Logical Step-Indexed Logical Relations

Derek Dreyer

Max Planck Institute for Software Systems
Saarbrücken, Germany

LICS 2009
UCLA
August 12, 2009

Joint work with Amal Ahmed and Lars Birkedal

Logical Relations

$$
\begin{aligned}
\mathcal{V}[\![\textsf{nat}]\!]\rho &= \{(n,n) \mid n \in \mathbb{N}\} \\[2ex]
\mathcal{V}[\![\tau' \to \tau'']\!]\rho &= \{(\lambda x.e_1, \lambda x.e_2) \mid \\
&\qquad \forall v_1, v_2. \\
&\qquad\quad (v_1, v_2) \in \mathcal{V}[\![\tau']\!]\rho \implies \\
&\qquad\quad (e_1[v_1/x], e_2[v_2/x]) \in \mathcal{E}[\![\tau'']\!]\rho\} \\[2ex]
\mathcal{V}[\![\exists\alpha.\,\tau]\!]\rho &= \{(\texttt{pack}\,\tau_1, v_1\ \textsf{as}\ \cdots, \texttt{pack}\,\tau_2, v_2\ \textsf{as}\ \cdots) \mid \\
&\qquad \exists\chi \in \mathbf{Rel}(\tau_1, \tau_2). \\
&\qquad\quad (v_1, v_2) \in \mathcal{V}[\![\tau]\!]\rho, \alpha \mapsto (\tau_1, \tau_2, \chi)\} \\[2ex]
\mathcal{V}[\![\alpha]\!]\rho &= \chi \quad \text{where } \rho(\alpha) = (\tau_1, \tau_2, \chi)
\end{aligned}
$$

Logical Relations for Recursive Types?

$$\begin{aligned}\mathcal{V}\,[\![\mu\alpha.\,\tau]\!]\,\rho \;=\; \{&(\texttt{fold}\,v_1, \texttt{fold}\,v_2) \mid \\ &\qquad (v_1, v_2) \in \mathcal{V}\,[\![\tau[\mu\alpha.\,\tau/\alpha]]\!]\,\rho\}\end{aligned}$$

Logical Relations for Recursive Types?

$$\mathcal{V}\,[\![\mu\alpha.\,\tau]\!]\,\rho \;=\; \{(\texttt{fold}\,v_1, \texttt{fold}\,v_2) \mid (v_1, v_2) \in \mathcal{V}\,[\![\tau[\mu\alpha.\,\tau/\alpha]]\!]\,\rho\}$$

Problem: The definition is no longer well-founded!

Step-Indexed Logical Relations (Appel-McAllester ’01)

Idea: Index logical relations by $n \in \mathbb{N}$ representing "the number of steps left until the clock runs out."

- Two terms are related "infinitely" iff they are n-related (for all n).

$$\begin{aligned}\mathcal{V}\,[\![\mu\alpha.\,\tau]\!]\,\rho \;=\; \{&(n, \texttt{fold}\,v_1, \texttt{fold}\,v_2) \mid \\ &(n-1, v_1, v_2) \in \mathcal{V}\,[\![\tau[\mu\alpha.\,\tau/\alpha]]\!]\,\rho\}\end{aligned}$$

Intuitively, this makes sense because it takes a step of computation to extract v_i from $\texttt{fold}\,v_i$.

Advantages of Step-Indexed Logical Relations

Easy to develop using only elementary mathematical constructions.

Applicable to “difficult” languages, *e.g.*, with higher-order state:

- Imperative self-adjusting computation (Acar *et al.*, POPL’08)
- Representation independence for “generative” ADTs (POPL’09)
- Parametricity in the presence of dynamic typing (ICFP’09)
- Compiler correctness (Benton *et al.*, *e.g.*, TLDI’09, ICFP’09)
- …

Comparison With Other Approaches

With more mathematically sophisticated approaches (*e.g.*, minimal invariance, FM-cpos, ultra-metric spaces):

✘ Hard to construct, not as (obviously) widely applicable

With step-indexed logical relations:

✔ Easy to construct, widely applicable

Comparison With Other Approaches

With more mathematically sophisticated approaches
(*e.g.*, minimal invariance, FM-cpos, ultra-metric spaces):

- ✘ Hard to construct, not as (obviously) widely applicable
- ✔ Easy to develop high-level equational proof principles

With step-indexed logical relations:

- ✔ Easy to construct, widely applicable
- ✘ Hard to develop high-level equational proof principles

You get what you pay for!

Problem #1: Step-Index Arithmetic Pervades Proofs

Steps make constructing the model easy,
but the *user* of the model shouldn't have to deal with them.

- Important to develop clean, abstract, step-free proof principles

Problem #1: Step-Index Arithmetic Pervades Proofs

Steps make constructing the model easy,
but the *user* of the model shouldn't have to deal with them.

- Important to develop clean, abstract, step-free proof principles

E.g. Appel-McAllester claim this extensionality property:

- f_1 and f_2 are infinitely related (*e.g.*, related for any # of steps) *iff* for all v_1 and v_2 that are infinitely related, $f_1 v_1$ and $f_2 v_2$ are, too.

Problem #1: Step-Index Arithmetic Pervades Proofs

Steps make constructing the model easy,
but the *user* of the model shouldn't have to deal with them.

- Important to develop clean, abstract, step-free proof principles

E.g. Appel-McAllester claim this extensionality property:

- f_1 and f_2 are infinitely related (*e.g.*, related for any # of steps) *iff* for all v_1 and v_2 that are infinitely related, $f_1 v_1$ and $f_2 v_2$ are, too.

Unfortunately, it is false!

- In fact, f_1 and f_2 are infinitely related *iff*, for any step level n, for all v_1 and v_2 that are n-related, $f_1 v_1$ and $f_2 v_2$ are, too.

Problem #2: Lack of Equational Proof Principles

Step-indexed logical relations are fundamentally *asymmetric*, *i.e.*, they model approximation ($\leq$), not equivalence ($\equiv$).

- We can define $e_1 \equiv e_2$ to mean $e_1 \leq e_2 \wedge e_2 \leq e_1$.

Problem #2: Lack of Equational Proof Principles

Step-indexed logical relations are fundamentally *asymmetric*, *i.e.*, they model approximation ($\leq$), not equivalence ($\equiv$).

- We can define $e_1 \equiv e_2$ to mean $e_1 \leq e_2 \wedge e_2 \leq e_1$.

We would like a symmetric extensionality principle, *e.g.*,

- $f_1 \equiv f_2$ iff $\forall v_1, v_2$. we have that $v_1 \equiv v_2$ implies $f_1 v_1 \equiv f_2 v_2$.

Problem #2: Lack of Equational Proof Principles

Step-indexed logical relations are fundamentally *asymmetric*, *i.e.*, they model approximation ($\leq$), not equivalence ($\equiv$).

- We can define $e_1 \equiv e_2$ to mean $e_1 \leq e_2 \wedge e_2 \leq e_1$.

We would like a symmetric extensionality principle, *e.g.*,

- $f_1 \equiv f_2$ iff $\forall v_1, v_2$. we have that $v_1 \equiv v_2$ implies $f_1 v_1 \equiv f_2 v_2$.

But even ignoring Problem #1, this is false:

- To show $f_1 \equiv f_2$, we must show that $v_1 \leq v_2$ implies $f_1 v_1 \leq f_2 v_2$, and that $v_2 \leq v_1$ implies $f_2 v_2 \leq f_1 v_1$.

Our Contributions

Define a relational modal logic, LSLR, for expressing step-indexed logical relations without mentioning steps.

Define a step-free logical relation in LSLR for reasoning about program (in-)equivalence in System F + recursive types.

Show logical relation is sound w.r.t. contextual equivalence by defining a suitable "step-indexed" model of LSLR.

Develop a set of useful derivable rules concerning the logical relation.

Demonstrate the effectiveness of our approach by proving several representative examples of contextual equivalences from the literature.

Outline

1. The Language F^{μ}
2. The Logic LSLR
3. Encoding a Logical Relation for F^{μ} in LSLR
4. Derivable Rules

Outline

1. The Language F^μ
2. The Logic LSLR
3. Encoding a Logical Relation for F^μ in LSLR
4. Derivable Rules

The Language F^μ

$$
\begin{array}{llll}
\textit{Types} & \tau & ::= & \alpha \mid \textsf{unit} \mid \textsf{int} \mid \textsf{bool} \mid \tau_1 \times \tau_2 \mid \tau_1 + \tau_2 \mid \\
 & & & \tau_1 \rightarrow \tau_2 \mid \forall \alpha.\, \tau \mid \exists \alpha.\, \tau \mid \mu \alpha.\, \tau \\
\textit{Prim Ops} & o & ::= & + \mid - \mid = \mid < \mid \leq \mid \ldots \\
\textit{Terms} & e & ::= & x \mid () \mid \pm n \mid o(e_1, \ldots, e_n) \mid \\
 & & & \texttt{true} \mid \texttt{false} \mid \texttt{if}\, e\, \texttt{then}\, e_1\, \texttt{else}\, e_2 \mid \\
 & & & \langle e_1, e_2 \rangle \mid \texttt{fst}\, e \mid \texttt{snd}\, e \mid \texttt{inl}_\tau\, e \mid \texttt{inr}_\tau\, e \mid \\
 & & & \texttt{case}\, e\, \texttt{of}\, \texttt{inl}\, x_1 \Rightarrow e_1 \mid \texttt{inr}\, x_2 \Rightarrow e_2 \mid \\
 & & & \lambda x : \tau.\, e \mid e_1\, e_2 \mid \Lambda \alpha.\, e \mid e\, [\tau] \mid \\
 & & & \texttt{pack}\, \tau, e\, \texttt{as}\, \exists \alpha.\, \tau' \mid \texttt{unpack}\, e_1\, \texttt{as}\, \alpha, x\, \texttt{in}\, e_2 \mid \\
 & & & \texttt{fold}_\tau\, e \mid \texttt{unfold}\, e \\
\textit{Values} & v & ::= & x \mid () \mid \pm n \mid \texttt{true} \mid \texttt{false} \mid \langle v_1, v_2 \rangle \mid \\
 & & & \texttt{inl}_\tau\, v \mid \texttt{inr}_\tau\, v \mid \lambda x : \tau.\, e \mid \Lambda \alpha.\, e \mid \\
 & & & \texttt{pack}\, \tau_1, v\, \texttt{as}\, \exists \alpha.\, \tau \mid \texttt{fold}_\tau\, v
\end{array}
$$

Outline

1. The Language F^{μ}
2. The Logic LSLR
3. Encoding a Logical Relation for F^{μ} in LSLR
4. Derivable Rules

The Logic LSLR (Basic Idea)

Start with Plotkin and Abadi's "logic for parametric polymorphism" (TLCA'93)

- Adapt it to reason operationally about CBV small-step semantics

Extend it with recursively defined relations

- Enables straightforward logical relation for recursive types
- To make sense of circularity, introduce "later" operator $\triangleright A$ from Appel, Melliès, Richards, and Vouillon's "very modal model" paper (POPL'07), which in turn was adapted from Gödel-Löb logic of provability

The Logic LSLR (Syntax)

$$
\begin{array}{llcl}
\textit{Rel. Var's} & r, s & \in & \textit{RelVar} \\
\mathsf{F}^{\mu}\ \textit{Ctxt's} & \Gamma & ::= & \cdot \mid \Gamma, \alpha \mid \Gamma, x : \tau \mid \Gamma, t : \tau \\
\textit{Rel. Ctxt's} & \Delta & ::= & \cdot \mid \Delta, r : \mathrm{VRel}(\tau_1, \tau_2) \mid r : \mathrm{TRel}(\tau_1, \tau_2) \\
\textit{Log. Ctxt's} & \Theta & ::= & \cdot \mid \Theta, A \\
\textit{Atomic Prop's} & P & ::= & e_1 = e_2 \mid e_1 \overset{*}{\mapsto} e_2 \mid e_1 \overset{0}{\mapsto} e_2 \mid e_1 \overset{1}{\mapsto} e_2 \\
\textit{Propositions} & A, B & ::= & P \mid \top \mid \bot \mid A \wedge B \mid A \vee B \mid \\
 & & & A \supset B \mid \forall \Gamma.A \mid \exists \Gamma.A \mid \\
 & & & \forall \Delta.A \mid \exists \Delta.A \mid (e_1, e_2) \in R \mid \triangleright A \\
\textit{Relations} & R, S & ::= & r \mid (x_1 : \tau_1, x_2 : \tau_2).A \mid \\
 & & & (t_1 : \tau_1, t_2 : \tau_2).A \mid \mu r.R
\end{array}
$$

LSLR Main Judgment

$$\Gamma; \Delta; \Theta \vdash A$$

Relational Axioms

$$(v_1, v_2) \in (x_1 : \tau_1, x_2 : \tau_2).A \equiv A[v_1/x_1, v_2/x_2]$$

$$(e_1, e_2) \in (t_1 : \tau_1, t_2 : \tau_2).A \equiv A[e_1/t_1, e_2/t_2]$$

$$(e_1, e_2) \in \mu r.R \equiv (e_1, e_2) \in R[\mu r.R/r]$$

Monotonicity

$$A \supset \triangleright A$$

Löb Rule

$$(\triangleright A \supset A) \supset A$$

Distributivity Laws

$$
\begin{aligned}
\rhd(A \land B) &\equiv \rhd A \land \rhd B \\
\rhd(A \lor B) &\equiv \rhd A \lor \rhd B \\
\rhd(A \supset B) &\equiv \rhd A \supset \rhd B \\
\rhd\forall\Gamma.A &\equiv \forall\Gamma.\rhd A \\
\rhd\forall\Delta.A &\equiv \forall\Delta.\rhd A \\
\rhd\exists\Gamma.A &\equiv \exists\Gamma.\rhd A \\
\rhd\exists\Delta.A &\equiv \exists\Delta.\rhd A
\end{aligned}
$$

Outline

1. The Language F^{μ}
2. The Logic LSLR
3. Encoding a Logical Relation for F^{μ} in LSLR
4. Derivable Rules

Logical Relation for Values

$$\mathcal{V}\,[\![\alpha]\!]\,\rho \stackrel{\text{def}}{=} R, \quad \text{where } \rho(\alpha) = (\tau_1, \tau_2, R)$$

$$\mathcal{V}\,[\![\tau_b]\!]\,\rho \stackrel{\text{def}}{=} (x_1 : \tau_b, x_2 : \tau_b).\; x_1 = x_2, \quad \text{where } \tau_b \in \{\mathsf{unit}, \mathsf{int}, \mathsf{bool}\}$$

$$\begin{aligned}\mathcal{V}\,[\![\tau' \times \tau'']\!]\,\rho \stackrel{\text{def}}{=}\; & (x_1 : \rho_1(\tau' \times \tau''), x_2 : \rho_2(\tau' \times \tau'')).\\ & \quad \exists x_1', x_1'', x_2', x_2''.\; x_1 = \langle x_1', x_1'' \rangle \wedge x_2 = \langle x_2', x_2'' \rangle \wedge \\ & \quad\; (x_1', x_2') \in \mathcal{V}\,[\![\tau']\!]\,\rho \;\wedge\; (x_1'', x_2'') \in \mathcal{V}\,[\![\tau'']\!]\,\rho\end{aligned}$$

$$\begin{aligned}\mathcal{V}\,[\![\tau' + \tau'']\!]\,\rho \stackrel{\text{def}}{=}\; & (x_1 : \rho_1(\tau' + \tau''), x_2 : \rho_2(\tau' + \tau'')).\\ & \quad (\exists x_1', x_2'.\; x_1 = \mathtt{inl}\, x_1' \wedge x_2 = \mathtt{inl}\, x_2' \wedge (x_1', x_2') \in \mathcal{V}\,[\![\tau']\!]\,\rho)\\ & \vee\; (\exists x_1'', x_2''.\; x_1 = \mathtt{inr}\, x_1'' \wedge x_2 = \mathtt{inr}\, x_2'' \wedge (x_1'', x_2'') \in \mathcal{V}\,[\![\tau'']\!]\,\rho))\end{aligned}$$

$$\begin{aligned}\mathcal{V}\,[\![\tau' \to \tau'']\!]\,\rho \stackrel{\text{def}}{=}\; & (x_1 : \rho_1(\tau' \to \tau''), x_2 : \rho_2(\tau' \to \tau'')).\\ & \quad \forall y_1, y_2.\; (y_1, y_2) \in \mathcal{V}\,[\![\tau']\!]\,\rho \supset (x_1 y_1, x_2 y_2) \in \mathcal{E}\,[\![\tau'']\!]\,\rho\end{aligned}$$

Logical Relation for Values (of Quantified Types)

$$
\begin{aligned}
\mathcal{V}\llbracket \forall\alpha.\,\tau \rrbracket\,\rho \stackrel{\text{def}}{=} \; & (x_1 : \rho_1(\forall\alpha.\,\tau), x_2 : \rho_2(\forall\alpha.\,\tau)).\\
& \quad \forall\alpha_1, \alpha_2.\ \forall r : \mathrm{VRel}(\alpha_1, \alpha_2).\\
& \qquad (x_1\,[\alpha_1], x_2\,[\alpha_2]) \in \mathcal{E}\llbracket \tau \rrbracket\,\rho, \alpha \mapsto (\alpha_1, \alpha_2, r)
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{V}\llbracket \exists\alpha.\,\tau \rrbracket\,\rho \stackrel{\text{def}}{=} \; & (x_1 : \rho_1(\exists\alpha.\,\tau), x_2 : \rho_2(\exists\alpha.\,\tau)).\\
& \quad \exists\alpha_1, \alpha_2, y_1, y_2.\ \exists r : \mathrm{VRel}(\alpha_1, \alpha_2).\\
& \qquad x_1 = \mathtt{pack}\,\alpha_1, y_1\,\mathtt{as}\,\cdots \;\wedge\; x_2 = \mathtt{pack}\,\alpha_2, y_2\,\mathtt{as}\,\cdots \;\wedge\\
& \qquad (y_1, y_2) \in \mathcal{V}\llbracket \tau \rrbracket\,\rho, \alpha \mapsto (\alpha_1, \alpha_2, r)
\end{aligned}
$$

Logical Relation for Values (of Recursive Type)

$$\begin{aligned}\mathcal{V}\,[\![\mu\alpha.\,\tau]\!]\,\rho \stackrel{\text{def}}{=}\; & \mu r.(x_1 : \rho_1(\mu\alpha.\,\tau), x_2 : \rho_2(\mu\alpha.\,\tau)).\\ & \quad \exists y_1, y_2.\; x_1 = \texttt{fold}\, y_1 \;\wedge\; x_2 = \texttt{fold}\, y_2 \wedge \\ & \qquad \triangleright(y_1, y_2) \in \mathcal{V}\,[\![\tau]\!]\,\rho, \alpha \mapsto (\rho_1(\mu\alpha.\,\tau), \rho_2(\mu\alpha.\,\tau), r)\end{aligned}$$

Outline

1. The Language F^{μ}
2. The Logic LSLR
3. Encoding a Logical Relation for F^{μ} in LSLR
4. Derivable Rules

Coincidence of Value and Term Relations

$$\frac{\Gamma; \Delta; \Theta \vdash (v_1, v_2) \in \mathcal{V}\,[\![\tau]\!]\,\rho}{\Gamma; \Delta; \Theta \vdash (v_1, v_2) \in \mathcal{E}\,[\![\tau]\!]\,\rho}$$

Extensionality

$$\frac{\Gamma, x_1, x_2; \Delta; \Theta, (x_1, x_2) \in \mathcal{V}\left[\!\left[\tau'\right]\!\right] \rho \vdash (v_1 x_1, v_2 x_2) \in \mathcal{E}\left[\!\left[\tau''\right]\!\right] \rho}{\Gamma; \Delta; \Theta \vdash (v_1, v_2) \in \mathcal{V}\left[\!\left[\tau' \rightarrow \tau''\right]\!\right] \rho}$$

Evaluation Rules

$$\frac{\Gamma;\Delta;\Theta \vdash e_1 \stackrel{*}{\longmapsto} e_1' \quad \Gamma;\Delta;\Theta \vdash e_2 \stackrel{*}{\longmapsto} e_2' \qquad \Gamma;\Delta;\Theta \vdash (e_1', e_2') \in \mathcal{E}\,[\![\tau]\!]\,\rho}{\Gamma;\Delta;\Theta \vdash (e_1, e_2) \in \mathcal{E}\,[\![\tau]\!]\,\rho}$$

$$\frac{\Gamma;\Delta;\Theta \vdash (e_1, e_2) \in \mathcal{E}\,[\![\tau]\!]\,\rho \qquad \Gamma, x_1, x_2;\Delta;\Theta, e_1 \stackrel{*}{\longmapsto} x_1, e_2 \stackrel{*}{\longmapsto} x_2, (x_1, x_2) \in \mathcal{V}\,[\![\tau]\!]\,\rho \vdash (E_1[x_1], E_2[x_2]) \in \mathcal{E}\,[\![\tau']\!]\,\rho'}{\Gamma;\Delta;\Theta \vdash (E_1[e_1], E_2[e_2]) \in \mathcal{E}\,[\![\tau']\!]\,\rho'}$$

Useful Rules Concerning the $\triangleright$ Modality

$$\frac{\Gamma; \Delta; \Theta_1, \Theta_2 \vdash B}{\Gamma; \Delta; \Theta_1, \triangleright\Theta_2 \vdash \triangleright B}$$

$$\frac{\Gamma; \Delta; \Theta \vdash e_1 \overset{1}{\mapsto} e_1' \quad \Gamma; \Delta; \Theta \vdash e_2 \overset{1}{\mapsto} e_2' \qquad \Gamma; \Delta; \Theta \vdash \triangleright(e_1', e_2') \in \mathcal{E}\left[\!\left[\tau\right]\!\right]\rho}{\Gamma; \Delta; \Theta \vdash (e_1, e_2) \in \mathcal{E}\left[\!\left[\tau\right]\!\right]\rho}$$

$$\frac{\Gamma; \Delta; \Theta, \triangleright A \vdash A}{\Gamma; \Delta; \Theta \vdash A}$$

Useful Rules Concerning the $\triangleright$ Modality

$$\frac{\Gamma; \Delta; \Theta_1, \Theta_2 \vdash B}{\Gamma; \Delta; \Theta_1, \triangleright\Theta_2 \vdash \triangleright B}$$

$$\frac{\Gamma; \Delta; \Theta \vdash e_1 \overset{1}{\mapsto} e_1' \quad \Gamma; \Delta; \Theta \vdash e_2 \overset{1}{\mapsto} e_2' \quad \Gamma; \Delta; \Theta \vdash \triangleright(e_1', e_2') \in \mathcal{E}\,[\![\tau]\!]\,\rho}{\Gamma; \Delta; \Theta \vdash (e_1, e_2) \in \mathcal{E}\,[\![\tau]\!]\,\rho}$$

$$\frac{\Gamma; \Delta; \Theta, \triangleright A \vdash A}{\Gamma; \Delta; \Theta \vdash A}$$

Fixed-Point Induction

$$\frac{\begin{array}{c} F_i = \mathtt{fun}\, f(x_i)\ \mathtt{is}\ e_i \\ \Gamma, x_1, x_2; \Delta; \Theta, (F_1, F_2) \in \mathcal{V}\,[\![\tau' \to \tau'']\!]\,\rho, (x_1, x_2) \in \mathcal{V}\,[\![\tau']\!]\,\rho \\ \vdash (e_1[F_1/f], e_2[F_2/f]) \in \mathcal{E}\,[\![\tau'']\!]\,\rho \end{array}}{\Gamma; \Delta; \Theta \vdash (F_1, F_2) \in \mathcal{V}\,[\![\tau' \to \tau'']\!]\,\rho}$$

What Else Is In the Paper

- Encoding of $\mathcal{E}\,[\![\tau]\!]\,\rho$ in the logic
- More derivable rules (both equational and inequational)
- Model of the logic
- Proof of soundness of LR w.r.t. contextual equivalence
- Example proofs of contextual equivalences
- Comparison with related work

Future Work

- Generalize our approach to handle (higher-order) state
  - We've already done this (paper under submission)
- Explore connection to bisimulation-based methods (Sumii, Pierce, Sangiorgi, *et al.*)
- Mechanize our metatheory!

Thank You!