

On the Logical Structure of OMDoc

Palaiseau, 14th December 2004

Claudio Sacerdoti Coen*

<sacerdot@lix.polytechnique.fr>

* Project PCRI, CNRS, École Polytechnique, INRIA, Université Paris-Sud.

Overview

1. Motivations
2. A gentle introduction to the $\bar{\lambda}\mu\tilde{\mu}$ -calculus
3. Natural language rendering of intuitionistic $\lambda\mu$ -normal $\bar{\lambda}\mu\tilde{\mu}$ -terms
4. Encoding the intuitionistic $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus in OMDoc
5. Encoding OMDoc in the intuitionistic $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus

OMDoc before MoWGLI

The original OMDoc module for proofs

- designed for sequent calculus proofs in Hilbert style
- plus pointers to form a dag structure and share formulae
- an “open format” according to the Kohlhase’s philosophy
- not well-suited for natural deduction proofs
- a flat markup that does not exploit XML nesting for scoping, telescoping rendering, etc.
- no distinction between top-down and bottom-up proof steps, no local definitions
- too general to provide generic operations (e.g. rendering of both ND and SC proofs or both SC classical and SC intuitionistic

proofs)

An Hot Thread (1/2)

- George Gogvadze starts the “proof structure” thread on the MoWGLI m.l. the 19th July 2002
- I provide a list of possible proof representations: ND, SC, ND via Curry-Howard, etc.
- Hugo lists SC via Curry-Howard, but
 - “...but it is probably less useful for rendering”
 - he focus on finding a tree representation of proofs where the context is implicit
 - “If I can show that B is contradictory and know A then I know that $A \Rightarrow B$ is contradictory. Is this natural?”
 - “for both ND and SC ... a λ -notation à la Church is the more appropriate since it carrier **the minimal information** needed to

reconstruct any of the other representations”

An Hot Thread (2/2)

- Andrea summarizes
 - Michael: let us forget about the λ -like notation and let us focus on the tree-like encoding
 - Bologna: we cannot neglect the λ -like notation; it cannot be merged with the tree-like encoding
 - Hugo: let us forget about the tree-like encoding and let us develop a generic, sufficiently general markup for the λ -like notation
- A few user contributions along the line “the DTD is good if we can do things over the documents” (operational approach)
- Andrea: “I give up. If the DTD “leaks” we shall use this to simplify our job”

The End of the Story

- We developed a DTD that is as generic as possible
- We map Coq λ -terms only to a subset of OMDoc
- We provide a rough rendering semantics to the subset that extends somehow to the whole language
- We provide an encoding of CIC terms in OMDoc along the lines of Yann Coscoy's thesis

The Sequel

I tried to use OMDoc for the proof objects of Matita but

- the language (even the restricted language) is too big; implementing even simple operations is time consuming and error prone
- no logical meaning \Rightarrow need for a forgetting map from OMDoc to CIC to apply tactics etc.
- extra-logical operations have type $\text{OMDoc} \Rightarrow \text{OMDoc}$ (e.g. changing the rendering)
- but $\text{OMDoc} \Rightarrow \text{CIC} \Rightarrow \text{OMDoc}'$ is not the identity function; thus applying a tactic undoes all the logical operations

The Big Questions

What is the logical structure of the restricted OMDoc syntax?

- a **subsyntax** of a λ -calculus (e.g. no β -redexes)
- extended with `Let ...Ins` for forward-reasoning steps
- extended with `Casts` to type **most** of the sub-expressions
- extended with a dynamic variable `previous` to refer to the last bound variable and define chains of reasoning

Several similarities to the $\bar{\lambda}\mu\tilde{\mu}$ -calculus

- is there a subsyntax of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus with a reasonable rendering semantics?
- are OMDoc and the $\bar{\lambda}\mu\tilde{\mu}$ -calculus equivalent?
- what is the relation between the syntactic restrictions over

OMDoc and the $\bar{\lambda}\mu\tilde{\mu}$ -calculus?

Overview

1. Motivations
2. A gentle introduction to the $\bar{\lambda}\mu\tilde{\mu}$ -calculus
3. Natural language rendering of intuitionistic $\lambda\mu$ -normal $\bar{\lambda}\mu\tilde{\mu}$ -terms
4. Encoding the intuitionistic $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus in OMDoc
5. Encoding OMDoc in the intuitionistic $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus

Why the $\bar{\lambda}\mu\tilde{\mu}$ -calculus?

- $\bar{\lambda}$ Bottom-up vs top-down in SC and ND \Rightarrow we look for a λ -calculus for SC
- $\tilde{\mu}$ Bottom-up/top-down vs call-by-value/call-by-name \Rightarrow we look for the symmetries of computation
- μ Inner types, CPS and classical logic \Rightarrow we look for a sub-calculus of a calculus for classical logic

Moreover, the $\bar{\lambda}\mu\tilde{\mu}$ -calculus is very small (and nice).

Commands, contexts and terms

Command $c := (\text{print } (\lambda x.\lambda y.y M N))$

Commands, contexts and terms

Command $c := (\text{print } (\lambda x.\lambda y.y M N))$

Term $v := \lambda x.\lambda y.y$

Context $E := (\text{print } (\square M N))$

Commands, contexts and terms

Command $c := (\text{print } (\lambda x.\lambda y.y M N))$

Term $v := \lambda x.\lambda y.y$

Context $E := (\text{print } (\square M N))$
 $= M \circ N \circ \text{print}$

Commands, contexts and terms

Command $c := (\text{print } (\lambda x.\lambda y.y M N))$

Term $v := \lambda x.\lambda y.y$

Context $E := (\text{print } (\square M N))$
 $= M \circ N \circ \text{print}$

Term $v ::= \lambda x.v \mid x$

Environment $E ::= v \circ E \mid \alpha$

Command $c ::= \langle v \parallel E \rangle$

Commands, contexts and terms

Term $v ::= \lambda x. \lambda y. y$

Context $E ::= M \circ N \circ \text{print}$

Context $\text{print} ::= \tilde{\mu}z. \langle \text{format} || z \circ \text{raw_print} \rangle$

Term $v ::= \lambda x. v \mid x$

Environment $E ::= v \circ E \mid \alpha \mid \tilde{\mu}x. c$

Command $c ::= \langle v || E \rangle$

Commands, contexts and terms

Term $v := \lambda x.\lambda y.y$

Context $E := M \circ N \circ \text{print}$

Context $\text{print} := \tilde{\mu}z.\langle \text{format} || z \circ \text{raw_print} \rangle$

Term $\text{format} := \lambda x.\mu\alpha.\langle \text{justify} || x \circ 15\text{cm} \circ \alpha \rangle$

Term $v ::= \lambda x.v \mid x \mid \mu\alpha.c$

Environment $E ::= v \circ E \mid \alpha \mid \tilde{\mu}x.c$

Command $c ::= \langle v || E \rangle$

Reduction rules

$\langle \lambda x. \lambda y. y \mid M \circ N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

Reduction rules

$\langle \lambda x. \lambda y. y \mid M \circ N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle \lambda y. y \mid N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

Reduction rules

$\langle \lambda x. \lambda y. y \mid M \circ N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle \lambda y. y \mid N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle N \mid \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

Reduction rules

$\langle \lambda x. \lambda y. y \mid M \circ N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle \lambda y. y \mid N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle N \mid \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid N \circ \text{raw_print} \rangle \triangleright$

Reduction rules

$\langle \lambda x. \lambda y. y \mid M \circ N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle \lambda y. y \mid N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle N \mid \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid N \circ \text{raw_print} \rangle \triangleright$

$\langle \mu \alpha. \langle \text{justify} \mid N \circ 15\text{cm} \circ \alpha \rangle \mid \text{raw_print} \rangle \triangleright$

Reduction rules

$\langle \lambda x. \lambda y. y \mid M \circ N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle \lambda y. y \mid N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle N \mid \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid N \circ \text{raw_print} \rangle \triangleright$

$\langle \mu \alpha. \langle \text{justify} \mid N \circ 15\text{cm} \circ \alpha \rangle \mid \text{raw_print} \rangle \triangleright$

$\langle \text{justify} \mid N \circ 15\text{cm} \circ \text{raw_print} \rangle$

Reduction rules

$\langle \lambda x. \lambda y. y \mid M \circ N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle \lambda y. y \mid N \circ \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle N \mid \tilde{\mu} z. \langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid z \circ \text{raw_print} \rangle \rangle \triangleright$

$\langle \lambda x. \mu \alpha. \langle \text{justify} \mid x \circ 15\text{cm} \circ \alpha \rangle \mid N \circ \text{raw_print} \rangle \triangleright$

$\langle \mu \alpha. \langle \text{justify} \mid N \circ 15\text{cm} \circ \alpha \rangle \mid \text{raw_print} \rangle \triangleright$

$\langle \text{justify} \mid N \circ 15\text{cm} \circ \text{raw_print} \rangle$

$\langle \lambda x. v_1 \mid v_2 \circ E \rangle \triangleright \langle v_1 \{v_2/x\} \mid E \rangle$

$\langle \mu \alpha. c \mid E \rangle \triangleright c \{E/\alpha\}$

$\langle v \mid \tilde{\mu} x. c \rangle \triangleright c \{v/x\}$

The Type-Free Calculus

Term $v ::= \lambda x.v \mid x \mid \mu\alpha.c$

Environment $E ::= v \circ E \mid \alpha \mid \tilde{\mu}x.c$

Command $c ::= \langle v \parallel E \rangle$

$\langle \mu\alpha.c \parallel E \rangle \triangleright c\{E/\alpha\}$

$\langle v \parallel \tilde{\mu}x.c \rangle \triangleright c\{v/x\}$

$\langle \lambda x.v_1 \parallel v_2 \circ E \rangle \triangleright \langle v_2 \parallel \tilde{\mu}x.v_1 \circ E \rangle$

The Type-Free Calculus

Term $v ::= \lambda x.v \mid x \mid \mu\alpha.c$

Environment $E ::= v \circ E \mid \alpha \mid \tilde{\mu}x.c$

Command $c ::= \langle v \parallel E \rangle$

$\langle \mu\alpha.c \parallel E \rangle \triangleright c\{E/\alpha\}$

$\langle v \parallel \tilde{\mu}x.c \rangle \triangleright c\{v/x\}$

$\langle \lambda x.v_1 \parallel v_2 \circ E \rangle \triangleright \langle v_2 \parallel \tilde{\mu}x.v_1 \circ E \rangle$

$\langle \mu\alpha.\langle f \parallel 2 \circ \alpha \rangle \parallel \tilde{\mu}x.\langle g \parallel x \circ \text{print} \rangle \rangle$
 $(\lambda x.\text{print}(g\ x)\ (f\ 2))$

$\langle f \parallel 2 \circ \tilde{\mu}x.\langle g \parallel x \circ \text{print} \rangle \rangle$

`let $x := f\ 2$ in $\text{print}(g\ x)$`

$\langle g \parallel \mu\alpha.\langle f \parallel 2 \circ \alpha \rangle \circ \text{print} \rangle$

`$\text{print}(g(f\ 2))$`

The Simply Typed Calculus

Term $v ::= \lambda x : T.v \mid x \mid \mu\alpha : T.c$

Environment $E ::= v \circ E \mid \alpha \mid \tilde{\mu}x : T.c$

Command $c ::= \langle v \parallel E \rangle$

$$\frac{\Gamma \vdash v : T \mid \Delta \quad \Gamma \mid E : T \vdash \Delta}{\langle v \parallel E \rangle : (\Gamma \vdash \Delta)} \quad \frac{}{\Gamma ; x : T \vdash x : T \mid \Delta} \quad \frac{}{\Gamma \mid \alpha : T \vdash \alpha : T ; \Delta}$$

$$\frac{\Gamma x : T \vdash v : T' \mid \Delta}{\Gamma \vdash \lambda x : T.v : T \rightarrow T' \mid \Delta} \quad \frac{\Gamma \vdash v : T \mid \Delta \quad \Gamma \mid E : T' \vdash \Delta}{\Gamma \mid v \circ E : T \rightarrow T' \vdash \Delta}$$

$$\frac{c : (\Gamma \vdash \alpha : T ; \Delta)}{\Gamma \vdash \mu\alpha : T.c : T \mid \Delta} \quad \frac{c : (\Gamma ; x : T \vdash \Delta)}{\Gamma \mid \tilde{\mu}x : T.c : T \vdash \Delta}$$

The intuitionistic case just says that binding a continuation α in a command c (i.e. $\mu\alpha.T.c$) clears the previously bounded continuation \Rightarrow no backtracking, the computation becomes deterministic, **well-nesting of computations.**

Overview

1. Motivations
2. A gentle introduction to the $\bar{\lambda}\mu\tilde{\mu}$ -calculus
3. Natural language rendering of intuitionistic $\lambda\mu$ -normal $\bar{\lambda}\mu\tilde{\mu}$ -terms
4. Encoding the intuitionistic $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus in OMDoc
5. Encoding OMDoc in the intuitionistic $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus

A $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus

<i>Term</i>	$w ::= x \mid \nu$	$\nu ::= \lambda x : T.\nu \mid \mu\alpha : T.c$
<i>Environment</i>	$E ::= w \circ E \mid \alpha \mid \tilde{\mu}x : T.c$	
<i>Command</i>	$c ::= \langle \lambda x : T.w \mid \alpha \rangle$ $\mid \langle x \mid w \circ E \rangle$ $\mid \langle x \mid \alpha \rangle$ $\mid \langle \mu\alpha : T.c \mid \tilde{\mu}x : T.c' \rangle$	
<i>Pruned commands</i>	$c ::= \langle \lambda x : T.w \mid w \circ E \rangle$ $\mid \langle \lambda x : T.w \mid \tilde{\mu}x : T'.c \rangle$ $\mid \langle \mu\alpha : T.w \mid w \circ E \rangle$ $\mid \langle x \mid \tilde{\mu}y : T.c \rangle$ $\mid \langle \mu\alpha : T.w \mid \beta \rangle$	β -redex $\tilde{\mu}$ -redex μ -redex $\tilde{\mu}$ -redex (renaming) μ -redex (renaming)

Producing terms in the fragment

μ -expansion: $v \Rightarrow \mu\alpha : T.\langle v || \alpha \rangle$

$\tilde{\mu}$ -expansion: $E \Rightarrow \tilde{\mu}x : T.\langle x || E \rangle$

Pruned commands:

$\langle x || \tilde{\mu}y : T.c \rangle \Rightarrow c\{x/y\}$

$\langle \mu\alpha : T.w || \beta \rangle \Rightarrow w\{\beta/\alpha\}$

$\langle \lambda x : T.w || \tilde{\mu}x : T'.c \rangle \Rightarrow \langle \mu\alpha : T'.\langle \lambda x : T.w || \alpha \rangle || \tilde{\mu}x : T'.c \rangle$

$\langle \mu\alpha : T.w || w \circ E \rangle \Rightarrow \langle \mu\alpha : T.w || \tilde{\mu}x : T.\langle x || w \circ E \rangle \rangle$

$\langle \lambda x : T.w || w \circ E \rangle \Rightarrow \langle \mu\alpha : T'.\langle \lambda x : T.w || \alpha \rangle || \tilde{\mu}y : T'.\langle y || w \circ E \rangle \rangle$

or $\langle w\{w/x\} || E \rangle$

A term $\lambda x_1 : T_1 \dots \lambda x_n : T_n.x$ can be easily μ -expanded to $\lambda x_1 : T_1 \dots \lambda x_n : T_n.\mu\alpha : T.\langle x || \alpha \rangle$

Terms in β -normal form can be easily (i.e. syntactically, no substitution, no type-checking) translated into the fragment.

Natural language rendering of intuitionistic $\lambda\mu$ -normal $\bar{\lambda}\mu\tilde{\mu}$ -terms

<i>Terms :</i>	$[[\lambda x : T.v]]^d$	=	suppose $T(x)$; $[[v]]^d$
	$[[x]]^T$	=	consider x
	$[[x]]^<$	=	by hypothesis x
	$[[\mu\alpha : T.c]]^<$	=	we need to prove T ; $[[c]]^<$
	$[[\mu\alpha : T.c]]_*^<$	=	we proceed by proving T ; $[[c]]^<$
	$[[\mu\alpha : T.c]]^>$	=	$[[c]]^T$
<i>Environments :</i>	$[[\alpha]]^<$	=	done
	$[[\alpha]]^T$	=	we proved T
	$[[\tilde{\mu}x : T.c]]^d$	=	we proved $T(x)$; $[[c]]^d$
<i>Commands :</i>	$[[\langle \lambda x : T.v \mid \alpha \rangle]]^d$	=	$[[\lambda x : T.v]]^>$; $[[\alpha]]^d$
	$[[\langle x \mid w_1 \circ \dots \circ w_n \circ E \rangle]]^d$	=	by x $x_{i_1} \dots x_{i_m}$; $[[\nu_{j_1}]]^<$; \dots $[[\nu_{j_l}]]^<$; $[[E]]^d$
	$[[\langle x \mid \alpha \rangle]]^d$	=	$[[x]]^d$; $[[\alpha]]^d$
	$[[\langle \mu\alpha : T.c \mid \tilde{\mu}x : T.c' \rangle]]^d$	=	$[[\mu\alpha : T.c]]_*^<$; $[[\tilde{\mu}x : T.c']]^d$

Non compositional enhancements

- new goals that are both “forward” (they have the same conclusion) and “backward” (they add new hypothesis) (e.g. `and_ind`)

$$\llbracket \mu\alpha : T.c \rrbracket^{<T} = \llbracket c \rrbracket^{<T}$$

$$\llbracket \mu\alpha : T.c \rrbracket^{<T'} = \boxed{\text{we need to prove } T; \llbracket c \rrbracket^{<T}}$$

- “previous”:

$$\llbracket \tilde{\mu}x : T.\langle y \mid v_1 \circ \dots x \dots v_n \circ E \rangle \rrbracket^d =$$

we proved T ; by y ...previous ...; ...

when x does not occur in $\nu_{i_1}, \dots, \nu_{i_l}, E$

Examples 1/3

$\mu\alpha : C.\langle k || \mu\alpha : B.\langle h || \mu\alpha : A.\langle a || \alpha \rangle \circ \alpha \rangle \circ \alpha \rangle$

we need to prove C

by k ;

$h : A \rightarrow B$

we need to prove B

$k : B \rightarrow C$

by h ;

$a : A$

we need to prove A ;

C^α

by hypothesis a

done

done

done

Examples 2/3

$\mu\alpha : C.\langle a \mid \tilde{\mu}x : A.\langle h \mid x \circ \tilde{\mu}y : B.\langle k \mid y \circ \alpha \rangle \rangle \rangle$

we need to prove C

$h : A \rightarrow B$

consider a

$k : B \rightarrow C$

we proved $A(x)$

$a : A$

by $h x$;

C^α

we proved $B(y)$

by $k y$;

done;

Examples 3/3

	$\mu\alpha : C.\langle h a \circ \tilde{\mu}x : B.\langle k x \circ \alpha \rangle\rangle$
$h : A \rightarrow B$	we need to prove C ;
$k : B \rightarrow C$	by h a ;
$a : A$	we proved B (x)
<hr/>	
C^α	by k x ;
	done

Other Examples

Top-down conversion:

$$\mu\alpha : T'.\langle id \mid \mu\alpha : T.c \circ \alpha \rangle$$

we need to prove T' ; by id we need to prove T ; $\llbracket c \rrbracket$; done

Bottom-up conversion:

$$\tilde{\mu}x : T.\langle id \mid x \circ \tilde{\mu}x : T'.c \rangle$$

we proved T ; by id previous we proved $T' (x)$; $\llbracket c \rrbracket$

Overview

1. Motivations
2. A gentle introduction to the $\bar{\lambda}\mu\tilde{\mu}$ -calculus
3. Natural language rendering of intuitionistic $\lambda\mu$ -normal $\bar{\lambda}\mu\tilde{\mu}$ -terms
4. Encoding the intuitionistic $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus in OMDoc
5. Encoding OMDoc in the intuitionistic $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus

OMDoc

Proof ::= *Name?*, *Fcontent*

Fcontent ::= *Derive*

| (*Derive* | *Hypothesis*), *Fcontent*

Derive ::= *Name?*, *Type*, *Justification*

Justification ::= (*Name* | *Proof*)*

Hypothesis ::= *Name*, *Type*

Omitted: commented mathematical properties (CMPs), local definitions, primitive proofs for justifications, method names and method argument attributes (rank, to classify premises or subproofs, and arity for subproofs).

From the DTD to EBNF: *Ccontent* and *Fcontent** merged in *Fcontent* (to capture the constraint that *Fcontent** must end in a *Derive*); *Justification* and *Method* simplified to *Justification*

Encoding the $\bar{\lambda}\mu\tilde{\mu}$ -calculus in OMDoc

Overview

1. Motivations
2. A gentle introduction to the $\bar{\lambda}\mu\tilde{\mu}$ -calculus
3. Natural language rendering of intuitionistic $\lambda\mu$ -normal $\bar{\lambda}\mu\tilde{\mu}$ -terms
4. Encoding the intuitionistic $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus in OMDoc
5. Encoding OMDoc in the intuitionistic $\lambda\mu$ -normal fragment of the $\bar{\lambda}\mu\tilde{\mu}$ -calculus

Encoding OMDoc in the $\bar{\lambda}\mu\tilde{\mu}$ -calculus

First try: an *Fcontent* is a term.

$$\llbracket \text{Hyp}[Name, Type] \rrbracket^{v:T} = \lambda Name : Type. v : Type \rightarrow T$$

$$\llbracket \text{Hyp}, Fcontent \rrbracket = \llbracket \text{Hyp} \rrbracket^{\llbracket Fcontent \rrbracket}$$

$$\llbracket \text{Derive}, Fcontent \rrbracket = \text{let } v : T = \llbracket Fcontent \rrbracket \text{ in } \llbracket \text{Derive} \rrbracket^{v:T} : T$$

$$\llbracket \text{Derive}[Name?, Type, Justification[Just_1, \dots, Just_n]] \rrbracket =$$

$$\mu\alpha : Type. \langle \llbracket Just_1 \rrbracket \parallel \llbracket Just_2 \rrbracket \circ \dots \circ \llbracket JustItem_n \rrbracket \circ \alpha \rangle : Type$$

$$\llbracket \text{Derive}[Name?, Type, Justification[Just_1, \dots, Just_n]] \rrbracket^{v:T} =$$

$$\mu\alpha : T. \langle \llbracket Just_1 \rrbracket \parallel \llbracket Just_2 \rrbracket \circ \dots \circ \llbracket Just_n \rrbracket \circ \tilde{\mu}Name : Type. \langle v \parallel \alpha \rangle \rangle$$

$$\llbracket Name \rrbracket = Name$$

$$\llbracket \text{Proof}[Name?, Fcontent] \rrbracket = \llbracket Fcontent \rrbracket$$

A brief analysis [1/3]

1. The OMDoc DTD does not force the omission of the *Name* of the last derive step. However, this name is not used in the translation.
2. `previous` is embedded in OMDoc; there is no way to translate it in $\bar{\lambda}\mu\tilde{\mu}$ -calculus.
3. The $\bar{\lambda}\mu\tilde{\mu}$ -calculus gives no name (and no other attributes too) to arguments.
4. The translation generates only restricted terms for free (i.e. no $\lambda x_1 \dots \lambda x_n . x$).

A brief analysis [2/3]

5 The following constraints are induced over OMDoc documents:

- $Derive[Name?, T, Justification[Name]], Fcontent$ not admitted ($\tilde{\mu}$ -renaming)
- The element content of a $Fcontent$ element must be non empty
- The first justification must be a $Name$. Otherwise a β -redex is generated. The β -redex can be avoided using for the first justification item the rule:

$$\llbracket Proof[Name?, Fcontent] \rrbracket =$$

$$\text{let } v : T = \llbracket Fcontent \rrbracket \text{ in } \mu\alpha : T.\langle v || \alpha \rangle$$

But this rule can introduce redexes of the form $\langle \mu\alpha.c || v \circ E \rangle$ or $\langle \mu\alpha.c || \alpha \rangle$ (μ -renaming), requiring further non-structural fixes.

A brief analysis [3/3]

The translation does not preserve the natural language rendering semantics.

		we need to prove T''
		p we proved $T(x)$
p we proved $T(x)$		we need to prove T''
q we proved $T'(y)$	\Rightarrow	q we proved $T'(y)$
r		r
		done
		done
$Derive[x, T, p], Derive[y, T', q], r$		$\mu\alpha : T''. \langle p \tilde{\mu}x : T. \langle \mu\alpha : T''.$
		$\langle q \tilde{\mu}y. \langle r \alpha \rangle \rangle \alpha \rangle$

Faithful translation (an *Fcontent* is a command): $\langle p || \tilde{\mu}x : T. \langle q || \tilde{\mu}y : T'. r \rangle \rangle$

Encoding OMDoc in the $\bar{\lambda}\mu\tilde{\mu}$ -calculus

Second try: an *Fcontent* is a command.

[[