A Relational Modal Logic for Higher-Order Stateful ADTs

Derek Dreyer
Georg Neis
Andreas Rossberg
Lars Birkedal

MPI-SWS, Germany
IT University of Copenhagen, Denmark

POPL 2010
Madrid

Program Equivalence

Program verification

- Show program P is observationally equivalent to some reference implementation

Compiler correctness

- Show input and output of compiler phases are semantically equivalent

Representation independence and data abstraction

- Modules M_1 and M_2 can employ different data representations and local invariants, yet be observationally equivalent

Observational (aka Contextual) Equivalence

Canonical notion of program equivalence:

- $M_1 \equiv M_2$ if no program context can distinguish them
- Difficult to reason about directly,
  due to the universal quantification over contexts

Several decades of work on various methods for
local reasoning about observational equivalence:

- Logical relations, bisimulations, Hoare-style logics, ...
- Mostly for restricted languages (purely functional, Algol-like, etc.)

... in ML-Like Languages

Algebraic data types, recursive types $(\tau_1 \times \tau_2, \tau_1 + \tau_2, \mu\alpha.\tau)$

Higher-order functions $(\tau_1 \to \tau_2)$

Polymorphism, generics $(\forall\alpha.\tau)$

Modules, ADTs $(\exists\alpha.\tau)$

Mutable references of unrestricted type $(\textsf{ref}\,\tau)$

Symbol Generator Example

$$\tau = \exists\alpha.\,(\mathsf{unit} \to \alpha) \times (\alpha \times \alpha \to \mathsf{bool})$$

$$e_1 = \mathsf{pack}\,\mathsf{ref}\,\mathsf{unit}, \langle \lambda\_.\mathsf{ref}\,\langle\rangle, \lambda y.\mathsf{fst}\,y == \mathsf{snd}\,y\rangle \text{ as } \tau$$

$$e_2 = \mathsf{let}\ x = \mathsf{ref}\ 0\ \mathsf{in}\ \mathsf{pack}\,\mathsf{int}, \langle \lambda\_.{+}{+}x, \lambda y.\mathsf{fst}\,y = \mathsf{snd}\,y\rangle \text{ as } \tau$$

Our Contribution

We give the first logic for reasoning about observational equivalence in ML-like languages

Our logic synthesizes several ideas from prior work:

- Plotkin-Abadi logic for relational parametricity
- Gödel-Löb logic (after Appel *et al.*'s "very modal model")
- S4 modal logic
- Relational separation logic (Yang, Benton)

Our Contribution

We give the first logic for reasoning about observational equivalence in ML-like languages

Our logic synthesizes several ideas from prior work:

- Plotkin-Abadi logic for relational parametricity
- Gödel-Löb logic (after Appel *et al.*'s "very modal model")
- S4 modal logic
- Relational separation logic (Yang, Benton)
- Our own step-indexed Kripke logical relations model (Ahmed-Dreyer-Rossberg, POPL'09)

Kripke Logical Relations for State

Kripke logical relations models for reasoning about state:

- Term relation indexed by "possible world" W
- W characterizes invariants about contents of heap, *e.g.*, $x \hookrightarrow n$ in program 1 and $x \hookrightarrow -n$ in program 2

The trouble with higher-order state (general references):

- W may depend on "logical relatedness" of heap contents
- Leads to circularity in the construction of possible worlds

Step-Indexed Kripke Logical Relations for Higher-Order State

Step-indexed logical relations (Appel-McAllester, Ahmed):

- Stratify construction of possible worlds by "step index" n
- Intuition: n-level worlds only care whether heap contents are logically related for $n - 1$ steps

A key contribution of our POPL'09 paper:

- A step-indexed relational model for higher-order state (as opposed to the unary models of previous work)

The Trouble With Step-Indexed Models

Step-indexed models are great . . .

- Easy to construct, simple intuition
- Applicable to a variety of "semantically difficult" features

The Trouble With Step-Indexed Models

Step-indexed models are great . . .

- Easy to construct, simple intuition
- Applicable to a variety of “semantically difficult” features

. . . except for the steps!

- To prove M_1 and M_2 equivalent, you pick an arbitrary n and prove they are related for n steps.
- Step-index arithmetic pervaded our POPL’09 proofs.

The Trouble With Step-Indexed Models

Step-indexed models are great . . .

- Easy to construct, simple intuition
- Applicable to a variety of "semantically difficult" features

. . . except for the steps!

- To prove M_1 and M_2 equivalent, you pick an arbitrary n and prove they are related for n steps.
- Step-index arithmetic pervaded our POPL'09 proofs.

Important to develop step-free proof principles

An Obvious Idea That Doesn't Work

Define M_1 and M_2 to be infinitely related if they are related for any # of steps.

An Obvious Idea That Doesn't Work

Define M_1 and M_2 to be infinitely related if they are related for any # of steps.

Prove infinite relatedness enjoys an extensionality principle:

- f_1 and f_2 are infinitely related *iff* they map infinitely-related arguments to infinitely-related results.

An Obvious Idea That Doesn't Work

Define M_1 and M_2 to be infinitely related if they are related for any # of steps.

Prove infinite relatedness enjoys an extensionality principle:

- f_1 and f_2 are infinitely related *iff* they map infinitely-related arguments to infinitely-related results.

Unfortunately, it is false! In fact:

- f_1 and f_2 are infinitely related *iff*, for any n, they map n-related arguments to n-related results.

Why Our Logic is Useful

It abstracts away the boring stuff

Messy details of the step-indexed construction are confined to the model

Example: The Extensionality Principle

$$\frac{\mathcal{C}, x_1, x_2, x_1 \equiv x_2 : \sigma \vdash f_1\, x_1 \equiv f_2\, x_2 : \tau}{\mathcal{C} \vdash f_1 \equiv f_2 : \sigma \rightarrow \tau}$$

Quantification over step indices and possible worlds is confined to the model:

$$[\![\mathcal{C} \vdash P]\!] \approx \forall n.\, \forall W \in World_n.\ [\![\mathcal{C}]\!]\, nW \Rightarrow [\![P]\!]\, nW$$

Conclusion

First logic for reasoning about observational equivalence in ML-like languages

Similar reasoning ability to our POPL'09 model, but at a much higher level of abstraction