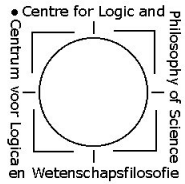
Practical Reasoning with Proofs and Types

Giuseppe Primiero

FWO - Flemish Research Foundation
Centre for Logic and Philosophy of Science, Ghent University
IEG - Oxford University



Giuseppe.Primiero@UGent.be
http://www.philosophy.ugent.be/giuseppeprimiero/

Isaac Newton Institute for Mathematical Sciences, Cambridge – 09 February 2012

Outline

Turing's Practical Type Theory

Nested-Type System in [Turing, 1948]: a theory of types with small use of type themselves, in a way that reflects the practice of proving by mathematicians.

Turing's Practical Type Theory

Nested-Type System in [Turing, 1948]: a theory of types with small use of type themselves, in a way that reflects the practice of proving by mathematicians.

- Hierarchy of types: type $n+1$ is the type of functions from type n to type n (construed from below):
  - individuals (type 0): $U_1, \ldots, U_n$
  - functions (type 1): taking arguments from $U_1, \ldots, U_n$ and returning them as values
  - ...
- Introduce an individual as the value of undefined functions (to prevent bad-typing);
- Introduce "Interpretability under hypotheses": hypothesis "x is of type A", satisfied by construction and substitution of the free variable.

What do we mean by Practical Reasoning

- Focus on the use of hypothetical judgements;
- Interpret partially and fully evaluated expressions;
- Apply this to reasoning with valid and true (global and local) assumptions.

What do we mean by Proofs and Types

(Extensions of the) Provability and Realizability models intended by the BHK semantics:

What do we mean by Proofs and Types

(Extensions of the) Provability and Realizability models intended by the BHK semantics:

- modal type theories to express: partial termination and distributed computing;
- The current work (at INI): *LP* with a notion of dependent justification.

ILP

- LP provides an explicit reading of modal logic $S4$ with an intended provability semantics for the propositional intuitionistic logic IPC;
- knowledge and belief modalities are decrypted as justification terms;
- justifications (e.g. formal proofs) are abstract objects which have structure and operations on them;
  - basic operations: application (for implication) and sum (for adding proofs to proofs);
  - in our minimal setting: application and proof checking;

(Standard) Intuitionistic Logic of Proofs

Definition (Language)

We denote with *ILP* a language that contains:

- a countable set of symbols $A, B, \ldots$ for propositions;
- individual variables $[\![x]\!], [\![y]\!], \ldots$ and
- constants $a, b, \ldots$ for proof terms;
- predicative expressions $A(x)$ where x is a bounded variable;
- functional symbols for operations on proof terms: $\cdot$, !.

Axioms and Inference Schemes [Artëmov and Bonelli, 2007]

Definition (Axioms)

Axioms of the system are:

A0. Axioms schemes of minimal logic in the the language of LP

A1. $[\![s]\!]A \supset A$ (Unconditional Evidence)

A2. $[\![s]\!]A \supset [\![!s]\!][\![s]\!]A$ (Proof Checker)

A3. $[\![s]\!](A \supset B) \supset ([\![t]\!]A \supset [\![s \cdot t]\!]B)$ (Application)

R1. $\Gamma \vdash A \supset B$ and $\Gamma \vdash A$ implies $\Gamma \vdash B$ (Modus Ponens)

R2. If **A** is an axiom **A0**. – **A3**. and c is a proof constant, then $\vdash [\![c]\!]\mathbf{A}$ (Necessitation)

Dependent terms in *ILP*

- The notion of dependent term in *LP* is inspired by its formal counterpart in theories of dependent types:

Dependent terms in *ILP*

- The notion of dependent term in *LP* is inspired by its formal counterpart in theories of dependent types:
  - A dependent type is a type expression of the form $B[x]$ with x a free variable ranging over A *type* saying that B is a type whenever $x \in A$;
  - Propositional functions under the props-as-types analogy;
  - Σ type: type of all pairs $\langle a, b \rangle$ where $a \in A$ and $b \in B[a]$;
  - Π type: type of all functions $\lambda x.b[x]$ where $b[a] \in B[a]$ for any $a \in A$.

Tasks

1. Give a notion of dependent proof term in (Intuitionistic) Logic of Proofs for expressions of the form

   "t is a proof term for B, whenever A has a proof term s"

Tasks

1. Give a notion of dependent proof term in (Intuitionistic) Logic of Proofs for expressions of the form

   "t is a proof term for B, whenever A has a proof term s"

2. Interpret the previous sentence with two distinct readings:
   - *A* actually justified/valid
   - *A* possibly justified/assumed true

Tasks

1. Give a notion of dependent proof term in (Intuitionistic) Logic of Proofs for expressions of the form

   "t is a proof term for B, whenever A has a proof term s"

2. Interpret the previous sentence with two distinct readings:
   - *A* actually justified/valid
   - *A* possibly justified/assumed true

3. Translate to derivability in a ND calculus and prove some metatheoretical results: equality rules, substitution lemmas, contractions on connectives, normalization.

Logic of Proofs with Dependent Terms

Definition (Proof Terms)

In ILP_{dep} each proof variable or proof constant is a proof term:

- we denote the fact that s is the proof term of proposition A by the formula $[\![s]\!]A$;
- we denote the fact that t is the proof term of proposition B whenever s is the proof term of proposition A by the formula $\ll s \gg [\![t]\!]B[A]$
- if $[\![s]\!]$ and $[\![t]\!]$ are proof terms, so are: $[\![s \cdot t]\!]$, $[\![!s]\!][\![s]\!]$, $[\![s]\!] \cdot [\![t]\!]$, $[\![(s)t]\!]$;
- it allows multiple dependencies: $\ll s_1 \ldots s_n \gg [\![t]\!]B[A_1 \ldots A_n]$;
- we can add quantification over proof terms: $\forall [\![x]\!]A.B(x)$ and $\exists [\![x]\!]A.B(x)$;

Additional Inference Schemes

Definition (Axioms)

Additional rule schemes of the system are:

R3. If $[\![s]\!]A$ and $A \vdash [\![t]\!]B$, then $\vdash \ll s \gg [\![t]\!]B[A]$ (Dependent Evidence)

R4. If $\ll s \gg [\![t]\!]B[A]$ and $[\![s]\!]A$, then $[\![s]\!] \cdot [\![t]\!]B$ (Application for Dependent Evidence)

Functions in ND

- Derivability of a term under valid assumptions (global validity) defines Unconditional Evidence;

$$\Delta; \cdot \vdash A \mid s \qquad \text{UnEvid}$$

- Derivability of a term under true assumptions (local validity) defines Dependent Evidence;

$$\Delta; \Gamma \vdash A \mid\mid s \qquad \text{DepEvid}$$

Definition (Language)

The syntax is defined by the following alphabet:

Proof Terms
$s := x \mid s \cdot s \mid !s \mid XTRT\ s\ AS\ v:A\ IN\ s \mid ?s \mid ASSM\ s\ AS\ a:A\ INs$

Propositions
$A := P \mid A \supset B \mid B[A] \mid [\![s]\!]A \mid \ll s \gg A \mid \ll s \gg [\![t]\!]B[A]$

Truth Contexts $\Gamma := \cdot \mid \Gamma, a:A$

Validity Contexts $\Delta := \cdot \mid \Delta, v:A$

Definition (The Logic $LP_{nd\diamond}$)

$LP_{nd\diamond}$ is defined by the following schemes:

$$\frac{}{\Delta; v:A, \Delta' \vdash A \mid v}\ \mathit{ValVar}$$

$$\frac{\Delta, v:A; \cdot \vdash B \mid s}{\Delta; \Gamma \vdash A \supset B \mid \lambda v:A.s}\ \supset I \qquad \frac{\Delta; \cdot \vdash A \supset B \mid s \qquad \Delta; \cdot \vdash A \mid t}{\Delta; \Gamma \vdash B \mid s \cdot t}\ \supset E$$

$$\frac{}{\Delta; a:A; \cdot \vdash A \,||\, a}\ \mathit{TruVar}$$

$$\frac{\Delta; a:A \vdash B \,||\, t}{\Delta; \cdot \vdash \ll s \gg [\![t]\!]B[A]}\ \mathit{DepEvidence\ Formation}$$

$$\frac{\Delta; \Gamma \vdash \ll s \gg [\![t]\!]B[A] \qquad \Delta; \Gamma \vdash [\![s]\!]A \mid !s}{\Delta; \Gamma \vdash B \mid !([\![s]\!] \cdot [\![t]\!])}\ \mathit{DepEvidence\ Application}$$

Now modalities can be used to internalise dependencies:

Definition

$$\frac{\Delta;\cdot \vdash A \mid s}{\Delta;\Gamma \vdash [\![s]\!]A \mid !s}\ \Box I \qquad \frac{\Delta;\cdot \vdash [\![r]\!]A \mid !s \qquad \Delta, v:A \vdash C \mid t}{\Delta;\Gamma \vdash C_r^v \mid XTRT\ s\ AS\ v:A\ IN\ t}\ \Box E$$

$$\frac{\Delta;\Gamma \vdash A \mid\mid s}{\Delta;\Gamma;\cdot \vdash \ll s \gg A \mid ?s}\ \Diamond I$$

$$\frac{\Delta;\Gamma \vdash \ll r \gg A \mid ?s \qquad \Delta, a:A;\cdot \vdash C \mid\mid t}{\Delta;\Gamma;\cdot \vdash C_r^a \mid\mid ASSM\ s\ AS\ a:A\ IN\ t}\ \Diamond E$$

What the system satisfies

Properties

The system satisfies:

- *structural properties for unconditional and dependent evidence (restricted Exchange)*
- *substitution on terms*
- *context equivalence*
- *reflexivity on unconditional and dependent evidence*
- *symmetry on unconditional and dependent evidence*
- *transitivity on unconditional and dependent evidence*
- *equivalence on λ-terms and application for implication*
- *equivalence on β/η redexes for $\Box$, $\Diamond$*
- *equivalence on Introduction/Elimination Rules for $\Box$, $\Diamond$*
- *equivalence on Functional Terms and Application*

What is the problem with Normalization?

(Weak and Strong) Normalisation require detour imposed by the newly added dependent evidence, as $\beta\eta$ equivalent redexes might not reduce to each other.

$$\begin{array}{lll} \Delta;\cdot \vdash \ll s \gg [\![t]\!] B[A] & \Rightarrow_{Eq\beta} & \Delta;\Gamma \vdash B \mid s_t^v \equiv s \cdot t \\ \Downarrow_{Eq\eta} & & \Uparrow \\ \Delta;\Gamma \vdash A \supset B \mid s \cdot t & \Rightarrow_{Eq\Box\beta} & \Delta;\Gamma \vdash B_s^v \mid t_s^v \equiv XTRT\ !s\ AS\ v : A\ IN\ t \end{array}$$

What is the problem with Normalization?

(Weak and Strong) Normalisation require detour imposed by the newly added dependent evidence, as $\beta\eta$ equivalent redexes might not reduce to each other.

$$\begin{array}{lll} \Delta;\cdot \vdash \ll s \gg [\![t]\!] B[A] & \Rightarrow_{Eq\beta} & \Delta;\Gamma \vdash B \mid s_t^v \equiv s \cdot t \\ \Downarrow_{Eq\eta} & & \Uparrow \\ \Delta;\Gamma \vdash A \supset B \mid s \cdot t & \Rightarrow_{Eq\Box\beta} & \Delta;\Gamma \vdash B_s^v \mid t_s^v \equiv XTRT\ !s\ AS\ v : A\ IN\ t \end{array}$$

$$\begin{array}{lll} \Delta;\cdot \vdash \ll s \gg [\![t]\!] B[A] & \Rightarrow_{Eq\beta} & \Delta;\Gamma \vdash B \mid s_t^v \equiv s \cdot t \\ \Downarrow_{Eq\eta} & & \\ \Delta;\Gamma \vdash A \supset B \mid s \cdot t & \Rightarrow_{Eq\Diamond\beta} & \Delta;\Gamma;\cdot \vdash B_s^a \mid t_s^a \equiv ASSM\ ?s\ AS\ a : A\ IN\ t \end{array}$$

A strategy of two Normal Forms ([Abel et al., 2007])

Definition (Predicates *INF* and *FNF*)

The normal form predicates *INF* and *FNF* are defined according to the following schemes:

$$\frac{\Delta;\cdot \vdash A \mid s}{\Delta;\Gamma \vdash FNF(s)} \qquad \frac{\Gamma \vdash A \mid\mid s}{\Gamma;\cdot \vdash INF(s)}$$

$$\frac{\Delta;\cdot \vdash FNF(A) \qquad \Delta;a:A \vdash FNF(t)}{\Delta;\Gamma;\cdot \vdash FNF([a/v]\cdot t)}$$

$$\frac{\Gamma;\cdot \vdash INF(A) \qquad \Delta;a:A \vdash FNF(t)}{\Delta;\Gamma;\cdot \vdash INF(t[a:A])}$$

TO DO

1. Define η-expansion rewriting rules for *INF*/*FNF* predicates;
2. show that every INF/FNF reduction ends in a β normal redux;
3. equivalence preserves beta reduction.

TO PROVE

Lemma (Normalisation)

If $\Delta; \Gamma \vdash FNF(t)$*, then* t *is in normal form.*

TO PROVE

Lemma (Normalisation)

If $\Delta; \Gamma \vdash FNF(t)$*, then* t *is in normal form.*

Lemma (Confluence)

1. $\rightarrow_{INF/FNF}$*-normal forms are unique;*
2. *confluence: every term reduces to a normal formal;*
3. *reductions on* $\rightarrow_{\eta INF/FNF}$ *preserve* β*-normal forms;*
4. *normalisation of* $\rightarrow_{INF/FNF}$ *is reduced to normalisation of* $\rightarrow_{\eta INF/FNF}$*.*

TO PROVE

Lemma (Strong Normalization)

There are no infinite sequences of reductions
$\Delta; \Gamma \vdash t \rightarrow_{\eta INF/FNF} t' \rightarrow_{\eta INF/FNF} t'' \ldots$

Summary

- We introduced functional expressions over evidences as in *LP*;
- Defined a natural deduction calculus which distinguishes between unconditional and dependent evidence;
- Extended it to extensional equivalence;
- MAIN TASK: prove that this extension is conservative w.r.t. the calculus with simple evidence from [Artëmov and Bonelli, 2007] by showing (Strong) Normalization.

References I


Abel, A., Aehlig, K., and Dybjer, P. (2007).
Normalization by evaluation for Martin-Löf type theory with one universe.
In Fiore, M., editor, *Proceedings of the 23rd Conference on the Mathematical Foundations of Programming Semantics (MFPS XXIII), New Orleans, LA, USA, 11-14 April 2007*, volume 173 of *Electronic Notes in Theoretical Computer Science*, pages 17–39. Elsevier.


Artëmov, S. N. and Bonelli, E. (2007).
The intensional lambda calculus.
In Artëmov, S. N. and Nerode, A., editors, *LFCS*, volume 4514 of *Lecture Notes in Computer Science*, pages 12–25. Springer.


Turing, A. (1948).
Practical forms of type theory.
Journal of Symbolic Logic, 13:80–94.