STEP-INDEXED RELATIONAL REASONING FOR COUNTABLE NONDETERMINISM

JAN SCHWINGHAMMER AND LARS BIRKEDAL

Saarland University
e-mail address: jan@ps.uni-saarland.de

IT University of Copenhagen
e-mail address: birkedal@itu.dk

Abstract. Programming languages with countable nondeterministic choice are computationally interesting since countable nondeterminism arises when modeling fairness for concurrent systems. Because countable choice introduces non-continuous behaviour, it is well-known that developing semantic models for programming languages with countable nondeterminism is challenging. We present a step-indexed logical relations model of a higher-order functional programming language with countable nondeterminism and demonstrate how it can be used to reason about contextually defined may- and must-equivalence. In earlier step-indexed models, the indices have been drawn from ω. Here the step-indexed relations for must-equivalence are indexed over an ordinal greater than ω. Finally, we define step-indexed logical relations for showing the adequacy of a continuation-passing-style transformation of the language.

1. Introduction

Programming languages with countable nondeterministic choice are computationally interesting since countable nondeterminism arises when modeling fairness for concurrent systems. In this paper we show how to construct simple semantic models for reasoning about may- and must-equivalence in a call-by-value higher-order functional programming language with countable nondeterminism, recursive types and impredicative polymorphism.

Models for languages with nondeterminism have originally been studied using denotational techniques. In the case of countably branching nondeterminism it is not enough to consider standard ω-continuous complete partial orders and the denotational models become quite involved [3, 6]. This has sparked research in operationally-based theories of equivalence for nondeterministic higher-order languages [1, 10, 11, 12, 13, 19]. In particular, Lassen investigated operationally-based relational methods for countable nondeterminism

1998 ACM Subject Classification: F.3.2 Semantics of Programming Languages.

Key words and phrases: Countable choice, lambda calculus, program equivalence.

A preliminary version of this work has been presented at the 20th EACSL Annual Conference on Computer Science Logic (CSL'11), 12-15 September 2011, Bergen, Norway. In addition to including more proofs, the present version also includes a section with a new application of step-indexed relational reasoning, for showing the adequacy of a cps-transformation for our language with countable nondeterminism.

LOGICAL METHODS
IN COMPUTER SCIENCE

DOI:10.2168/LMCS-???

and suggested that it would be interesting to consider also methods based on logical relations, i.e., where the *types* of the programming languages are given a relational interpretation [10, page 47]. Such an interpretation would allow one to relate terms of different types, as needed for reasoning about parametricity properties of polymorphic types.

For languages with recursive types, however, logical relations cannot be defined by induction on types. In the case of deterministic languages, this problem has been addressed by the technique of syntactic minimal invariance [4] (inspired by domain theory [15]). The idea here is that one proves that a syntactically definable fixed point on a recursive type is contextually equivalent to the identity function, and then uses a so-called unwinding theorem for syntactically definable fixed points when showing the existence of the logical relations. However, in the presence of countable nondeterminism it is not clear how to define the unwindings of the syntactic fixed point in the programming language. Indeed, Lassen proved an unwinding theorem for his language with countable nondeterminism, but he did so by extending the language with new terms needed for representing the unwindings and left open the question of whether this is a conservative extension of the language.

Here we give a logical relations model of our language where we do not rely on syntactic minimal invariance for constructing the logical relations. Instead, we use the idea of step-indexed logical relations [2]. In particular, we show how to use step-indexing over ordinals larger than ω to reason about must-equivalence in the presence of countable nondeterminism.

This approach turns out to be both simple and also useful for reasoning about concrete may- and must-equivalences. We show that our logical relations are sound and complete with respect to the contextually defined notions of may- and must-equivalence. Moreover, we show how to use our logical relations to establish some concrete equivalences. In particular, we prove the recursion-induction rule from Lassen [10] and establish the syntactic minimal invariance property (without extending the language with new unwinding terms). We also include an example to show that the model can be used to prove parametricity properties (free theorems) of polymorphic types.

As another application we show how to define and use step-indexed logical relations for showing the adequacy of a continutation-passing-style (cps) transformation for our language with countable nondeterminism. Specifically, we define a cps translation for the language and show that if the cps-transforms of two expressions are may- and must-contextually equivalent, then also the original expressions are may- and must-equivalent.

Overview of the technical development. One way to understand the failure of ω-continuity in an operational setting is to consider the must-convergence predicate $e \Downarrow$, which by Tarski's fixed point theorem can be defined as the least fixed point of the monotone functional $\Phi(R) = \{e \mid \forall e'.\ e \longmapsto e' \ \Rightarrow\ e' \in R\}$ on sets of terms. Here $e \longmapsto e'$ means that e reduces to e' in one step. However, due to the countable branching the fixed point is not reached by ω-many iterations $\bigcup_{n\in\omega} \Phi^n(\emptyset)$. The reason is that even when a program has no infinite reduction sequences, we cannot in general bound the length of reduction sequences by any $n < \omega$.

The idea of step-indexed semantics is a stratified construction of relations which facilitates the interpretation of recursive types, and in previous applications this stratification has typically been realized by indexing over ω. However, as we pointed out, the closure ordinal of the inductively defined must-convergence predicate is strictly larger than ω: the least fixed point $\Downarrow$ is reached after ω_1-many iterations, for ω_1 the least uncountable ordinal. (In fact, the least non-recursive ordinal would suffice [3].) Thus, one of the key steps in our

$$\tau ::= \alpha \mid \mathbf{1} \mid \tau_1 \times \tau_2 \mid \tau_1 \to \tau_2 \mid \mu\alpha.\tau_1 + \ldots + \tau_n \mid \forall\alpha.\tau$$
$$v ::= x \mid \langle\rangle \mid \langle v_1, v_2\rangle \mid \lambda x.e \mid \mathtt{in}_i\, v \mid \Lambda\alpha.e$$
$$e ::= v \mid \mathtt{?} \mid \mathtt{proj}_i\, v \mid v\, e \mid \mathtt{case}\ v\ \mathtt{of}\ \mathtt{in}_1\, x_1.\, e_1 \mid \ldots \mid \mathtt{in}_n\, x_n.\, e_n \mid v\,\tau$$
$$E ::= [\,] \mid v\, E$$

Figure 1: Types, terms and evaluation contexts

development is the definition of α-indexed uniform relations, for arbitrary ordinals α, in Section 3.

In Section 4 we define a logical ω-indexed uniform relation, and use this relation to prove a CIU theorem for may-contextual equivalence. The logical relation combines step-indexing and biorthogonality, and we can prove that it coincides with may-contextual equivalence; the proofs are similar to those in [17]. Section 5 considers the case of must-contextual equivalence. The only modifications that this requires, compared to Section 4, are the use of ω_1-indexed uniform relations and of a suitably adapted notion of biorthogonality.

In Section 7 we define the cps-transformation and show adequacy of it by means of step-indexed logical relations; for the must-equivalence we use indexing over ω_1.

Summary of contributions. In summary, the contribution of this paper is a simple, operationally-based model of countable nondeterminism in a higher-order language, and the use of this model for proving several non-trivial applications in Section 6. In particular, we derive a least-fixed point property for recursive functions in our language, answering a question raised by Lassen [10]. Moreover, we show adequacy of a cps-transformation for the language; this appears to be the first such result for a language with countable nondeterminism.

Laird [9] has developed a fully abstract denotational model based on bidomains for a calculus similar to the one studied here but without recursive and polymorphic types; our model appears to be the first model of countable nondeterminism for a language with impredicative polymorphism.

2. A lambda calculus with countable choice

Syntax and operational semantics. Figure 1 gives the syntax of a higher-order functional language with recursive and polymorphic types, and a (countably branching) choice construct. We assume disjoint, countably infinite sets of *type variables*, ranged over by α, and *term variables*, ranged over by x. The free type variables of types and terms, $ftv(\tau)$ and $ftv(e)$, and free term variables $fv(e)$, are defined in the usual way. The notation $(\cdot)[\vec{\tau}/\vec{\alpha}]$ denotes the simultaneous capture-avoiding substitution of types $\vec{\tau}$ for the free type variables $\vec{\alpha}$ in types and terms; similarly, $e[\vec{v}/\vec{x}]$ denotes simultaneous capture-avoiding substitution of values $\vec{v}$ for the free term variables $\vec{x}$ in e.

The syntax is kept minimal, and in examples we may use additional syntactic sugar, for instance writing $\mathtt{let}\ x = e\ \mathtt{in}\ e'$ for $(\lambda x.e')\, e$ and $e\,\tau$ for $\mathtt{let}\ f = e\ \mathtt{in}\ f\,\tau$ for some fresh f. We define the unary natural numbers datatype as $\mathtt{nat} = \mu\alpha.\mathbf{1} + \alpha$ and write $\underline{0} = \mathtt{in}_1\,\langle\rangle$ and $\underline{n+1} = \mathtt{in}_2(\underline{n})$. The 'erratic' (finitely branching) choice construct $e_1\ \mathtt{or}\ e_2$ can be defined from $\mathtt{?}$ as $\mathtt{let}\ x = \mathtt{?}\ \mathtt{in}\ \mathtt{case}\ x\ \mathtt{of}\ \mathtt{in}_1\, y.\, e_1 \mid \mathtt{in}_2\, y.\, e_2$ for fresh x, y.

$$\begin{array}{ll} \mathtt{proj}_i\,\langle v_1, v_2\rangle \longmapsto v_i & \mathtt{case}\ (\mathtt{in}_j\, v)\ \mathtt{of}\ (\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots) \longmapsto e_j[v/x_j] \\ (\lambda x.e)\, v \longmapsto e[v/x] & \mathtt{?} \longmapsto \underline{n} \qquad (n \in \mathbb{N}) \\ (\Lambda\alpha.e)\,\tau \longmapsto e[\tau/\alpha] & v\, e \longmapsto v\, e' \qquad \text{if } e \longmapsto e' \end{array}$$

Figure 2: Operational semantics

$$\frac{x{:}\tau \in \Gamma \qquad \Delta \vdash \Gamma}{\Delta;\Gamma \vdash x : \tau} \qquad \frac{\Delta \vdash \Gamma}{\Delta;\Gamma \vdash \langle\rangle : \mathbf{1}} \qquad \frac{\Delta;\Gamma \vdash v_1 : \tau_1 \qquad \Delta;\Gamma \vdash v_2 : \tau_2}{\Delta;\Gamma \vdash \langle v_1, v_2\rangle : \tau_1 \times \tau_2}$$

$$\frac{\Delta;\Gamma, x{:}\tau_1 \vdash e : \tau_2}{\Delta;\Gamma \vdash \lambda x.e : \tau_1 \to \tau_2} \qquad \frac{\Delta;\Gamma \vdash v : \tau_j[\mu\alpha.\tau_1 + \ldots + \tau_n/\alpha]}{\Delta;\Gamma \vdash \mathtt{in}_j\, v : \mu\alpha.\tau_1 + \ldots + \tau_n}\ 1 \le j \le n$$

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

$$\frac{\Delta;\Gamma \vdash v : \mu\alpha.\tau_1 + \ldots + \tau_n \qquad \ldots \qquad \Delta;\Gamma, x_j{:}\tau_j[\mu\alpha.\tau_1 + \ldots + \tau_n/\alpha] \vdash e_j : \tau \qquad \ldots}{\Delta;\Gamma \vdash \mathtt{case}\ v\ \mathtt{of}\ (\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots) : \tau}$$

$$\frac{\Delta;\Gamma \vdash v : \forall\alpha.\tau \qquad \Delta \vdash \tau'}{\Delta;\Gamma \vdash v\,\tau' : \tau[\tau'/\alpha]} \qquad \frac{\Delta \vdash \Gamma}{\Delta;\Gamma \vdash \mathtt{?} : \mathtt{nat}}$$

$$\frac{\varnothing \vdash \tau}{\vdash [] : \tau \multimap \tau} \qquad \frac{\varnothing;\varnothing \vdash v : \tau \to \tau_2 \qquad \vdash E : \tau_1 \multimap \tau}{\vdash v\, E : \tau_1 \multimap \tau_2}$$

Figure 3: Typing of terms and evaluation contexts, where $\Gamma ::= \emptyset \mid \Gamma, x{:}\tau$ and $\Delta ::= \emptyset \mid \Delta, \alpha$. The notation $\Delta \vdash \tau$ means that $\mathit{ftv}(\tau) \subseteq \Delta$, and $\Delta \vdash \Gamma$ means that $\Delta \vdash \tau$ holds for all $x{:}\tau \in \Gamma$.

The operational semantics of the language is given in Figure 2 by a reduction relation $e \longmapsto e'$. In particular, the choice operator ? evaluates nondeterministically to any numeral $\underline{n}$ ($n \in \mathbb{N}$). We also consider evaluation contexts E, and write $E[e]$ for the term obtained by plugging e into E. It is easy to see that $e \longmapsto e'$ holds if and only if $E[e] \longmapsto E[e']$.

Typing judgements take the form $\Delta;\Gamma \vdash e : \tau$ where Γ is a typing context $x_1{:}\tau_1, \ldots, x_n{:}\tau_n$ and where Δ is a finite set of type variables that contains the free type variables of $\tau_1, \ldots, \tau_n$ and τ. The rules defining this judgement are summarized in Figure 3. The typing judgement for evaluation contexts, $\vdash E : \tau \multimap \tau'$, means that $\varnothing;\varnothing \vdash E[e] : \tau'$ holds whenever $\varnothing;\varnothing \vdash e : \tau$.

We write *Type* for the set of closed types τ, i.e., where $\mathit{ftv}(\tau) = \emptyset$. We write $\mathit{Val}(\tau)$ and $\mathit{Tm}(\tau)$ for the sets of closed values and terms of type τ, resp., and $\mathit{Stk}(\tau)$ for the set of τ-accepting evaluation contexts. For a typing context $\Gamma = x_1{:}\tau_1, \ldots, x_n{:}\tau_n$ with $\tau_1, \ldots, \tau_n \in \mathit{Type}$, let $\mathit{Subst}(\Gamma) = \{\gamma \in \mathit{Val}^{\vec{x}} \mid \forall 1 \le i \le n.\ \gamma(x_i) \in \mathit{Val}(\tau_i)\}$ denote the set of type-respecting value substitutions. In particular, if $\Delta;\Gamma \vdash e : \tau$ then $\varnothing;\varnothing \vdash e\delta\gamma : \tau\delta$ for any $\delta \in \mathit{Type}^{\Delta}$ and $\gamma \in \mathit{Subst}(\Gamma\delta)$, and the type system satisfies the standard progress and preservation theorems.

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

$$\frac{\Delta;\Gamma \vdash v_1 \mathrel{\mathcal{R}} v_1' : \tau_1 \qquad \Delta;\Gamma \vdash v_2 \mathrel{\mathcal{R}} v_2' : \tau_2}{\Delta;\Gamma \vdash \langle v_1, v_2\rangle \mathrel{\mathcal{R}} \langle v_1', v_2'\rangle : \tau_1 \times \tau_2} \qquad \frac{\Delta;\Gamma, x{:}\tau_1 \vdash e \mathrel{\mathcal{R}} e' : \tau_2}{\Delta;\Gamma \vdash \lambda x.e \mathrel{\mathcal{R}} \lambda x.e' : \tau_1 \to \tau_2}$$

$$\frac{\Delta;\Gamma \vdash v \mathrel{\mathcal{R}} v' : \tau_j[\mu\alpha.\tau_1 + \ldots + \tau_n/\alpha]}{\Delta;\Gamma \vdash \mathtt{in}_j\, v \mathrel{\mathcal{R}} \mathtt{in}_j\, v' : \mu\alpha.\tau_1 + \ldots + \tau_n}\ 1 \le j \le n \qquad \frac{\Delta,\alpha;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau}{\Delta;\Gamma \vdash \Lambda\alpha.e \mathrel{\mathcal{R}} \Lambda\alpha.e' : \forall\alpha.\tau}$$

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

$$\frac{\Delta;\Gamma \vdash v \mathrel{\mathcal{R}} v' : \tau \qquad \ldots \qquad \Delta;\Gamma, x_j{:}\tau_j[\tau/\alpha] \vdash e_j \mathrel{\mathcal{R}} e_j' : \tau' \qquad \ldots}{\Delta;\Gamma \vdash \mathtt{case}\ v\ \mathtt{of}\,(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots) \mathrel{\mathcal{R}} \mathtt{case}\ v'\ \mathtt{of}\,(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots) : \tau'}\ \tau = \mu\alpha.\tau_1 + \ldots + \tau_n$$

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

Figure 4: Compatibility properties of type-indexed relations

We let $\mathtt{fix} : \forall\alpha,\beta.((\alpha \to \beta) \to (\alpha \to \beta)) \to (\alpha \to \beta)$ denote a variant of the (call-by-value) fixed point combinator from untyped lambda calculus, $\mathtt{fix} = \Lambda\alpha,\beta.\lambda f.\delta_f(\mathtt{in}\,\delta_f)$ where δ_f is the term $\lambda y.\mathtt{case}\ y\ \mathtt{of}\ \mathtt{in}\, y'.\, f(\lambda x.\mathtt{let}\ r = y'\, y\ \mathtt{in}\ r\, x)$, and we write $\Omega : \forall\alpha.\alpha$ for the term $\Lambda\alpha.\mathtt{fix}\,\mathbf{1}\,\alpha\,(\lambda f.f)\,\langle\rangle$. Note that reduction from Ω is deterministic and non-terminating.

Contextual approximation. We follow Lassen's approach [10] and define contextual approximation as the largest relation that satisfies certain compatibility and adequacy properties (also see, e.g. [16, 17]). The technical advantage of this approach, compared to the more traditional one of universally quantifying over program contexts, is that in proofs there will be no need to explicitly take care of contexts and of term occurrences within contexts. In our terminology, we keep close to Pitts [16], except for suitably adapting the definitions to take the nondeterministic outcomes of evaluation into account.

The observables on which contextual approximation is based are given by may- and must-convergence. A closed term e *may-converges*, written $e\downarrow$, if $e \longmapsto^* v$ for some $v \in \mathit{Val}$, and e *may-diverges*, written $e\uparrow$, if there is an infinite reduction sequence starting from e. The *must-convergence* predicate $e \Downarrow$ is the complement of may-divergence, and it can be defined inductively by $e \Downarrow$ if and only if for all e', if $e \longmapsto e'$ then $e' \Downarrow$.

Definition 1 (Type-indexed relation)**.** *A* type-indexed relation *is a set of tuples* $(\Delta,\Gamma,e,e',\tau)$ *such that* $\Delta;\Gamma \vdash e : \tau$ *and* $\Delta;\Gamma \vdash e' : \tau$ *holds. We write* $\Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau$ *if* $(\Delta,\Gamma,e,e',\tau) \in \mathcal{R}$*.*

Definition 2 (Precongruence)**.** *A type-indexed relation* $\mathcal{R}$ *is* reflexive *if* $\Delta;\Gamma \vdash e : \tau$ *implies* $\Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e : \tau$*. It is* transitive *if* $\Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau$ *and* $\Delta;\Gamma \vdash e' \mathrel{\mathcal{R}} e'' : \tau$ *implies* $\Delta;\Gamma \vdash e \mathrel{\mathcal{R}} e'' : \tau$*. A* precongruence *is a reflexive and transitive type-indexed relation* $\mathcal{R}$ *that is closed under the inference rules in Figure 4.*

$$\texttt{let}\ x = \texttt{?}\ \texttt{in}\ e \cong^{ctx} e \quad (x \notin fv(e)) \qquad \texttt{let}\ x = v\ \texttt{in}\ e \cong^{ctx} e[v/x] \qquad \texttt{let}\ x = e\ \texttt{in}\ x \cong^{ctx} e$$

$$e\ \texttt{or}\ e \cong^{ctx} e \qquad \Omega \lesssim^{ctx}_{\downarrow} e \qquad \Omega \lesssim^{ctx}_{\Downarrow} e$$

$$e_1\ \texttt{or}\ e_2 \cong^{ctx} e_2\ \texttt{or}\ e_1 \qquad e_1 \lesssim^{ctx}_{\downarrow} e_1\ \texttt{or}\ e_2 \qquad e_1\ \texttt{or}\ e_2 \lesssim^{ctx}_{\Downarrow} e_1$$

$$(e_1\ \texttt{or}\ e_2)\ \texttt{or}\ e_3 \cong^{ctx} e_1\ \texttt{or}\ (e_2\ \texttt{or}\ e_3) \qquad e\ \texttt{or}\ \Omega \cong^{ctx}_{\downarrow} e \qquad e\ \texttt{or}\ \Omega \cong^{ctx}_{\Downarrow} \Omega$$

Figure 5: Basic may- and must-theory, where $e_1\ \texttt{or}\ e_2$ is an abbreviation for the term $\texttt{let}\ x = \texttt{?}\ \texttt{in}\ \texttt{case}\ x\ \texttt{of}\ \texttt{in}_1\, y.\, e_1\ \texttt{|}\ \texttt{in}_2\, y.\, e_2$

Definition 3 (May- and must-adequate relations)**.** *A type-indexed relation $\mathcal{R}$ is* may-adequate *if, whenever $\varnothing; \varnothing \vdash e\ \mathcal{R}\ e' : \tau$ holds, then $e \downarrow$ implies $e' \downarrow$. It is* must-adequate *if, whenever $\varnothing; \varnothing \vdash e\ \mathcal{R}\ e' : \tau$ holds, then $e \Downarrow$ implies $e' \Downarrow$.*

Definition 4 (Contextual approximations and equivalences)**.** May-contextual approximation*, written $\lesssim^{ctx}_{\downarrow}$, is the largest may-adequate precongruence.* May-contextual equivalence*, $\cong^{ctx}_{\downarrow}$, is the symmetrization of $\lesssim^{ctx}_{\downarrow}$. Analogously,* must-contextual approximation*, written $\lesssim^{ctx}_{\Downarrow}$, is the largest must-adequate precongruence, and* must-contextual equivalence*, $\cong^{ctx}_{\Downarrow}$, is its symmetrization.* Contextual approximation*, $\lesssim^{ctx}$, and* contextual equivalence*, $\cong^{ctx}$, are given as intersections of the respective may- and must-relations, and thus $\cong^{ctx}$ is also the symmetrization of $\lesssim^{ctx}$.*

That this largest (may-, must-) adequate precongruence exists can be shown as in [16], by proving that the relation $S = \bigcup\{R \mid R \text{ compatible and (may-, must-) adequate}\}$ is an adequate precongruence.

In principle, to establish an equivalence $\Delta; \Gamma \vdash e \cong^{ctx} e' : \tau$ it suffices to find some may- and must-adequate congruence $\mathcal{R}$ that contains the tuple $(\Delta, \Gamma, e, e', \tau)$ since $\cong^{ctx}$ is the largest such relation. However, in practice it is difficult to verify that a relation $\mathcal{R}$ has the necessary compatibility properties in Figure 4. An alternative characterization of the contextual approximation and equivalence relations can be given in terms of CIU preorders [14], which we define next.

Definition 5 (CIU preorders)**.** May- and must-CIU preorder*, written $\lesssim^{ciu}_{\downarrow}$ and $\lesssim^{ciu}_{\Downarrow}$ resp., are the type-indexed relations defined as follows: for all e, e' with $\Delta; \Gamma \vdash e : \tau$ and $\Delta; \Gamma \vdash e' : \tau$,*

- $\Delta; \Gamma \vdash e \lesssim^{ciu}_{\downarrow} e' : \tau \quad \Leftrightarrow \quad \forall \delta \in \mathit{Type}^{\Delta}, \gamma \in \mathit{Subst}(\Gamma\delta), E \in \mathit{Stk}(\tau\delta).\ E[e\delta\gamma] \downarrow \ \Rightarrow\ E[e'\delta\gamma] \downarrow$
- $\Delta; \Gamma \vdash e \lesssim^{ciu}_{\Downarrow} e' : \tau \quad \Leftrightarrow \quad \forall \delta \in \mathit{Type}^{\Delta}, \gamma \in \mathit{Subst}(\Gamma\delta), E \in \mathit{Stk}(\tau\delta).\ E[e\delta\gamma] \Downarrow \ \Rightarrow\ E[e'\delta\gamma] \Downarrow$

The CIU preorder is defined as the intersection of $\lesssim^{ciu}_{\downarrow}$ and $\lesssim^{ciu}_{\Downarrow}$.

Theorem 6 (CIU theorem)**.** *The (may-, must-) CIU preorder coincides with (may-, must-) contextual approximation.*

Using the CIU theorem, it is easy to verify that all the deterministic reductions are also valid equivalences, and that the various call-by-value eta laws hold. Moreover, we can establish the laws of Moggi's computational lambda calculus and the basic (inequational) theory of erratic choice (Figure 5). We will prove the CIU theorem in Section 4 (for the may-CIU preorder) and Section 5 (for the must-CIU preorder).

3. Uniform relations

For an ordinal number α and a set X we define an *α-indexed uniform relation on X* to be a family $(R_\beta)_{\beta<\alpha}$ of relations $R_\beta \subseteq X$ such that

- $R_0 = X$,
- $R_{\beta+1} \subseteq R_\beta$ for all $\beta < \alpha$, and
- $R_\lambda = \bigcap_{\beta<\lambda} R_\beta$ for every limit ordinal $\lambda < \alpha$.

Let $\mathit{Rel}_\alpha(X)$ denote the α-indexed uniform relations on X.

Recursive definitions. The notions of n-equivalence, non-expansiveness and contractiveness (e.g., [5]) all generalize from the case of ω-indexed uniform relations: Given α-indexed uniform relations $R, S \in \mathit{Rel}_\alpha(X)$ and $\nu < \alpha$ we say that R and S are *ν-equivalent*, written $R \overset{\nu}{=} S$, if $R_\beta = S_\beta$ for all $\beta \leq \nu$. In particular, $R = S$ if and only if $R \overset{\nu}{=} S$ for all $\nu < \alpha$.

A function $F : \mathit{Rel}_\alpha(X_1) \times \cdots \times \mathit{Rel}_\alpha(X_n) \to \mathit{Rel}_\alpha(X)$ is *non-expansive* if $\vec{R} \overset{\nu}{=} \vec{S}$ implies $F(\vec{R}) \overset{\nu}{=} F(\vec{S})$, and F is *contractive* if $\vec{R} \overset{\nu}{=} \vec{S}$ implies $F(\vec{R}) \overset{\nu+1}{=} F(\vec{S})$. If $R \in \mathit{Rel}_\alpha(X)$ then $\triangleright R \in \mathit{Rel}_\alpha(X)$ is the uniform relation determined by $\triangleright R_{\beta+1} = R_\beta$; this operation gives rise to a contractive function on $\mathit{Rel}_\alpha(X)$.

Proposition 7 (Unique fixed points)**.** *If $F : \mathit{Rel}_\alpha(X) \to \mathit{Rel}_\alpha(X)$ is contractive, then F has a unique fixed point $\mathit{fix}r.F(r)$.*

Proof. First note that F has at most one fixed point: if R, S are fixed points of F then, by the contractiveness of F, we can establish that $R = F(R) \overset{\nu}{=} F(S) = S$ holds for all $\nu < \alpha$ by induction and thus $R = S$.

Because of the uniformity conditions it is sufficient to give the components of the fixed point $\mathit{fix}r.F(r)$ that are indexed by successor ordinals. We set $\mathit{fix}r.F(r)_{\nu+1} = F(R)_{\nu+1}$ where $R \in \mathit{Rel}_\alpha(X)$ is defined by $R_\beta = \mathit{fix}r.F(r)_\beta$ for $\beta \leq \nu$ and $R_\beta = \emptyset$ for $\beta > \nu$. By induction, it is easy to see that $\mathit{fix}r.F(r) \in \mathit{Rel}_\alpha(X)$ and that $F(\mathit{fix}r.F(r))_\nu = \mathit{fix}r.F(r)_\nu$ holds for all $\nu < \alpha$, and thus $F(\mathit{fix}r.F(r)) = \mathit{fix}r.F(r)$. □

Proposition 7 is an instance of Di Gianantonio and Miculan's sheaf-theoretic fixed point theorem [7]. Indeed, an α-indexed uniform relation on X corresponds to a subobject of the constant sheaf on X in the sheaf topos on α.

Uniform relations on syntax. For $\tau, \tau' \in \mathit{Type}$ we consider the collections of α-indexed uniform relations between values, terms and evaluation contexts: we write $\mathit{VRel}_\alpha(\tau, \tau')$ for $\mathit{Rel}_\alpha(\mathit{Val}(\tau) \times \mathit{Val}(\tau'))$, we write $\mathit{SRel}_\alpha(\tau, \tau')$ for $\mathit{Rel}_\alpha(\mathit{Stk}(\tau) \times \mathit{Stk}(\tau'))$, and we use $\mathit{TRel}_\alpha(\tau, \tau')$ for $\mathit{Rel}_\alpha(\mathit{Tm}(\tau) \times \mathit{Tm}(\tau'))$.

The description of the logical relations in the sections below makes use of the following (non-expansive) constructions on uniform relations:

- $R_1 \times R_2 \in \mathit{VRel}_\alpha(\tau_1 \times \tau_2, \tau_1' \times \tau_2')$, for $R_1 \in \mathit{VRel}_\alpha(\tau_1, \tau_1')$ and $R_2 \in \mathit{VRel}_\alpha(\tau_2, \tau_2')$, is defined by $(R_1 \times R_2)_\beta = \{(\langle v_1, v_2\rangle, \langle v_1', v_2'\rangle) \mid (v_1, v_1') \in (R_1)_\beta \ \wedge \ (v_2, v_2') \in (R_2)_\beta\}$.
- $R_1 \to R_2 \in \mathit{VRel}_\alpha(\tau_1 \to \tau_2, \tau_1' \to \tau_2')$, for $R_1 \in \mathit{VRel}_\alpha(\tau_1, \tau_1')$ and $R_2 \in \mathit{TRel}_\alpha(\tau_2, \tau_2')$, is given by $(R_1 \to R_2)_\beta = \{(\lambda x.e, \lambda x.e') \mid \forall \nu \leq \beta.\, \forall (v, v') \in (R_1)_\nu.\, (e[v/x], e'[v'/x]) \in (R_2)_\nu\}$.

- $\forall r.F(r) \in \mathit{VRel}_\alpha(\forall\alpha.\tau_1, \forall\alpha.\tau_1')$, for $F_{\tau,\tau'} : \mathit{VRel}_\alpha(\tau,\tau') \to \mathit{TRel}_\alpha(\tau_1[\tau/\alpha], \tau_1'[\tau'/\alpha])$ a family of non-expansive maps, is the uniform relation that is defined by $\forall r.F(r)_\beta = \{(\Lambda\alpha.e, \Lambda\alpha.e') \mid \forall\tau,\tau' \in \mathit{Type}, R \in \mathit{VRel}_\alpha(\tau,\tau').\ (e[\tau/\alpha], e'[\tau'/\alpha] \in F_{\tau,\tau'}(R)_\beta\}$.
- $\mathtt{in}_j R \in \mathit{VRel}_\alpha(\tau,\tau')$, for $\tau = \mu\alpha.\tau_1 + \ldots + \tau_m$ and $\tau' = \mu\alpha.\tau_1' + \ldots + \tau_n'$ and $R \in \mathit{VRel}_\alpha(\tau_j[\tau/\alpha], \tau_j'[\tau'/\alpha])$, is given by $(\mathtt{in}_j R)_\beta = \{(\mathtt{in}_j\, v, \mathtt{in}_j\, v') \mid (v,v') \in R_\beta\}$.

4. May equational theory

In this section, we will define a logical uniform relation that is used to prove that may-CIU preorder and may-contextual approximation coincide. The key idea of the definition is the usual one of step-indexing [2], i.e., that the observables can be stratified based on step-counting in the operational semantics. We write $e \downarrow_n$ if $e \longmapsto \ldots \longmapsto v$ for some $v \in \mathit{Val}$ in at most n reduction steps, thus $e\downarrow$ holds if and only if $e \downarrow_n$ for some n.

Logical ω-indexed uniform relation for may-approximation. In the case of may-approximation, it suffices to consider ω-indexed uniform relations. Using the constructions on relations given above, we define a relational interpretation $[\![\tau]\!]\,(\vec{r}) \in \mathit{VRel}_\omega(\tau[\vec{\tau}/\vec{\alpha}], \tau[\vec{\tau}'/\vec{\alpha}])$ by induction on the type $\vec{\alpha} \vdash \tau$, given closed types $\tau_1, \tau_1', \ldots, \tau_k, \tau_k' \in \mathit{Type}$ and relations $r_1 \in \mathit{VRel}_\omega(\tau_1,\tau_1'), \ldots, r_k \in \mathit{VRel}_\omega(\tau_k,\tau_k')$:

$$[\![\alpha_i]\!]\,(\vec{r}) = r_i \qquad [\![\tau_1 \times \tau_2]\!]\,(\vec{r}) = [\![\tau_1]\!]\,(\vec{r}) \times [\![\tau_2]\!]\,(\vec{r})$$

$$[\![\mathbf{1}]\!]\,(\vec{r}) = (\mathit{Id}_{\mathbf{1}})_{n<\omega} \qquad [\![\tau_1 \to \tau_2]\!]\,(\vec{r}) = [\![\tau_1]\!]\,(\vec{r}) \to [\![\tau_2]\!]\,(\vec{r})^{\perp\!\!\perp}$$

$$[\![\forall\alpha.\tau]\!]\,(\vec{r}) = \forall r.[\![\tau]\!]\,(\vec{r}, r)^{\perp\!\!\perp} \qquad [\![\mu\alpha.\tau_1 + \ldots + \tau_m]\!]\,(\vec{r}) = \mathit{fix}\, s. \bigcup_j \mathtt{in}_j(\triangleright\, [\![\tau_j]\!]\,(\vec{r}, s))$$

Here, value relations $r \in \mathit{VRel}_\omega(\tau,\tau')$ are lifted to relations $r^\perp \in \mathit{SRel}_\omega(\tau,\tau')$ on evaluation contexts and to relations $r^{\perp\!\!\perp} \in \mathit{TRel}_\omega(\tau,\tau')$ on terms by biorthogonality, much as in [8]:

$$r_n^\perp = \{(E,E') \mid \forall j \le n.\ \forall (v,v') \in r_j.\ E[v]\downarrow_j \Rightarrow\ E'[v']\downarrow\}$$

$$r_n^{\perp\!\!\perp} = \{(e,e') \mid \forall j \le n.\ \forall (E,E') \in r_j^\perp.\ E[e]\downarrow_j \Rightarrow\ E'[e']\downarrow\}$$

The fixed point in the interpretation of recursive types is well-defined by Proposition 7 since each $[\![\tau]\!]$ denotes a family of non-expansive functions, and thus composition with $\triangleright$ yields a contractive function.

The following observation is useful for calculations:

Lemma 8 (Context composition)**.** *If* $(v,v') \in [\![\tau_1 \to \tau_2]\!]\,\vec{r}_n$ *and* $(E,E') \in [\![\tau_2]\!]\,\vec{r}_n^\perp$ *then* $(E[v\,[]], E'[v'\,[]]) \in [\![\tau_1]\!]\,\vec{r}_{n+1}^\perp$.

Proof. Let $j \le n+1$, $(v_1,v_1') \in [\![\tau_1]\!]\,\vec{r}_j$. Assume $E[v\,v_1]\downarrow_j$. We have $v = \lambda x.e$ and $v' = \lambda x.e'$ and $(\lambda x.e, \lambda x.e') \in [\![\tau_1 \to \tau_2]\!]\,\vec{r}_n$ for some x, e, e' and necessarily $E[v\,v_1] \longmapsto E[e[v_1/x]]\downarrow_{j-1}$. By definition, $(e[v_1/x], e'[v_1'/x]) \in [\![\tau_2]\!]\,\vec{r}_{j-1}^{\perp\!\!\perp}$. From $(E,E') \in [\![\tau_2]\!]\,\vec{r}_n^\perp$ we obtain $E'[e'[v_1'/x]]\downarrow$. Thus, $E'[v'\,v_1']\downarrow$. □

The relational interpretation extends pointwise to value substitutions: $(\gamma,\gamma') \in [\![\Gamma]\!]\,\vec{r}_n$ if $(\gamma(x),\gamma(x')) \in [\![\tau]\!]\,\vec{r}_n$ for all $x{:}\tau \in \Gamma$. Based on this interpretation we consider the following type-indexed relation:

$$\begin{aligned}\Delta;\Gamma \vdash e \lesssim^{log}_{\downarrow} e' : \tau \;\; &\text{where } \Delta = \vec{\alpha}\\ \Leftrightarrow\;\; &\forall \vec{\tau},\vec{\tau}'.\,\forall \vec{r} \in \mathit{VRel}_\omega(\vec{\tau},\vec{\tau}').\,\forall n<\omega.\,\forall(\gamma,\gamma') \in [\![\Gamma]\!]\,\vec{r}_n.\;(e[\vec{\tau}/\vec{\alpha}]\gamma, e'[\vec{\tau}'/\vec{\alpha}]\gamma') \in [\![\tau]\!]\,\vec{r}_n^{\perp\perp}\end{aligned}$$

The definition of $\lesssim^{log}_{\downarrow}$ builds in enough closure properties to prove its compatibility.

Proposition 9 (Fundamental property)**.** *The relation $\lesssim^{log}_{\downarrow}$ has the compatibility properties given in Figure 4. In particular, it is reflexive: if $\Delta;\Gamma \vdash e : \tau$ then $\Delta;\Gamma \vdash e \lesssim^{log}_{\downarrow} e : \tau$.*

Proof. We consider the inference rules from Figure 4 in turn.

- For the introduction of recursive types, we assume that, for every $1 \le j \le m$, $\Delta;\Gamma \vdash v \lesssim^{log}_{\downarrow} v' : \tau_j[\mu\alpha.\tau_1 + \ldots + \tau_m/\alpha]$, and then prove that $\Delta;\Gamma \vdash \mathtt{in}_j\, v \lesssim^{log}_{\downarrow} \mathtt{in}_j\, v' : \mu\alpha.\tau_1 + \ldots + \tau_m$.

  For notational convenience we only consider the case of closed terms. Let τ abbreviate the type $\mu\alpha.\tau_1 + \ldots + \tau_m$. Note that $[\![\tau]\!]\,\vec{r} = \bigcup_j \mathtt{in}_j\big(\triangleright\,[\![\tau_j]\!]\,(\vec{r}, [\![\tau]\!]\,\vec{r})\big) = \bigcup_j \mathtt{in}_j\big(\triangleright\,[\![\tau_j[\tau/\alpha]]\!]\,(\vec{r})\big)$ by definition and a substitution lemma, and that the inclusion $[\![\tau_j[\tau/\alpha]]\!]\,(\vec{r}) \subseteq \triangleright\,[\![\tau_j[\tau/\alpha]]\!]\,(\vec{r})$ holds. Thus, assuming $(E,E') \in [\![\tau]\!]\,\vec{r}_n^{\perp}$ it follows from Lemma 8 that $(E[(\lambda x.\mathtt{in}_j\, x)\,[]], E'[(\lambda x.\mathtt{in}_j\, x)\,[]]) \in [\![\tau_j[\tau/\alpha]]\!]\,\vec{r}_{n+1}^{\perp}$. Thus, if $E[\mathtt{in}_j\, v]\downarrow_i$ for some $i \le n$ then $E'[(\lambda x.\mathtt{in}_j\, x)\, v'])\downarrow$ follows from $(v,v') \in [\![\tau_j[\tau/\alpha]]\!]\,\vec{r}_{n+1}^{\perp\perp}$. Therefore we can conclude $E'[\mathtt{in}_j\, v']\downarrow$, and have shown $(\mathtt{in}_j\, v, \mathtt{in}_j\, v') \in [\![\tau]\!]\,\vec{r}_n^{\perp\perp}$. Since n was chosen arbitrarily, we have $\Delta;\Gamma \vdash \mathtt{in}_j\, v \lesssim^{log}_{\downarrow} \mathtt{in}_j\, v' : \tau$.
- For the elimination of recursive types, we assume that τ is of the form $\mu\alpha.\tau_1 + \ldots + \tau_m$, $\Delta;\Gamma, x_j{:}\tau_j[\tau/\alpha] \vdash e_j \lesssim^{log}_{\downarrow} e'_j : \tau'$ for all $1 \le j \le m$ and $\Delta;\Gamma \vdash v \lesssim^{log}_{\downarrow} v' : \tau$. We prove $\Delta;\Gamma \vdash \mathtt{case}\ v\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots) \lesssim^{log}_{\downarrow} \mathtt{case}\ v'\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e'_j \mid \ldots) : \tau'$.

  For simplicity we only consider the case of closed terms. By definition and by a substitution lemma we have $[\![\tau]\!]\,\vec{r} = \bigcup_j \mathtt{in}_j\big(\triangleright\,[\![\tau_j]\!]\,(\vec{r}, [\![\tau]\!]\,\vec{r})\big) = \bigcup_j \mathtt{in}_j(\triangleright\,[\![\tau_j[\tau/\alpha]]\!]\,\vec{r})$. Moreover, $(\lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots), \lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e'_j \mid \ldots)) \in [\![\tau \to \tau']\!]\,\vec{r}_n$ for any n. To see this, assume $k \le n$, let $(a,a') \in [\![\tau]\!]\,\vec{r}_n$ and $(E,E') \in [\![\tau']\!]\,\vec{r}_n^{\perp}$ such that $E[\mathtt{case}\ a\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots)]\downarrow_k$. By the above observation we have $a = \mathtt{in}_j a_j$ and $a' = \mathtt{in}_j a'_j$ for some $(a_j, a'_j) \in [\![\tau_j[\tau/\alpha]]\!]\,\vec{r}_{k-1}$. From $E[\mathtt{case}\ a\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots)]\downarrow_k$ we obtain $E[e_j[a_j/x_j]]\downarrow_{k-1}$, and thus the assumption on e_j and e'_j gives $E'[e'_j[a'_j/x_j]]\downarrow$. From this we can conclude that $E'[\mathtt{case}\ a'\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e'_j \mid \ldots)]\downarrow$ holds.

  To prove the case, assume next that $(E,E') \in [\![\tau']\!]\,\vec{r}_n^{\perp}$. From Lemma 8 we obtain $(E[(\lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots))\,[]], E'[(\lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e'_j \mid \ldots))\,[]]) \in [\![\tau]\!]\,\vec{r}_{n+1}^{\perp}$. Now, since we know $(v,v') \in [\![\tau]\!]\,\vec{r}_{n+1}^{\perp\perp}$ by assumption, we obtain that $E[\mathtt{case}\ v\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots)]\downarrow_n$ implies $E[\mathtt{case}\ v'\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e'_j \mid \ldots)]\downarrow$ as required.
- For choice, we assume $\Delta \vdash \Gamma$ and show $\Delta;\Gamma \vdash \mathtt{?} \lesssim^{log}_{\downarrow} \mathtt{?} : \mathtt{nat}$. Suppose $(E,E') \in [\![\mathtt{nat}]\!]\,\vec{r}_n^{\perp}$ and $E[\mathtt{?}]\downarrow_j$ for some $j \le n$. Then $E[\mathtt{?}] \longmapsto E[\underline{k}]$ and $E[\underline{k}]\downarrow_{j-1}$ for some $k \in \mathbb{N}$. By induction on k, and using the compatibility for the introduction of recursive types, we obtain that $(\underline{k},\underline{k}) \in [\![\mathtt{nat}]\!]\,\vec{r}_n^{\perp\perp}$, and thus $E'[\underline{k}]\downarrow$. Hence $E'[\mathtt{?}]\downarrow$.

$$\frac{\Delta;\Gamma \vdash v \mathrel{\mathcal{R}} v' : \tau \qquad \Delta;\Gamma, x{:}\tau \vdash e \mathrel{\mathcal{R}} e' : \tau'}{\Delta;\Gamma \vdash e[v/x] \mathrel{\mathcal{R}} e'[v'/x] : \tau'} \qquad \frac{\Delta, \alpha;\Gamma \vdash e \mathrel{\mathcal{R}} e' : \tau'}{\Delta;\Gamma[\tau/\alpha] \vdash e \mathrel{\mathcal{R}} e' : \tau'[\tau/\alpha]}\ \Delta \vdash \tau$$

Figure 6: Substitutivity properties of type-indexed relations

The proofs for the remaining rules are similar. and can be found in Appendix A.2. □

Theorem 10 (Coincidence)**.** $\Delta;\Gamma \vdash e \lesssim^{log}_{\downarrow} e' : \tau$ *if and only if* $\Delta;\Gamma \vdash e \lesssim^{ciu}_{\downarrow} e' : \tau$.

Proof. For the direction from left to right, let $\delta \in \mathit{Type}^{\Delta}$, $\gamma \in \mathit{Subst}(\Gamma\delta)$ and $E \in \mathit{Stk}(\tau\delta)$, and assume $E[e\delta\gamma]\downarrow$, i.e., $E[e\delta\gamma]\downarrow_n$ for some n. We must show $E[e'\delta\gamma]\downarrow$. As a consequence of Proposition 9, $(\gamma,\gamma) \in [\![\Gamma\delta]\!]_n$ and $(E,E) \in [\![\tau\delta]\!]^{\perp}_n$. By definition of $\Delta;\Gamma \vdash e \lesssim^{log}_{\downarrow} e' : \tau$ and a substitution lemma we have $(e\delta\gamma, e'\delta\gamma) \in [\![\tau\delta]\!]^{\perp\perp}_n$, and thus $E[e\delta\gamma]\downarrow_n$ gives $E[e'\delta\gamma]\downarrow$.

For the direction from right to left, first note that the logical relation is closed under may-CIU approximation; more precisely, if $\Delta;\Gamma \vdash e \lesssim^{log}_{\downarrow} e' : \tau$ and $\Delta;\Gamma \vdash e' \lesssim^{ciu}_{\downarrow} e'' : \tau$ then $\Delta;\Gamma \vdash e \lesssim^{log}_{\downarrow} e'' : \tau$. This observation follows from the definition of $(\cdot)^{\perp\perp}$ used in $\Delta;\Gamma \vdash e \lesssim^{log}_{\downarrow} e' : \tau$ and the definition of CIU approximation. Now assume that $\Delta;\Gamma \vdash e \lesssim^{ciu}_{\downarrow} e' : \tau$. By Proposition 9, $\Delta;\Gamma \vdash e \lesssim^{log}_{\downarrow} e : \tau$, and thus $\Delta;\Gamma \vdash e \lesssim^{log}_{\downarrow} e' : \tau$. □

Proof of CIU Theorem 6(1). We first show that $\lesssim^{ciu}_{\downarrow}$ is contained in $\lesssim^{ctx}_{\downarrow}$. By definition, $\lesssim^{ctx}_{\downarrow}$ is the largest may-adequate precongruence, thus it is sufficient to establish that $\lesssim^{ciu}_{\downarrow}$ is a may-adequate precongruence. From the definition it is immediate that $\lesssim^{ciu}_{\downarrow}$ is may-adequate, reflexive and transitive. By Theorem 10, $\lesssim^{ciu}_{\downarrow}$ coincides with $\lesssim^{log}_{\downarrow}$ which is compatible by Proposition 9.

For the other direction, following Pitts [17], we first consider the special case where $\varnothing;\varnothing \vdash e \lesssim^{ctx}_{\downarrow} e' : \tau$. To prove $\varnothing;\varnothing \vdash e \lesssim^{ciu}_{\downarrow} e' : \tau$, note that $\varnothing;\varnothing \vdash E[e] \lesssim^{ctx}_{\downarrow} E[e'] : \tau'$ holds for all evaluation contexts E such that $\vdash E : \tau \multimap \tau'$ since $\lesssim^{ctx}_{\downarrow}$ is reflexive and compatible. Hence, that $E[e]\downarrow$ implies $E[e']\downarrow$ follows since $\lesssim^{ctx}_{\downarrow}$ is may-adequate.

The general case reduces to this special case since may-contextual approximation has the substitutivity properties given in Figure 6. For the first of these, assume $\Delta;\Gamma \vdash v \lesssim^{ctx}_{\downarrow} v' : \tau$ and $\Delta;\Gamma, x{:}\tau \vdash e \lesssim^{ctx}_{\downarrow} e' : \tau'$. From the definition of may-CIU approximation it is easy to see

$$\Delta;\Gamma \vdash e[v/x] \lesssim^{ciu}_{\downarrow} (\lambda x.e)\, v : \tau' \quad \text{and} \quad \Delta;\Gamma \vdash (\lambda x.e')\, v' \lesssim^{ciu}_{\downarrow} e'[v'/x] : \tau' \ .$$

Since we have already shown that $\lesssim^{ciu}_{\downarrow}$ is contained in $\lesssim^{ctx}_{\downarrow}$, and since $\Delta;\Gamma \vdash (\lambda x.e)\, v \lesssim^{ctx}_{\downarrow} (\lambda x.e')\, v' : \tau'$ by compatibility, we can conclude $\Delta;\Gamma \vdash e[v/x] \lesssim^{ctx}_{\downarrow} e'[v'/x] : \tau'$ by transitivity. The second substitutivity property is proved similarly, using a weakening property of may-contextual approximation. □

5. Must equational theory

To define the logical relation for must-approximation, we need to stratify the observables again. For terms e and ordinals β we define $e \Downarrow_\beta$ inductively, as the least relation such that $e \Downarrow_\beta$ if for all e' such that $e \longmapsto e'$ there exists $\nu < \beta$ and $e' \Downarrow_\nu$. The essential observation is that $\Downarrow_\beta$ indeed captures must-convergent behaviour.

Proposition 11 (Stratified must-convergence)**.** *$e \Downarrow$ if and only if $e \Downarrow_\beta$ for some $\beta < \omega_1$ (for ω_1 the least uncountable ordinal).*

Proof. The proof from left to right is by induction on $e \Downarrow$. By induction hypothesis there exists ordinals $\nu(e') < \omega_1$ for each term e' such that $e \longmapsto e'$. Let $\beta = \bigcup \nu(e')$, then $\beta + 1 < \omega_1$ (since there are only countably many such e' and each $\nu(e')$ is countable) and $e \Downarrow_{\beta+1}$. The direction from right to left is by induction on β. □

Logical ω_1-indexed uniform relation for must-approximation. Proposition 11 indicates that logical relations for must-approximation need to be indexed over ω_1. The lifting of value relations $r \in \mathit{VRel}_{\omega_1}(\tau, \tau')$ to relations $r^\perp \in \mathit{SRel}_{\omega_1}(\tau, \tau')$ on evaluation contexts and to relations $r^{\perp\!\perp} \in \mathit{TRel}_{\omega_1}(\tau, \tau')$ on terms is defined with respect to must termination.

$$r^\perp_\beta = \{(E, E') \mid \forall \nu \le \beta.\ \forall (v, v') \in r_\nu.\ E[v] \Downarrow_\nu \Rightarrow\ E'[v'] \Downarrow \}$$
$$r^{\perp\!\perp}_\beta = \{(e, e') \mid \forall \nu \le \beta.\ \forall (E, E') \in r^\perp_\nu.\ E[e] \Downarrow_\nu \Rightarrow\ E'[e'] \Downarrow \}$$

Except for this difference, the relational interpretation $[\![\tau]\!]\,(\vec{r}) \in \mathit{VRel}_{\omega_1}(\tau[\vec{\tau}/\vec{\alpha}], \tau[\vec{\tau}'/\vec{\alpha}])$ is literally the same as in Section 4 and defined by induction on the type $\vec{\alpha} \vdash \tau$, given closed types $\tau_1, \tau_1', \ldots, \tau_k, \tau_k' \in \mathit{Type}$ and relations $r_1 \in \mathit{VRel}_{\omega_1}(\tau_1, \tau_1'), \ldots, r_k \in \mathit{VRel}_{\omega_1}(\tau_k, \tau_k')$:

$$[\![\alpha_i]\!]\,(\vec{r}) = r_i \qquad [\![\tau_1 \times \tau_2]\!]\,(\vec{r}) = [\![\tau_1]\!]\,(\vec{r}) \times [\![\tau_2]\!]\,(\vec{r})$$
$$[\![\mathbf{1}]\!]\,(\vec{r}) = (\mathit{Id}_{\mathbf{1}})_{\beta<\omega_1} \qquad [\![\tau_1 \to \tau_2]\!]\,(\vec{r}) = [\![\tau_1]\!]\,(\vec{r}) \to [\![\tau_2]\!]\,(\vec{r})^{\perp\!\perp}$$
$$[\![\forall\alpha.\tau]\!]\,(\vec{r}) = \forall r.[\![\tau]\!]\,(\vec{r}, r)^{\perp\!\perp} \qquad [\![\mu\alpha.\tau_1 + \ldots + \tau_m]\!]\,(\vec{r}) = \mathit{fix}\, s.\bigcup_j \mathtt{in}_j(\triangleright\, [\![\tau_j]\!]\,(\vec{r}, s))$$

Logical must-approximation is defined as follows:

$$\Delta; \Gamma \vdash e \lesssim^{\mathit{log}}_{\Downarrow} e' : \tau \ \text{ where } \Delta = \vec{\alpha}$$
$$\Leftrightarrow\ \forall \vec{\tau}, \vec{\tau}'.\, \forall \vec{r} \in \mathit{VRel}_{\omega_1}(\vec{\tau}, \vec{\tau}').\, \forall \beta < \omega_1.\, \forall (\gamma, \gamma') \in [\![\Gamma]\!]\,\vec{r}_\beta.\ (e[\vec{\tau}/\vec{\alpha}]\gamma, e'[\vec{\tau}'/\vec{\alpha}]\gamma') \in [\![\tau]\!]\,\vec{r}^{\perp\!\perp}_\beta$$

Proposition 12 (Fundamental property)**.** *The relation $\lesssim^{\mathit{log}}_{\Downarrow}$ has the compatibility properties given in Figure 4. In particular, it is reflexive: if $\Delta; \Gamma \vdash e : \tau$ then $\Delta; \Gamma \vdash e \lesssim^{\mathit{log}}_{\Downarrow} e : \tau$.*

Proof. The proof is similar to the one for Proposition 9. We give only the case for choice, where we assume $\Delta \vdash \Gamma$ and prove $\Delta; \Gamma \vdash \mathtt{?} \lesssim^{\mathit{log}}_{\Downarrow} \mathtt{?} : \mathtt{nat}$. Suppose $(E, E') \in [\![\mathtt{nat}]\!]\,\vec{r}^\perp_\beta$ and $E[\mathtt{?}] \Downarrow_\beta$. Then $E[\mathtt{?}] \longmapsto e$ implies that e is of the form $E[\underline{k}]$ and $E[\underline{k}] \Downarrow_{\nu_k}$ for some $k \in \mathbb{N}$ and $\nu_k < \beta$. Using the compatibility for the introduction form of recursive types, an induction on k shows that $(\underline{k}, \underline{k}) \in [\![\mathtt{nat}]\!]\,\vec{r}^{\perp\!\perp}_{\nu_k}$, and thus $E'[\underline{k}] \Downarrow$ for all $k \in \mathbb{N}$. Hence $E'[\mathtt{?}] \Downarrow$. □

Theorem 13 (Coincidence)**.** *$\Delta; \Gamma \vdash e \lesssim^{\mathit{log}}_{\Downarrow} e' : \tau$ if and only if $\Delta; \Gamma \vdash e \lesssim^{\mathit{ciu}}_{\Downarrow} e' : \tau$.*

Proof. The proof is completely analogous to that of Theorem 10. For the direction from left to right one uses the characterization of $\Downarrow$ in terms of $\Downarrow_\beta$ (Proposition 11) and then appeals to Proposition 12. The direction from right to left uses the fact that $\lesssim^{\mathit{log}}_{\Downarrow}$ is closed under must-CIU approximation. □

$$\frac{\Delta;\Gamma \vdash v\, v' \lesssim^{ctx}_{\downarrow} v' : \tau_1 \to \tau_2}{\Delta;\Gamma \vdash \mathtt{fix}\,\tau_1\tau_2\, v \lesssim^{ctx}_{\downarrow} v' : \tau_1 \to \tau_2} \qquad \frac{\Delta;\Gamma \vdash v\, v' \lesssim^{ctx}_{\Downarrow} v' : \tau_1 \to \tau_2}{\Delta;\Gamma \vdash \mathtt{fix}\,\tau_1\tau_2\, v \lesssim^{ctx}_{\Downarrow} v' : \tau_1 \to \tau_2}$$

Figure 7: Recursion induction: least fixed point property of `fix`

Proof of CIU Theorem 6(2). The proof is analogous to that of Theorem 6(1). From the definition, $\lesssim^{ciu}_{\Downarrow}$ is a must-adequate reflexive and transitive relation, by Proposition 12 and Theorem 13 it is also compatible, and thus contained in $\lesssim^{ctx}_{\Downarrow}$. From this containment and the closure of $\lesssim^{ciu}_{\Downarrow}$ under beta conversion it follows that $\lesssim^{ctx}_{\Downarrow}$ has the substitutivity properties in Figure 6. Thus it suffices to prove the containment of $\lesssim^{ctx}_{\Downarrow}$ in $\lesssim^{ciu}_{\Downarrow}$ for closed terms, which is clear by the compatibility and must-adequacy of $\lesssim^{ctx}_{\Downarrow}$. □

6. Applications

This section illustrates how the logical relation characterization of contextual approximation can be used to derive interesting examples and further proof principles. We consider three such applications: a recursion-induction principle for recursively defined functions, syntactic minimal invariance of a recursive type, and a "free theorem" about a polymorphic type.

Proving recursion-induction for a similar language (without polymorphic types) has been an open problem [10]. Here, the proof is essentially a straightforward induction, using the indexing of the logical relations.

Recursion-induction. Recall from the introduction that $\mathtt{fix} : \forall\alpha,\beta.((\alpha\to\beta)\to(\alpha\to\beta))\to(\alpha\to\beta)$ is given by the term $\Lambda\alpha,\beta.\lambda f.\delta_f(\mathtt{in}\,\delta_f)$ where δ_f is an abbreviation for the term $\lambda y.\mathtt{case}\ y\ \mathtt{of}\ \mathtt{in}\,y'.\, f(\lambda x.(\lambda r.r\,x)(y'\,y))$. We now prove that `fix` is a *least* fixed point combinator, i.e., we prove the soundness of the recursion-induction rules in Figure 7. We only include the proof for $\lesssim^{ctx}_{\Downarrow}$ and for notational simplicity we assume that the contexts Δ and Γ are empty. We assume the premise of the rule, and to show the conclusion we first prove that $(h, v') \in [\![\tau_1 \to \tau_2]\!]_\beta$ where h is $\lambda x.(\lambda r.r\,x)\,(\delta_v\,(\mathtt{in}\,\delta_v))$, for all $\beta < \omega_1$. The result then follows from the agreement of the logical relation with contextual approximation and transitivity, since $\mathtt{fix}\,\tau_1\tau_2\, v \cong^{ctx} v\, h \lesssim^{ctx}_{\Downarrow} v\, v' \lesssim^{ctx}_{\Downarrow} v'$.

To prove $(h, v') \in [\![\tau_1 \to \tau_2]\!]_\beta$ we proceed by induction on β and assume that $(h, v') \in [\![\tau_1 \to \tau_2]\!]_{\nu_1}$, for all $\nu < \beta$; we are then to show that $(h, v') \in [\![\tau_1 \to \tau_2]\!]_\beta$. From the typing rules, v' must be of the form $\lambda x.e'$ for some e'. So let $\beta_1 \le \beta$ and $(u, u') \in [\![\tau_1]\!]_{\beta_1}$, then it remains to show $((\lambda r.r\,u)(\delta_v\,(\mathtt{in}\,\delta_v)), e'[u'/x]) \in [\![\tau_2]\!]^{\perp\perp}_{\beta_1}$.

Suppose $\beta_2 \le \beta_1$, $(E, E') \in [\![\tau_2]\!]^{\perp}_{\beta_2}$ and $E[(\lambda r.r\,u)(\delta_v\,(\mathtt{in}\,\delta_v))] \Downarrow_{\beta_2}$; we are to show $E'[e'[u'/x]] \Downarrow$. By (the must-analogue of) Lemma 8 and the fundamental property of the logical relation applied to v we obtain $(E[(\lambda r.r\,u)\,((\lambda x.v\,x)\,[])], E'[(\lambda r.r\,u')\,((\lambda x.v\,x)\,[])]) \in [\![\tau_1 \to \tau_2]\!]^{\perp}_{\beta_2}$. Then, since $\delta_v\,(\mathtt{in}\,\delta_v) \longmapsto^2 v\,h$ and $(\lambda x.v\,x)\,h \longmapsto v\,h$, we have $E[(\lambda r.r\,u)(v\,h)] \Downarrow_{\beta_3}$ for $\beta_3 < \beta_2 \le \beta$, and hence also $E'[(\lambda r.r\,u')\,(v\,v')] \Downarrow$ by induction hypothesis.

By the premise and Theorem 13 we have that $v\,v'$ CIU-approximates v', and thus we get $E'[(\lambda r.r\,u')\,v'] \Downarrow$. Finally, since $(\lambda r.r\,u')\,v' \longmapsto^* e'[u'/x]$ we obtain the required $E'[e'[u'/x]] \Downarrow$.

Syntactic minimal invariance. Consider the type $\tau = \mu\alpha.\mathtt{nat} + \alpha \to \alpha$. Let $id = \lambda x.x$ and consider the term

$$f \;\equiv\; \lambda h, x.\mathtt{case}\ x\ \mathtt{of}\ \mathtt{in}_1\, y.\, \mathtt{in}_1\, y \mid \mathtt{in}_2\, g.\, \mathtt{in}_2\, \lambda y.h(g(h\, y)) \;.$$

We shall show that $\mathtt{fix}\,\tau\tau\, f \cong^{ctx} id : \tau \to \tau$. This equivalence corresponds to the characterization of solutions to recursive domain equations as minimal invariants in domain-theoretic work [15], from which Pitts derives several (co-) induction principles.

By the soundness of the call-by-value beta- and eta-laws for contextual equivalence (Figure 5) and the transitivity of $\lesssim^{ctx}$, it is easy to see that $f\, id \cong^{ctx} id : \tau \to \tau$. The recursion-induction principle therefore yields $\mathtt{fix}\,\tau\tau\, f \lesssim^{ctx} id : \tau \to \tau$.

For the reverse approximation we first show $id \lesssim^{log}_{\Downarrow} h : \tau \to \tau$ where h is again the term $\lambda x.(\lambda r.r\, x)(\delta_f\, (\mathtt{in}\, \delta_f))$. We show this by proving $(id, h) \in [\![\tau \to \tau]\!]_\beta$ for all $\beta < \omega_1$ by induction on β. (The case for may-approximation is similar.) By definition, we need to show that for all $\nu \le \beta$ and all $(v, v') \in [\![\tau]\!]_\nu$, $(id\, v, h\, v') \in [\![\tau]\!]^{\perp\!\perp}_\nu$. Since $[\![\tau]\!] = \mathtt{in}_1(\triangleright [\![\mathtt{nat}]\!]) \cup \mathtt{in}_2(\triangleright [\![\tau \to \tau]\!])$ there are two cases to consider:

- Case $(v, v') \in \mathtt{in}_1(\triangleright [\![\mathtt{nat}]\!])_\nu$. Then there exist $u, u' \in \mathit{Val}(\mathtt{nat})$ such that $v = \mathtt{in}_1\, u$, $v' = \mathtt{in}_1\, u'$ and $(u, u') \in [\![\mathtt{nat}]\!]_{\nu'}$ for all $\nu' < \nu \le \beta$. Note that in this case $(\lambda x.(\lambda r.r\, x)(\delta_f\, (\mathtt{in}\, \delta_f)))\, v' \cong^{ctx} v' : \tau$. Thus, given $(E, E') \in [\![\tau]\!]^{\perp}_\nu$ such that $E[id\, v] \Downarrow_\nu$, it suffices to show $E'[v'] \Downarrow$ which easily follows from $(v, v') \in [\![\tau]\!]_\nu$.
- Case $(v, v') \in \mathtt{in}_2(\triangleright [\![\tau \to \tau]\!])_\nu$. Then there exist $g, g' \in \mathit{Val}(\tau \to \tau)$ such that $v = \mathtt{in}_2\, g$, $v' = \mathtt{in}_2\, g'$ and $(g, g') \in [\![\tau \to \tau]\!]_{\nu'}$ for all $\nu' < \nu \le \beta$. In this case, we have the equivalence $(\lambda x.(\lambda r.r\, x)(\delta_f\, (\mathtt{in}\, \delta_f)))\, v' \cong^{ctx} \mathtt{in}_2(\lambda y.h(g'(h\, y))) : \tau$. Thus, it suffices to show $(g, \lambda y.h(g'(h\, y))) \in [\![\tau \to \tau]\!]_{\nu'}$ for all $\nu' < \nu$, or equivalently, $(g\, u, h(g'(h\, u'))) \in [\![\tau]\!]^{\perp\!\perp}_{\nu'}$ for all $\nu' < \nu$ and all $(u, u') \in [\![\tau]\!]_{\nu'}$. Let $(E, E') \in [\![\tau]\!]^{\perp}_{\nu'}$ and suppose $E[g\, u] \Downarrow_{\nu'}$; we have to show $E'[h(g'(h\, u'))] \Downarrow$. From the induction hypothesis we obtain $(E[id\, []], E'[h\, []]) \in [\![\tau]\!]^{\perp}_{\nu'+1}$, and thus $(E, E'[h\, []]) \in [\![\tau]\!]^{\perp}_{\nu'}$. Since $(g, g') \in [\![\tau \to \tau]\!]_{\nu'}$ the latter entails $(E[g\, []], E'[h(g'\, [])]) \in [\![\tau]\!]^{\perp}_{\nu'}$. Now, applying the induction hypothesis again this shows $(E[g(id\, [])], E'[h(g'(h\, []))]) \in [\![\tau]\!]^{\perp}_{\nu'+1}$, and thus the assumptions $E[g\, u] \Downarrow_{\nu'}$ and $(u, u') \in [\![\tau]\!]_{\nu'}$ imply $E'[h(g'(h\, u))] \Downarrow$.

By Theorem 13 and the CIU theorem, $id \lesssim^{log}_{\Downarrow} h : \tau \to \tau$ implies $id \lesssim^{ctx}_{\Downarrow} h : \tau \to \tau$. Since $id \cong^{ctx} f\, id : \tau \to \tau$ and $f\, h \cong^{ctx} \mathtt{fix}\,\tau\tau\, f : \tau \to \tau$ we obtain $id \lesssim^{ctx}_{\Downarrow} \mathtt{fix}\,\tau\tau\, f : \tau \to \tau$ by compatibility and transitivity of must-contextual equivalence.

Parametricity. Let $\tau_1, \tau_2 \in \mathit{Type}$ be closed types. Then the contextual approximation

$$\varnothing; h{:}\forall\alpha.\alpha \times \alpha \to \alpha, f{:}\tau_1 \to \tau_2, x{:}\tau_1, y{:}\tau_1 \;\vdash\; h\, \tau_2\, \langle f\, x, f\, y\rangle \lesssim^{ctx} f(h\, \tau_1\, \langle x, y\rangle) : \tau_2 \;. \qquad (6.1)$$

holds. For the proof of (6.1), we will consider the case of must-approximation only (may-approximation is completely analogous) and show

$$\varnothing; h{:}\forall\alpha.\alpha \times \alpha \to \alpha, f{:}\tau_1 \to \tau_2, x{:}\tau, y{:}\tau \;\vdash\; h\, \tau_2\, \langle f\, x, f\, y\rangle \lesssim^{log}_{\Downarrow} f(h\, \tau_1\, \langle x, y\rangle) : \tau_2 \;.$$

Fix $\beta < \omega_1$, $h \in \mathit{Val}(\forall\alpha.\alpha \times \alpha \to \alpha)$, $f \in \mathit{Val}(\tau_1 \to \tau_2)$ and $x, y \in \mathit{Val}(\tau_1)$. We need to show

$$(h\, \tau_2\, \langle f\, x, f\, y\rangle, f(h\, \tau_1\, \langle x, y\rangle) \in [\![\tau_2]\!]^{\perp\!\perp}_\beta \;. \qquad (6.2)$$

We have $(h, h) \in [\![\forall\alpha.\alpha \times \alpha \to \alpha]\!]^{\perp\!\perp}_\beta$ by Proposition 12, and we will instantiate α by (the opposite of) the graph of f. More precisely, consider the relation $r \in \mathit{VRel}(\tau_2, \tau_1)$ given by

$r_\nu = \{(v, v') \mid (v, f\,v') \in [\![\tau_2]\!]^{\perp\!\!\perp}_{\nu+1}\}$. Note that we have $(id, f) \in [\![\alpha \to \tau_2]\!]\, r_\beta$. Hence, to prove (6.2) it suffices to show $(h\,\tau_2\,\langle f\,x, f\,y\rangle, h\,\tau_1\,\langle x, y\rangle) \in r^{\perp\!\!\perp}_\beta$.

By definition of the logical relation we have $(h\,\tau_2, h\,\tau_1) \in [\![\alpha \times \alpha \to \alpha]\!]\, r^{\perp\!\!\perp}_\beta$, and by the compatibility properties it remains to show $(f\,x, x) \in r^{\perp\!\!\perp}_\beta$ and $(f\,y, y) \in r^{\perp\!\!\perp}_\beta$. We consider the former: Let $(E, E') \in r^\perp_\nu$ for $\nu \le \beta$ such that $E[f\,x] \Downarrow_\nu$; we must prove $E'[x] \Downarrow$. We have $(f, id) \in [\![\tau_1 \to \alpha]\!]\, r_\nu$ from which $(E[f\,[]], E'[]) \in [\![\tau_1]\!]^\perp_\nu$ follows. By Proposition 12 we have $(x, x) \in [\![\tau_1]\!]^{\perp\!\!\perp}_\nu$, and thus $E[f\,x] \Downarrow_\nu$ implies $E'[x] \Downarrow$.

Let us now consider the reverse approximation of (6.1), which holds under the condition that f is total and deterministic, i.e., that for all $v \in \mathit{Val}(\tau_1)$ there exists $u \in \mathit{Val}(\tau_2)$ such that $f\,v \cong^{ctx} u : \tau_2$.

We proceed as above and show only for the case of must-approximation. For $\beta < \omega_1$, $h \in \mathit{Val}(\forall\alpha.\alpha \times \alpha \to \alpha)$, $f \in \mathit{Val}(\tau_1 \to \tau_2)$ and $x, y \in \mathit{Val}(\tau_1)$ we will prove

$$(f(h\,\tau_1\,\langle x, y\rangle), h\,\tau_2\,\langle f\,x, f\,y\rangle) \in [\![\tau_2]\!]^{\perp\!\!\perp}_\beta\ . \tag{6.3}$$

We use $(h, h) \in [\![\forall\alpha.\alpha \times \alpha \to \alpha]\!]^{\perp\!\!\perp}_\beta$ where we instantiate α by the relation $s \in \mathit{VRel}(\tau_1, \tau_2)$, given by $s_\nu = \{(v, v') \mid (f\,v, v') \in [\![\tau_2]\!]^{\perp\!\!\perp}_{\nu+1}\}$. First note that $(f, id) \in [\![\alpha \to \tau_2]\!]\, s_\beta$, and thus the proof of (6.3) reduces to showing $(h\,\tau_1\,\langle x, y\rangle, h\,\tau_2\,\langle f\,x, f\,y\rangle) \in s^{\perp\!\!\perp}_\beta$.

Since we have $(h\,\tau_1, h\,\tau_2) \in [\![\alpha \times \alpha \to \alpha]\!]\, s^{\perp\!\!\perp}_\beta$ it suffices to show $(x, f\,x) \in s^{\perp\!\!\perp}_\beta$ and $(y, f\,y) \in s^{\perp\!\!\perp}_\beta$, and we consider the former. Let $(E, E') \in s^\perp_\nu$ for $\nu \le \beta$ such that $E[x] \Downarrow_\nu$; we must prove $E'[f\,x] \Downarrow$. By the assumption that f is total there exists $u \in \mathit{Val}(\tau_2)$ such that $f\,x \cong^{ctx} u : \tau_2$, and so it suffices to prove $E'[u] \Downarrow$. But this follows from $(x, u) \in s_\nu$, and the latter is immediate from the definition of s.

7. Adequacy of CPS transformation

In this section we consider a standard, call-by-value, cps transformation [18], extended in the obvious way to our language with countable nondeterminism. In [4] Birkedal and Harper give a correctness proof of such a cps-transformation by means of a logical relation for a language with recursive types (but no polymorphism and no nondeterminism), where the logical relation is constructed using syntactic minimal invariance. Here we show how to define and use step-indexed logical relations for showing the adequacy of the cps-transformation, both with respect to may- and must-equivalence. As far as we know, this is the first such result for a language with countable nondeterminism.

We fix the answer type ρ of the cps-transformation to $\rho = \mathbf{1}$. The cps-transformation of types, values and expressions is defined in Figure 8.

We will show the following adequacy theorem for the cps-transformation:

Theorem 14 (Adequacy)**.** *If* $\Delta; \Gamma \vdash e_1 : \tau$ *and* $\Delta; \Gamma \vdash e_2 : \tau$*, then* $\Delta; \Gamma^\circ \vdash \lambda k.(e_1)^\bullet_k \lesssim^{ctx} \lambda k.(e_2)^\bullet_k : \tau^\bullet$ *implies* $\Delta; \Gamma \vdash e_1 \lesssim^{ctx} e_2 : \tau$.

Thus program transformations on the cps-translated programs are sound with respect to the source level programs.

Proposition 15 (Type preservation)**.**

- *If* $\Delta; \Gamma \vdash v : \tau$ *then* $\Delta; \Gamma^\circ \vdash v^\circ : \tau^\circ$.
- *If* $\Delta; \Gamma \vdash e : \tau$ *then* $\Delta; \Gamma^\circ, k : \tau^\circ \to \rho \vdash (e)^\bullet_k : \rho$.

- Translation of types:

$$\begin{aligned}
\alpha^\circ &= \alpha \\
\mathbf{1}^\circ &= \mathbf{1} \\
(\tau_1 \times \tau_2)^\circ &= {\tau_1}^\circ \times {\tau_2}^\circ \\
(\tau_1 \to \tau_2)^\circ &= {\tau_1}^\circ \to {\tau_2}^\bullet \\
(\mu\alpha.\tau_1 + \ldots + \tau_n)^\circ &= \mu\alpha.{\tau_1}^\circ + \ldots + {\tau_n}^\circ \\
(\forall\alpha.\tau)^\circ &= \forall\alpha.\tau^\bullet
\end{aligned}$$

  where $\tau^\bullet = (\tau^\circ \to \rho) \to \rho$

- Translation of values:

$$\begin{aligned}
x^\circ &= x \\
\langle\rangle^\circ &= \langle\rangle \\
\langle v_1, v_2\rangle^\circ &= \langle {v_1}^\circ, {v_2}^\circ\rangle \\
(\lambda x.e)^\circ &= \lambda x.\lambda k.(e)^\bullet_k \\
(\mathtt{in}_i v)^\circ &= \mathtt{in}_i v^\circ \\
(\Lambda\alpha.e)^\circ &= \Lambda\alpha.\lambda k.(e)^\bullet_k
\end{aligned}$$

- Translation of computations:

$$\begin{aligned}
(v)^\bullet_k &= k(v^\circ) \\
(\mathtt{?})^\bullet_k &= k(\mathtt{?}) \\
(\mathtt{proj}_i\, v)^\bullet_k &= k(\mathtt{proj}_i\, v^\circ) \\
(v\, e)^\bullet_k &= (e)^\bullet_{(\lambda x.\mathtt{let}\ y = v^\circ\ x\ \mathtt{in}\ y\, k)} \\
(\mathtt{case}\ v\ \mathtt{of}\ \mathtt{in}_1\, x_1.\, e_1 \mid \ldots \mid \mathtt{in}_n\, x_n.\, e_n)^\bullet_k &= \mathtt{case}\ v^\circ\ \mathtt{of}\ \mathtt{in}_1\, x_1.\, (e_1)^\bullet_k \mid \ldots \mid \mathtt{in}_n\, x_n.\, (e_n)^\bullet_k \\
(v\, \tau)^\bullet_k &= \mathtt{let}\ y = v^\circ\, \tau^\circ\ \mathtt{in}\ y\, k
\end{aligned}$$

- Translation of evaluation contexts:

$$\begin{aligned}
([])^\bullet_k &= k \\
(v\, E)^\bullet_k &= (E)^\bullet_{(\lambda x.\mathtt{let}\ y = v^\circ\ x\ \mathtt{in}\ y\, k)}
\end{aligned}$$

Figure 8: Translation of types, terms and evaluation contexts.

- *If* $\vdash E : \tau_1 \multimap \tau_2$ *then* $\varnothing; k : {\tau_2}^\circ \to \rho \vdash (E)^\bullet_k : {\tau_1}^\circ \to \rho$

Proof. The first two parts are proved simultaneously by an induction on the respective typing derivation. The final part is by an induction on E. □

The cps translation is substitutive and compositional.

Proposition 16 (Substitution)**.**

- $(e[v/x])^\bullet_k = (e)^\bullet_k[v^\circ/x]$
- $(e[\tau/\alpha])^\bullet_k = (e)^\bullet_k[\tau^\circ/\alpha]$

Lemma 17. *If* $\vdash E : \tau' \multimap \tau$ *and* $\varnothing;\varnothing \vdash e : \tau'$ *then, for any* $\varnothing;\varnothing \vdash v : \tau^\circ \to \rho$, $(\lambda k.(e)^\bullet_k)(E)^\bullet_v \longmapsto (e)^\bullet_{(E)^\bullet_v}$.

Proposition 18 (Compositionality)**.** *If* $\vdash E : \tau' \multimap \tau$ *and* $\varnothing;\varnothing \vdash e : \tau'$ *then* $\varnothing;\varnothing \vdash \lambda k.(E[e])^\bullet_k \cong^{ctx} \lambda k.(e)^\bullet_{(E)^\bullet_k} : \tau^\bullet$.

Proof. By an induction on E, using Lemma 17 and the fact that cbv beta-reduction respects contextual equivalence. □

Computation steps of source language terms can be simulated. In fact, terms and their image under the translation have the same termination behaviour.

Proposition 19 (Simulation)**.** *Let* $\varnothing;\varnothing \vdash e_1 : \tau$ *and* $\varnothing;\varnothing \vdash e_2 : \tau$*, and let* $\varnothing;\varnothing \vdash k : \tau^\circ \to \rho$ *be any value. Then* $e_1 \longmapsto e_2$ *implies* $(e_1)^\bullet_k \longmapsto^+ (e_2)^\bullet_k$.

Proof. The proof is by an induction on the derivation of $e_1 \longmapsto e_2$, using the substitution lemma (Prop. 16). □

Proposition 20 (Equitermination)**.** *If* $\top$ *denotes the continuation* $\varnothing;\varnothing \vdash \lambda x.\langle\rangle : \tau^\circ \to \rho$ *then the following holds.*

(1) $e\downarrow \Leftrightarrow\ (e)^\bullet_\top\downarrow$*, and*
(2) $(e)^\bullet_\top \Downarrow \Leftrightarrow\ e\Downarrow$.

Proof. The direction from left to right of claim (1) follows from Proposition 19 since $e \longmapsto^* v$ entails $(e)^\bullet_\top \longmapsto^* (v)^\bullet_\top \longmapsto^* \langle\rangle$. Similarly, for the direction from left to right of claim (2), any infinite reduction sequence starting at e gives rise to a corresponding infinite reduction sequence starting at $(e)^\bullet_\top$.

The converse directions of claims (1) and (2) are more involved, and we will establish them using logical relations arguments below. □

The proof of Theorem 14 then is straightforward:

Proof. Let $\Delta;\Gamma \vdash e_1 : \tau$ and $\Delta;\Gamma \vdash e_2 : \tau$ be such that $\Delta;\Gamma^\circ \vdash \lambda k.(e_1)^\bullet_k \lesssim^{ctx} \lambda k.(e_2)^\bullet_k : \tau^\bullet$. By the ciu theorem it suffices to show $\Delta;\Gamma \vdash e_1 \lesssim^{ciu} e_2 : \tau$.

We first consider the case of may-ciu approximation. To this end, let δ, γ and E be such that $E[e_1\delta\gamma]\downarrow$. We need to prove that $E[e_2\delta\gamma]\downarrow$. Proposition 20(1) yields $(E[e_1\delta\gamma])^\bullet_\top\downarrow$, or equivalently by Propositions 16 and 18,

$$(e_1)^\bullet_{(E)^\bullet_\top}\delta^\circ\gamma^\circ\downarrow$$

From the assumption $\Delta;\Gamma^\circ \vdash \lambda k.(e_1)^\bullet_k \lesssim^{ctx} \lambda k.(e_2)^\bullet_k : \tau^\bullet$ we thus have $(e_2)^\bullet_{(E)^\bullet_\top}\delta^\circ\gamma^\circ\downarrow$, i.e., $(E[e_2\delta\gamma])^\bullet_\top\downarrow$. Proposition 20(1) then yields $E[e_2\delta\gamma]\downarrow$.

Next, we consider the case of must-ciu approximation, which is similar. Let δ, γ and E be such that $E[e_1\delta\gamma]\Downarrow$. We need to prove that $E[e_2\delta\gamma]\Downarrow$. By Proposition 20(2), $(E[e_1\delta\gamma])^\bullet_\top\Downarrow$, or equivalently,

$$(e_1)^\bullet_{(E)^\bullet_\top}\delta^\circ\gamma^\circ\Downarrow$$

From the assumption $\Delta;\Gamma^\circ \vdash \lambda k.(e_1)^\bullet_k \lesssim^{ctx} \lambda k.(e_2)^\bullet_k : \tau^\bullet$ we obtain $(e_2)^\bullet_{(E)^\bullet_\top}\delta^\circ\gamma^\circ\Downarrow$, i.e., $(E[e_2\delta\gamma])^\bullet_\top\Downarrow$. Proposition 20(2) now gives $E[e_2\delta\gamma]\Downarrow$. □

For closed types τ, τ' and for $r \in \mathit{VRel}_\omega(\tau^\circ, \tau')$ we define $r^\perp \in \mathit{Rel}_\omega(\mathit{Val}(\tau^\circ \to \rho) \times \mathit{Stk}(\tau'))$ and $r^{\perp\!\perp} \in \mathit{TRel}_\omega(\tau^\bullet, \tau')$ by:

$$r_i^\perp = \{(k, E') \mid \forall j \le i.\ \forall (v, v') \in r_j.\, k\, v \downarrow_j \Rightarrow\ E'[v'] \downarrow\}$$
$$r_i^{\perp\!\perp} = \{(e, e') \mid \forall j \le i.\ \forall (k, E') \in r_j^\perp.\, \mathtt{let}\ x = e\ \mathtt{in}\ x\, k \downarrow_j \Rightarrow\ E'[e'] \downarrow\}$$

Figure 9: A may-adequate logical relation

7.1. **Logical relation for may-adequacy.** Analogously to the earlier sections, we consider soundness for may and must approximation separately. In this subsection, we begin by proving $(e)^\bullet_\top \downarrow \Rightarrow\ \ e \downarrow$. Figure 9 gives a notion of biorthogonality. Using this notion of biorthogonality, the logical relation for the may-adequacy proof is defined as in the earlier sections:[1] we obtain a relational interpretation $[\![\tau]\!]\,(\vec{r}) \in \mathit{VRel}_\omega(\tau^\circ[\vec{\tau}^\circ/\vec{\alpha}], \tau[\vec{\tau}'/\vec{\alpha}])$ by induction on the type $\vec{\alpha} \vdash \tau$, given closed types $\tau_1, \tau_1', \ldots, \tau_k, \tau_k' \in \mathit{Type}$ and relations $r_1 \in \mathit{VRel}_\omega(\tau_1{}^\circ, \tau_1'), \ldots, r_k \in \mathit{VRel}_\omega(\tau_k{}^\circ, \tau_k')$

$$[\![\alpha_i]\!]\,(\vec{r}) = r_i \qquad [\![\tau_1 \times \tau_2]\!]\,(\vec{r}) = [\![\tau_1]\!]\,(\vec{r}) \times [\![\tau_2]\!]\,(\vec{r})$$
$$[\![\mathbf{1}]\!]\,(\vec{r}) = (\mathit{Id}_{\mathbf{1}})_{n<\omega} \qquad [\![\tau_1 \to \tau_2]\!]\,(\vec{r}) = [\![\tau_1]\!]\,(\vec{r}) \to [\![\tau_2]\!]\,(\vec{r})^{\perp\!\perp}$$
$$[\![\forall\alpha.\tau]\!]\,(\vec{r}) = \forall r.[\![\tau]\!]\,(\vec{r}, r)^{\perp\!\perp} \qquad [\![\mu\alpha.\tau_1 + \ldots + \tau_m]\!]\,(\vec{r}) = \mathit{fix}\, s. \bigcup_j \mathtt{in}_j(\triangleright\, [\![\tau_j]\!]\,(\vec{r}, s))$$

and we define a relation on (open) values and expressions by

$\Delta; \Gamma \vdash v \lhd_\downarrow v' : \tau$ where $\Delta = \vec{\alpha}$

$\Leftrightarrow\ \forall \vec{\tau}, \vec{\tau}'. \forall \vec{r} \in \mathit{VRel}_\omega(\vec{\tau}^\circ, \vec{\tau}'). \forall n < \omega. \forall (\gamma, \gamma') \in [\![\Gamma]\!]\,\vec{r}_n.\ (v[\vec{\tau}^\circ/\vec{\alpha}]\gamma, v'[\vec{\tau}'/\vec{\alpha}]\gamma') \in [\![\tau]\!]\,\vec{r}_n$

and

$\Delta; \Gamma \vdash e \blacktriangleleft_\downarrow e' : \tau$ where $\Delta = \vec{\alpha}$

$\Leftrightarrow\ \forall \vec{\tau}, \vec{\tau}'. \forall \vec{r} \in \mathit{VRel}_\omega(\vec{\tau}^\circ, \vec{\tau}'). \forall n < \omega. \forall (\gamma, \gamma') \in [\![\Gamma]\!]\,\vec{r}_n.\ (e[\vec{\tau}^\circ/\vec{\alpha}]\gamma, e'[\vec{\tau}'/\vec{\alpha}]\gamma') \in [\![\tau]\!]\,\vec{r}_n^{\perp\!\perp}$

In the remainder of this subsection we prove the fundamental theorem for this logical relation.

Proposition 21 (Fundamental property)**.**

- *If* $\Delta; \Gamma \vdash v : \tau$ *then* $\Delta; \Gamma \vdash v^\circ \lhd_\downarrow v : \tau$.
- *If* $\vdash E : \tau \multimap \tau'$ *then* $(\lambda k.(E)^\bullet_k, E) \in [\![\tau]\!]^\perp$
- *If* $\Delta; \Gamma \vdash e : \tau$ *then* $\Delta; \Gamma \vdash \lambda k.(e)^\bullet_k \blacktriangleleft_\downarrow e : \tau$.

In particular, if $\varnothing; \varnothing \vdash e : \tau$ then $(\lambda k.(e)^\bullet_k, e) \in [\![\tau]\!]^{\perp\!\perp}_n$ and $(\top, []) \in [\![\tau]\!]^\perp_n$ hold for all n, and thus $(e)^\bullet_\top \downarrow$ implies $e \downarrow$.

The proof of Proposition21 is by induction on the typing derivations.

7.2. **Logical relation for must-adequacy.** In this subsection, we prove $e \Downarrow \Rightarrow\ (e)^\bullet_\top \Downarrow$. Figure 10 gives the notion of biorthogonality and using this notion of biorthogonality, the logical relation for the must-adequacy proof is defined as for may-adequacy. Thus the relational interpretation $[\![\tau]\!]\,(\vec{r}) \in \mathit{VRel}_{\omega_1}(\tau[\vec{\tau}'/\vec{\alpha}], \tau^\circ[\vec{\tau}^\circ/\vec{\alpha}])$ is defined by induction on the type

[1] It is not a may-adequate relation in the technical sense of Definition 3, since it is not type-indexed: the types of e and e' are not identical, but related via the cps type translation.

For $r \in \mathit{VRel}_{\omega_1}(\tau, \tau^\circ)$ we define $r^\perp \in \mathit{Rel}_{\omega_1}(\mathit{Val}(\mathit{Stk}\tau \times \tau^\circ \to \rho))$ and $r^{\perp\perp} \in \mathit{TRel}_{\omega_1}(\tau, \tau^\bullet)$ by:

$$r^\perp_\alpha = \{(E, k') \mid \forall \beta \leq \alpha.\ \forall (v, v') \in r_\beta.\, E[v] \Downarrow_\beta \Rightarrow\ k\, v' \Downarrow\}$$
$$r^{\perp\perp}_\alpha = \{(e, e') \mid \forall \beta \leq \alpha.\ \forall (E, k') \in r^\perp_\beta.\, E[e] \Downarrow_\beta \Rightarrow\ e'\, k' \Downarrow\}$$

Figure 10: A must-adequate logical relation

$\vec{\alpha} \vdash \tau$, given closed types $\tau_1, \tau_1', \ldots, \tau_k, \tau_k' \in \mathit{Type}$ and relations $r_1 \in \mathit{VRel}_{\omega_1}(\tau_1', \tau_1^\circ), \ldots, r_k \in \mathit{VRel}_{\omega_1}(\tau_k', \tau_k^\circ)$:

$$[\![\alpha_i]\!]\,(\vec{r}) = r_i \qquad [\![\tau_1 \times \tau_2]\!]\,(\vec{r}) = [\![\tau_1]\!]\,(\vec{r}) \times [\![\tau_2]\!]\,(\vec{r})$$
$$[\![\mathbf{1}]\!]\,(\vec{r}) = (\mathit{Id}_{\mathbf{1}})_{\beta<\omega_1} \qquad [\![\tau_1 \to \tau_2]\!]\,(\vec{r}) = [\![\tau_1]\!]\,(\vec{r}) \to [\![\tau_2]\!]\,(\vec{r})^{\perp\perp}$$
$$[\![\forall\alpha.\tau]\!]\,(\vec{r}) = \forall r.[\![\tau]\!]\,(\vec{r}, r)^{\perp\perp} \qquad [\![\mu\alpha.\tau_1 + \ldots + \tau_m]\!]\,(\vec{r}) = \mathit{fix}\, s. \bigcup_j \mathtt{in}_j(\triangleright\, [\![\tau_j]\!]\,(\vec{r}, s))$$

and we define a relation on (open) values and expressions as follows:

$$\Delta; \Gamma \vdash v \lhd_\Downarrow v' : \tau \ \text{ where } \Delta = \vec{\alpha}$$
$$\Leftrightarrow\ \forall \vec{\tau}, \vec{\tau}'. \forall \vec{r} \in \mathit{VRel}_{\omega_1}(\vec{\tau}', \vec{\tau}^\circ). \forall \beta < \omega_1. \forall (\gamma, \gamma') \in [\![\Gamma]\!]\,\vec{r}_\beta.\ (v[\vec{\tau}/\vec{\alpha}]\gamma, v'[\vec{\tau}'/\vec{\alpha}]\gamma') \in [\![\tau]\!]\,\vec{r}_\beta$$

and

$$\Delta; \Gamma \vdash e \blacktriangleleft_\Downarrow e' : \tau \ \text{ where } \Delta = \vec{\alpha}$$
$$\Leftrightarrow\ \forall \vec{\tau}, \vec{\tau}'. \forall \vec{r} \in \mathit{VRel}_{\omega_1}(\vec{\tau}', \vec{\tau}^\circ). \forall \beta < \omega_1. \forall (\gamma, \gamma') \in [\![\Gamma]\!]\,\vec{r}_\beta.\ (e[\vec{\tau}/\vec{\alpha}]\gamma, e'[\vec{\tau}'/\vec{\alpha}]\gamma') \in [\![\tau]\!]\,\vec{r}^{\perp\perp}_\beta$$

Proposition 22 (Fundamental property)**.**

- *If* $\Delta; \Gamma \vdash v : \tau$ *then* $\Delta; \Gamma \vdash v \lhd_\Downarrow v^\circ\tau$.
- *If* $\vdash E : \tau \multimap \tau'$ *then* $(E, \lambda k.(E)^\bullet_k) \in [\![\tau]\!]^\perp$
- *If* $\Delta; \Gamma \vdash e : \tau$ *then* $\Delta; \Gamma \vdash e \blacktriangleleft_\Downarrow \lambda k.(e)^\bullet_k : \tau$.

In particular, if $\varnothing; \varnothing \vdash e : \tau$ then $(e, \lambda k.(e)^\bullet_k) \in [\![\tau]\!]^{\perp\perp}_\beta$ and $([], \top) \in [\![\tau]\!]^\perp_\beta$ hold for all $\beta < \omega_1$, and thus $e \Downarrow$ implies $(e)^\bullet_\top \Downarrow$.

The proof of Proposition 22 is by induction on the typing derivations; the proof case for choice is similar in structure to the proof case shown for choice for must-equivalence earlier and thus uses that we index the relations over ω_1 rather than ω (see the proof of Proposition 12).

References

[1] G. Agha, I. A. Mason, S. F. Smith, and C. L. Talcott. A foundation for actor computation. *J. Funct. Program.*, 7(1):1–72, 1997.

[2] A. W. Appel and D. A. McAllester. An indexed model of recursive types for foundational proof-carrying code. *ACM Trans. Program. Lang. Syst.*, 23(5):657–683, 2001.

[3] K. R. Apt and G. D. Plotkin. Countable nondeterminism and random assignment. *J. ACM*, 33(4):724–767, 1986.

[4] L. Birkedal and R. Harper. Relational interpretations of recursive types in an operational setting. *Inf. Comput.*, 155(1-2):3–63, 1999.

[5] L. Birkedal, B. Reus, J. Schwinghammer, K. Støvring, J. Thamsborg, and H. Yang. Step-indexed Kripke models over recursive worlds. In *POPL*, pages 119–132, 2011.

[6] P. Di Gianantonio, F. Honsell, and G. D. Plotkin. Uncountable limits and the lambda calculus. *Nord. J. Comput.*, 2(2):126–145, 1995.
[7] P. Di Gianantonio and M. Miculan. Unifying recursive and co-recursive definitions in sheaf categories. In *FOSSACS*, pages 136–150, 2004.
[8] D. Dreyer, G. Neis, and L. Birkedal. The impact of higher-order state and control effects on local relational reasoning. In *ICFP*, pages 143–156, 2010.
[9] J. Laird. Bidomains and full abstraction for countable nondeterminism. In *FOSSACS*, pages 352–366, 2006.
[10] S. B. Lassen. *Relational Reasoning about Functions and Nondeterminism*. PhD thesis, University of Aarhus, 1998.
[11] S. B. Lassen and A. Moran. Unique fixed point induction for McCarthy's amb. In *MFCS*, pages 198–208, 1999.
[12] S. B. Lassen and C. Pitcher. Similarity and bisimilarity for countable non-determinism and higher-order functions. *Electr. Notes Theor. Comput. Sci.*, 10, 1997.
[13] P. B. Levy. Infinitary Howe's method. In *CMCS*, pages 85–104, 2006.
[14] I. A. Mason and C. L. Talcott. Equivalence in functional languages with effects. *J. Funct. Program.*, 1(3):287–327, 1991.
[15] A. M. Pitts. Relational properties of domains. *Inf. Comput.*, 127(2):66–90, 1996.
[16] A. M. Pitts. Typed operational reasoning. In B. C. Pierce, editor, *Advanced Topics in Types and Programming Languages*, chapter 7, pages 245–289. MIT Press, 2005.
[17] A. M. Pitts. Step-indexed biorthogonality: a tutorial example. In *Modelling, Controlling and Reasoning About State*, Dagstuhl Seminar Proceedings, 2010.
[18] J. C. Reynolds. On the relation between direct and continuation semantics. In *ICALP*, volume 174 of *LNCS*, pages 141–156, 1974.
[19] D. Sabel and M. Schmidt-Schauß. A call-by-need lambda calculus with locally bottom-avoiding choice: context lemma and correctness of transformations. *Math. Struct. Comp. Sci.*, 18(3):501–553, 2008.

Appendix A. Definitions and proofs

Figure 3 defines the typing judgement $\Delta; \Gamma \vdash e : \tau$, for which the usual operational type soundness properties hold.

Lemma 23 (Evaluation context typing)**.** *$\varnothing; \varnothing \vdash E[e] : \tau$ holds if and only if there is some $\tau' \in \mathit{Type}$ such that $\vdash E : \tau' \multimap \tau$ and $\varnothing; \varnothing \vdash e : \tau'$ hold.*

Definition 24 (Typed values, terms, evaluation contexts and substitutions)**.**

- $\mathit{Val}(\tau) = \{v \mid \varnothing; \varnothing \vdash v : \tau\}$
- $\mathit{Tm}(\tau) = \{e \mid \varnothing; \varnothing \vdash e : \tau\}$
- $\mathit{Stk}(\tau) = \{E \mid \exists \tau' \in \mathit{Type}.\ \vdash E : \tau \multimap \tau'\}$
- $\mathit{Subst}(\Gamma) = \{\gamma : \mathit{dom}(\Gamma) \to \mathit{Val} \mid \forall x{:}\tau \in \Gamma.\ \gamma(x) \in \mathit{Val}(\tau)\}$

Lemma 25 (Substitution)**.** *If $\Delta; \Gamma \vdash e : \tau$ then $\varnothing; \varnothing \vdash e\delta\gamma : \tau\delta$ for all $\delta \in \mathit{Type}^{\Delta}$ and $\gamma \in \mathit{Subst}(\Gamma\delta)$.*

Lemma 26 (Canonical forms)**.**

- *If $v \in \mathit{Val}(\mathbf{1})$ then v is $\langle\rangle$.*
- *If $v \in \mathit{Val}(\tau_1 \times \tau_2)$ then v is of the form $\langle v_1, v_2\rangle$ for $v_i \in \mathit{Val}(\tau_i)$.*
- *If $v \in \mathit{Val}(\tau_1 \to \tau_2)$ then v is of the form $\lambda x.t$ for some x and e.*
- *If $v \in \mathit{Val}(\mu\alpha.\tau_1 + \ldots + \tau_m)$ then v is of the form $\mathtt{in}_j\, v'$ for some $1 \le j \le m$ and $v' \in \mathit{Val}(\tau_j[\mu\alpha.\tau_1 + \ldots + \tau_m/\alpha])$.*
- *If $v \in \mathit{Val}(\forall\alpha.\tau)$ then v is of the form $\Lambda\alpha.e$ for some α and e.*

Proposition 27 (Preservation and progress)**.**

- *If $e \in \mathit{Tm}(\tau)$ and $e \longmapsto e'$ then $e' \in \mathit{Tm}(\tau)$.*
- *If $e \in \mathit{Tm}(\tau) \setminus \mathit{Val}(\tau)$ then $e \longmapsto e'$ for some e'.*

A.1. Contextual equivalence.

Proposition 28 (Existence)**.** *The largest (may-, must-) adequate precongruence exists.*

Proof. We can define the relation as $S = \bigcup\{R \mid R$ compatible and (may-, must-) adequate$\}$. As in [16], we can show that S is an adequate precongruence as follows.

(1) The identity relation $\mathit{Id} = \{(\Delta, \Gamma, e, e, \tau) \mid \Delta; \Gamma \vdash e : \tau\}$ is an adequate congruence, hence $\mathit{Id} \subseteq S$, i.e., S is reflexive.
(2) The adequate relations are closed under taking non-empty unions, hence S is adequate.
(3) The compatible and adequate relations are closed under relation composition, hence S is transitive.
(4) A non-empty union of compatible relations that is transitive is compatible, hence S is compatible.

Clearly S is the largest adequate and compatible relation, since it contains any other such relation. □

Proposition 29 (Weakening). *If* $\Delta;\Gamma \vdash e \lesssim^{log}_{(\downarrow,\Downarrow)} e' : \tau$ *then* $\Delta,\Delta';\Gamma,\Gamma' \vdash e \lesssim^{log}_{(\downarrow,\Downarrow)} e' : \tau$.

Proof. For a type-indexed relation R let R^{wk} be defined by

$$\Delta,\Delta';\Gamma,\Gamma' \vdash e\ R^{wk}\ e' : \tau \ \Leftrightarrow\ \Delta;\Gamma \vdash e\ R\ e' : \tau\ .$$

- If R is reflexive then so is R^{wk}.
- If R is transitive then so is R^{wk}.
- If R is (may-, must-) adequate then so is R^{wk}.
- If R is compatible then so is R^{wk}.

Thus, $\left(\lesssim^{log}_{(\downarrow,\Downarrow)}\right)^{wk}$ is contained in $\lesssim^{log}_{(\downarrow,\Downarrow)}$. □

A.2. **Logical relation for may-approximation.**

Lemma 30 (Substitution). *If* $\Delta,\alpha \vdash \tau$ *and* $\Delta \vdash \tau'$ *then* $[\![\tau[\tau'/\alpha]]\!](\vec{r}) = [\![\tau]\!](\vec{r}, [\![\tau']\!](\vec{r}))$.

Lemma 31 (Extensiveness). *For all* $r \in \mathit{VRel}(\tau,\tau')$, $r \subseteq r^{\perp\perp}$.

Lemma 32 (Monotonicity). *For all* $r,s \in \mathit{VRel}(\tau,\tau')$, *if* $r \subseteq s$ *then* $r^{\perp\perp} \subseteq s^{\perp\perp}$.

Lemma 33 (Application). *If* $(e,e') \in [\![\tau_1]\!]\,\vec{r}_n^{\perp\perp}$ *and* $(v,v') \in [\![\tau_1 \to \tau_2]\!]\,\vec{r}_n$ *then* $(v\,e, v'\,e') \in [\![\tau_2]\!]\,\vec{r}_n^{\perp\perp}$.

Proof. For any $(E,E') \in [\![\tau_2]\!]\,\vec{r}_n^{\perp}$, $(E[v\,[\,]], E'[v'\,[\,]]) \in [\![\tau_1]\!]\,\vec{r}_n^{\perp}$ by Lemma 8. Thus, if $E[v\,e]\downarrow_j$ for $j \le n$ then $E'[v'\,e']\downarrow$. □

Lemma 34 (Compatibility: var). *If* $\Delta \vdash \Gamma$ *and* $x{:}\tau \in \Gamma$ *then* $\Delta;\Gamma \vdash x \lesssim^{log}_{\downarrow} x : \tau$.

Proof. Immediate from the definition and Lemma 31. □

Lemma 35 (Compatibility: unit). *If* $\Delta \vdash \Gamma$ *then* $\Delta;\Gamma \vdash \langle\rangle \lesssim^{log}_{\downarrow} \langle\rangle : \mathbf{1}$.

Proof. Immediate from the definition and Lemma 31. □

Lemma 36 (Compatibility: $\times$-intro). *If* $\Delta;\Gamma \vdash v_1 \lesssim^{log}_{\downarrow} v_1' : \tau_1$ *and* $\Delta;\Gamma \vdash v_2 \lesssim^{log}_{\downarrow} v_2' : \tau_2$ *then* $\Delta;\Gamma \vdash \langle v_1, v_2\rangle \lesssim^{log}_{\downarrow} \langle v_1', v_2'\rangle : \tau_1 \times \tau_2$.

Proof. For simplicity we only consider the case of closed terms. Let $(E,E') \in [\![\tau_1 \times \tau_2]\!]\,\vec{r}_n^{\perp}$. and assume $E[\langle v_1, v_2\rangle]\downarrow_j$ for some $j \le n$. Then $E[(\lambda y.\langle v_1, y\rangle)\,v_2)]\downarrow_{j+1}$. By Lemma 8 it suffices to show $(\lambda y.\langle v_1, y\rangle, \lambda y.\langle v_1', y\rangle) \in [\![\tau_2 \to \tau_1 \times \tau_2]\!]\,\vec{r}_n$, for then $E'[[(\lambda y.\langle v_1', y\rangle)\,v_2')]\downarrow$ and necessarily also $E'[\langle v_1', v_2'\rangle]\downarrow$.

From the definitions it is easy to see that $(\lambda x, y.\langle x, y\rangle, \lambda x, y.\langle x, y\rangle) \in [\![\tau_1 \to \tau_2 \to \tau_1 \times \tau_2]\!]\,\vec{r}_n$, and thus whenever $(E,E') \in [\![\tau_2 \to \tau_1 \times \tau_2]\!]\,\vec{r}_n^{\perp}$ then $(E[(\lambda x, y.\langle x, y\rangle)\,[\,]], E'[(\lambda x, y.\langle x, y\rangle)\,[\,]]) \in [\![\tau_1]\!]\,\vec{r}_{n+1}^{\perp}$ so that the result follows from the assumption that $(v_1, v_1') \in [\![\tau_1]\!]\,\vec{r}_{n+1}^{\perp\perp}$. □

Lemma 37 (Compatibility: $\to$-intro). *If* $\Delta;\Gamma, x{:}\tau_1 \vdash e \lesssim^{log}_{\downarrow} e' : \tau_2$ *then* $\Delta;\Gamma \vdash \lambda x.e \lesssim^{log}_{\downarrow} \lambda x.e' : \tau_1 \to \tau_2$.

Proof. The claim follows from the definition of $[\![\tau_1 \to \tau_2]\!]$ and assumption $\Delta;\Gamma, x{:}\tau_1 \vdash e \lesssim^{log}_{\downarrow} e' : \tau_2$, and Lemma 31. □

Lemma 38 (Compatibility: μ-intro)**.** *If* $\Delta;\Gamma \vdash v \lesssim^{log}_{\downarrow} v' : \tau_j[\mu\alpha.\tau_1 + \ldots + \tau_n/\alpha]$ *and* $1 \leq j \leq n$ *then* $\Delta;\Gamma \vdash \mathtt{in}_j\, v \lesssim^{log}_{\downarrow} \mathtt{in}_j\, v' : \mu\alpha.\tau_1 + \ldots + \tau_n$.

Proof. For simplicity we only consider the case of closed terms. Let τ abbreviate the type $\mu\alpha.\tau_1 + \ldots + \tau_n$. Note that $[\![\tau]\!]\,\vec{r} = \bigcup_j \mathtt{in}_j\big(\triangleright\,[\![\tau_j]\!]\,(\vec{r}, [\![\tau]\!]\,\vec{r})\big) = \bigcup_j \mathtt{in}_j\big(\triangleright\,[\![\tau_j[\tau/\alpha]]\!]\,(\vec{r})\big)$ by definition and the substitution lemma, and that $[\![\tau_j[\tau/\alpha]]\!]\,(\vec{r}) \subseteq \triangleright\,[\![\tau_j[\tau/\alpha]]\!]\,(\vec{r})$. Thus, assuming $(E, E') \in [\![\tau]\!]\,\vec{r}^{\perp}_n$ it follows from Lemma 8 that $(E[(\lambda x.\mathtt{in}_j\, x)\,[]], E'[(\lambda x.\mathtt{in}_j\, x)\,[]]) \in [\![\tau_j[\tau/\alpha]]\!]\,\vec{r}^{\perp}_{n+1}$. Thus, if $E[\mathtt{in}_j\, v]\downarrow_i$ for some $i \leq n$ then $E'[(\lambda x.\mathtt{in}_j\, x)\, v'])\downarrow$ follows from $(v, v') \in [\![\tau_j[\tau/\alpha]]\!]\,\vec{r}^{\perp\perp}_{n+1}$. Therefore we can conclude $E'[\mathtt{in}_j\, v']\downarrow$. □

Lemma 39 (Compatibility: $\forall$-intro)**.** *If* $\Delta, \alpha;\Gamma \vdash e \lesssim^{log}_{\downarrow} e' : \tau$ *then* $\Delta;\Gamma \vdash \Lambda\alpha.e \lesssim^{log}_{\downarrow} \Lambda\alpha.e' : \forall\alpha.\tau$.

Proof. The claim follows from the definition of $[\![\forall\alpha.\tau]\!]$ and assumption $\Delta, \alpha;\Gamma \vdash e \lesssim^{log}_{\downarrow} e' : \tau$, and Lemma 31. □

Lemma 40 (Compatibility: $\times$-elim)**.** *If* $\Delta;\Gamma \vdash v \lesssim^{log}_{\downarrow} v' : \tau_1 \times \tau_2$ *then* $\Delta;\Gamma \vdash \mathtt{proj}_i\, v \lesssim^{log}_{\downarrow} \mathtt{proj}_i\, v' : \tau_i$ *for* $i = 1, 2$.

Proof. For simplicity we only consider the case of closed terms. First observe that we have $(\lambda x.\mathtt{proj}_i\, x, \lambda x.\mathtt{proj}_i\, x) \in [\![\tau_1 \times \tau_2 \to \tau_i]\!]\,\vec{r}_n$. Thus, given evaluation contexts $(E, E') \in [\![\tau_i]\!]\,\vec{r}^{\perp}_n$, $(E[(\lambda x.\mathtt{proj}_i\, x)\,[]], E'[(\lambda x.\mathtt{proj}_i\, x)\,[]]) \in [\![\tau_1 \times \tau_2]\!]\,\vec{r}^{\perp}_{n+1}$ by Lemma 8. So if $E[\mathtt{proj}_i\, v]\downarrow_j$ for $j \leq n$ then $E[(\lambda x.\mathtt{proj}_i\, x)\, v]\downarrow_{j+1}$, hence by $(v, v') \in [\![\tau_1 \times \tau_2]\!]\,\vec{r}^{\perp\perp}_{n+1}$ also $E'[(\lambda x.\mathtt{proj}_i\, x)\, v']\downarrow$ and we can conclude $E'[\mathtt{proj}_i\, v']\downarrow$. □

Lemma 41 (Compatibility: $\to$-elim)**.** *If* $\Delta;\Gamma \vdash v \lesssim^{log}_{\downarrow} v' : \tau_1 \to \tau_2$ *and* $\Delta;\Gamma \vdash e \lesssim^{log}_{\downarrow} e' : \tau_1$ *then* $\Delta;\Gamma \vdash v\, e \lesssim^{log}_{\downarrow} v'\, e' : \tau_2$.

Proof. For simplicity we only consider the case of closed terms. First observe that $(e, e') \in [\![\tau_1]\!]\,\vec{r}_n{}^{\perp\perp}$ by assumption, and that this implies $(\lambda f.f\, e, \lambda f.f\, e') \in [\![(\tau_1 \to \tau_2) \to \tau_2]\!]\,\vec{r}_n$. Thus, given $(E, E') \in [\![\tau_2]\!]\,\vec{r}^{\perp}_n$, $(E[(\lambda f.f\, e)\,[]], E'[(\lambda f.f\, e')\,[]]) \in [\![\tau_1 \to \tau_2]\!]\,\vec{r}^{\perp}_{n+1}$ by Lemma 8. So if $E[v\, e]\downarrow_j$ for $j \leq n$ then $E[(\lambda f.f\, e)\, v]\downarrow_{j+1}$, hence by $(v, v') \in [\![\tau_1 \to \tau_2]\!]\,\vec{r}^{\perp\perp}_n$ also $E'[(\lambda f.f\, e')\, v']\downarrow$ and we can conclude $E'[v'\, e']\downarrow$. □

Lemma 42 (Compatibility: μ-elim)**.** *If* $\tau \equiv \mu\alpha.\tau_1 + \ldots + \tau_m$, $\Delta;\Gamma, x_j{:}\tau_j[\tau/\alpha] \vdash e_j \lesssim^{log}_{\downarrow} e'_j : \tau'$ *for all* $1 \leq j \leq m$ *and* $\Delta;\Gamma \vdash v \lesssim^{log}_{\downarrow} v' : \tau$ *then* $\Delta;\Gamma \vdash \mathtt{case}\ v\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e_j|\ldots) \lesssim^{log}_{\downarrow} \mathtt{case}\ v'\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e'_j|\ldots) : \tau'$.

Proof. For simplicity we only consider the case of closed terms. Note that by definition and the substitution lemma we have $[\![\tau]\!]\,\vec{r} = \bigcup_j \mathtt{in}_j\big(\triangleright\,[\![\tau_j]\!]\,(\vec{r}, [\![\tau]\!]\,\vec{r})\big) = \bigcup_j \mathtt{in}_j(\triangleright\,[\![\tau_j[\tau/\alpha]]\!]\,\vec{r})$. Moreover, $(\lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e_j|\ldots), \lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e'_j|\ldots)) \in [\![\tau \to \tau']\!]\,\vec{r}_n$ for any n. To see this, assume $k \leq n$, let $(a, a') \in [\![\tau]\!]\,\vec{r}_n$ and $(E, E') \in [\![\tau']\!]\,\vec{r}^{\perp}_n$ such that $E[\mathtt{case}\ a\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e_j|\ldots)]\downarrow_k$. By the above observation we have $a = \mathtt{in}_j a_j$ and $a' = \mathtt{in}_j a'_j$ for some $(a_j, a'_j) \in [\![\tau_j[\tau/\alpha]]\!]\,\vec{r}_{k-1}$. From $E[\mathtt{case}\ a\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e_j|\ldots)]\downarrow_k$ we obtain $E[e_j[a_j/x_j]]\downarrow_{k-1}$, and thus the assumption on e_j, e'_j gives $E'[e'_j[a'_j/x_j]]\downarrow$ from which we can conclude $E'[\mathtt{case}\ a'\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e'_j|\ldots)]\downarrow$.

To prove the lemma, assume next that $(E, E') \in [\![\tau']\!]\,\vec{r}^{\perp}_n$. From Lemma 8 we obtain $(E[(\lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e_j|\ldots))\,[]], E'[(\lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e'_j|\ldots))\,[]]) \in$

$[\![\tau]\!]\,\vec{r}^{\,\perp}_{n+1}$. Since $(v, v') \in [\![\tau]\!]\,\vec{r}^{\,\perp\perp}_{n+1}$ by assumption, we obtain that $E[\mathtt{case}\ v\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e_j|\ldots)]\downarrow_n$ implies $E[\mathtt{case}\ v'\ \mathtt{of}(\ldots|\mathtt{in}_j\, x_j.\, e'_j|\ldots)]\downarrow$ as required. □

Lemma 43 (Compatibility: ∀-elim)**.** *If* $\Delta;\Gamma \vdash v \lesssim^{log}_{\downarrow} v' : \forall\alpha.\tau$ *and* $\Delta \vdash \tau'$ *then* $\Delta;\Gamma \vdash v\,\tau' \lesssim^{log}_{\downarrow} v'\,\tau' : \tau[\tau'/\alpha]$.

Proof. We consider the case of closed terms. Let $(E, E') \in [\![\tau[\tau'/\alpha]]\!]\,\vec{r}^{\,\perp}_n$. Note that $(\lambda x.x\,\tau', \lambda x.x\,\tau') \in [\![\forall\alpha.\tau \to \tau[\tau'/\alpha]]\!]_n$ and thus $(E[(\lambda x.x\,\tau')\,[]], E'[(\lambda x.x\,\tau')\,[]]) \in [\![\forall\alpha.\tau]\!]\,\vec{r}^{\,\perp}_{n+1}$ by Lemma 8. If $E[v\,\tau']\downarrow_n$ then $E[(\lambda x.x\,\tau')\,v]\downarrow_{n+1}$. By assumption, $E'[(\lambda x.x\,\tau')\,v']\downarrow$, and hence $E'[v'\,\tau']\downarrow$ follows. □

Lemma 44 (Compatibility: choice)**.** *If* $\Delta \vdash \Gamma$ *then* $\Delta;\Gamma \vdash \mathtt{?} \lesssim^{log}_{\downarrow} \mathtt{?} : \mathtt{nat}$.

Proof. Suppose $(E, E') \in [\![\mathtt{nat}]\!]\,\vec{r}^{\,\perp}_n$ and $E[\mathtt{?}]\downarrow_j$ for some $j \le n$. Then $E[\mathtt{?}] \longmapsto E[\underline{k}]$ and $E[\underline{k}]\downarrow_{j-1}$ for some $k \in \mathbb{N}$. Using Lemma 38, induction on k shows that $(\underline{k}, \underline{k}) \in [\![\mathtt{nat}]\!]\,\vec{r}^{\,\perp\perp}_n$, and thus $E'[\underline{k}]\downarrow$. Hence $E'[\mathtt{?}]\downarrow$. □

A.3. **Logical relation for must-approximation.**

Lemma 45 (Substitution)**.** *If* $\Delta, \alpha \vdash \tau$ *and* $\Delta \vdash \tau'$ *then* $[\![\tau[\tau'/\alpha]]\!]\,(\vec{r}) = [\![\tau]\!]\,(\vec{r}, [\![\tau']\!]\,(\vec{r}))$.

Lemma 46 (Extensiveness)**.** *For all* $r \in \mathit{VRel}(\tau, \tau')$, $r \subseteq r^{\perp\perp}$.

Lemma 47 (Monotonicity)**.** *For all* $r, s \in \mathit{VRel}(\tau, \tau')$, *if* $r \subseteq s$ *then* $r^{\perp\perp} \subseteq s^{\perp\perp}$.

Lemma 48 (Context composition)**.** *If* $(v, v') \in [\![\tau_1 \to \tau_2]\!]\,\vec{r}_\beta$ *and* $(E, E') \in [\![\tau_2]\!]\,\vec{r}^{\,\perp}_\beta$ *then* $(E[v\,[]], E'[v'\,[]]) \in [\![\tau_1]\!]\,\vec{r}^{\,\perp}_{\beta+1}$.

Proof. Let $\nu \le \beta + 1$, $(v_1, v'_1) \in [\![\tau_1]\!]\,\vec{r}_\nu$. Assume $E[v\,v_1]\Downarrow_\nu$. We have $v = \lambda x.e$ and $v' = \lambda x.e'$ and $(\lambda x.e, \lambda x.e') \in [\![\tau_1 \to \tau_2]\!]\,\vec{r}_\beta$ for some x, e, e' and necessarily $E[v\,v_1] \longmapsto E[e[v_1/x]]\Downarrow_{\nu'}$ for $\nu' < \nu$, i.e., $\nu' \le \beta$. By definition, $(e[v_1/x], e'[v'_1/x]) \in [\![\tau_2]\!]\,\vec{r}^{\,\perp\perp}_{\nu'}$. From $(E, E') \in [\![\tau_2]\!]\,\vec{r}^{\,\perp}_\beta$ we obtain $E'[e'[v'_1/x]]\Downarrow$. Thus, $E'[v'\,v'_1]\Downarrow$. □

Lemma 49 (Application)**.** *If* $(e, e') \in [\![\tau_1]\!]\,\vec{r}^{\,\perp\perp}_n$ *and* $(v, v') \in [\![\tau_1 \to \tau_2]\!]\,\vec{r}_n$ *then* $(v\,e, v'\,e') \in [\![\tau_2]\!]\,\vec{r}^{\,\perp\perp}_n$.

Proof. For any $(E, E') \in [\![\tau_2]\!]\,\vec{r}^{\,\perp}_\beta$, $(E[v\,[]], E'[v'\,[]]) \in [\![\tau_1]\!]\,\vec{r}^{\,\perp}_\beta$ by Lemma 48. Thus, if $E[v\,e]\downarrow_\nu$ for $\nu \le \beta$ then $E'[v'\,e']\downarrow$. □

Lemma 50 (Compatibility: var)**.** *If* $\Delta \vdash \Gamma$ *and* $x{:}\tau \in \Gamma$ *then* $\Delta;\Gamma \vdash x \lesssim^{log}_{\Downarrow} x : \tau$.

Proof. Immediate from the definition and Lemma 46. □

Lemma 51 (Compatibility: unit)**.** *If* $\Delta \vdash \Gamma$ *then* $\Delta;\Gamma \vdash \langle\rangle \lesssim^{log}_{\Downarrow} \langle\rangle : \mathbf{1}$.

Proof. Immediate from the definition and Lemma 46. □

Lemma 52 (Compatibility: ×-intro)**.** *If* $\Delta;\Gamma \vdash v_1 \lesssim^{log}_{\Downarrow} v'_1 : \tau_1$ *and* $\Delta;\Gamma \vdash v_2 \lesssim^{log}_{\Downarrow} v'_2 : \tau_2$ *then* $\Delta;\Gamma \vdash \langle v_1, v_2\rangle \lesssim^{log}_{\Downarrow} \langle v'_1, v'_2\rangle : \tau_1 \times \tau_2$.

Proof. Analogous to Lemma 36, using Lemma 48. □

Lemma 53 (Compatibility: $\to$-intro)**.** *If* $\Delta;\Gamma, x{:}\tau_1 \vdash e \lesssim^{log}_{\Downarrow} e' : \tau_2$ *then* $\Delta;\Gamma \vdash \lambda x.e \lesssim^{log}_{\Downarrow} \lambda x.e' : \tau_1 \to \tau_2$.

Proof. The claim follows from the definition of $[\![\tau_1 \to \tau_2]\!]$ and assumption $\Delta;\Gamma, x{:}\tau_1 \vdash e \lesssim^{log}_{\downarrow} e' : \tau_2$, and Lemma 46. □

Lemma 54 (Compatibility: μ-intro)**.** *If* $\Delta;\Gamma \vdash v \lesssim^{log}_{\Downarrow} v' : \tau_j[\mu\alpha.\tau_1 + \ldots + \tau_n/\alpha]$ *and* $1 \le j \le n$ *then* $\Delta;\Gamma \vdash \mathtt{in}_j\, v \lesssim^{log}_{\Downarrow} \mathtt{in}_j\, v' : \mu\alpha.\tau_1 + \ldots + \tau_n$.

Proof. Analogous to Lemma 38, using the facts that $[\![\tau]\!]\,\vec{r} = \bigcup_j \mathtt{in}_j\big(\triangleright\,[\![\tau_j]\!]\,(\vec{r}, [\![\tau]\!]\,\vec{r})\big) = \bigcup_j \mathtt{in}_j\big(\triangleright\,[\![\tau_j[\tau/\alpha]]\!]\,(\vec{r})\big)$ and $[\![\tau_j[\tau/\alpha]]\!]\,(\vec{r}) \subseteq \triangleright\,[\![\tau_j[\tau/\alpha]]\!]\,(\vec{r})$, and using Lemma 48. □

Lemma 55 (Compatibility: $\forall$-intro)**.** *If* $\Delta, \alpha;\Gamma \vdash e \lesssim^{log}_{\Downarrow} e' : \tau$ *then* $\Delta;\Gamma \vdash \Lambda\alpha.e \lesssim^{log}_{\Downarrow} \Lambda\alpha.e' : \forall\alpha.\tau$.

Proof. The claim follows from the definition of $[\![\forall\alpha.\tau]\!]$ and assumption $\Delta, \alpha;\Gamma \vdash e \lesssim^{log}_{\downarrow} e' : \tau$, and Lemma 46. □

Lemma 56 (Compatibility: $\times$-elim)**.** *If* $\Delta;\Gamma \vdash v \lesssim^{log}_{\Downarrow} v' : \tau_1 \times \tau_2$ *then* $\Delta;\Gamma \vdash \mathtt{proj}_i\, v \lesssim^{log}_{\Downarrow} \mathtt{proj}_i\, v' : \tau_i$ *for* $i = 1, 2$.

Proof. Analogous to Lemma 40, using Lemma 48. □

Lemma 57 (Compatibility: $\to$-elim)**.** *If* $\Delta;\Gamma \vdash v \lesssim^{log}_{\Downarrow} v' : \tau_1 \to \tau_2$ *and* $\Delta;\Gamma \vdash e \lesssim^{log}_{\Downarrow} e' : \tau_1$ *then* $\Delta;\Gamma \vdash v\, e \lesssim^{log}_{\Downarrow} v'\, e' : \tau_2$.

Proof. Analogous to Lemma 41, using Lemma 48. □

Lemma 58 (Compatibility: μ-elim)**.** *If* $\tau \equiv \mu\alpha.\tau_1 + \ldots + \tau_m$, $\Delta;\Gamma, x_j{:}\tau_j[\tau/\alpha] \vdash e_j \lesssim^{log}_{\Downarrow} e'_j : \tau'$ *for all* $1 \le j \le m$ *and* $\Delta;\Gamma \vdash v \lesssim^{log}_{\Downarrow} v' : \tau$ *then* $\Delta;\Gamma \vdash \mathtt{case}\ v\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots) \lesssim^{log}_{\Downarrow} \mathtt{case}\ v'\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e'_j \mid \ldots) : \tau'$.

Proof. The proof is analogous to Lemma 42. As before, $[\![\tau]\!]\,\vec{r} = \bigcup_j \mathtt{in}_j\big(\triangleright\,[\![\tau_j]\!]\,(\vec{r}, [\![\tau]\!]\,\vec{r})\big) = \bigcup_j \mathtt{in}_j(\triangleright\,[\![\tau_j[\tau/\alpha]]\!]\,\vec{r})$ and $(\lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots), \lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e'_j \mid \ldots)) \in [\![\tau \to \tau']\!]\,\vec{r}_n$ for any β. To see the latter, assume $\nu \le \beta$, let $(a, a') \in [\![\tau]\!]\,\vec{r}_\nu$ and $(E, E') \in [\![\tau']\!]\,\vec{r}^{\perp}_\nu$ such that $E[\mathtt{case}\ a\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots)] \Downarrow_\nu$. We have $a = \mathtt{in}_j a_j$ and $a' = \mathtt{in}_j a'_j$ for some a_j, a'_j such that $(a_j, a'_j) \in [\![\tau_j[\tau/\alpha]]\!]\,\vec{r}_{\nu'}$ for all $\nu' < \nu$. From the assumption $E[\mathtt{case}\ a\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots)] \Downarrow_\nu$ we obtain $E[e_j[a_j/x_j]] \Downarrow_{\nu'}$ for some $\nu' < \nu \le \beta$, and thus the assumption on e_j, e'_j gives $E'[e'_j[a'_j/x_j]] \Downarrow$. Hence $E'[\mathtt{case}\ a'\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e'_j \mid \ldots)] \Downarrow$.

To prove the lemma, assume next that $(E, E') \in [\![\tau']\!]\,\vec{r}^{\perp}_\beta$. From Lemma 48 we obtain $(E[(\lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots))\,[]], E'[(\lambda x.\mathtt{case}\ x\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e'_j \mid \ldots))\,[]]) \in [\![\tau]\!]\,\vec{r}^{\perp\perp}_{\beta+1}$. Since $(v, v') \in [\![\tau]\!]\,\vec{r}^{\perp\perp}_{\beta+1}$ by assumption, we obtain that $E[\mathtt{case}\ v\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e_j \mid \ldots)] \Downarrow_\beta$ implies $E[\mathtt{case}\ v'\ \mathtt{of}(\ldots \mid \mathtt{in}_j\, x_j.\, e'_j \mid \ldots)] \Downarrow$ as required. □

Lemma 59 (Compatibility: $\forall$-elim)**.** *If* $\Delta;\Gamma \vdash v \lesssim^{log}_{\Downarrow} v' : \forall\alpha.\tau$ *and* $\Delta \vdash \tau'$ *then* $\Delta;\Gamma \vdash v\,\tau' \lesssim^{log}_{\Downarrow} v'\,\tau' : \tau[\tau'/\alpha]$.

Proof. Analogous to Lemma 43, using Lemma 48. □

Lemma 60 (Compatibility: choice)**.** *If* $\Delta \vdash \Gamma$ *then* $\Delta; \Gamma \vdash \mathtt{?} \lesssim^{log}_{\Downarrow} \mathtt{?} : \mathtt{nat}$.

Proof. Suppose $(E, E') \in [\![\mathtt{nat}]\!]\, \vec{r}^{\perp}_{\beta}$ and $E[\mathtt{?}] \Downarrow_\beta$. Then $E[\mathtt{?}] \longmapsto e$ implies that e is of the form $E[\underline{k}]$ and $E[\underline{k}] \Downarrow_{\nu_k}$ for some $k \in \mathbb{N}$ and $\nu_k < \beta$. Using Lemma 54, induction on k shows that $(\underline{k}, \underline{k}) \in [\![\mathtt{nat}]\!]\, \vec{r}^{\perp\perp}_{\nu_k}$, and thus $E'[\underline{k}] \Downarrow$ for all $k \in \mathbb{N}$. Hence $E'[\mathtt{?}] \Downarrow$. □