Notice

The exercises due January 31, 2008 are along this document on slides with a blue background like the present one.

They consist of 10 exercices. Each one deals with what has just been presented in the previous slides and mostly requires only a basic understanding of it.

Solutions should be sent:

- in PDF by e-mail to `joly@pps.jussieu.fr` (scanned handwritten versions in order to avoid typewriting proof trees are welcome)

or

- by snail-mail to:

Thierry Joly
Laboratoire PPS
Université Paris Diderot - Paris 7
Case 7014
75205 PARIS Cedex 13
FRANCE

Extracting programs from classical proofs

Thierry Joly

Université Paris 7

November 30 - December 1, 2007

Outline

1^{st} order language

Defined by a set of predicate symbols $P, P', P'' \ldots$ and a set of function symbols $f, f', f'' \ldots$. Each one of these symbols is given with a fixed arity. A predicate symbol of arity 0 is called *propositional symbol*. A function symbol of arity 0 is called *constant*.

Individual terms:

$$t = x \mid f(t, \ldots, t)^{\dagger}$$

Formulas:

$$A = P(t, \ldots, t)^{\dagger} \mid A \to A \mid A \wedge A \mid A \vee A \mid \neg A \mid \forall x\, A \mid \exists x\, A$$

† The arity of the symbol must be respected.

Hilbert style proof system

In Hilbert style systems, the meaning of the logical symbols is defined by a lot of axioms, e.g.

$A \to B \to A$
$(A \to B \to C) \to (A \to B) \to A \to C$
$((A \to B) \to A) \to A$

$A \wedge B \to A$
$A \wedge B \to B$
$A \to B \to A \wedge B$

$A \to A \vee B$
$B \to A \vee B$
$A \vee B \to (A \to C) \to (B \to C) \to C$

$(A \to B) \to \neg B \to \neg A$
$\neg A \to A \to B$

$\forall x\, A \to A[t/x]$
$\forall x\, (A \to B) \to A \to \forall x\, B$, only if $x \notin A$

$A[t/x] \to \exists x\, A$
$\forall x\, (A \to B) \to \exists x\, A \to B$, only if $x \notin B$

Hilbert style proof system

In Hilbert style systems, the meaning of the logical symbols is defined by a lot of axioms, e.g.

$A \to B \to A$
$(A \to B \to C) \to (A \to B) \to A \to C$
$((A \to B) \to A) \to A$

$A \wedge B \to A$
$A \wedge B \to B$
$A \to B \to A \wedge B$

$A \to A \vee B$
$B \to A \vee B$
$A \vee B \to (A \to C) \to (B \to C) \to C$

$(A \to B) \to \neg B \to \neg A$
$\neg A \to A \to B$

$\forall x\, A \to A[t/x]$
$\forall x\, (A \to B) \to A \to \forall x\, B$, only if $x \notin A$

$A[t/x] \to \exists x\, A$
$\forall x\, (A \to B) \to \exists x\, A \to B$, only if $x \notin B$

and only two inference rules are used to articulate these axioms, Modus Ponens and Universalization:

$$\frac{A \to B \quad A}{B}\ MP \qquad \frac{A}{\forall x\, A}\ U$$

Sequents

In the sequent calculus LK, the formulas are replaced by sequents:

$$A_1, A_2, \ldots, A_n \vdash B_1, B_2, \ldots, B_p \quad (n \geqslant 0,\ p \geqslant 0)$$

Intended meaning:

$$A_1 \wedge A_2 \wedge \ldots \wedge A_n \longrightarrow B_1 \vee B_2 \vee \ldots \vee B_p$$

In case $n = 0$:

$$B_1 \vee B_2 \vee \ldots \vee B_p$$

In case $p = 0$:

$$\neg(A_1 \wedge A_2 \wedge \ldots \wedge A_n)$$

$A_1, A_2, \ldots, A_n$ and $B_1, B_2, \ldots, B_n$ are multisets of formulas.
If Γ and Δ are such multisets then Γ, Δ denotes the sum of these multisets.

LK rules

Only one axiom rule!

I. Axiom rule: $A \vdash A$

LK rules

Only one axiom rule!

A lot of inference rules (two for every logical symbol)

I. Axiom rule:

$$A \vdash A$$

II. Logical rules:

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \to B \vdash \Delta}\, l{\to} \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\, r{\to}$$

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\, l{\wedge} \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \wedge B, \Delta}\, r{\wedge}$$

$$\vdots \qquad\qquad \vdots$$

$$\frac{\Gamma, A[t/x] \vdash \Delta}{\Gamma, \forall x\, A \vdash \Delta}\, l{\forall} \qquad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash \forall x\, A, \Delta}\, r{\forall}\,{}^{\dagger}$$

† Only if x does not occur free in Γ, Δ.

LK rules

Only one axiom rule!

A lot of inference rules (two for every logical symbol and more . . .)

I. Axiom rule:

$$A \vdash A$$

II. Logical rules:

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \to B \vdash \Delta}\, l\to \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\, r\to$$

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\, l\wedge \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \wedge B, \Delta}\, r\wedge$$

$$\vdots \qquad\qquad \vdots$$

$$\frac{\Gamma, A[t/x] \vdash \Delta}{\Gamma, \forall x\, A \vdash \Delta}\, l\forall \qquad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash \forall x\, A, \Delta}\, r\forall\ ^{\dagger}$$

III. Structural rules:

$$\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}\, lw \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash A, \Delta}\, rw$$

$$\frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta}\, lc \qquad \frac{\Gamma \vdash A, A, \Delta}{\Gamma \vdash A, \Delta}\, rc$$

† Only if x does not occur free in Γ, Δ.

LK rules

Only one axiom rule!

A lot of inference rules (two for every logical symbol and more . . .)

I. Axiom rule:

$$A \vdash A$$

II. Logical rules:

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \to B \vdash \Delta}\, l\to \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\, r\to$$

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\, l\wedge \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \wedge B, \Delta}\, r\wedge$$

$$\frac{\Gamma, A[t/x] \vdash \Delta}{\Gamma, \forall x\, A \vdash \Delta}\, l\forall \qquad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash \forall x\, A, \Delta}\, r\forall\ ^{\dagger}$$

III. Structural rules:

$$\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}\, lw \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash A, \Delta}\, rw$$

$$\frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta}\, lc \qquad \frac{\Gamma \vdash A, A, \Delta}{\Gamma \vdash A, \Delta}\, rc$$

IV. Cut rule:

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

† Only if x does not occur free in Γ, Δ.

Cut elimination

The main property of LK (with quite a few consequences in proof theory):

Cut rule is redundant
(i.e. all what can be proved by using it can be proved without).

$\rightsquigarrow$ Subformula property: In a cut free proof, all the formulas are subformulas of a formula of the conclusion.

Moreover:

Cut rules can recursively be removed from any proof throughout a rewrite sequence of the proof.

Cut elimination procedure (sketch)

The formula that is erased from the premisses to the conclusion of a cut rule is called *cut formula*:

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ cut$$

3 kinds of cuts:

- *ax-cut* (axiomatic cut): one of the premisses (at least) is an axiom.
- *l-cut* (logical cut): in every pemiss the cut formula has just been built by a logical rule.
- *s-cut* (structural cut): otherwise.

Axiomatics cut are removed at once:

$$\frac{\vdots\,\Pi \qquad \vdots}{\dfrac{A \vdash A \quad \Gamma, A \vdash \Delta}{\Gamma \vdash \Delta}\ ax\text{-}cut} \longrightarrow \begin{array}{c}\vdots\,\Pi\\ \Gamma, A \vdash \Delta\end{array} \qquad \frac{\vdots\,\Pi \qquad \vdots}{\dfrac{\Gamma \vdash A, \Delta \quad A \vdash A}{\Gamma \vdash \Delta}\ ax\text{-}cut} \longrightarrow \begin{array}{c}\vdots\,\Pi\\ \Gamma, A \vdash \Delta\end{array}$$

Elimination of the logical cuts

Every logical cut can be replaced by cuts with smaller cut formulas:

$$\dfrac{\dfrac{\dfrac{\vdots\,\Pi}{\Gamma, A \vdash B, \Delta}}{\Gamma \vdash A \to B, \Delta}\, r\to \quad \dfrac{\dfrac{\vdots\,\Pi'_1}{\Gamma' \vdash A, \Delta'} \quad \dfrac{\vdots\,\Pi'_2}{\Gamma', B \vdash \Delta'}}{\Gamma', A \to B \vdash \Delta'}\, l\to}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, l\text{-}cut \quad \longrightarrow \quad \dfrac{\dfrac{\dfrac{\vdots\,\Pi'_1}{\Gamma' \vdash A, \Delta'} \quad \dfrac{\vdots\,\Pi}{\Gamma, A \vdash B, \Delta}}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\, cut \quad \dfrac{\vdots\,\Pi'_2}{\Gamma', B \vdash \Delta'}}{\Gamma, \Gamma', \Gamma' \vdash \Delta, \Delta', \Delta'}\, cut}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, lc/rc$$

$$\dfrac{\dfrac{\dfrac{\vdots\,\Pi_1}{\Gamma \vdash A, \Delta} \quad \dfrac{\vdots\,\Pi_2}{\Gamma \vdash B, \Delta}}{\Gamma \vdash A \wedge B, \Delta}\, r\wedge \quad \dfrac{\vdots\,\Pi'}{\Gamma', A, B \vdash \Delta'}\, l\wedge}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, l\text{-}cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi_2}{\Gamma \vdash B, \Delta} \quad \dfrac{\dfrac{\vdots\,\Pi_1}{\Gamma \vdash A, \Delta} \quad \dfrac{\vdots\,\Pi'}{\Gamma', A, B \vdash \Delta'}}{\Gamma, \Gamma', B \vdash \Delta, \Delta'}\, cut}{\Gamma, \Gamma, \Gamma' \vdash \Delta, \Delta, \Delta'}\, cut}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, lc/rc$$

. . .

Elimination of the structural cuts

Suppose that the last rule of Π creates the cut formula A but not the last rule of Π':

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ \textit{s-cut}$$

Elimination of the structural cuts

Suppose that the last rule of Π creates the cut formula A but not the last rule of Π':

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ \textit{s-cut}$$

Then we make Π climb up Π' as follows:

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A, \Delta}\qquad \dfrac{\vdots\,\Pi'}{\Gamma', A, B \vdash C, \Delta'}\ r\!\rightarrow}{\Gamma, \Gamma' \vdash B \rightarrow C, \Delta, \Delta'}\ \textit{s-cut}$$

Elimination of the structural cuts

Suppose that the last rule of Π creates the cut formula A but not the last rule of Π':

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ \textit{s-cut}$$

Then we make Π climb up Π' as follows:

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \dfrac{\vdots\,\Pi' \quad \Gamma', A, B \vdash C, \Delta'}{\Gamma', A \vdash B \to C, \Delta'}\ r\to \\ \Gamma \vdash A, \Delta & \end{array}}{\Gamma, \Gamma' \vdash B \to C, \Delta, \Delta'}\ \textit{s-cut} \quad \longrightarrow \quad \dfrac{\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A, B \vdash C, \Delta'\end{array}}{\Gamma, \Gamma', B \vdash C, \Delta, \Delta'}\ \textit{cut}}{\Gamma, \Gamma' \vdash B \to C, \Delta, \Delta'}\ r\to$$

Elimination of the structural cuts

Suppose that the last rule of Π creates the cut formula A but not the last rule of Π':

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ \textit{s-cut}$$

Then we make Π climb up Π' as follows:

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \dfrac{\vdots\,\Pi' \quad \Gamma', A, B \vdash C, \Delta'}{\Gamma', A \vdash B \to C, \Delta'}\ r\to \\ \Gamma \vdash A, \Delta & \end{array}}{\Gamma, \Gamma' \vdash B \to C, \Delta, \Delta'}\ \textit{s-cut} \quad \longrightarrow \quad \dfrac{\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A, B \vdash C, \Delta'\end{array}}{\Gamma, \Gamma', B \vdash C, \Delta, \Delta'}\ \textit{cut}}{\Gamma, \Gamma' \vdash B \to C, \Delta, \Delta'}\ r\to$$

Elimination of the structural cuts

Suppose that the last rule of Π creates the cut formula A but not the last rule of Π':

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ s\text{-}cut$$

Then we make Π climb up Π' as follows:

$$\dfrac{\vdots\,\Pi \quad \dfrac{\vdots\,\Pi'}{\Gamma', A, B \vdash C, \Delta'}\ r\!\to}{\dfrac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash B \to C, \Delta'}{\Gamma, \Gamma' \vdash B \to C, \Delta, \Delta'}\ s\text{-}cut} \longrightarrow \dfrac{\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A, B \vdash C, \Delta'\end{array}}{\Gamma, \Gamma', B \vdash C, \Delta, \Delta'}\ cut}{\Gamma, \Gamma' \vdash B \to C, \Delta, \Delta'}\ r\!\to$$

$$\dfrac{\vdots\,\Pi \quad \dfrac{\begin{array}{cc}\vdots\,\Pi'_1 & \vdots\,\Pi'_2 \\ \Gamma', A \vdash B, \Delta' & \Gamma', A, C \vdash \Delta'\end{array}}{\Gamma', A, B \to C \vdash \Delta'}\ l\!\to}{\dfrac{\Gamma \vdash A, \Delta \quad \Gamma', A, B \to C \vdash \Delta'}{\Gamma, \Gamma', B \to C \vdash \Delta, \Delta'}\ s\text{-}cut}$$

Elimination of the structural cuts

Suppose that the last rule of Π creates the cut formula A but not the last rule of Π':

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ s\text{-}cut$$

Then we make Π climb up Π' as follows:

$$\dfrac{\vdots\,\Pi \quad \dfrac{\vdots\,\Pi'}{\Gamma', A, B \vdash C, \Delta'}\ r\!\to}{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash B \to C, \Delta'}\ s\text{-}cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi \quad \vdots\,\Pi'}{\Gamma \vdash A, \Delta \quad \Gamma', A, B \vdash C, \Delta'}\ cut}{\Gamma, \Gamma', B \vdash C, \Delta, \Delta'}\ r\!\to$$

$$\dfrac{\vdots\,\Pi \quad \dfrac{\vdots\,\Pi_1' \quad \vdots\,\Pi_2'}{\Gamma', A \vdash B, \Delta' \quad \Gamma', A, C \vdash \Delta'}\ l\!\to}{\Gamma \vdash A, \Delta \quad \Gamma', A, B \to C \vdash \Delta'}\ s\text{-}cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi \quad \vdots\,\Pi_1'}{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash B, \Delta'} \quad \dfrac{\vdots\,\Pi \quad \vdots\,\Pi_2'}{\Gamma \vdash A, \Delta \quad \Gamma', A, C \vdash \Delta'}}{\Gamma, \Gamma', B \to C \vdash \Delta, \Delta'}\ l\!\to$$

Elimination of the structural cuts

Suppose that the last rule of Π creates the cut formula A but not the last rule of Π':

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ s\text{-}cut$$

Then we make Π climb up Π' as follows:

$$\dfrac{\vdots\,\Pi \quad \dfrac{\vdots\,\Pi' \quad \Gamma', A, B \vdash C, \Delta'}{\Gamma', A \vdash B \to C, \Delta'}\ r\to}{\Gamma, \Gamma' \vdash B \to C, \Delta, \Delta'}\ s\text{-}cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi \quad \vdots\,\Pi' \quad \Gamma \vdash A, \Delta \quad \Gamma', A, B \vdash C, \Delta'}{\Gamma, \Gamma', B \vdash C, \Delta, \Delta'}\ cut}{\Gamma, \Gamma' \vdash B \to C, \Delta, \Delta'}\ r\to$$

$$\dfrac{\vdots\,\Pi \quad \dfrac{\vdots\,\Pi_1' \quad \vdots\,\Pi_2' \quad \Gamma', A \vdash B, \Delta' \quad \Gamma', A, C \vdash \Delta'}{\Gamma', A, B \to C \vdash \Delta'}\ l\to}{\Gamma, \Gamma', B \to C \vdash \Delta, \Delta'}\ s\text{-}cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi \quad \vdots\,\Pi_1' \quad \Gamma \vdash A, \Delta \quad \Gamma', A \vdash B, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'} \quad \dfrac{\vdots\,\Pi \quad \vdots\,\Pi_2' \quad \Gamma \vdash A, \Delta \quad \Gamma', A, C \vdash \Delta'}{\Gamma, \Gamma', C \vdash \Delta, \Delta'}}{\Gamma, \Gamma', B \to C \vdash \Delta, \Delta'}\ l\to$$

Elimination of the structural cuts

Suppose that the last rule of Π creates the cut formula A but not the last rule of Π':

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ s\text{-}cut \quad (\Gamma \vdash A, \Delta \curvearrowright \Gamma', A \vdash \Delta_0)$$

Then we make Π climb up Π' as follows:

$$\dfrac{\vdots\,\Pi \quad \dfrac{\vdots\,\Pi' \quad \Gamma', A, B \vdash C, \Delta'}{\Gamma', A \vdash B \to C, \Delta'}\ r\to}{\Gamma, \Gamma' \vdash B \to C, \Delta, \Delta'}\ s\text{-}cut \quad (\Gamma \vdash A, \Delta \curvearrowright \Gamma', A \vdash B \to C, \Delta') \longrightarrow \dfrac{\dfrac{\vdots\,\Pi \quad \vdots\,\Pi' \quad \Gamma \vdash A, \Delta \curvearrowright \Gamma', A, B \vdash C, \Delta'}{\Gamma, \Gamma', B \vdash C, \Delta, \Delta'}\ cut}{\Gamma, \Gamma' \vdash B \to C, \Delta, \Delta'}\ r\to$$

$$\dfrac{\vdots\,\Pi \quad \dfrac{\vdots\,\Pi'_1 \quad \vdots\,\Pi'_2 \quad \Gamma', A \vdash B, \Delta' \quad \Gamma', A, C \vdash \Delta'}{\Gamma', A, B \to C \vdash \Delta'}\ l\to}{\Gamma, \Gamma', B \to C \vdash \Delta, \Delta'}\ s\text{-}cut \quad (\Gamma \vdash A, \Delta \curvearrowright \Gamma', A, B \to C \vdash \Delta') \longrightarrow \dfrac{\dfrac{\vdots\,\Pi \quad \vdots\,\Pi'_1 \quad \Gamma \vdash A, \Delta \curvearrowright \Gamma', A \vdash B, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'} \quad \dfrac{\vdots\,\Pi \quad \vdots\,\Pi'_2 \quad \Gamma \vdash A, \Delta \curvearrowright \Gamma', A, C \vdash \Delta'}{\Gamma, \Gamma', C \vdash \Delta, \Delta'}}{\Gamma, \Gamma', B \to C \vdash \Delta, \Delta'}\ l\to$$

. . .

Elimination of the structural cuts

. . .

$$\dfrac{\Gamma \vdash A, \Delta \quad \rightsquigarrow \quad \dfrac{\Gamma' \vdash \Delta'}{\Gamma', A \vdash \Delta'}\, lw}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, s\text{-}cut$$

Elimination of the structural cuts

. . .

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A, \Delta} \quad \dfrac{\dfrac{\vdots\,\Pi'}{\Gamma' \vdash \Delta'}\,lw}{\Gamma', A \vdash \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,s\text{-}cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi'}{\Gamma' \vdash \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,lw/rw$$

Elimination of the structural cuts

. . .

$$\dfrac{\vdots\,\Pi \qquad \dfrac{\vdots\,\Pi'}{\Gamma' \vdash \Delta'}\,lw}{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}\,s\text{-}cut \quad \longrightarrow \quad \dfrac{\vdots\,\Pi'}{\dfrac{\Gamma' \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,lw/rw}$$

. . . until all the remaining copies of the cut reach on their r.h.s. premisses either an axiom or a logical rule creating the cut formula A

Elimination of the structural cuts

. . .

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A, \Delta} \quad \dfrac{\dfrac{\vdots\,\Pi'}{\Gamma' \vdash \Delta'}}{\Gamma', A \vdash \Delta'}\,lw}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,s\text{-}cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi'}{\Gamma' \vdash \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,lw/rw$$

. . . until all the remaining copies of the cut reach on their r.h.s. premisses either an axiom or a logical rule creating the cut formula A, and then become rules *ax-cut* or *l-cut* that we eliminate as previously.

Elimination of the structural cuts

. . .

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A, \Delta} \quad \dfrac{\dfrac{\vdots\,\Pi'}{\Gamma' \vdash \Delta'}}{\Gamma', A \vdash \Delta'}\,lw}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,s\text{-}cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi'}{\Gamma' \vdash \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,lw/rw$$

. . . until all the remaining copies of the cut reach on their r.h.s. premisses either an axiom or a logical rule creating the cut formula A, and then become rules *ax-cut* or *l-cut* that we eliminate as previously.

If the last rule of Π creates the cut formula A but not the last rule of Π':

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A, \Delta} \qquad \dfrac{\vdots\,\Pi'}{\Gamma', A \vdash \Delta_0}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\,s\text{-}cut$$

Elimination of the structural cuts

. . .

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A, \Delta} \quad \dfrac{\dfrac{\vdots\,\Pi'}{\Gamma' \vdash \Delta'}}{\Gamma', A \vdash \Delta'}\,lw}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,s\text{-}cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi'}{\Gamma' \vdash \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,lw/rw$$

. . . until all the remaining copies of the cut reach on their r.h.s. premisses either an axiom or a logical rule creating the cut formula A, and then become rules *ax-cut* or *l-cut* that we eliminate as previously.

If the last rule of Π creates the cut formula A but not the last rule of Π':

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A, \Delta} \quad \dfrac{\vdots\,\Pi'}{\Gamma', A \vdash \Delta_0}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\,s\text{-}cut$$

then we symetrically make Π' climb up Π'.

Elimination of the structural cuts

At last, if none of the subproofs Π, Π' creates the cut formula A:

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ \textit{s-cut}$$

Elimination of the structural cuts

At last, if none of the subproofs Π, Π' creates the cut formula A:

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ s\text{-}cut$$

we may

- first make Π climb up Π' until all the copies of the cut reach on their r.h.s. premisses an axiom or a logical rule creating A

Elimination of the structural cuts

At last, if none of the subproofs Π, Π' creates the cut formula A:

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ s\text{-}cut$$

we may

- first make Π climb up Π' until all the copies of the cut reach on their r.h.s. premisses an axiom or a logical rule creating A
- and then make these cuts go up left until each of their copies becomes an *ax-cut* or a *l-cut*.

Elimination of the structural cuts

At last, if none of the subproofs Π, Π' creates the cut formula A:

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ \textit{s-cut}$$

we may

- first make Π climb up Π' until all the copies of the cut reach on their r.h.s. premisses an axiom or a logical rule creating A
- and then make these cuts go up left until each of their copies becomes an *ax-cut* or a *l-cut*.

or we may

Elimination of the structural cuts

At last, if none of the subproofs Π, Π' creates the cut formula A:

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ s\text{-}cut$$

we may

- first make Π climb up Π' until all the copies of the cut reach on their r.h.s. premisses an axiom or a logical rule creating A
- and then make these cuts go up left until each of their copies becomes an *ax-cut* or a *l-cut*.

or we may first make the cut go up left

Elimination of the structural cuts

At last, if none of the subproofs Π, Π' creates the cut formula A:

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ \textit{s-cut}$$

we may

- first make Π climb up Π' until all the copies of the cut reach on their r.h.s. premisses an axiom or a logical rule creating A
- and then make these cuts go up left until each of their copies becomes an *ax-cut* or a *l-cut*.

or we may first make the cut go up left then right.

Elimination of the structural cuts

At last, if none of the subproofs Π, Π' creates the cut formula A:

$$\dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A, \Delta & \Gamma', A \vdash \Delta_0\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta_0}\ s\text{-}cut$$

we may

- first make Π climb up Π' until all the copies of the cut reach on their r.h.s. premisses an axiom or a logical rule creating A
- and then make these cuts go up left until each of their copies becomes an *ax-cut* or a *l-cut*.

or we may first make the cut go up left then right.

THIS SINGLE CHOICE LEADS IN GENERAL
TO ESSENTIALLY DIFFERENT CUT FREE PROOFS.

Natural deduction ND: the differences between LK and ND

In ND:

- Only one formula on the r.h.s. of a sequent: $\Gamma \vdash C$.
- No more logical left introduction rules, right elimination rules instead

  E.g. $\dfrac{\Gamma, A, B \vdash C}{\Gamma, A \wedge B \vdash C}\, I\wedge$ is replaced by the rules $\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\, \wedge e_1$ and $\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\, \wedge e_2$
- No explicit structural rule: the l.h.s. sides of the sequents are now *sets* of labelled formulas $\Gamma = \{C_1^{x_1}, \ldots, C_n^{x_n}\} \rightsquigarrow$ in binary rules such as $\dfrac{\Gamma \vdash A \to B \quad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B} \to e$ $(\Gamma, \Gamma' \stackrel{\text{def}}{=} \Gamma \cup \Gamma')$, contraction rules are implicitly performed, e.g. $\dfrac{C^x, D^y \vdash A \to B \quad C^x, D^z \vdash A}{C^x, D^y, D^z \vdash B} \to e$.
- No explicit cut rule: a cut is now just a (right) introduction rule immediatly followed by a (right) elimination rule destroying the created formula, that we still call *cut formula*.

  E.g. $\dfrac{\dfrac{\Gamma \vdash A \quad \Gamma' \vdash B}{\Gamma, \Gamma' \vdash A \wedge B}\, \wedge i}{\Gamma, \Gamma' \vdash A}\, \wedge e_1 \qquad \dfrac{\dfrac{\Gamma, A^x \vdash B}{\Gamma \vdash A \to B} \to i \quad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B} \to e$

ND rules

Only one axiom rule as in LK and still a lot of inference rules.

I. Axiom rule:

$$A^x \vdash A$$

II. Logical rules:

$$\frac{\Gamma \vdash A \to B \quad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B} \to e \qquad \frac{\Gamma, A^x \vdash B}{\Gamma \vdash A \to B} \to i$$

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A} \wedge e_1 \quad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B} \wedge e_2 \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B} \wedge i$$

$$\vdots \qquad\qquad \vdots$$

$$\frac{\Gamma \vdash \forall x\, A}{\Gamma \vdash A[t/x]} \forall e \qquad \frac{\Gamma \vdash A}{\Gamma \vdash \forall x\, A} \forall i^{\dagger}$$

III. Structural rule:

$$\frac{\Gamma \vdash C}{\Gamma, \Gamma' \vdash C} w$$

† Only if x does not occur free in Γ.

But we just said: No structural rule in ND!

Well, in the more traditional formulation of ND with single formulas (i.e. as for Hilbert style systems), there is no rule w.

But we chose the sequent formulation of ND in order to make it look closer to LK.

Even in the sequent formulation of ND, rule w is not necessary in case we adopt axiom rules of the form: $\Gamma, A^x \vdash A$.

Nevertheless, ND is more natural when introducing additional assumptions only at the point where they are required, e.g.

$$\dfrac{\dfrac{\begin{array}{c} C^z \vdash C \\ \vdots \\ \Gamma \vdash B \end{array}}{\Gamma, A^x \vdash B}\, w}{\Gamma \vdash A \to B} \to i \qquad \text{instead of} \qquad \dfrac{\begin{array}{c} A^x, C^z \vdash C \\ \vdots \\ A^x, \Gamma \vdash B \end{array}}{\Gamma \vdash A \to B} \to i \qquad \text{(material implication)}$$

Cuts & subformula property in ND

Since we consider ND with an explicit rule w, cuts are slightly more complex than previously stated: rules w may occur between the introduction rule and the elimination rule below.

$$\dfrac{\dfrac{\dfrac{\Gamma, A^x \vdash B}{\Gamma \vdash A \to B}\to i}{\Gamma, \Gamma'' \vdash A \to B}\, w \qquad \Gamma' \vdash A}{\Gamma, \Gamma', \Gamma'' \vdash B}\to e \qquad \dfrac{\dfrac{\dfrac{\Gamma \vdash A \quad \Gamma' \vdash B}{\Gamma, \Gamma' \vdash A \wedge B}\wedge i}{\Gamma, \Gamma', \Gamma'' \vdash A \wedge B}\, w}{\Gamma, \Gamma', \Gamma'' \vdash A}\wedge e_1 \qquad \cdots$$

If there is no such cut in the proof, then every formula not in the last sequent is either a subformula of a formula in the sequent just below or a *proper* subformula of another formula in the proof.
$\rightsquigarrow$ Every formula is a subformula of a formula in the last sequent.

Hence, subformula property holds for this definition of a cut.

Eliminating cuts in ND

In order to get rid of the rules w between the introduction rule and the elimination rule of a cut, we can always commute them with the elimination rule:

$$\dfrac{\dfrac{\dfrac{\vdots\,\Pi}{\dfrac{\Gamma, A^x \vdash B}{\Gamma \vdash A \to B}\to i}}{\Gamma, \Gamma'' \vdash A \to B}\,w \qquad \dfrac{\vdots\,\Pi'}{\Gamma' \vdash A}}{\Gamma, \Gamma', \Gamma'' \vdash B}\to e \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi}{\dfrac{\Gamma, A^x \vdash B}{\Gamma \vdash A \to B}\to i} \qquad \dfrac{\vdots\,\Pi'}{\Gamma' \vdash A}}{\dfrac{\Gamma, \Gamma' \vdash B}{\Gamma, \Gamma', \Gamma'' \vdash B}\,w}\to e$$

$$\dfrac{\dfrac{\dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma \vdash A \qquad \Gamma' \vdash B}\wedge i}{\Gamma, \Gamma' \vdash A \wedge B}\,w}{\Gamma, \Gamma', \Gamma'' \vdash A \wedge B}\wedge e_1 \quad \longrightarrow \quad \dfrac{\dfrac{\dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma \vdash A \qquad \Gamma' \vdash B}\wedge i}{\Gamma, \Gamma' \vdash A \wedge B}\wedge e_1}{\Gamma, \Gamma', \Gamma'' \vdash A}\,w$$

• • •

Eliminating cuts in ND

It remains to eliminate the cuts as formerly defined:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma, A^x \vdash B\end{array}}{\Gamma \vdash A \to B}\to i \qquad \begin{array}{c}\vdots\,\Pi'\\ \Gamma' \vdash A\end{array}}{\Gamma, \Gamma' \vdash B}\to e \longrightarrow \dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma, A^x \vdash B\end{array} \overset{x}{\frown} \begin{array}{c}\vdots\,\Pi'\\ \Gamma' \vdash A\end{array}}{\Gamma, \Gamma' \vdash B}$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma \vdash A\end{array} \quad \begin{array}{c}\vdots\,\Pi'\\ \Gamma' \vdash B\end{array}}{\Gamma, \Gamma' \vdash A \wedge B}\wedge i}{\Gamma, \Gamma' \vdash A}\wedge e_1 \longrightarrow \dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma \vdash A\end{array}}{\Gamma, \Gamma' \vdash A}\, w$$

. . .

Here, $\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma, A^x \vdash B\end{array} \overset{x}{\frown} \begin{array}{c}\vdots\,\Pi'\\ \Gamma' \vdash A\end{array}}{\Gamma, \Gamma' \vdash B}$ does not denote an explicit rule *s-cut* with cut formula A on the point to climb up its left premiss. It is a notation for the proof obtained by removing every occurrence of A^x in the proof Π and by plugging at once the proof Π' to every axiom rule $[A^x] \vdash A$ of Π.

Eliminating cuts in ND

Recursive definition of $\dfrac{\begin{array}{c}\vdots\,\Pi \\ \Gamma, A^x \vdash B\end{array} \overset{x}{\frown} \begin{array}{c}\vdots\,\Pi' \\ \Gamma' \vdash A\end{array}}{\Gamma, \Gamma' \vdash B}$:

$$\dfrac{\begin{array}{c}\vdots\,\Pi' \\ A^x \vdash A\end{array} \overset{x}{\frown} \begin{array}{c}\vdots\,\Pi' \\ \Gamma' \vdash A\end{array}}{\Gamma' \vdash A} \quad \overset{\text{def}}{=\!=} \quad \begin{array}{c}\vdots\,\Pi' \\ \Gamma' \vdash A\end{array}$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\,\Pi \\ \Gamma_0 \vdash C\end{array}}{\Gamma, A^x \vdash C}\, w \overset{x}{\frown} \begin{array}{c}\vdots\,\Pi' \\ \Gamma' \vdash A\end{array}}{\Gamma, \Gamma' \vdash C} \quad \overset{\text{def}}{=\!=} \quad \dfrac{\begin{array}{c}\vdots\,\Pi \\ \Gamma_0 \vdash C\end{array}}{\Gamma, \Gamma' \vdash C}\, w \qquad \text{if } A^x \notin \Gamma_0$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\,\Pi_1 \\ \Gamma_1 \vdash C_1\end{array} \cdots \begin{array}{c}\vdots\,\Pi_n \\ \Gamma_n \vdash C_n\end{array}}{\Gamma \vdash C}\, R \overset{x}{\frown} \begin{array}{c}\vdots\,\Pi' \\ \Gamma' \vdash A\end{array}}{\Gamma \smallsetminus \{A^x\}, \Gamma' \vdash C} \quad \overset{\text{def}}{=\!=} \quad \dfrac{\begin{array}{c}\vdots\,\Pi_1 \\ \Gamma_1 \vdash C_1\end{array} \overset{x}{\frown} \begin{array}{c}\vdots\,\Pi' \\ \Gamma' \vdash A\end{array} \quad \cdots \quad \begin{array}{c}\vdots\,\Pi_n \\ \Gamma_n \vdash C_n\end{array} \overset{x}{\frown} \begin{array}{c}\vdots\,\Pi' \\ \Gamma' \vdash A\end{array}}{\Gamma \smallsetminus \{A^x\}, \Gamma' \vdash C}\, R$$

otherwise

Eliminating cuts in ND

Church-Rosser Property (CR) If $\Pi \twoheadrightarrow \Pi_1$ and $\Pi \twoheadrightarrow \Pi_2$ (i.e. if a proof Π can be rewritten into proofs Π_1 and Π_2, possibly $\Pi = \Pi_1$ or $\Pi = \Pi_2$) then there is a proof Π' such that $\Pi_1 \twoheadrightarrow \Pi'$ and $\Pi_2 \twoheadrightarrow \Pi'$.

$\rightsquigarrow$ If Π_1 and Π_2 are cut free then $\Pi_1 = \Pi_2$.
In other words, all the rewrite sequences may lead to a unique cut free proof, called *normal form of the proof* Π.
A rewrite sequence leading to this cut free proof is called *normalization of* Π.

Strong Normalization Property (SN) Every rewrite sequence $\Pi \to \Pi_1 \to \Pi_2 \to \Pi_3 \to \cdots$ is finite.

$\rightsquigarrow$ Every rewrite sequence leads to the normal form of Π.

ND is not classically complete

Unfortunately, ND does not prove every classically true formula. It is a system for *intuitionistic logic*.

Every intuitionistically provable statement is classically provable but not vice versa.

E.g. $(A \to B \vee C) \to (A \to B) \vee (A \to C)$ is classically provable but not intuitionistically.

Heyting semantics

- A proof of $A \wedge B$ is an ordered pair (π, π'), where π is a proof of A and π' a proof of B.
- A proof of $A \vee B$ is either $\mathrm{left}\,\pi$ where π proves A or $\mathrm{right}\,\pi'$ where π' proves B.
- A proof of $A \longrightarrow B$ is a procedure[†] f which maps any proof π of A to a proof $f(\pi)$ of B.
- A proof of $\neg A$ is a procedure[†] f which maps any couple (π, B) where π proves A to a proof of B.
- A proof of $\forall x\, A$ is a procedure[†] f taking any individual t to a proof $f(t)$ of $A[t/x]$.
- A proof of $\exists x\, A$ is a couple (t, π) where t is an individual and π a proof of $A[t/x]$.

[†] By a procedure, we mean a constructive one.

The usual way to handle negation in intuitionistic logic

According to Heyting semantics:

- A proof of $\neg A$ is a procedure f which maps any couple (π, B) where π proves A to a proof of B.
- A proof of $A \longrightarrow B$ is a procedure f which maps any proof π of A to a proof $f(\pi)$ of B.

$\rightsquigarrow$ There is a formula $\bot$ together with a procedure g which maps any couple (π, A) where π proves $\bot$ to a proof of A. Take e.g. $\bot = P \wedge \neg P$, where P is any formula.

The usual way to handle negation in intuitionistic logic

According to Heyting semantics:

- A proof of $\neg A$ is a procedure f which maps any couple (π, B) where π proves A to a proof of B.
- A proof of $A \longrightarrow B$ is a procedure f which maps any proof π of A to a proof $f(\pi)$ of B.

$\rightsquigarrow$ There is a formula $\bot$ together with a procedure g which maps any couple (π, A) where π proves $\bot$ to a proof of A. Take e.g. $\bot = P \wedge \neg P$, where P is any formula.

$\rightsquigarrow$ Proving $\neg A$ amounts to proving $A \longrightarrow \bot$.

The usual way to handle negation in intuitionistic logic

According to Heyting semantics:

- A proof of $\neg A$ is a procedure f which maps any couple (π, B) where π proves A to a proof of B.
- A proof of $A \longrightarrow B$ is a procedure f which maps any proof π of A to a proof $f(\pi)$ of B.

$\rightsquigarrow$ There is a formula $\bot$ together with a procedure g which maps any couple (π, A) where π proves $\bot$ to a proof of A. Take e.g. $\bot = P \wedge \neg P$, where P is any formula.

$\rightsquigarrow$ Proving $\neg A$ amounts to proving $A \longrightarrow \bot$.

In the intuitionistic systems, $\bot$ is usually taken as a primitive instead of $\neg$. The corresponding procedure g then has to be explicited by a formal rule, called *rule of intuitionistic absurdity*, e.g. in ND:

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash A}\ int.abs.$$

and $\neg A$ then is just a notation:

$$\neg A \stackrel{\text{def}}{=} A \longrightarrow \bot$$

Ways & means

Besides the classical and intuitionistic variants, we may play on the expressivity of the logical language:

- We may quantify over the predicates just as we quantify over the individuals

Ways & means

Besides the classical and intuitionistic variants, we may play on the expressivity of the logical language:

- We may quantify over the predicates just as we quantify over the individuals: Let us add to the language predicate variables $X^n, Y^p, Z^q \ldots$ of fixed arities $n, p, q \ldots$ ($X(u_1, \ldots, u_n)$ is now a well formed formula; if $n = 0$ then X is a propositional variable and a well formed formula on its own). Let us allow quantification over these variables (e.g. $\forall X(\forall x\, X(x) \to X(0))$ is a well formed formula).

Ways & means

Besides the classical and intuitionistic variants, we may play on the expressivity of the logical language:

- We may quantify over the predicates just as we quantify over the individuals: Let us add to the language predicate variables $X^n, Y^p, Z^q \ldots$ of fixed arities $n, p, q \ldots$ ($X(u_1, \ldots, u_n)$ is now a well formed formula; if $n = 0$ then X is a propositional variable and a well formed formula on its own). Let us allow quantification over these variables (e.g. $\forall X(\forall x\, X(x) \to X(0))$ is a well formed formula).
  $\rightsquigarrow$ This yields 2^{nd} *order logic*.

Ways & means

Besides the classical and intuitionistic variants, we may play on the expressivity of the logical language:

- We may quantify over the predicates just as we quantify over the individuals: Let us add to the language predicate variables $X^n, Y^p, Z^q \ldots$ of fixed arities $n, p, q \ldots$ ($X(u_1, \ldots, u_n)$ is now a well formed formula; if $n = 0$ then X is a propositional variable and a well formed formula on its own). Let us allow quantification over these variables (e.g. $\forall X(\forall x\, X(x) \to X(0))$ is a well formed formula).
  $\rightsquigarrow$ This yields 2^{nd} *order logic*.
- We may also restrict the 1^{st} order language to a (still interesting) minimum

Ways & means

Besides the classical and intuitionistic variants, we may play on the expressivity of the logical language:

- We may quantify over the predicates just as we quantify over the individuals: Let us add to the language predicate variables $X^n, Y^p, Z^q \ldots$ of fixed arities $n, p, q \ldots$ ($X(u_1, \ldots, u_n)$ is now a well formed formula; if $n = 0$ then X is a propositional variable and a well formed formula on its own). Let us allow quantification over these variables (e.g. $\forall X(\forall x\, X(x) \to X(0))$ is a well formed formula).
  $\rightsquigarrow$ This yields 2^{nd} *order logic*.
- We may also restrict the 1^{st} order language to a (still interesting) minimum: Allow only predicate symbols of arity 0 (i.e. propositional symbols), no quantifier (since they are now meaningless) and only the connective $\to$.

Ways & means

Besides the classical and intuitionistic variants, we may play on the expressivity of the logical language:

- We may quantify over the predicates just as we quantify over the individuals: Let us add to the language predicate variables $X^n, Y^p, Z^q \ldots$ of fixed arities $n, p, q \ldots$ ($X(u_1, \ldots, u_n)$ is now a well formed formula; if $n = 0$ then X is a propositional variable and a well formed formula on its own). Let us allow quantification over these variables (e.g. $\forall X(\forall x\, X(x) \to X(0))$ is a well formed formula).
  $\rightsquigarrow$ This yields 2^{nd} *order logic*.
- We may also restrict the 1^{st} order language to a (still interesting) minimum: Allow only predicate symbols of arity 0 (i.e. propositional symbols), no quantifier (since they are now meaningless) and only the connective $\to$.
  $\rightsquigarrow$ This gives the *minimal logic*.

2×3

	Classical logic	Intuitionistic logic
2^{nd} order logic		
1^{st} order logic		
Minimal logic		

A nice feature of 2^{nd} order logic: Impredicativity

From $A \wedge B$ it follows that for all propositions X, if $A \to B \to X$ then X.

A nice feature of 2^{nd} order logic: Impredicativity

From $A \wedge B$ it follows that for all propositions X, if $A \to B \to X$ then X.
In a 2^{nd} order logic formula: $\forall X((A \to B \to X) \to X)$.

A nice feature of 2^{nd} order logic: Impredicativity

From $A \wedge B$ it follows that for all propositions X, if $A \to B \to X$ then X.
In a 2^{nd} order logic formula: $\forall X((A \to B \to X) \to X)$.
Conversely, suppose that $\forall X((A \to B \to X) \to X)$. Then in particular for $X = A \wedge B$: $(A \to B \to A \wedge B) \to A \wedge B$, hence $A \wedge B$.

A nice feature of 2^{nd} order logic: Impredicativity

From $A \wedge B$ it follows that for all propositions X, if $A \to B \to X$ then X.
In a 2^{nd} order logic formula: $\forall X((A \to B \to X) \to X)$.
Conversely, suppose that $\forall X((A \to B \to X) \to X)$. Then in particular for $X = A \wedge B$: $(A \to B \to A \wedge B) \to A \wedge B$, hence $A \wedge B$.
$\rightsquigarrow$ $A \wedge B$ is naturally equivalent to $\forall X((A \to B \to X) \to X)$.

A nice feature of 2^{nd} order logic: Impredicativity

From $A \wedge B$ it follows that for all propositions X, if $A \to B \to X$ then X.
In a 2^{nd} order logic formula: $\forall X((A \to B \to X) \to X)$.
Conversely, suppose that $\forall X((A \to B \to X) \to X)$. Then in particular for $X = A \wedge B$: $(A \to B \to A \wedge B) \to A \wedge B$, hence $A \wedge B$.
$\rightsquigarrow$ $A \wedge B$ is naturally equivalent to $\forall X((A \to B \to X) \to X)$.

Similarly, $\bot$, $A \vee B$, $\exists z\, A$ and $\exists Z\, A$ are naturally and respectively equivalent to $\forall X\, X$, $\forall X((A \to X) \to (B \to X) \to X)$, $\forall X(\forall z(A \to X) \to X)$ and $\forall X(\forall Z(A \to X) \to X)$.

A nice feature of 2^{nd} order logic: Impredicativity

From $A \wedge B$ it follows that for all propositions X, if $A \to B \to X$ then X.
In a 2^{nd} order logic formula: $\forall X((A \to B \to X) \to X)$.
Conversely, suppose that $\forall X((A \to B \to X) \to X)$. Then in particular for $X = A \wedge B$: $(A \to B \to A \wedge B) \to A \wedge B$, hence $A \wedge B$.
$\rightsquigarrow$ $A \wedge B$ is naturally equivalent to $\forall X((A \to B \to X) \to X)$.

Similarly, $\bot$, $A \vee B$, $\exists z\, A$ and $\exists Z\, A$ are naturally and respectively equivalent to $\forall X\, X$, $\forall X((A \to X) \to (B \to X) \to X)$, $\forall X(\forall z(A \to X) \to X)$ and $\forall X(\forall Z(A \to X) \to X)$.
$\rightsquigarrow$ No need to have $\bot$, $\wedge$, $\vee$, $\exists$ (1^{st} and 2^{nd} order versions) as primitives in the definition of 2^{nd} order classical logic.

A nice feature of 2^{nd} order logic: Impredicativity

From $A \wedge B$ it follows that for all propositions X, if $A \to B \to X$ then X.
In a 2^{nd} order logic formula: $\forall X((A \to B \to X) \to X)$.
Conversely, suppose that $\forall X((A \to B \to X) \to X)$. Then in particular for $X = A \wedge B$: $(A \to B \to A \wedge B) \to A \wedge B$, hence $A \wedge B$.
$\rightsquigarrow$ $A \wedge B$ is naturally equivalent to $\forall X((A \to B \to X) \to X)$.

Similarly, $\bot$, $A \vee B$, $\exists z\, A$ and $\exists Z\, A$ are naturally and respectively equivalent to $\forall X\, X$, $\forall X((A \to X) \to (B \to X) \to X)$, $\forall X(\forall z(A \to X) \to X)$ and $\forall X(\forall Z(A \to X) \to X)$.
$\rightsquigarrow$ No need to have $\bot$, $\wedge$, $\vee$, $\exists$ (1^{st} and 2^{nd} order versions) as primitives in the definition of 2^{nd} order classical logic.

By "naturally", we meant "from natural deductions", i.e. intuitionistically.

A nice feature of 2^{nd} order logic: Impredicativity

From $A \wedge B$ it follows that for all propositions X, if $A \to B \to X$ then X.
In a 2^{nd} order logic formula: $\forall X((A \to B \to X) \to X)$.
Conversely, suppose that $\forall X((A \to B \to X) \to X)$. Then in particular for $X = A \wedge B$: $(A \to B \to A \wedge B) \to A \wedge B$, hence $A \wedge B$.
$\rightsquigarrow$ $A \wedge B$ is naturally equivalent to $\forall X((A \to B \to X) \to X)$.

Similarly, $\bot$, $A \vee B$, $\exists z\, A$ and $\exists Z\, A$ are naturally and respectively equivalent to $\forall X\, X$, $\forall X((A \to X) \to (B \to X) \to X)$, $\forall X(\forall z(A \to X) \to X)$ and $\forall X(\forall Z(A \to X) \to X)$.
$\rightsquigarrow$ No need to have $\bot$, $\wedge$, $\vee$, $\exists$ (1^{st} and 2^{nd} order versions) as primitives in the definition of 2^{nd} order classical logic.

By "naturally", we meant "from natural deductions", i.e. intuitionistically.
$\rightsquigarrow$ No need to have primitives $\bot$, $\wedge$, $\vee$, $\exists$ in the definition of 2^{nd} order intuitionistic logic either.

A nice feature of 2nd order logic: Impredicativity (continued)

Leibniz Principle: "From an equality $t = u$, it follows that t can be replaced by u in every statement without changing its meaning."
Hence if $t = u$ then $\forall X\,(X(t) \to X(u))$.

A nice feature of 2^{nd} order logic: Impredicativity (continued)

Leibniz Principle: "From an equality $t = u$, it follows that t can be replaced by u in every statement without changing its meaning."
Hence if $t = u$ then $\forall X\,(X(t) \to X(u))$.
Conversely, suppose that $\forall X\,(X(t) \to X(u))$. Then in particular for $X(\,\cdot\,) = (t = \,\cdot\,)$, we have: $t = t \to t = u$, hence $t = u$.

A nice feature of 2^{nd} order logic: Impredicativity (continued)

Leibniz Principle: "From an equality $t = u$, it follows that t can be replaced by u in every statement without changing its meaning."
Hence if $t = u$ then $\forall X\,(X(t) \to X(u))$.
Conversely, suppose that $\forall X\,(X(t) \to X(u))$. Then in particular for $X(\,\cdot\,) = (t = \,\cdot\,)$, we have: $t = t \to t = u$, hence $t = u$.
$\rightsquigarrow$ No need to add a predicate symbol for equality in 2^{nd} order (intuitionistic or classical) logic, we only have to adopt the notation:

$$(t = u) \stackrel{\text{def}}{=} \forall X\,(X(t) \to X(u)).$$

A nice feature of 2^{nd} order logic: Impredicativity (continued)

Leibniz Principle: "From an equality $t = u$, it follows that t can be replaced by u in every statement without changing its meaning."
Hence if $t = u$ then $\forall X\,(X(t) \to X(u))$.
Conversely, suppose that $\forall X\,(X(t) \to X(u))$. Then in particular for $X(\,\cdot\,) = (t = \,\cdot\,)$, we have: $t = t \to t = u$, hence $t = u$.
$\rightsquigarrow$ No need to add a predicate symbol for equality in 2^{nd} order (intuitionistic or classical) logic, we only have to adopt the notation:

$$(t = u) \stackrel{\text{def}}{=} \forall X\,(X(t) \to X(u)).$$

Induction Principle: "If u is an integer, then $X(u)$ holds for every statement X such that $\forall z\,(X(x) \to X(sx))$ and $X(0)$."
Hence if $u \in \mathbb{N}$, then $\forall X\,(\forall z\,(X(x) \to X(sx)) \to X(0) \to X(u))$.

A nice feature of 2^{nd} order logic: Impredicativity (continued)

Leibniz Principle: "From an equality $t = u$, it follows that t can be replaced by u in every statement without changing its meaning."
Hence if $t = u$ then $\forall X\,(X(t) \to X(u))$.
Conversely, suppose that $\forall X\,(X(t) \to X(u))$. Then in particular for $X(\,\cdot\,) = (t = \,\cdot\,)$, we have: $t = t \to t = u$, hence $t = u$.
$\rightsquigarrow$ No need to add a predicate symbol for equality in 2^{nd} order (intuitionistic or classical) logic, we only have to adopt the notation:

$$(t = u) \stackrel{\text{def}}{=} \forall X\,(X(t) \to X(u)).$$

Induction Principle: "If u is an integer, then $X(u)$ holds for every statement X such that $\forall z\,(X(x) \to X(sx))$ and $X(0)$."
Hence if $u \in \mathbb{N}$, then $\forall X\,(\forall z\,(X(x) \to X(sx)) \to X(0) \to X(u))$.
Conversely, suppose the latter. Then in particular for $X(\,\cdot\,) = \,\cdot \in \mathbb{N}$: $\forall z\,(z \in \mathbb{N} \to sz \in \mathbb{N}) \to 0 \in \mathbb{N} \to u \in \mathbb{N}$, hence $u \in \mathbb{N}$.

A nice feature of 2^{nd} order logic: Impredicativity (continued)

Leibniz Principle: "From an equality $t = u$, it follows that t can be replaced by u in every statement without changing its meaning."
Hence if $t = u$ then $\forall X\,(X(t) \to X(u))$.
Conversely, suppose that $\forall X\,(X(t) \to X(u))$. Then in particular for $X(\,\cdot\,) = (t = \,\cdot\,)$, we have: $t = t \to t = u$, hence $t = u$.
$\rightsquigarrow$ No need to add a predicate symbol for equality in 2^{nd} order (intuitionistic or classical) logic, we only have to adopt the notation:

$$(t = u) \stackrel{\text{def}}{=} \forall X\,(X(t) \to X(u)).$$

Induction Principle: "If u is an integer, then $X(u)$ holds for every statement X such that $\forall z\,(X(x) \to X(sx))$ and $X(0)$."
Hence if $u \in \mathbb{N}$, then $\forall X\,(\forall z\,(X(x) \to X(sx)) \to X(0) \to X(u))$.
Conversely, suppose the latter. Then in particular for $X(\,\cdot\,) = \,\cdot \in \mathbb{N}$:
$\forall z\,(z \in \mathbb{N} \to sz \in \mathbb{N}) \to 0 \in \mathbb{N} \to u \in \mathbb{N}$, hence $u \in \mathbb{N}$.
$\rightsquigarrow$ No need to add a predicate symbol for $\mathbb{N}$ in 2^{nd} order (intuitionistic or classical) logic, we only have to adopt the notation:

$$N(u) \stackrel{\text{def}}{=} \forall X\,(\forall z\,(X(x) \to X(sx)) \to X(0) \to X(u)).$$

Just 2×2 logics to come!

Although our concern would rather be 1^{st} order logic, in the rest of this course, we will only consider:

- 2^{nd} order logic, just because it has a lighter syntax. Its formulas are indeed generated by: $A = X(u_1, \ldots, u_n) \mid A \to A \mid \forall x\, A \mid \forall X\, A$ hence only 3×2 logical rules in LK or ND.
- Minimal logic, because it is the *kernel* of all proof $\to$ program systems. Once this kernel is extended to classical logic, full logic follows straightfowardly.

Formal definition of 2nd order ND: Formulas & predicate substitution

Individual terms: as usual from individual variables and function symbols.
Formulas: $A = X(u_1, \ldots, u_n)^\dagger \mid A \to A \mid \forall x\, A \mid \forall X\, A$

Definition of $A[F/X(x_1, \ldots, x_n)]^\ddagger$ ($A[F/X\vec{x}]$ for short):

- $X(u_1, \ldots, u_n)[F/X\vec{x}] = F[u_1/x_1, \ldots, u_n/x_n]$
- $(A \to B)[F/X\vec{x}] = A[F/X\vec{x}] \to B[F/X\vec{x}]$
- $(\forall z\, A)[F/X\vec{x}] = \forall z\, A[F/X\vec{x}]$
- $(\forall Z\, A)[F/X\vec{x}] = \forall Z\, A[F/X\vec{x}]$

† For any 2nd order variable X of arity n.

‡ This notation $[F/X(x_1, \ldots, x_n)]$ binds the free occurences of $x_1, \ldots, x_n$ in the formula F.

Formal definition of 2^{nd} order ND: Rules

I. Axiom rule:

$$A^x \vdash A$$

II. Logical rules:

$$\frac{\Gamma \vdash A \to B \quad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B} \to e \qquad \frac{\Gamma, A^x \vdash B}{\Gamma \vdash A \to B} \to i$$

$$\frac{\Gamma \vdash \forall x\, A}{\Gamma \vdash A[t/x]} \forall e \qquad \frac{\Gamma \vdash A}{\Gamma \vdash \forall x\, A} \forall i\,{}^{\dagger}$$

$$\frac{\Gamma \vdash \forall X\, A}{\Gamma \vdash A[F\vec{x}/X\vec{x}]} \forall e \qquad \frac{\Gamma \vdash A}{\Gamma \vdash \forall X\, A} \forall i\,{}^{\ddagger}$$

III. Structural rule:

$$\frac{\Gamma \vdash C}{\Gamma, \Gamma' \vdash C}\, w$$

† Only if x does not occur free in Γ.

‡ Only if X does not occur free in Γ.

Formal definition of 2nd order ND: Normalization steps

- $$\dfrac{\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma, A^x \vdash B\end{array}}{\Gamma \vdash A \to B}\to i \qquad \begin{array}{c}\vdots\,\Pi'\\ \Gamma' \vdash A\end{array}}{\Gamma, \Gamma' \vdash B}\to e \longrightarrow \dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma, A^x \vdash B\end{array} \overset{x}{\curvearrowleft} \qquad \begin{array}{c}\vdots\,\Pi'\\ \Gamma' \vdash A\end{array}}{\Gamma, \Gamma' \vdash B}$$

  where the right hand side is defined as previously.

- $$\dfrac{\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma \vdash A\end{array}}{\Gamma \vdash \forall x\, A}\to i}{\Gamma \vdash A[t/x]}\to e \longrightarrow \begin{array}{c}\vdots\,\Pi[t/x]\\ \Gamma \vdash A[t/x]\end{array}$$

  where $\Pi[t/x]$ denotes the proof obtained by replacing every free occurrence of x by t in Π.

- $$\dfrac{\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma \vdash A\end{array}}{\Gamma \vdash \forall X\, A}\to i}{\Gamma \vdash A[F/X\vec{x}]}\to e \longrightarrow \begin{array}{c}\vdots\,\Pi[F/X\vec{x}]\\ \Gamma \vdash A[F/X\vec{x}]\end{array}$$

  where $\Pi[F/X\vec{x}]$ denotes the proof obtained by replacing every formula C by $C[F/X\vec{x}]$ in Π.

Formal definition of minimal ND:

In the same way with no rule about quantifiers and no 1^{st} order.

2^{nd} order & minimal classical logics

We have just given in terms of ND a definition of 2^{nd} order & minimal intuitionistic logics.
A lazy way to extend them to classical logics is to add to them Peirce Law as an extra axiom scheme:

$$\vdash ((A \to B) \to A) \to A$$

$\rightsquigarrow$ It then allows to prove every classical statement in both logics,
. . . but it also ruins the normalization procedure!

A few words about Peirce Law $((A \to B) \to A) \to A$

Peirce Law may look obscure at first sight. It is actually an adaptation of the classical *reductio ad absurdum* $\neg\neg A \to A$ (i.e. $((A \to \bot) \to \bot) \to A$) to minimal logic where the rule of *intuitionistic absurdity* does not apply and where $\bot$ may just be used as any other propositional symbol:

- *Reductio ad absurdum* is first modified into the intuitionistically equivalent formula $((A \to \bot) \to A) \to A$.
- In this way, we now may replace the remaining occurrence of $\bot$ by any proposition B without loosing the classical validity[†] of the formula: $((A \to B) \to A) \to A$.

Therefore, Peirce Law is just a tricky generalization of the *reductio ad absurdum* principle expressible in the poorer language of minimal logic.

[†] Without the first modification, we would get the formula $((A \to B) \to \bot) \to A$ which is not classically valid because we must view $\bot$ as *any* propositional symbol whereas $((A \to B) \to C) \to A$ is not a tautology.

Exercise 1: Extending minimal intuitionistic logic to minimal classical logic

Another way to extend minimal intuitionistic logic to minimal classical logic is to adapt the *excluded middle* principle (instead of *reductio ad absurdum* yielding Peirce Law). First give the *excluded middle* principle the following form: $(C \to A) \to (\neg C \to A) \to A$ ("if we proved A from the assumption C and then from the assumption $\neg C$, then we really proved A from no assumption at all"), then replace the unique occurrence of $\bot$ by any proposition B:

$$(C \to A) \to ((C \to B) \to A) \to A \qquad \text{(mEM)}$$

The goal of this exercise is to show that this axiom scheme yields the same as Peirce Law.

1. Write down all the rules of ND for minimal intuitionistic logic. Call mND this proof system.
2. Give a formal proof of Peirce Law in mND+(mEM), i.e. mND with the additional axiom scheme: $\vdash (C \to A) \to ((C \to B) \to A) \to A$.
3. Give a formal proof in mND of the formula: $(((A \to B) \to A) \to A) \to (C \to A) \to ((C \to B) \to A) \to A$.
4. Show that every formula of minimal logic is provable in mND+(mEM) iff it is provable in mND+(Peirce Law).

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$A^x \vdash \quad A \qquad \frac{\Gamma \vdash \quad A}{\Gamma, \Gamma' \vdash \quad A}\, w$$

$$\frac{\Gamma, \quad A^x \vdash \quad B}{\Gamma \vdash \qquad A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash \quad A \to B \qquad \Gamma' \vdash \quad A}{\Gamma, \Gamma' \vdash \qquad B}\, {\to}e$$

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$A^x \vdash x : A \qquad \frac{\Gamma \vdash \quad A}{\Gamma, \Gamma' \vdash \quad A}\, w$$

$$\frac{\Gamma, \quad A^x \vdash \quad B}{\Gamma \vdash \quad A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash \quad A \to B \qquad \Gamma' \vdash \quad A}{\Gamma, \Gamma' \vdash \quad B}\, {\to}e$$

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$A^x \vdash x : A \qquad \frac{\Gamma \vdash \quad A}{\Gamma, \Gamma' \vdash \quad A}\, w$$

$$\frac{\Gamma, \quad A^x \vdash t : B}{\Gamma \vdash \quad A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash \quad A \to B \qquad \Gamma' \vdash \quad A}{\Gamma, \Gamma' \vdash \quad B}\, {\to}e$$

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$A^x \vdash x : A \qquad \frac{\Gamma \vdash \quad A}{\Gamma, \Gamma' \vdash \quad A}\, w$$

$$\frac{\Gamma, \quad A^x \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash \quad A \to B \qquad \Gamma' \vdash \quad A}{\Gamma, \Gamma' \vdash \quad B}\, {\to}e$$

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$A^x \vdash x : A \qquad \frac{\Gamma \vdash \quad A}{\Gamma, \Gamma' \vdash \quad A}\, w$$

$$\frac{\Gamma, \quad A^x \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash \qquad B}\, {\to}e$$

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$A^x \vdash x : A \qquad \frac{\Gamma \vdash \quad A}{\Gamma, \Gamma' \vdash \quad A}\, w$$

$$\frac{\Gamma, \quad A^x \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash (tu) : B}\, {\to}e$$

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$A^x \vdash x : A \qquad \frac{\Gamma \vdash t : A}{\Gamma, \Gamma' \vdash t : A}\, w$$

$$\frac{\Gamma, \quad A^x \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash (tu) : B}\, {\to}e$$

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$A^x \vdash x : A \qquad \frac{\Gamma \vdash t : A}{\Gamma, \Gamma' \vdash t : A}\, w$$

$$\frac{\Gamma,\quad A^x \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash (tu) : B}\, {\to}e$$

The term t is called *λ-term extracted from the proof*.

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$A^x \vdash x : A \qquad \frac{\Gamma \vdash t : A}{\Gamma, \Gamma' \vdash t : A}\, w$$

$$\frac{\Gamma,\quad A^x \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash (tu) : B}\, {\to}e$$

The term t is called *λ-term extracted from the proof*.
Let us note the l.h.s. $\Gamma = A_1^{x_1}, \ldots, A_n^{x_n}$ of the sequents in the same way: $\Gamma = x_1 : A_1, \ldots, x_n : A_n$.

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$x : A \vdash x : A \qquad \frac{\Gamma \vdash t : A}{\Gamma, \Gamma' \vdash t : A}\, w$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash (tu) : B}\, {\to}e$$

The term t is called *λ-term extracted from the proof*.
Let us note the l.h.s. $\Gamma = A_1^{x_1}, \ldots, A_n^{x_n}$ of the sequents in the same way: $\Gamma = x_1 : A_1, \ldots, x_n : A_n$.

A stenography of the proofs

Proof in minimal ND of conclusion $\Gamma \vdash A \rightsquigarrow \Gamma \vdash t : A$, where t encodes the complete structure of the proof (order of the inferences rules, places of the abstracted assumptions), but not its formulas.

$$x : A \vdash x : A \qquad \frac{\Gamma \vdash t : A}{\Gamma, \Gamma' \vdash t : A}\, w$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash (tu) : B}\, {\to}e$$

The term t is called *λ-term extracted from the proof*.
Let us note the l.h.s. $\Gamma = A_1^{x_1}, \ldots, A_n^{x_n}$ of the sequents in the same way: $\Gamma = x_1 : A_1, \ldots, x_n : A_n$. Γ is then called *context of the λ-term t*.

The effect of normalisation on the λ-terms

Recall that $\dfrac{\dfrac{\vdots\,\Pi}{\Gamma, x:A \vdash B}}{\dfrac{\Gamma \vdash A \to B \qquad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B}} \quad \vdots\,\Pi' \longrightarrow \dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma, x:A \vdash B \overset{x}{\curvearrowleft} \qquad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B}$

If say $\vdots\,\Pi$ $\Gamma, x:A \vdash t:B$ and $\vdots\,\Pi'$ $\Gamma' \vdash u:A$, let $t[x:=u]$ denote the λ-term of the r.h.s.:

$$\dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma, x:A \vdash t:B \overset{x}{\curvearrowleft} \qquad \Gamma' \vdash u:A}{\Gamma, \Gamma' \vdash t[x:=u]:B}$$

The effect of normalisation on the λ-terms

$$\dfrac{\dfrac{}{x:A\vdash x:A}\,{\scriptstyle x}\quad \begin{array}{c}\vdots\,\Pi'\\ \Gamma'\vdash u:A\end{array}}{\Gamma'\vdash x[x:=u]:A}\quad\overset{\text{def}}{=\!=}\quad \begin{array}{c}\vdots\,\Pi'\\ \Gamma'\vdash u:A\end{array}$$

hence $x[x:=u]=u$

$$\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma\vdash t:C\end{array}\quad{\scriptstyle x}\quad\begin{array}{c}\vdots\,\Pi'\\ \Gamma'\vdash u:A\end{array}}{\Gamma,\Gamma'\vdash t[x:=u]:C}\quad\overset{\text{def}}{=\!=}\quad\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma\vdash t:C\end{array}}{\Gamma,\Gamma'\vdash t:C}$$

if $x:A\notin\Gamma$, hence $t[x:=u]=t$ if $x\notin t$

and otherwise:

$$\dfrac{\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma,x:A,z:C\vdash t:D\end{array}}{\Gamma,x:A\vdash\lambda z\,t:C\to D}\,{\scriptstyle x}\quad\begin{array}{c}\vdots\,\Pi'\\ \Gamma'\vdash u:A\end{array}}{\Gamma,\Gamma'\vdash(\lambda z\,t)[x:=u]:C\to D}\quad\overset{\text{def}}{=\!=}\quad\dfrac{\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma,x:A,z:C\vdash t:D\end{array}\quad{\scriptstyle x}\quad\begin{array}{c}\vdots\,\Pi'\\ \Gamma'\vdash u:A\end{array}}{\Gamma,z:C,\Gamma'\vdash t[x:=u]:D}}{\Gamma,\Gamma'\vdash\lambda z\,(t[x:=u]):C\to D}$$

hence $(\lambda z\,t)[x:=u]=\lambda z\,(t[x:=u])$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\,\Pi_1\\ \Gamma_1\vdash t:C\to D\end{array}\quad\begin{array}{c}\vdots\,\Pi_2\\ \Gamma_2\vdash t':C\end{array}}{\Gamma,x:A\vdash tt':C}\,{\scriptstyle x}\quad\begin{array}{c}\vdots\,\Pi'\\ \Gamma'\vdash u:A\end{array}}{\Gamma,\Gamma'\vdash(tt')[x:=u]:C}\quad\overset{\text{def}}{=\!=}\quad\dfrac{\begin{array}{c}\vdots\,\Pi_1\\ \Gamma_1\vdash t:C\to D\end{array}\quad{\scriptstyle x}\quad\begin{array}{c}\vdots\,\Pi'\\ \Gamma'\vdash u:A\end{array}\qquad\begin{array}{c}\vdots\,\Pi_2\\ \Gamma_2\vdash t':C\end{array}\quad{\scriptstyle x}\quad\begin{array}{c}\vdots\,\Pi'\\ \Gamma'\vdash u:A\end{array}}{\Gamma,\Gamma'\vdash t[x:=u]\,t'[x:=u]:C}$$

hence $(tt')[x:=u]=t[x:=u]\,t'[x:=u]$

The effect of normalisation on the λ-terms

Summing up:

- $x[x := u] = u$
- $t[x := u] = t$, if $x \notin t$
- $(\lambda z\, t)[x := u] = \lambda z\, (t[x := u])$
- $(tt')[x := u] = t[x := u]\, t'[x := u]$

$\rightsquigarrow$ $t[x := u]$ is just the λ-term t after replacing x by u!

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma, x : A \vdash t : B}}{\Gamma \vdash \lambda x\, t : A \to B} \quad \dfrac{\vdots\,\Pi'}{\Gamma' \vdash u : A}}{\Gamma, \Gamma' \vdash (\lambda x\, t)u : B} \longrightarrow \dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma, x : A \vdash B \overset{x}{\curvearrowleft} \quad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash t[x := u] : B}$$

$\rightsquigarrow$ The normalization corresponds to a sequence of rewritings of the form:

$$(\lambda x\, t)u \to_\beta t[x := u]$$

which are called β-reductions.

Summing up

λ-terms:

$$t \;=\; x\ (\textit{variable}) \;\big|\; \lambda x\, t\ (\textit{abstraction}) \;\big|\; tt\ (\textit{application})$$

Precedence: $\textit{application} > \lambda$

Reduction rule:

$$(\lambda x\, t)u \to_\beta t[x := u]$$

By this single line, we actually mean that $t_1 \to_\beta t_2$ whenever t_2 is obtained by replacing a subterm of t_1 of the form $(\lambda x\, t)u$ (corresponding to a subproof ending with a cut) by $t[x := u]$.

$$\twoheadrightarrow_\beta \stackrel{\text{def}}{=} \text{reflexive \& transitive closure of } \to_\beta.$$
$$=_\beta \stackrel{\text{def}}{=} \text{equivalence relation generated by } \twoheadrightarrow_\beta.$$

Exercise 2

Write down the λ-term extracted from the proof in minimal ND of

$$(((A \to B) \to A) \to A) \to (C \to A) \to ((C \to B) \to A) \to A$$

obtained at question 3 of Exercise 1 and normalize it (in case it is not already a normal λ-term).

The untyped λ-calculus on its own

λ-calculus (Church, 1932) was already living its own life before ND was invented by Gentzen in 1934!

λ-calculus was first conceived as a formal system about functions.

- not functions in the so called Dedekind style (= defined by an arbitrary set theoretical graph): extensional point of view
- but functions as something that can be computed accordingly to some algorithm: intensional point of view.

The untyped λ-calculus as a naive function theory (v. naive set theory)

Just as in set theory every object is a set (of sets!), every λ-term denotes some function . . . to be applied to other functions!

Examples:

- $\mathbf{I} \stackrel{\text{def}}{=} \lambda x\, x$, a universal identity function: $\mathbf{I}F =_\beta F$ for *all* F.
- $\mathbf{B} \stackrel{\text{def}}{=} \lambda f \lambda g \lambda x\, f(gx)$, a universal composition function: $\mathbf{B}F\,G =_\beta F \circ G \stackrel{\text{def}}{=} \lambda x\, F(Gx)$
- $\overline{n} \stackrel{\text{def}}{=} \lambda f \lambda x\, \underbrace{f(f(\ldots(f}_{n}\, x)))$ ($= \lambda f \lambda x\, f^n x$ for short), iterating n times: $\overline{n}\,F =_\beta \underbrace{F \circ \cdots \circ F}_{n}$.

  $\overline{n}$ is the natural incarnation of the numeral n within λ-calculus and is called a *Church numeral*.

Every recursive function $f : \mathbb{N}^k \to \mathbb{N}$ can be represented by a λ-term F: for all $n_1, \ldots, n_k$, we have $F\overline{n}_1 \ldots \overline{n}_k =_\beta \overline{f(n_1, \ldots, n_k)}$ (iff $f(n_1, \ldots, n_k)$ is defined, of course).

Some extensionality in untyped λ-calculus

The following infinitary inference rule (expressing some purely syntactical extensionality):

$$\forall u \;\; tu = t'u \;\Rightarrow\; t = t'$$

is actually equivalent to the axiom scheme:

$$\lambda x\, tx = t \qquad \text{where } x \notin t.$$

This leads to the following definitions.
Reduction rule:

$$\lambda x\, tx \to_\eta t \qquad \text{only if } x \notin t.$$

(By the latter, we mean as for $\to_\beta$ that $t_1 \to_\eta t_2$ whenever t_2 is obtained by replacing a subterm of t_1 of the form $\lambda x\, tx$ by t.)

$\to_{\beta\eta} \stackrel{\text{def}}{=} \to_\beta \cup \to_\eta$.
$\twoheadrightarrow_{\beta\eta} \stackrel{\text{def}}{=}$ reflexive & transitive closure of $\to_{\beta\eta}$.
$=_{\beta\eta} \stackrel{\text{def}}{=}$ equivalence relation generated by $\twoheadrightarrow_{\beta\eta}$.

About properties of untyped λ-calculus

CR Property If $t \twoheadrightarrow_{\beta(\eta)} t_1$ and $t \twoheadrightarrow_{\beta(\eta)} t_2$ then there is t' such that $t_1 \twoheadrightarrow_{\beta(\eta)} t'$ and $t_2 \twoheadrightarrow_{\beta(\eta)} t'$.

. . . But we have no SN Property for the untyped λ-calculus:

$$(\lambda x\, xx)(\lambda x\, xx) \to_\beta (\lambda x\, xx)(\lambda x\, xx) \to_\beta \dots$$

This term endlessly rewriting into itself is the λ-equivalent of Russell's paradox (about the set of the x's such that $x \notin x$: does it belong to itself?).

Back to logic: Typing derivation systems (minimal logic)

Typing rules:

$$x : A \vdash x : A$$

$$\frac{\Gamma \vdash C}{\Gamma, \Gamma' \vdash C}\ w$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\ \to i \qquad \frac{\Gamma \vdash t : A \to B \quad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash tu : B}\ \to e$$

SN Property If $\Gamma \vdash t : A$ can be derived from the above typing rules then every reduction sequence $t \to_{\beta\eta} t_1 \to_{\beta\eta} t_2 \to_{\beta\eta} \ldots$ is finite.

Subject reduction Property If $\Gamma \vdash t : A$ is derivable from the above typing rules and $t \twoheadrightarrow_{\beta\eta} t'$ then $\Gamma \vdash t : A$ is also derivable.

Back to logic: Typing derivation systems (2^{nd} order logic)

Typing rules:

$$x : A \vdash x : A \qquad \frac{\Gamma \vdash C}{\Gamma, \Gamma' \vdash C}\, w$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\, \to i \qquad \frac{\Gamma \vdash t : A \to B \quad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash tu : B}\, \to e$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash t : \forall x\, A}\, \forall i^{\dagger} \qquad \frac{\Gamma \vdash t : \forall x\, A}{\Gamma \vdash t : A[u/x]}\, \forall e$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash t : \forall X\, A}\, \forall i^{\dagger} \qquad \frac{\Gamma \vdash t : \forall X\, A}{\Gamma \vdash t : A[F/X\vec{x}]}\, \forall e$$

† only if x (resp. X) $\notin \Gamma$

SN Property If $\Gamma \vdash t : A$ can be derived from the above typing rules then every reduction sequence $t \to_{\beta\eta} t_1 \to_{\beta\eta} t_2 \to_{\beta\eta} \ldots$ is finite.

Subject reduction Property If $\Gamma \vdash t : A$ is derivable from the above typing rules and $t \twoheadrightarrow_{\beta} t'$ then $\Gamma \vdash t : A$ is also derivable.

. . . λ-calculus had itself an elder brother!

In the early 20's, Shonfinkel had already conceived Combinatory Logic (CL for short), a calculus close to λ-calculus where two constants K, S play the role of the abstractor λ.

Syntax of CL:

$$t = x \mid K \mid S \mid tt$$

Reduction rules:

$$Ktu \longrightarrow t$$
$$Stuv \longrightarrow tv(uv)$$

As usual, $t_1 \longrightarrow t_2$ means that t_2 is obtained by replacing a subterm of t_1 of the form of the l.h.s. of a rule by its r.h.s. and $\twoheadrightarrow$ is the reflexive & transitive closure of $\longrightarrow$.
An *applicative combination of say* $u_1, \ldots, u_n$ is a term $t = u_1 \mid \cdots \mid u_n \mid tt$

Combinatorial completeness For every applicative combination t of $x_1, \ldots, x_n$, there is a closed term C of CL (i.e. an applicative combination of K, S) such that:

$$Cx_1 \ldots x_n \twoheadrightarrow t$$

Correspondence: CL ↔ λ-calculus

The constants K, S with their reduction rules $Kxy \to x$, $Sxyz \to xz(yz)$ are naturally played by the λ-terms:

$$\mathbf{K} \stackrel{\text{def}}{=} \lambda x \lambda y\ x \qquad \mathbf{S} \stackrel{\text{def}}{=} \lambda x \lambda y \lambda z\ xz(yz)$$

If $t \twoheadrightarrow u$ in CL, then $t[K := \mathbf{K}, S := \mathbf{S}] \twoheadrightarrow_\beta u[K := \mathbf{K}, S := \mathbf{S}]$ in λ-calculus.

There is a converse translation $\stackrel{\{\ \}}{\longrightarrow}$ of λ-calculus into CL inductively given by:

- $\{x\} = x$
- $\{tu\} = \{t\}\{u\}$
- $\{\lambda x\, t\} = \lambda^* x\, \{t\}$

where for any term t of CL, $\lambda^* x\, t$ is defined by an inner induction as:

- $\lambda^* x\, x = SKK$
- $\lambda^* x\, t = Kt$ if $x \notin t$
- $\lambda^* x\, tu = S(\lambda^* x\, t)(\lambda^* x\, u)$

Proposition For every λ-term t: $\{t\}[K := \mathbf{K}, S := \mathbf{S}] \twoheadrightarrow_\beta t$.

Exercise 3

Prove the **Proposition** of the previous slide.

Curry-Howard correspondence

Since λ-calculus was invented before ND, it had to be noticed that the former could have been extracted from the latter as we did here.

Historically, this was remarked in the 50's by H. Curry who was studying CL, which actually is the computional counterpart of a Hilbert style system for intuitionistic minimal logic.

Computational systems	Formal proof systems
CL	Hilbert style system
application	Modus Ponens inference rule
constants K, S	axioms $A \to B \to A$, $(A \to B \to C) \to (A \to B) \to A \to C$
term SKK of CL	proof of $A \to A$ from the just above axioms
closed λ-terms $\mathbf{K}, \mathbf{S}$	proofs of the same axioms in ND
λ-abstraction	$\to$-introduction rule of ND
λ-calculus	ND

Exercise 4

Justify the 7 line table of the previous slide. The explanations can be totally inexistent for lines 2, 6 and 7 of the table (which have already been treated here) but should be accurate about the other lines.

In particular:

- A typing derivation system for CL (and minimal logic) should be described to justify line 1 (about CL & Hilbert style) and used to justify line 4 (about the term SKK).
- Line 5 should be justified by typing the closed λ-terms **K**, **S** with their corresponding formulas as types in the minimal logic typing derivation system.

The platonician world

$\mathcal{U}$ = "mathematical universe" which may contain *anything*:
integers, rational numbers, ordinals, trees ($\rightsquigarrow$ lists, matrices . . .) . . .
We may only access to $\mathcal{U}$ through 1st order syntax:

- constants & function symbols to denote its elements,
- equations between individual terms to give the intended meaning of these symbols.

In general these function symbols denote partial functions
$\rightsquigarrow$ some individual terms are meaningless.

Very simple example:

- 1st order symbols: 0, s (successor), p (predecessor), $+$, $-$, . and !.
- equations:

$$\begin{bmatrix} psx = x \\ spx = x \end{bmatrix} \quad \begin{bmatrix} x+0=x \\ x+sy=s(x+y) \\ (x+y)+z=x+(y+z) \\ x+y=y+x \end{bmatrix} \quad \begin{bmatrix} x-0=x \\ x-sy=p(x-y) \\ (x+y)-z=x+(y-z) \\ x-x=0 \end{bmatrix} \quad \begin{bmatrix} x.0=0 \\ x.sy=x.y+x \end{bmatrix} \quad \begin{bmatrix} 0!=s0 \\ x!=x.(px)! \end{bmatrix}$$

The meaningful terms denote the elements of $\mathbb{Z}$.
E.g. the term $(p0)!$ is meaningless.

The platonician world: Formally

$\mathcal{L}$: set of function symbols (of fixed arities $\geqslant 0$).
$\mathcal{E}$: set of equations built from $\mathcal{L}$ and variables.
$\rightsquigarrow$ $\mathcal{C}$: set of the *closed* terms built from $\mathcal{L}$.
$\rightsquigarrow \sim$: equivalence relation on $\mathcal{C}$ generated by:

- $t \sim u$ whenever $t = u$ is an equation of $\mathcal{E}$ in which every variable has been replaced by a closed term of $\mathcal{C}$.
- $t_1 \sim u_1, \ldots, t_n \sim u_n \Rightarrow f(t_1, \ldots, t_n) \sim f(u_1, \ldots, u_n)$ for every function symbol $f \in \mathcal{L}$ of arity n.

$\rightsquigarrow \mathcal{U} = \mathcal{C}/\sim$.
As usual, we represent the elements of $\mathcal{U}$ by anyone of their terms.
$\mathcal{U}$ is obviously interesting only if it does not collapse into a single class.

The same very simple example, formally:

- $\mathcal{L} = \{0, s, p, +, -, ., !\}$
- $\mathcal{E}$: the same set of equation as previously given.

$\rightsquigarrow$ $\mathcal{U}$ is an abelian group containing $\mathbb{Z}$ with two additional complex operations $., !$. Multiplication is well defined on $\mathbb{Z}$, but does not even make $\mathcal{U}$ a ring because $0.(pp0)! \neq 0$ in $\mathcal{U}$.

The platonician world considered in the rest of this lecture

$\mathcal{L}$ contains at least the function symbols of the previous very simple example and a function symbol for every (possibly partial) recursive function (on the positive integers).

$\mathcal{E}$ contains at least the equations of the previous very simple example and equations defining every recursive function (in a sensible way, i.e. such that we do not get $s^k 0 \sim s^n 0$ for some $k \neq n$).

Every element $s^n 0$ of $\mathcal{U}$ will simply be denoted by n.

Realizability semantics

Notations:
Λ = set of the λ-terms, $\mathcal{P}_\beta(\Lambda)$ = set of the parts of Λ closed under $=_\beta$.
For all $K, L \in \mathcal{P}_\beta(\Lambda)$, $K \to L \stackrel{\text{def}}{=} \{t \in \Lambda\ ;\ \forall u \in K \quad tu \in L\} \in \mathcal{P}_\beta(\Lambda)$.

Every 2nd order formula is going to be interpreted by an element of $\mathcal{P}_\beta(\Lambda)$.
For convenience in the definition of this interpretation, for every map $P \in \mathcal{P}_\beta(\Lambda)^{\mathcal{U}^n}$ $(n \geqslant 0)$ we add to the language a predicate symbol of arity n that we denote the same.

Every closed 2nd order formula A built from these predicate symbols and the individual terms in $\mathcal{C}$ then is interpreted by a set $|A| \in \mathcal{P}_\beta(\Lambda)$ as follows:

- $|P(\vec{u})| = P(\vec{u})$ for every $P \in \mathcal{P}_\beta(\Lambda)^{\mathcal{U}^n}$
- $|A \to B| = |A| \to |B|$
- $|\forall x\, A| = \bigcap_{u \in \mathcal{U}} |A[u/x]|$
- $|\forall X^n\, A| = \bigcap_{P \in \mathcal{P}_\beta(\Lambda)^{\mathcal{U}^n}} |A[X := P]|$

Data correctness

Recall that $N(u) \stackrel{\text{def}}{=} \forall X\,(\forall z\,(X(z) \to X(sz)) \to X(0) \to X(u))$.

Correctness of numerical data If for any $t \in \Lambda$ and $u \in \mathcal{C}$:
$t \in |N(u)|$, then there is $n \in \mathbb{N}$ such that $u = n$ in $\mathcal{U}$ and $t =_{\beta\eta} \overline{n}$.

Proof Let $P \in \mathcal{P}_\beta(\Lambda)^{\mathcal{U}}$ be defined by: $r \in P(n) \stackrel{\text{def}}{\Leftrightarrow} r =_\beta f^n x$ and
$P(w) = \varnothing$ for every $w \in \mathcal{U} \setminus \mathbb{N}$.
For all $w \in \mathcal{U}$: $\forall r \in |P(w)|$ $\quad fr \in |P(sw)| \rightsquigarrow f \in |P(w) \to P(sw)|$
$\rightsquigarrow f \in |\forall z\,(P(z) \to P(sz))|$.
By assumption: $t \in |\forall z\,(P(z) \to P(sz)) \to (P(0) \to P(u))|$
$\rightsquigarrow tf \in |P(0) \to P(u)|$. $\quad$ Moreover $x \in |P(0)|$,
$\rightsquigarrow tfx \in |P(u)|$.
$\rightsquigarrow$ By definition of P, there is n such that $u = n$ in $\mathcal{U}$ and $tfx =_\beta f^n x$
$\rightsquigarrow t =_\eta \lambda f\, tf =_\eta \lambda f \lambda x\, tfx =_\beta \lambda f \lambda x\, f^n x = \overline{n}$.

□

Substitutions

Let us call *(2^{nd} order formula) closure* an application $\overline{\cdot}$ mapping

- every individual variable x to some closed individual term $\overline{x} \in \mathcal{C}$
- every predicate variable X of arity say n to some $\overline{X} \in \mathcal{P}_\beta(\Lambda)^{\mathcal{U}^n}$

For every 2^{nd} order formula A, $\overline{A}$ then denotes the closed formula obtained by substituting $\overline{x}$ for every free individual variable x of A and $\overline{X}$ for every free predicate variable X.

Substitution lemma For any closure $\overline{\cdot}$, all $P \in \mathcal{P}_\beta(\Lambda)^{\mathcal{U}^n}$ defined by $P(\vec{u}) = \left|\overline{F[\vec{u}/\vec{x}]}\right|$ where F is any 2^{nd} order formula, then

$$\left|\overline{A[F/X\vec{x}]}\right| = \left|\overline{A[X := P]}\right|$$

Proof By a straightforward induction on the 2^{nd} order formula A. $\square$

The engine

Typing rules:

$$x : A \vdash x : A \qquad \frac{\Gamma \vdash t : A}{\Gamma, \Gamma' \vdash t : A}\, w$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x\, t : A \to B}\, \to i \qquad \frac{\Gamma \vdash t : A \to B \quad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash tu : B}\, \to e$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash t : \forall x\, A}\, \forall i^{\dagger} \qquad \frac{\Gamma \vdash t : \forall x\, A}{\Gamma \vdash t : A[u/x]}\, \forall e$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash t : \forall X\, A}\, \forall i^{\dagger} \qquad \frac{\Gamma \vdash t : \forall X\, A}{\Gamma \vdash t : A[F/X\vec{x}]}\, \forall e$$

† only if x (resp. X) $\notin \Gamma$

Adequacy lemma For any closure $\overline{\cdot}$ and all $t_1 \in \left|\overline{C_1}\right|, \ldots, t_n \in \left|\overline{C_n}\right|$:

$$x_1 : C_1, \ldots, x_n : C_n \vdash t : A \;\Rightarrow\; t[x_1 := t_1, \ldots, x_n := t_n,] \in \left|\overline{A}\right|$$

Exercise 5

Prove the adequacy lemma of the previous slide by assuming the substitution lemma of the last but one slide.

Program correctness

Adequacy lemma (weak form) For every closed formula A:

$$\vdash t : A \;\Rightarrow\; t \in |A|$$

Correctness of numerical data (recall) If for any $t \in \Lambda$ and $u \in \mathcal{C}$: $t \in |N(u)|$, then there is $k \in \mathbb{N}$ such that $u = k$ in $\mathcal{U}$ and $t =_{\beta\eta} \overline{k}$.

Correctness of the programs int $\longrightarrow$ int

If $\vdash t : \forall x\,(N(x) \to N(f(x)))$ then for all $n \in \mathbb{N}$: $t\,\overline{n} =_{\beta\eta} \overline{f(n)}$.

Proof For all $n \in \mathbb{N}$, $\vdash \overline{n} : N(n)$. Moreover $\vdash t : N(n) \to N(f(n))$, hence $\vdash t\,\overline{n} : N(f(n))$ $\rightsquigarrow$ by the adequacy lemma: $t\,\overline{n} \in |N(f(n))|$ $\rightsquigarrow$ by the correctness of data, there is $k \in \mathbb{N}$ such that $f(n) = k$ in $\mathcal{U}$ and $t\,\overline{n} =_{\beta\eta} \overline{k}$. $\square$

Executing the programs

Undeterministic execution of the programs

If $\vdash t : \forall x\,(N(x) \to N(f(x)))$ then for all $n \in \mathbb{N}$:

- Program halting. Every rewrite sequence $t\,\overline{n} \to_{\beta\eta} u_1 \to_{\beta\eta} u_2 \to_{\beta\eta} \cdots$ ultimately stops at a λ-term that can no more be rewritten, i.e. a $\beta\eta$-*normal* λ-term r.
- Result correctness. This λ-term r is the result of execution: $r = \overline{f(n)}$ (except in case $f(n) = 1$ where we then have $r = \mathbf{I}$).

Proof The halting property is just SN property applied to $\vdash t\,\overline{n} : N(f(n))$. By the correctness of the program: $\overline{f(n)} =_{\beta\eta} t\,\overline{n} =_{\beta\eta} r$. Therefore by CR property r is the $\beta\eta$-normal form of $\overline{f(n)}$, i.e. the λ-term $\overline{f(n)}$ itself (except for $\overline{f(n)} = \overline{1}$ which has the $\beta\eta$-normal form $\mathbf{I}$).

□

But how to prove $\forall x\,(N(x) \to N(f(x)))$?

Let us call *instance of an equation* $u_1 = u_2 \in \mathcal{E}$ an equation $\tilde{u}_1 = \tilde{u}_2$ where $\tilde{u}_i$ denotes the term obtained from u_i by replacing its variables x with fixed (possibly open) terms $\tilde{x} \in \mathcal{C}$.
For every instance $\tilde{u}_1 = \tilde{u}_2$ of some $u_1 = u_2 \in \mathcal{E}$ and every closure $\overline{\cdot}$ we have $\overline{\tilde{u}_1} = \overline{\tilde{u}_2}$ in $\mathcal{U}$, hence $\left|\overline{A[\tilde{u}_1/x]}\right| = \left|\overline{A[\tilde{u}_2/x]}\right|$. It follows that we may add to the typing system (and we do!) the derivation rules:

$$\frac{\Gamma \vdash t : A[\tilde{u}_1/x]}{\Gamma \vdash t : A[\tilde{u}_2/x]} \qquad \frac{\Gamma \vdash t : A[\tilde{u}_2/x]}{\Gamma \vdash t : A[\tilde{u}_1/x]} \quad \text{for any instance } \tilde{u}_1 = \tilde{u}_2 \text{ of some } u_1 = u_2 \in \mathcal{E}.$$

Equality can also be used within the formulas. Indeed, its usual axioms: reflexivity, symetry, transitivity and $t = u \to A[t/x] \to A[u/x]$ are obtained at once in the system from the definition $(t = u) \stackrel{\text{def}}{=} \forall X\,(X(t) \to X(u))$.

In the same way, the induction principle comes at once from the definition $N(u) \stackrel{\text{def}}{=} \forall X\,(\forall z\,(X(z) \to X(sz)) \to X(0) \to X(u))$.

Exercise 6

1. Derive the typings:

$$\vdash \mathbf{Succ} \ : \ \forall x \, (N(x) \to N(sx))$$
$$\vdash \mathbf{Add} \ : \ \forall x \forall y \, (N(x) \to N(y) \to N(x+y))$$

where $\mathbf{Succ} = \lambda x \lambda f \lambda z \, xf(fz)$ and $\mathbf{Add} = \lambda x \lambda y \lambda f \lambda z \, xf(yfz)$ with the help of the two derivation rules of the previous slide and the set $\mathcal{E}$ of equations of the "very simple example".

2. Check without the help of realizability semantics that for all $n \in \mathbb{N}, p \in \mathbb{N}$:

$$\mathbf{Succ} \, \overline{n} \ \to\!\!\!\!\to_\beta \ \overline{n+1}$$
$$\mathbf{Add} \, \overline{n} \, \overline{p} \ \to\!\!\!\!\to_\beta \ \overline{n+p}$$

An idea

Before defining a λ-calculus out of ND (as we did previously), adding to ND a minimal stuff from LK to get classical logic

An idea

Before defining a λ-calculus out of ND (as we did previously),
adding to ND a minimal stuff from LK to get classical logic,
i.e. to get a proof of Peirce Law.

In LK:

$$\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\,rw}{\vdash A \to B,\ A}\,r\to \qquad A \vdash A}{(A \to B) \to A \vdash A}\,l\to}{\vdash ((A \to B) \to A) \to A}\,r\to$$

3 Big Problems

① A fundamental one: How to carry computations? i.e. how to normalize?

② A technical one: How to know the active formula of a sequent, i.e. the formula used in the next rule? E.g. if t expresses a proof of $A \vdash B, C$, the assumption A being represented by the variable x, does $\lambda x\, t$ represent a proof of $\vdash A \to B, C$ or a proof of $\vdash B, A \to C$?

③ A semantical one: If the λ-terms no longer express a proof of a well-defined active formula, what do they mean?

① How to normalize this?

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\,rw}{\vdash A \to B,\ A}\,r\to \qquad A \vdash A}{(A \to B) \to A \vdash A}\,l\to}{\vdash ((A \to B) \to A) \to A}\,r\to \qquad \begin{array}{c}\vdots\\ \Pi\end{array}}{\Gamma \vdash A}\;\Gamma \vdash (A \to B) \to A \;\to e$$

$$\rightsquigarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\,rw}{\vdash A \to B,\ A}\,r\to \qquad A \vdash A}{(A \to B) \to A \vdash A}\,l\to}{\vdash ((A \to B) \to A) \to A}\,r\to \qquad \begin{array}{c}\vdots\\ \Pi\end{array}}{\Gamma \vdash A}\;\curvearrowleft ?\; \Gamma \vdash (A \to B) \to A$$

① Normalizing with explicit cut rules ($\rightsquigarrow$ small steps)

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \quad A \vdash A}{(A \to B) \to A \vdash A} \qquad \vdots\ \Pi}{\vdash ((A \to B) \to A) \to A} \quad \Gamma \vdash (A \to B) \to A}{\Gamma \vdash A}$$

① Normalizing with explicit cut rules ($\rightsquigarrow$ small steps)

$$\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \quad A \vdash A}{(A \to B) \to A \vdash A}}{\vdash ((A \to B) \to A) \to A} \qquad \begin{array}{c}\vdots\ \Pi \\ \Gamma \vdash (A \to B) \to A\end{array}}{\Gamma \vdash A}$$

$$\rightsquigarrow$$

$$\dfrac{\begin{array}{c}\vdots\ \Pi \\ \Gamma \vdash (A \to B) \to A\end{array} \qquad \dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \quad A \vdash A}{(A \to B) \to A \vdash A}}{\Gamma \vdash A}\ \textit{s-cut}$$

① Normalizing with explicit cut rules ($\rightsquigarrow$ small steps)

$$\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \quad A \vdash A}{(A \to B) \to A \vdash A}}{\vdash ((A \to B) \to A) \to A} \qquad \begin{array}{c}\vdots\\ \Pi\end{array}}{\Gamma \vdash (A \to B) \to A} \qquad \Gamma \vdash A$$

$$\rightsquigarrow$$

$$\dfrac{\begin{array}{c}\vdots\\ \Pi\end{array} \qquad \dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \quad A \vdash A}{(A \to B) \to A \vdash A}}{\Gamma \vdash (A \to B) \to A \quad \curvearrowleft \quad (A \to B) \to A \vdash A}\ \textit{s-cut}$$

$$\Gamma \vdash A$$

① Normalizing with explicit cut rules ($\rightsquigarrow$ small steps)

$$\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \qquad A \vdash A}{(A \to B) \to A \vdash A}}{\vdash ((A \to B) \to A) \to A} \qquad \begin{array}{c}\vdots\ \Pi \\ \Gamma \vdash (A \to B) \to A\end{array}}{\Gamma \vdash A}$$

$\rightsquigarrow$

$$\dfrac{\begin{array}{c}\vdots\ \Pi \\ \Gamma \vdash (A \to B) \to A\end{array} \qquad \dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \qquad A \vdash A}{(A \to B) \to A \vdash A}}{\Gamma \vdash A}\ \textit{s-cut}$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\ \Pi' \\ \Gamma',\ A \to B \vdash A\end{array}}{\Gamma' \vdash (A \to B) \to A} \qquad \dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \qquad A \vdash A}{(A \to B) \to A \vdash A}}{\Gamma' \vdash A}\ \textit{l-cut}$$

① Normalizing with explicit cut rules ($\rightsquigarrow$ small steps)

$$\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \quad A \vdash A}{(A \to B) \to A \vdash A}}{\vdash ((A \to B) \to A) \to A} \quad \begin{matrix}\vdots\ \Pi \\ \Gamma \vdash (A \to B) \to A\end{matrix}}{\Gamma \vdash A} \rightsquigarrow \dfrac{\begin{matrix}\vdots\ \Pi \\ \Gamma \vdash (A \to B) \to A\end{matrix} \quad \dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \quad A \vdash A}{(A \to B) \to A \vdash A}}{\Gamma \vdash A}\ \textit{s-cut}$$

$$\dfrac{\begin{matrix}\vdots\ \Pi' \\ \Gamma',\ A \to B \vdash A\end{matrix}}{\Gamma' \vdash (A \to B) \to A} \quad \dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \quad A \vdash A}{(A \to B) \to A \vdash A}}{\Gamma' \vdash A}\ \textit{l-cut} \rightsquigarrow \dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \quad \begin{matrix}\vdots\ \Pi' \\ \Gamma',\ A \to B \vdash A\end{matrix}}{\Gamma' \vdash A}\ \textit{s-cut}$$

① Normalizing with explicit cut rules ($\rightsquigarrow$ small steps)

$$\dfrac{\dfrac{\vdots\,\Pi'}{\Gamma',\ A\to B\vdash A}}{\Gamma'\vdash (A\to B)\to A}\qquad \dfrac{\dfrac{\dfrac{A\vdash A}{A\vdash B,\ A}\,w}{\vdash A\to B,\ A}\qquad A\vdash A}{(A\to B)\to A\vdash A}$$
$$\Gamma'\vdash A\ \ l\text{-}cut$$

$$\rightsquigarrow$$

$$\dfrac{\dfrac{\dfrac{A\vdash A}{A\vdash B,\ A}\,w}{\vdash A\to B,\ A}\qquad \dfrac{\vdots\,\Pi'}{\Gamma',\ A\to B\vdash A}}{\Gamma'\vdash A}\ s\text{-}cut$$

① Normalizing with explicit cut rules ($\rightsquigarrow$ small steps)

$$\dfrac{\dfrac{\vdots\,\Pi'}{\Gamma',\ A\to B\vdash A}}{\Gamma'\vdash (A\to B)\to A}\qquad \dfrac{\dfrac{\dfrac{A\vdash A}{A\vdash B,\ A}\,w}{\vdash A\to B,\ A}\qquad A\vdash A}{(A\to B)\to A\vdash A}$$
$$\Gamma'\vdash A\ \ l\text{-}cut$$

$$\rightsquigarrow$$

$$\dfrac{\dfrac{\dfrac{A\vdash A}{A\vdash B,\ A}\,w}{\vdash A\to B,\ A}\qquad \vdots\,\Pi'\quad \Gamma',\ A\to B\vdash A}{\Gamma'\vdash A}\ s\text{-}cut$$

① Normalizing with explicit cut rules ($\rightsquigarrow$ small steps)

$$\dfrac{\dfrac{\vdots\,\Pi'}{\Gamma',\ A\to B\vdash A}}{\Gamma'\vdash (A\to B)\to A}\qquad \dfrac{\dfrac{\dfrac{A\vdash A}{A\vdash B,\ A}\,w}{\vdash A\to B,\ A}\qquad A\vdash A}{(A\to B)\to A\vdash A}$$
$$\Gamma'\vdash A\ \ l\text{-}cut$$

$\rightsquigarrow$

$$\dfrac{\dfrac{\dfrac{A\vdash A}{A\vdash B,\ A}\,w}{\vdash A\to B,\ A}\qquad \vdots\,\Pi'\quad \Gamma',\ A\to B\vdash A}{\Gamma'\vdash A}\ \ s\text{-}cut$$

$$\dfrac{\dfrac{\dfrac{A\vdash A}{A\vdash B,\ A}\,w}{\vdash A\to B,\ A}\qquad \dfrac{\vdots\,\Pi_1''\quad \Gamma''\vdash A\qquad \vdots\,\Pi_2''\quad \Gamma'',\ B\vdash A}{\Gamma'',\ A\to B\vdash A}}{\Gamma''\vdash A}\ \ l\text{-}cut$$

① Normalizing with explicit cut rules ($\rightsquigarrow$ small steps)

$$\dfrac{\dfrac{\vdots\,\Pi'}{\Gamma',\ A\to B\vdash A}}{\Gamma'\vdash (A\to B)\to A}\qquad \dfrac{\dfrac{\dfrac{A\vdash A}{A\vdash B,\ A}\,w}{\vdash A\to B,\ A}\qquad A\vdash A}{(A\to B)\to A\vdash A}$$
$$\Gamma'\vdash A\ \ l\text{-}cut$$

$\rightsquigarrow$

$$\dfrac{\dfrac{\dfrac{A\vdash A}{A\vdash B,\ A}\,w}{\vdash A\to B,\ A}\qquad \vdots\,\Pi'\quad \Gamma',\ A\to B\vdash A}{\Gamma'\vdash A}\ \ s\text{-}cut$$

$$\dfrac{\dfrac{\dfrac{A\vdash A}{A\vdash B,\ A}\,w}{\vdash A\to B,\ A}\qquad \dfrac{\vdots\,\Pi_1''\quad \Gamma''\vdash A\qquad \vdots\,\Pi_2''\quad \Gamma'',\ B\vdash A}{\Gamma'',\ A\to B\vdash A}}{\Gamma''\vdash A}\ \ l\text{-}cut$$

$\rightsquigarrow$

$$\begin{array}{c}\vdots\,\Pi_1''\\ \Gamma''\vdash A\end{array}$$

② How to manage active formulas?

The active formula in a premiss of a logical rule is the formula used to build the new formula in the conclusion. Classical λ-terms will have to keep track of them in order to make the difference between e.g.

$$\frac{\vdash A \to B,\ A \qquad A \vdash A}{(A \to B) \to A \vdash A}\ I{\to} \qquad \text{and} \qquad \frac{\vdash A \to B,\ A \qquad A \vdash A}{A \to A \vdash A}\ I{\to}$$

In ND, this new formula is at the same time the active formula relatively to the next rule, because it is always the (unique) right hand side formula:

$$\frac{\Gamma \vdash A \to B \qquad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B}\ {\to}e \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\ {\to}i$$

In LK, the active formula may change from a rule to the next one:

$$\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{\vdots}}{A \vdash B,\ A}\ r{\to}}{\vdash A \to B,\ A \qquad A \vdash A}\ I{\to}}{\dfrac{(A \to B) \to A \vdash A}{\vdots}}{\vdash ((A \to B) \to A) \to A}\ r{\to}$$

② Changing the active formula

Warning From now on, the difference between active formulas A and normal ones A becomes a formal feature of the syntax of the sequents: $\Gamma \vdash A, A_1, \ldots, A_n$ (other notations: $\Gamma \vdash \underline{A}, A_1, \ldots, A_n$, $\Gamma \vdash A; A_1, \ldots, A_n$)

It is natural to consider that the cut formula of a cut rule is the active formula of both premisses and that the conclusion has no active formula:

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ cut$$

Together with axiom rules having their active formula on the right hand side: $A \vdash A$ or on the left hand side: $A \vdash A$ ($\leftarrow$ used in Peirce Law proof), this yields a way to unselect the active formula:

$$\frac{A \vdash A \qquad \Gamma, A \vdash \Delta}{\Gamma, A \vdash \Delta}\ cut \qquad\qquad \frac{\Gamma \vdash A, \Delta \qquad A \vdash A}{\Gamma \vdash A, \Delta}\ cut$$

Peirce Law proof requires the selection of new active formulas on the right hand side only:

$$\frac{\Gamma \vdash A, \Delta}{\Gamma \vdash A, \Delta}\ \mu$$

② Classical ND with explicit active formulas (summing up)

$$A \vdash A$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ w$$

$$\frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\ r{\to}i$$

$$\frac{\Gamma \vdash A \to B, \Delta \qquad \Gamma' \vdash A, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\ {\to}e$$

② Classical ND with explicit active formulas (summing up)

$$A \vdash A \qquad A \vdash A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash A \to B, \Delta \qquad \Gamma' \vdash A, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\, {\to}e$$

② Classical ND with explicit active formulas (summing up)

$$A \vdash A \qquad A \vdash A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash A \to B, \Delta \qquad \Gamma' \vdash A, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

② Classical ND with explicit active formulas (summing up)

$$A \vdash A \qquad A \ \vdash A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, A \ \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash A \to B, \Delta \qquad \Gamma' \vdash A, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

$$\frac{\Gamma \vdash A \ , \Delta}{\Gamma \vdash A \ , \Delta}\, \mu$$

② Classical ND with explicit active formulas (summing up)

$$A \vdash A \qquad A^x \vdash A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, A^x \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash A \to B, \Delta \qquad \Gamma' \vdash A, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

$$\frac{\Gamma \vdash A\ , \Delta}{\Gamma \vdash A\ , \Delta}\, \mu$$

To be exact:
Every non active l.h.s. formula is labelled by a small latin letter as previously.

② Classical ND with explicit active formulas (summing up)

$$A \vdash A^{\alpha} \qquad A^{x} \vdash A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, A^{x} \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash A \to B, \Delta \qquad \Gamma' \vdash A, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash A, \Delta \qquad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

$$\frac{\Gamma \vdash A^{\alpha}, \Delta}{\Gamma \vdash A\ , \Delta}\, \mu$$

To be exact:
Every non active l.h.s. formula is labelled by a small latin letter as previously.
Every non active r.h.s. formula is labelled by a small greek letter.
Γ, Γ', Δ, Δ' = sets of labelled formulas. $\Gamma, \Gamma' \stackrel{\text{def}}{=} \Gamma \cup \Gamma'$, $A^{\alpha}, \Delta \stackrel{\text{def}}{=} \{A^{\alpha}\} \cup \Delta \ldots$

③ Logical intrepretation of the sequents with explicit active formulas

- $\Gamma \vdash A, \Delta$ = Proof of A:

  "If Γ then A unless one of Δ holds"

- $\Gamma, A \vdash \Delta$ = Refutation of A:

  "If Γ then A does not hold unless one of Δ does"

- $\Gamma \vdash \Delta$ = Contradiction:

  "Γ is contradictory unless one of Δ holds"

Back to ①. Elimination of the implicit cuts: as previously

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma, A^x \vdash B,\ \Delta}}{\Gamma \vdash A \to B,\ \Delta} \quad \dfrac{\vdots\,\Pi'}{\Gamma' \vdash A, \Delta'} \qquad \longrightarrow \qquad \dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma, A^x \vdash B, \Delta \overset{x}{\frown} \Gamma' \vdash A, \Delta'} \qquad \text{where}$$

$$\dfrac{\vdots\,\Pi'}{A^x \vdash A \overset{x}{\frown} \Gamma' \vdash A, \Delta'} \quad \overset{\text{def}}{=\!=} \quad \dfrac{\vdots\,\Pi'}{\Gamma' \vdash A, \Delta'}$$

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma_0 \vdash \Delta_0}\, w}{\Gamma, A^x \vdash \Delta} \quad \dfrac{\vdots\,\Pi'}{\overset{x}{\frown} \Gamma' \vdash A, \Delta'} \quad \overset{\text{def}}{=\!=} \quad \dfrac{\dfrac{\vdots\,\Pi}{\Gamma_0 \vdash \Delta_0}\, w}{\Gamma, \Gamma' \vdash \Delta, \Delta'} \quad \text{if } A^x \notin \Gamma_0$$

$$\dfrac{\dfrac{\vdots\,\Pi_1 \qquad \vdots\,\Pi_n}{\Gamma_1 \vdash \Delta_1 \cdots \Gamma_n \vdash \Delta_n}\, R}{\Gamma \vdash \Delta} \quad \dfrac{\vdots\,\Pi'}{\overset{x}{\frown} \Gamma' \vdash A, \Delta'} \quad \overset{\text{def}}{=\!=} \quad \dfrac{\vdots\,\Pi_1 \quad \vdots\,\Pi' \qquad\qquad \vdots\,\Pi_n \quad \vdots\,\Pi'}{\Gamma_1 \vdash \Delta_1 \overset{x}{\frown} \Gamma' \vdash A, \Delta' \quad \cdots \quad \Gamma_n \vdash \Delta_n \overset{x}{\frown} \Gamma' \vdash A, \Delta'}\, R$$

otherwise $(n = 1, 2)$

Back to ①. Elimination of the explicit cuts: same way

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A^{\alpha}, \Delta}\,\mu \qquad \vdots\,\Pi'}{\Gamma, \Gamma' \vdash \Delta, \Delta'} \longrightarrow \dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma \vdash A^{\alpha}, \Delta \overset{\alpha}{\curvearrowleft} \Gamma', A \vdash \Delta'}$$

where

$$\dfrac{\vdots\,\Pi'}{A \vdash A^{\alpha} \overset{\alpha}{\curvearrowleft} \Gamma', A \vdash \Delta'} \overset{\text{def}}{=\!=} \vdots\,\Pi' \quad \Gamma', A \vdash \Delta'$$

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma_0 \vdash \Delta_0}\,w}{\Gamma \vdash A^{\alpha}, \Delta} \overset{\alpha}{\curvearrowleft} \vdots\,\Pi' \quad \Gamma', A \vdash \Delta' \overset{\text{def}}{=\!=} \dfrac{\vdots\,\Pi}{\Gamma_0 \vdash \Delta_0}\,w \quad \text{if } A^{\alpha} \notin \Delta_0$$

$$\dfrac{\vdots\,\Pi_1 \quad \vdots\,\Pi_n}{\Gamma_1 \vdash \Delta_1 \cdots \Gamma_n \vdash \Delta_n}\,R \overset{\alpha}{\curvearrowleft} \vdots\,\Pi' \quad \Gamma', A \vdash \Delta' \overset{\text{def}}{=\!=} \dfrac{\vdots\,\Pi_1 \quad \vdots\,\Pi' \qquad \vdots\,\Pi_n \quad \vdots\,\Pi'}{\Gamma_1 \vdash \Delta_1 \overset{\alpha}{\curvearrowleft} \Gamma', A \vdash \Delta' \quad \cdots \quad \Gamma_n \vdash \Delta_n \overset{\alpha}{\curvearrowleft} \Gamma', A \vdash \Delta'}\,R$$

otherwise $(n = 1, 2)$

Back to ①. Elimination of the explicit cuts (continued)

In case the l.h.s. premiss of the cut rule is not the conclusion of a μ-rule, we look at the r.h.s. premiss: it can only be the conclusion of a w-rule, of a $l{\longrightarrow}i$-rule or an axiom.

If it is the conclusion of a w-rule:

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A,\ \Delta} \qquad \dfrac{\vdots\,\Pi'}{\Gamma', A \vdash \Delta'}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''} \quad\longrightarrow\quad \dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A,\ \Delta} \qquad \dfrac{\vdots\,\Pi'}{\Gamma', A \vdash \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

$$\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''$$

The cut moves up.

Back to ①. Elimination of the explicit cuts (continued)

If the r.h.s. premiss is the conclusion of a $l{\longrightarrow}i$-rule:

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A \to B,\ \Delta} \qquad \dfrac{\vdots\,\Pi' \qquad \vdots\,\Pi''}{\Gamma' \vdash A, \Delta' \qquad \Gamma'', B \vdash \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''} \longrightarrow \dfrac{\dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma \vdash A \to B, \Delta \qquad \Gamma' \vdash A, \Delta'} \qquad \vdots\,\Pi''}{\dfrac{\Gamma, \Gamma' \vdash B, \Delta, \Delta' \qquad \Gamma'', B \vdash \Delta''}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}}$$

The cut formula get smaller.

Remark. If $\Gamma \vdash A \to B,\ \Delta$ is the conclusion of a μ-rule, we may do the inverse rewriting in order to get rid of the explicit cut rule as previously.

Back to ①. Elimination of the explicit cuts (continued)

If the r.h.s. premiss is the conclusion of a $l{\to}i$-rule:

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A \to B,\ \Delta} \quad \dfrac{\vdots\,\Pi' \qquad \vdots\,\Pi''}{\Gamma' \vdash A, \Delta' \quad \Gamma'', B \vdash \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''} \equiv \dfrac{\dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma \vdash A \to B, \Delta \quad \Gamma' \vdash A, \Delta'} \qquad \vdots\,\Pi''}{\Gamma, \Gamma' \vdash B, \Delta, \Delta' \qquad \Gamma'', B \vdash \Delta''}}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

The cut formula get smaller.

Remark. If $\Gamma \vdash A \to B,\ \Delta$ is the conclusion of a μ-rule, we may do the inverse rewriting in order to get rid of the explicit cut rule as previously.

$\rightsquigarrow$ We do not want to choose the direction of this rewriting.

Back to ①. Elimination of the explicit cuts (continued)

At last, if the cut is a right *ax-cut* (i.e. if its r.h.s. premiss is an axiom):

$$\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma \vdash A, \Delta\end{array} \qquad A \vdash A}{\Gamma \vdash A^{\alpha}, \Delta}$$

This is just the way of desactivating A (before activating a new formula with a μ-rule).

Back to ①. Elimination of the explicit cuts (continued)

At last, if the cut is a right *ax-cut* (i.e. if its r.h.s. premiss is an axiom):

$$\dfrac{\begin{array}{c}\vdots\,\Pi\\ \Gamma \vdash A, \Delta\end{array} \qquad A \vdash A}{\Gamma \vdash A^{\alpha}, \Delta}$$

This is just the way of desactivating A (before activating a new formula with a μ-rule).

$\rightsquigarrow$ Unlike in LK, an ax-cut cannot be removed in our calculus . . .

Back to ①. Elimination of the explicit cuts (continued)

At last, if the cut is a right *ax-cut* (i.e. if its r.h.s. premiss is an axiom):

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A, \Delta} \qquad A \vdash A}{\dfrac{\Gamma \vdash A^{\alpha}, \Delta}{\Gamma \vdash A, \Delta}\,\mu} \longrightarrow \begin{array}{c}\vdots\,\Pi\\ \Gamma \vdash A, \Delta\end{array}$$

This is just the way of desactivating A (before activating a new formula with a μ-rule).

$\rightsquigarrow$ Unlike in LK, an ax-cut cannot be removed in our calculus . . .
. . . unless the same formula is activated immediatly afterwards

Back to ①. Elimination of the explicit cuts (continued)

At last, if the cut is a right *ax-cut* (i.e. if its r.h.s. premiss is an axiom):

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A, \Delta} \qquad A \vdash A}{\dfrac{\Gamma \vdash A^{\alpha}, \Delta}{\Gamma \vdash A, \Delta}\,\mu} \longrightarrow \begin{array}{c}\vdots\,\Pi\\ \Gamma \vdash A, \Delta\end{array} \qquad \text{only if } A^{\alpha} \notin \Delta$$

This is just the way of desactivating A (before activating a new formula with a μ-rule).

$\rightsquigarrow$ Unlike in LK, an ax-cut cannot be removed in our calculus . . .
. . . unless the same formula is activated immediatly afterwards
. . . and $A^{\alpha} \notin \Delta$! (otherwise the μ-rule makes an explicit contraction of the active formula with one of Δ).

Back to ①. Elimination of the explicit cuts (continued)

Here again, the inverse rewriting is useful to get rid of the cut rule:

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A \to B^{\,\alpha}, \Delta}\,\mu}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\;\dfrac{\vdots\,\Pi'}{\Gamma' \vdash A, \Delta'} \quad\longrightarrow^{-1}\quad \dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A \to B^{\,\alpha}, \Delta}\,\mu}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\;\dfrac{\vdots\,\Pi'}{\Gamma' \vdash A, \Delta'}\;\;B \vdash B}{\dfrac{\Gamma, \Gamma' \vdash B^{\beta}, \Delta, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\,\mu}$$

$$\equiv\quad \dfrac{\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A \to B^{\,\alpha}, \Delta}\,\mu}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\;\dfrac{\dfrac{\vdots\,\Pi'}{\Gamma' \vdash A, \Delta'}\;\;B \vdash B}{\Gamma', A \to B \vdash B, \Delta'}}{\dfrac{\Gamma, \Gamma' \vdash B^{\beta}, \Delta, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\,\mu} \quad\longrightarrow\quad \dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A \to B^{\,\alpha}, \Delta}\;\overset{\alpha}{\curvearrowleft}\;\dfrac{\dfrac{\vdots\,\Pi'}{\Gamma' \vdash A, \Delta'}\;\;B \vdash B}{\Gamma', A \to B \vdash B, \Delta'}}{\dfrac{\Gamma, \Gamma' \vdash B^{\beta}, \Delta, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\,\mu}$$

$\rightsquigarrow$ We do not want to choose the direction of this rewriting either.

Cut elimination rules (summing up)

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma, A^x \vdash B,\ \Delta}}{\Gamma \vdash A \to B,\ \Delta} \quad \dfrac{\vdots\,\Pi'}{\Gamma' \vdash A, \Delta'} \longrightarrow \dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma, A^x \vdash B, \Delta \overset{\curvearrowleft x}{} & \Gamma' \vdash A, \Delta'\end{array}}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}$$

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A^\alpha, \Delta}\ \mu}{\Gamma \vdash A,\ \Delta} \quad \dfrac{\vdots\,\Pi'}{\Gamma', A \vdash \Delta'} \longrightarrow \dfrac{\begin{array}{cc}\vdots\,\Pi & \vdots\,\Pi' \\ \Gamma \vdash A^\alpha, \Delta \overset{\curvearrowleft \alpha}{} & \Gamma', A \vdash \Delta'\end{array}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

$$\dfrac{\vdots\,\Pi}{\Gamma \vdash A \to B,\ \Delta} \quad \dfrac{\vdots\,\Pi' \quad \vdots\,\Pi''}{\Gamma', \Gamma'', A \to B \vdash \Delta', \Delta''} \equiv \dfrac{\dfrac{\vdots\,\Pi \quad \vdots\,\Pi'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'} \quad \vdots\,\Pi''}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}$$

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A, \Delta} \quad A \vdash A}{\dfrac{\Gamma \vdash A^\alpha, \Delta}{\Gamma \vdash A, \Delta}\ \mu} \equiv \vdots\,\Pi \quad \Gamma \vdash A, \Delta \qquad \text{only if } A^\alpha \notin \Delta$$

Cut elimination rules (summing up)

These four rewrite rules and the commutation rule between w and *cut*:

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A,\ \Delta} \qquad \dfrac{\dfrac{\vdots\,\Pi'}{\Gamma', A \vdash \Delta'}}{\Gamma', \Gamma'', A \vdash \Delta', \Delta''}\,w}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}\,cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash A,\ \Delta} \qquad \dfrac{\vdots\,\Pi'}{\Gamma', A \vdash \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\,cut}{\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta''}\,w$$

are enough to remove all the implicit cuts and all the explicit cut rules except the rules *ax-cut*.

Moreover:

SN Property For every rewrite sequence $\Pi_1 \xrightarrow[\text{or}]{} \Pi_2 \xrightarrow[\text{or}]{} \Pi_3 \xrightarrow[\text{or}]{} \Pi_4 \xrightarrow[\text{or}]{} \cdots$ there is n such that $\Pi_n \equiv \Pi_{n+1} \equiv \Pi_{n+2} \equiv \cdots$

CR Property If $\Pi \twoheadrightarrow \Pi_1$ and $\Pi \twoheadrightarrow \Pi_2$ ($\twoheadrightarrow$ = refl. trans. closure of $\xrightarrow[\text{or}]{}$) then there is Π' such that $\Pi_1 \twoheadrightarrow \Pi'$ and $\Pi_2 \twoheadrightarrow \Pi'$.

$\rightsquigarrow$ Every Π has a normal form up to $\equiv$.

Conclusion: Our hybrid of ND and LK is a well born natural deduction.

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \to B \vdash \Delta, \Delta'}\ l{\to}i \qquad \frac{\Gamma, A^x \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\ r{\to}i \qquad \frac{\Gamma \vdash A \to B, \Delta \quad \Gamma' \vdash A, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\ {\to}e$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ cut \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ w$$

$$\frac{\Gamma \vdash A^\alpha, \Delta}{\Gamma \vdash A, \Delta}\ \mu$$

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.

$$A \vdash \alpha : A \qquad x : A \vdash A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, x : A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash A \to B, \Delta \quad \Gamma' \vdash A, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

$$\frac{\Gamma \vdash \alpha : A, \Delta}{\Gamma \vdash A, \Delta}\, \mu$$

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.
- At any step, the active formula is the type of the λ-term built so far.

$$A \vdash \alpha : A \qquad x : A \vdash x : A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma \vdash \lambda x\, t : A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

$$\frac{\Gamma \vdash \alpha : A, \Delta}{\Gamma \vdash A, \Delta}\, \mu$$

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.
- At any step, the active formula is the type of the λ-term built so far.

$$\alpha : A \vdash \alpha : A \qquad x : A \vdash x : A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma \vdash \lambda x\, t : A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

$$\frac{\Gamma \vdash \alpha : A, \Delta}{\Gamma \vdash A, \Delta}\, \mu$$

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.
- At any step, the active formula is the type of the λ-term built so far.

$$\alpha : A \vdash \alpha : A \qquad x : A \vdash x : A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : B \vdash \Delta'}{\Gamma, \Gamma', \quad A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma \vdash \lambda x\, t : A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

$$\frac{\Gamma \vdash \alpha : A\,, \Delta}{\Gamma \vdash \quad A\,, \Delta}\, \mu$$

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.
- At any step, the active formula is the type of the λ-term built so far.
- Every new inference rule correspond to a new specific constructor or binder of the λ-calculus.

$$\alpha : A \vdash \alpha : A \qquad x : A \vdash x : A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : B \vdash \Delta'}{\Gamma, \Gamma', \quad A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma \vdash \lambda x\, t : A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

$$\frac{\Gamma \vdash \alpha : A\,, \Delta}{\Gamma \vdash \quad A\,, \Delta}\, \mu$$

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.
- At any step, the active formula is the type of the λ-term built so far.
- Every new inference rule correspond to a new specific constructor or binder of the λ-calculus.

$$\alpha : A \vdash \alpha : A \qquad x : A \vdash x : A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : B \vdash \Delta'}{\Gamma, \Gamma', t.\sigma : A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma \vdash \lambda x\, t : A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, cut$$

$$\frac{\Gamma \vdash \alpha : A\,, \Delta}{\Gamma \vdash \qquad A\,, \Delta}\, \mu$$

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.
- At any step, the active formula is the type of the λ-term built so far.
- Every new inference rule correspond to a new specific constructor or binder of the λ-calculus.

$$\alpha : A \vdash \alpha : A \qquad x : A \vdash x : A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : B \vdash \Delta'}{\Gamma, \Gamma', t.\sigma : A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma \vdash \lambda x\, t : A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : A \vdash \Delta'}{t \star \sigma : [\Gamma, \Gamma' \vdash \Delta, \Delta']}\, cut$$

$$\frac{\Gamma \vdash \alpha : A\,, \Delta}{\Gamma \vdash \qquad A\,, \Delta}\, \mu$$

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.
- At any step, the active formula is the type of the λ-term built so far.
- Every new inference rule correspond to a new specific constructor or binder of the λ-calculus.

$$\alpha : A \vdash \alpha : A \qquad x : A \vdash x : A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : B \vdash \Delta'}{\Gamma, \Gamma', t.\sigma : A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma \vdash \lambda x\, t : A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : A \vdash \Delta'}{t \star \sigma : [\Gamma, \Gamma' \vdash \Delta, \Delta']}\, cut \qquad \left(\frac{c : [\Gamma \vdash \Delta]}{c : [\Gamma, \Gamma' \vdash \Delta, \Delta']}\, w \right)$$

$$\frac{\Gamma \vdash \alpha : A, \Delta}{\Gamma \vdash A, \Delta}\, \mu$$

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.
- At any step, the active formula is the type of the λ-term built so far.
- Every new inference rule correspond to a new specific constructor or binder of the λ-calculus.

$$\alpha : A \vdash \alpha : A \qquad x : A \vdash x : A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : B \vdash \Delta'}{\Gamma, \Gamma', t.\sigma : A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma \vdash \lambda x\, t : A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : A \vdash \Delta'}{t \star \sigma : [\Gamma, \Gamma' \vdash \Delta, \Delta']}\, cut \qquad \left(\frac{c : [\Gamma \vdash \Delta]}{c : [\Gamma, \Gamma' \vdash \Delta, \Delta']}\, w \right)$$

$$\frac{c : [\Gamma \vdash \alpha : A, \Delta]}{\Gamma \vdash A, \Delta}\, \mu$$

$\lambda\mu\sigma$-calculus: Syntax

The λ-calculus corresponding to our classical natural deduction is straightforwardly defined if we keep in mind the following principles:

- The labels of the non active formulas are the variables of the λ-calculus.
- At any step, the active formula is the type of the λ-term built so far.
- Every new inference rule correspond to a new specific constructor or binder of the λ-calculus.

$$\alpha : A \vdash \alpha : A \qquad x : A \vdash x : A \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\, w$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : B \vdash \Delta'}{\Gamma, \Gamma', t.\sigma : A \to B \vdash \Delta, \Delta'}\, l{\to}i \qquad \frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma \vdash \lambda x\, t : A \to B, \Delta}\, r{\to}i \qquad \frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}\, {\to}e$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : A \vdash \Delta'}{t \star \sigma : [\Gamma, \Gamma' \vdash \Delta, \Delta']}\, cut \qquad \left(\frac{c : [\Gamma \vdash \Delta]}{c : [\Gamma, \Gamma' \vdash \Delta, \Delta']}\, w \right)$$

$$\frac{c : [\Gamma \vdash \alpha : A, \Delta]}{\Gamma \vdash \mu\alpha\, c : A, \Delta}\, \mu$$

$\lambda\mu\sigma$-calculus: Substitutions

As for intuitionistic ND, the proof $\dfrac{\vdots\,\Pi \qquad \vdots\,\Pi' \\ \Gamma, A^x \vdash B, \Delta \quad \overset{x}{\frown} \quad \Gamma' \vdash A, \Delta'}{\Gamma, \Gamma' \vdash B, \Delta, \Delta'}$

is naturally lifted to a derivation $\dfrac{\vdots\,\Pi \qquad \vdots\,\Pi' \\ \Gamma, x : A \vdash t : B, \Delta \quad \overset{x}{\frown} \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash t[x := u] : B, \Delta, \Delta'}$

and similarly the proof $\dfrac{\vdots\,\Pi \qquad \vdots\,\Pi' \\ \Gamma \vdash A^\alpha, \Delta \quad \overset{\alpha}{\frown} \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$

is lifted to a derivation $\dfrac{\vdots\,\Pi \qquad \vdots\,\Pi' \\ c : [\Gamma \vdash \alpha : A, \Delta] \quad \overset{\alpha}{\frown} \quad \Gamma', \sigma : A \vdash \Delta'}{c[\alpha := \sigma] : [\Gamma, \Gamma' \vdash \Delta, \Delta']}$

$\lambda\mu\sigma$-calculus: Reduction rules

The commutation *cut*-rule/*w*-rule has no effect on the λ-terms:

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash t : A, \Delta} \qquad \dfrac{\dfrac{\vdots\,\Pi'}{\Gamma', \sigma : A \vdash \Delta'}}{\Gamma', \Gamma'', \sigma : A \vdash \Delta', \Delta''}\, w}{t \star \sigma : [\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta'']}\, cut \quad \longrightarrow \quad \dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash t : A, \Delta} \qquad \dfrac{\vdots\,\Pi'}{\Gamma', \sigma : A \vdash \Delta'}}{t \star \sigma : [\Gamma, \Gamma' \vdash \Delta, \Delta']}\, cut}{t \star \sigma : [\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta'']}\, w$$

Other rewritings:

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma, x : A \vdash t : B, \Delta}}{\Gamma \vdash \lambda x\, t : A \to B, \Delta} \qquad \dfrac{\vdots\,\Pi'}{\Gamma' \vdash u : A, \Delta'}}{\Gamma, \Gamma' \vdash (\lambda x\, t) u : B, \Delta, \Delta'} \quad \longrightarrow \quad \dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma, x : A \vdash t : B, \Delta \overset{x}{\frown} \Gamma' \vdash u : A, \Delta'}}{\Gamma, \Gamma' \vdash t[x := u] : B, \Delta, \Delta'}$$

When removing the logical part: $(\lambda x\, t) u \to_\beta t[x := u]$

$$\dfrac{\dfrac{\vdots\,\Pi}{c : [\Gamma \vdash \alpha : A, \Delta]}}{\Gamma \vdash \mu\alpha\, c : A, \Delta}\, \mu \qquad \dfrac{\vdots\,\Pi'}{\Gamma', \sigma : A \vdash \Delta'}}{(\mu\alpha\, c) \star \sigma : [\Gamma, \Gamma' \vdash \Delta, \Delta']} \quad \longrightarrow \quad \dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{c : [\Gamma \vdash \alpha : A, \Delta] \overset{\alpha}{\frown} \Gamma', \sigma : A \vdash \Delta'}}{c[\alpha := \sigma] : [\Gamma, \Gamma' \vdash \Delta, \Delta']}$$

When removing the logical part: $(\mu\alpha\, c) \star \sigma \to_\gamma c[\alpha := \sigma]$

$\lambda\mu\sigma$-calculus: Reduction rules

$$\dfrac{\dfrac{\vdots\,\Pi \qquad \dfrac{\vdots\,\Pi' \qquad \vdots\,\Pi''}{\Gamma' \vdash u : A, \Delta' \quad \Gamma'', \pi : B \vdash \Delta''}}{\Gamma \vdash t : A \to B, \Delta \quad \Gamma', \Gamma'', u.\pi : A \to B \vdash \Delta', \Delta''}}{t \star u.\pi : [\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta'']} \equiv \dfrac{\dfrac{\vdots\,\Pi \qquad \vdots\,\Pi'}{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'} \qquad \vdots\,\Pi''}{tu \star \pi : [\Gamma, \Gamma', \Gamma'' \vdash \Delta, \Delta', \Delta'']}$$

When removing the logical part: $t \star u.\pi =_\sigma tu \star \pi$

$$\dfrac{\dfrac{\vdots\,\Pi}{\Gamma \vdash t : A, \Delta \quad \alpha : A \vdash A}}{\dfrac{t \star \alpha : [\Gamma \vdash A, \Delta]}{\Gamma \vdash \mu\alpha\, t \star \alpha : A, \Delta}\,\mu} \equiv \vdots\,\Pi \quad \Gamma \vdash t : A, \Delta \text{ only if } \alpha : A \notin \Delta$$

When removing the logical part: $\mu\alpha\, t \star \alpha =_\theta t$ only if $\alpha \notin t$

In summary:

$$(\lambda x\, t)\, u \to_\beta t[x := u] \qquad t \star u.\pi =_\sigma tu \star \pi$$
$$(\mu\alpha\, c) \star \sigma \to_\gamma c[x := \sigma] \qquad \mu\alpha\, t \star \alpha =_\theta t \text{ only if } \alpha \notin t$$

$\lambda\mu\sigma$-calculus (summing up)

The original intuitionistic calculus:

λ-terms: $t = \underset{\textit{(assumption)}}{x} \mid \underset{\textit{(Modus Ponens)}}{tt} \mid \underset{\textit{(abstraction of a hyp.)}}{\lambda x\, t}$ *(proofs)*

Notations λ-variables: $x, y, z \ldots$ λ-terms: $t, u, v \ldots$

Precedences *application* $>$ λ

Reduction rules

$$(\lambda x\, t)\, u \rightarrow_\beta t[x := u]$$

$\lambda\mu\sigma$-calculus (summing up)

λ-terms: $t = \underset{\text{(assumption)}}{x} \mid \underset{\text{(Modus Ponens)}}{tt} \mid \underset{\text{(abstraction of a hyp.)}}{\lambda x\, t}$ *(proofs)*

σ-terms: $\sigma = \underset{\text{(assumption to the cont.)}}{\alpha}$ *(refutations)*

Notations λ-variables: $x, y, z \ldots$ λ-terms: $t, u, v \ldots$
σ-variables: $\alpha, \beta, \gamma \ldots$

Precedences *application* $>$ λ

Reduction rules

$$(\lambda x\, t)\, u \rightarrow_\beta t[x := u]$$

$\lambda\mu\sigma$-calculus (summing up)

λ-terms: $t \quad = \quad \underset{\textit{(assumption)}}{x} \;\Big|\; \underset{\textit{(Modus Ponens)}}{tt} \;\Big|\; \underset{\textit{(abstraction of a hyp.)}}{\lambda x\, t}$ *(proofs)*

σ-terms: $\sigma \quad = \quad \underset{\textit{(assumption to the cont.)}}{\alpha} \;\Big|\; \underset{\textit{(refutation of an impl.)}}{t.\sigma}$ *(refutations)*

Notations
λ-variables: $x, y, z \ldots$ λ-terms: $t, u, v \ldots$
σ-variables: $\alpha, \beta, \gamma \ldots$ σ-terms: $\pi, \rho, \sigma \ldots$

Precedences *application* $>$ λ $>$ $.$

Reduction rules

$$(\lambda x\, t)\, u \;\to_\beta\; t[x := u]$$

$\lambda\mu\sigma$-calculus (summing up)

λ-terms: $t = \underbrace{x}_{\text{(assumption)}} \mid \underbrace{tt}_{\text{(Modus Ponens)}} \mid \underbrace{\lambda x\, t}_{\text{(abstraction of a hyp.)}}$ *(proofs)*

c-terms: $c = t \star \sigma$ *(contradictions)*

σ-terms: $\sigma = \underbrace{\alpha}_{\text{(assumption to the cont.)}} \mid \underbrace{t.\sigma}_{\text{(refutation of an impl.)}}$ *(refutations)*

Notations λ-variables: $x, y, z \ldots$ λ-terms: $t, u, v \ldots$
σ-variables: $\alpha, \beta, \gamma \ldots$ σ-terms: $\pi, \rho, \sigma \ldots$

Precedences *application* $>$ λ $>$. $>$ $\star$

Reduction rules

$$(\lambda x\, t)\, u \rightarrow_\beta t[x := u]$$

$\lambda\mu\sigma$-calculus (summing up)

λ-terms: $t = \underbrace{x}_{\text{(assumption)}} \mid \underbrace{tt}_{\text{(Modus Ponens)}} \mid \underbrace{\lambda x\, t}_{\text{(abstraction of a hyp.)}}$ *(proofs)*

c-terms: $c = t \star \sigma$ *(contradictions)*

σ-terms: $\sigma = \underbrace{\alpha}_{\text{(assumption to the cont.)}} \mid \underbrace{t.\sigma}_{\text{(refutation of an impl.)}}$ *(refutations)*

Notations λ-variables: $x, y, z \ldots$ λ-terms: $t, u, v \ldots$
σ-variables: $\alpha, \beta, \gamma \ldots$ σ-terms: $\pi, \rho, \sigma \ldots$

Precedences *application* $>$ λ $>$. $>$ $\star$

Reduction rules

$$(\lambda x\, t)\, u \to_\beta t[x := u] \qquad t \star u.\sigma =_\sigma tu \star \sigma$$

$\lambda\mu\sigma$-calculus (summing up)

λ-terms: $t = \underset{\text{(assumption)}}{x} \mid \underset{\text{(Modus Ponens)}}{tt} \mid \underset{\text{(abstraction of a hyp.)}}{\lambda x\, t} \mid \underset{\text{(Reductio ad Absurdum)}}{\mu\alpha\, c}$ *(proofs)*

c-terms: $c = t \star \sigma$ *(contradictions)*

σ-terms: $\sigma = \underset{\text{(assumption to the cont.)}}{\alpha} \mid \underset{\text{(refutation of an impl.)}}{t.\sigma}$ *(refutations)*

Notations
λ-variables: $x, y, z \ldots$ λ-terms: $t, u, v \ldots$
σ-variables: $\alpha, \beta, \gamma \ldots$ σ-terms: $\pi, \rho, \sigma \ldots$

Precedences *application* $>$ λ $>$. $>$ $\star$ $>$ μ

Reduction rules

$$(\lambda x\, t)\, u \to_\beta t[x := u] \qquad t \star u.\sigma =_\sigma tu \star \sigma$$

$\lambda\mu\sigma$-calculus (summing up)

λ-terms: $t = \underset{\text{(assumption)}}{x} \mid \underset{\text{(Modus Ponens)}}{tt} \mid \underset{\text{(abstraction of a hyp.)}}{\lambda x\, t} \mid \underset{\text{(Reductio ad Absurdum)}}{\mu\alpha\, c}$ *(proofs)*

c-terms: $c = t \star \sigma$ *(contradictions)*

σ-terms: $\sigma = \underset{\text{(assumption to the cont.)}}{\alpha} \mid \underset{\text{(refutation of an impl.)}}{t.\sigma}$ *(refutations)*

Notations
λ-variables: $x, y, z \ldots$ λ-terms: $t, u, v \ldots$
σ-variables: $\alpha, \beta, \gamma \ldots$ σ-terms: $\pi, \rho, \sigma \ldots$

Precedences *application* $>$ λ $>$. $>$ $\star$ $>$ μ

Reduction rules

$$(\lambda x\, t)\, u \to_\beta t[x := u] \qquad t \star u.\sigma =_\sigma tu \star \sigma$$
$$(\mu\alpha\, c) \star \sigma \to_\gamma c[x := \sigma]$$

$\lambda\mu\sigma$-calculus (summing up)

λ-terms: $t = \underset{\text{(assumption)}}{x} \mid \underset{\text{(Modus Ponens)}}{tt} \mid \underset{\text{(abstraction of a hyp.)}}{\lambda x\, t} \mid \underset{\text{(Reductio ad Absurdum)}}{\mu\alpha\, c}$ *(proofs)*

c-terms: $c = t \star \sigma$ *(contradictions)*

σ-terms: $\sigma = \underset{\text{(assumption to the cont.)}}{\alpha} \mid \underset{\text{(refutation of an impl.)}}{t.\sigma}$ *(refutations)*

Notations
λ-variables: $x, y, z \ldots$ λ-terms: $t, u, v \ldots$
σ-variables: $\alpha, \beta, \gamma \ldots$ σ-terms: $\pi, \rho, \sigma \ldots$

Precedences *application* $>$ λ $>$. $>$ $\star$ $>$ μ

Reduction rules

$$(\lambda x\, t)\, u \to_\beta t[x := u] \qquad t \star u.\sigma =_\sigma tu \star \sigma$$
$$(\mu\alpha\, c) \star \sigma \to_\gamma c[x := \sigma] \qquad \mu\alpha\, t \star \alpha =_\theta t \text{ only if } \alpha \notin t$$

$\lambda\mu\sigma$-calculus (summing up)

$$\left.\begin{array}{lrcl} \lambda\text{-terms:} & t & = & x \mid tt \mid \lambda x\, t \mid \mu\alpha\, c \\ c\text{-terms:} & c & = & t \star \sigma \\ \sigma\text{-terms:} & \sigma & = & \alpha \mid t.\sigma \end{array}\right\} \text{3 sorted calculus}$$

Notations λ-variables: $x, y, z \ldots$ λ-terms: $t, u, v \ldots$
σ-variables: $\alpha, \beta, \gamma \ldots$ σ-terms: $\pi, \rho, \sigma \ldots$

Precedences $\mathit{application} > \lambda > . > \star > \mu$

Reduction rules

$$(\lambda x\, t)\, u \to_\beta t[x := u] \qquad t \star u.\sigma =_\sigma tu \star \sigma$$
$$(\mu\alpha\, c) \star \sigma \to_\gamma c[x := \sigma] \qquad \mu\alpha\, t \star \alpha =_\theta t \quad \text{only if} \quad \alpha \notin t$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

Write down the simplest $\lambda\mu\sigma$-term of a proof of Peirce Law.

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

Write down the simplest $\lambda\mu\sigma$-term of a proof of Peirce Law.

We start from its simplest proof in LK:

$$\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{\vdash A \to B,\ A} \qquad A \vdash A}{(A \to B) \to A \vdash A}}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{A \vdash B,\ A}\ \mu}{\vdash A \to B,\ A} \qquad A \vdash A}{(A \to B) \to A \vdash A}}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{(A \to B) \to A \vdash (A \to B) \to A \qquad \dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{A \vdash B,\ A}\ \mu}{\vdash A \to B,\ A} \qquad \dfrac{A \vdash A}{(A \to B) \to A \vdash A}}{(A \to B) \to A \vdash A}\ cut}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{(A \to B) \to A \vdash (A \to B) \to A \qquad \dfrac{\dfrac{\dfrac{A \vdash A}{A \vdash B,\ A}\ w}{A \vdash B,\ A}\ \mu}{\vdash A \to B,\ A} \qquad A \vdash A}{(A \to B) \to A \vdash A}\ cut}{\dfrac{(A \to B) \to A \vdash A}{(A \to B) \to A \vdash A}\ \mu}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A \qquad A \vdash A}{A \vdash A}\,cut}{A \vdash B,\ A}\,w}{A \vdash B,\ A}\,\mu}{\vdash A \to B,\ A} \qquad A \vdash A}{(A \to B) \to A \vdash (A \to B) \to A \qquad (A \to B) \to A \vdash A}\,cut}{(A \to B) \to A \vdash A}\,\mu}{(A \to B) \to A \vdash A}\,\mu}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A} \quad \dfrac{}{A \vdash A}}{cut}}{A \vdash A}}{w}}{A \vdash B, A}}{\mu}}{\vdash A \to B, A} \qquad \dfrac{}{A \vdash A}}{cut}}{(A \to B) \to A \vdash (A \to B) \to A \qquad (A \to B) \to A \vdash A}}{cut}}{(A \to B) \to A \vdash A}}{\mu}}{(A \to B) \to A \vdash A}}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A \qquad A \vdash A}{A \vdash A}\,cut}{A \vdash B, A}\,w}{A \vdash B, A}\,\mu}{\vdash A \to B, A \qquad A \vdash A}}{(A \to B) \to A \vdash (A \to B) \to A \qquad (A \to B) \to A \vdash A}\,cut$$

$$\dfrac{(A \to B) \to A \vdash A}{f : (A \to B) \to A \vdash A}\,\mu$$

$$\vdash ((A \to B) \to A) \to A$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A \qquad A \vdash A}{A \vdash A}\,cut}{A \vdash B, A}\,w}{A \vdash B, A}\,\mu}{\vdash A \to B, A}}{f : (A \to B) \to A \vdash (A \to B) \to A \qquad (A \to B) \to A \vdash A}\,cut}{f : (A \to B) \to A \vdash A}\,\mu$$
$$\dfrac{f : (A \to B) \to A \vdash A}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{f : (A \to B) \to A \vdash (A \to B) \to A \qquad \dfrac{\dfrac{\dfrac{\dfrac{A \vdash A \qquad A \vdash A}{A \vdash A}\,cut}{A \vdash B, A}\,w}{A \vdash B, A}\,\mu}{\vdash A \to B, A}\,\dots \qquad A \vdash A}{(A \to B) \to A \vdash A}\,\dots}{f : (A \to B) \to A \vdash \alpha : A}\,cut$$

$$\dfrac{f : (A \to B) \to A \vdash \alpha : A}{f : (A \to B) \to A \vdash A}\,\mu$$

$$\dfrac{f : (A \to B) \to A \vdash A}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \vdash A \qquad A \vdash \alpha : A}{A \vdash \alpha : A}\,cut}{A \vdash B,\ \alpha : A}\,w}{\vdash A \to B,\ \alpha : A}\,\mu \qquad A \vdash \alpha : A}{f : (A \to B) \to A \vdash (A \to B) \to A \qquad (A \to B) \to A \vdash \alpha : A}\,cut}{f : (A \to B) \to A \vdash \alpha : A}\,\mu$$
$$\dfrac{f : (A \to B) \to A \vdash A}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A \vdash \quad A \qquad\qquad A \vdash \alpha : A}{A \vdash \alpha : A}\,cut}{A \vdash \quad B,\ \alpha : A}\,w}{x : A \vdash \qquad B,\ \alpha : A}\,\mu}{\vdash \qquad A \to B,\ \alpha : A} \qquad\qquad A \vdash \alpha : A}{f : (A \to B) \to A \vdash \quad (A \to B) \to A \qquad\qquad (A \to B) \to A \vdash \alpha : A}\,cut}{f : (A \to B) \to A \vdash \alpha : A}\,\mu}{f : (A \to B) \to A \vdash \qquad A}}{\vdash \qquad ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \vdash \quad A \qquad\qquad A \vdash \alpha : A}{x : A \vdash \alpha : A}\,cut}{x : A \vdash \quad B,\ \alpha : A}\,w}{\vdash \quad A \to B,\ \alpha : A}\,\mu \qquad\qquad A \vdash \alpha : A}{f : (A \to B) \to A \vdash \quad (A \to B) \to A \qquad\qquad (A \to B) \to A \vdash \alpha : A}\,cut}{f : (A \to B) \to A \vdash \alpha : A}\,\mu$$

$$\dfrac{f : (A \to B) \to A \vdash \quad A}{\vdash \quad ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{f : (A \to B) \to A \vdash \quad (A \to B) \to A \qquad\qquad (A \to B) \to A \vdash \alpha : A}{f : (A \to B) \to A \vdash \alpha : A}\,cut}{\dfrac{f : (A \to B) \to A \vdash \quad A}{\vdash \quad ((A \to B) \to A) \to A}\,\mu}$$

$$\dfrac{\dfrac{x : A \vdash \quad A \qquad\qquad A \vdash \alpha : A}{x : A \vdash \alpha : A}\,cut}{\dfrac{x : A \vdash \delta : B,\ \alpha : A}{x : A \vdash \quad B,\ \alpha : A}\,\mu}\,w$$

$$\dfrac{\vdash \quad A \to B,\ \alpha : A \qquad\qquad A \vdash \alpha : A}{(A \to B) \to A \vdash \alpha : A}\,cut$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \vdash x : A \qquad \alpha : A \vdash \alpha : A}{x : A \vdash \alpha : A}\,cut}{x : A \vdash \delta : B,\ \alpha : A}\,w}{x : A \vdash B,\ \alpha : A}\,\mu}{\vdash A \to B,\ \alpha : A}}{f : (A \to B) \to A \vdash f : (A \to B) \to A \qquad (A \to B) \to A \vdash \alpha : A}\,cut}{\dfrac{f : (A \to B) \to A \vdash \alpha : A}{f : (A \to B) \to A \vdash A}\,\mu}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \vdash x : A \qquad \alpha : A \vdash \alpha : A}{x \star \alpha : [x : A \vdash \alpha : A]}\,cut}{x : A \vdash \delta : B,\ \alpha : A}\,w}{x : A \vdash B,\ \alpha : A}\,\mu}{\vdash A \to B,\ \alpha : A}}{f : (A \to B) \to A \vdash f : (A \to B) \to A \qquad (A \to B) \to A \vdash \alpha : A}\,cut}{f : (A \to B) \to A \vdash \alpha : A}\,\mu}{f : (A \to B) \to A \vdash A}}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \vdash x : A \qquad \alpha : A \vdash \alpha : A}{x \star \alpha : [x : A \vdash \alpha : A]}\,cut}{x \star \alpha : [x : A \vdash \delta : B,\ \alpha : A]}\,w}{x : A \vdash B,\ \alpha : A}\,\mu}{\vdash A \to B,\ \alpha : A}}{f : (A \to B) \to A \vdash f : (A \to B) \to A \qquad (A \to B) \to A \vdash \alpha : A}\,cut}{f : (A \to B) \to A \vdash \alpha : A}\,\mu}{f : (A \to B) \to A \vdash A}}{\vdash ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \vdash x : A \qquad \alpha : A \vdash \alpha : A}{x \star \alpha : [x : A \vdash \alpha : A]}\,cut}{x \star \alpha : [x : A \vdash \delta : B,\ \alpha : A]}\,w}{x : A \vdash \mu\delta\ x \star \alpha : B,\ \alpha : A}\,\mu}{\vdash \quad A \to B,\ \alpha : A} \qquad \alpha : A \vdash \alpha : A}{f : (A \to B) \to A \vdash f : (A \to B) \to A \qquad (A \to B) \to A \vdash \alpha : A}\,cut}{f : (A \to B) \to A \vdash \alpha : A}\,\mu}{f : (A \to B) \to A \vdash \quad A}}{\vdash \quad ((A \to B) \to A) \to A}$$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \vdash x : A \qquad \alpha : A \vdash \alpha : A}{x \star \alpha : [x : A \vdash \alpha : A]}\,cut}{x \star \alpha : [x : A \vdash \delta : B,\ \alpha : A]}\,w}{x : A \vdash \mu\delta\ x \star \alpha : B,\ \alpha : A}\,\mu}{\vdash \mathbf{k}_\alpha : A \to B,\ \alpha : A} \qquad \alpha : A \vdash \alpha : A}{f : (A \to B) \to A \vdash f : (A \to B) \to A \qquad (A \to B) \to A \vdash \alpha : A}\,cut}{\dfrac{f : (A \to B) \to A \vdash \alpha : A}{f : (A \to B) \to A \vdash A}\,\mu}{\vdash ((A \to B) \to A) \to A}$$

where $\mathbf{k}_\sigma \overset{\text{def}}{=} \lambda x\, \mu\delta\ x \star \sigma$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \vdash x : A \qquad \alpha : A \vdash \alpha : A}{x \star \alpha : [x : A \vdash \alpha : A]}\,cut}{x \star \alpha : [x : A \vdash \delta : B,\ \alpha : A]}\,w}{x : A \vdash \mu\delta\ x \star \alpha : B,\ \alpha : A}\,\mu}{\vdash \mathbf{k}_\alpha : A \to B,\ \alpha : A} \qquad \dfrac{}{\alpha : A \vdash \alpha : A}}{f : (A \to B) \to A \vdash f : (A \to B) \to A \qquad \mathbf{k}_\alpha.\alpha : (A \to B) \to A \vdash \alpha : A}\,cut}{f : (A \to B) \to A \vdash \alpha : A}\,\mu$$

$$\dfrac{f : (A \to B) \to A \vdash A}{\vdash ((A \to B) \to A) \to A}$$

where $\mathbf{k}_\sigma \stackrel{\text{def}}{=} \lambda x\, \mu\delta\ x \star \sigma$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \vdash x : A \qquad \alpha : A \vdash \alpha : A}{x \star \alpha : [x : A \vdash \alpha : A]}\, cut}{x \star \alpha : [x : A \vdash \delta : B,\ \alpha : A]}\, w}{x : A \vdash \mu\delta\ x \star \alpha : B,\ \alpha : A}\, \mu}{\vdash \mathbf{k}_\alpha : A \to B,\ \alpha : A} \qquad \dfrac{\alpha : A \vdash \alpha : A}{\mathbf{k}_\alpha.\alpha : (A \to B) \to A \vdash \alpha : A}}{f : (A \to B) \to A \vdash f : (A \to B) \to A}\, cut}{\dfrac{f \star \mathbf{k}_\alpha.\alpha : [f : (A \to B) \to A \vdash \alpha : A]}{f : (A \to B) \to A \vdash A}\, \mu}{\vdash ((A \to B) \to A) \to A}$$

where $\mathbf{k}_\sigma \stackrel{\text{def}}{=} \lambda x\, \mu\delta\ x \star \sigma$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \vdash x : A \qquad \alpha : A \vdash \alpha : A}{x \star \alpha : [x : A \vdash \alpha : A]}\, cut}{x \star \alpha : [x : A \vdash \delta : B,\ \alpha : A]}\, w}{x : A \vdash \mu\delta\ x \star \alpha : B,\ \alpha : A}\, \mu}{\vdash \mathbf{k}_\alpha : A \to B,\ \alpha : A \qquad \alpha : A \vdash \alpha : A}}{f : (A \to B) \to A \vdash f : (A \to B) \to A \qquad \mathbf{k}_\alpha.\alpha : (A \to B) \to A \vdash \alpha : A}\, cut$$

$$\dfrac{\dfrac{f \star \mathbf{k}_\alpha.\alpha : [f : (A \to B) \to A \vdash \alpha : A]}{f : (A \to B) \to A \vdash \mu\alpha\ f \star \mathbf{k}_\alpha.\alpha : A}\, \mu}{\vdash \quad ((A \to B) \to A) \to A}$$

where $\mathbf{k}_\sigma \stackrel{\text{def}}{=} \lambda x\, \mu\delta\ x \star \sigma$

Exercise: from Peirce Law to its $\lambda\mu\sigma$-term

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \vdash x : A \qquad \alpha : A \vdash \alpha : A}{x \star \alpha : [x : A \vdash \alpha : A]}\,cut}{x \star \alpha : [x : A \vdash \delta : B,\ \alpha : A]}\,w}{x : A \vdash \mu\delta\ x \star \alpha : B,\ \alpha : A}\,\mu}{\vdash \mathbf{k}_\alpha : A \to B,\ \alpha : A} \qquad \dfrac{\alpha : A \vdash \alpha : A}{\mathbf{k}_\alpha.\alpha : (A \to B) \to A \vdash \alpha : A}}{f : (A \to B) \to A \vdash f : (A \to B) \to A \qquad f \star \mathbf{k}_\alpha.\alpha : [f : (A \to B) \to A \vdash \alpha : A]}\,cut}{f : (A \to B) \to A \vdash \mu\alpha\ f \star \mathbf{k}_\alpha.\alpha : A}\,\mu}{\vdash \mathbf{cc} : ((A \to B) \to A) \to A}$$

where $\mathbf{k}_\sigma \stackrel{\text{def}}{=} \lambda x\, \mu\delta\ x \star \sigma$ and $\mathbf{cc} \stackrel{\text{def}}{=} \lambda f\, \mu\alpha\ f \star \mathbf{k}_\alpha.\alpha$.

Classical realizability: Orthogonality

Let $\perp\!\!\!\perp$ be any set of c-terms closed under $=_{\beta\gamma\sigma}$.

Notations:

Λ: set of the λ-terms $\quad$ Σ: set of the σ-terms $\quad$ $t \perp\!\!\!\perp \sigma \overset{\text{def}}{\Leftrightarrow} t \star \sigma \in \perp\!\!\!\perp$

For all $L \subseteq \Lambda$, $S \subseteq \Sigma$:

- $L \perp\!\!\!\perp S \overset{\text{def}}{\Leftrightarrow} \forall t \in L \, \forall \sigma \in S \quad t \perp\!\!\!\perp \sigma$
- $L^{\perp\!\!\!\perp} \overset{\text{def}}{=} \{\sigma \in \Sigma \; ; \; \forall t \in L \;\; t \perp\!\!\!\perp \sigma\}$, $\quad S^{\perp\!\!\!\perp} \overset{\text{def}}{=} \{t \in \Lambda \; ; \; \forall \sigma \in S \;\; t \perp\!\!\!\perp \sigma\}$
- $L.S \overset{\text{def}}{=} \{t.\sigma \; ; \; t \in S \text{ and } \sigma \in S\}$
- L is said *classical* if(f) $L^{\perp\!\!\!\perp\perp\!\!\!\perp} = L$, equivalently: $L \in \mathcal{P}(\Sigma)^{\perp\!\!\!\perp}$ (i.e. L of the form $S^{\perp\!\!\!\perp}$), because $S^{\perp\!\!\!\perp\perp\!\!\!\perp\perp\!\!\!\perp} = S^{\perp\!\!\!\perp}$ for all S.

Classical realizability: Sorted formulas

2^{nd} order formulas defined as previously, but now using two sorts of predicate variables:

- *intuitionistic* ones as before, that we now denote ${}^{i}X$, ${}^{i}Y$, ${}^{i}Z \ldots$
- *classical* ones, that we denote $X, Y, Z \ldots$

Intuitionistic predicate variables are meant for any $P \in \mathcal{P}_\beta(\Lambda)^{\mathcal{U}^n}$ as before whereas classical predicate variables are meant for $P \in (\mathcal{P}(\Sigma)^{\perp\!\!\!\perp})^{\mathcal{U}^n}$ only.

In particular, from now on 2^{nd} order formula closures $\overline{\cdot}$ are assumed to map the classical predicate variables to functions ranging over $\mathcal{P}(\Sigma)^{\perp\!\!\!\perp}$ only.

A 2^{nd} order formula is said *classical* if its rightmost predicate variable occurrence is classical and *intuitionistic* otherwise.

Classical realizability: Semantics & Typing rules

- $|P(\vec{u})| = P(\vec{u})$
- $|A \to B| = |A| \to |B|$
- $|\forall x\, A| = \bigcap_{u \in \mathcal{U}} |A[u/x]|$
- $|\forall^i X^n\, A| = \bigcap_{P \in \mathcal{P}_\beta(\Lambda)^{\mathcal{U}^n}} |A[{}^iX := P]|$
- $|\forall X^n\, A| = \bigcap_{P \in (\mathcal{P}(\Sigma)^\perp)^{\mathcal{U}^n}} |A[X := P]|$

- $\|P(\vec{u})\| = P(\vec{u})^{\perp\!\!\!\perp}$
- $\|A \to B\| = |A|.\|B\|$
- $\|\forall x\, A\| = \bigcup_{u \in \mathcal{U}} \|A[u/x]\|$
- $\|\forall^i X^n\, A\| = \bigcup_{P \in \mathcal{P}_\beta(\Lambda)^{\mathcal{U}^n}} \|A[{}^iX := P]\|$
- $\|\forall X^n\, A\| = \bigcup_{P \in (P(\Sigma)^\perp)^{\mathcal{U}^n}} \|A[X := P]\|$

$$\alpha : A \vdash \alpha : A$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : B \vdash \Delta'}{\Gamma, \Gamma', t.\sigma : A \to B \vdash \Delta, \Delta'}\ l{\to}i$$

$$\frac{\Gamma, t : A[u/x] \vdash \Delta}{\Gamma, t : \forall x\, A \vdash \Delta}\ l\forall i$$

$$\frac{\Gamma, t : A[F/{}^iX\vec{x}] \vdash \Delta}{\Gamma, t : \forall^i X\, A \vdash \Delta}\ l\forall^i i$$

$$\frac{\Gamma, t : A[F/X\vec{x}] \vdash \Delta}{\Gamma, t : \forall X\, A \vdash \Delta}\ l\forall^c i^\ddagger$$

$$\frac{\Gamma \vdash t : A, \Delta \quad \Gamma', \sigma : A \vdash \Delta'}{t \star \sigma : [\Gamma, \Gamma' \vdash \Delta, \Delta']}\ cut$$

$$x : A \vdash x : A$$

$$\frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma \vdash \lambda x\, t : A \to B, \Delta}\ r{\to}i$$

$$\frac{\Gamma \vdash t : A, \Delta}{\Gamma \vdash t : \forall x\, A, \Delta}\ r\forall i^\dagger$$

$$\frac{\Gamma \vdash t : A, \Delta}{\Gamma \vdash t : \forall^i X\, A, \Delta}\ r\forall^i i^\dagger$$

$$\frac{\Gamma \vdash t : A, \Delta}{\Gamma \vdash t : \forall X\, A, \Delta}\ r\forall^c i^\dagger$$

$$\frac{c : [\Gamma \vdash \alpha : F, \Delta]}{\Gamma \vdash \mu\alpha\, c : F, \Delta}\ \mu^\ddagger$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ w$$

$$\frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}\ {\to}e$$

$$\frac{\Gamma \vdash t : \forall x\, A, \Delta}{\Gamma \vdash t : A[u/x], \Delta}\ \forall e$$

$$\frac{\Gamma \vdash t : \forall^i X\, A, \Delta}{\Gamma \vdash t : A[F/{}^iX\vec{x}], \Delta}\ \forall^i e$$

$$\frac{\Gamma \vdash t : \forall X\, A, \Delta}{\Gamma \vdash t : A[F/X\vec{x}], \Delta}\ \forall^c e^\ddagger$$

† only if x (resp. ${}^iX$, X) $\notin \Gamma, \Delta$

‡ only if F is classical

Classical realizability: Adequacy lemma

Remark $\left|\overline{F}\right|$ is classical for every classical formula F and every closure $\overline{\cdot}$ because of the relations: $L \to S^{\perp\!\!\!\perp} = (L.S)^{\perp\!\!\!\perp}$, $\bigcap_{i\in I} S_i^{\perp\!\!\!\perp} = (\bigcup_{i\in I} S_i)^{\perp\!\!\!\perp}$.

Orthogonality lemma For every closed formula A: $\left|A\right| \perp\!\!\!\perp \left\|A\right\|$

Substitution lemma If $P \in \mathcal{P}_\beta(\Lambda)^{\mathcal{U}^n}$ is defined by: $P(\vec{u}) = \left|\overline{F[\vec{u}/\vec{x}]}\right|$ then

$$\left|\overline{A[F/X\vec{x}]}\right| = \left|\overline{A[X:=P]}\right| \quad \text{and} \quad \left\|\overline{A[F/X\vec{x}]}\right\| = \left\|\overline{A[X:=P]}\right\|.$$

Adequacy lemma For any closure $\overline{\cdot}$ and all $t_1 \in \left|\overline{C_1}\right|, \dots, t_n \in \left|\overline{C_n}\right|$, $\sigma_1 \in \left\|\overline{D_1}\right\|, \dots, \sigma_p \in \left\|\overline{D_p}\right\|$, $T[x_1 := t_1, \dots, x_n := t_n, \alpha_1 := \sigma_1, \dots, \alpha_p := \sigma_p]$ belongs to:

- $\left|\overline{A}\right|$ if $x_1 : C_1, \dots, x_n : C_n \vdash T : A, \alpha_1 : D_1, \dots, \alpha_n : D_n$
- $\perp\!\!\!\perp$ if $T : [\, x_1 : C_1, \dots, x_n : C_n \vdash \alpha_1 : D_1, \dots, \alpha_n : D_n \,]$
- $\left\|\overline{A}\right\|$ if $x_1 : C_1, \dots, x_n : C_n, T : A \vdash \alpha_1 : D_1, \dots, \alpha_n : D_n$

Exercise 7

1. Prove the orthogonality lemma of the previous slide.
2. Assume the substitution lemma and prove the adequacy lemma of the previous slide.

Classical realizability: Data correctness problem

. . . We now have two sorts of numeral types:

$$^iN(u) \stackrel{\text{def}}{=} \forall^i X\,(\forall z\,(^iX(z) \to {}^iX(sz)) \to {}^iX(0) \to {}^iX(u))$$
$$N(u) \stackrel{\text{def}}{=} \forall X\,(\forall z\,(X(z) \to X(sz)) \to X(0) \to X(u))$$

Fact: $|^iN(u)| \subset |N(u)|$ and $N(u)$ contains a lot of λ-terms not reducible to Church numerals, e.g. $\lambda f\lambda x\mu\alpha\, f(\mu\beta\, f(f(\mu\delta\, fx \star \beta)) \star \alpha) \star \alpha \in |N(2)|$.
$\vdash \mathbf{I} : \forall x\,(^iN(x) \to N(x))$ but $\vdash\ ? : \forall x\,(N(x) \to {}^iN(x))$ impossible.

Problem 1: How are we going to recycle all our intuitionistic programming (i.e. λ-terms of types $\forall x\,(^iN(x) \to {}^iN(f(x)))$) in this classical system?

Solution: $\vdash T : \forall X\,(\forall x\,(^iN(x) \to X(x)) \to \forall x\,(N(x) \to X(x)))$ where $T = \lambda f\lambda x\, x(\lambda g\lambda y\, g(\mathrm{Succ}\, y))f\,\overline{0}$, **Succ** being any λ-term such that $\vdash \mathbf{Succ} : \forall x\,(^iN(x) \to {}^iN(sx))$

$\rightsquigarrow$ $\vdash t : \forall x\,(^iN(x) \to N(f(x)))$ $\Rightarrow$ $\vdash Tt : \forall x\,(N(x) \to N(f(x)))$
. . . but the result is still in the undecipherable set $|N(f(n))|$.

Exercise 8

Derive the typing stated in the previous slide, i.e.

$$\vdash T : \forall X\big(\forall x\,({}^{i}N(x) \to X(x)) \to \forall x\,(N(x) \to X(x))\big),$$

where

$$T = \lambda f \lambda x\, x(\lambda g \lambda y\, g(\mathbf{Succ}\, y))f\,\overline{0},$$

with the help of the set $\mathcal{E}$ of equations of the "very simple example" and the two additional equational derivation rules (see Section 3 about intuitionistic realizability).
No need to detail again the subterm **Succ** found in Exercise 6.

[Hint: First derive
$\vdash \lambda g \lambda y\, g(\mathbf{Succ}\, y) : \forall z\big(({}^{i}N(x-z) \to X(x)) \to ({}^{i}N(x-sz) \to X(x))\big)$]

Classical realizability: Data correctness (a last trick)

Problem 2: How are we going to read the result in $|N(f(n))|$?
Recall that in the purely intuitionistic system, it was possible because

$$r \in |^i N(f(n))| \;\Rightarrow\; r =_{\beta\eta} \overline{f(n)}.$$

Solution: We let $\perp\!\!\!\perp = \{c \;;\; \exists t \in |^i N(f(n))| \quad \exists \sigma \in \Sigma \quad c =_{\beta\gamma\sigma} k \star t.\sigma\}$.
In other words, we let $\perp\!\!\!\perp$ guess the solution $|^i N(f(n))|$! It is possible because $|^i N(f(n))|$ does not depend on the choice of $\perp\!\!\!\perp$ and because although we pretended to have fixed $\perp\!\!\!\perp$, we never said how.

In that way, we get $k \in |^i N(f(n)) \to \bot|$ where $\bot = \Sigma^{\perp\!\!\!\perp}$. Since $\bot$ is *classical*, we obtain $Tk \in |N(f(n)) \to \bot|$. Hence for all $r \in |N(f(n))|$, $Tkr \in |\bot| = \Sigma^{\perp\!\!\!\perp}$. It then follows for all $\pi \in \Sigma$: $Tkr \star \pi \in \perp\!\!\!\perp$, i.e. for some $\sigma \in \Sigma$: $Tkr \star \pi =_{\beta\eta\gamma\sigma} k \star \overline{f(n)}.\sigma$. To sum up:

$$r \in |N(f(n))|,\ \pi \in \Sigma \;\Rightarrow\; \exists \sigma \in \Sigma \quad Tkr \star \pi =_{\beta\eta\gamma\sigma} k \star \overline{f(n)}.\sigma.$$

CL formulation of $\lambda\mu\sigma$-calculus

Translation $\lambda\mu\sigma \xrightarrow{\{\,\}}$ CL not quite possible by using only λ-terms.
Possible by using λ-terms and σ-terms.

$$\mathbf{cc} \stackrel{\text{def}}{=} \lambda f\, \mu\alpha\ \ f \star \mathbf{k}_\alpha.\alpha \qquad \mathbf{k}_\sigma \stackrel{\text{def}}{=} \lambda x\, \mu\delta\ \ x \star \sigma$$

Proposition Every λ-term of the $\lambda\mu\sigma$-calculus is $\beta\gamma\sigma$-equal to an applicative combination of $\mathbf{K}, \mathbf{S}, \mathbf{cc}$ and $\mathbf{k}_\alpha$ for every σ-variable α.

Translation:

- $\{x\} = x$
- $\{tu\} = \{t\}\{u\}$
- $\{\lambda x\, t\} = \lambda^* x\, \{t\}$ where λ^* is defined as for intuitionistic λ-calculus
- $\{\mu\alpha\, c\} = \mathbf{cc}(\lambda^*\alpha\, \{c\})$ where λ^* is the same no matter term sorts
- $\{t \star u_1.u_2 \ldots u_n.\alpha\} = \alpha(\{t\}\{u_1\}\{u_2\} \ldots \{u_n\})$

Then for every λ-term t with free classical variables $\alpha_1, \ldots, \alpha_n$:

$$\{t\}[\alpha_1 := \mathbf{k}_{\alpha_1}, \ldots, \alpha_n := \mathbf{k}_{\alpha_n}] =_{\beta\gamma\sigma} t$$

Corollary Every *closed* λ-term of the $\lambda\mu\sigma$-calculus is $\beta\gamma\sigma$-equal to an applicative combination of $\mathbf{K}, \mathbf{S}, \mathbf{cc}$.

Exercise 9

Prove what is stated in the previous slide, i.e. that for every λ-term t of $\lambda\mu\sigma$-calculus with free classical variables $\alpha_1, \ldots, \alpha_n$:

$$\{t\}[\alpha_1 := \mathbf{k}_{\alpha_1}, \ldots, \alpha_n := \mathbf{k}_{\alpha_n}] =_{\beta\gamma\sigma} t$$

Classical combinatory logic CLσ

. . . but CL normalization of these terms requires terms of the form k_σ.
This naturally comes from the structure of the λ-term **cc**:

$$\mathbf{cc} \stackrel{\text{def}}{=} \lambda f\, \mu\alpha\ \ f \star \mathbf{k}_\alpha.\alpha \qquad \mathbf{k}_\sigma \stackrel{\text{def}}{=} \lambda x\, \mu\delta\ \ x \star \sigma$$

Syntax of CLσ:

$$\lambda\text{-terms:}\quad t = x \mid K \mid S \mid \mathsf{cc} \mid tt \mid \mathsf{k}_\sigma$$

$$\sigma\text{-terms:}\quad \sigma = \alpha \mid t.\sigma$$

Reduction rules:

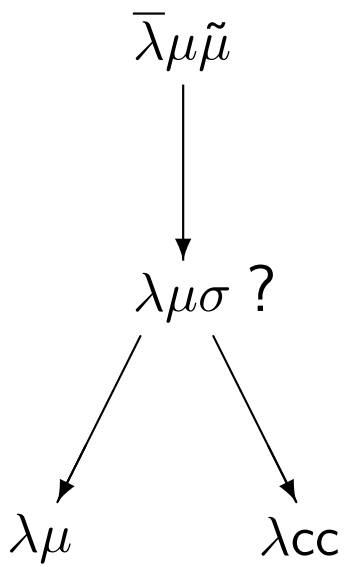
$$\mathsf{cc}\, t \star \sigma \to t\, \mathsf{k}_\sigma \star \sigma$$
$$\mathsf{k}_\sigma\, t \star \pi \to t \star \sigma$$
$$t \star u.\sigma = tu \star \sigma$$

. . . together with the intuitionistic rules:

$$Ktu \to t$$
$$Stuv \to tv(uv)$$

What is $\lambda\mu\sigma$-calculus?

No occurrence of $\lambda\mu\sigma$ in the literature. σ is for several S.



Small fragment of Herbelin's $\overline{\lambda}\mu\tilde{\mu}$-calculus

Slack version of Parigot's $\lambda\mu$-calculus

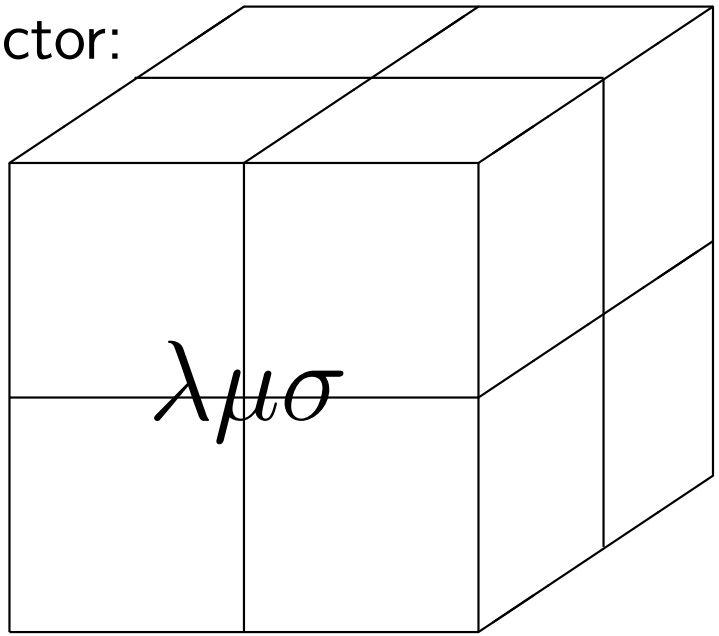
Study-oriented calculus

$\lambda\mu\sigma$-calculus is indeed the lub of two representative classical dialects: $\lambda\mu$-calculus and Krivine's λcc-calculus.

A classical λ-cube

① Removing { stack constructor:
application:

② Classical features in CL { NO:
YES:

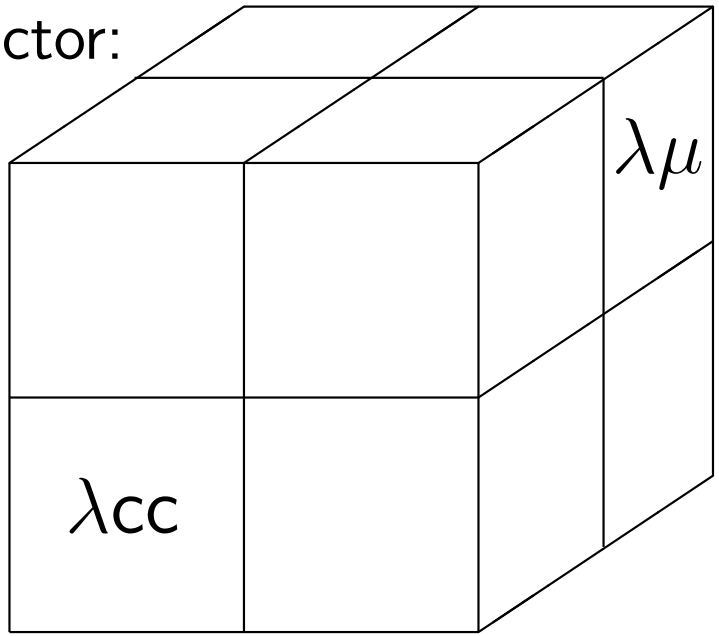


③ Specializing reduction to left reduction

Next: two opposite corners of our classical λ-cube

① Removing { stack constructor: / application:

② Classical features in CL { NO: / YES: λcc



③ Specializing reduction to left reduction

$\lambda\mu$-calculus: a purebred classical λ-calculus

By choosing the negative settings:

1. removal of the stack constructor (corresponding to $l \longrightarrow i$-rule) which is a redundancy of the application (corresponding to $r \longrightarrow e$-rule),
2. no CL-like constants for the classical features of the calculus,
3. no specialization of the normalization to any reduction strategy,

$\lambda\mu$-calculus sticks to traditional λ-calculus style:

- just a new pair abstractor/application $(\mu, \star)$ in addition to the intuitionistic one $(\lambda, \mathit{application})$ to treat the classical features.
- $\lambda\mu$-calculus remains an undeterministic rewrite system enjoying CR property.

$\lambda\mu$-calculus: disappearance of the stack constructor

In order to remove the stack constructor from $\lambda\mu\sigma$-calculus, we only have to orient the σ-rule as follows: $t \star u.\sigma \to_\sigma tu \star \sigma$.

- σ-reduction $\twoheadrightarrow_\sigma$ has the SN property.
- The λ-terms and c-terms in σ-normal form are exactly the ones with no occurrence of the stack constructor.

$\rightsquigarrow$ In $\lambda\mu$-calculus, only λ-terms and c-terms in σ-normal form are written.

Problem: γ-reductions usually stuck by σ-normalization, e.g.
$c[\alpha := u.\beta] \;{}_\gamma\!\leftarrow (\mu\alpha\, c) \star u.\beta \twoheadrightarrow_\sigma ((\mu\alpha\, c) \star u.\beta)^{\sigma\text{-}nf} = (\mu\alpha\, c^{\sigma\text{-}nf})u \star \beta \to ?$
This can only be overcome by allowing an exceptional σ-conversion in the "wrong" way: $(\mu\alpha\, c)u =_\theta \mu\beta\, (\mu\alpha\, c)u \star \beta \to_{\sigma^{-1}} \mu\beta\, (\mu\alpha\, c) \star u.\beta$
$\to_\gamma \mu\beta\, c[\alpha := u.\beta] \twoheadrightarrow_\sigma \mu\beta\, (c[\alpha := u.\beta])^{\sigma\text{-}nf}$

In $\lambda\mu$-calculus, the last reduction sequence is viewed as a single reduction step called *μ-reduction*:

$$(\mu\alpha\, c)u \to_\mu \mu\beta\, (c[\alpha := u.\beta])^{\sigma\text{-}nf}$$

$\lambda\mu$-calculus: σ-terms as **S**treams

In $\lambda\mu$-calculus, every σ-term is just a variable playing the role of a stream of λ-terms fed by the μ abstractor that binds it:

$$\begin{aligned}(\mu\alpha\, c)u_1 u_2 \dots u_n \star \beta &\to_\mu (\mu\alpha\, (c[\alpha := u_1.\alpha])^{\sigma\text{-}nf})u_2 \dots u_n \star \beta \\ &\to_\mu (\mu\alpha\, (c[\alpha := u_1.u_2.\alpha])^{\sigma\text{-}nf})u_3 \dots u_n \star \beta \\ &\vdots \\ &\to_\mu (\mu\alpha\, (c[\alpha := u_1 \dots u_n.\alpha])^{\sigma\text{-}nf}) \star \beta\end{aligned}$$

In order to finish the work done by a γ-reduction of $\lambda\mu\sigma$-calculus, i.e. up to σ-conversions: $(\mu\alpha\, c)u_1 \dots u_n \star \beta \to_\gamma (c[\alpha := u_1.u_2 \dots u_n.\beta])^{\sigma\text{-}nf}$, we have to add to the calculus the particular case of γ-reduction where the substituted σ-term is a variable:

$$(\mu\alpha\, c) \star \beta \to_\rho c[\alpha := \beta]$$

which is a stream redirecting.

At last, θ-rule is oriented in $\lambda\mu$-calculus as follows:

$$\mu\alpha\, t \star \alpha \to_\theta t \quad \text{only if } \alpha \notin t$$

and becomes a stream closure.

$\lambda\mu$-calculus: complete picture

Syntax:

$$\begin{array}{lrcl} \lambda\text{-terms:} & t & = & x \mid tt \mid \lambda x\, t \mid \mu\alpha\, c \\ c\text{-terms:} & c & = & t \star \sigma \\ \sigma\text{-terms:} & \sigma & = & \alpha \end{array}$$

Reduction rules:

$$\begin{array}{rcl} (\lambda x\, t)u & \to_\beta & t[x := u] \\ (\mu\alpha\, c)u & \to_\mu & \mu\beta\, (c[\alpha := u.\beta])^{\sigma\text{-}nf} \\ (\mu\alpha\, c) \star \beta & \to_\rho & c[\alpha := \beta] \\ \mu\alpha\, t \star \alpha & \to_\theta & t \quad \text{only if } \alpha \notin t \end{array}$$

Actual notations & wording about $\lambda\mu$-calculus in the literature:

- The only σ-terms existing in $\lambda\mu$-calculus, i.e. σ-variables, are called *μ-variables* or *classical variables* and are never considered as terms on their own.
- The *c*-terms are called *named terms* and written $[\alpha]t$ instead of $t \star \alpha$.

λcc-calculus: an imperative calculus

By choosing the opposite settings:

1. removal of the application (corresponding to $r{\longrightarrow}e$-rule) instead of the stack constructor (corresponding to $l{\longrightarrow}i$-rule),
2. classical features in CL,
3. normalization only performed by left reduction on c-terms (i.e. by rewriting at every step the leftmost rewritable subterm),

λcc-calculus is quite different from $\lambda\mu$-calculus.

Because of this choice of the left reduction, λcc-calculus is a deterministic rewrite system (in which CR property is therefore meaningless) and looks rather like an imperative programming language than a λ-calculus.

λcc-calculus: an imperative calculus

By choosing the opposite settings:

1. removal of the application (corresponding to $r{\longrightarrow}e$-rule) instead of the stack constructor (corresponding to $l{\longrightarrow}i$-rule),
2. classical features in CL,
3. normalization only performed by left reduction on c-terms (i.e. by rewriting at every step the leftmost rewritable subterm),

λcc-calculus is quite different from $\lambda\mu$-calculus.

Because of this choice of the left reduction, λcc-calculus is a deterministic rewrite system (in which CR property is therefore meaningless) and looks rather like an imperative programming language than a λ-calculus.

Huge contradiction between (1) and (2): CL requires application!
$\rightsquigarrow$ applications will just be *morally* removed, not formally because of (2).

λcc-calculus: trying to combine ① with ②

① We may indeed replace every application by a stack constructor: $tu =_\theta \mu\alpha\, tu \star \alpha =_\sigma \mu\alpha\, t \star u.\alpha$ and keep application just as a meta-notation for short:

$$tu \stackrel{\text{def}}{=} \mu\alpha\, t \star u.\alpha$$

But we then must reformulate β-rule without mention of a primitive application:

$$\lambda x\, t \star u.\sigma \to_{\text{pop}} t[x := u] \star \sigma$$

(similar to $\lambda\mu\sigma$-calculus: $\lambda x\, t \star u.\sigma =_\sigma (\lambda x\, t)u \star \sigma \to_\beta t[x := u] \star \sigma$)

② Now all the μ's of the resulting calculus without stack constructor would be hidden in λ-terms $\mathbf{cc}$, $\mathbf{k}_\sigma$ by replacing every λ-term t with $\{t\}[f_{\alpha_1} := \mathbf{k}_{\alpha_1}, f_{\alpha_2} := \mathbf{k}_{\alpha_2}, \ldots]$ $(=_{\text{pop}\gamma} t)$ where $\{t\}$ is defined by:

- $\{x\} = x$
- $\{\lambda x\, t\} = \lambda x\, \{t\}$
- $\{\mu\alpha\, c\} = \mathbf{cc}(\lambda f_\alpha\, \{c\})$
- $\{t \star u_1.u_2 \ldots u_n.\alpha\} = f_\alpha(\{t\}\{u_1\}\{u_2\} \ldots \{u_n\})$

λcc-calculus: trying to combine ① with ②

① We may indeed replace every application by a stack constructor: $tu =_\theta \mu\alpha\, tu \star \alpha =_\sigma \mu\alpha\, t \star u.\alpha$ and keep application just as a meta-notation for short:

$$tu \stackrel{\text{def}}{=} \mu\alpha\, t \star u.\alpha$$

But we then must reformulate β-rule without mention of a primitive application:

$$\lambda x\, t \star u.\sigma \to_{\text{pop}} t[x := u] \star \sigma$$

(similar to $\lambda\mu\sigma$-calculus: $\lambda x\, t \star u.\sigma =_\sigma (\lambda x\, t)u \star \sigma \to_\beta t[x := u] \star \sigma$)

② Now all the μ's of the resulting calculus without stack constructor would be hidden in λ-terms $\mathbf{cc}$, $\mathbf{k}_\sigma$ by replacing every λ-term t with $\{t\}[f_{\alpha_1} := \mathbf{k}_{\alpha_1}, f_{\alpha_2} := \mathbf{k}_{\alpha_2}, \ldots]$ $(=_{\text{pop}\gamma} t)$ where $\{t\}$ is defined by:

- $\{x\} = x$
- $\{\lambda x\, t\} = \lambda x\, \{t\}$
- $\{\mu\alpha\, c\} = \mathbf{cc}(\lambda f_\alpha\, \{c\})$
- $\{t \star u_1.u_2 \ldots u_n.\alpha\} = f_\alpha(\{t\}\{u_1\}\{u_2\} \ldots \{u_n\})$

. . . The μ's not all hidden in λ-terms $\mathbf{cc}$, $\mathbf{k}_\sigma$, also hidden in the applicative meta-notation (used in the last two clauses).

λcc-calculus: σ-terms as **S**tacks

Conclusion: Ok, we really have to reintroduce a primitive application, but we just let it play the role of our meta-notation, i.e. according to choice ③ (= left reduction on c-terms only):

$$tu \star \sigma \overset{\text{def}}{=} (\mu\alpha\ t \star u.\alpha) \star \sigma \to_\gamma t \star u.\sigma$$

$$\mathbf{cc} \star t.\sigma \overset{\text{def}}{=} (\lambda f\ \mu\alpha\ \ f \star \mathbf{k}_\alpha.\alpha) \star t.\sigma \to_{\mathsf{pop}} (\mu\alpha\ \ t \star \mathbf{k}_\alpha.\alpha) \star \sigma \to_\gamma t \star \mathbf{k}_\sigma.\sigma$$

$$\mathbf{k}_\sigma \star t.\pi \overset{\text{def}}{=} (\lambda x\ \mu\delta\ \ x \star \sigma) \star t.\pi \to_{\mathsf{pop}} (\mu\delta\ \ t \star \sigma) \star \pi \to_\gamma t \star \sigma$$

Now, replace applicative meta-notation back by a primitive application, **cc** by a constant cc, $\mathbf{k}_\sigma$ by k_σ where k is a new σ-term$\to\lambda$-term construct and view the above reduction sequences as primitive reduction rules.

λcc-calculus: σ-terms as **S**tacks

Conclusion: Ok, we really have to reintroduce a primitive application, but we just let it play the role of our meta-notation, i.e. according to choice ③ (= left reduction on c-terms only):

Push:

$$t u \star \sigma \stackrel{\text{def}}{=} (\mu\alpha\, t \star u.\alpha) \star \sigma \to_\gamma t \star u.\sigma$$

Save current stack:

$$\mathbf{cc} \star t.\sigma \stackrel{\text{def}}{=} (\lambda f\, \mu\alpha\ \ f \star \mathbf{k}_\alpha.\alpha) \star t.\sigma \to_{\mathsf{pop}} (\mu\alpha\ \ t \star \mathbf{k}_\alpha.\alpha) \star \sigma \to_\gamma t \star \mathbf{k}_\sigma.\sigma$$

Restore stack:

$$\mathbf{k}_\sigma \star t.\pi \stackrel{\text{def}}{=} (\lambda x\, \mu\delta\ \ x \star \sigma) \star t.\pi \to_{\mathsf{pop}} (\mu\delta\ \ t \star \sigma) \star \pi \to_\gamma t \star \sigma$$

Now, replace applicative meta-notation back by a primitive application, **cc** by a constant cc, $\mathbf{k}_\sigma$ by k_σ where k is a new σ-term$\to\lambda$-term construct and view the above reduction sequences as primitive reduction rules.

This is λcc-calculus.

$\lambda\mu$-calculus v. λcc-calculus

$\lambda\mu$-calculus

$t \star u.\sigma \to_\sigma tu \star \sigma$ at once

λ-terms: $t = x \mid tt \mid \lambda x\, t \mid \mu\alpha\, c$

c-terms: $c = t \star \sigma$

σ-terms: $\sigma = \alpha$

Reduction rules:

$(\lambda x\, t)u \to_\beta t[x := u]$

$(\mu\alpha\, c)u \to_\mu \mu\beta\, (c[\alpha := u.\beta])^{\sigma\text{-}nf}$

$(\mu\alpha\, c) \star \beta \to_\rho c[\alpha := \beta]$

$\mu\alpha\, t \star \alpha \to_\theta t$ only if $\alpha \notin t$

λcc-calculus

$tu \star \sigma \to_{\sigma^{-1}} t \star u.\sigma$

λ-terms: $t = x \mid \mathsf{cc} \mid tt \mid \lambda x\, t \mid \mathsf{k}_\sigma$

c-terms: $c = t \star \sigma$

σ-terms: $\sigma = \alpha \mid t.\sigma$

Reduction rules:

$\lambda x\, t \star u.\sigma \to_{\mathsf{pop}} t[x := u] \star \sigma$

$tu \star \sigma \to_{\mathsf{push}} t \star u.\sigma$

$\mathsf{cc} \star t.\sigma \to_{\mathsf{save}} t \star \mathsf{k}_\sigma.\sigma$

$\mathsf{k}_\sigma \star t.\pi \to_{\mathsf{restore}} t \star \sigma$

$\lambda\mu$-calculus v. λcc-calculus: Minimal typing systems complete for minimal classical logic

$\lambda\mu$-calculus

$$x : A \vdash x : A$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

$$\frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}$$

$$\frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma, \vdash \lambda x\, t : A \to B, \Delta}$$

$$\frac{\Gamma \vdash t : A, \Delta}{t \star \alpha : [\Gamma \vdash \alpha : A, \Delta]}$$

$$\frac{c : [\Gamma \vdash \alpha : A, \Delta]}{\Gamma \vdash \mu\alpha\, c : A, \Delta}$$

λcc-calculus

$$x : A \vdash x : A$$

$$\frac{\Gamma \vdash t : C}{\Gamma, \Gamma' \vdash t : C}$$

$$\frac{\Gamma \vdash t : A \to B \quad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash tu : B}$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma, \vdash \lambda x\, t : A \to B}$$

$$\vdash \mathsf{cc} : ((A \to B) \to A) \to A$$

$\lambda\mu$-calculus v. λcc-calculus: Minimal typing systems complete for minimal classical logic

$\lambda\mu$-calculus

$$x : A \vdash x : A$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

$$\frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}$$

$$\frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma, \vdash \lambda x\, t : A \to B, \Delta}$$

$$\frac{\Gamma \vdash t : A, \Delta}{t \star \alpha : [\Gamma \vdash \alpha : A, \Delta]}$$

$$\frac{c : [\Gamma \vdash \alpha : A, \Delta]}{\Gamma \vdash \mu\alpha\, c : A, \Delta}$$

. . . also a maximal typing system:
one derivation rule for every term construct.

λcc-calculus

$$x : A \vdash x : A$$

$$\frac{\Gamma \vdash t : C}{\Gamma, \Gamma' \vdash t : C}$$

$$\frac{\Gamma \vdash t : A \to B \quad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash tu : B}$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma, \vdash \lambda x\, t : A \to B}$$

$$\vdash \mathsf{cc} : ((A \to B) \to A) \to A$$

$\lambda\mu$-calculus v. λcc-calculus: Minimal typing systems complete for minimal classical logic

$\lambda\mu$-calculus

$$x : A \vdash x : A$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

$$\frac{\Gamma \vdash t : A \to B, \Delta \quad \Gamma' \vdash u : A, \Delta'}{\Gamma, \Gamma' \vdash tu : B, \Delta, \Delta'}$$

$$\frac{\Gamma, x : A \vdash t : B, \Delta}{\Gamma, \vdash \lambda x\, t : A \to B, \Delta}$$

$$\frac{\Gamma \vdash t : A, \Delta}{t \star \alpha : [\Gamma \vdash \alpha : A, \Delta]}$$

$$\frac{c : [\Gamma \vdash \alpha : A, \Delta]}{\Gamma \vdash \mu\alpha\, c : A, \Delta}$$

. . . also a maximal typing system:
one derivation rule for every term construct.

λcc-calculus

$$x : A \vdash x : A$$

$$\frac{\Gamma \vdash t : C}{\Gamma, \Gamma' \vdash t : C}$$

$$\frac{\Gamma \vdash t : A \to B \quad \Gamma' \vdash u : A}{\Gamma, \Gamma' \vdash tu : B}$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma, \vdash \lambda x\, t : A \to B}$$

$$\vdash \mathsf{cc} : ((A \to B) \to A) \to A$$

. . . λcc-calculus shows how to execute proofs written in intuitionistic ND + Peirce Law!

Exercise 10

Answer these questions for each typing system of the previous slide:

1. Prove Peirce Law $((A \to B) \to A) \to A$ in the simplest way as possible, i.e. get a typing derivation of $\vdash t : ((A \to B) \to A) \to A$ where the λ-term t of $\lambda\mu$-calculus (resp. of λcc-calculus) is as small as possible. Compare this λ-term t with the λ-term $\mathbf{cc}$ of $\lambda\mu\sigma$-calculus.
2. Prove $(C \to A) \to ((C \to B) \to A) \to A$ in the simplest way as possible, i.e. get a typing derivation of $\vdash u : (C \to A) \to ((C \to B) \to A) \to A$ where the λ-term u is as small as possible.
3. Derive the typing $\vdash u\,\mathbf{I} : ((A \to B) \to A) \to A$ where u is the same λ-term as in previous question and $\mathbf{I} \stackrel{\text{def}}{=} \lambda x\, x$.
4. Normalize this λ-term $u\,\mathbf{I}$ within $\lambda\mu$-calculus (resp. within λcc-calculus) and compare its normal form with the λ-term t found at question 1.

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.
$\mathsf{cc} \star t^{(A \to B) \to A} . \sigma$

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.
$\mathsf{cc} \star t^{(A \to B) \to A} . \sigma$

cc *(lying)*: My proof of $((A \to B) \to A) \to A$ is fairly simple. I actually have a proof k_σ of $A \to B$, then just apply it your proof of $(A \to B) \to A$ to get A.

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.
$\mathsf{cc} \star t^{(A \to B) \to A} . \sigma$

cc *(lying)*: My proof of $((A \to B) \to A) \to A$ is fairly simple. I actually have a proof k_σ of $A \to B$, then just apply it your proof of $(A \to B) \to A$ to get A.
$\to t \star \mathsf{k}_\sigma^{A \to B} . \sigma$

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.
$\mathsf{cc} \star t^{(A \to B) \to A} . \sigma$

cc *(lying)*: My proof of $((A \to B) \to A) \to A$ is fairly simple. I actually have a proof k_σ of $A \to B$, then just apply it your proof of $(A \to B) \to A$ to get A.
$\to t \star \mathsf{k}_\sigma^{A \to B} . \sigma$

P *(after computing a while)*: You gave me a proof of $A \to B$ that I now must use.
I have brought a proof u of A. I want one of B in return.

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.
$\mathsf{cc} \star t^{(A \to B) \to A} . \sigma$

cc *(lying)*: My proof of $((A \to B) \to A) \to A$ is fairly simple. I actually have a proof k_σ of $A \to B$, then just apply it your proof of $(A \to B) \to A$ to get A.
$\to t \star \mathsf{k}_\sigma^{A \to B} . \sigma$

P *(after computing a while)*: You gave me a proof of $A \to B$ that I now must use. I have brought a proof u of A. I want one of B in return.
$\ldots \to \mathsf{k}_\sigma^{A \to B} \star u^A . \pi$

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.
$\mathsf{cc} \star t^{(A \to B) \to A} . \sigma$

cc *(lying)*: My proof of $((A \to B) \to A) \to A$ is fairly simple. I actually have a proof k_σ of $A \to B$, then just apply it your proof of $(A \to B) \to A$ to get A.
$\to t \star \mathsf{k}_\sigma^{A \to B} . \sigma$

P *(after computing a while)*: You gave me a proof of $A \to B$ that I now must use. I have brought a proof u of A. I want one of B in return.
$\ldots \to \mathsf{k}_\sigma^{A \to B} \star u^A . \pi$

cc: But you just needed a proof of A some time ago. Remember, precisely when we were in the context σ.

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.

$\mathsf{cc} \star t^{(A \to B) \to A} . \sigma$

cc *(lying)*: My proof of $((A \to B) \to A) \to A$ is fairly simple. I actually have a proof k_σ of $A \to B$, then just apply it your proof of $(A \to B) \to A$ to get A.

$\to t \star \mathsf{k}_\sigma^{A \to B} . \sigma$

P *(after computing a while)*: You gave me a proof of $A \to B$ that I now must use. I have brought a proof u of A. I want one of B in return.

$\ldots \to \mathsf{k}_\sigma^{A \to B} \star u^A . \pi$

cc: But you just needed a proof of A some time ago. Remember, precisely when we were in the context σ. Here it is.

$\to u^A \star \sigma$

End of the play

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.

$\mathsf{cc} \star t^{(A \to B) \to A} . \sigma$

cc *(lying)*: My proof of $((A \to B) \to A) \to A$ is fairly simple. I actually have a proof k_σ of $A \to B$, then just apply it your proof of $(A \to B) \to A$ to get A.

$\to t \star \mathsf{k}_\sigma^{A \to B} . \sigma$

P *(after computing a while)*: You gave me a proof of $A \to B$ that I now must use. I have brought a proof u of A. I want one of B in return.

$\ldots \to \mathsf{k}_\sigma^{A \to B} \star u^A . \pi$

cc: But you just needed a proof of A some time ago. Remember, precisely when we were in the context σ. Here it is.

$\to u^A \star \sigma$

End of the play?

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.

$\mathsf{cc} \star t^{(A \to B) \to A} . \sigma$

cc *(lying)*: My proof of $((A \to B) \to A) \to A$ is fairly simple. I actually have a proof k_σ of $A \to B$, then just apply it your proof of $(A \to B) \to A$ to get A.

$\to t \star \mathsf{k}_\sigma^{A \to B} . \sigma$

P *(after computing a while)*: You gave me a proof of $A \to B$ that I now must use. I have brought a proof u of A. I want one of B in return.

$\ldots \to \mathsf{k}_\sigma^{A \to B} \star u^A . \pi$

cc: But you just needed a proof of A some time ago. Remember, precisely when we were in the context σ. Here it is.

$\to u^A \star \sigma$

End of the play?

Certainly not as long as there are remaining occurrences of k_σ in the term.

A play in λcc-calculus

P: I have to use your proof of $((A \to B) \to A) \to A$. I give you one of $(A \to B) \to A$, waiting for your proof of A.

$\mathsf{cc} \star t^{(A \to B) \to A} . \sigma$

cc *(lying)*: My proof of $((A \to B) \to A) \to A$ is fairly simple. I actually have a proof k_σ of $A \to B$, then just apply it your proof of $(A \to B) \to A$ to get A.

$\to t \star \mathsf{k}_\sigma^{A \to B} . \sigma$

P *(after computing a while)*: You gave me a proof of $A \to B$ that I now must use. I have brought a proof u of A. I want one of B in return.

$\ldots \to \mathsf{k}_\sigma^{A \to B} \star u^A . \pi$

cc: But you just needed a proof of A some time ago. Remember, precisely when we were in the context σ. Here it is.

$\to u^A \star \sigma$

End of the play?

Certainly not as long as there are remaining occurrences of k_σ in the term. Everytime one of them comes in head position, followed by an always newer proof u_n of A: $\mathsf{k}_\sigma \star u_n . \pi_n$ the show goes on ($\to u_n \star \sigma \to \ldots$).