Logic and type theory

Ansten Klev
Czech Academy of Sciences

Logic department at Celetná

Type theory

Type theory is the theory of types from logic and mathematics.

- natural number, integer, real number, set, function, ...
- proposition, propositional connective, quantifier, ...

Type theory makes judgements of two forms:

$$
\begin{array}{rl}
a : \alpha & \text{“}a\text{ is an object of type }\alpha\text{”} \\
a = b : \alpha & \text{“}a\text{ and }b\text{ are equal objects of type }\alpha\text{”}
\end{array}
$$

Propositional logic: language

Uninterpreted propositions	$p_0, p_1, p_2, \ldots$
Absurdity	$\perp$
Negation (no, not)	$\neg A$
Conjunction (and)	$A \wedge B$
Disjunction (or)	$A \vee B$
Implication (if – then)	$A \supset B$

Propositional logic: theorems and rules

Examples of theorems

$$A \supset A$$

$$A \supset (B \supset A)$$

$$A \supset (A \vee B)$$

$$A \supset (B \supset (A \wedge B))$$

Examples of rules

$$\frac{A \supset B \qquad A}{B} \qquad \frac{A \qquad B}{A \wedge B} \qquad \frac{A}{A \vee B} \qquad \frac{B}{A \vee B}$$

Logic in type theory?

How can we express logic in type theory?

We want to say, e.g., that

$$A \supset A$$

is a true proposition.

But in the language type theory, everything we can say must have either of these two forms:

$$a : \alpha$$
$$a = b : \alpha$$

Solution: propositions as types

There is a type of propositions.

We stipulate that also every proposition itself is a type.

So, if A is a proposition, it is also a type.

Which type? The type of its truthmakers (proofs).

What is a proposition?

A common answer:

- A set of possible worlds.

Useless in a logical analysis of mathematics, since there will then be only two propositions.

We stipulate instead

- A type of truthmakers/proofs.

This account allows for there be infinitely many propositions.

Truthmakers

A truthmaker is an object that makes a proposition true.

Example: the red of this rose is a (the) truthmaker of

This rose is red.

Truthmaker theory is one way of making precise the idea that truth is correspondence with "reality".

In a typed universe, every object has a type (its genus, kind, sort).

The type of a truthmaker a is the proposition that a makes true.

Truth

Let A be a proposition. The judgement

$$a : A$$

should be read as "a is a truthmaker of A".

If that judgement is correct, then A has a truthmaker, hence we can infer

$$A \text{ is true}$$

So

$$\text{truth} = \text{existence of truthmaker}$$

Logic in type theory

We wish to define each proposition A as a type, such that

$$A \text{ is true (is a theorem)}$$

if, and only if, A is inhabited as a type, i.e., there is some a such that

$$a : A$$

is correct.

Conjunction

Logic

$$\frac{A \qquad B}{A \wedge B}$$

$$\frac{A \wedge B}{A} \qquad\qquad \frac{A \wedge B}{B}$$

Type theory

$$\frac{a : A \qquad b : B}{\langle a, b \rangle : A \times B}$$

$$\frac{c : A \times B}{\mathsf{fst}(c) : A} \qquad\qquad \frac{c : A \times B}{\mathsf{snd}(c) : B}$$

Implication

Logic

$$\dfrac{\begin{array}{c}[A] \\ | \\ B\end{array}}{A \supset B} \qquad \dfrac{A \supset B \qquad A}{B}$$

Type theory

$$\dfrac{\begin{array}{c}[x : A] \\ | \\ b : B\end{array}}{\lambda x.b : A \to B} \qquad \dfrac{f : A \to B \qquad a : A}{f(a) : B}$$

Absurdity and negation

Logic

$$\frac{\perp}{C}$$

Define negation by

$$\neg A \equiv A \supset \perp$$

Type theory

$$\frac{a : \mathbb{O}}{R_{\mathbb{O}}(a) : C}$$

Define a negation operation on types:

$$\neg A \equiv A \to \mathbb{O}$$

Disjunction

Logic

$$\frac{A}{A \vee B} \qquad \frac{B}{A \vee B} \qquad \frac{A \vee B \qquad \begin{array}{c}[A]\\ \vdots \\ C\end{array} \qquad \begin{array}{c}[B]\\ \vdots \\ C\end{array}}{C}$$

Type theory

$$\frac{a : A}{\mathrm{inl}(a) : A + B} \qquad \frac{b : B}{\mathrm{inr}(b) : A + B}$$

$$\frac{c : A + B \qquad \begin{array}{c}[x : A]\\ \vdots \\ d(x) : C\end{array} \qquad \begin{array}{c}[y : B]\\ \vdots \\ e(y) : C\end{array}}{\mathrm{D}(c, x.d(x), y.e(y)) : C}$$

Uninterpreted propositions

Logic

$$p_0, p_1, p_2, \ldots$$

Type theory

$$X_0, X_1, X_2, \ldots \text{ (type variables)}$$

Curry–Howard correspondence

A translation from type theory to logic.

type theory	logic
A	A^*
X_i	p_i
$\mathbb{0}$	$\perp$
$A \times B$	$A^* \wedge B^*$
$A \to B$	$A^* \to B^*$
$A + B$	$A^* \vee B^*$

Curry–Howard isomorphism

Theorem

Let A be a type built up from type variables and $\mathbb{O}$ by means of $\times$, $\to$, and $+$. Then there is a closed term a such that the judgement

$$a : A$$

is derivable in type theory if, and only if, A^* is a theorem of intuitionistic logic.

In fact, from the term a we can reconstruct a derivation of A^* in intuitionistic logic.

Predicate logic

In addition to the logical connectives from propositional logic, we need

- Individual domain. Let us choose $\mathbb{N}$.
- Predicates. Let us choose the binary identity predicate, $=$.
- Quantifiers, $\forall$ (for all) and $\exists$ (there is).

Individual domain

We already know how to think of $\mathbb{N}$ type-theoretically.

Identity

Logic

$$n = n \qquad\qquad \frac{m = n \qquad A(m)}{A(n)}$$

Type theory

Corresponding to the proposition $m = n$, there is a type $\mathbb{I}(m, n)$.

$$\frac{n : \mathbb{N}}{\mathrm{refl}(n) : \mathbb{I}(n, n)} \qquad\qquad \frac{p : \mathbb{I}(m, n) \qquad a : A(m)}{\mathrm{subst}(p, a) : A(n)}$$

Type families

As m and n vary, so does the type $\mathbb{I}(m, n)$.

- The type $\mathbb{I}(m, n)$ depends on m and n.

We call $\mathbb{I}(m, n)$ a type family (over $\mathbb{N}$).

Once we have one type family, we can form more by means type-theoretical operators.

Universal quantifier

Logic

$$\frac{A(x)}{\forall x\, A(x)} \qquad\qquad \frac{\forall x\, A(x)}{A(a)}$$

Type theory

Corresponding to an open formula $A(x)$, there is a type family $A(x)$ over $\mathbb{N}$.
Corresponding to the proposition $\forall x\, A(x)$, there is a type $(\Pi x : \mathbb{N})A(x)$.

$$\frac{\begin{array}{c} x : \mathbb{N} \\ \vdots \\ a(x) : A(x) \end{array}}{\lambda x.a(x) : (\Pi x : \mathbb{N})A(x)} \qquad\qquad \frac{f : (\Pi x : \mathbb{N})A(x) \qquad n : \mathbb{N}}{f(n) : A(n)}$$

Note that the type of $f(n)$ depends on n.

Full type theory

So far we have assumed that there is just one individual domain $\mathbb{N}$.

To move to full type theory, let every type (including every proposition) be an individual domain.

Then we can form the identity type

$$\mathbb{I}_A(a, b)$$

for every type A and objects $a, b : A$.

And we can form the dependent function type

$$(\Pi x : A)B(x)$$

for every type A and type family $B(x)$ over A.

Summary

- Type theory is a theory of mathematical and logical types.
- It makes judgements of the form $a : \alpha$ and $a = b : \alpha$.
- An important case is where α is an inductively defined type.
- In fact, all the types considered in these two lectures can be seen to be inductively defined.
- Logic is embedded into type theory by treating propositions as types.
- This embedding yields a close correspondence between type theory and intuitionistic logic.