Pr}\left(E'[v'] \Downarrow\right) + \frac{1}{2}\mathbf{Pr}\left(E'[e\,v'] \Downarrow\right)\end{aligned}$$

  and from this we get $\frac{1}{2}\mathbf{Pr}\left(E'[e\,v'] \Downarrow\right) = \frac{1}{2}\mathbf{Pr}\left(E'[v'] \Downarrow\right)$ by simple algebraic manipulation and thus $\mathbf{Pr}\left(E'[e\,v'] \Downarrow\right) = \mathbf{Pr}\left(E'[v'] \Downarrow\right)$. Using this property it is a triviality to finish the proof.

D.1 Further simple examples

The following example is a proof of *perfect security* for the one-time pad encryption scheme. Define the following functions

$$\begin{aligned}
&\texttt{not} : \mathbf{2} \to \mathbf{2}\\
&\texttt{not} = \lambda x.\texttt{if } x \texttt{ then false else true}\\
&\texttt{xor} : \mathbf{2} \to \mathbf{2} \to \mathbf{2}\\
&\texttt{xor} = \lambda x.\lambda y.\texttt{if } x \texttt{ then not } y \texttt{ else } y\\
&\texttt{gen} : \mathbf{2}\\
&\texttt{gen} = \texttt{true} \oplus \texttt{false}
\end{aligned}$$

`xor` is supposed to be the encryption function, with the first argument the plaintext and the second one the encryption key.

We now encode a game with two players. The first player chooses two plaintexts and gives them to the second player, who encrypts one of them (using `xor`) chosen at random with uniform probability and gives the result back to the first player. The first player should not be able to guess which of the plaintexts was encrypted. This is expressed as contextual equivalence of the following two programs

$$\begin{aligned}
\texttt{exp} &= \lambda x.\lambda y.\texttt{xor } (x \oplus y) \texttt{ gen}\\
\texttt{rnd} &= \lambda x.\lambda y.\texttt{gen}
\end{aligned}$$

To show $\texttt{exp} =^{ctx} \texttt{rnd}$ we first use extensionality for values so we only need to show that for all $v, u \in \mathbf{Val}\,(\mathbf{2})$

$$\texttt{xor } (v \oplus u) \texttt{ gen} =^{ctx} \texttt{gen}$$

and the easiest way to do this is by using CIU equivalence. Given an evaluation context E we have

$$\mathbf{Pr}\,(E[\texttt{xor } (v \oplus u) \texttt{ gen}] \Downarrow) = \frac{1}{4}\begin{pmatrix}\mathbf{Pr}\,(E[\texttt{xor } v \texttt{ true}] \Downarrow) +\\ \mathbf{Pr}\,(E[\texttt{xor } v \texttt{ false}] \Downarrow) +\\ \mathbf{Pr}\,(E[\texttt{xor } u \texttt{ true}] \Downarrow) +\\ \mathbf{Pr}\,(E[\texttt{xor } u \texttt{ false}] \Downarrow)\end{pmatrix}$$

and by the canonical forms lemma u and v can be either `true` or `false`. It is easy to see that the sum evaluates to

$$\frac{1}{4}(2 \cdot \mathbf{Pr}\,(E[\texttt{true}] \Downarrow) + 2 \cdot \mathbf{Pr}\,(E[\texttt{false}] \Downarrow))$$

quickly leading to the desired conclusion.

If we had used the logical relation directly we would not need the canonical forms lemma, but then we would have to take care of step-indexing.

A similar example is when in one instance we choose to encrypt the first plaintext and in the second instance the second one. Since the key is generated uniformly at random, the first player should not be able to distinguish those two instances. Concretely, this is expressed as contextual equivalence of the following two programs

$$\mathtt{exp}_1 = \lambda x.\lambda y.\mathtt{xor}\ x\ \mathtt{gen}$$
$$\mathtt{exp}_2 = \lambda x.\lambda y.\mathtt{xor}\ y\ \mathtt{gen}$$

The proof is basically the same as the one above. Use extensionality and then CIU equivalence.

D.2 Restrictions in the free theorem are necessary

We show that the free theorem in Section 5 does not hold without the assumptions on the behaviour of functions f and g.

First, if $f = (\lambda x.\underline{1}) \oplus (\lambda x.\underline{2})$, g is the identity function $\lambda x.x$ and xs is the list $[\langle\rangle, \langle\rangle]$ then the term $\mathtt{map}[][](f \circ g)\, xs$ can reduce to the list $[\underline{1}, \underline{2}]$, however the term $((\mathtt{map}[][]\, f) \circ (\mathtt{map}[][]\, g))\, xs$ cannot. The reason is that in the first case the reduction of f is performed for each element of the list separately, but in the latter case, f is first reduced to a value and then the same value is applied to all the elements of the list. Technically, the condition we need for f is that there exists a value f', such that $f =^{ctx} f'$, but this version is easily derived from the version stated above by congruence.

Second, if g diverges with a non-zero probability for some value v, we take m to be the constant function returning the empty list and the list xs to be the singleton list containing only the value v. Then, if f is any value, $m[][]\,(f \circ g)\, xs$ reduces to the empty list with probability 1, however $((m[][]f \circ \mathtt{map}[][]\, g))\, xs$ reduces to the empty list with a probability smaller than 1, since g is still applied, since we are in a *call-by-value* language.

Third, if $g = \lambda x.\underline{1} \oplus \underline{2}$, f is the identity function and xs is the singleton list containing $\langle\rangle$ we take m to be the function that first appends the given list to itself and *then* applies `map` to it. We then have that $m[][]\,(f \circ g)\, xs$ *can* reduce to the list $[\underline{1}, \underline{2}]$, but $((m[][]f) \circ (\mathtt{map}[][]\, g))\, xs$ cannot, since g is only mapped over the singleton list producing lists $[\underline{1}]$ and $[\underline{2}]$, which are then appended to themselves, giving lists $[\underline{1}, \underline{1}]$ and $[\underline{2}, \underline{2}]$.

And last, if m is not equivalent to a term of the form $\Lambda.\Lambda.\lambda x.e$ then the term on the left reduces to two different (not equivalent) values (or even diverges), but the term on the right does not. We can use this to construct a distinguishing evaluation context.

D.3 A property of `map`

The result in Section 5 does not allow us to conclude

$$\mathtt{map}[][]\,(f \circ g) =^{ctx} \mathtt{map}[][]\, f \circ \mathtt{map}[][]\, g.$$

for all $f \in \mathbf{Val}\,(\sigma \to \rho)$ and $g \in \mathbf{Val}\,(\tau \to \sigma)$, however we can show, using the definition of $\mathtt{map}$, that this does in fact hold. By using extensionality (Lemma 8) we need to show for any list xs we have

$$\mathtt{map}[][]\,(f \circ g)\,xs =^{ctx} (\mathtt{map}[][]\,f \circ \mathtt{map}[][]\,g)\;xs.$$

If f and g are values, E an evaluation context and xs a list of length n, it is easy to see that

$$\mathbf{Pr}\,(E[\mathtt{map}\,f\,xs] \Downarrow) = \sum_{us} \left(\prod_{i=1}^{n} \mathcal{D}(f\,x_i)(u_i) \right) \cdot \mathbf{Pr}\,(E[us] \Downarrow)$$

where the first sum is over all the lists of length n and x_i and u_i are the i-th elements of lists xs and us, respectively. This then gives us that

$$\mathbf{Pr}\,(E[\mathtt{map}\,f\,(\mathtt{map}\,g\,xs)] \Downarrow)$$

is equal to

$$\begin{aligned}
&\sum_{vs} \left(\prod_{i=1}^{n} \mathcal{D}(g\,x_i)(v_i) \right) \cdot \mathbf{Pr}\,(E[\mathtt{map}\,f\,vs] \Downarrow) \\
&= \sum_{vs} \left(\prod_{i=1}^{n} \mathcal{D}(g\,x_i)(v_i) \right) \cdot \left(\sum_{us} \left(\prod_{i=1}^{n} \mathcal{D}(f\,v_i)(u_i) \right) \cdot \mathbf{Pr}\,(E[us] \Downarrow) \right) \\
&= \sum_{vs} \sum_{us} \left(\prod_{i=1}^{n} \mathcal{D}(g\,x_i)(v_i) \cdot \mathcal{D}(f\,v_i)(u_i) \right) \cdot \mathbf{Pr}\,(E[us] \Downarrow).
\end{aligned}$$

On the other hand, we have that $\mathbf{Pr}\,(E[\mathtt{map}\,(f \circ g)\,xs] \Downarrow)$ is equal to

$$\sum_{us} \left(\prod_{i=1}^{n} \mathcal{D}((f \circ g)\,x_i)(u_i) \right) \cdot \mathbf{Pr}\,(E[us] \Downarrow)$$

and

$$\mathcal{D}((f \circ g)\,x_i)(u_i) = \sum_{v} \mathcal{D}(g\,x_i)(v) \cdot \mathcal{D}(f\,v)(u_i)$$

together giving us

$$\sum_{us} \left(\prod_{i=1}^{n} \left(\sum_{v} \mathcal{D}(g\,x_i)(v) \cdot \mathcal{D}(f\,v)(u_i) \right) \right) \cdot \mathbf{Pr}\,(E[us] \Downarrow)$$

which by Fubini's theorem and the fact that lists of length n correspond to n-tuples, is equal to

$$\sum_{us} \sum_{vs} \left(\prod_{i=1}^{n} (\mathcal{D}(g\,x_i)(v_i) \cdot \mathcal{D}(f\,v_i)(u_i)) \right) \cdot \mathbf{Pr}\,(E[us] \Downarrow)$$

which is the same as $\mathbf{Pr}\,(E[\mathtt{map}\,f\,(\mathtt{map}\,g\,xs)]\Downarrow)$.

If f and g are not equivalent to values, then the above result for `map` does not hold. Consider, for instance, $f = \lambda x.\underline{1} \oplus \lambda x.\underline{2}$ and g the identity or conversely, when applied to the list $xs = [\langle\rangle, \langle\rangle]$. The expression $\mathtt{map}[][]\,(f \circ g)\,xs$ can reduce to the list $[1, 2]$, whereas the expression $(\mathtt{map}[][]\,f \circ \mathtt{map}[][]\,g)\,xs$ cannot. We can generalize this to show that if f is not equivalent to a value or g is not, then the stated equality does not hold.

$$\frac{x{:}\tau \in \Gamma \qquad \Delta \vdash \Gamma}{\Delta \mid \Gamma \vdash x : \tau} \qquad \frac{\Delta \vdash \Gamma}{\Delta \mid \Gamma \vdash \langle\rangle : \mathbf{1}} \qquad \frac{\Delta \mid \Gamma \vdash e_1 : \tau_1 \qquad \Delta \mid \Gamma \vdash e_2 : \tau_2}{\Delta \mid \Gamma \vdash \langle e_1, e_2 \rangle : \tau_1 \times \tau_2}$$

$$\frac{\Delta \mid \Gamma, x{:}\tau_1 \vdash e : \tau_2}{\Delta \mid \Gamma \vdash \lambda x.e : \tau_1 \to \tau_2} \qquad \frac{\Delta \mid \Gamma \vdash e : \tau_1 \qquad \Delta \vdash \tau_2}{\Delta \mid \Gamma \vdash \mathtt{inl}\ e : \tau_1 + \tau_2} \qquad \frac{\Delta \mid \Gamma \vdash e : \tau_2 \qquad \Delta \vdash \tau_1}{\Delta \mid \Gamma \vdash \mathtt{inr}\ e : \tau_1 + \tau_2}$$

$$\frac{\Delta \mid \Gamma, x_1{:}\tau_1 \vdash e_1 : \tau \qquad \Delta \mid \Gamma, x_2{:}\tau_2 \vdash e_2 : \tau \qquad \Delta \mid \Gamma \vdash e : \tau_1 + \tau_2}{\Delta \mid \Gamma \vdash \mathtt{match}\,(e, x_1.e_1, x_2.e_2) : \tau}$$

$$\frac{\Delta, \alpha \mid \Gamma \vdash e : \tau}{\Delta \mid \Gamma \vdash \Lambda.e : \forall\alpha.\tau} \qquad \frac{\Delta \mid \Gamma \vdash e : \tau_1 \times \tau_2}{\Delta \mid \Gamma \vdash \mathtt{proj}_i\, e : \tau_i} \qquad \frac{\Delta \mid \Gamma \vdash e : \tau' \to \tau \qquad \Delta \mid \Gamma \vdash e' : \tau'}{\Delta \mid \Gamma \vdash e\, e' : \tau}$$

$$\frac{\Delta \vdash \tau_1 \qquad \Delta \mid \Gamma \vdash e : \tau[\tau_1/\alpha]}{\Delta \mid \Gamma \vdash \mathtt{pack}\, e : \exists\alpha.\tau}$$

$$\frac{\Delta \mid \Gamma \vdash e : \exists\alpha.\tau_1 \qquad \Delta \vdash \tau \qquad \Delta, \alpha \mid \Gamma, x : \tau_1 \vdash e' : \tau}{\Delta \mid \Gamma \vdash \mathtt{unpack}\ e\ \mathtt{as}\ x\ \mathtt{in}\ e' : \tau}$$

$$\frac{\Delta \mid \Gamma \vdash e : \mu\alpha.\tau}{\Delta \mid \Gamma \vdash \mathtt{unfold}\, e : \tau[\mu\alpha.\tau/\alpha]} \qquad \frac{\Delta \mid \Gamma \vdash e : \tau[\mu\alpha.\tau/\alpha]}{\Delta \mid \Gamma \vdash \mathtt{fold}\, e : \mu\alpha.\tau}$$

$$\frac{\Delta \mid \Gamma \vdash e : \forall\alpha.\tau \qquad \Delta \vdash \tau'}{\Delta \mid \Gamma \vdash e[] : \tau[\tau'/\alpha]} \qquad \frac{\Delta \mid \Gamma \vdash e : \mathtt{nat}}{\Delta \mid \Gamma \vdash \mathtt{rand}\ e : \mathtt{nat}}$$

$$\frac{\Delta \mid \Gamma \vdash e : \mathtt{nat} \qquad \Delta \mid \Gamma \vdash e_1 : \tau \qquad \Delta \mid \Gamma \vdash e_2 : \tau}{\Delta \mid \Gamma \vdash \mathtt{if}_1\ e\ \mathtt{then}\ e_1\ \mathtt{else}\ e_2 : \tau} \qquad \frac{\Delta \mid \Gamma \vdash e : \mathtt{nat}}{\Delta \mid \Gamma \vdash \mathtt{P}\, e : \mathtt{nat}}$$

$$\frac{\Delta \mid \Gamma \vdash e : \mathtt{nat}}{\Delta \mid \Gamma \vdash \mathtt{S}\, e : \mathtt{nat}}$$

Fig. 4. Typing of terms, where $\Gamma ::= \emptyset \mid \Gamma, x{:}\tau$ and $\Delta ::= \emptyset \mid \Delta, \alpha$.

Basic reductions $\stackrel{\cdot}{\longmapsto}$

$$\mathtt{proj}_i\,\langle v_1, v_2\rangle \stackrel{1}{\longmapsto} v_i \qquad\qquad \mathtt{unfold}\,(\mathtt{fold}\,v) \stackrel{1}{\longmapsto} v$$

$$(\lambda x.e)\,v \stackrel{1}{\longmapsto} e[v/x] \qquad\qquad \mathtt{unpack}\ (\mathtt{pack}\,v)\ \mathtt{as}\ x\ \mathtt{in}\ e \stackrel{1}{\longmapsto} e[v/x]$$

$$(\Lambda.e)[] \stackrel{1}{\longmapsto} e \qquad\qquad \mathtt{match}\,(\mathtt{inl}\,v, x_1.e_1, x_2.e_2) \stackrel{1}{\longmapsto} e_1[v/x_1]$$

$$\mathtt{rand}\,\underline{n} \stackrel{\frac{1}{n}}{\longmapsto} \underline{k} \quad (k \in \{1, 2, \ldots, n\}) \qquad \mathtt{match}\,(\mathtt{inr}\,v, x_1.e_1, x_2.e_2) \stackrel{1}{\longmapsto} e_2[v/x_2]$$

$$\mathtt{P}\,\underline{n} \stackrel{1}{\longmapsto} \underline{\max\{n-1, 1\}} \qquad\qquad \mathtt{S}\,\underline{n} \stackrel{1}{\longmapsto} \underline{n+1}$$

$$\mathtt{if}_1\ \underline{1}\ \mathtt{then}\ e_1\ \mathtt{else}\ e_2 \stackrel{1}{\longmapsto} e_1 \qquad\qquad \mathtt{if}_1\ \mathtt{S}\,\underline{n}\ \mathtt{then}\ e_1\ \mathtt{else}\ e_2 \stackrel{1}{\longmapsto} e_2$$

One step reduction relation $\stackrel{\cdot}{\rightsquigarrow}$

$$E[e] \stackrel{p}{\rightsquigarrow} E[e'] \qquad \text{if } e \stackrel{p}{\longmapsto} e'$$

Fig. 5. Operational semantics.

$$\frac{x{:}\tau \in \Gamma}{\Delta \mid \Gamma \vdash x \mathrel{\mathcal{R}} x : \tau} \qquad \frac{}{\Delta \mid \Gamma \vdash \langle\rangle \mathrel{\mathcal{R}} \langle\rangle : \mathbf{1}}$$

$$\frac{\Delta \mid \Gamma \vdash e_1 \mathrel{\mathcal{R}} e_1' : \tau_1 \qquad \Delta \mid \Gamma \vdash e_2 \mathrel{\mathcal{R}} e_2' : \tau_2}{\Delta \mid \Gamma \vdash \langle e_1, e_2\rangle \mathrel{\mathcal{R}} \langle e_1', e_2'\rangle : \tau_1 \times \tau_2} \qquad \frac{\Delta \mid \Gamma, x{:}\tau_1 \vdash e \mathrel{\mathcal{R}} e' : \tau_2}{\Delta \mid \Gamma \vdash \lambda x.e \mathrel{\mathcal{R}} \lambda x.e' : \tau_1 \to \tau_2}$$

$$\frac{\Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau_1}{\Delta \mid \Gamma \vdash \mathtt{inl}\, e \mathrel{\mathcal{R}} \mathtt{inl}\, e' : \tau_1 + \tau_2} \qquad \frac{\Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau_2}{\Delta \mid \Gamma \vdash \mathtt{inr}\, e \mathrel{\mathcal{R}} \mathtt{inr}\, e' : \tau_1 + \tau_2}$$

$$\frac{\Delta \mid \Gamma, x_1{:}\tau_1 \vdash e_1 \mathrel{\mathcal{R}} e_1' : \tau \qquad \Delta \mid \Gamma, x_2{:}\tau_2 \vdash e_2 \mathrel{\mathcal{R}} e_2' : \tau \qquad \Delta \mid \Gamma \vdash e\, e' {:}\, \tau_1 + \tau_2}{\Delta \mid \Gamma \vdash \mathtt{match}\,(e, x_1.e_1, x_2.e_2) \mathrel{\mathcal{R}} \mathtt{match}\,\big(e', x_1.e_1', x_2.e_2'\big) : \tau}$$

$$\frac{\Delta, \alpha \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau}{\Delta \mid \Gamma \vdash \Lambda.e \mathrel{\mathcal{R}} \Lambda.e' : \forall\alpha.\tau} \qquad \frac{\Delta \vdash \tau_1 \qquad \Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau[\tau_1/\alpha]}{\Delta \mid \Gamma \vdash (\mathtt{pack}\, e) \mathrel{\mathcal{R}} (\mathtt{pack}\, e') : \exists\alpha.\tau}$$

$$\frac{\Delta \mid \Gamma \vdash e_1 \mathrel{\mathcal{R}} e_1' : \exists\alpha.\tau_1 \qquad \Delta \vdash \tau \qquad \Delta, \alpha \mid \Gamma, x : \tau_1 \vdash e \mathrel{\mathcal{R}} e' : \tau}{\Delta \mid \Gamma \vdash (\mathtt{unpack}\ e_1\ \mathtt{as}\ x\ \mathtt{in}\ e) \mathrel{\mathcal{R}} (\mathtt{unpack}\ e_1'\ \mathtt{as}\ x\ \mathtt{in}\ e') : \tau}$$

$$\frac{\Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau_1 \times \tau_2}{\Delta \mid \Gamma \vdash \mathtt{proj}_i\, e \mathrel{\mathcal{R}} \mathtt{proj}_i\, e' : \tau_i} \qquad \frac{\Delta \mid \Gamma \vdash e_1 \mathrel{\mathcal{R}} e_1' : \tau' \to \tau \qquad \Delta \mid \Gamma \vdash e_2 \mathrel{\mathcal{R}} e_2' : \tau'}{\Delta \mid \Gamma \vdash e_1\, e_2 \mathrel{\mathcal{R}} e_1'\, e_2' : \tau}$$

$$\frac{\Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \mu\alpha.\tau}{\Delta \mid \Gamma \vdash \mathtt{unfold}\, e \mathrel{\mathcal{R}} \mathtt{unfold}\, e' : \tau[\mu\alpha.\tau/\alpha]} \qquad \frac{\Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau[\mu\alpha.\tau/\alpha]}{\Delta \mid \Gamma \vdash \mathtt{fold}\, e \mathrel{\mathcal{R}} \mathtt{fold}\, e' : \mu\alpha.\tau}$$

$$\frac{\Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \forall\alpha.\tau}{\Delta \mid \Gamma \vdash e[] \mathrel{\mathcal{R}} e'[] : \tau[\tau'/\alpha]}\ \mathit{ftv}(\tau') \subseteq \Delta \qquad \frac{\Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \mathtt{nat}}{\Delta \mid \Gamma \vdash \mathtt{rand}\, e \mathrel{\mathcal{R}} \mathtt{rand}\, e' : \mathtt{nat}}$$

$$\frac{\Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \mathtt{nat}}{\Delta \mid \Gamma \vdash \mathtt{P}\, e \mathrel{\mathcal{R}} \mathtt{P}\, e' : \mathtt{nat}} \qquad \frac{\Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \mathtt{nat}}{\Delta \mid \Gamma \vdash \mathtt{S}\, e \mathrel{\mathcal{R}} \mathtt{S}\, e' : \mathtt{nat}}$$

$$\frac{\Delta \mid \Gamma \vdash e \mathrel{\mathcal{R}} e' : \mathtt{nat} \qquad \Delta \mid \Gamma, \vdash e_1 \mathrel{\mathcal{R}} e_1' : \tau \qquad \Delta \mid \Gamma, \vdash e_2 \mathrel{\mathcal{R}} e_2' : \tau}{\Delta \mid \Gamma \vdash \mathtt{if}_1\ e\ \mathtt{then}\ e_1\ \mathtt{else}\ e_2 \mathrel{\mathcal{R}} \mathtt{if}_1\ e'\ \mathtt{then}\ e_1'\ \mathtt{else}\ e_2' : \tau}$$

Fig. 6. Compatibility properties of type-indexed relations

$$
\begin{aligned}
[\![\Delta \vdash \mathtt{nat}]\!](\varphi)(n) &= \{(\underline{k},\underline{k}) \mid k \in \mathbb{N}, k > 0\} \\
[\![\Delta \vdash \tau \times \sigma]\!](\varphi)(n) &= \left\{ \left(\langle v,u\rangle, \langle v',u'\rangle\right) \;\middle|\; \begin{array}{l} (v,v') \in [\![\Delta \vdash \tau]\!](\varphi)(n), \\ (u,u') \in [\![\Delta \vdash \sigma]\!](\varphi)(n) \end{array} \right\} \\
[\![\Delta \vdash \tau + \sigma]\!](\varphi)(n) &= \left\{ \left(\mathtt{inl}\ v, \mathtt{inl}\ v'\right) \mid (v,v') \in [\![\Delta \vdash \tau]\!](\varphi)(n) \right\} \\
&\quad \cup \left\{ \left(\mathtt{inr}\ v, \mathtt{inr}\ v'\right) \mid (v,v') \in [\![\Delta \vdash \sigma]\!](\varphi)(n) \right\} \\
[\![\Delta \vdash \tau \to \sigma]\!](\varphi)(n) &= \left\{ \left(\lambda x.e, \lambda y.e'\right) \;\middle|\; \begin{array}{l} \forall j \leq n, \forall (v,v') \in [\![\Delta \vdash \tau]\!](\varphi)(j), \\ ((\lambda x.e)\, v, (\lambda y.e')\, v') \in [\![\Delta \vdash \sigma]\!](\varphi)^{\top\top}(j) \end{array} \right\} \\
[\![\Delta \vdash \forall \alpha.\tau]\!](\varphi)(n) &= \left\{ \left(\Lambda.e, \Lambda.e'\right) \;\middle|\; \begin{array}{l} \forall \sigma,\sigma' \in \mathfrak{T}, \forall r \in \mathbf{VRel}\,(\sigma,\sigma'), \\ (e,e') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi\,[\alpha \mapsto r])^{\top\top}(n) \end{array} \right\} \\
[\![\Delta \vdash \exists \alpha.\tau]\!](\varphi)(n) &= \left\{ \left(\mathtt{pack}\, v, \mathtt{pack}\, v'\right) \;\middle|\; \begin{array}{l} \exists \sigma,\sigma' \in \mathfrak{T}, \exists r \in \mathbf{VRel}\,(\sigma,\sigma'), \\ (v,v') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi\,[\alpha \mapsto r])\,(n) \end{array} \right\} \\
[\![\Delta \vdash \mu\alpha.\tau]\!](\varphi)(0) &= \mathbf{Val}\,(\varphi_1(\mu\alpha.\tau)) \times \mathbf{Val}\,(\varphi_2(\mu\alpha.\tau)) \\
[\![\Delta \vdash \mu\alpha.\tau]\!](\varphi)(n+1) &= \left\{ \left(\mathtt{fold}\, v, \mathtt{fold}\, v'\right) \mid (v,v') \in [\![\Delta, \alpha \vdash \tau]\!](\varphi\,[\alpha \mapsto [\![\Delta \vdash \mu\alpha.\tau]\!](\varphi)])\,(n) \right\}
\end{aligned}
$$

Fig. 7. Interpretation of types.

$$
e \oplus e =^{ctx} e \qquad e_1 \oplus e_2 =^{ctx} e_2 \oplus e_1 \qquad e \oplus \Omega \lesssim^{ctx} e
$$

$$
\text{if } e_1 \stackrel{\text{cf}}{\Longrightarrow} e_2 \text{ then } e_1 =^{ctx} e_2 \qquad \text{if } e_1 \oplus e_2 =^{ctx} e_1 \text{ then } e_1 =^{ctx} e_2
$$

Fig. 8. Basic properties of $\lesssim^{ctx}$ and $=^{ctx}$. We write Ω for any diverging term (i.e. $\mathbf{Pr}\,(\Omega \Downarrow) = 0$) and $e \oplus e'$ as syntactic sugar for $\mathtt{if}_1\ \mathtt{rand}\,\underline{2}\ \mathtt{then}\ e\ \mathtt{else}\ e'$. Note that the choice when evaluating $e \oplus e'$ is made *before* e and e' are evaluated.