HOAS on top of FOAS

Andrei Popescu

Joint work with Elsa Gunter and Chris Osborn

University of Illinois at Urbana-Champaign

Motto (and excuse)

"When you try to convey an idea, do not aim at being complete. Rather, select from that idea scattered things you like most."

~ Jorge Luis Borges

Overview

- Motivation: why (still) study syntax with bindings?
- HOAS recalled
- HOAS on top of FOAS
- Case study: a formal proof of strong normalization for System F in Isabelle/HOL

Overview

- Motivation: why (still) study syntax with bindings?

- HOAS recalled

- HOAS on top of FOAS

Omitted from the presentation:

- Case study: a formal proof of strong normalization for System F in Isabelle/HOL

Overview

- Motivation: why (still) study syntax with bindings?
- HOAS recalled
- HOAS on top of FOAS

Terms and alpha-equivalence

- Raw terms of λ-calculus:

  X ::= Var x | App X Y | Lam x X

- Let $\equiv$ be the alpha- (naming-) equivalence relation on raw terms

Interpretation in semantic domains

- APP : D → D → D
- LAM : (D → D) → D
- env = (var → D)
- [[_]] _ : Term → Env → D, defined recursively on the first argument, by:
  - [[x]] ρ = ρ x
  - [[App X Y]] ρ = APP ([[X]] ρ) ([[Y]] ρ)
  - [[Lam x X]] ρ = LAM (λ d. X [[ρ (x := d)]])

Exercise

- It is "intuitively obvious" that:
  - Interpretation respects alpha:

    $$\forall\ X\ X'.\ X \equiv X' \text{ implies } [[\ X\ ]] = [[\ X'\ ]]$$

  - The following "substitution lemma" holds:

    $$[[\ X\ [Y\ /\ y]\ ]]\ \rho = [[\ X\ ]]\ (\rho\ (y := ([[\ Y\ ]]\ \rho)))$$

Exercise

- It is "intuitively obvious" that:
  - Interpretation respects alpha:

    $$\forall\, X\, X'.\ X \equiv X' \ \text{ implies } \ [[\, X\,]] = [[\, X'\,]]$$

  - The following "substitution lemma" holds:

    $$[[\, X\, [Y\, /\, y]\,]]\, \rho = [[\, X\,]]\, (\rho\, (y := ([[\, Y\,]]\, \rho)))$$

- Nobody wants to prove these ☺

Exercise

- It is “intuitively obvious” that:
  - Interpretation respects alpha:

    $\forall$ X X’. X $\equiv$ X’ implies [[X]] = [[X’]]
  - The following “substitution lemma” holds:

    [[X [Y / y]]] ρ = [[X]] (ρ (y := ([[Y]] ρ)))

- Nobody wants to prove these

- But some have to ☹ (those who formalize)

Exercise

Please send me solution to uuomul@yahoo.com

- May use any (correct) definition of alpha-equivalence
- Or may assume alpha-equivalence (and also swapping, substitution, free variables, etc.) already defined
- May assume any basic property of these (e.g., anything in the equational theory of alpha)
- May consult any textbook or research paper

A. M. Pitts: Alpha-structural recursion and induction, J. ACM, 2006.

Overview

- Motivation: why (still) study syntax with bindings?
- HOAS recalled
- HOAS on top of FOAS

Higher-Order Abstract Syntax

- Represent object systems (e.g., logics, operational semantics of PL, etc.) in a fixed logical framework
- Object-level binding and inference mechanisms are captured by corresponding ones in the logical framework

Higher-Order Abstract Syntax

- Represent object system (e.g., logic, operational semantics of PL, etc.) in a fixed logical framework
- Object-level binding and inference mechanisms are captured by corresponding ones in the logical framework
- Why?

Higher-Order Abstract Syntax

- Represent object system (e.g., logic, operational semantics of PL, etc.) in a fixed logical framework
- Object-level binding and inference mechanisms are captured by corresponding ones in the logical framework
- Why?
- Formalize/implement tedious "details" once and for all, when defining the logical framework

HOAS and meta-reasoning

- Originally: for reasoning in the object systems

Edinburgh LF, Generic Isabelle

- Later: meta-theory of the object systems too (i.e., reason about the object system)

TWELF, Abella, Hybrid, Delphin, ATS, Beluga

- Subtle problems and challenges arise when combining HOAS with meta-reasoning

Running example: Syntax

First-order syntax (up to α):

- Curry-style: no type annotations
- Data variables x, y, z, data terms X,Y, Z, data abstractions A, B

X ::= Var x | App X Y | Lam A   A ::= x . X

- Type variables tx, ty, tz, type terms tX, tY, tZ, type abstractions tA, tB

tX ::= Tvar tx | Arr tX tY

Running example: β-reduction for untyped λ-calculus

```
App (Lam (x . Y)) X  ~~>  Y [X / x]    (Beta)

              Y ~~> Y'
----------------------------------------(Xi)
  Lam (x . Y)  ~~>  Lam (x . Y')

              X ~~> X'
--------------------------------(App-Left)
  App X Y  ~~> App X' Y
```

Running example: Curry-style simple typing

$$\frac{.}{\Gamma, x : tX \mid\!- x : tX} \; [x \text{ fresh } \Gamma] \quad (\text{Asm})$$

$$\frac{\Gamma \mid\!- Y : tY}{\Gamma, x : tX \mid\!- Y : tY} \; [x \text{ fresh } \Gamma] \quad (\text{Weak})$$

$$\frac{\Gamma, x : tX \mid\!- Y : tY}{\Gamma \mid\!- \text{Lam } (x \,.\, Y) : \text{Arr } tX \; tY} \; [x \text{ fresh } \Gamma] \quad (\text{Arr-I})$$

$$\frac{\Gamma \mid\!- Z : \text{Arr } tX \; tY \qquad \Gamma \mid\!- X : tX}{\Gamma \mid\!- \text{App } Z \; X : tY} \; (\text{Arr-E})$$

HOAS representation

- In pure intuitionistic HOL (similarly, in LF)
- Declare
  - An HOL type: tm
  - Constants app : tm → tm → tm

    lam : (tm → tm) → tm

    beta : tm → tm → bool
- State axioms, e.g.:

  beta (app (lam (λ x : tm. Y x)) X) (Y X)

HOAS idea rephrased

For an “observer” from inside the logical framework:

- Object bindings are taken ad literam!
- E.g., the term Lam x . (Var x) is not ``syntax”, but is actually the function $\lambda X.\ X$

HOAS idea rephrased

For an "observer" from inside the logical framework:

- Object bindings are taken ad literam!
- E.g., the term Lam x . (Var x) is not "syntax", but is actually the function $\lambda X.\ X$
- Well, almost: it is really lam ($\lambda X.\ X$)

(recall lam : (tm → tm) → tm)

Overview

- Motivation: why (still) study syntax with bindings?
- HOAS recalled
- HOAS on top of FOAS

HOAS on top of FOAS

- Stronger (meta-)logical-framework: strong enough to develop general mathematics (e.g., the logic of Isabelle/HOL)
- Terms are still "syntax" (defined in the standard way)
- HOAS comes not as a "representation", but as a higher-order view of the same syntax
- Thus, e.g., Lam x x is both ``itself" (as a finite piece of syntax) and lam (λX. X)

HOAS view of syntax: Abstractions as functions

- FOAS definition/construction: A = (x . X)
- HOAS treatment: A _ Y = "A applied Y", defined to be X [Y / x]
- May regard abstractions as forming a subspace of **tm → tm**
- This view accommodates:
  - HOAS structural recursion principles (omitted from this presentation)
  - a certain way to represent inference relations

HOAS representation of β-reduction

App (Lam (x . Y)) X ~~> Y [X / x] (Beta-FOAS)

App (Lam A) X ~~> A _ X (Beta-HOAS)

Y ~~> Y'
-------------------------------------(Xi-FOAS)
Lam (x . Y) ~~> Lam (x . Y')

∀ X. A _ X ~~> A' _ X
------------------------------(Xi-HOAS)
Lam A ~~> Lam A'

HOAS representation of typing

$\forall\,\Gamma$ - (typing) context, i.e., list of pairs (data variable, type term):

$$x_1 : tX_1, \ldots, x_n : tX_n$$

$\forall\,\Delta$ - HOAS context, i.e., list of pairs (data term, type term):

$$X_1 : tX_1, \ldots, X_n : tX_n$$

- Note: we close under substitution

HOAS representation of typing

$$\frac{\Gamma, x : tX \ \vdash \ Y : tZ}{\Gamma \vdash \mathrm{Lam}\ (x\ .\ Y) : \mathrm{Arr}\ tX\ tZ}\ [x \text{ fresh for } \Gamma] \quad \text{(Arr-I-FOAS)}$$

$$\frac{\forall X.\ \Delta, X : tX \ \Vdash \ A \mathbin{\_} X : tZ}{\Delta \Vdash \mathrm{Lam}\ A : \mathrm{Arr}\ tX\ tZ} \quad \text{(Arr-I-HOAS)}$$

How HOAS is this?

- No more freshness side conditions √
- Object-level bindings pushed to the meta level √
- Meta-reasoning capabilities kept intact √
- Also push inference contexts to the meta level?

Parenthesis: pure HOAS representation

- In intuitionistic HOL:
- Declare tpOf : tm $\to$ tp $\to$ bool
- State axioms, such as:

$$\frac{\forall\, X.\ \text{tpOf } X \text{ tX} \;\Rightarrow\; \text{tpOf } (A\ X) \text{ tY}}{\text{tpOf (Lam A) (Arr tX tY)}}$$

to capture

$$\frac{\Gamma,\ x : \text{tX} \ \ |\text{-}\ \ Y : \text{tZ}}{\Gamma\ |\text{-}\ \text{Lam } (x\ .\ Y) : \text{Arr tX tZ}}\quad \begin{matrix}[x \text{ fresh } \Gamma] \\ (\text{Arr-I})\end{matrix}$$

"Context-free" induction principle for typing

If $H : tm \rightarrow tp \rightarrow bool$ s.t.:

$$\frac{\forall X.\ H\ X\ tX \;\Rightarrow\; H\ (A\ \_\ X)\ tZ}{H\ (Lam\ A)\ (Arr\ tX\ tZ)}\ (\text{ArrI-H})$$

etc., then $\forall X\ \ tX.\ [\,]\ ||\text{-}\ X : tX \;\Rightarrow\; H\ X\ tX$

(Higher degree of HOAS – not only bindings and substitution, but also inference contexts are pushed to the meta-level)

Conclusions

- Worth still studying syntax with bindings
- HOAS:
  - Exterior view: capture object-level bindings by bindings in the logical framework
  - Inner view: syntactic bindings become true semantic bindings
- HOAS technique available atop of FOAS

HOAS on top of FOAS

- FOAS operators still available if needed
- Purely definitional development of HOAS
- General-purpose logical framework (standard mathematics)
- Adequacy statable and provable in the logical framework itself

Credits and very related work

- HOAS on top of FOAS ideas previously employed in the Hybrid logical framework

(work by A. Momigliano, A. Felty, S. Ambler, R. L. Crole, and others)

- A quasi-HOAS proof of strong normalization for System F previously given in the ATS logical framework

(work by C. Chen, H. Xi, K. Donnelly and others)

Thank you