

## Normalisation of Second Order Arithmetic

Alexandre Miquel — PPS &amp; U. Paris 7

Alexandre.Miquel@pps.jussieu.fr

Types Summer School 2005

August 15–26 — Göteborg

<b>Variables</b>	$x, y, z, \dots$ $\alpha^n, \beta^n, \gamma^n, \dots$	of individuals of predicates	(i.e. natural numbers) (for each arity $n \geq 0$ )
<b>Individuals</b>	$t, u$	$::= x \mid 0 \mid s(t)$	
<b>Formulæ</b>	$A, B$	$::= \alpha^n(t_1, \dots, t_n)$ $\mid A \Rightarrow B$ $\mid \forall x B$ $\mid \forall \alpha^n B$	(for all $n \geq 0$ ) (first-order) (second order, for all $n \geq 0$ )
<b>Contexts</b>	$\Gamma, \Delta$	$::= A_1, \dots, A_n$	(lists of formulæ)

- Predicate variables of arity 0 represent **propositions**
- Predicate variables represent **sets** (of numerals, of pairs, etc.)
- **Real numbers** can be represented as predicate variables (**intuitionistic analysis**)

## Substitution

- **Term substitution**  $u\{x := t\} \Rightarrow$  defined in the usual way
- **First-order substitution**  $B\{x := t\} \Rightarrow$  defined in the usual way
- **Second-order substitution**  $B\{\alpha^n := \lambda x_1, \dots, x_n. A\}$

In the formula  $B$ , replace each atomic subformula of the form

$$\alpha^n(t_1, \dots, t_n)$$

by the (substituted) formula

$$A\{x_1 := t_1; \dots; x_n := t_n\}$$

The notation ' $\lambda x_1, \dots, x_n. A$ ' is not part of the syntax

## Encoding missing constructions

- Other connectives can be encoded:

$$\top \equiv \forall \gamma^0 (\gamma^0 \Rightarrow \gamma^0)$$

$$\perp \equiv \forall \gamma^0 \gamma^0$$

$$A \wedge B \equiv \forall \gamma^0 ((A \Rightarrow B \Rightarrow \gamma^0) \Rightarrow \gamma^0)$$

$$A \vee B \equiv \forall \gamma^0 ((A \Rightarrow \gamma^0) \Rightarrow (B \Rightarrow \gamma^0) \Rightarrow \gamma^0)$$

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

- Existential quantifier (1st + 2nd order)

$$\exists x B[x] \equiv \forall \gamma^0 (\forall x (B[x] \Rightarrow \gamma^0) \Rightarrow \gamma^0)$$

$$\exists \alpha^n B[\alpha^n] \equiv \forall \gamma^0 (\forall \alpha^n (B[\alpha^n] \Rightarrow \gamma^0) \Rightarrow \gamma^0)$$

- Leibniz equality:

$$t = u \equiv \forall \gamma^1 (\gamma^1(t) \Rightarrow \gamma^1(u))$$

## Deduction rules of HA2

- General rules for second-order intuitionistic logic:

$$\frac{}{\Gamma \vdash A} \quad A \in \Gamma$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}$$

$$\frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}$$

$$\frac{\Gamma \vdash B}{\Gamma \vdash \forall x B} \quad x \notin FV_1(\Gamma)$$

$$\frac{\Gamma \vdash \forall x B}{\Gamma \vdash B\{x := t\}}$$

$$\frac{\Gamma \vdash B}{\Gamma \vdash \forall \alpha^n B} \quad \alpha^n \notin FV_2(\Gamma)$$

$$\frac{\Gamma \vdash \forall \alpha^n B}{\Gamma \vdash B\{\alpha := \lambda x_1, \dots, x_n . A\}}$$

- Specific rules (axioms) for arithmetic:

$$\frac{}{\Gamma \vdash \forall x \forall y (s(x) = s(y) \Rightarrow x = y)}$$

$$\frac{}{\Gamma \vdash \forall x \neg s(x) = 0}$$



Remember that constructions ' $t = u$ ' and ' $\neg A$ ' are not primitive, but encoded!

## Derivable rules (2/2)

- Logical connectives:  $\vee$

$$\frac{\Gamma \vdash A}{\Gamma \vdash A \vee B} \quad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}$$

$$\frac{\Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma \vdash C} \quad \frac{\Gamma \vdash A \vee B \quad \Gamma, B \vdash C}{\Gamma \vdash C}$$

- Existential quantifier: 1st and 2nd-order

$$\frac{\Gamma \vdash B\{x := t\}}{\Gamma \vdash \exists x B}$$

$$\frac{\Gamma, B \vdash C \quad \Gamma \vdash \exists x B}{\Gamma \vdash C} \quad x \notin FV_1(\Gamma, C)$$

$$\frac{\Gamma \vdash B\{\alpha^n := \lambda x_1, \dots, x_n . A\}}{\Gamma \vdash \exists \alpha^n B}$$

$$\frac{\Gamma, B \vdash C \quad \Gamma \vdash \exists \alpha^n B}{\Gamma \vdash C} \quad \alpha^n \notin FV_2(\Gamma, C)$$

## Derivable rules (1/2)

Logical deduction rules of HA2 only talk about the **primitive constructions** ' $\Rightarrow$ ' and ' $\forall$ ' (implication + 1st/2nd-order universal quantification)

But in this framework, the other constructions ( $\top$ ,  $\perp$ ,  $\wedge$ ,  $\vee$ ,  $\exists$  etc.) are **definable** and their (standard) deduction rules can be derived:

- Logical connectives:  $\top$ ,  $\perp$  and  $\wedge$

$$\frac{}{\Gamma \vdash \top} \quad \frac{\Gamma \vdash \perp}{\Gamma \vdash \perp}$$

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \quad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A} \quad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}$$

## Equality rules

Leibniz equality is defined as:  $t = u \equiv \forall \gamma^1 (\gamma^1(t) \Rightarrow \gamma^1(u))$

- The following formulæ are **provable** (by purely logical means):

$$\forall x (x = x)$$

$$\forall x \forall y (x = y \Rightarrow y = x)$$

$$\forall x \forall y \forall z (x = y \Rightarrow y = z \Rightarrow x = z)$$

$$\forall \alpha^1 \forall x \forall y (\alpha^1(x) \Rightarrow x = y \Rightarrow \alpha^1(y))$$

- Moreover, HA2 assumes the following two **axioms**:

$$\text{(Injectivity)} \quad \forall x \forall y (s(x) = s(y) \Rightarrow x = y)$$

$$\text{(Non-surjectivity)} \quad \forall x \neg (s(x) = 0)$$

## Induction principle

- **Induction** can be recovered via the predicate:

$$\text{Nat}(x) \equiv \forall \alpha^1 \left( \alpha^1(0) \Rightarrow \forall y (\alpha^1(y) \Rightarrow \alpha^1(s(y))) \Rightarrow \alpha^1(x) \right)$$

$\Rightarrow$  defines the **smallest class** containing zero and closed under successor

- In particular, we have:  $\text{Nat}(0)$  and  $\forall x (\text{Nat}(x) \Rightarrow \text{Nat}(s(x)))$

- All the first-order quantifications should be restricted to this class:

$$\Rightarrow \text{Systematically use } \forall x (\text{Nat}(x) \Rightarrow A) \text{ and } \exists x (\text{Nat}(x) \wedge A)$$

- Thanks to this trick, induction becomes provable:

$$\forall \alpha^1 \left( \alpha^1(0) \Rightarrow \forall x (\text{Nat}(x) \Rightarrow \alpha^1(x) \Rightarrow \alpha^1(s(x))) \Rightarrow \forall x (\text{Nat}(x) \Rightarrow \alpha^1(x)) \right)$$

## The notion of cut (1/2)

- A **cut** is a piece of a proof constituted by an **introduction rule** immediately followed by the corresponding **elimination rule**

- Each cut can be contracted in order to make the reasoning more direct. . .  
... but not necessarily shorter [And actually, usually larger!]

- **Implication cut:**

$$\frac{\frac{[\Gamma, A, \Gamma' \vdash A] \quad \vdots \pi_1}{\Gamma, A \vdash B} \quad \frac{\vdots \pi_2}{\Gamma \vdash A}}{\Gamma \vdash A \Rightarrow B} \quad \Gamma \vdash B \quad \rightsquigarrow \quad \frac{\vdots \pi_2}{\Gamma, \Gamma' \vdash A} \quad \vdots \pi_1}{\Gamma \vdash B}$$

Here,  $[\Gamma, A, \Gamma' \vdash A]$  represents all the instances of an axiom with the formula  $A$  in the proof  $\pi_1$ . (Such instances may occur in extended contexts of the form  $\Gamma, A, \Gamma'$ .) These instances are then used as **placeholders** that are filled by the proof  $\pi_2$  during the contraction of the cut (after some weakenings due to the presence of extra contexts  $\Gamma'$ )

## The notion of cut (2/2)

- **Cut of the 1st-order universal quantification:**

$$\frac{\frac{\vdots \pi}{\Gamma \vdash B}}{\Gamma \vdash \forall x. B} \quad \rightsquigarrow \quad \frac{\vdots \pi\{x:=t\}}{\Gamma \vdash B\{x:=t\}}$$

The first piece of proof is replaced by the proof  $\pi$  in which the 1st-order variable  $x$  is replaced by the term  $t$  recursively. Notice that the substitution has no effect on  $\Gamma$ , since  $x \notin FV(\Gamma)$ . (Of course, the substitution has to be performed on each context too.)

- **Cut of the 2nd-order universal quantification:**

$$\frac{\frac{\vdots \pi}{\Gamma \vdash B}}{\Gamma \vdash \forall \alpha^n. B} \quad \rightsquigarrow \quad \frac{\vdots \pi\{\alpha^n := \dots\}}{\Gamma \vdash B\{\alpha^n := \lambda x_1, \dots, x_n. A\}}$$

Same principle, but with a 2nd-order substitution (ie. with a predicate  $\lambda x_1, \dots, x_n. A$ )

## Derived cuts

From the encoding of the connectives  $\wedge$  and  $\vee$ , one can derive other cuts:

- **Cuts of the conjunction:**

$$\frac{\frac{\vdots \pi_1}{\Gamma \vdash A} \quad \frac{\vdots \pi_2}{\Gamma \vdash B}}{\Gamma \vdash A \wedge B} \quad \rightsquigarrow \quad \frac{\vdots \pi_1}{\Gamma \vdash A} \quad (+ \text{ symmetric cut with } \wedge\text{-elim}_2)$$

- **Cuts of the disjunction:**

$$\frac{\frac{[\Gamma, A, \Gamma' \vdash A] \quad \vdots \pi_1}{\Gamma, A \vdash C} \quad \frac{[\Gamma, B, \Gamma' \vdash B] \quad \vdots \pi_2}{\Gamma, B \vdash C} \quad \frac{\vdots \pi}{\Gamma \vdash A \vee B}}{\Gamma \vdash C} \quad \rightsquigarrow \quad \frac{\vdots \pi}{\Gamma, \Gamma' \vdash A} \quad \vdots \pi_1}{\Gamma \vdash C}$$

(+ symmetric cut with  $\vee$ -intro<sub>2</sub>)

Filling placeholders in  $\pi_1$  with  $\pi$  is done in the same way as for the cut of implication

## Cut-free proofs

A **cut-free proof** is a proof that contains no cut

⇒ Cut-free proofs have a simpler structure that make them easier to analyse

### Fact (Cut-free consistency)

- 1 If  $\pi$  is a cut-free proof of the formula  $t = u$  [ $\equiv \forall \alpha^1 (\alpha^1(t) \Rightarrow \alpha^1(u))$ ] in the empty context, then the terms  $t$  and  $u$  are **syntactically identical**
- 2 There is no cut-free proof of  $\perp$  [ $\equiv \forall \alpha^0 \alpha^0$ ] in the empty context

Proof. Both properties are proved simultaneously by induction on the size of the cut-free proof. Notice that a cut-free proof of  $\vdash t = t$  has one of the following two forms:

$$\frac{\frac{\alpha^1(t) \vdash \alpha^1(t)}{\vdash \alpha^1(t) \Rightarrow \alpha^1(t)}}{\vdash \underbrace{\forall \alpha^1 (\alpha^1(t) \Rightarrow \alpha^1(t))}_{t=t}}$$

$$\frac{\frac{\frac{\vdash \forall x \forall y (s(x) = s(y) \Rightarrow x = y)}{\vdash \forall y (s(t) = s(y) \Rightarrow t = y)} \quad (\text{cut-free})}{\vdash s(t) = s(t) \Rightarrow t = t} \quad \vdash s(t) = s(t)}{\vdash t = t}$$

⇒ Reasoning on cut-free proofs is **purely combinatorial**

## Outline of the proof

**Idea:** Deduce cut-elimination of HA2 from strong normalisation of system  $F$

- 1 Map each formula  $A$  of HA2 to a type  $A^*$  of system  $F$
- 2 Map each logical context  $\Gamma$  of HA2 to a typing context  $\Gamma^*$  of system  $F$
- 3 Map each proof  $\pi$  of a sequent  $\Gamma \vdash A$  in HA2 to a term  $\pi^*$  of system  $F$  such that the judgement  $\Gamma^* \vdash \pi^* : A^*$  is derivable
- 4 Check that each cut of  $\pi$  becomes a redex in  $\pi^*$

[Note: this works only for  $\Rightarrow$ -cuts and 2nd-order  $\forall$ -cuts. The case of 1st-order  $\forall$ -cuts is treated separately, using a combinatorial argument similar to the one we used for 2nd-kind redexes, when we proved that  $\text{SN}(F\text{-Curry})$  entails  $\text{SN}(F\text{-Church})$ ]

- 5 Conclude that cuts can be eliminated in any proof of HA2 (using any strategy)

## Cut-elimination

- $\perp \equiv \forall \alpha^0 \alpha^0$  has no cut-free proof (in the empty context)
  - ⇒ Means that a proof of  $\perp$  necessarily contains **at least one cut**
- But each cut can be individually **contracted**
  - (Keeping in mind that contracting a cut may produce several new cuts)

### Question [Takeuti]

Is there a strategy for contracting cuts in a proof such that the process converges to a cut-free proof ?

### Theorem (Cut-elimination [Girard])

Any strategy for contracting cuts converges to a cut-free proof (in a finite number of contraction steps)

### Corollary (Cut-free proofs & Consistency)

- 1 Any proposition that has a proof has also a cut-free proof
- 2 The proposition  $\perp$  has no proof in the empty context

## Translating HA2 formulæ (1/2)

- Each **predicate variable** of HA2 is mapped to a **type variable** of system  $F$  (We keep the same names for simplicity)
- Formulæ of HA2 are translated into the types of system  $F$ :

$$\begin{aligned} (\alpha^n(t_1, \dots, t_n))^* &\equiv \alpha \\ (A \Rightarrow B)^* &\equiv A^* \rightarrow B^* \\ (\forall x. B)^* &\equiv B^* \\ (\forall \alpha^n. B)^* &\equiv \forall \alpha B \end{aligned}$$

- **Remarks:**
  - arity of predicate variables is lost
  - all the first-order constructions disappear

⇒ The translation only preserves (pure) second-order constructions

- **Substitutivity:**

$$\begin{aligned} (B\{x := t\}) &\equiv A^* \\ (B\{\alpha^n := \lambda x_1, \dots, x_n. A\})^* &\equiv B^*\{\alpha := A^*\} \end{aligned}$$

## Translating HA2 formulæ (2/2)

- We can test the translation on derived formulæ:

$$(A \wedge B)^* \equiv A^* \times B^* \quad (\text{cartesian product of system } F)$$

$$(A \vee B)^* \equiv A^* + B^* \quad (\text{disjoint union})$$

$$(t = u)^* \equiv (\forall \alpha^1 \alpha^1(t) \Rightarrow \alpha^1(u))^* \equiv \forall \alpha \alpha \rightarrow \alpha \equiv \text{Unit}$$

$\Rightarrow$  Equality proofs have **no computational contents**

- Translation of contexts:** Each logical context

$$\Gamma \equiv A_1, \dots, A_n$$

is translated into a typing context of system  $F$

$$\Gamma^* \equiv \xi_1 : A_1^*, \dots, \xi_n : A_n^*$$

by associating a **term variable**  $\xi_i$  (a 'name') to each hypothesis

## Translating proofs (1/4)

**Principle:** Translate each proof  $\pi$  of a sequent  $\Gamma \vdash A$  into a term  $\pi^*$  such that  $\Gamma^* \vdash \pi^* : A^*$  is derivable

- Axiom:**

$$\left( \frac{}{\Gamma, A \vdash A} \right)^* = \xi$$

where  $\xi$  is the variable associated to the formula  $A$  in the context  $\Gamma, A$

- Introduction of the implication:**

$$\left( \frac{\begin{array}{c} \vdots \\ \pi \\ \Gamma, A \vdash B \end{array}}{\Gamma \vdash A \Rightarrow B} \right)^* = \lambda \xi : A^* . \pi^*$$

where  $\xi$  is the variable associated to  $A$  in the context  $\Gamma, A$

## Translating proofs (2/4)

- Elimination of the implication:**

$$\left( \frac{\begin{array}{c} \vdots \pi_1 \quad \vdots \pi_2 \\ \Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A \end{array}}{\Gamma \vdash B} \right)^* = \pi_1^* \pi_2^*$$

- Introduction of the 1st-order universal quantification:**

$$\left( \frac{\begin{array}{c} \vdots \pi \\ \Gamma \vdash B \end{array}}{\Gamma \vdash \forall x B} \right)^* = \pi^*$$

- Elimination of the 1st-order universal quantification:**

$$\left( \frac{\begin{array}{c} \vdots \pi \\ \Gamma \vdash \forall x B \end{array}}{\Gamma \vdash B\{x := t\}} \right)^* = \pi^*$$

**Remark:** 1st-order  $\forall$ -intro/elim are invisible in the extracted system  $F$  term

## Translating proofs (3/4)

- Introduction of the 2nd-order universal quantification:**

$$\left( \frac{\begin{array}{c} \vdots \pi \\ \Gamma \vdash B \end{array}}{\Gamma \vdash \forall \alpha^n B} \right)^* = \Lambda \alpha . \pi^*$$

- Elimination of the 2nd-order universal quantification:**

$$\left( \frac{\begin{array}{c} \vdots \pi \\ \Gamma \vdash \forall \alpha^n B \end{array}}{\Gamma \vdash B\{\alpha^n := \lambda x_1, \dots, x_n . A\}} \right)^* = \pi^* A^*$$

### Properties:

Each stage preserves the invariant  $\Gamma^* \vdash \pi^* : A^*$

- Cuts of *implication* become 1st-kind redexes
- Cuts of *2nd-order universal quantification* become 2nd-kind redexes ...
- ... but cuts of *1st-order universal quantification* **disappear**

## Translating proofs (4/4)

- **Injectivity:** Since

$$(\forall x \forall y (s(x) = s(y) \Rightarrow x = y))^* \equiv \text{Unit} \rightarrow \text{Unit}$$

it is natural to set:

$$\left( \overline{\Gamma \vdash \forall x \forall y (s(x) = s(y) \Rightarrow x = y)} \right)^* \equiv \lambda \xi : \text{Unit} . \xi$$

- **Non-surjectivity:** Quite problematic, since the type

$$(\forall x \neg s(x) = 0)^* \equiv \text{Unit} \rightarrow \perp$$

has no closed inhabitant in system  $F$ .

**Solution (hack ?):** Add a dummy constant  $\Omega : \perp$  in the system and put:

$$\left( \overline{\Gamma \vdash \forall x \neg s(x) = 0} \right)^* \equiv \lambda \xi : \text{Unit} . \Omega$$

## Natural numbers

- **Problem:** The translation of formulæ and proofs **erased all the terms!**  
 $\Rightarrow$  *Where did my numerals go ?*

- **Answer:** To benefit from induction, we restricted all the 1st-order quantifications with the predicate

$$\text{Nat}(x) \equiv \forall \alpha^1 \left( \alpha^1(0) \Rightarrow \forall y (\alpha^1(y) \Rightarrow \alpha^1(s(y))) \Rightarrow \alpha^1(x) \right)$$

whose translation in system  $F$  is:

$$(\text{Nat}(x))^* \equiv \forall \alpha (\alpha \rightarrow (\alpha \rightarrow \alpha) \rightarrow \alpha) \equiv \text{Nat} \text{ (of system } F)$$

### Fact (Translation of natural numbers)

For each term of the form  $s^n(0)$  (*concrete numeral*)

- 1 The proposition  $\text{Nat}(s^n(0))$  has exactly one cut-free proof in HA2 ...
- 2 ... whose translation in system  $F$  is precisely Church numeral  $\bar{n}$

## Cut-elimination

- 1 Each proof of (intuitionistic) second-order arithmetic has been translated into a well-typed term of system  $F$  (+ constant  $\Omega$ )

**Note:** From the point of view of normalisation, system  $F + \Omega$  is the same as system  $F$ :  $\Omega$  merely acts as a free variable that we have declared in all contexts once and for all

- 2 Via the translation of proofs:

- Cuts of **implication** become **1st kind redexes**
- Cuts of **2nd-order quantification** become **2nd kind redexes**
- cuts of **1st-order quantification** disappear

Treat the last kind of cuts as we did with 2nd-kind redexes when we proved  $\text{SN}(F\text{-Curry}) \Rightarrow \text{SN}(F\text{-Church})$ , noticing that

### Fact (Contraction of 1st-order $\forall$ cuts)

Each time we contract a cut of 1st-order quantification, the number of first-order  $\forall$ -intro **decreases** in the proof

- 3 Then we conclude that HA2 enjoys the property of **cut-elimination**

## Extracting programs from proofs

### Representation theorem

Any function whose totality can be proved in HA2 is representable in system  $F$  by a term of type  $\text{Nat} \rightarrow \text{Nat}$  [Converse is also true]

**Proof.** Consider a proof  $\pi$  in HA2 of a statement of the form

$$\forall x (\text{Nat}(x) \Rightarrow \exists y (\text{Nat}(y) \wedge P[x, y]))$$

By translating the proof  $\pi$  into system  $F$ , we obtain a term

$$\pi^* : \text{Nat} \rightarrow \forall \alpha ((\text{Nat} \times P^* \rightarrow \alpha) \rightarrow \alpha)$$

(using the 2nd-order encoding of  $\exists$  given in slide 3), so that the term

$$\lambda \xi : \text{Nat} . \pi^* \xi \text{ Nat fst} : \text{Nat} \rightarrow \text{Nat}$$

(where  $\text{fst} : \text{Nat} \times P^* \rightarrow \text{Nat}$  is the first projection) actually computes the desired function

**Remark:** We cheated a little bit, since  $\pi^*$  may contain the dummy constant  $\Omega$  that could block some computations. There are two solutions to fix this:

- 1 Use the shape of cut-free proofs of  $\text{Nat}(s^n(0))$  to show that this never happens
- 2 Define a modified translation that avoids the use of  $\Omega$  [cf Proofs and Types]