$FO\lambda^{\nabla}$ in Higher Order Abstract Syntax

Marino Miculan Ivan Scagnetto

Department of Mathematics and Computing Science
University of Udine

TYPES 2004, Jouy-en-Josas, December 16, 2004

Reasoning with abstractions

In recent years, much work on systems for reasoning about object systems featuring *abstractions*.
Two approaches (among several others):

- HOAS(+ToC): weak higher-order abstract syntax in usual TT (e.g., CIC) possibly "patched" with specific properties for reasoning about second-order terms (the Theory of Contexts)
- $FO\lambda^{\nabla}$ [Miller&Tiu]: FO logic specifically designed for dealing with "schematic" properties

A given object system with abstractions (e.g., λ-calculus, π-calculus) can be easily encoded in any of them.

Question

Which is the relation between $FO\lambda^{\nabla}$ and HOAS-based encodings?

Weak HOAS encodings: the untyped λ-calculus

Canonical example: take $M, N ::= x \mid MN \mid \lambda x.M$.
Its "weak" HOAS encoding in Coq is

```
Parameter Var : Set.
Inductive Tm : Set :=
  var : Var -> Tm
| app : Tm -> Tm -> Tm
| lam : (Var -> Tm) -> Tm.
```

Example:
$\lambda x.xy \quad \mapsto \quad$ `(lam (fun x:Var => (app (var x) (var y))))`

Advantage

α-conversion *and* induction/recursion over terms are automatically provided by the metalanguage.

Weak HOAS encodings: capture-avoiding substitution

- Capture avoiding substitution $M[N/x]$ is not given for free
- It can be represented by means of a ternary predicate `subst`:

  `Inductive subst (M:Tm) : (Var->Tm) -> Tm -> Prop :=...`

  Intended meaning: `(subst M N O)` holds iff `O` is the result of "filling" the hole of `N` with `M`.
- Some crucial properties of substitution are needed, such as:

```
forall x:Var, forall M:Var->Tm, forall N:Tm,
      (subst (var x) M N) -> N=(M x).

forall M:Tm,forall N:Var->Tm, exists O:Tm,(subst M N O)
```

- These results cannot be proved in pure CIC, but they can be proved assuming the Theory of Contexts

The Theory of Contexts (ToC)

The Theory of Contexts is a set of axioms formalizing some simple properties about variables and contexts (i.e., terms with *holes*):

Decidability of equality over variables

$\forall x, y. x = y \vee x \neq y$

Existence of fresh variables

$\forall M. \exists x. x \notin FV(M)$

β-expansion

$\forall M. \forall x. \exists M'[\cdot]. x \notin FV(M'[\cdot]) \wedge M = M'[x]$

Restricted extensionality

$\forall M[\cdot], N[\cdot]. \forall x. x \notin FV(M[\cdot], N[\cdot]) \Rightarrow M[x] = N[x] \Rightarrow M[\cdot] = N[\cdot]$

Syntax of $FO\lambda^\nabla$ [Miller&Tiu, LICS 2003]

- Types: usual simple types: $\tau ::= o \mid \gamma \mid \tau_1 \to \tau_2$
- Terms: usual simply typed λ-calculus: $\Sigma \vdash t : \tau$
- Object-level datatypes can be represented by adding types and constructors (even higher-order)
- (Basic) Formulas: terms of type o:

$$A, B ::= \top \mid \bot \mid A \supset B \mid \cdots \mid \forall_\gamma x.A \mid \exists_\gamma x.A \mid \nabla_\gamma x.A$$

- Generic Judgments:

$$\mathcal{A}, \mathcal{B} ::= \overbrace{(x_1 : \tau_1, \ldots, x_n : \tau_n)}^{\sigma} \rhd B$$

Think of $x_1, \ldots, x_n$ as *locally scoped constants*, i.e., *eigenvariables*.

Proof system

Original proof system is in Gentzen-style, for deriving sequents

$$\Sigma : \mathcal{B}_1, \ldots \mathcal{B}_n \vdash \mathcal{B}$$

Here we consider an equivalent variant:

Proof system in Natural Deduction—some simple rules

$$\frac{\sigma \rhd A \quad \sigma \rhd B}{\sigma \rhd A \wedge B}\ \wedge\mathcal{I} \qquad \frac{\sigma \rhd A \wedge B}{\sigma \rhd A}\ \wedge\mathcal{E}l \qquad \frac{\sigma \rhd A \wedge B}{\sigma \rhd B}\ \wedge\mathcal{E}r$$

$$\frac{\begin{array}{c}(\sigma \rhd A)\\ \vdots \\ \sigma \rhd B\end{array}}{\sigma \rhd A \supset B}\ \supset\mathcal{I} \qquad \frac{\sigma \rhd A \supset B \quad \sigma \rhd A}{\sigma \rhd B}\ \supset\mathcal{E}$$

$$\frac{\sigma, x{:}\gamma \rhd B}{\sigma \rhd \nabla_\gamma x.B}\ \nabla\mathcal{I} \qquad \frac{\sigma \rhd \nabla_\gamma x.B}{\sigma, x{:}\gamma \rhd B}\ \nabla\mathcal{E} \qquad \frac{}{\sigma \rhd \top}\ \top\mathcal{I} \qquad \frac{\sigma \rhd \bot}{\mathcal{B}}\ \bot\mathcal{E}$$

Proof system

Rules using *raising*

$$\frac{\begin{array}{c}(h : |\sigma| \to \gamma)\\ \vdots \\ \sigma \rhd B[(h\ \sigma)/x]\end{array}}{\sigma \rhd \forall_\gamma x.B}\ \forall\mathcal{I} \qquad \frac{\begin{array}{c}(\sigma)\\ \vdots \\ t : \gamma\end{array} \quad \sigma \rhd \forall_\gamma x.B}{\sigma \rhd B[t/x]}\ \forall\mathcal{E}$$

Notice: when $\sigma = ()$, we get back the usual intuitionistic FOL.

$FO\lambda^{\nabla}$ vis-a-vis weak HOAS

	$FO\lambda^{\nabla}$	weak HOAS
schematic terms	$\lambda x : \tau.t$	$\lambda x : Var_\tau.t$
bound variables in terms	metavariables of type τ	metavariables of type Var_τ
generic quantifier	$\nabla x.B$	—
generic judgments	$(x_1, \ldots, x_n) \rhd B$	$\lambda x_1{:}Var \ldots \lambda x_n{:}Var.B$
eigenvariables in judgments	locally scoped constants	locally scoped *distinct metavariables*

Translation idea

We can map $FO\lambda^{\nabla}$'s generic judgments into CIC propositions, by formalizing this correspondence, and by internalizing the notion of "locally scoped distinct metavariable".

Encoding $FO\lambda^{\nabla}$ syntax in weak HOAS

(Harmless simplification: just 1 type of terms)

Terms and Formulas

```
Parameter Var:Set.
Inductive Term:Set :=...

Inductive Form : Set :=
        tt : Form
      | And : Form -> Form -> Form
      | Imp : Form -> Form -> Form
      | Forall : (Var->Form) -> Form
      | Nabla : (Var->Form) -> Form
      | ...
```


Encoding $FO\lambda^{\nabla}$ syntax in weak HOAS

Generic Judgments

```
Fixpoint GenJ (n:nat) {struct n} : Set :=
  match n with
  | O => Form
  | (S m) => Var -> (GenJ m)
  end.
```

Thus, a generic judgment is a "formula with holes":

$(x_1, \ldots, x_n) \rhd B \quad \mapsto$

```
fun x1 ... xn :Var => B
: (GenJ n)
  = Var ->...-> Var -> Form
```


Translating $FO\lambda^{\nabla}$ formulae to CIC using HOAS

By means of an inductive predicate, capturing the "intuitive" meaning of formulas:

```
Inductive T_F : Form -> Prop :=  ...
| T_F_And:forall A B:Form,(T_F A)->(T_F B)->(T_F (And A B))
| T_F_Forall : forall A:Var->Form,
    (forall t:Term,forall B:Form,(substf t A B) -> (T_F B))
    -> (T_F (Forall A))
| T_F_Nabla  : forall A:Var->Form,
    (forall x:Var,(notin_Form x (Nabla A)) -> (T_F (A x)))
    -> (T_F (Nabla A)).
```

where `notin_Form` is as usual in weak HOAS encodings:

```
Inductive notin_Form (x:Var): Form -> Prop :=  ...
```

Formal meaning: `(notin_Form x A)` holds iff $x \notin FV(A)$.

Translating $FO\lambda^{\nabla}$ judgments to CIC using HOAS

Generic judgments are translated by recursion on the length of the local signature, similarly to ∇:

```
Fixpoint T_GJ (n:nat) {struct n} : (GenJ n) -> Prop :=
   match n return (GenJ n) -> Prop with
   | 0 => T_Form
   | (S m) => (fun t:Var->(GenJ m) =>
                forall x:Var, (notin_GenJ (S m) x t)
                                -> (T_GJ m (t x))
              )
   end.
```

Notice

The proof system of $FO\lambda^{\nabla}$ *is not* encoded: only the "intended" meaning of formulas and judgment is captured.

The translation is complete...

Theorem

If $\Sigma : \sigma_1 \rhd B_1, \ldots, \sigma_n \rhd B_n \vdash \sigma \rhd B$ *in* $FO\lambda^{\nabla}$, *then*

```
Goal (T_GJ s1 B1) -> ...  (T_GJ sn Bn) -> (T_GJ s B)
```

is provable in CIC+ToC.

Proof.

Every rule of $FO\lambda^{\nabla}$ translates into a Theorem in CIC, e.g.:

```
Lemma AND_I: forall n:nat, forall A B:GJ n,
     (T_GJ n A) -> (T_GJ n B) -> (T_GJ n (AND n A B)).
```

which can be proved using standard techniques and possibly the axioms from the Theory of Contexts. □

...but CIC+ToC is strictly stronger

Some non-theorems of $FO\lambda^{\nabla}$ are provable in CiC+ToC:

- $\forall x.A \supset \nabla x.A$ translates to

  ```
  (forall t:Term,forall B:Form,(substf t A B)->(T_F B))
  ->
  (forall x:Var,(notin_Form x (Nabla A)) -> (T_F (A x)))
  ```

  directly provable *without* using ToC axioms (apart in properties of `subst`)
- $\nabla x.A \supset \exists x.A$ translates to

  ```
  (forall x:Var,(notin_Forall x (Nabla A)) -> (T_F (A x))
  ->
  (exists t:Term,forall B:Form,(substf t A B)->(T_F B))
  ```

  provable using "freshness" axiom (and properties `subst`)

Conclusions

$FO\lambda^{\nabla}$ is easily translated into CIC with HOAS

The translation is complete, but CIC+ToC is strictly stronger (which is not necessarily good).

(Some) Future work

- How to weaken ToC in order to match $FO\lambda^{\nabla}$'s strength?
- Or shall $FO\lambda^{\nabla}$ be strengthened to match CIC+ToC (and FM-logics), still keeping the good proof theoretical properties?
- Generalize to multi-sorted variable sets
- Move to $FO\lambda^{\Delta\nabla}$, by accommodating also *definitions*.