A Classical Linear λ-Calculus based on Contraposition

joint work with **Pablo Barenbaum** and **Eduardo Bonelli**

10th of September, 2026

Leopoldo Lerena

Instituto de Ciencias de la Computación (ICC), UBA-CONICET.

Buenos Aires, Argentina.

Inference system for IMLL.

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

Inference system for IMLL.

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

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

$$\frac{\Gamma \vdash A \quad \Gamma' \vdash B}{\Gamma, \Gamma' \vdash A \otimes B}\ \otimes_i \qquad \frac{\Gamma \vdash A \otimes B \quad \Gamma', A, B \vdash C}{\Gamma, \Gamma' \vdash C}\ \otimes_e$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\ \multimap_i \qquad \frac{\Gamma \vdash A \multimap B \quad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B}\ \multimap_{e_1}$$

Inference system for MLL.

$$A, B, \ldots ::= \alpha \mid \overline{\alpha} \mid A \otimes B \mid A \multimap B$$

$$\alpha^{\perp} = \overline{\alpha} \quad (A \multimap B)^{\perp} = A \otimes B^{\perp} \quad (A \otimes B)^{\perp} = A \multimap B^{\perp}$$

$$\frac{}{A \vdash A}\,\mathsf{ax}$$

$$\frac{\Gamma \vdash A \quad \Gamma' \vdash B}{\Gamma, \Gamma' \vdash A \otimes B}\otimes_i \qquad \frac{\Gamma \vdash A \otimes B \quad \Gamma', A, B \vdash C}{\Gamma, \Gamma' \vdash C}\otimes_e$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\multimap_i \qquad \frac{\Gamma \vdash A \multimap B \quad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B}\multimap_{e_1}$$

Inference system for MLL.

$$A, B, \ldots ::= \alpha \mid \overline{\alpha} \mid A \otimes B \mid A \multimap B$$

$$\alpha^{\perp} = \overline{\alpha} \qquad (A \multimap B)^{\perp} = A \otimes B^{\perp} \qquad (A \otimes B)^{\perp} = A \multimap B^{\perp}$$

$$\boxed{A \multimap B = B^{\perp} \multimap A^{\perp}}$$

$$\frac{}{A \vdash A}\,\mathsf{ax}$$

$$\frac{\Gamma \vdash A \quad \Gamma' \vdash B}{\Gamma, \Gamma' \vdash A \otimes B}\,\otimes_i \qquad \frac{\Gamma \vdash A \otimes B \quad \Gamma', A, B \vdash C}{\Gamma, \Gamma' \vdash C}\,\otimes_e$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\,\multimap_i \qquad \frac{\Gamma \vdash A \multimap B \quad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B}\,\multimap_{e_1}$$

Inference system for MLL.

$$A, B, \ldots ::= \alpha \mid \overline{\alpha} \mid A \otimes B \mid A \multimap B$$

$$\alpha^{\perp} = \overline{\alpha} \qquad (A \multimap B)^{\perp} = A \otimes B^{\perp} \qquad (A \otimes B)^{\perp} = A \multimap B^{\perp}$$

$$\boxed{A \multimap B = B^{\perp} \multimap A^{\perp}}$$

$$\frac{}{A \vdash A}\,\mathsf{ax}$$

$$\frac{\Gamma \vdash A \quad \Gamma' \vdash B}{\Gamma, \Gamma' \vdash A \otimes B}\otimes_i \qquad \frac{\Gamma \vdash A \otimes B \quad \Gamma', A, B \vdash C}{\Gamma, \Gamma' \vdash C}\otimes_e$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\multimap_i \qquad \frac{\Gamma \vdash A \multimap B \quad \Gamma' \vdash A}{\Gamma, \Gamma' \vdash B}\multimap_{e_1}$$

$$\frac{\Gamma \vdash A \multimap B \quad \Gamma' \vdash B^{\perp}}{\Gamma, \Gamma' \vdash A^{\perp}}\multimap_{e_2}$$

The λ_{MLL} calculus.

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

The $\lambda_{\mathbf{MLL}}$ calculus.

$$t, s, \ldots \quad ::= \quad a \mid \langle t, s \rangle \mid s[\langle a, b \rangle := t] \mid \lambda a.\, t \mid t\ s \mid t \blacktriangledown s$$

The term $t \blacktriangledown s$ we call the *contra-application*.

The λ_{MLL} calculus.

$$t, s, \ldots \quad ::= \quad a \mid \langle t, s \rangle \mid s[\langle a, b \rangle := t] \mid \lambda a.\, t \mid t\ s \mid t \blacktriangledown s$$

The term $t \blacktriangledown s$ we call the *contra-application*.

$$\frac{}{a : A \vdash a : A}\ \text{m-ax}$$

$$\frac{\Gamma \vdash t : A \quad \Gamma' \vdash s : B}{\Gamma, \Gamma' \vdash \langle t, s \rangle : A \otimes B}\ \text{m-i}\otimes \qquad \frac{\Gamma \vdash t : A \otimes B \quad \Gamma', a : A, b : B \vdash s : C}{\Gamma, \Gamma' \vdash s[\langle a, b \rangle := t] : C}\ \text{m-e}\otimes$$

$$\frac{\Gamma, a : A \vdash t : B}{\Gamma \vdash \lambda a.\, t : A \multimap B}\ \text{m-i}\multimap$$

$$\frac{\Gamma \vdash t : A \multimap B \quad \Gamma' \vdash s : A}{\Gamma, \Gamma' \vdash t\ s : B}\ \text{m-e}\multimap_1 \qquad \frac{\Gamma \vdash t : A \multimap B \quad \Gamma' \vdash s : B^{\perp}}{\Gamma, \Gamma' \vdash t \blacktriangledown s : A^{\perp}}\ \text{m-e}\multimap_2$$

Redexes in λ_{MLL}.

We have the following three redexes:

$$t[\langle c, d\rangle := \langle s_1, s_2\rangle] \to t\{c := s_1\}\{d := s_2\}$$
$$(\lambda a.\, t)\ r \to t\{a := r\}$$

Redexes in λ_{MLL}.

We have the following three redexes:

$$t[\langle c, d\rangle := \langle s_1, s_2\rangle] \to t\{c := s_1\}\{d := s_2\}$$

$$(\lambda a.\, t)\ r \to t\{a := r\}$$

$$(\lambda a.\, t)\blacktriangledown s \to$$

Redexes in $\lambda_{\mathbf{MLL}}$.

We have the following three redexes:

$$t[\langle c, d\rangle := \langle s_1, s_2\rangle] \to t\{c := s_1\}\{d := s_2\}$$
$$(\lambda a.\, t)\ r \to t\{a := r\}$$
$$(\lambda a.\, t)\blacktriangledown s \to$$

What is the typing rule for the left hand side?

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

Redexes in λ_{MLL}.

We have the following three redexes:

$$t[\langle c, d\rangle := \langle s_1, s_2\rangle] \to t\{c := s_1\}\{d := s_2\}$$
$$(\lambda a.\, t)\ r \to t\{a := r\}$$
$$(\lambda a.\, t)\blacktriangledown s \to$$

What is the typing rule for the left hand side?

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

Contrasubstitution lemma:
The following rule is admissible:

$$\dfrac{\Gamma_1, a : A \vdash t : B \quad \Gamma_2 \vdash s : B^\perp}{\Gamma_1, \Gamma_2 \vdash t\{a \diagdown\!\!\diagdown s\} : A^\perp}\ \text{contra}$$

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

Redexes in λ_{MLL}.

We have the following three redexes:

$$t[\langle c, d\rangle := \langle s_1, s_2\rangle] \to t\{c := s_1\}\{d := s_2\}$$
$$(\lambda a.\, t)\ r \to t\{a := r\}$$
$$(\lambda a.\, t)\blacktriangledown s \to\ \ t\{a \between s\}$$

What is the typing rule for the left hand side?

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

Contrasubstitution lemma:
The following rule is admissible:

$$\dfrac{\Gamma_1, a : A \vdash t : B \quad \Gamma_2 \vdash s : B^\perp}{\Gamma_1, \Gamma_2 \vdash t\{a \between s\} : A^\perp}\ \text{contra}$$

where $t\{a \between s\}$ is a meta-level operation called **contrasubstitution**.

Contrasubstitution

Intuition: Think of $t\{a \backslash\!\backslash s\}$ as a process of first constructing a cut between t and s.

Contrasubstitution

Intuition: Think of $t\{a \diagdown\!\diagdown s\}$ as a process of first constructing a cut between t and s. Then, guided by the lone occurrence of the linear variable a in t we peel off the eliminators (resp. introductions) of t and accumulate them as introductions (resp. eliminators) onto s.

Contrasubstitution

Intuition: Think of $t\{a \backslash\!\backslash s\}$ as a process of first constructing a cut between t and s. Then, guided by the lone occurrence of the linear variable a in t we peel off the eliminators (resp. introductions) of t and accumulate them as introductions (resp. eliminators) onto s. We continue this process until we find a.

Contrasubstitution

Intuition: Think of $t\{a \diagdown\!\!\diagdown s\}$ as a process of first constructing a cut between t and s. Then, guided by the lone occurrence of the linear variable a in t we peel off the eliminators (resp. introductions) of t and accumulate them as introductions (resp. eliminators) onto s. We continue this process until we find a.

Example: If $t = \langle t_1, t_2 \rangle$ then either $a \in \mathrm{fv}(t_1)$ or $a \in \mathrm{fv}(t_2)$.

Contrasubstitution

Intuition: Think of $t\{a \diagdown\!\!\diagdown s\}$ as a process of first constructing a cut between t and s. Then, guided by the lone occurrence of the linear variable a in t we peel off the eliminators (resp. introductions) of t and accumulate them as introductions (resp. eliminators) onto s. We continue this process until we find a.

Example: If $t = \langle t_1, t_2 \rangle$ then either $a \in \mathrm{fv}(t_1)$ or $a \in \mathrm{fv}(t_2)$. Suppose $a \in \mathrm{fv}(t_1)$ then:

Contrasubstitution

Intuition: Think of $t\{a \backslash\!\backslash s\}$ as a process of first constructing a cut between t and s. Then, guided by the lone occurrence of the linear variable a in t we peel off the eliminators (resp. introductions) of t and accumulate them as introductions (resp. eliminators) onto s. We continue this process until we find a.

Example: If $t = \langle t_1, t_2 \rangle$ then either $a \in \mathrm{fv}(t_1)$ or $a \in \mathrm{fv}(t_2)$. Suppose $a \in \mathrm{fv}(t_1)$ then:

$$\begin{array}{ll} B \otimes C & A \;\; B \multimap C^{\perp} \\ \langle t_1, t_2 \rangle & \{ a \backslash\!\backslash \quad s \quad \} \end{array}$$

Contrasubstitution

Intuition: Think of $t\{a \Vdash s\}$ as a process of first constructing a cut between t and s. Then, guided by the lone occurrence of the linear variable a in t we peel off the eliminators (resp. introductions) of t and accumulate them as introductions (resp. eliminators) onto s. We continue this process until we find a.

Example: If $t = \langle t_1, t_2 \rangle$ then either $a \in \mathrm{fv}(t_1)$ or $a \in \mathrm{fv}(t_2)$. Suppose $a \in \mathrm{fv}(t_1)$ then:

$$\begin{array}{cc} B \otimes C & A \quad B \multimap C^{\perp} \\ \langle t_1, t_2 \rangle & \{a \Vdash \quad s \quad\} \end{array}$$

$$\begin{array}{cc} B & A \quad B^{\perp} \\ t_1 & \{a \Vdash s \blacktriangledown t_2\} \end{array}$$

Contrasubstitution

Intuition: Think of $t\{a \Vdash s\}$ as a process of first constructing a cut between t and s. Then, guided by the lone occurrence of the linear variable a in t we peel off the eliminators (resp. introductions) of t and accumulate them as introductions (resp. eliminators) onto s. We continue this process until we find a.

Example: If $t = \langle t_1, t_2 \rangle$ then either $a \in \mathrm{fv}(t_1)$ or $a \in \mathrm{fv}(t_2)$. Suppose $a \in \mathrm{fv}(t_1)$ then:

$$\begin{array}{cc} B \otimes C & A \quad B \multimap C^{\perp} \\ \langle t_1, t_2 \rangle & \{a \Vdash \quad s \quad\} \\ & \\ B & A \quad B^{\perp} \\ t_1 & \{a \Vdash s \blacktriangledown t_2\} \\ & \\ \vdots & \vdots \end{array}$$

Contrasubstitution

Intuition: Think of $t\{a \Vdash s\}$ as a process of first constructing a cut between t and s. Then, guided by the lone occurrence of the linear variable a in t we peel off the eliminators (resp. introductions) of t and accumulate them as introductions (resp. eliminators) onto s. We continue this process until we find a.

Example: If $t = \langle t_1, t_2 \rangle$ then either $a \in \mathrm{fv}(t_1)$ or $a \in \mathrm{fv}(t_2)$. Suppose $a \in \mathrm{fv}(t_1)$ then:

$$\begin{array}{ll} B \otimes C & A \quad B \multimap C^{\perp} \\ \langle t_1, t_2 \rangle & \{a \Vdash \quad s \quad\} \end{array}$$

$$\begin{array}{ll} B & A \quad B^{\perp} \\ t_1 & \{a \Vdash s \blacktriangledown t_2\} \end{array}$$

$$\begin{array}{ll} \vdots & \vdots \end{array}$$

$$\begin{array}{ll} A & A \quad A^{\perp} \\ a & \{a \Vdash \quad r \quad\} \end{array}$$

Contrasubstitution

Intuition: Think of $t\{a \Vdash s\}$ as a process of first constructing a cut between t and s. Then, guided by the lone occurrence of the linear variable a in t we peel off the eliminators (resp. introductions) of t and accumulate them as introductions (resp. eliminators) onto s. We continue this process until we find a.

Example: If $t = \langle t_1, t_2 \rangle$ then either $a \in \mathrm{fv}(t_1)$ or $a \in \mathrm{fv}(t_2)$. Suppose $a \in \mathrm{fv}(t_1)$ then:

$$\begin{array}{ll} B \otimes C & A \quad B \multimap C^{\perp} \\ \langle t_1, t_2 \rangle & \{a \Vdash \quad s \quad\} \end{array}$$

$$\begin{array}{ll} B & A \quad B^{\perp} \\ t_1 & \{a \Vdash s \blacktriangledown t_2\} \end{array}$$

$$\begin{array}{ll} \vdots & \vdots \end{array}$$

$$\begin{array}{ll} A & A \quad A^{\perp} \\ a & \{a \Vdash \quad r \quad\} \end{array}$$

$$= \quad A^{\perp} \\ \quad\; r$$

Definition of contrasubstitution

A term is **linear** if each free variable occurs exactly once, and there is exactly one occurrence of each bound variable inside the scope of its binder.

Definition of contrasubstitution

A term is **linear** if each free variable occurs exactly once, and there is exactly one occurrence of each bound variable inside the scope of its binder.

Typable terms are linear!

Definition of contrasubstitution

A term is **linear** if each free variable occurs exactly once, and there is exactly one occurrence of each bound variable inside the scope of its binder.

Typable terms are linear!

The full definition of **contrasubstitution** is given by induction on the structure of a term t such that $a \in \mathrm{fv}(t)$.

Definition of contrasubstitution

A term is **linear** if each free variable occurs exactly once, and there is exactly one occurrence of each bound variable inside the scope of its binder.

Typable terms are linear!

The full definition of **contrasubstitution** is given by induction on the structure of a term t such that $a \in \mathrm{fv}(t)$.

$$a\{a \backslash\!\backslash s\} \quad := \quad s$$

Definition of contrasubstitution

A term is **linear** if each free variable occurs exactly once, and there is exactly one occurrence of each bound variable inside the scope of its binder.

Typable terms are linear!

The full definition of **contrasubstitution** is given by induction on the structure of a term t such that $a \in \mathrm{fv}(t)$.

$$
\begin{aligned}
a\{a \backslash\!\backslash s\} &:= s \\
\langle t_1, t_2 \rangle \{a \backslash\!\backslash s\} &:= \begin{cases} t_1\{a \backslash\!\backslash s \blacktriangledown t_2\} & \text{if } a \in \mathrm{fv}(t_1) \\ t_2\{a \backslash\!\backslash s\ t_1\} & \text{if } a \in \mathrm{fv}(t_2) \end{cases}
\end{aligned}
$$

Definition of contrasubstitution

A term is **linear** if each free variable occurs exactly once, and there is exactly one occurrence of each bound variable inside the scope of its binder.

Typable terms are linear!

The full definition of **contrasubstitution** is given by induction on the structure of a term t such that $a \in \mathrm{fv}(t)$.

$$
\begin{aligned}
a\{a \backslash\!\backslash s\} &:= s \\
\langle t_1, t_2\rangle\{a \backslash\!\backslash s\} &:= \begin{cases} t_1\{a \backslash\!\backslash s \blacktriangledown t_2\} & \text{if } a \in \mathrm{fv}(t_1) \\ t_2\{a \backslash\!\backslash s\ t_1\} & \text{if } a \in \mathrm{fv}(t_2) \end{cases} \\
(t_1[\langle b, c\rangle := t_2])\{a \backslash\!\backslash s\} &:= \begin{cases} t_1\{a \backslash\!\backslash s\}[\langle b, c\rangle := t_2] & \text{if } a \in \mathrm{fv}(t_1) \\ t_2\{a \backslash\!\backslash \lambda b.\, t_1\{c \backslash\!\backslash s\}\} & \text{if } a \in \mathrm{fv}(t_2) \end{cases}
\end{aligned}
$$

Definition of contrasubstitution

A term is **linear** if each free variable occurs exactly once, and there is exactly one occurrence of each bound variable inside the scope of its binder.

Typable terms are linear!

The full definition of **contrasubstitution** is given by induction on the structure of a term t such that $a \in \mathrm{fv}(t)$.

$$
\begin{aligned}
a\{a \backslash\!\backslash s\} &:= s \\
\langle t_1, t_2\rangle\{a \backslash\!\backslash s\} &:= \begin{cases} t_1\{a \backslash\!\backslash s \blacktriangledown t_2\} & \text{if } a \in \mathrm{fv}(t_1) \\ t_2\{a \backslash\!\backslash s\ t_1\} & \text{if } a \in \mathrm{fv}(t_2) \end{cases} \\
(t_1[\langle b, c\rangle := t_2])\{a \backslash\!\backslash s\} &:= \begin{cases} t_1\{a \backslash\!\backslash s\}[\langle b, c\rangle := t_2] & \text{if } a \in \mathrm{fv}(t_1) \\ t_2\{a \backslash\!\backslash \lambda b.\, t_1\{c \backslash\!\backslash s\}\} & \text{if } a \in \mathrm{fv}(t_2) \end{cases} \\
(\lambda b.\, t')\{a \backslash\!\backslash s\} &:= t'\{a \backslash\!\backslash c\}[\langle b, c\rangle := s] \\
(t_1\ t_2)\{a \backslash\!\backslash s\} &:= \begin{cases} t_1\{a \backslash\!\backslash \langle t_2, s\rangle\} & \text{if } a \in \mathrm{fv}(t_1) \\ t_2\{a \backslash\!\backslash t_1 \blacktriangledown s\} & \text{if } a \in \mathrm{fv}(t_2) \end{cases} \\
(t_1 \blacktriangledown t_2)\{a \backslash\!\backslash s\} &:= \begin{cases} t_1\{a \backslash\!\backslash \langle s, t_2\rangle\} & \text{if } a \in \mathrm{fv}(t_1) \\ t_2\{a \backslash\!\backslash t_1\ s\} & \text{if } a \in \mathrm{fv}(t_2) \end{cases}
\end{aligned}
$$

Reduction and Structural equivalence

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

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

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

Reduction and Structural equivalence

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

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

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

⚠ The calculus is not confluent as it is.

Reduction and Structural equivalence

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

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

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

⚠ The calculus is not confluent as it is.

Structural equivalence: We define the equivalence $\equiv$ that allows $\otimes$-eliminators to "float":

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

Properties of the calculus.

- **Logical soundness/completeness**.
  $\vdash \Gamma, A$ is valid in MLL iff there is a term t such that $\Gamma^{\perp} \vdash t : A$.
- **Subject reduction**.
  If $t \to t'$ and $\Gamma \vdash t : A$ then $\Gamma \vdash t' : A$.

Properties of the calculus.

- **Logical soundness/completeness**.
  $\vdash \Gamma, A$ is valid in MLL iff there is a term t such that $\Gamma^{\perp} \vdash t : A$.
- **Subject reduction**.
  If $t \to t'$ and $\Gamma \vdash t : A$ then $\Gamma \vdash t' : A$.
- **Structural equivalence is a strong bisimulation**.
  If $t \equiv s$ and $t \to t'$ then there exists s' such that $s \to s'$ and $s' \equiv t'$.

Properties of the calculus.

- **Logical soundness/completeness**.
  $\vdash \Gamma, A$ is valid in MLL iff there is a term t such that $\Gamma^{\perp} \vdash t : A$.
- **Subject reduction**.
  If $t \to t'$ and $\Gamma \vdash t : A$ then $\Gamma \vdash t' : A$.
- **Structural equivalence is a strong bisimulation**.
  If $t \equiv s$ and $t \to t'$ then there exists s' such that $s \to s'$ and $s' \equiv t'$.
- **Strong normalization**.
  There are no infinite reduction sequences.
  Every reduction step strictly decreases the size of the term.

Properties of the calculus.

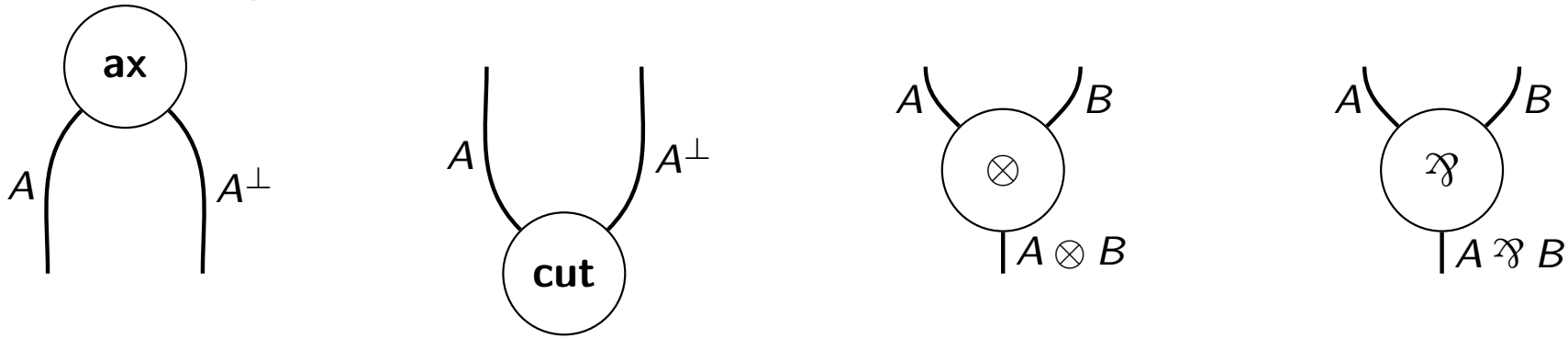
- **Logical soundness/completeness**.
  $\vdash \Gamma, A$ is valid in MLL iff there is a term t such that $\Gamma^{\perp} \vdash t : A$.
- **Subject reduction**.
  If $t \to t'$ and $\Gamma \vdash t : A$ then $\Gamma \vdash t' : A$.
- **Structural equivalence is a strong bisimulation**.
  If $t \equiv s$ and $t \to t'$ then there exists s' such that $s \to s'$ and $s' \equiv t'$.
- **Strong normalization**.
  There are no infinite reduction sequences.
  Every reduction step strictly decreases the size of the term.
- **Confluence modulo** $\equiv$.
  If $t \twoheadrightarrow t_1$ and $t \twoheadrightarrow t_2$ then there are t_1' and t_2' such that: $t_1 \twoheadrightarrow t_1'$, $t_2 \twoheadrightarrow t_2'$ and $t_1' \equiv t_2'$.
  It suffices to look at all the co-initial steps

Properties of the calculus.

- **Logical soundness/completeness**.
  $\vdash \Gamma, A$ is valid in MLL iff there is a term t such that $\Gamma^{\perp} \vdash t : A$.
- **Subject reduction**.
  If $t \to t'$ and $\Gamma \vdash t : A$ then $\Gamma \vdash t' : A$.
- **Structural equivalence is a strong bisimulation**.
  If $t \equiv s$ and $t \to t'$ then there exists s' such that $s \to s'$ and $s' \equiv t'$.
- **Strong normalization**.
  There are no infinite reduction sequences.
  Every reduction step strictly decreases the size of the term.
- **Confluence modulo** $\equiv$.
  If $t \twoheadrightarrow t_1$ and $t \twoheadrightarrow t_2$ then there are t_1' and t_2' such that: $t_1 \twoheadrightarrow t_1'$, $t_2 \twoheadrightarrow t_2'$ and $t_1' \equiv t_2'$.
  It suffices to look at all the co-initial steps

Proof nets for MLL

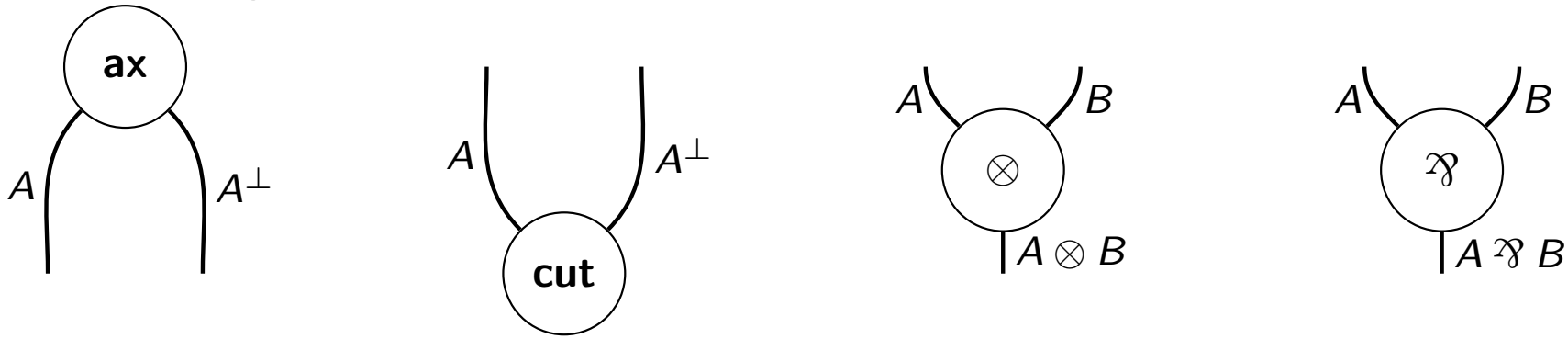
Proof nets are finite acyclic oriented graphs built over the alphabet of the following nodes:



Such that they satisfy certain correctness criteria

Proof nets for MLL

Proof nets are finite acyclic oriented graphs built over the alphabet of the following nodes:

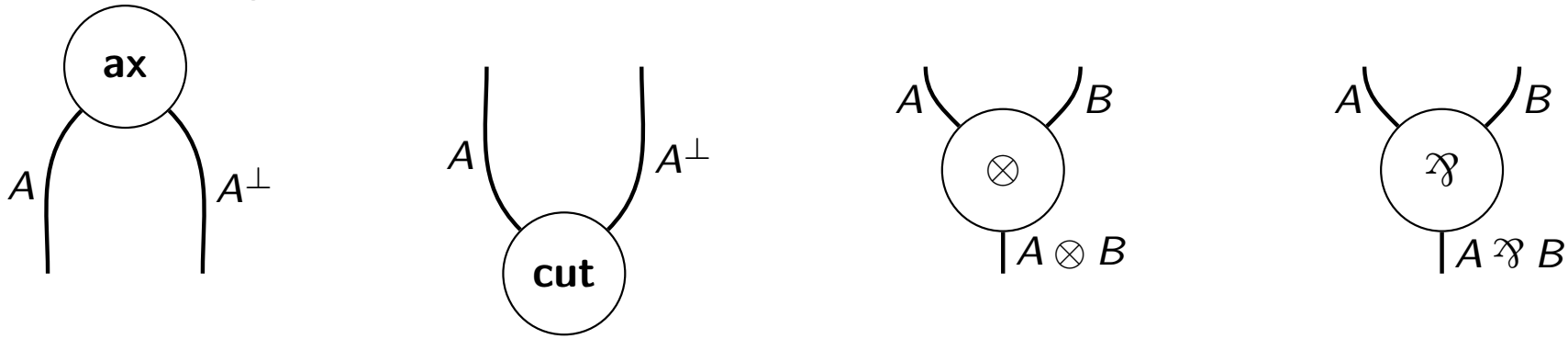


Such that they satisfy certain correctness criteria

- The incident (top) edges are **premises**.

Proof nets for MLL

Proof nets are finite acyclic oriented graphs built over the alphabet of the following nodes:

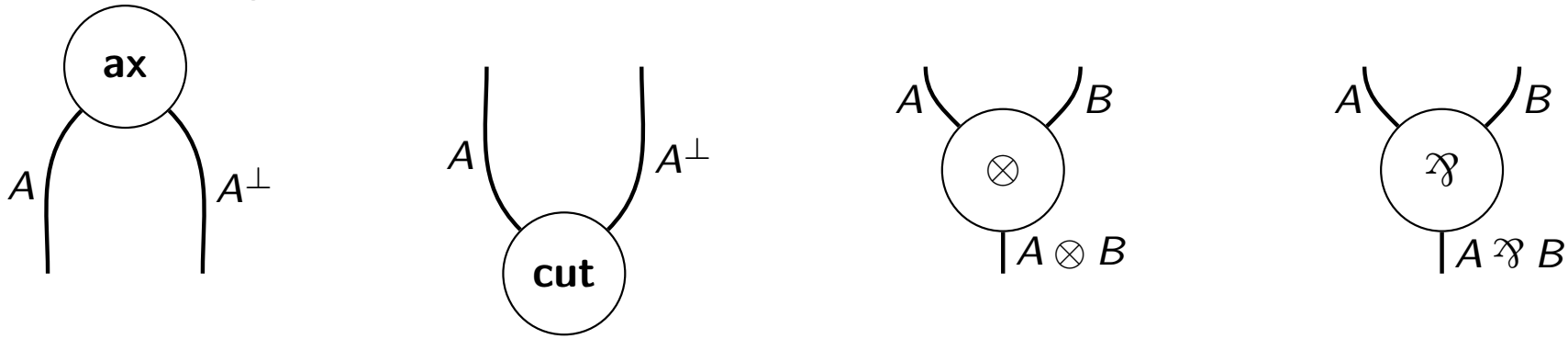


Such that they satisfy certain correctness criteria

- The incident (top) edges are **premises**.
- The emergent (bottom) edges are **conclusions**.

Proof nets for MLL

Proof nets are finite acyclic oriented graphs built over the alphabet of the following nodes:



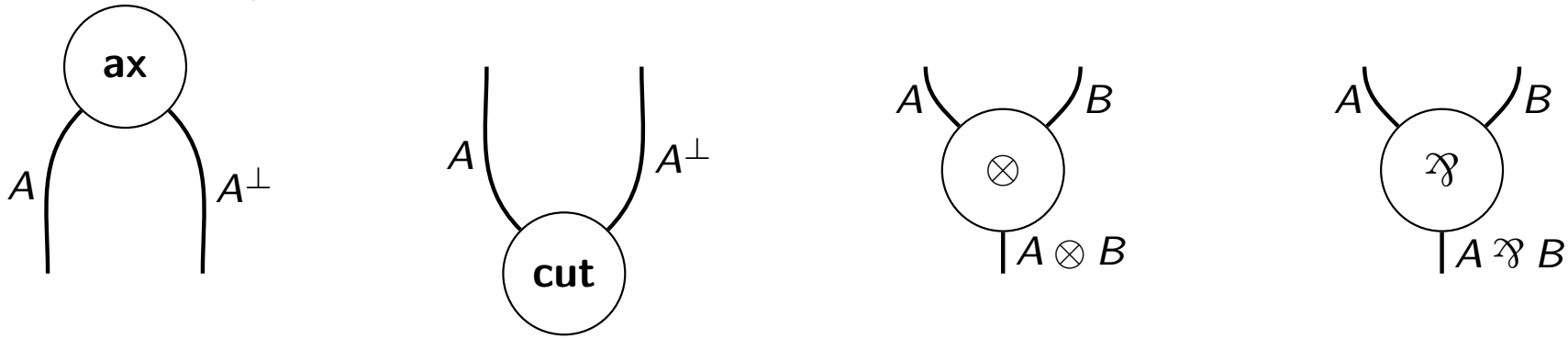
Such that they satisfy certain correctness criteria

- The incident (top) edges are **premises**.
- The emergent (bottom) edges are **conclusions**.

Derivation trees in MLL sequent calculus are in correspondence with proof nets.

Proof nets for MLL

Proof nets are finite acyclic oriented graphs built over the alphabet of the following nodes:



Such that they satisfy certain correctness criteria

- The incident (top) edges are **premises**.
- The emergent (bottom) edges are **conclusions**.

Derivation trees in MLL sequent calculus are in correspondence with proof nets.

The proof nets come equipped with a cut elimination reduction for ax, $\parr/\otimes$.

Distinguished proof nets

Following Laurent (2003) we will distinguish one of the conclusions of a proof net. Graphically, the distinguished conclusion will be denoted by $\rightarrow$, while the old distinguished conclusions will be denoted by $\multimap$.

Distinguished proof nets

Following Laurent (2003) we will distinguish one of the conclusions of a proof net. Graphically, the distinguished conclusion will be denoted by $\rightarrow$, while the old distinguished conclusions will be denoted by $\multimap$.

Idea

We will map terms from the calculus λ_{MLL} to a proof net with a distinguished conclusion

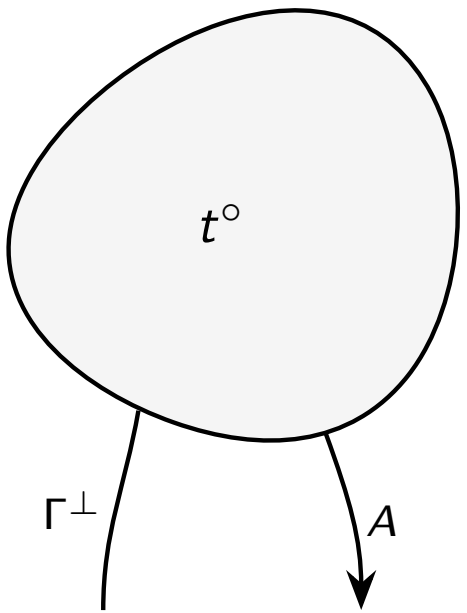
Distinguished proof nets

Following Laurent (2003) we will distinguish one of the conclusions of a proof net. Graphically, the distinguished conclusion will be denoted by $\rightarrow$, while the old distinguished conclusions will be denoted by $\multimap$.

Idea

We will map terms from the calculus λ_{MLL} to a proof net with a distinguished conclusion

$$\Gamma \vdash t : A \qquad \overset{\circ}{\longmapsto}$$



Translation

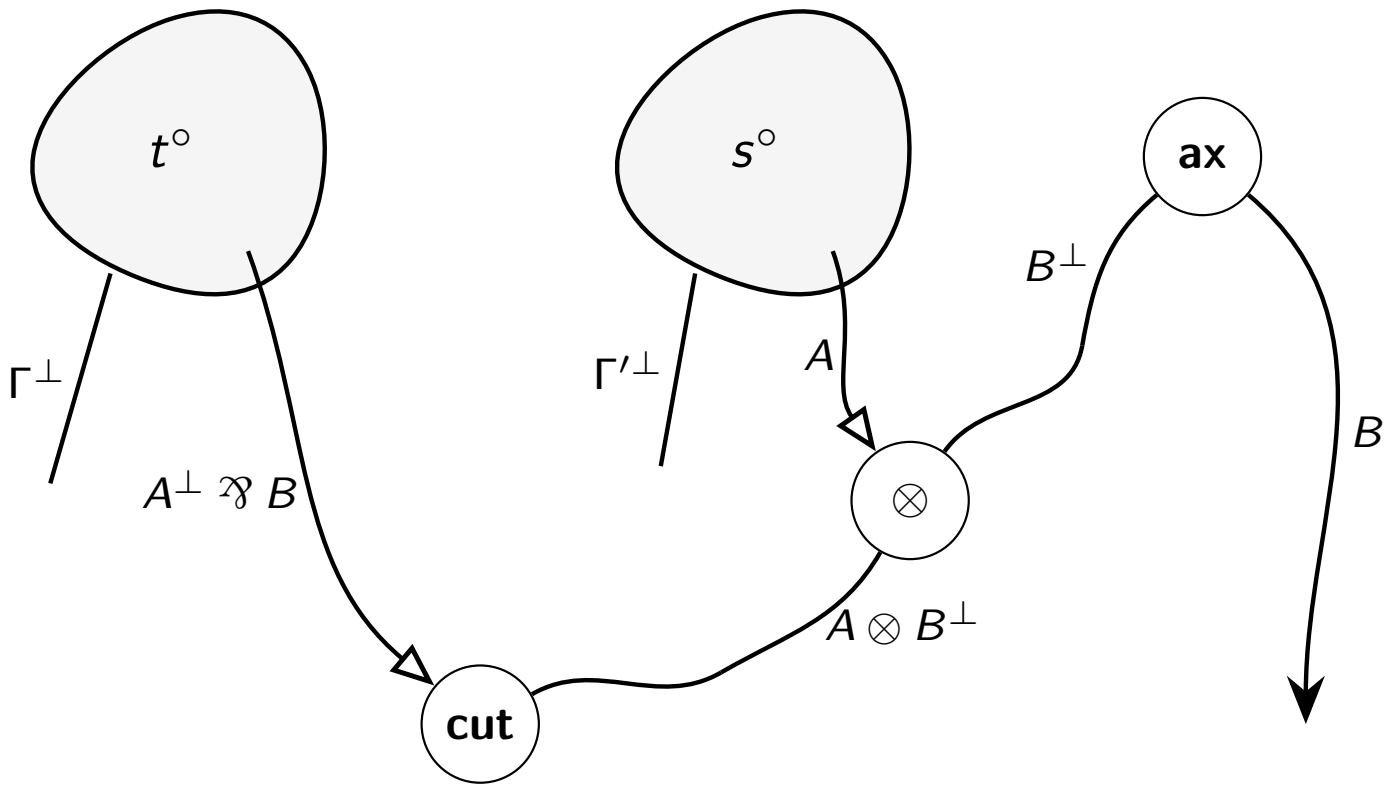
We define the translation by induction on the derivation of the term. For example if the last rule is an application:

Translation

We define the translation by induction on the derivation of the term. For example if the last rule is an application:

$$\left(\frac{\Gamma \vdash t : A \multimap B \quad \Gamma' \vdash s : A}{\Gamma, \Gamma' \vdash t\ s : B}\ \text{m-e}{\multimap}_1 \right)^{\circ}$$

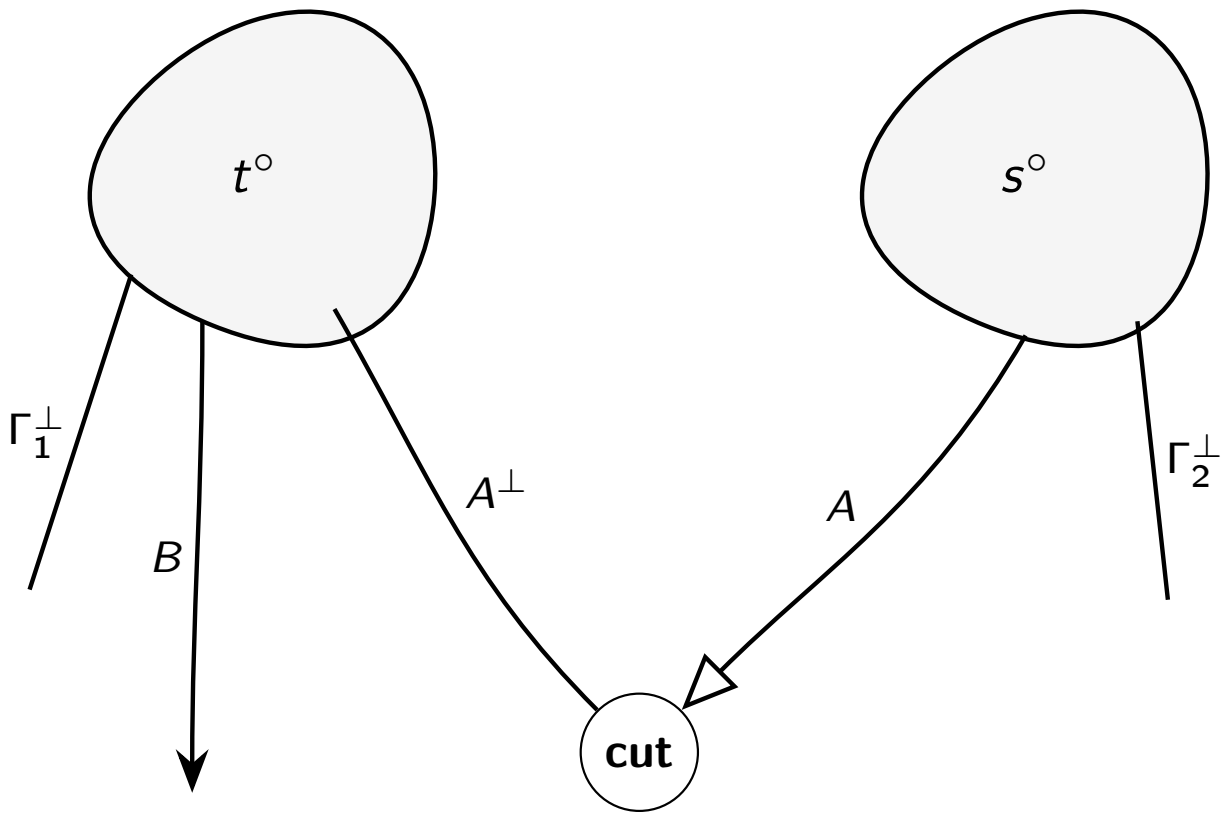
$$\Big\Vert$$



Substitution

Substitution lemma

$$\frac{\Gamma_1, a : A \vdash t : B \quad \Gamma_2 \vdash s : A}{\Gamma_1, \Gamma_2 \vdash t\{a := s\} : B}\ \text{sub}$$

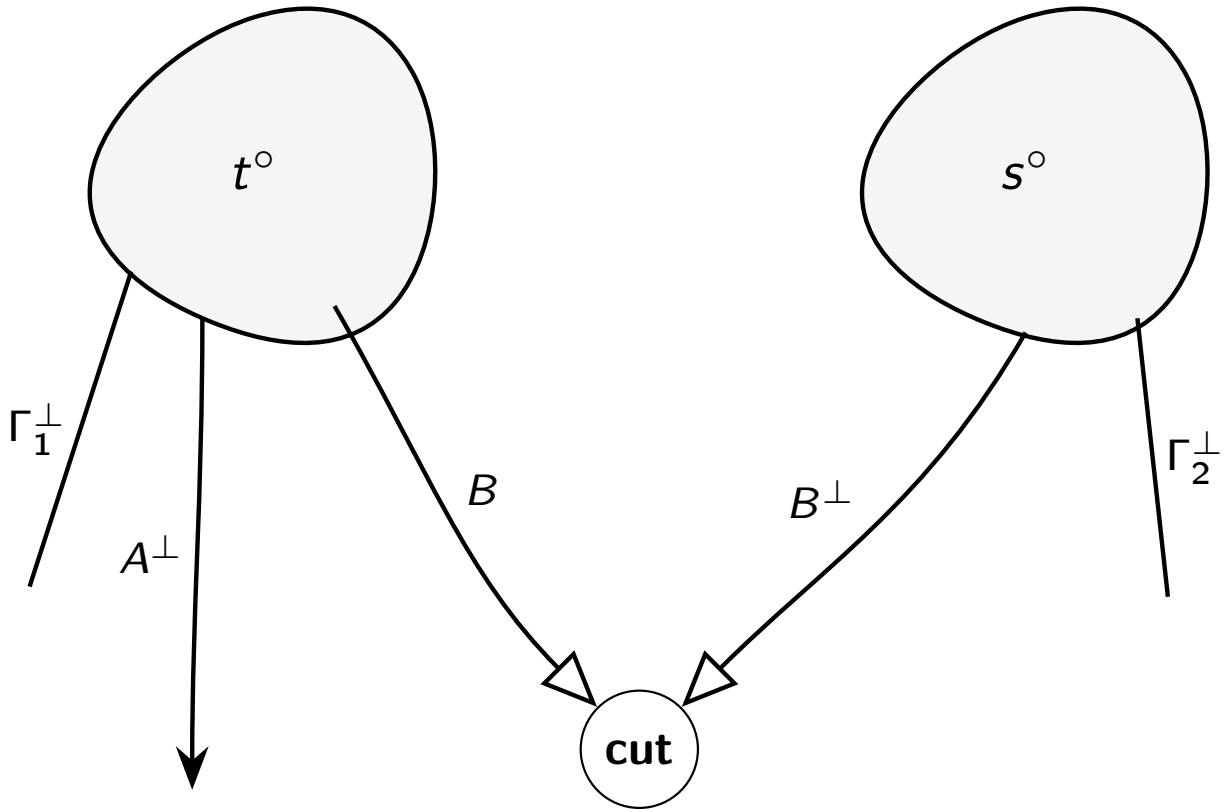


$\twoheadrightarrow (t\{a := s\})^\circ$

Contrasubstitution

Contrasubstitution lemma

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



$\twoheadrightarrow (t\{a \backslash\!\backslash s\})^\circ$

Syntax for $\lambda_{\mathbf{MELL}}$

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

Syntax for $\lambda_{\mathbf{MELL}}$

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

Since $A^{\perp} \multimap B = A \parr B$, we just use $\parr$ instead of $\multimap$.

$$\begin{array}{rcll} t & ::= & a & \text{(linear)} \\ & \mid & u & \text{(unrestricted)} \end{array}$$

Syntax for $\lambda_{\mathbf{MELL}}$

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

Since $A^{\perp} \multimap B = A \parr B$, we just use $\parr$ instead of $\multimap$.

$$\begin{array}{rcll} t & ::= & a & \text{(linear)} \\ & \mid & u & \text{(unrestricted)} \\ & \mid & \langle t, s \rangle & (\otimes_i) \\ & \mid & t[\langle a, b \rangle := s] & (\otimes_e) \end{array}$$

Syntax for $\lambda_{\mathbf{MELL}}$

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

Since $A^{\perp} \multimap B = A \parr B$, we just use $\parr$ instead of $\multimap$.

$$\begin{array}{rcll} t & ::= & a & \text{(linear)} \\ & \mid & u & \text{(unrestricted)} \\ & \mid & \langle t, s \rangle & (\otimes_i) \\ & \mid & t[\langle a, b \rangle := s] & (\otimes_e) \\ & \mid & \parr(a, b).t & (\parr_i) \\ & \mid & t\ s & (\parr_{e_1}) \\ & \mid & t \blacktriangledown s & (\parr_{e_2}) \end{array}$$

Syntax for $\lambda_{\mathbf{MELL}}$

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

Since $A^{\perp} \multimap B = A \parr B$, we just use $\parr$ instead of $\multimap$.

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

Syntax for $\lambda_{\mathbf{MELL}}$

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

Since $A^{\perp} \multimap B = A \parr B$, we just use $\parr$ instead of $\multimap$.

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

Typing rules 1/2

λ_{MELL} uses two contexts, as DILL: (Barber, 1996)

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

Typing rules 1/2

λ_{MELL} uses two contexts, as DILL: (Barber, 1996)

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

Variables typing rules

$$\frac{}{\Delta; a : A \vdash a : A}\ \text{m-ax} \qquad \frac{}{\Delta, u : A; \cdot \vdash u : A}\ \text{m-uax}$$

Typing rules 1/2

λ_{MELL} uses two contexts, as DILL: (Barber, 1996)

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

Variables typing rules

$$\frac{}{\Delta; a : A \vdash a : A}\ \text{m-ax} \qquad \frac{}{\Delta, u : A; \cdot \vdash u : A}\ \text{m-uax}$$

The rules for the tensor $\otimes$ remain identical to the ones of λ_{MLL}.

Typing rules 1/2

λ_{MELL} uses two contexts, as DILL: (Barber, 1996)

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

Variables typing rules

$$\frac{}{\Delta; a : A \vdash a : A}\ \text{m-ax} \qquad \frac{}{\Delta, u : A; \cdot \vdash u : A}\ \text{m-uax}$$

The rules for the tensor $\otimes$ remain identical to the ones of λ_{MLL}.

$\parr$ typing rules

$$\frac{\Delta; \Gamma, a : A, b : B \vdash t : \bot}{\Delta; \Gamma \vdash \parr(a, b).t : A^{\bot} \parr B^{\bot}}\ \text{m-i}\parr$$

$$\frac{\Delta; \Gamma \vdash t : A \parr B \quad \Delta; \Gamma' \vdash s : A^{\bot}}{\Delta; \Gamma, \Gamma' \vdash t\ s : B}\ \text{m-e}\parr_1 \qquad \frac{\Delta; \Gamma \vdash t : A \parr B \quad \Delta; \Gamma' \vdash s : B^{\bot}}{\Delta; \Gamma, \Gamma' \vdash t \blacktriangledown s : A}\ \text{m-e}\parr_2$$

Typing rules 2/2

Exponential typing rules

$$\frac{\Delta; a : A \vdash t : \bot}{\Delta; \cdot \vdash !a.t : !A^{\bot}}\ \text{m-i!} \qquad \frac{\Delta; \Gamma \vdash s : !A \quad \Delta, u : A; \Gamma' \vdash t : B}{\Delta; \Gamma, \Gamma' \vdash t[!u := s] : B}\ \text{m-e!}$$

$$\frac{\Delta, u : A^{\bot}; \Gamma \vdash t : \bot}{\Delta; \Gamma \vdash ?u.t : ?A}\ \text{m-i?} \qquad \frac{\Delta; \Gamma \vdash s : ?A \quad \Delta; a : A \vdash t : \bot}{\Delta; \Gamma \vdash t \blacktriangleright^{a} s : \bot}\ \text{m-e?}$$

Typing rules 2/2

Exponential typing rules

$$\frac{\Delta; a : A \vdash t : \bot}{\Delta; \cdot \vdash !a.t : !A^{\bot}}\ \text{m-i!} \qquad \frac{\Delta; \Gamma \vdash s : !A \quad \Delta, u : A; \Gamma' \vdash t : B}{\Delta; \Gamma, \Gamma' \vdash t[!u := s] : B}\ \text{m-e!}$$

$$\frac{\Delta, u : A^{\bot}; \Gamma \vdash t : \bot}{\Delta; \Gamma \vdash ?u.t : ?A}\ \text{m-i?} \qquad \frac{\Delta; \Gamma \vdash s : ?A \quad \Delta; a : A \vdash t : \bot}{\Delta; \Gamma \vdash t \blacktriangleright^{a} s : \bot}\ \text{m-e?}$$

Units typing rules

$$\frac{\Delta; \Gamma \vdash t : B \quad \Delta; \Gamma' \vdash s : B^{\bot}}{\Delta; \Gamma, \Gamma' \vdash t \lightning s : \bot}\ \text{m-i}\bot$$

$$\frac{}{\Delta; \cdot \vdash \star : \mathbb{1}}\ \text{m-i}\mathbb{1} \qquad \frac{\Delta; \Gamma \vdash s : \mathbb{1} \quad \Delta; \Gamma' \vdash t : A}{\Delta; \Gamma, \Gamma' \vdash t[\star := s] : A}\ \text{m-e}\mathbb{1}$$

Observations regarding our typing system

- $\perp$ serves as a pivot for introducing the exponentials and $\parr$.

Observations regarding our typing system

- $\bot$ serves as a pivot for introducing the exponentials and $\parr$.
- There are many other alternative rules we can take and get a complete/sound system:

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

$$\frac{\Delta; \Gamma \vdash \bot}{\Delta; \Gamma \vdash ?A}\, \mathsf{w} \qquad \frac{\Delta; \Gamma \vdash A}{\Delta; \Gamma \vdash ?A}\, \mathsf{d} \qquad \frac{\Delta; \Gamma, !A^{\bot} \vdash ?A}{\Delta; \Gamma \vdash ?A}\, \mathsf{c}$$

but they don't lead to a well-behaved contra-substitution operation.

Observations regarding our typing system

- $\bot$ serves as a pivot for introducing the exponentials and $\parr$.
- There are many other alternative rules we can take and get a complete/sound system:

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

$$\frac{\Delta; \Gamma \vdash \bot}{\Delta; \Gamma \vdash ?A}\, \mathsf{w} \qquad \frac{\Delta; \Gamma \vdash A}{\Delta; \Gamma \vdash ?A}\, \mathsf{d} \qquad \frac{\Delta; \Gamma, !A^\bot \vdash ?A}{\Delta; \Gamma \vdash ?A}\, \mathsf{c}$$

  but they don't lead to a well-behaved contra-substitution operation.
- If we add the rule (in the style of Barbanera, Berardi (1996))

$$\frac{\Delta; \Gamma, a : A \vdash t : \bot}{\Delta; \Gamma \vdash \lambda a.\, t : A^\bot}$$

  where $\lambda a.\, t$ is shorthand for $\parr(a, b).t \leftlightning b$

  Then we could drop all of m-e$\otimes$, m-e$\parr_1$, m-e$\parr_2$, m-e!, m-e? and m-e$\mathbb{1}$. However, it ends up being non-confluent.

Reduction

We have the following rules for reduction that arise from introduction/elimination pairs. We assume both sides have equal type.

Reduction

We have the following rules for reduction that arise from introduction/elimination pairs. We assume both sides have equal type. Let $\mathsf{L}, \mathsf{K}, \ldots$ stand for $\mathsf{L} ::= \Box \mid \mathsf{L}[\langle a, b\rangle := t] \mid \mathsf{L}[\![!u := t]\!] \mid \mathsf{L}[\![\star := t]\!]$

Reduction

We have the following rules for reduction that arise from introduction/elimination pairs. We assume both sides have equal type. Let $\mathsf{L}, \mathsf{K}, \ldots$ stand for $\mathsf{L} ::= \Box \mid \mathsf{L}[\langle a, b\rangle := t] \mid \mathsf{L}[!u := t] \mid \mathsf{L}[\star := t]$

$$
\begin{array}{lll}
(\parr(a,b).t)\mathsf{L}\ r & \mapsto_{\beta\parr\mathsf{L}} & t\{a := r\}\{b \backslash\!\backslash \star\}\mathsf{L} \quad (b \notin \mathrm{fv}(r)) \\
(\parr(a,b).t)\mathsf{L}\blacktriangledown r & \mapsto_{\beta\parr\mathsf{R}} & t\{b := r\}\{a \backslash\!\backslash \star\}\mathsf{L} \quad (a \notin \mathrm{fv}(r)) \\
t[\langle a, b\rangle := \langle s, r\rangle \mathsf{L}] & \mapsto_{\beta\otimes} & t\{a := s\}\{b := r\}\mathsf{L} \quad (b \notin \mathrm{fv}(s)) \\
t[!u := (!a.s)\mathsf{L}] & \mapsto_{\beta!} & t\{u := s\{a \backslash\!\backslash \star\}\}\mathsf{L} \\
t\blacktriangleright^{a}(?u.s)\mathsf{L} & \mapsto_{\beta?} & s\{u := t\{a \backslash\!\backslash \star\}\}\mathsf{L}
\end{array}
$$

Reduction

We have the following rules for reduction that arise from introduction/elimination pairs. We assume both sides have equal type. Let $\mathsf{L}, \mathsf{K}, \ldots$ stand for $\mathsf{L} ::= \Box \mid \mathsf{L}[\langle a, b\rangle := t] \mid \mathsf{L}[!u := t] \mid \mathsf{L}[\star := t]$

$$
\begin{array}{llll}
(\parr(a,b).t)\mathsf{L}\ r & \mapsto_{\beta\parr\otimes\mathsf{L}} & t\{a := r\}\{b \backslash\!\backslash \star\}\mathsf{L} & (b \notin \mathrm{fv}(r)) \\
(\parr(a,b).t)\mathsf{L}\blacktriangledown r & \mapsto_{\beta\parr\otimes\mathsf{R}} & t\{b := r\}\{a \backslash\!\backslash \star\}\mathsf{L} & (a \notin \mathrm{fv}(r)) \\
t[\langle a, b\rangle := \langle s, r\rangle \mathsf{L}] & \mapsto_{\beta\otimes} & t\{a := s\}\{b := r\}\mathsf{L} & (b \notin \mathrm{fv}(s)) \\
t[!u := (!a.s)\mathsf{L}] & \mapsto_{\beta!} & t\{u := s\{a \backslash\!\backslash \star\}\}\mathsf{L} & \\
t\blacktriangleright^{a}(?u.s)\mathsf{L} & \mapsto_{\beta?} & s\{u := t\{a \backslash\!\backslash \star\}\}\mathsf{L} &
\end{array}
$$

- If $\Delta; \Gamma, a : A \vdash t : \bot$ then $\Delta; \Gamma \vdash t\{a \backslash\!\backslash \star\} : A^{\bot}$

Reduction

We have the following rules for reduction that arise from introduction/elimination pairs. We assume both sides have equal type. Let $\mathsf{L}, \mathsf{K}, \ldots$ stand for $\mathsf{L} ::= \Box \mid \mathsf{L}[\langle a, b\rangle := t] \mid \mathsf{L}[!u := t] \mid \mathsf{L}[\star := t]$

$$
\begin{array}{llll}
(\parr(a,b).t)\mathsf{L}\ r & \mapsto_{\beta\parr\mathsf{L}} & t\{a := r\}\{b \backslash\!\backslash \star\}\mathsf{L} & (b \notin \mathrm{fv}(r)) \\
(\parr(a,b).t)\mathsf{L}\blacktriangledown r & \mapsto_{\beta\parr\mathsf{R}} & t\{b := r\}\{a \backslash\!\backslash \star\}\mathsf{L} & (a \notin \mathrm{fv}(r)) \\
t[\langle a, b\rangle := \langle s, r\rangle \mathsf{L}] & \mapsto_{\beta\otimes} & t\{a := s\}\{b := r\}\mathsf{L} & (b \notin \mathrm{fv}(s)) \\
t[!u := (!a.s)\mathsf{L}] & \mapsto_{\beta!} & t\{u := s\{a \backslash\!\backslash \star\}\}\mathsf{L} & \\
t\blacktriangleright^{a}(?u.s)\mathsf{L} & \mapsto_{\beta?} & s\{u := t\{a \backslash\!\backslash \star\}\}\mathsf{L} &
\end{array}
$$

- If $\Delta; \Gamma, a : A \vdash t : \bot$ then $\Delta; \Gamma \vdash t\{a \backslash\!\backslash \star\} : A^{\bot}$
- We have no reduction rules for the introduction/elimination of $\mathbb{1}$

Reduction

We have the following rules for reduction that arise from introduction/elimination pairs. We assume both sides have equal type. Let $\mathsf{L}, \mathsf{K}, \ldots$ stand for $\mathsf{L} ::= \Box \mid \mathsf{L}[\langle a, b\rangle := t] \mid \mathsf{L}[!u := t] \mid \mathsf{L}[\star := t]$

$$
\begin{array}{lll}
(\parr(a,b).t)\mathsf{L}\ r & \mapsto_{\beta\parr\mathsf{L}} & t\{a := r\}\{b \backslash\!\backslash \star\}\mathsf{L} \quad (b \notin \mathrm{fv}(r)) \\
(\parr(a,b).t)\mathsf{L}\blacktriangledown r & \mapsto_{\beta\parr\mathsf{R}} & t\{b := r\}\{a \backslash\!\backslash \star\}\mathsf{L} \quad (a \notin \mathrm{fv}(r)) \\
t[\langle a, b\rangle := \langle s, r\rangle \mathsf{L}] & \mapsto_{\beta\otimes} & t\{a := s\}\{b := r\}\mathsf{L} \quad (b \notin \mathrm{fv}(s)) \\
t[!u := (!a.s)\mathsf{L}] & \mapsto_{\beta!} & t\{u := s\{a \backslash\!\backslash \star\}\}\mathsf{L} \\
t\blacktriangleright^{a}(?u.s)\mathsf{L} & \mapsto_{\beta?} & s\{u := t\{a \backslash\!\backslash \star\}\}\mathsf{L}
\end{array}
$$

- If $\Delta; \Gamma, a : A \vdash t : \bot$ then $\Delta; \Gamma \vdash t\{a \backslash\!\backslash \star\} : A^{\bot}$
- We have no reduction rules for the introduction/elimination of $\mathbb{1}$

⚠Unluckily, this doesn't guarantee confluence, thus we are forced to add the following two pairs of symmetrical reduction rules:

$$
\begin{array}{lll}
(\parr(a,b).t)\mathsf{L} \lightning \langle s, r\rangle \mathsf{K} & \mapsto_{\parr\otimes} & t\{a := s\}\{b := r\}\mathsf{LK} \quad (b \notin \mathrm{fv}(s)) \\
\langle s, r\rangle \mathsf{L} \lightning (\parr(a,b).t)\mathsf{K} & \mapsto_{\otimes\parr} & t\{a := s\}\{b := r\}\mathsf{LK} \quad (b \notin \mathrm{fv}(s)) \\
(!a.t)\mathsf{L} \lightning (?u.r)\mathsf{K} & \mapsto_{!?} & r\{u := t\{a \backslash\!\backslash \star\}\}\mathsf{LK} \\
(?u.r)\mathsf{L} \lightning (!a.t)\mathsf{K} & \mapsto_{?!} & r\{u := t\{a \backslash\!\backslash \star\}\}\mathsf{LK}
\end{array}
$$

Structural equivalence

We define it as the reflexive, symmetric, transitive and contextual closure of the following axioms. For each equation, we assume both sides to have identical type.

Structural equivalence

We define it as the reflexive, symmetric, transitive and contextual closure of the following axioms. For each equation, we assume both sides to have identical type.

$$\mathsf{S}\langle t\rangle[\mathrm{p} := r] \quad \equiv_{\mathsf{Prop}} \quad \mathsf{S}\langle t[\mathrm{p} := r]\rangle \quad (\mathrm{fv}(\mathrm{p}) \cap \mathrm{fv}(\mathsf{S}) = \varnothing \text{ and } r \text{ is free for } \mathsf{S})$$

We let eliminators float.

- Where $[\mathrm{p} := r]$ is any eliminator that is not an ?-eliminator.
- Where S is a context such that its hole is not inside $!a.\Box$ or $\Box\blacktriangleright^a s$

Structural equivalence

We define it as the reflexive, symmetric, transitive and contextual closure of the following axioms. For each equation, we assume both sides to have identical type.

$$\mathsf{S}\langle t\rangle[\mathrm{p} := r] \quad \equiv_{\mathsf{Prop}} \quad \mathsf{S}\langle t[\mathrm{p} := r]\rangle \quad (\mathrm{fv}(\mathrm{p}) \cap \mathrm{fv}(\mathsf{S}) = \varnothing \text{ and } r \text{ is free for } \mathsf{S})$$

We let eliminators float.

- Where $[\mathrm{p} := r]$ is any eliminator that is not an ?-eliminator.
- Where S is a context such that its hole is not inside $!a.\square$ or $\square \blacktriangleright^a s$

$$t[\star := \star] \quad \equiv_{\beta\star} \quad t$$

The $I\bot/E\bot$ redex is treated as an equation

Structural equivalence

We define it as the reflexive, symmetric, transitive and contextual closure of the following axioms. For each equation, we assume both sides to have identical type.

$$\mathsf{S}\langle t\rangle[\mathrm{p} := r] \quad \equiv_{\mathsf{Prop}} \quad \mathsf{S}\langle t[\mathrm{p} := r]\rangle \quad (\mathrm{fv}(\mathrm{p}) \cap \mathrm{fv}(\mathsf{S}) = \varnothing \text{ and } r \text{ is free for } \mathsf{S})$$

We let eliminators float.

- Where $[\mathrm{p} := r]$ is any eliminator that is not an ?-eliminator.
- Where S is a context such that its hole is not inside $!a.\Box$ or $\Box \blacktriangleright^a s$

$$t[\star := \star] \quad \equiv_{\beta\star} \quad t$$

The $I\bot/E\bot$ redex is treated as an equation

$$t[\star := s] \quad \equiv_{\mathsf{Symm}\star} \quad s[\star := t]$$

Proofs of $\mathbb{1}$ are interchangeable when they are eliminated

Structural equivalence

We define it as the reflexive, symmetric, transitive and contextual closure of the following axioms. For each equation, we assume both sides to have identical type.

$$\mathsf{S}\langle t\rangle[\mathsf{p} := r] \quad \equiv_{\mathsf{Prop}} \quad \mathsf{S}\langle t[\mathsf{p} := r]\rangle \quad (\mathrm{fv}(\mathsf{p}) \cap \mathrm{fv}(\mathsf{S}) = \varnothing \text{ and } r \text{ is free for } \mathsf{S})$$

We let eliminators float.

- Where $[\mathsf{p} := r]$ is any eliminator that is not an ?-eliminator.
- Where S is a context such that its hole is not inside $!a.\Box$ or $\Box\blacktriangleright^a s$

$$t[\star := \star] \quad \equiv_{\beta\star} \quad t$$

The $I\bot/E\bot$ redex is treated as an equation

$$t[\star := s] \quad \equiv_{\mathsf{Symm}\star} \quad s[\star := t]$$

Proofs of $\mathbb{1}$ are interchangeable when they are eliminated

$$t\{a \Vdash \star\}\lightning r \quad \equiv_{\mathsf{split}} \quad t\{a := r\}$$

Every term of type $\bot$ arises as a contradiction

Properties of the calculus

- **Logical soundness / completeness**.
  $\vdash \Gamma, A$ is valid in MELL iff there is a term t such that $\cdot; \Gamma^{\perp} \vdash t : A$.
- **Subject reduction**.
  If $\Delta; \Gamma \vdash t : A$ and $t \to s$ then $\Delta; \Gamma \vdash s : A$.
- **Structural equivalence is a strong bisimulation**.
  If $t \equiv s \to s'$ there exists t' such that $t \to t' \equiv s'$.

Properties of the calculus

- **Logical soundness / completeness**.
  $\vdash \Gamma, A$ is valid in MELL iff there is a term t such that $\cdot; \Gamma^{\perp} \vdash t : A$.
- **Subject reduction**.
  If $\Delta; \Gamma \vdash t : A$ and $t \to s$ then $\Delta; \Gamma \vdash s : A$.
- **Structural equivalence is a strong bisimulation**.
  If $t \equiv s \to s'$ there exists t' such that $t \to t' \equiv s'$.
- **Strong normalization**.
  Typable terms have no infinite reduction paths.
  Reducibility model inspired by the work `Accatoli (RTA 2013)`

Properties of the calculus

- **Logical soundness / completeness**.
  $\vdash \Gamma, A$ is valid in MELL iff there is a term t such that $\cdot; \Gamma^{\perp} \vdash t : A$.
- **Subject reduction**.
  If $\Delta; \Gamma \vdash t : A$ and $t \to s$ then $\Delta; \Gamma \vdash s : A$.
- **Structural equivalence is a strong bisimulation**.
  If $t \equiv s \to s'$ there exists t' such that $t \to t' \equiv s'$.
- **Strong normalization**.
  Typable terms have no infinite reduction paths.
  Reducibility model inspired by the work `Accatoli (RTA 2013)`
- **Confluence modulo** $\equiv$.
  If $t \twoheadrightarrow t_1$ and $t \twoheadrightarrow t_2$ then there are t_1' and t_2' such that: $t_1 \twoheadrightarrow t_1'$, $t_2 \twoheadrightarrow t_2'$ and $t_1' \equiv t_2'$.

The $\lambda\mu$-calculus

It is a calculus for classical logic, first appeared in Parigot (1992).

The $\lambda\mu$-calculus

It is a calculus for classical logic, first appeared in Parigot (1992).

Syntax

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

The $\lambda\mu$-calculus

It is a calculus for classical logic, first appeared in Parigot (1992).

Syntax

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

Typing rules (for the new constructors)

$$\frac{\Gamma \vdash M : A \mid \Sigma, a : A}{\Gamma \vdash [a]M : \bot \mid \Sigma, a : A}\ \text{p-name} \qquad \frac{\Gamma \vdash M : \bot \mid \Sigma, a : A}{\Gamma \vdash \mu a.M : A \mid \Sigma}\ \text{p-mu}$$

The $\lambda\mu$-calculus

It is a calculus for classical logic, first appeared in Parigot (1992).

Syntax

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

Typing rules (for the new constructors)

$$\frac{\Gamma \vdash M : A \mid \Sigma, a : A}{\Gamma \vdash [a]M : \bot \mid \Sigma, a : A}\ \text{p-name} \qquad \frac{\Gamma \vdash M : \bot \mid \Sigma, a : A}{\Gamma \vdash \mu a.M : A \mid \Sigma}\ \text{p-mu}$$

Reduction rules ($\rightarrow_\mu$)

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

$M\{a \lhd N\}$ is the **structural substitution** replaces subterms of M of the form $[a]O$ by $[a](O\ N)$.

The $\lambda\mu$-calculus

It is a calculus for classical logic, first appeared in Parigot (1992).

Syntax

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

Typing rules (for the new constructors)

$$\frac{\Gamma \vdash M : A \mid \Sigma, a : A}{\Gamma \vdash [a]M : \bot \mid \Sigma, a : A}\ \text{p-name} \qquad \frac{\Gamma \vdash M : \bot \mid \Sigma, a : A}{\Gamma \vdash \mu a.M : A \mid \Sigma}\ \text{p-mu}$$

Reduction rules ($\rightarrow_\mu$)

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

$M\{a \lhd N\}$ is the **structural substitution** replaces subterms of M of the form $[a]O$ by $[a](O\ N)$.

T-Translation for $\lambda\mu$-calculus

T-translation for $\lambda\mu$ (formulae) cf. Danos (1995)

$$\begin{array}{rcl@{\qquad}rcl} \bot^{\mathsf{T}} & := & \bot & \alpha^{\mathsf{T}} & := & \alpha \\ (A \supset B)^{\mathsf{T}} & := & !?A^{\mathsf{T}} \multimap ?B^{\mathsf{T}} = ?!(A^{\mathsf{T}})^{\bot} \parr ?B^{\mathsf{T}} & & & \end{array}$$

T-Translation for $\lambda\mu$-calculus

T-translation for $\lambda\mu$ (formulae) cf. Danos (1995)

$$\begin{array}{rcl@{\qquad}rcl} \bot^{\mathsf{T}} & := & \bot & \alpha^{\mathsf{T}} & := & \alpha \\ (A \supset B)^{\mathsf{T}} & := & !?A^{\mathsf{T}} \multimap ?B^{\mathsf{T}} = ?!(A^{\mathsf{T}})^{\bot} \parr ?B^{\mathsf{T}} & & & \end{array}$$

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

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

where $!t$ abbreviates $!a.(t \lightning a)$ and k is a fresh unrestricted variable

T-Translation for $\lambda\mu$-calculus

T-translation for $\lambda\mu$ (formulae) cf. Danos (1995)

$$\begin{array}{rcl@{\qquad}rcl} \bot^{\mathsf{T}} & := & \bot & \alpha^{\mathsf{T}} & := & \alpha \\ (A \supset B)^{\mathsf{T}} & := & !?A^{\mathsf{T}} \multimap ?B^{\mathsf{T}} = ?!(A^{\mathsf{T}})^{\bot} \parr ?B^{\mathsf{T}} & & & \end{array}$$

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

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

where $!t$ abbreviates $!a.(t \lightning a)$ and k is a fresh unrestricted variable

Proposition

- If $\Gamma \vdash M : A \mid \Sigma$ holds in $\lambda\mu$, then $?\Gamma^{\mathsf{T}}, (\Sigma^{\mathsf{T}})^{\bot}, k : (A^{\mathsf{T}})^{\bot}; \cdot \vdash M^{\mathsf{T}} : \bot$ holds in λ_{MELL}
- If $M \to_{\mu} N$, then $M^{\mathsf{T}} \twoheadrightarrow N^{\mathsf{T}}$.

T-Translation for $\lambda\mu$-calculus

T-translation for $\lambda\mu$ (formulae) cf. Danos (1995)

$$\begin{array}{rcl@{\qquad}rcl} \bot^{\mathsf{T}} & := & \bot & \alpha^{\mathsf{T}} & := & \alpha \\ (A \supset B)^{\mathsf{T}} & := & !?A^{\mathsf{T}} \multimap ?B^{\mathsf{T}} = ?!(A^{\mathsf{T}})^{\bot} \parr ?B^{\mathsf{T}} & & & \end{array}$$

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

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

where $!t$ abbreviates $!a.(t \lightning a)$ and k is a fresh unrestricted variable

Proposition

- If $\Gamma \vdash M : A \mid \Sigma$ holds in $\lambda\mu$, then $?\Gamma^{\mathsf{T}}, (\Sigma^{\mathsf{T}})^{\bot}, k : (A^{\mathsf{T}})^{\bot}; \cdot \vdash M^{\mathsf{T}} : \bot$ holds in λ_{MELL}
- If $M \to_{\mu} N$, then $M^{\mathsf{T}} \twoheadrightarrow N^{\mathsf{T}}$.

Conclusion

This work

- New calculi for MLL and MELL based on the construction of the **contrasubstitution**.
- Good properties such as confluence (modulo $\equiv$), strong normalization, etc.
- Translations from classical calculi via T and Q translations cf. Danos (1995):
  - T and Q translation for $\lambda\mu$ Parigot (1992).
  - T translation for $\overline{\lambda}\mu\tilde{\mu}$ Curien-Herbelin (2000).
  - Translation for μDCLL Hasegawa (2005).
- Subformula property
- Intuitionistic fragment based on input/output formula.

Conclusion

This work

- New calculi for MLL and MELL based on the construction of the **contrasubstitution**.
- Good properties such as confluence (modulo $\equiv$), strong normalization, etc.
- Translations from classical calculi via T and Q translations cf. Danos (1995):
  - T and Q translation for $\lambda\mu$ Parigot (1992).
  - T translation for $\overline{\lambda}\mu\tilde{\mu}$ Curien-Herbelin (2000).
  - Translation for μDCLL Hasegawa (2005).
- Subformula property
- Intuitionistic fragment based on input/output formula.

Future work

- Relate λ_{MELL} with proof nets.
- Extensions: additives, fixed points, etc
- Can we formulate an untyped version of λ_{MELL}?
- Translations *into* other calculi.

Contrasubstitution – Definition in λ_{MELL}

A term t is **linear** in λ_{MELL} if it is in the λ_{MLL} sense and moreover. 1) subterms of the form $!a.t$ have no free linear variables; 2) and in subterms of the form $t \blacktriangleright^a s$, only a occurs as a free linear variable in t.
Typable terms are linear!

We define contrasubstitution by induction on the structure of a linear term t such that $a \in \mathrm{fv}(t)$. We omit the repeated and impossible cases.

$$
\begin{array}{rcl}
(t_1[\langle b,c\rangle := t_2])\{a \backslash\backslash s\} & := & \begin{cases} t_1\{a \backslash\backslash s\}[\langle b,c\rangle := t_2] & \text{if } a \in \mathrm{fv}(t_1), a \notin \{b,c\} \\ t_2\{a \backslash\backslash \parr(b,c).(t_1 \lightning s)\} & \text{if } a \in \mathrm{fv}(t_2) \end{cases} \\
(\parr(b,c).t)\{a \backslash\backslash s\} & := & t\{a \backslash\backslash \star\}[\langle b,c\rangle := s] \\
t_1[!u := t_2]\{a \backslash\backslash s\} & := & \begin{cases} t_1\{a \backslash\backslash s\}[!u := t_2] & \text{if } a \in \mathrm{fv}(t_1) \\ t_2\{a \backslash\backslash ?u.t_1 \lightning s\} & \text{if } a \in \mathrm{fv}(t_2) \end{cases} \\
(?u.t)\{a \backslash\backslash s\} & := & t\{a \backslash\backslash \star\}[!u := s] \\
(t_1 \blacktriangleright^b t_2)\{a \backslash\backslash s\} & := & t_2\{a \backslash\backslash !b.t_1\}[\star := s] \\
(t_1 \lightning t_2)\{a \backslash\backslash s\} & := & \begin{cases} t_1\{a \backslash\backslash t_2\}[\star := s] & \text{if } a \in \mathrm{fv}(t_1) \\ t_2\{a \backslash\backslash t_1\}[\star := s] & \text{if } a \in \mathrm{fv}(t_2) \end{cases} \\
t_1[\star := t_2]\{a \backslash\backslash s\} & := & \begin{cases} t_1\{a \backslash\backslash s\}[\star := t_2] & \text{if } a \in \mathrm{fv}(t_1) \\ t_2\{a \backslash\backslash t_1 \lightning s\} & \text{if } a \in \mathrm{fv}(t_2) \end{cases}
\end{array}
$$

Contrasubstitution – Example in $\lambda_{\mathbf{MELL}}$

Let $t = ?u.(\parr(c,d).r \lightning \langle a, e\rangle)$, where r is a linear term with no occurrence of the linear variable a. We compute $t\{a \backslash\!\backslash s\}$ for a linear term s. First, by definition of contra-substitution on ?-introduction:

$$t\{a \backslash\!\backslash s\} = \parr(c,d).r \lightning \langle a, e\rangle\{a \backslash\!\backslash \star\}[!u := s].$$

Because the sole occurrence of a lies in the right subterm of $\lightning$, this rewrites to $\langle a, e\rangle\{a \backslash\!\backslash \parr(c,d).r\}[!u := s][\star := \star]$. Finally, since a occurs in the left position of the pair, we obtain:

$$a\{a \backslash\!\backslash ((\parr(c,d).r)\blacktriangledown e)\}[!u := s][\star := \star] \;=\; ((\parr(c,d).r)\blacktriangledown e)[!u := s][\star := \star].$$

Need of structural substitution for Confluence – Example in $\lambda_{\mathbf{MELL}}$

Let $t = \overline{(\parr(a,b).\underline{(\parr(c,d).s)[\star := a]\ p)} \blacktriangledown r}$, where, say, $b \in \mathrm{fv}(s)$.