Normalisation of Second Order Arithmetic

Alexandre Miquel — PPS & U. Paris 7

Alexandre.Miquel@pps.jussieu.fr

Types Summer School 2005

August 15–26 — Göteborg

Syntax of HA2

Variables	$x,\ y, z,\ \ldots$		of individuals	(i.e. natural numbers)
	$\alpha^n,\ \beta^n,\ \gamma^n,\ \ldots$		of predicates	(for each arity $n \geq 0$)
Individuals	t, u	$::=$	$x \mid 0 \mid s(t)$	
Formulæ	A, B	$::=$	$\alpha^n(t_1, \ldots, t_n)$	(for all $n \geq 0$)
		$\mid$	$A \Rightarrow B$	
		$\mid$	$\forall x\ B$	(first-order)
		$\mid$	$\forall \alpha^n\ B$	(second order, for all $n \geq 0$)
Contexts	Γ, Δ	$::=$	$A_1, \ldots, A_n$	(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** $u\{x := t\} \quad \Rightarrow \quad$ defined in the usual way
- **First-order substitution** $B\{x := t\} \quad \Rightarrow \quad$ defined in the usual way
- **Second-order substitution** $B\{\alpha^n := \lambda x_1, \ldots, x_n \,.\, A\}$

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

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

  by the (substituted) formula

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

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

Encoding missing constructions

- Other connectives can be encoded:

$$\begin{array}{lcl} \top & \equiv & \forall\gamma^0\ (\gamma^0 \Rightarrow \gamma^0) \\ \bot & \equiv & \forall\gamma^0\ \gamma^0 \\ A \wedge B & \equiv & \forall\gamma^0\ ((A \Rightarrow B \Rightarrow \gamma^0) \Rightarrow \gamma^0) \\ A \vee B & \equiv & \forall\gamma^0\ ((A \Rightarrow \gamma^0) \Rightarrow (B \Rightarrow \gamma^0) \Rightarrow \gamma^0) \\ \neg A & \equiv & A \Rightarrow \bot \end{array}$$

- Existential quantifier (1st + 2nd order)

$$\begin{array}{lcl} \exists x\ B[x] & \equiv & \forall\gamma^0\ (\forall x\ (B[x] \Rightarrow \gamma^0)\ \Rightarrow\ \gamma^0) \\ \exists\alpha^n\ B[\alpha^n] & \equiv & \forall\gamma^0\ (\forall\alpha^n\ (B[\alpha^n] \Rightarrow \gamma^0)\ \Rightarrow\ \gamma^0) \end{array}$$

- Leibniz equality:

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

Deduction rules of HA2

- General rules for second-order intuitionistic logic:

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

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

$$\frac{\Gamma \vdash B}{\Gamma \vdash \forall x\ B}\ {\scriptstyle x\notin FV_1(\Gamma)} \qquad \frac{\Gamma \vdash \forall x\ B}{\Gamma \vdash B\{x := t\}}$$

$$\frac{\Gamma \vdash B}{\Gamma \vdash \forall \alpha^n\ B}\ {\scriptstyle \alpha^n\notin FV_2(\Gamma)} \qquad \frac{\Gamma \vdash \forall \alpha^n\ B}{\Gamma \vdash B\{\alpha := \lambda x_1, \ldots, x_n \,.\, A\}}$$

- Specific rules (axioms) for arithmetic:

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

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

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 ($\top$, $\bot$, $\wedge$, $\vee$, $\exists$ etc.) are definable and their (standard) deduction rules can be derived:

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

$$\frac{}{\Gamma \vdash \top} \qquad \frac{\Gamma \vdash \bot}{\Gamma \vdash C}$$

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

Derivable rules (2/2)

- Logical connectives: $\vee$

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

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

- Existential quantifier: 1st and 2nd-order

$$\frac{\Gamma \vdash B\{x := t\}}{\Gamma \vdash \exists x\ B} \qquad \frac{\Gamma, B \vdash C \qquad \Gamma \vdash \exists x\ B}{\Gamma \vdash C}\ {\scriptstyle x \notin FV_1(\Gamma, C)}$$

$$\frac{\Gamma \vdash B\{\alpha^n := \lambda x_1, \ldots, x_n\,.\,A\}}{\Gamma \vdash \exists \alpha^n\ B} \qquad \frac{\Gamma, B \vdash C \qquad \Gamma \vdash \exists \alpha^n\ B}{\Gamma \vdash C}\ {\scriptstyle \alpha^n \notin FV_2(\Gamma, C)}$$

Equality rules

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

- 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:

(Injectivity) $\forall x \; \forall y \; (s(x) = s(y) \;\Rightarrow\; x = y)$

(Non-surjectivity) $\forall x \; \neg \; (s(x) = 0)$

Induction principle

- Induction can be recovered via the predicate:

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

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

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

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

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

- Thanks to this trick, induction becomes provable:

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

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:**

$$\dfrac{\dfrac{\begin{array}{c}[\Gamma, A, \Gamma' \vdash A] \\ \vdots\ \pi_1 \\ \Gamma, A \vdash B\end{array}}{\Gamma \vdash A \Rightarrow B} \qquad \begin{array}{c}\vdots\ \pi_2 \\ \Gamma \vdash A\end{array}}{\Gamma \vdash B} \quad \rightsquigarrow \quad \begin{array}{c}\vdots\ \pi_2 \\ \Gamma, \Gamma' \vdash A \\ \vdots\ \pi_1 \\ \Gamma \vdash B\end{array}$$

Here, $[\Gamma, A, \Gamma' \vdash A]$ represents all the instances of an axiom with the formula A in the proof π_1. (Such instances may occur in extended contexts of the form Γ, A, Γ'.)
These instances are then used as placeholders that are filled by the proof π_2 during the contraction of the cut (after some weakenings due to the presence of extra contexts Γ')

The notion of cut (2/2)

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

$$\dfrac{\dfrac{\begin{array}{c}\vdots\ \pi\\ \Gamma \vdash B\end{array}}{\Gamma \vdash \forall x\,.\,B}}{\Gamma \vdash B\{x := t\}} \quad \rightsquigarrow \quad \begin{array}{c}\vdots\ \pi\{x{:=}t\}\\ \Gamma \vdash B\{x := t\}\end{array}$$

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

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

$$\dfrac{\dfrac{\begin{array}{c}\vdots\ \pi\\ \Gamma \vdash B\end{array}}{\Gamma \vdash \forall \alpha^n\,.\,B}}{\Gamma \vdash B\{\alpha^n := \lambda x_1, \ldots, x_n\,.\,A\}} \quad \rightsquigarrow \quad \begin{array}{c}\vdots\ \pi\{\alpha^n{:=}\cdots\}\\ \Gamma \vdash B\{\alpha^n := \lambda x_1, \ldots, x_n\,.\,A\}\end{array}$$

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

Derived cuts

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

- **Cuts of the conjunction**:

$$\dfrac{\dfrac{\overset{\vdots\ \pi_1}{\Gamma \vdash A} \qquad \overset{\vdots\ \pi_2}{\Gamma \vdash B}}{\Gamma \vdash A \wedge B}}{\Gamma \vdash A} \quad \rightsquigarrow \quad \overset{\vdots\ \pi_1}{\Gamma \vdash A}$$

(+ symmetric cut with $\wedge$-elim$_2$)

- **Cuts of the disjunction**:

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

(+ symmetric cut with $\vee$-intro$_2$)

Filling placeholders in π_1 with π 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

$\Rightarrow$ Cut-free proofs have a simpler structure that make them easier to analyse

Fact (Cut-free consistency)

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

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

$$\dfrac{\dfrac{\dfrac{\overline{\vdash \forall x \, \forall y \; (s(x) = s(y) \Rightarrow x = y)}}{\vdash \forall y \; (s(t) = s(y) \Rightarrow t = y)}}{\vdash s(t) = s(t) \Rightarrow t = t} \qquad \begin{array}{c}(\text{cut-free}) \\ \vdots \\ \vdash s(t) = s(t)\end{array}}{\vdash t = t}$$

$\Rightarrow$ Reasoning on cut-free proofs is purely combinatorial

Cut-elimination

- $\bot \;\equiv\; \forall\alpha^0\;\alpha^0$ has no cut-free proof (in the empty context)
  $\Rightarrow$ Means that a proof of $\bot$ 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 $\bot$ has no proof in the empty context*

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 Γ of HA2 to a typing context Γ^* of system F
3. Map each proof π of a sequent $\Gamma \vdash A$ in HA2 to a term π^* of system F such that the judgement $\Gamma^* \vdash \pi^* : A^*$ is derivable
4. Check that each cut of π becomes a redex in π^*

   [**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 SN(F-Curry) entails SN(F-Church)]

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

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:

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

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

$\Rightarrow$ The translation only preserves (pure) second-order constructions

- **Substitutivity:**

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

Translating HA2 formulæ (2/2)

- We can test the translation on derived formulæ:

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

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

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

$\Rightarrow$ Equality proofs have no computational contents

- **Translation of contexts:** Each logical context

$$\Gamma \quad \equiv \quad A_1,\ \ldots,\ A_n$$

is translated into a typing context of system F

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

by associating a term variable ξ_i (a 'name') to each hypothesis

Translating proofs (1/4)

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

- **Axiom:**

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

where ξ is the variable associated to the formula A in the context Γ, A

- **Introduction of the implication:**

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

where ξ is the variable associated to A in the context Γ, A

Translating proofs (2/4)

- **Elimination of the implication:**

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

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

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

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

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

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:**

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

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

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

Properties:

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

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

Translating proofs (4/4)

- **Injectivity:** Since

$$\big(\forall x\ \forall y\ (s(x) = s(y)\ \Rightarrow\ x = y)\big)^* \quad \equiv \quad \mathrm{Unit} \to \mathrm{Unit}$$

it is natural to set:

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

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

$$(\forall x\ \neg\ s(x) = 0)^* \quad \equiv \quad \mathrm{Unit} \to \bot$$

has no closed inhabitant in system F.

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

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

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

   Note: From the point of view of normalisation, system $F + \Omega$ is the same as system F: Ω 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 SN(F-Curry) $\Rightarrow$ SN(F-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

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

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

whose translation in system F is:

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

Fact (Translation of natural numbers)

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

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

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 $\mathsf{Nat} \to \mathsf{Nat}$ *[Converse is also true]*

Proof. Consider a proof π in HA2 of a statement of the form

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

By translating the proof π into system F, we obtain a term

$$\pi^* \quad : \quad \mathsf{Nat} \to \forall \alpha\ ((\mathsf{Nat} \times P^* \to \alpha) \to \alpha)$$

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

$$\lambda \xi : \mathsf{Nat}\,.\,\pi^*\ \xi\ \mathsf{Nat}\ \mathsf{fst} \quad : \quad \mathsf{Nat} \to \mathsf{Nat}$$

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

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

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