Step-Indexed Biorthogonality: a Tutorial Example

Andrew Pitts
University of Cambridge Computer Laboratory

1 Introduction

The purpose of this note is to illustrate the use of step-indexing [2] combined with biorthogonality [10, 9] to construct syntactical logical relations. It walks through the details of a syntactically simple, yet non-trivial example: a proof of the "CIU Theorem" for contextual equivalence in the untyped call-by-value λ-calculus with recursively defined functions. I took as inspiration two works: Ahmed's step-indexed syntactic logical relations for recursive types [1] and Benton & Hur's work on compiler correctness that combines biorthogonality with step-indexing [4]. The logical relation constructed here will come as no surprise to those familiar with these works. However, compared with Ahmed, we do not regard biorthogonality as "complex machinery" to be avoided—in my view it simplifies matters; and compared with Benton & Hur, I work entirely with operational semantics and with a high-level language. Both things are true of the recent work by Dreyer *et al* [7]; indeed I believe everything in this note can be deduced from their logical relation for call-by-value System F extended with recursive types, references and continuations. Nevertheless, it seems useful, for tutorial purposes, to extract a specific example of what the combination of step-indexing and biorthogonality can achieve, in as simple yet non-trivial a setting as possible.

Of course there are other ways to prove the CIU theorem for untyped call-by-value λ-calculus; for example, by using Howe's method (see [12]). However, two points about the logical relation constructed here are of interest. First, and the main point of the technique as far as I am concerned, is the way step-indexing is used to break the vicious circle in the mixed-variance specification of the logical relation—see definition (10). Second is the fact that, unlike for some other forms of syntactical logical relation (see [11] for example), no compactness property (also known as an "unwinding theorem") is needed to deal with recursively defined functions—see the proof of Lemma 4.3(ii).

2 Programming language

We use the untyped call-by-value λ-calculus with explicit recursive function definitions. Since we are going use biorthogonality, we use expressions in "A-normal" form and use frame stacks to define termination of call-by-value evaluation. So starting with a fixed,

Dagstuhl Seminar Proceedings 10351
Modelling, Controlling and Reasoning About State
http://drops.dagstuhl.de/opus/volltexte/2010/2806

countably infinite set $\mathbb{V}$ of variables, we define:

$$
\begin{array}{rrcll}
\text{Values} & v \in V & ::= & x, f & \text{variables } (x, f \in \mathbb{V}) \\
 & & \mid & \mathsf{fun}(f\, x = e) & \text{recursively defined function} \\
\text{Expressions} & e \in \Lambda & ::= & v & \text{value} \\
 & & \mid & v\, v & \text{application} \\
 & & \mid & \mathsf{let}\, x = e \,\mathsf{in}\, e & \text{sequencing} \\
\text{Frame stacks} & E \in \Lambda^* & ::= & \mathsf{Id} & \text{empty} \\
 & & \mid & E \circ (x \to e) & \text{non-empty}
\end{array}
$$

We identify values/expressions/frame stacks up to α-equivalence of bound variables (the binding forms being $\mathsf{fun}(f\, x = \_)$, $\mathsf{let}\, x = e \,\mathsf{in}\, \_$ and $E \circ (x \to \_)$).

The finite sets $fv(v)/fv(e)/fv(s)$ of free variables of a value/expression/frame stack are defined as usual. Given a finite subset $\overline{x} \subseteq \mathbb{V}$, we write

$$V(\overline{x}) \triangleq \{v \in V \mid fv(v) \subseteq \overline{x}\} \tag{1}$$

$$\Lambda(\overline{x}) \triangleq \{e \in \Lambda \mid fv(e) \subseteq \overline{x}\} \tag{2}$$

$$\Lambda^*(\overline{x}) \triangleq \{E \in \Lambda^* \mid fv(E) \subseteq \overline{x}\}. \tag{3}$$

Note that $V(\overline{x}) \subseteq \Lambda(\overline{x})$.

Capture-avoiding substitution of values $\overline{v}$ for free variables $\overline{x}$ in an expression e is denoted

$$e[\overline{v}/\overline{x}]$$

and similarly for substitution into values and frame stacks. Given a closed value substitution $\sigma \in V(\emptyset)^{\overline{x}}$

$$e[\sigma]$$

denotes the substituted expression $e[\sigma(x)/x \mid x \in \overline{x}]$.

Definition 2.1 (termination). The relation

$$E \perp_n e \quad (n \in \mathbb{N}, E \in \Lambda^*(\emptyset), e \in \Lambda(\emptyset))$$

says that call-by-value evaluation of the closed expression e with respect to the closed frame stack E terminates properly in at most n steps. It is inductively defined by the rules

$$\frac{}{\mathsf{Id} \perp_n v} \qquad \frac{E \perp_n e[v/x]}{E \circ (x \to e) \perp_{n+1} v}$$

$$\frac{E \perp_n e[v/f, v'/x] \qquad v = \mathsf{fun}(f\, x = e)}{E \perp_{n+1} v\, v'} \qquad \frac{E \circ (x \to e') \perp_n e}{E \perp_{n+1} \mathsf{let}\, x = e \,\mathsf{in}\, e'}$$

Then we define

$$E \perp e \triangleq (\exists n \in \mathbb{N})\ E \perp_n e \tag{4}$$

$$e{\downarrow} \triangleq \mathsf{Id} \perp e. \tag{5}$$

3 Contextual pre-order

$$\overline{x} \vdash e \leq_{\mathrm{ctx}} e' \quad (\overline{x} \subseteq_{\mathrm{fin}} \mathbb{V}, e, e' \in \Lambda(\overline{x}))$$

is the greatest relation (with respect to inclusion) which is **pre-ordered**

- $e \in \Lambda(\overline{x}) \Rightarrow \overline{x} \vdash e \leq_{\mathrm{ctx}} e$
- $\overline{x} \vdash e \leq_{\mathrm{ctx}} e' \wedge \overline{x} \vdash e' \leq_{\mathrm{ctx}} e'' \Rightarrow \overline{x} \vdash e \leq_{\mathrm{ctx}} e''$

compatible

- $\overline{x}, f, x \vdash e \leq_{\mathrm{ctx}} e' \Rightarrow \overline{x} \vdash \mathsf{fun}(f\, x = e) \leq_{\mathrm{ctx}} \mathsf{fun}(f\, x = e')$
- $\overline{x} \vdash v_1 \leq_{\mathrm{ctx}} v_1' \wedge \overline{x} \vdash v_2 \leq_{\mathrm{ctx}} v_2' \Rightarrow \overline{x} \vdash v_1\, v_2 \leq_{\mathrm{ctx}} v_1'\, v_2'$
- $\overline{x} \vdash e_1 \leq_{\mathrm{ctx}} e_1' \wedge \overline{x}, x \vdash e_2 \leq_{\mathrm{ctx}} e_2' \Rightarrow \overline{x} \vdash \mathsf{let}\, x = e_1 \,\mathsf{in}\, e_2 \leq_{\mathrm{ctx}} \mathsf{let}\, x = e_1' \,\mathsf{in}\, e_2'$

and **adequate**

- $\emptyset \vdash e \leq_{\mathrm{ctx}} e' \wedge e{\downarrow} \Rightarrow e'{\downarrow}$.

(It is an exercise to check that the greatest such relation does indeed exist. You can define it more explicitly in terms of contexts if you want to.)

Definition 3.1 (CIU pre-order). The relation

$$e \leq_{\mathrm{ciu}} e' \quad (e, e' \in \Lambda(\emptyset))$$

is defined to hold if $(\forall E \in \Lambda^*(\emptyset))\; E \perp e \Rightarrow E \perp e'$. It is extended to open expressions via closing value substitutions: given $e, e' \in \Lambda(\overline{x})$ we define

$$\overline{x} \vdash e \leq_{\mathrm{ciu}} e' \triangleq (\forall \sigma \in V(\emptyset)^{\overline{x}})\; e[\sigma] \leq_{\mathrm{ciu}} e'[\sigma].$$

We wish to prove the following theorem. We will do so using a certain logical relation constructed in the next section.

Theorem 3.2 (CIU theorem). *$\leq_{\mathrm{ctx}}$ is equal to $\leq_{\mathrm{ciu}}$.*

4 Logical step-indexed relations

Definition 4.1. A **step-indexed relation** (SIR) on a set X is by definition an $\mathbb{N}$-indexed family of sets $R = (R_n \mid n \in \mathbb{N})$ satisfying

$$X \supseteq R_0 \supseteq R_1 \supseteq R_2 \supseteq \cdots \tag{6}$$

We define

$$\blacktriangleleft \in SIR(V(\emptyset) \times V(\emptyset)) \tag{7}$$
$$\triangleleft \in SIR(\Lambda(\emptyset) \times \Lambda(\emptyset)) \tag{8}$$
$$\triangleleft^* \in SIR(\Lambda^*(\emptyset) \times \Lambda^*(\emptyset)) \tag{9}$$

as follows:

$$v \blacktriangleleft_n v' \triangleq (\forall m < n)(\forall v_1, v_1')\ v_1 \blacktriangleleft_m v_1' \Rightarrow e[v/f, v_1/x] \lhd_m e'[v'/f, v_1'/x] \quad (10)$$
$$\text{where } v = \mathsf{fun}(f\,x = e) \text{ and } v' = \mathsf{fun}(f\,x = e')$$

$$e \lhd_n e' \triangleq (\forall m \leq n)(\forall E, E')\ E \lhd_m^* E' \wedge E \perp_m e \Rightarrow E' \perp e' \quad (11)$$

$$E \lhd_n^* E' \triangleq (\forall m \leq n)(\forall v, v')\ v \blacktriangleleft_m v' \wedge E \perp_m v \Rightarrow E' \perp v'. \quad (12)$$

(Thus $\blacktriangleleft_n$ is defined by recursion on n using the auxiliary SIRs $\lhd^*$ and $\lhd$ that are defined directly in terms of $\blacktriangleleft$. It is easy to see that the relations do satisfy the decreasing property (6).)

These relations are extended to open values/expressions/frame stacks via closing value-substitutions as follows. Given closed value substitutions $\sigma, \sigma' \in V(\emptyset)^{\overline{x}}$ on a finite set of variables $\overline{x}$, we define

$$\sigma \blacktriangleleft_n \sigma' \triangleq (\forall x \in \overline{x})\ \sigma(x) \blacktriangleleft_n \sigma'(x). \quad (13)$$

Then for $v, v' \in V(\overline{x})$, $e, e' \in \Lambda(\overline{x})$ and $s, s' \in \Lambda^*(\overline{x})$, we define

$$\overline{x} \vdash v \blacktriangleleft v' \triangleq (\forall n)(\forall \sigma, \sigma' \in V(\emptyset)^{\overline{x}})\ \sigma \blacktriangleleft_n \sigma' \Rightarrow v[\sigma] \blacktriangleleft_n v'[\sigma'] \quad (14)$$

$$\overline{x} \vdash e \lhd e' \triangleq (\forall n)(\forall \sigma, \sigma' \in V(\emptyset)^{\overline{x}})\ \sigma \blacktriangleleft_n \sigma' \Rightarrow e[\sigma] \lhd_n e'[\sigma'] \quad (15)$$

$$\overline{x} \vdash E \lhd^* E' \triangleq (\forall n)(\forall \sigma, \sigma' \in V(\emptyset)^{\overline{x}})\ \sigma \blacktriangleleft_n \sigma' \Rightarrow E[\sigma] \lhd_n^* E'[\sigma']. \quad (16)$$

Lemma 4.2. *Given $n \in \mathbb{N}$, $x \in \mathbb{V}$ and $e, e' \in \Lambda(x)$, suppose*

$$(\forall m \leq n)(\forall v, v' \in V(\emptyset))\ v \blacktriangleleft_m v' \Rightarrow e[v/x] \lhd_m e'[v'/x] \quad (17)$$

holds. Then for all $m \leq n$

$$E \lhd_m^* E' \Rightarrow E \circ (x \to e) \lhd_m^* E' \circ (x \to e') \quad (18)$$

$$e_1 \lhd_m e_1' \Rightarrow \mathsf{let}\, x = e_1 \,\mathsf{in}\, e \lhd_m \mathsf{let}\, x = e_1' \,\mathsf{in}\, e'. \quad (19)$$

Proof. For (18), suppose $E \lhd_m^* E'$, $k \leq m$ and $v \blacktriangleleft_k v'$. If $E \circ (x \to e) \perp_k v$, then ($k > 0$ and) $E \perp_{k-1} e[v/x]$. By hypothesis (17) we have $e[v/x] \lhd_{k-1} e'[v'/x]$. So from $E \lhd_m^* E'$ and $E \perp_{k-1} e[v/x]$ we get $E' \perp e'[v'/x]$ and hence also $E' \circ (x \to e') \perp v'$. Therefore by definition of $\lhd_m^*$, we have $E \circ (x \to e) \lhd_m^* E' \circ (x \to e')$, as required.

For (19), suppose $e_1 \lhd_m e_1'$, $k \leq m$ and $E \lhd_k^* E'$. If $E \perp_k \mathsf{let}\, x = e_1 \,\mathsf{in}\, e$, then ($k > 0$ and) $E \circ (x \to e) \perp_{k-1} e_1$; but by (18) we have $E \circ (x \to e) \lhd_{k-1}^* E' \circ (x \to e')$ and hence $E' \circ (x \to e') \perp e_1'$ and therefore also $E' \perp \mathsf{let}\, x = e' \,\mathsf{in}\, e_1'$. Thus by definition of $\lhd_m$ we have $\mathsf{let}\, x = e_1 \,\mathsf{in}\, e \lhd_m \mathsf{let}\, x = e_1' \,\mathsf{in}\, e'$, as required. □

Lemma 4.3. *(i) If $x \in \overline{x}$, then $\overline{x} \vdash x \blacktriangleleft x$.*

(ii) If $\overline{x}, f, x \vdash e \lhd e'$, then $\overline{x} \vdash \mathsf{fun}(f\,x = e) \blacktriangleleft \mathsf{fun}(f\,x = e')$.

(iii) If $\overline{x} \vdash v \blacktriangleleft v'$, then $\overline{x} \vdash v \lhd v'$.

(iv) If $\overline{x} \vdash v_1 \blacktriangleleft v_1'$ and $\overline{x} \vdash v_2 \blacktriangleleft v_2'$, then $\overline{x} \vdash v_1\, v_2 \lhd v_1'\, v_2'$.

(v) If $\overline{x} \vdash e_1 \lhd e_1'$ and $\overline{x}, x \vdash e_2 \lhd e_2'$, then $\overline{x} \vdash \mathsf{let}\, x = e_1 \,\mathsf{in}\, e_2 \lhd \mathsf{let}\, x = e_1' \,\mathsf{in}\, e_2'$.

(vi) $\overline{x} \vdash \mathsf{Id} \lhd^* \mathsf{Id}$.

(vii) *If* $\overline{x} \vdash E \lhd^* E'$ *and* $\overline{x}, x \vdash e \lhd e'$, *then* $\overline{x} \vdash E \circ (x \to e) \lhd^* E' \circ (x \to e')$.

Proof. (i) This follows directly from (13) and (14).

(ii) Suppose

$$\overline{x}, f, x \vdash e \lhd e'. \tag{20}$$

We prove $(\forall n)(\forall \sigma, \sigma' \in V(\emptyset)^{\overline{x}})\ \sigma \blacktriangleleft_n \sigma' \;\Rightarrow\; \mathsf{fun}(f\,x = e[\sigma]) \blacktriangleleft_n \mathsf{fun}(f\,x = e'[\sigma'])$ by induction on n. So suppose

$$(\forall m < n)(\forall \sigma, \sigma' \in V(\emptyset)^{\overline{x}})\ \sigma \blacktriangleleft_m \sigma' \;\Rightarrow\; \mathsf{fun}(f\,x = e[\sigma]) \blacktriangleleft_m \mathsf{fun}(f\,x = e'[\sigma']) \tag{21}$$

and that $\sigma \blacktriangleleft_n \sigma' \in V(\emptyset)^{\overline{x}}$. Writing $v \triangleq \mathsf{fun}(f\,x = e[\sigma])$ and $v' \triangleq \mathsf{fun}(f\,x = e'[\sigma'])$, we have to show that $v \blacktriangleleft_n v'$. By definition of $\blacktriangleleft_n$ this means that we have to prove for all $m < n$ and $v_1 \blacktriangleleft_m v_1'$ that $e[\sigma][v/f, v_1/x] \lhd_m e'[\sigma'][v'/f, v_1'/x]$.

So suppose $m < n$ and $v_1 \blacktriangleleft_m v_1'$. Since $\sigma \blacktriangleleft_n \sigma'$ we also have $\sigma \blacktriangleleft_m \sigma'$; and hence from the induction hypothesis (21) we get $v \blacktriangleleft_m v'$. Then from (20) we get $e[\sigma][v/f, v_1/x] \lhd_m e[\sigma'][v'/f, v_1'/x]$, as required.

(iii) It suffices to show that $\blacktriangleleft_n \subseteq \lhd_n$. Suppose $v \blacktriangleleft_n v'$. For any $m \leq n$ and $E \lhd_m^* E'$, since $v \blacktriangleleft_m v'$ holds, by definition of $\lhd_m^*$ we have $E \perp_m v \Rightarrow E' \perp v'$. Hence by definition of $\lhd$, we have $v \lhd_m v'$, as required.

(iv) It suffices to show for all n that if $v \blacktriangleleft_n v'$ and $v_1 \blacktriangleleft_n v_1'$, then $v\,v_1 \lhd_n v'\,v_1'$. By definition of $\lhd_n$, this means that we have to prove for all $m \leq n$ and $E \lhd_m^* E'$ that $E \perp_m v\,v_1$ implies $E' \perp v'\,v_1'$.

So suppose $m \leq n$ and $E \lhd_m^* E'$ that $E \perp_m v\,v_1$. Let $v = \mathsf{fun}(f\,x = e)$ and $v' = \mathsf{fun}(f\,x = e')$. Then by definition of $\_ \perp_m \_$ we must have ($m > 0$ and) $E \perp_{m-1} e[v/f, v_1/x]$. Since $v \blacktriangleleft_n v'$, $m - 1 < n$ and $v_1 \blacktriangleleft_{m-1} v_1'$, by definition of $\blacktriangleleft_n$ we have $e[v/f, v_1/x] \lhd_{m-1} e'[v'/f, v_1'/x]$. Then since $E \lhd_m^* E'$, we get $E' \perp e'[v'/f, v_1'/x]$ and hence also $E' \perp v'\,v_1'$, as required.

(v) This is a corollary of Lemma 4.2.

(vi) Note that $\mathsf{Id} \lhd_n^* \mathsf{Id}$ holds because for all $v \in V(\emptyset)$, $\mathsf{Id} \perp v$ holds.

(vii) This is a corollary of Lemma 4.2.

□

Remark 4.4. Definition (10) is delicate. It seems that one cannot replace it with the simpler clause

$$v \blacktriangleleft_n v' \;=\; (\forall m < n)(\forall v_1, v_1')\ v_1 \blacktriangleleft_m v_1' \;\Rightarrow\; v\,v_1 \lhd_m v'\,v_1'$$

and still prove part (iv) of Lemma 4.3.

Theorem 4.5 (Fundamental property of the logical relation). *For all* $v \in V(\overline{x})$, $e \in \Lambda(\overline{x})$ *and* $E \in \Lambda^*(\overline{x})$

$$\overline{x} \vdash v \blacktriangleleft v, \quad \overline{x} \vdash e \lhd e \quad \text{and} \quad \overline{x} \vdash E \lhd^* E.$$

Proof. By induction on the structure of $v/e/E$ using Lemma 4.3. □

Lemma 4.6. *If* $\overline{x} \vdash e \lhd e'$ *and* $\overline{x} \vdash e' \leq_{\mathrm{ciu}} e''$, *then* $\overline{x} \vdash e \lhd e''$.

Proof. It suffices to show

$$e \lhd_n e' \;\wedge\; e \leq_{\mathrm{ciu}} e'' \;\Rightarrow\; e \lhd_n e''$$

and this follows immediately from the definition of $\lhd_n$ in (11) and Definition 3.1. □

Lemma 4.7. *If $\emptyset \vdash e \lhd e'$, then $e \leq_{\mathrm{ciu}} e'$.*

Proof. Suppose $\emptyset \vdash e \lhd e'$. For any $E \in \Lambda^*(\emptyset)$ we have to show $E \perp e \;\Rightarrow\; E \perp e'$. By Theorem 4.5 we have $\emptyset \vdash E \lhd^* E$. So if $E \perp e$ holds, then by definition of $\perp$, we have $E \perp_n e$ for some n; and since $E \lhd^*_n E$ and $e \lhd_n e'$, by definition of $\lhd_n$ we do indeed have $E \perp e'$. □

Theorem 4.8. *$\lhd$ is equal to $\leq_{\mathrm{ciu}}$.*

Proof. For any closed value substitution $\sigma \in V(\emptyset)^{\overline{x}}$ from Theorem 4.5 we have $(\forall n \in \mathbb{N})\ \sigma \blacktriangleleft_n \sigma$. So if $\overline{x} \vdash e \lhd e'$, then $(\forall n \in \mathbb{N})\ e[\sigma] \lhd_n e'[\sigma]$. Hence by Lemma 4.7 we have $e[\sigma] \leq_{\mathrm{ciu}} e'[\sigma]$. Therefore $\overline{x} \vdash e \leq_{\mathrm{ciu}} e'$ holds.

Conversely, if $\overline{x} \vdash e \leq_{\mathrm{ciu}} e'$, since by Theorem 4.5 we have $\overline{x} \vdash e \lhd e$, it follows from Lemma 4.6 that $\overline{x} \vdash e \lhd e'$. □

Lemma 4.9. *For all $n \in \mathbb{N}$, $E, E' \in \Lambda^*(\emptyset)$, $v, v' \in V(\emptyset)$ and $f \in \mathbb{V}$*

$$E \lhd^*_n E' \;\wedge\; v \blacktriangleleft_n v' \;\Rightarrow\; E \circ (f \to f\,v) \lhd^*_{n+2} E' \circ (f \to f\,v').$$

Proof. Suppose $m \leq n+2$, $v_1 \blacktriangleleft_m v'_1$, with $v_1 = \mathsf{fun}(f\,x = e)$ and $v'_1 = \mathsf{fun}(f\,x = e')$ say, and that $E \circ (f \to f\,v) \perp_m v_1$. We have to show that $E' \circ (f \to f\,v') \perp v'_1$.

Since $E \circ (f \to f\,v) \perp_m v_1$, by definition of $\perp$ it must be the case that $m \geq 2$ and $E \perp_{m-2} e[v_1/f, v/x]$. Note that $m-2 \leq n$, so $E \lhd^*_{m-2} E'$ and $v \blacktriangleleft_{m-2} v'$; also $m-2 < m$, so by definition of $v_1 \blacktriangleleft_m v'_1$ we have $e[v_1/f, v/x] \lhd_{m-2} e'[v'_1/f, v'/x]$. Therefore from $E \perp_{m-2} e[v_1/f, v/x]$ we get $E' \perp e'[v'_1/f, v'/x]$ and hence also $E' \circ (f \to f\,v') \perp v'_1$, as required. □

Corollary 4.10. *For all $n \in \mathbb{N}$ and $v, v' \in V(\emptyset)$*

$$v \lhd_{n+1} v' \;\Rightarrow\; v \blacktriangleleft_n v' \tag{22}$$

and hence in particular

$$\emptyset \vdash v \lhd v' \;\Rightarrow\; \emptyset \vdash v \blacktriangleleft v'. \tag{23}$$

Proof. Suppose $v = \mathsf{fun}(f\,x = e)$, $v' = \mathsf{fun}(f\,x = e')$ and $v \lhd_{n+1} v'$. To see that $v \blacktriangleleft_n v'$ we have to show for any $m < n$ and $v_1 \blacktriangleleft_m v'_1$ that $e[v/f, v_1/x] \lhd_m e'[v'/f, v'_1/x]$; that is, for any $k \leq m$ and $E \lhd^*_k E'$, $E \perp_k e[v/f, v_1/x]$ implies $E' \perp e'[v'/f, v'_1/x]$.

But if $E \perp_k e[v/f, v_1/x]$, then $E \circ (f \to f\,v_1) \perp_{k+2} v$. Note that $k+2 \leq n+1$; so by assumption we have $v \lhd_{k+2} v'$; and since $k \leq m$ we can apply Lemma 4.9 to get $E \circ (f \to f\,v_1) \lhd^*_{k+2} E' \circ (f \to f\,v'_1)$. Therefore by definition of $\lhd_{k+2}$, from $E \circ (f \to f\,v_1) \perp_{k+2} v$ we get $E' \circ (f \to f\,v'_1) \perp v'$ and hence also $E' \perp e'[v'/f, v'_1/x]$, as required. □

Lemma 4.11. *$\leq_{\mathrm{ciu}}$ is contained in $\leq_{\mathrm{ctx}}$.*

Proof. It suffices to show that $\leq_{\text{ciu}}$ is an adequate, compatible pre-order, because $\leq_{\text{ctx}}$ is the greatest such. It is immediate from its definition that $\leq_{\text{ciu}}$ is an adequate pre-order. For its compatibility properties we use the fact that it coincides with $\triangleleft$ (Theorem 4.8). Compatibility with $\mathsf{fun}(f\,x = \_)$ is thus a consequence of parts (ii) and (iii) of Lemma 4.3; and compatibility with $\mathsf{let}\,x = \_ \;\mathsf{in}\; \_$ is part (v) of that lemma. Compatibility with application, that is, the property

$$\overline{x} \vdash v \leq_{\text{ciu}} v' \;\wedge\; \overline{x} \vdash v_1 \leq_{\text{ciu}} v_1' \;\Rightarrow\; \overline{x} \vdash v\,v_1 \leq_{\text{ciu}} v'\,v_1' \tag{24}$$

is not a direct consequence of part (iv) of the lemma even though we know that $\leq_{\text{ciu}}$ coincides with $\triangleleft$. However, note that to prove (24) it suffices to prove the particular case when $\overline{x} = \emptyset$, because of the way $\leq_{\text{ciu}}$ is defined for open expressions; and by Theorem 4.8 this is equivalent to proving

$$\emptyset \vdash v \triangleleft v' \;\wedge\; \emptyset \vdash v_1 \triangleleft v_1' \;\Rightarrow\; \emptyset \vdash v\,v_1 \triangleleft v'\,v_1'.$$

Now we can apply Corollary 4.10 to deduce this from Lemma 4.3(iv). □

Lemma 4.12 (value-substitutivity for $\leq_{\text{ctx}}$). *If $\overline{x}, x \vdash e \leq_{\text{ctx}} e'$ and $\overline{x} \vdash v \leq_{\text{ctx}} v'$, then $\overline{x} \vdash e[v/x] \leq_{\text{ctx}} e'[v/x]$.*

Proof. If $\overline{x}, x \vdash e \leq_{\text{ctx}} e'$ and $\overline{x} \vdash v \leq_{\text{ctx}} v'$, then by the compatibility properties of $\leq_{\text{ctx}}$ we have $\overline{x} \vdash (\mathsf{fun}(f\,x = e))\,v \leq_{\text{ctx}} (\mathsf{fun}(f\,x = e'))\,v'$, where $f \notin \overline{x}, x$. So the result follows by transitivity on $\leq_{\text{ctx}}$ once we know

$$\overline{x} \vdash e[v/x] \leq_{\text{ctx}} (\mathsf{fun}(f\,x = e))\,v \quad \text{and} \quad \overline{x} \vdash (\mathsf{fun}(f\,x = e'))\,v' \leq_{\text{ctx}} e'[v'/x].$$

It is easy to see from the definition of the CIU pre-order that these hold up to $\leq_{\text{ciu}}$; so we can apply Lemma 4.11. □

Proof of CIU Theorem 3.2. We have already shown that $\leq_{\text{ciu}}$ is contained in $\leq_{\text{ctx}}$ (Lemma 4.11). For the converse, in view of Lemma 4.12 it suffices to show that $\emptyset \vdash e \leq_{\text{ctx}} e'$ implies $e \leq_{\text{ciu}} e'$. Given $\emptyset \vdash e \leq_{\text{ctx}} e'$, we prove $E \perp e \;\Rightarrow\; E \perp e'$ by induction on the length of frame stack $E \in \Lambda^*(\emptyset)$.

The base case $E = \mathsf{Id}$ holds because $\leq_{\text{ctx}}$ is an adequate relation.

For the induction step for a non-empty frame stack $E \circ (x \to e_1)$, note that

$$\emptyset \vdash \mathsf{let}\,x = e \;\mathsf{in}\; e_1 \leq_{\text{ctx}} \mathsf{let}\,x = e' \;\mathsf{in}\; e_1$$

holds by the compatibility (and pre-order) property of $\leq_{\text{ctx}}$. So the result follows from $(\forall E, e, e')\; E \circ (x \to e') \perp e \;\Leftrightarrow\; E \perp \mathsf{let}\,x = e \;\mathsf{in}\; e'$, which is a consequence of the definition of $\perp$. □

5 Toward abstract sense

In this note I have purposely kept things as concrete as possible. However, to understand what is "really" going on (as a category theorist would say) and apply step-indexing techniques in more complicated situations, the level of mathematical sophistication needs to rise. We need to do some category theory.

Recall $SIR(X)$ from Definition 4.1. It becomes a complete Heyting algebra once endowed with the ordering

$$R \leq R' \triangleq (\forall n \in \mathbb{N})\ R_n \subseteq R'_n. \tag{25}$$

Furthermore, given a function $f : X \to Y$, taking inverse images of subsets along f yields a morphism of complete Heyting algebras $f^* : SIR(Y) \to SIR(X)$:

$$(f^* R)_n \triangleq \{x \in X \mid f(x) \in R_n\}.$$

This makes $SIR(\_)$ into a $\mathcal{S}et$-based tripos [8]; indeed it is isomorphic to the tripos of H-valued sets where H is the complete linear order

$$\mathsf{F} < p_0 < p_1 < p_2 < \cdots < \mathsf{T}.$$

(Exercise: show that $SIR(X)$ is isomorphic to H^X, naturally in X.) Thus the topos associated with the tripos $SIR(\_)$ is the category $\mathcal{S}h(H)$ of sheaves on the complete Heyting algebra H. The internal higher-order logic of this topos is probably a good place to study step-indexing from an abstract point of view. Here are two little pieces of evidence.

1. *The structure of implication.* Binary meet in $SIR(X)$ is given by index-wise binary intersection. The Heyting implication $R \to R'$ of two elements of $SIR(X)$ by definition satisfies

   $$(\forall R'')\ R'' \leq R \to R' \ \Leftrightarrow\ R'' \wedge R \leq R'$$

   and a simple calculation shows that is it given by

   $$(R \to R')_n \triangleq \{x \in S \mid (\forall m \leq n)\ x \in R_m \Rightarrow x \in R'_m\} \quad (n \in \mathbb{N}).$$

   Compare this with the various uses of the bounded quantifier $(\forall m \leq n)(\_)$ in Sect. 4.

2. *The "later" modality.* The crucial definition Sect. 4 is (10); it makes use of the bounded quantifier $(\forall m < n)(\_)$. Since Appel *et al* [3], we realize that this has to do with the provability logic of Gödel-Löb. The monotone function $\diamond : SIR(\_) \to SIR(\_)$ is given by

   $$(\diamond R)_n \triangleq \bigcap_{m<n} R_m = \begin{cases} X & \text{if } n = 0 \\ R_{n-1} & \text{if } n > 0. \end{cases}$$

   For example, we can restate Corollary 4.10 as saying $\diamond(\lhd) \leq \blacktriangleleft$.

   The $\diamond$ operation satisfies the **Gödel-Löb rule**:

   $$\diamond R \leq R \ \Rightarrow\ R = \top.$$

   (For if $\diamond R \leq R$, then $X \subseteq R_0$ and $R_{n-1} \subseteq R_n$ for $n > 0$; in view of (6), it follows by induction on n that $R_n = X$ for all n; and thus $R = \top$.) The aim is to replace induction over step-indexes by use of this rule; see [6] for how this can work in practice.

6 Conclusion

Where is this development headed? The aim is similar to recent work of Birkedal *et al* on relational methods using complete, 1-bounded ultrametric (CBU) spaces. In practice it seems that only "bisected" CBU spaces [5, Defintion 2.2] are needed; and the latter are closely connected with $SIR(\_)$. It should be possible to develop a theory of types and relations defined by "guarded recursion" in the tripos $SIR(\_)$ (or maybe better, in the topos $\mathcal{S}h(H)$) so that the construction of $\blacktriangleleft$ in Sect. 4 and its fundamental properties, such as Lemma 4.3 and Corollary 4.10, fall out automatically from a fixed point specification.

References

[1] A. Ahmed. Step-indexed syntactic logical relations for recursive and quantified types. In P. Sestoft, editor, *Programming Languages and Systems, 15th European Symposium on Programming, ESOP 2006, Vienna, Austria*, volume 3924 of *Lecture Notes in Computer Science*, pages 69–83. Springer, 2006.

[2] A. Appel and D. McAllester. An indexed model of recursive types for foundational proof-carrying code. *Transactions on Programming Languages and Systems*, 23(5):657–683, 2001.

[3] A. W. Appel, P.-A. Melliès, C. D. Richards, and J. Vouillon. A very modal model of a modern, major, general type system. In *POPL '07: Proceedings of the 34th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 109–122, New York, NY, USA, 2007. ACM.

[4] N. Benton and C.-K. Hur. Biorthogonality, step-indexing and compiler correctness. In *ICFP '09: Proceedings of the 14th ACM SIGPLAN International Conference on Functional Programming*, pages 97–108, New York, NY, USA, 2009. ACM.

[5] L. Birkedal, B. Reus, J. Schwinghammer, K. Støvring, J. Thamsborg, and H. Yang. Step-indexed kripke models over recursive worlds. In *POPL '11: Proceedings of the 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages ?–?, New York, NY, USA, 2011. ACM.

[6] D. Dreyer, A. Ahmed, and L. Birkedal. Logical step-indexed logical relations. Submitted for publication, January 2010.

[7] D. Dreyer, G. Neis, and L. Birkedal. The impact of higher-order state and control effects on local relational reasoning. In *ICFP 2010: Proceedings of the 15th ACM SIGPLAN International Conference on Functional Programming*, pages 143–156, New York, NY, USA, 2010. ACM.

[8] J. M. E. Hyland, P. T. Johnstone, and A. M. Pitts. Tripos theory. *Math. Proc. Cambridge Philos. Soc.*, 88:205–232, 1980.

[9] P.-A. Melliès and J. Vouillon. Recursive polymorphic types and parametricity in an operational framework. In *20th Annual IEEE Symposium on Logic in Computer Science (LICS'05)*, pages 82–91. IEEE Computer Society Press, June 2005.

[10] A. M. Pitts. Parametric polymorphism and operational equivalence. *Mathematical Structures in Computer Science*, 10:321–359, 2000.

[11] A. M. Pitts. Typed operational reasoning. In B. C. Pierce, editor, *Advanced Topics in Types and Programming Languages*, chapter 7, pages 245–289. The MIT Press, 2005.

[12] A. M. Pitts. Howe's method for proving congruence properties in higher-order languages. In D. Sangiorgi and J. Rutten, editors, *Bisimulation and Coinduction: Advanced Topics*, pages ?–? Cambridge University Press, 2011. To appear.