Higher-Order Logic through the Curry-Howard Isomorphism

Similarities

Remember the Π-operator from HOL that constructs dependent types:

$\Pi x{:}X\ Y(x)$: The type of functions that return an inhabitant of $Y(x)$, for each $x{:}X$.

It has a derived form $X \Rightarrow Y$, for the case where x does not occur in Y.

Similarities 2

The Π is similar to $\forall$, and the $\Rightarrow$ is similar to $\rightarrow$:

- If one has a function of type $\Pi x{:}X\ Y(x)$, then one has a method of obtaining an inhabitant of $Y(x)$ for each $x{:}X$ that one can construct.
- If one has a function of type $X \Rightarrow Y$, then one has a method of constructing an inhabitant of Y from each inhabitant of X that one can construct.

Curry-Howard isomorphism

The similarity can be made explicit in the Formulas as Types paradigm:

Identify a proposition with the set of its proofs.

- Proving a formula becomes: Finding an inhabitant of the formula.
- Declaring a formula becomes: Declaring an arbitrary inhabitant of the formula.
- Checking a proof becomes: Checking that a λ-term has a certain type.

Consequences

- No logical calculus is needed. The rules for defining types have become logical rules as well. The rules define HOL with $\forall$ and $\rightarrow$. (Other operators can be defined, as with HOL)
- Because proofs are inhabitants of types of the λ-calculus, proofs are programs that can be executed. The calculus defines methods of automatically deriving programs. (It should be said that good proofs usually give bad programs. Good programs are extracted from bad proofs)

Notation

For the type checking calculus, we use the typing rules of HOL, with the following (notation only!) modifications:

- $\Rightarrow$ is replaced by $\rightarrow$.
- Type is (sometimes) called Set.
- Form is (sometimes) called Prop.

Developing a Theory

One formalizes a mathematical theory, by building up a context Γ : Γ consists of

Declarations These are either real declarations, or definitions of axioms. A definition of an axiom has form $x{:}X$, where X is the axiom, and x is the name that we will use for it.

Definitions These are either real definitions, or theorems. A proof of a theorem has form $x := y{:}Y$. Here Y is the theorem, y is the proof, and x is the name of the theorem.

Intuitionistic vs. Classical

The typing rules of HOL, define intuitionistic Higher Order Logic. (Essentially because they have form $\Gamma \vdash x{:}X$ with only one formula on the right)

One can obtain classical Higher Order Logic from this by adding the law of excluded middle.

In practice one should not do this. It is better to add weak forms of the law of excluded middle for the types that one needs.

Embedding of FOL into Type Theory

As with HOL, first order logic can be defined in two ways:

By declaration One declares the logical operators and gives axioms that describe their behaviour.

By definition One defines the logical operators by terms that behave in the way the operators should.

Declaration of Logical Operators

$\bot : \mathrm{Prop}$

$\top : \mathrm{Prop}$

$\neg : \mathrm{Prop} \to \mathrm{Prop}$

$\vee : \mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop}$

$\wedge : \mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop}$

$\leftrightarrow : \mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop}$

$\exists : \Pi D{:}\mathrm{Set}\ P{:}D \to \mathrm{Prop}\ (P \to \mathrm{Prop}) \to \mathrm{Prop}$

$\approx : \Pi D{:}\mathrm{Set}\ D \to D \to \mathrm{Prop}$

Axioms for Logical Operators (1)

$\forall P{:}\,\mathrm{Prop}\ \bot \to P$

$\top$

$\forall P{:}\,\mathrm{Prop}\ (P \to \bot) \to \neg P$

$\forall P{:}\,\mathrm{Prop}\ P \to (\neg P) \to \bot$

Axioms for Logical Operators (2)

$\forall P, Q{:}\,\text{Prop}\ P \to Q \to P \wedge Q$
$\forall P, Q{:}\,\text{Prop}\ P \wedge Q \to P$
$\forall P, Q{:}\,\text{Prop}\ P \wedge Q \to Q$

$\forall P, Q{:}\,\text{Prop}\ P \to P \vee Q$
$\forall P, Q{:}\,\text{Prop}\ Q \to P \vee Q$
$\forall P, Q{:}\,\text{Prop}\ P \vee Q \to \forall C{:}\,\text{Prop}\ (P \to C) \to (Q \to C) \to C$

$\forall P, Q{:}\,\text{Prop}\ (P \to Q) \to (Q \to P) \to (P \leftrightarrow Q)$
$\forall P, Q{:}\,\text{Prop}\ (P \leftrightarrow Q) \to (P \to Q)$
$\forall P, Q{:}\,\text{Prop}\ (P \leftrightarrow Q) \to (Q \to P)$

Axioms for Logical Operators (3)

$$\forall D{:}\,\mathrm{Type}\ \forall P{:}\,D \to \mathrm{Prop}\ \forall d{:}\,D\ (P\ d) \to (\exists\ D\ P)$$

$$\forall D{:}\,\mathrm{Type}\ \forall P{:}\,D \to \mathrm{Prop}$$
$$(\exists\ D\ P) \to \forall C{:}\,\mathrm{Prop}\ (\forall d{:}\,D\ (P\ d) \to C) \to C$$

$$\forall D{:}\,\mathrm{Type}\ (\approx\ D\ d\ d)$$

$$\forall D{:}\,\mathrm{Type}\ \forall d_1, d_2{:}\,D \to \mathrm{Prop}$$
$$(\approx\ D\ d_1\ d_2) \to \forall P{:}\,D \to \mathrm{Prop}\ (P\ d_1) \to (P\ d_2)$$

Definitions of Logical Operators

Defining the Logical Operators has the advantage that the logical operators are invisible, because definitions can be expanded without additional proof steps. This makes the proofs usually shorter. Unfortunately proofs also become harder to understand.

$\bot := \Pi P{:}\,\mathrm{Prop}{:}\,\mathrm{Prop}$

$\neg := \lambda P{:}\ \mathrm{Prop}\ (P \to \bot){:}\ \mathrm{Prop} \to \mathrm{Prop}$

$\vee := \lambda P, Q{:}\,\mathrm{Prop}$

$\qquad\qquad \forall R{:}\,\mathrm{Prop}\ (P \to R) \to (Q \to R) \to R{:}\,\mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop}$

$\wedge := \lambda P, Q{:}\,\mathrm{Prop}$

$\qquad\qquad \forall R{:}\,\mathrm{Prop}\ (P \to Q \to R) \to R{:}\,\mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop}$

$\leftrightarrow := \lambda P, Q{:}\,\mathrm{Prop}\ (P \to Q) \wedge (Q \to P){:}\,\mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop}$

$$\exists := \lambda D{:}\,\mathrm{Type}\ \lambda P{:}\,D \to \mathrm{Prop}$$
$$\forall Q{:}\,\mathrm{Prop}\ (\forall d{:}\,D\ (P\ d) \to Q) \to Q$$
$$:\Pi D{:}\,\mathrm{Type}\ \Pi P{:}\,D \to \mathrm{Prop}\ (P \to \mathrm{Prop}) \to \mathrm{Prop}$$

$$\approx := \lambda D{:}\,\mathrm{Type}\ \lambda d_1, d_2{:}\,D$$
$$\forall P{:}\,D \to \mathrm{Prop}\ (P\ d_1) \to (P\ d_2){:}\,\Pi D{:}\,\mathrm{Type}\ D \to D \to \mathrm{Prop}$$

An alternative definition is the following:

$$\approx := \lambda D{:}\,\mathrm{Type}\ \lambda d_1, d_2{:}\,D$$
$$\forall P{:}\,D \to D \to \mathrm{Prop}\ \forall d{:}\,D\ (P\ d\ d) \to (P\ d_1\ d_2){:}$$
$$\Pi D{:}\,\mathrm{Type}\ D \to D \to \mathrm{Prop}$$