

Type Theory: Impredicative Part (4/5)

## Higher-Order Logic

Alexandre Miquel

Paris-sud University – Orsay, France

Alexandre.Miquel@lri.fr

## Church's theory of simple types

**Types**  $\tau, \sigma ::= \text{Prop} \mid \tau \rightarrow \sigma$

**Terms**  $M, N, A, B ::= x \mid \lambda x : \tau . M \mid MN$   
 $\mid A \Rightarrow B \mid \forall x : \tau . B$

**Reduction**  $(\lambda x : \tau . M)N \succ M\{x := N\}$

### Remarks:

- ▶ Types are the simple types with one ground type **Prop** (type of **propositions**)
- ▶ Terms are the terms of the simply typed  $\lambda$ -calculus with specific constructions  $A \Rightarrow B$  and  $\forall x : \tau . A$  to build propositions (i.e. terms of type Prop)
- ▶ Reduction is the usual  $\beta$ -reduction rule  
[One can also extend the system with extra reduction rules for specific purposes. . .]

## Signatures and typing rules

- **Signatures** (or typing contexts) are finite lists of the form

$$\Sigma ::= x_1 : \tau_1, \dots, x_n : \tau_n$$

[Signatures should not be confused with (logical) contexts that will be defined later]

- **Typing rules:**

$$\frac{}{\Sigma \vdash x : \tau} \quad (x:\tau) \in \Sigma$$

$$\frac{\Sigma, x : \tau \vdash M : \sigma}{\Sigma \vdash \lambda x : \tau . M : \tau \rightarrow \sigma}$$

$$\frac{\Sigma \vdash M : \tau \rightarrow \sigma \quad \Sigma \vdash N : \tau}{\Sigma \vdash MN : \sigma}$$

$$\frac{\Sigma \vdash A : \text{Prop} \quad \Sigma \vdash B : \text{Prop}}{\Sigma \vdash A \Rightarrow B : \text{Prop}}$$

$$\frac{\Sigma, x : \tau \vdash B : \text{Prop}}{\Sigma \vdash \forall x : \tau . B : \text{Prop}}$$

## Intuitionistic natural deduction

- **Logical contexts:**  $\Gamma ::= A_1, \dots, A_n$

A logical context  $\Gamma$  is **well-formed** in a signature  $\Sigma$  (notation:  $\Sigma \vdash \Gamma$  context) if  $\Sigma \vdash A : \text{Prop}$  for each  $A \in \Gamma$

- **Deduction rules:**

$$\frac{\Sigma \vdash \Gamma \text{ context}}{\langle \Sigma \rangle \Gamma \vdash A} \quad A \in \Gamma$$

$$\frac{\langle \Sigma \rangle \Gamma, A \vdash B}{\langle \Sigma \rangle \Gamma \vdash A \Rightarrow B}$$

$$\frac{\langle \Sigma \rangle \Gamma \vdash A \Rightarrow B \quad \langle \Sigma \rangle \Gamma \vdash A}{\langle \Sigma \rangle \Gamma \vdash B}$$

$$\frac{\langle \Sigma, x : \tau \rangle \Gamma \vdash B}{\langle \Sigma \rangle \Gamma \vdash \forall x : \tau . B} \quad x \notin FV(\Gamma)$$

$$\frac{\langle \Sigma \rangle \Gamma \vdash \forall x : \tau . B \quad \Sigma \vdash N : \tau}{\langle \Sigma \rangle \Gamma \vdash B\{x := N\}}$$

+ conversion rule (next slide)

## The conversion rule

The last deduction rule

$$\frac{\langle \Sigma \rangle \Gamma \vdash A \quad \Sigma \vdash A' : \text{Prop}}{\langle \Sigma \rangle \Gamma \vdash A'} \quad A =_{\beta} A'$$

allows to perform any computation in a proposition.

**Example:** Identity function  $\text{id}_{\tau} \equiv \lambda x : \tau . x$  always returns its argument:

$$\frac{\frac{\frac{\langle x : \tau, p : \tau \rightarrow \text{Prop} \rangle px \vdash px}{\langle x : \tau, p : \tau \rightarrow \text{Prop} \rangle \vdash px \Rightarrow px}}{\langle x : \tau \rangle \vdash \underbrace{\forall p : \tau \rightarrow \text{Prop} . px \Rightarrow px}_{x =_{\tau} x}}}{\langle \rangle \vdash \forall x : \tau . x =_{\tau} x} \quad \vdash \forall x : \tau . \text{id } x \overset{\vdots}{=}_{\tau} x : \text{Prop}}{\langle \rangle \vdash \forall x : \tau . \text{id } x =_{\tau} x}$$

## Definable constructions

- **Logical connectives**

$$\top \quad \equiv \quad \forall p : \text{Prop} . p \Rightarrow p$$

$$\perp \quad \equiv \quad \forall p : \text{Prop} . p$$

$$A \wedge B \quad \equiv \quad \forall p : \text{Prop} . (A \Rightarrow B \Rightarrow p) \Rightarrow p$$

$$A \vee B \quad \equiv \quad \forall p : \text{Prop} . (A \Rightarrow p) \Rightarrow (B \Rightarrow p) \Rightarrow p$$

$$\neg A \quad \equiv \quad A \Rightarrow \perp$$

$$A \Leftrightarrow B \quad \equiv \quad (A \Rightarrow B) \wedge (B \Rightarrow A)$$

- **Existential quantifier**

$$\exists x : \tau . A[x] \quad \equiv \quad \forall p : \text{Prop} . (\forall x : \tau . A[x] \Rightarrow p) \Rightarrow p$$

- **Leibniz equality**

$$M_1 =_{\tau} M_2 \quad \equiv \quad \forall p : \tau \rightarrow \text{Prop} . p M_1 \Rightarrow p M_2$$

- **Extensional equality**

$$P_1 =_{\tau \rightarrow \text{Prop}}^{\text{ext}} P_2 \quad \equiv \quad \forall x : \tau . P_1 x \Leftrightarrow P_2 x$$

## Adding natural numbers

To be able to express (higher-order) arithmetic, we **extend the system with**

- ▶ A new ground type **Nat** with two constructors **0** : Nat and **s** : Nat → Nat
- ▶ Two constants (destructors) **pred** : Nat → Nat and **null** : Nat → Prop with computational rules

$$\begin{array}{ll} \text{pred } 0 & \gamma \quad 0 \\ \text{pred } (sn) & \gamma \quad n \end{array} \qquad \begin{array}{ll} \text{null } 0 & \gamma \quad \top \\ \text{null } (sn) & \gamma \quad \perp \end{array}$$

that are incorporated into the conversion rule, so that we can prove:

$$\forall x, y : \text{Nat} . sx =_{\text{Nat}} sy \Rightarrow x =_{\text{Nat}} y \quad \text{and} \quad \forall x : \text{Nat} . \neg sx =_{\text{Nat}} 0$$

- ▶ As usual, we recover induction by restricting quantifications with the predicate:

$$\text{is\_nat}(x) \equiv \forall p : \text{Nat} \rightarrow \text{Prop} . P 0 \Rightarrow (\forall y : \text{Nat} . P y \Rightarrow P (sy)) \Rightarrow Px$$

## System $F\omega$

**Principle:** Church's theory of simple types with **proof-terms**

**Types**  $\tau, \sigma ::= \text{Prop} \mid \tau \rightarrow \sigma$

**Terms**  $M, N, A, B ::= x \mid \lambda x : \tau . M \mid MN$   
 $\mid A \Rightarrow B \mid \forall x : \tau . B$

**Reduction**  $(\lambda x : \tau . M)N \succ M\{x := N\}$

**Proof-terms**  $t, u ::= \xi$  (Axiom)  
 $\mid \lambda \xi^A . t \mid tu$  ( $\Rightarrow$ -intro/elim)  
 $\mid \lambda x : \tau . t \mid tN$  ( $\forall$ -intro/elim)

**Proof reduction**  $(\lambda \xi^A . t)u \succ t\{\xi := u\}$  ( $\Rightarrow$  cut)  
 $(\lambda x : \tau . t)N \succ t\{x := N\}$  ( $\forall$  cut)

## Typing proof-terms

- **Logical contexts:**  $\Gamma ::= \xi_1 : A_1, \dots, \xi_n : A_n$   
 $\Rightarrow$  ‘ $\Sigma \vdash \Gamma$  context’ now expresses that  $\Sigma \vdash A : \text{Prop}$  for all  $(\xi : A) \in \Gamma$
- **Typing rules for proof-terms:**

$$\frac{\Sigma \vdash \Gamma \text{ context}}{\langle \Sigma \rangle \Gamma \vdash \xi : A} \quad (\xi:A) \in \Gamma$$

$$\frac{\langle \Sigma \rangle \Gamma, A \vdash t : B}{\langle \Sigma \rangle \Gamma \vdash \lambda \xi^A. t : A \Rightarrow B}$$

$$\frac{\langle \Sigma \rangle \Gamma \vdash t : A \Rightarrow B \quad \langle \Sigma \rangle \Gamma \vdash u : A}{\langle \Sigma \rangle \Gamma \vdash tu : B}$$

$$\frac{\langle \Sigma, x : \tau \rangle \Gamma \vdash t : B}{\langle \Sigma \rangle \Gamma \vdash \lambda x : \tau. t : \forall x : \tau. B} \quad x \notin FV(\Gamma)$$

$$\frac{\langle \Sigma \rangle \Gamma \vdash t : \forall x : \tau. B \quad \Sigma \vdash N : \tau}{\langle \Sigma \rangle \Gamma \vdash tN : B\{x := N\}}$$

$$\frac{\langle \Sigma \rangle \Gamma \vdash t : A \quad \Sigma \vdash A' : \text{Prop}}{\langle \Sigma \rangle \Gamma \vdash t : A'} \quad A =_{\beta} A'$$