universität innsbruck

Interactive Theorem Proving

Lecture & Exercises   Week 10

Daniel Ranalter

Summary

Previous Lecture

- Presentations
- λ_2

Today

- Higher Order Logic (HOL)
- Dependently-typed Higher Order Logic (DHOL)

Exercises?

Why HOL?

Simple Type Theory

- theoretic: Type Theory is used as mathematical foundation, created in response to the foundational crisis
- practical: also used as a model of computation (Functional Programming)

Higher Order Logic

- a type of Simple Type Theory
- usually includes quantifiers and “standard” logical connectives

Syntax

HOL Syntax

- Simple Type Theory a la Church with a base-type for booleans, implication and equality

$$
\begin{array}{lcll}
\Gamma & ::= & \circ \mid \Gamma, a\ tp \mid \Gamma, x : A \mid \Gamma, F & \text{context} \\
A, B & ::= & a \mid o \mid A \to B & \text{types} \\
t, u, v & ::= & x \mid \lambda x : A.t \mid tu \mid t \Rightarrow u \mid t =_A u \mid \bot & \text{terms}
\end{array}
$$

- Con- and Disjunction, Quantification, etc can be encoded
- $\forall f : nat \to nat \to nat.((\lambda n : nat.f\ 0\ n) =_{nat \to nat} f\ 0)$

Judgements

What can we do with it?

- $\forall f : nat \to nat \to nat.((\lambda n : nat.f\ 0\ n) =_{nat \to nat} f\ 0)$?
- Syntax has no meaning
- We give meaning by Judgements:

$\Gamma \vdash t$	Well-formed boolean term t is provable
$\Gamma \vdash t : A$	Term t is of (well-formed) type A
$\Gamma \vdash A \equiv B$	Well-formed types A and B are equal
$\Gamma \vdash A\ tp$	Type A is well-formed

Judgements

What can we do with it?

- $\forall f : nat \to nat \to nat.((\lambda n : nat.f\ 0\ n) =_{nat \to nat} f\ 0)$?
- Syntax has no meaning
- We give meaning by Judgements:

$\Gamma \vdash t$	Well-formed boolean term t is provable
$\Gamma \vdash t : A$	Term t is of (well-formed) type A
$\Gamma \vdash A \equiv B$	Well-formed types A and B are equal
$\Gamma \vdash A\ tp$	Type A is well-formed

The missing piece

But how do we arrive at a judgement?

Semantics

Some Natural Deduction Rules

$$\frac{\Gamma \vdash s \colon o \quad \Gamma \vdash t \colon o}{\Gamma \vdash (s \Rightarrow t) \colon o} \Rightarrow\mathrm{Type} \qquad \frac{\Gamma \vdash s \colon o \quad \Gamma, s \vdash t}{\Gamma \vdash s \Rightarrow t} \Rightarrow$$

$$\frac{\Gamma \vdash A \equiv A' \quad \Gamma \vdash B \equiv B'}{\Gamma \vdash A \to B \equiv A' \to B'} \to\mathrm{Cong} \qquad \frac{\Gamma \vdash A\ tp}{\Gamma \vdash A \equiv A} \mathrm{tpRefl}$$

$$\frac{a\ tp \ \in\ \Gamma}{\Gamma \vdash a\ tp} \mathrm{tp}$$

Why DHOL?

Dependent Type Theory

- theoretic: allows to express mathematical concepts like finite, fixed-size sets
- practical: allows to incorporate guards into the level of types (eg. unfailing head function)

DHOL

- combines DTT advantages with HOL comfort
- hopefully (continues to) bridges gap between practitioners and developers
- why? extensional and classical!

Extensions

DHOL Syntax

Now we can extend HOL to dependent types by replacing every occurrence of type-formation...

Γ	$::=$	$\circ \mid \Gamma, a\ tp \mid \Gamma, x : A \mid \Gamma, F$	context
A, B	$::=$	$a \mid o \mid A \to B$	types
t, u, v	$::=$	$x \mid \lambda x : A.t \mid tu \mid t \Rightarrow u \mid t =_A u \mid \bot$	terms

Extensions

DHOL Syntax

Now we can extend HOL to dependent types by replacing every occurrence of type-formation...

$$
\begin{array}{lll r}
\Gamma & ::= & \circ \mid \Gamma, a : (\Pi x : A.)^{*}\ tp \mid \Gamma, x : A \mid \Gamma, F & \text{context} \\
A, B & ::= & a t_1 \ldots t_n \mid o \mid \Pi x : A.B & \text{types} \\
t, u, v & ::= & x \mid \lambda x : A.t \mid t u \mid t \Rightarrow u \mid t =_A u \mid \perp & \text{terms}
\end{array}
$$

... with the more general, dependent variant.

Extensions

DHOL Syntax

Now we can extend HOL to dependent types by replacing every occurrence of type-formation...

$$
\begin{array}{lcll}
\Gamma & ::= & \circ \mid \Gamma, a : (\Pi x : A.)^{*}\ tp \mid \Gamma, x : A \mid \Gamma, F & \text{context} \\
A, B & ::= & a t_1 \ldots t_n \mid o \mid \Pi x : A.B & \text{types} \\
t, u, v & ::= & x \mid \lambda x : A.t \mid tu \mid t \Rightarrow u \mid t =_A u \mid \perp & \text{terms}
\end{array}
$$

... with the more general, dependent variant.

Judgements stay the same but are more complicated now.

Semantics

HOL-ND to DHOL-ND

$$\frac{\Gamma \vdash s : o \quad \Gamma \vdash t : o}{\Gamma \vdash (s \Rightarrow t) : o}{\Rightarrow}\mathrm{Type} \qquad \frac{\Gamma \vdash s : o \quad \Gamma, s \vdash t}{\Gamma \vdash s \Rightarrow t}{\Rightarrow}$$

$$\frac{\Gamma \vdash A \equiv A' \quad \Gamma \vdash B \equiv B'}{\Gamma \vdash A{\to}B \equiv A'{\to}B'}{\to}\mathrm{Cong} \qquad \frac{\Gamma \vdash A\ tp}{\Gamma \vdash A \equiv A}\mathrm{tpRefl}$$

Semantics

HOL-ND to DHOL-ND

$$\frac{\Gamma \vdash s : o \quad \Gamma, s \vdash t : o}{\Gamma \vdash (s \Rightarrow t) : o} \Rightarrow\text{Type} \qquad \frac{\Gamma \vdash s : o \quad \Gamma, s \vdash t}{\Gamma \vdash s \Rightarrow t} \Rightarrow$$

$$\frac{\Gamma \vdash A \equiv A' \quad \Gamma, x : A \vdash B \equiv B'}{\Gamma \vdash \Pi x : A.B \equiv \Pi x' : A'.B'} \Pi\text{Cong} \qquad \frac{\Gamma \vdash A \; tp}{\Gamma \vdash A \equiv A} \text{tpRefl}$$

Semantics

HOL-ND to DHOL-ND

$$\frac{\Gamma \vdash s : o \quad \Gamma, s \vdash t : o}{\Gamma \vdash (s \Rightarrow t) : o} \Rightarrow \mathrm{Type} \qquad \frac{\Gamma \vdash s : o \quad \Gamma, s \vdash t}{\Gamma \vdash s \Rightarrow t} \Rightarrow$$

$$\frac{\Gamma \vdash A \equiv A' \quad \Gamma, x : A \vdash B \equiv B'}{\Gamma \vdash \Pi x : A.B \equiv \Pi x' : A'.B'} \Pi\mathrm{Cong}$$

$$\frac{a : (\Pi x_1 : A_1, \ldots, \Pi x_n : A_n) \in \Gamma \quad \Gamma \vdash s_1 =_{A_1} t_1 \quad \ldots \quad \Gamma \vdash s_n =_{A_n[x_1/s_1, \ldots, x_{n-1}/s_{n-1}]} t_n}{\Gamma \vdash a s_1 \ldots s_n \equiv a t_1 \ldots t_n}$$

tpRefl

Erasure

Simplifying things by making them more complicated

- DHOL is currently barely supported
- To increase usability, a erasure from DHOL to HOL exists
- Basic idea: Capture information lost during erasure in a Partial Equivalence Relation (PER)
- PER $=$ Equivalence Relation $-$ Reflexivity
- PER a^* (for type a) is Equality Relation exactly for elements which are in the "refined" dependent type

$$o^*\ s\ t = \quad s =_o t$$

$$(a\ t_1 \ldots t_n)^*\ s\ t = \quad a^*\ \overline{t_1} \ldots \overline{t_n}\ s\ t$$

$$(\Pi x : A.B)^*\ s\ t = \quad \forall x, y : \overline{A}\,.\, A^*\ x\ y \Rightarrow B^*\ (s\ x)\ (t\ y)$$

Erasure

Erasure, abridged

$\overline{a t_1 \ldots t_n} =$

- a

$\overline{x : A} =$

- $x : \overline{A}$
- $A^* \, x \, x$

$\overline{a : \Pi x_1 : A_1, \ldots, \Pi x_n : A_n \; tp} =$

- $a \; tp$
- $a^* : \overline{A_1} \to \ldots \to \overline{A_n} \to a \to a \to o$
- Axiom(s) establishing PER properties for a^*

$\overline{s =_A t} =$

- $A^* \, \overline{s} \, \overline{t}$

$\overline{\Pi x : A.B} =$

- $\overline{A} \to \overline{B}$

$\overline{\forall \, x : A.t} =$

- $\forall x : \overline{A}.A^* \, x \, x \Rightarrow \overline{t}$