A MELL calculus based on contraposition

Work **in progress** with **Eduardo Bonelli** and **Leopoldo Lerena**

December 12, 2024

Pablo Barenbaum

Universidad Nacional de Quilmes (CONICET), Argentina
Universidad de Buenos Aires, Argentina

Outline

MLL in natural deduction style — Intuitionistic / Classical

$$A, B, \ldots ::= \alpha \mid \alpha^\perp \mid A \otimes B \mid A \multimap B$$

$$(A \otimes B)^\perp \stackrel{\text{def}}{=} A \multimap B^\perp \qquad (A \multimap B)^\perp \stackrel{\text{def}}{=} A \otimes B^\perp$$

$$\frac{}{A \vdash A}\text{ax}$$

$$\frac{\Gamma_1 \vdash A \quad \Gamma_2 \vdash B}{\Gamma_1, \Gamma_2 \vdash A \otimes B}\otimes_i \qquad \frac{\Gamma_1 \vdash A \otimes B \quad \Gamma_2, A, B \vdash C}{\Gamma_1, \Gamma_2 \vdash C}\otimes_e$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\multimap_i \qquad \frac{\Gamma_1 \vdash A \multimap B \quad \Gamma_2 \vdash A}{\Gamma_1, \Gamma_2 \vdash B}\multimap_e \multimap_{e_1} \qquad \frac{\Gamma_1 \vdash A \multimap B \quad \Gamma_2 \vdash B^\perp}{\Gamma_1, \Gamma_2 \vdash A^\perp}\multimap_{e_2}$$

What is missing to recover classical MLL?

$\multimap_{e_1}$ is *modus ponens*

$\multimap_{e_2}$ is *modus tollens*

Motivation

Can we derive a **calculus** for MLL from this system?

Start with a **term assignment**:

$$t ::= a \mid \langle t, s \rangle \mid t[\langle a, b \rangle := s] \mid \lambda a.\, t \mid t \,@\, s \mid t \blacktriangleleft s$$

$$\frac{}{a : A \vdash a : A}\,\mathrm{ax}$$

$$\frac{\Gamma_1 \vdash t : A \quad \Gamma_2 \vdash s : B}{\Gamma_1, \Gamma_2 \vdash \langle t, s \rangle : A \otimes B}\,\otimes_i \qquad \frac{\Gamma_1 \vdash t : A \otimes B \quad \Gamma_2, a : A, b : B \vdash s : C}{\Gamma_1, \Gamma_2 \vdash t[\langle a, b \rangle := s] : C}\,\otimes_e$$

$$\frac{\Gamma, a : A \vdash t : B}{\Gamma \vdash \lambda a.\, t : A \multimap B}\,\multimap_i \qquad \frac{\Gamma_1 \vdash t : A \multimap B \quad \Gamma_2 \vdash s : A}{\Gamma_1, \Gamma_2 \vdash t \,@\, s : B}\,\multimap_{e_1}$$

$$\frac{\Gamma_1 \vdash t : A \multimap B \quad \Gamma_2 \vdash s : B^\perp}{\Gamma_1, \Gamma_2 \vdash t \blacktriangleleft s : A^\perp}\,\multimap_{e_2}$$

The term constructor $t \blacktriangleleft s$ is called contra-application.

The λ_{MLL}-calculus — Reduction?

How can we reduce a $\multimap_i / \multimap_{e_2}$ redex?:

$$(\lambda a.\, t) \blacktriangleleft s \;\to\; \boxed{?}$$

The typing derivation for the left-hand side is:

$$\dfrac{\dfrac{\Gamma_1, a : A \vdash t : B}{\Gamma_1 \vdash \lambda a.\, t : A \multimap B}\multimap_i \qquad \Gamma_2 \vdash s : B^\perp}{\Gamma_1, \Gamma_2 \vdash (\lambda a.\, t) \blacktriangleleft s : A^\perp}\multimap_{e_2}$$

Key construction

The following rule is admissible:

$$\dfrac{\Gamma_1, a : A \vdash t : B \qquad \Gamma_2 \vdash s : B^\perp}{\Gamma_1, \Gamma_2 \vdash t\{a \mathbin{\backslash\!\backslash} s\} : A^\perp}\,\mathtt{CONTRA}$$

where $t\{a \mathbin{\backslash\!\backslash} s\}$ is a meta-level operation called contrasubstitution.

Contrasubstitution — Examples

$$\frac{\Gamma_1, a : A \vdash t : B \quad \Gamma_2 \vdash s : B^{\perp}}{\Gamma_1, \Gamma_2 \vdash t\{a \mathbin{\backslash\!\backslash} s\} : A^{\perp}}\text{CONTRA}$$

The construction of $t\{a \mathbin{\backslash\!\backslash} s\}$ proceeds by induction on t.

Example — variable

$$\frac{\dfrac{}{a : A \vdash a : A}\text{ax} \quad \Gamma_2 \vdash s : A^{\perp}}{\Gamma_2 \vdash a\{a \mathbin{\backslash\!\backslash} s\} : A^{\perp}}\text{CONTRA} \quad \rightsquigarrow \quad \text{Take } a\{a \mathbin{\backslash\!\backslash} s\} \stackrel{\text{def}}{=} s.$$

Note. The case $b\{a \mathbin{\backslash\!\backslash} s\}$ is impossible by the typing constraints.

Example — $\otimes$-introduction (left case)

$$\frac{\dfrac{\Gamma_{11}, a : A \vdash t_1 : B_1 \quad \Gamma_{12}, \vdash t_2 : B_2}{\Gamma_{11}, \Gamma_{12}\, a : A \vdash \langle t_1, t_2 \rangle : B_1 \times B_2}\otimes_i \quad \Gamma_2 \vdash s : B_1 \multimap B_2^{\perp}}{\Gamma_{11}, \Gamma_{12}, \Gamma_2 \vdash \langle t_1, t_2 \rangle\{a \mathbin{\backslash\!\backslash} s\} : A^{\perp}}\text{CONTRA}$$

$$\rightsquigarrow \quad \text{Take } \langle t_1, t_2 \rangle\{a \mathbin{\backslash\!\backslash} s\} \stackrel{\text{def}}{=} t_1\{a \mathbin{\backslash\!\backslash} s \blacktriangleleft t_2\}.$$

Contrasubstitution — Definition

The full definition of contrasubstitution is given by:

$$
\begin{array}{rcl}
a\{a \setminus\!\!\setminus s\} & \stackrel{\text{def}}{=} & s \\
\langle t_1, t_2\rangle\{a \setminus\!\!\setminus s\} & \stackrel{\text{def}}{=} & \begin{cases} t_1\{a \setminus\!\!\setminus s \blacktriangleleft t_2\} & \text{if } a \in \mathsf{fv}(t_1) \\ t_2\{a \setminus\!\!\setminus s \mathbin{@} t_1\} & \text{if } a \in \mathsf{fv}(t_2) \end{cases} \\
(t_1[\langle b, c\rangle := t_2])\{a \setminus\!\!\setminus s\} & \stackrel{\text{def}}{=} & \begin{cases} t_1\{a \setminus\!\!\setminus s\}[\langle b, c\rangle := t_2] & \text{if } a \in \mathsf{fv}(t_1) \\ t_2\{a \setminus\!\!\setminus \lambda b.\, t_1\{c \setminus\!\!\setminus s\}\} & \text{if } a \in \mathsf{fv}(t_2) \end{cases} \\
(\lambda b.\, t')\{a \setminus\!\!\setminus s\} & \stackrel{\text{def}}{=} & t'\{a \setminus\!\!\setminus c\}[\langle b, c\rangle := s] \\
(t_1 \mathbin{@} t_2)\{a \setminus\!\!\setminus s\} & \stackrel{\text{def}}{=} & \begin{cases} t_1\{a \setminus\!\!\setminus \langle t_2, s\rangle\} & \text{if } a \in \mathsf{fv}(t_1) \\ t_2\{a \setminus\!\!\setminus t_1 \blacktriangleleft s\} & \text{if } a \in \mathsf{fv}(t_2) \end{cases} \\
(t_1 \blacktriangleleft t_2)\{a \setminus\!\!\setminus s\} & \stackrel{\text{def}}{=} & \begin{cases} t_1\{a \setminus\!\!\setminus \langle s, t_2\rangle\} & \text{if } a \in \mathsf{fv}(t_1) \\ t_2\{a \setminus\!\!\setminus t_1 \mathbin{@} s\} & \text{if } a \in \mathsf{fv}(t_2) \end{cases}
\end{array}
$$

- Informally, $t\{a \setminus\!\!\setminus b\}$ turns t "inside-out".
  The ocurrence of a becomes the new root of the term.
  The root of t becomes a free occurrence of b.
  Introductions become eliminators of the dual connective.
- Contrasubstitution relies crucially on **linearity**.

Contrasubstitution — Properties

Definition (Structural equivalence)

The equivalence $\approx$ allows $\otimes$-eliminators to "float" (permutative rules):

$$\mathsf{C}\langle \ldots t[\langle a, b\rangle := s] \ldots \rangle \approx \mathsf{C}\langle \ldots t \ldots \rangle[\langle a, b\rangle := s]$$

Let $t\{a := s\}$ stand for the usual meta-level substitution.

Lemma ("Sub/contra" interaction)

$$\begin{array}{llcll}
\text{1a.} & t\{a \Lbag s\}\{b \Lbag r\} & \approx & t\{b := s\}\{a \Lbag r\} & \text{if } b \in \mathsf{fv}(t) \\
\text{1b.} & t\{a \Lbag s\}\{b \Lbag r\} & \approx & s\{b \Lbag t\{a := r\}\} & \text{if } b \in \mathsf{fv}(s) \\
\text{2a.} & t\{a \Lbag s\}\{b := r\} & = & t\{b := r\}\{a \Lbag s\} & \text{if } b \in \mathsf{fv}(t) \\
\text{2b.} & t\{a \Lbag s\}\{b := r\} & = & t\{a \Lbag s\{b := r\}\} & \text{if } b \in \mathsf{fv}(s) \\
\text{3a.} & t\{a := s\}\{b \Lbag r\} & \approx & s\{b \Lbag t\{a := r\}\} & \text{if } b \in \mathsf{fv}(t) \\
\text{3b.} & t\{a := s\}\{b \Lbag r\} & \approx & t\{a := s\{b \Lbag r\}\} & \text{if } b \in \mathsf{fv}(s)
\end{array}$$

Corollary (Involutivity)

$t\{a \Lbag b\}\{b \Lbag a\} \approx t$

The λ_{MLL}-calculus — Reduction

Let $\mathsf{L}, \mathsf{L}', \ldots$ stand for lists of $\otimes$-eliminators: $\mathsf{L} ::= \square \mid \mathsf{L}[\langle a, b\rangle := t]$.

Reduction rules

(at a distance; cf. Accattoli & Kesner, 2010)

$$
\begin{array}{rcl}
t[\langle a, b\rangle := \langle s, r\rangle \mathsf{L}] & \to & t\{a := s\}\{b := r\}\mathsf{L} \\
(\lambda a.\, t)\mathsf{L} \mathbin{@} s & \to & t\{a := s\}\mathsf{L} \\
(\lambda a.\, t)\mathsf{L} \blacktriangleleft s & \to & t\{a \mathbin{\|} s\}\mathsf{L}
\end{array}
$$

Note. Reduction is only defined over typable terms.

Example — reduction in λ_{MLL}

$$
\begin{array}{rll}
\vdash & \lambda p^{A\otimes B}.\, \langle b, a\rangle[\langle a^A, b^B\rangle := p] & : (A \otimes B) \multimap (B \otimes A) \\
f : B \multimap A^\perp \vdash & (\lambda p^{A\otimes B}.\, \langle b, a\rangle[\langle a^A, b^B\rangle := p]) \blacktriangleleft f & : A \multimap B^\perp
\end{array}
$$

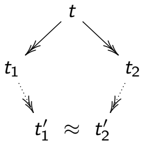
$$
\begin{array}{rl}
 & (\lambda p^{A\otimes B}.\, \langle b, a\rangle[\langle a^A, b^B\rangle := p]) \blacktriangleleft f \\
\to & (\langle b, a\rangle[\langle a^A, b^B\rangle := p])\{p \mathbin{\|} f\} \\
= & p\{p \mathbin{\|} \lambda a^A.\, \langle b, a\rangle\{b \mathbin{\|} f\}\} \\
= & \lambda a^A.\, \langle b, a\rangle\{b \mathbin{\|} f\} \\
= & \lambda a^A.\, b\{b \mathbin{\|} f \blacktriangleleft a\} \\
= & \lambda a^A.\, f \blacktriangleleft a
\end{array}
$$

The λ_{MLL}-calculus — Properties

Theorem

The λ_{MLL}-calculus enjoys the following properties:

1. **Logical soundness/completeness.**
   $\vdash \Gamma, A$ is valid in MLL iff there is a term t such that $\Gamma^{\perp} \vdash t : A$.
2. **Subject reduction.**
   If $\Gamma \vdash t : A$ and $t \ \rightarrow\ s$ then $\Gamma \vdash s : A$.
3. **Structural equivalence is a strong bisimulation.**
   If $t \approx s \ \rightarrow\ s'$ there exists t' such that $t \ \rightarrow\ t' \approx s'$.
4. **Confluence modulo structural equivalence.**



(The key is the "sub/contra" lemma).

5. **Strong normalization.**
   Typable terms have no infinite reduction paths. (Easy by linearity).

Outline

The $\lambda_{\mathsf{MELL}_0}$-calculus — First steps

$\lambda_{\mathsf{MELL_0}}$ uses two contexts, as DILL: (Barber, 1996)

- **Unrestricted** contexts $\Delta, \Delta', \ldots$ binding variables $u, v, \ldots$.
- **Linear** contexts $\Gamma, \Gamma', \ldots$ binding variables $a, b, \ldots$.

We have considered many variants and combinations of rules.

Example — some possible !-introduction rules

$$\frac{\Delta; \cdot \vdash A}{\Delta; \cdot \vdash !A}!_i \qquad \frac{\Delta; ?A^\perp \vdash \perp}{\Delta; \cdot \vdash !A}!'_i \qquad \frac{\Delta; A^\perp \vdash \perp}{\Delta; \cdot \vdash !A}!''_i$$

Example — some possible ?-introduction rules

$$\frac{\Delta; \Gamma \vdash \perp}{\Delta; \Gamma \vdash ?A}\mathsf{w} \quad \frac{\Delta; \Gamma \vdash A}{\Delta; \Gamma \vdash ?A}\mathsf{d} \quad \frac{\Delta; \Gamma, !A^\perp \vdash ?A}{\Delta; \Gamma \vdash ?A}\mathsf{c} \qquad \frac{\Delta, A^\perp; \Gamma \vdash \perp}{\Delta; \Gamma \vdash ?A}?_i \quad \ldots$$

Most of the combinations we tried seemed to be unsatisfactory.
(Due to the failure of completeness, confluence, involutivity, etc.).

The $\lambda_{\mathsf{MELL}_0}$-calculus — Syntax

Formulae are defined as follows:

$$A, B, \ldots ::= \alpha \mid \alpha^\perp \mid A \otimes B \mid A \parr B \mid \mathbb{1} \mid \bot \mid !A \mid ?A$$

- Units are needed to formulate the rules for exponentials.
- We also switch to $A \parr B$. As usual, $A \multimap B \stackrel{\text{def}}{=} A^\perp \parr B$.

The syntax of terms becomes:

$$
\begin{array}{rclllcll}
t & ::= & a & \text{(linear)} & & \mid & \star & (\mathbb{1}_i) \\
 & \mid & u & \text{(unrestricted)} & & \mid & t[\star := s] & (\mathbb{1}_e) \\
 & \mid & \langle t, s \rangle & (\otimes_i) & & \mid & t \lightning s & (\bot_i) \\
 & \mid & t[\langle a, b \rangle := s] & (\otimes_e) & & \mid & !a.t & (!_i) \\
 & \mid & \parr(a, b).t & (\parr_i) & & \mid & t[!u := s] & (!_e) \\
 & \mid & t \,@\, s & (\parr_{e_1}) & & \mid & ?u.t & (?_i) \\
 & \mid & t \blacktriangleleft s & (\parr_{e_2}) & & \mid & t[?a := s] & (?_e)
\end{array}
$$

The $\lambda_{\mathsf{MELL}_0}$-calculus — Typing rules (1/2)

The rules for linear variables and $\otimes$ are as before.

Typing rules for unrestricted variables and $⅋$

$$\frac{}{\Delta, u : A; \cdot \vdash u : A}\,\text{uax} \qquad \frac{\Delta; \Gamma, a : A^{\perp}, b : B^{\perp} \vdash t : \perp}{\Delta; \Gamma \vdash ⅋(a, b).t : A ⅋ B}\,⅋_i$$

$$\frac{\Delta; \Gamma_1 \vdash t : A ⅋ B \quad \Delta; \Gamma_2 \vdash t : A^{\perp}}{\Delta; \Gamma_1, \Gamma_2 \vdash t \,@\, s : B}\,⅋_{e_1} \qquad \frac{\Delta; \Gamma_1 \vdash t : A ⅋ B \quad \Delta; \Gamma_2 \vdash t : B^{\perp}}{\Delta; \Gamma_1, \Gamma_2 \vdash t \blacktriangleleft s : B}\,⅋_{e_2}$$

Typing rules for units

$$\frac{}{\Delta; \cdot \vdash \star : \mathbb{1}}\,\mathbb{1}_i \qquad \frac{\Delta; \Gamma_1 \vdash t : A \quad \Delta; \Gamma_2 \vdash s : \mathbb{1}}{\Delta; \Gamma_1, \Gamma_2 \vdash t[\star := s] : A}\,\mathbb{1}_e$$

$$\frac{\Delta; \Gamma_1 \vdash t : A \quad \Delta; \Gamma_2 \vdash s : A^{\perp}}{\Delta; \Gamma_1, \Gamma_2 \vdash t \lightning s : \perp}\,\perp_i$$

The $\lambda_{\mathsf{MELL}_0}$-calculus — Typing rules (2/2)

Typing rules for exponentials

$$\frac{\Delta; a : A^{\perp} \vdash t : \perp}{\Delta; \cdot \vdash !a.t : !A}\,!_i \qquad \frac{\Delta, u : A; \Gamma_1 \vdash t : B \quad \Delta; \Gamma_2 \vdash s : !A}{\Delta; \Gamma_1, \Gamma_2 \vdash t[!u := s] : B}\,!_e$$

$$\frac{\Delta, u : A^{\perp}; \Gamma \vdash t : \perp}{\Delta; \Gamma \vdash ?u.t : ?A}\,?_i \qquad \frac{\Delta; a : A \vdash t : \perp \quad \Delta; \Gamma \vdash s : !A}{\Delta; \Gamma \vdash t[?a := s] : \perp}\,?_e$$

Contrasubstitution — Extension for units and exponentials

Contrasubstitution for units

$$t_1[\star := t_2]\{a \,\backslash\!\backslash\, s\} \stackrel{\text{def}}{=} \begin{cases} t_1\{a \,\backslash\!\backslash\, s\}[\star := t_2] & \text{if } a \in \mathsf{fv}(t_1) \\ t_2\{a \,\backslash\!\backslash\, t_1 \lightning s\} & \text{if } a \in \mathsf{fv}(t_2) \end{cases}$$

$$(t_1 \lightning t_2)\{a \,\backslash\!\backslash\, s\} \stackrel{\text{def}}{=} \begin{cases} t_1\{a \,\backslash\!\backslash\, t_2[\star := s]\} & \text{if } a \in \mathsf{fv}(t_1) \\ t_2\{a \,\backslash\!\backslash\, t_1[\star := s]\} & \text{if } a \in \mathsf{fv}(t_2) \end{cases}$$

Contrasubstitution for exponentials

$$t_1[!u := t_2]\{a \,\backslash\!\backslash\, s\} \stackrel{\text{def}}{=} \begin{cases} t_1\{a \,\backslash\!\backslash\, s\}[!u := t_2] & \text{if } a \in \mathsf{fv}(t_1) \\ t_2\{a \,\backslash\!\backslash\, ?u.t_1 \lightning s\} & \text{if } a \in \mathsf{fv}(t_2) \end{cases}$$

$$(?u.t)\{a \,\backslash\!\backslash\, s\} \stackrel{\text{def}}{=} t\{a \,\backslash\!\backslash\, \star\}[!u := s]$$

$$t_1[?b := t_2]\{a \,\backslash\!\backslash\, s\} \stackrel{\text{def}}{=} t_2\{a \,\backslash\!\backslash\, (!b.t_1)[\star := s]\} \quad (\text{note that } a \in \mathsf{fv}(t_2))$$

- $t\{a \,\backslash\!\backslash\, s\}$ is only defined when a is a **linear variable**.
- Some cases are impossible, *e.g.* $\star\{a \,\backslash\!\backslash\, s\}$ or $(!a.t)\{b \,\backslash\!\backslash\, s\}$.
- If $\Delta; \Gamma, a : A^\perp \vdash t : \perp$ then $\Delta; \Gamma \vdash t\{a \,\backslash\!\backslash\, \star\} : A$.

Reduction rules

Now $\mathsf{L}, \mathsf{L}', \ldots$ are lists of eliminators of *positive* connectives $(\otimes, \mathbb{1}, !)$:

$$\mathsf{L} ::= \Box \mid \mathsf{L}[\langle a, b\rangle := t] \mid \mathsf{L}[\star := t] \mid \mathsf{L}[!u := t]$$

Reduction rules

$$\begin{array}{rcll}
t[\langle a, b\rangle := \langle s, r\rangle \mathsf{L}] & \to & t\{a := s\}\{b := r\}\mathsf{L} & \\
(\invamp(a, b).t)\mathsf{L} \mathbin{@} s & \to & t\{a := s\}\{b \mathbin{\backslash\!\backslash} \star\}\mathsf{L} & \\
(\invamp(a, b).t)\mathsf{L} \blacktriangleleft s & \to & t\{b := s\}\{a \mathbin{\backslash\!\backslash} \star\}\mathsf{L} & \\
t[!u := (!a.s)\mathsf{L}] & \to & t\{u := s\{b \mathbin{\backslash\!\backslash} \star\}\}\mathsf{L} & \\
t[?a := (?u.s)\mathsf{L}] & \to & s\{u := t\{a \mathbin{\backslash\!\backslash} \star\}\}\mathsf{L} & \\
\langle t, s\rangle \mathsf{L} \lightning (\invamp(a, b).r)\mathsf{K} & \to & r\{a := t\}\{b := s\}\mathsf{L}\mathsf{K} & \text{(+ symmetric rule)} \\
(!a.t)\mathsf{L} \lightning (?u.s)\mathsf{K} & \to & s\{u := t\{a \mathbin{\backslash\!\backslash} \star\}\}\mathsf{L}\mathsf{K} & \text{(+ symmetric rule)}
\end{array}$$

Note

There are no steps $t[\star := \star] \;\to\; t$. Instead, we shall have $t[\star := \star] \approx t$.
(This makes $\approx$ a strong bisimulation—there may be other ways).

Structural equivalence

Definition (Surface contexts)

A context S is *surface* if its hole is not inside a "$!a.\Box$" nor a "$\Box[?u := t]$".

Definition (Structural equivalence)

$$
\begin{array}{lcll}
\mathsf{S}\langle t[\mathtt{p} := r]\rangle & \approx & \mathsf{S}\langle t\rangle[\mathtt{p} := r] & \text{if S is surface} \\
 & & & \text{and } \mathtt{p} \text{ is a } \textit{positive pattern } (\star, \langle a, b\rangle, !u) \\
t[\star := \star] & \approx & t & \\
t[\star := s] & \approx & s[\star := t] & \\
t\{a \mathbin{\backslash\!\backslash} \star\} \lightning\, r & \approx & t\{a := r\} &
\end{array}
$$

Note

The equations only apply if the LHS and the RHS are both well-typed.
In particular, the third equation requires $t : \mathbb{1}$.

Example

$$\text{If } t : \bot, \qquad t \;=\; a\{a := t\} \;\approx\; a\{a \mathbin{\backslash\!\backslash} \star\} \lightning\, t \;=\; \star \lightning\, t$$

$$t \lightning s \approx s \lightning t \qquad \langle t, s\rangle \lightning r \approx (r \blacktriangleleft s) \lightning t \qquad \ldots$$

Structural equivalence

Theorem (Alternative characterization)

Structural equivalence is completely characterized by:

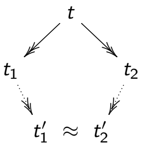
$$
\begin{array}{lcl}
\mathsf{S}\langle t\rangle[\mathrm{p} := r] & \approx & \mathsf{S}\langle t[\mathrm{p} := r]\rangle \\
t[\star := \star] & \approx & t \\
t[\star := s] & \approx & s[\star := t] \\
\star \lightning t & \approx & t \\
(s \blacktriangleleft t) \lightning r & \approx & t \lightning (s \,@\, r) \\
\langle r, t\rangle \lightning s & \approx & r \lightning (s \blacktriangleleft t) \\
(⅋(a, b).s) \lightning t & \approx & s[\langle a, b\rangle := t] \\
(?u.s) \lightning t & \approx & s[!u := t] \\
(!a.t) \lightning s & \approx & t[?a := s] \\
(r \lightning t) \lightning s & \approx & r \lightning t[\star := s]
\end{array}
$$

The $\lambda_{\mathsf{MELL}_0}$-calculus — Properties

Theorem

The $\lambda_{\mathsf{MELL_0}}$-calculus enjoys the following properties:

1. **Logical soundness/completeness.**
   $\vdash \Gamma, A$ is valid in MELL_0 iff there is a term t such that $\cdot; \Gamma^{\perp} \vdash t : A$.
2. **Subject reduction.**
   If $\Delta; \Gamma \vdash t : A$ and $t \rightarrow s$ then $\Delta; \Gamma \vdash s : A$.
3. **Structural equivalence is a strong bisimulation.**
   If $t \approx s \rightarrow s'$ there exists t' such that $t \rightarrow t' \approx s'$.
4. **Confluence modulo structural equivalence.**



5. **Strong normalization.**
   Typable terms have no infinite reduction paths.

Reducibility model, inspired by Accattoli's (RTA 2013).

Sketch of the reducibility model

Let $\mathcal{T}_A$ denote the terms of type A and SN_A the strongly normalizing terms of type A. Let us write:

- $t \lhd_A X \overset{\text{def}}{\iff} (t \in \mathcal{T}_A \implies t \in X)$ if $X \subseteq \mathcal{T}_A$.
- $X^\perp \overset{\text{def}}{=} \{t \in \mathcal{T}_{A^\perp} \mid \forall s \in X,\ t \not\downarrow s \lhd_\perp \mathsf{SN}_\perp\}$ if $X \subseteq \mathcal{T}_A$.
- If $X \subseteq \mathcal{T}_A$ and $Y \subseteq \mathcal{T}_B$:

$$\begin{array}{rcl} (X \underline{\otimes} Y) & \overset{\text{def}}{=} & \{\langle t, s\rangle \in \mathcal{T}_{A\otimes B} \mid t \in X \wedge s \in Y\} \\ \underline{?}X & \overset{\text{def}}{=} & \{?u.t \in \mathcal{T}_{?A} \mid \forall s \in X,\ t\{u := s\} \lhd_\perp \mathsf{SN}_\perp\} \end{array}$$

Definition (Reducibility candidates)

$$\begin{array}{rclrcl} [\![\alpha]\!] & \overset{\text{def}}{=} & \mathsf{SN}_\alpha & [\![\alpha^\perp]\!] & \overset{\text{def}}{=} & \mathsf{SN}_{\alpha^\perp} \\ [\![\perp]\!] & \overset{\text{def}}{=} & \mathsf{SN}_\perp & [\![\mathbb{1}]\!] & \overset{\text{def}}{=} & \mathsf{SN}_{\mathbb{1}} \\ [\![A \otimes B]\!] & \overset{\text{def}}{=} & ([\![A]\!] \underline{\otimes} [\![B]\!])^{\perp\perp} & [\![A \parr B]\!] & \overset{\text{def}}{=} & [\![A^\perp \otimes B^\perp]\!]^\perp \\ [\![?A]\!] & \overset{\text{def}}{=} & (\underline{?}[\![A^\perp]\!])^{\perp\perp} & [\![!A]\!] & \overset{\text{def}}{=} & [\![?A^\perp]\!]^\perp \end{array}$$

Theorem (Adequacy)

Let $\Delta; \Gamma \vdash t : A$. Then for every $\sigma \vDash \Delta, \Gamma$ we have that $t^\sigma \lhd_A [\![A]\!]$.

Outline

Calculi for classical and linear logic

It is well-known that classical logic can be embedded into linear logic.

- Danos, Joinet & Schellinx, 1997 — Q and T translations

There are several calculi in correspondence with classical logic:

1. Parigot, 1992 .... $\lambda\mu$-calculus
2. Krivine, ∼1993
3. Barbanera & Berardi, 1996
4. Curien & Herbelin, 2000 .... $\bar{\lambda}\mu\tilde{\mu}$-calculus
5. Munch-Maccagnoni, 2014
6. B. & Freund, 2021

   ...

and classical linear logic:

1. Albrecht, Crossley & Jeavons, 1997
2. Bierman, 1999
3. Martini & Masini, 1997

   ...

Translation of Parigot's $\lambda\mu$ into $\lambda_{\mathsf{MELL_0}}$ (1/2)

Parigot's $\lambda\mu$ can be translated into $\lambda_{\mathsf{MELL_0}}$.
The translation is based on Danos et al.'s **T translation**.

Syntax of $\lambda\mu$

$$
\begin{array}{lcl}
A, B, \ldots & ::= & \bot \mid \alpha \mid A \supset B \\
M, N, \ldots & ::= & x \mid \lambda x.\, M \mid M\, N \mid \underbrace{\mu\alpha^{\neg A}.M^{\bot}}_{A} \mid \underbrace{[\alpha^{\neg A}]M^{A}}_{\bot}
\end{array}
$$

Reduction in $\lambda\mu$

$$
\begin{array}{lcll}
(\lambda x.\, M)\ N & \to & M\{x := N\} & \\
(\mu\alpha.M)\ N & \to & \mu\alpha.(M\{\alpha \lhd N\}) & \\
[\alpha](\mu\alpha.M) & \to & M & \\
\mu\alpha.[\alpha]M & \to & M & \alpha \notin \mathrm{fv}(M)
\end{array}
$$

$M\{\alpha \lhd N\}$ replaces subterms of M of the form $[\alpha]O$ by $[\alpha](O\ N)$.

Translation of Parigot's $\lambda\mu$ into $\lambda_{\mathsf{MELL}_0}$ (2/2)

T-translation for $\lambda\mu$ (formulae)

$$
\begin{aligned}
\bot^{\mathsf{T}} &\stackrel{\text{def}}{=} \bot \\
\alpha^{\mathsf{T}} &\stackrel{\text{def}}{=} \alpha \\
(A \supset B)^{\mathsf{T}} &\stackrel{\text{def}}{=} ?!(A^{\mathsf{T}})^{\bot} \parr ?B^{\mathsf{T}}
\end{aligned}
$$

T-translation for $\lambda\mu$ (terms)

$$
\begin{aligned}
x^{\mathsf{T}} &\stackrel{\text{def}}{=} x \lightning !k \\
(\lambda x.\, M)^{\mathsf{T}} &\stackrel{\text{def}}{=} \parr(a, b).M^{\mathsf{T}}[!x := a][!k := b] \lightning k \\
(M\, N)^{\mathsf{T}} &\stackrel{\text{def}}{=} M^{\mathsf{T}}\{k := \langle !?k.N^{\mathsf{T}}, !k\rangle\} \\
([\alpha]M)^{\mathsf{T}} &\stackrel{\text{def}}{=} M^{\mathsf{T}}\{k := \alpha\} \lightning k \\
(\mu\alpha.M)^{\mathsf{T}} &\stackrel{\text{def}}{=} M^{\mathsf{T}}\{k := \star\}\{\alpha := k\}
\end{aligned}
$$

where $!t$ abbreviates $!a.(t \lightning a)$.

Theorem ($\lambda\mu$ simulation)

If $M \to N$ in $\lambda\mu$, then $M^{\mathsf{T}} \twoheadrightarrow\equiv N^{\mathsf{T}}$ in $\lambda_{\mathsf{MELL}_0}$.

Other translations

We have (so far) also given simulations for:

- Call-by-value $\lambda\mu$ ($\lambda\mu_V$) (Py, 1998) (**Q-translation**)
- Curien & Herbelin's $\bar{\lambda}\mu\tilde{\mu}$ (**T-translation**)
- Hasegawa's μDCLL (**CBN Girard's translation**)

Outline

Intuitionistic MELL_0

Definition (Input and output formulae)

$$
\begin{array}{rcllllllllll}
o & ::= & \alpha & \mid & o \otimes o & \mid & \iota \parr o & \mid & \mathbb{1} & \mid & !o \\
\iota & ::= & \overline{\alpha} & \mid & \iota \parr \iota & \mid & o \otimes \iota & \mid & \bot & \mid & ?\iota
\end{array}
$$

Definition (IMELL_0) (cf. Danos, 1990)

A MELL_0 sequent $\vdash \Gamma$ is *intuitionistic* iff Γ is of the form $\iota_1, \ldots, \iota_n, o$.

A sequent $\vdash \Gamma$ is *valid in* IMELL_0 if and only if it has a derivation in MELL_0 that involves only intuitionistic sequents.

Intuitionistic $\lambda_{\mathsf{MELL_0}}$

Definition (Intuitionistic $\lambda_{\mathsf{MELL_0}}$)

A $\lambda_{\mathsf{MELL_0}}$ typing judgment is *intuitionistic* if it is of one of the two following forms:

1. $\Delta; \Gamma \vdash t : o$
2. $\Delta; \Gamma, a : \iota_1 \vdash t : \iota_2$

where Δ and Γ contain only output formulae.

A judgment $\Delta; \Gamma \vdash t : A$ is *valid in* $\lambda_{\mathsf{IMELL_0}}$ if and only if it has a derivation in $\lambda_{\mathsf{MELL_0}}$ that involves only intuitionistic judgments.

Theorem (Intuitionistic soundness and completeness)

The following are equivalent:

- $\vdash \Gamma, A$ is valid in IMELL_0.
- There is a term t such that $\cdot; \Gamma^{\perp} \vdash t : A$ is valid in $\lambda_{\mathsf{IMELL_0}}$.

Outline

Conclusion

This work (in progress)

- **New calculi for MLL / MELL.**
  Key construction: contrasubstitution.
- Good properties: confluence (modulo $\approx$), strong normalization.
- It enjoys a form of the subformula property. (Not in this talk)
- Translations from classical calculi via T and Q translations.
- Intuitionistic fragment based on input/output formulae.

Future work

- Relate with proof nets. (cf. Linear Substitution Calculus)
- Extensions: additives, fixed points, 1st/2nd order quantifiers, ...
- Is there a way to formulate an untyped version of $\lambda_{\mathsf{MELL_o}}$?
- Translations for other classical/linear/process calculi.
- ...

Outline

λ_{MLL}-calculus

Example — structural equivalence is required for confluence

If $a \in \mathrm{fv}(t)$, then:

$$(\lambda a.\, (\lambda b.\, \langle d, c\rangle[\langle c, d\rangle := b]) \blacktriangleleft t) \blacktriangleleft s$$

$$\swarrow \qquad\qquad \searrow\!\!\searrow$$

$$t\{a \setminus\!\!\setminus \langle d, c\rangle\}[\langle c, d\rangle := s] \qquad \approx \qquad t\{a \setminus\!\!\setminus \langle d, c\rangle[\langle c, d\rangle := s]\}$$

Structural equivalence

Some derived equations

$$
\begin{array}{lcll}
\star \lightning t & \approx & t & \\
t_1 \lightning t_2 & \approx & t_2 \lightning t_1 & \\
t\{a \mathbin{\|} \star\}[\star := r] & \approx & t\{a \mathbin{\|} r\} & \\
t[\star := s] & \approx & t \lightning s & t : \perp \\
\star[\star := t] & \approx & t & \\
t\{a \mathbin{\|} r\} & \approx & t\{a := r\} & t : \perp
\end{array}
$$

Translation of $\lambda\mu_V$ into $\lambda_{\mathsf{MELL}_0}$ (1/2)

Syntax of $\lambda\mu_V$ (Py, 1998)

$$
\begin{array}{rcl}
A, B, \ldots & ::= & \bot \mid \alpha \mid A \supset B \\
M, N, \ldots & ::= & x \mid \lambda x.\, M \mid M\, N \mid \underbrace{\mu\alpha^{\neg A}.M^{\bot}}_{A} \mid \underbrace{[\alpha^{\neg A}]M^{A}}_{\bot} \\
V & ::= & x \mid \lambda x.\, M
\end{array}
$$

Reduction in $\lambda\mu_V$

$$
\begin{array}{lcll}
(\lambda x.\, M)\ V & \to & M\{x := V\} & \\
(\mu\alpha.M)\ V & \to & \mu\alpha.(M\{\alpha \lhd V\}) & \\
V\ (\mu\alpha.M) & \to & \mu\alpha.(M\{\alpha \lhd^{\star} V\}) & \\
[\alpha](\mu\alpha.M) & \to & M & \\
\mu\alpha.[\alpha]M & \to & M & \alpha \notin \mathrm{fv}(M)
\end{array}
$$

$M\{\alpha \lhd V\}$ replaces subterms of M of the form $[\alpha]O$ by $[\alpha](O\ V)$.
$M\{\alpha \lhd^{\star} V\}$ replaces subterms of M of the form $[\alpha]O$ by $[\alpha](V\ O)$.

Translation of $\lambda\mu_V$ into $\lambda_{\mathsf{MELL}_0}$ (2/2)

Q-translation for $\lambda\mu_V$ (formulae)

$$
\begin{aligned}
\bot^{\underline{\mathsf{Q}}} &\stackrel{\mathrm{def}}{=} \bot \\
\alpha^{\underline{\mathsf{Q}}} &\stackrel{\mathrm{def}}{=} \alpha \\
(A \supset B)^{\underline{\mathsf{Q}}} &\stackrel{\mathrm{def}}{=} (A^{\mathsf{Q}})^{\bot} \parr ?B^{\mathsf{Q}} \\
A^{\mathsf{Q}} &\stackrel{\mathrm{def}}{=} !A^{\underline{\mathsf{Q}}}
\end{aligned}
$$

Q-translation for $\lambda\mu_V$ (terms)

$$
\begin{aligned}
x^{\underline{\mathsf{Q}}} &\stackrel{\mathrm{def}}{=} x \\
(\lambda x.\, M)^{\underline{\mathsf{Q}}} &\stackrel{\mathrm{def}}{=} \parr(a, b).M^{\mathsf{Q}}[!x := a][!k := b] \\
V^{\mathsf{Q}} &\stackrel{\mathrm{def}}{=} !V^{\underline{\mathsf{Q}}} \lightning\, k \\
(M\, N)^{\mathsf{Q}} &\stackrel{\mathrm{def}}{=} M^{\mathsf{Q}}\{k := ?v.N^{\mathsf{Q}}\{k := v \blacktriangleleft !k\}\} \\
([\alpha]M)^{\mathsf{Q}} &\stackrel{\mathrm{def}}{=} M^{\mathsf{Q}}\{k := \alpha\} \lightning\, k \\
(\mu\alpha.M)^{\mathsf{Q}} &\stackrel{\mathrm{def}}{=} M^{\mathsf{Q}}\{k := \star\}\{\alpha := k\}
\end{aligned}
$$

Theorem ($\lambda\mu_V$ simulation)

If $M \to N$ in $\lambda\mu_V$, then $M^{\mathsf{Q}} \leftrightarrow^{*}\equiv N^{\mathsf{Q}}$ in $\lambda_{\mathsf{MELL_0}}$.

Translation of $\bar{\lambda}\mu\tilde{\mu}$ into $\lambda_{\mathsf{MELL}_0}$ (1/2)

Syntax of $\bar{\lambda}\mu\tilde{\mu}$

$$
\begin{array}{rcl}
A, B, \ldots & ::= & \alpha \mid A \supset B \\
v, v', \ldots & ::= & x^{A} \mid \underbrace{\mu\alpha^{\neg A}.\, c^{\perp}}_{A} \mid \underbrace{\lambda x^{A}.\, v^{B}}_{A \supset B} \\
E, E', \ldots & ::= & \alpha^{\neg A} \mid \underbrace{v^{A} \cdot E^{\neg B}}_{\neg(A \supset B)} \\
c, c', \ldots & ::= & \underbrace{\langle v^{A} \mid E^{\neg A} \rangle}_{\perp}
\end{array}
$$

Reduction in $\bar{\lambda}\mu\tilde{\mu}$

$$
\begin{array}{rcl}
\langle \lambda x.\, v_1 \mid v_2 \cdot E \rangle & \to & \langle v_1 \{x := v_2\} \mid E \rangle \\
\langle \mu\alpha.\, c \mid E \rangle & \to & c\{\alpha := E\}
\end{array}
$$

Translation of $\bar{\lambda}\mu\tilde{\mu}$ into $\lambda_{\mathsf{MELL}_0}$ (2/2)

T-translation for $\bar{\lambda}\mu\tilde{\mu}$ (formulae and judgments)

$$\begin{aligned} \alpha^{\mathsf{T}} &\stackrel{\text{def}}{=} \alpha \\ (A \supset B)^{\mathsf{T}} &\stackrel{\text{def}}{=} ?!(A^{\mathsf{T}})^{\perp} \parr ?B^{\mathsf{T}} \end{aligned}$$

$$\begin{aligned} c : \Gamma \vdash \Delta &\mapsto ?\Gamma^{\mathsf{T}}, \Delta^{\mathsf{T}\perp}; \cdot \vdash c^{\mathsf{T}} : \perp \\ E : \Gamma \mid A \vdash \Delta &\mapsto ?\Gamma^{\mathsf{T}}, \Delta^{\mathsf{T}\perp}; \cdot \vdash c^{\mathsf{T}} : !(A^{\mathsf{T}\perp}) \\ v : \Gamma \vdash A \mid \Delta &\mapsto ?\Gamma^{\mathsf{T}}, \Delta^{\mathsf{T}\perp}; \cdot \vdash c^{\mathsf{T}} : ?A^{\mathsf{T}} \end{aligned}$$

T-translation for $\bar{\lambda}\mu\tilde{\mu}$ (terms)

$$\begin{aligned} x^{\mathsf{T}} &\stackrel{\text{def}}{=} x \\ (\mu\alpha.\, c)^{\mathsf{T}} &\stackrel{\text{def}}{=} ?\alpha.c^{\mathsf{T}} \\ (\lambda x.\, v)^{\mathsf{T}} &\stackrel{\text{def}}{=} ?u.(u \lightning \parr(a, b).(v^{\mathsf{T}} \lightning b[!x := a])) \\ \alpha^{\mathsf{T}} &\stackrel{\text{def}}{=} !\alpha \\ (v \cdot E)^{\mathsf{T}} &\stackrel{\text{def}}{=} !\langle !v^{\mathsf{T}}, E^{\mathsf{T}} \rangle \\ \langle v \mid E \rangle^{\mathsf{T}} &\stackrel{\text{def}}{=} v^{\mathsf{T}} \lightning E^{\mathsf{T}} \end{aligned}$$

Translation of Hasegawa's μDCLL (1/2)

Syntax of μDCLL

$$
\begin{array}{lcl}
A, B & ::= & \bot \mid \alpha \mid A \supset B \mid A \multimap B \\
M, N & ::= & x \mid \underbrace{\Lambda x^{A}.\, M^{B}}_{A \supset B} \mid \underbrace{M^{A \supset B} \bullet N^{A}}_{B} \mid \underbrace{\lambda x^{A}.\, M^{B}}_{A \multimap B} \mid \underbrace{M^{A \multimap B} \,@\, N^{A}}_{B} \\
 & \mid & \underbrace{[\alpha^{\neg A}] M^{A}}_{\bot} \mid \underbrace{\mu \alpha^{\neg A}. M^{\bot}}_{A}
\end{array}
$$

Equivalence in μDCLL

$$
\begin{array}{lcll}
(\Lambda x.\, M) \bullet N & \doteq & M\{x := N\} & \\
(\lambda x.\, M) \,@\, N & \doteq & M\{x := N\} & \\
N\ (\mu\alpha.M) & \doteq_{\mu-R} & \mu\alpha.(M\{\alpha \lhd^{\star} N\}) & \\
\mu\alpha.[\alpha]M & \doteq & M & \alpha \notin \mathrm{fv}(M)
\end{array}
$$

Translation of Hasegawa's μDCLL (2/2)

Translation for μDCLL (formulae)

$$
\begin{aligned}
\bot^{\mathsf{H}} &\stackrel{\text{def}}{=} \bot \\
\alpha^{\mathsf{H}} &\stackrel{\text{def}}{=} \alpha \\
(A \supset B)^{\mathsf{H}} &\stackrel{\text{def}}{=} ?(A^{\mathsf{H}})^{\bot} \parr B^{\mathsf{H}} \\
(A \multimap B)^{\mathsf{H}} &\stackrel{\text{def}}{=} (A^{\mathsf{H}})^{\bot} \parr B^{\mathsf{H}}
\end{aligned}
$$

Translation for μDCLL (terms)

$$
\begin{aligned}
x^{\mathsf{H}} &\stackrel{\text{def}}{=} x \\
(\Lambda x.\, M)^{\mathsf{H}} &\stackrel{\text{def}}{=} \parr(a, k).(M^{\mathsf{H}}[!x := a] \lightning k) \\
(M \bullet N)^{\mathsf{H}} &\stackrel{\text{def}}{=} M^{\mathsf{H}} \mathbin{@} !N^{\mathsf{H}} \\
(\lambda x.\, M)^{\mathsf{H}} &\stackrel{\text{def}}{=} \parr(x, k).(M^{\mathsf{H}} \lightning k) \\
(M \mathbin{@} N)^{\mathsf{H}} &\stackrel{\text{def}}{=} M^{\mathsf{H}} \mathbin{@} N^{\mathsf{H}} \\
([\alpha]M)^{\mathsf{H}} &\stackrel{\text{def}}{=} M^{\mathsf{H}} \lightning \alpha \\
(\mu\alpha.M)^{\mathsf{H}} &\stackrel{\text{def}}{=} M^{\mathsf{H}}\{\alpha \mathbin{\backslash\!\backslash} \star\}
\end{aligned}
$$