The λ-calculus, typed and untyped

Lecture 3: Strong normalization of the simply typed λ-calculus and more about Curry–Howard

Giulio Guerrieri

✉ giulio.guerrieri@uibk.ac.at 🌐 https://cl-teaching.uibk.ac.at/26w/lc

Institut für Informatik, Universität Innsbruck

Innsbruck, Austria – 30 October 2026

Outline

1. Natural deduction for minimal logic, revisited
2. The Curry-Howard correspondence, more formally
3. Strong normalization of the simply typed λ-calculus
4. Logic and/vs Computation
5. Conclusion, exercises and bibliography

Outline

1. Natural deduction for minimal logic, revisited
2. The Curry-Howard correspondence, more formally
3. Strong normalization of the simply typed λ-calculus
4. Logic and/vs Computation
5. Conclusion, exercises and bibliography

ND revisited (refining the formulation from Lecture 1)

Def. We fix an infinite set of labels ranged over by $x, y, z, \dots$. A labeled formula is a pair (x, A), also denoted by $x : A$ or A^x, where A is a formula and x is a label.
A set Γ of labeled formulas is coherent if $A = B$ whenever $A^x, B^x \in \Gamma$.

Def. A derivation $\mathcal{D}$ in ND with hypotheses $h(\mathcal{D})$ and conclusion $c(\mathcal{D})$ is:

- either A^x for any formula A and any label x, with $h(\mathcal{D}) = \{A^x\}$ and $c(\mathcal{D}) = A$;
- or it is obtained from derivations $\mathcal{D}'$, $\mathcal{D}_1$, $\mathcal{D}_2$ by applying one of the inference rules

$\Rightarrow$ elimination

$$\dfrac{\begin{array}{c}\vdots\,\mathcal{D}_1\\ A \Rightarrow B\end{array} \quad \begin{array}{c}\vdots\,\mathcal{D}_2\\ A\end{array}}{B}\Rightarrow_e$$

provided that $h(\mathcal{D}_1) \cup h(\mathcal{D}_2)$ is coherent

with $h(\mathcal{D}) = h(\mathcal{D}_1) \cup h(\mathcal{D}_2)$ and $c(\mathcal{D}) = B$

$\Rightarrow$ introduction

$$\dfrac{\begin{array}{c}[A^x]\\ \vdots\,\mathcal{D}'\\ B\end{array}}{A \Rightarrow B}\Rightarrow_i^x$$

provided that *all* occurrences of A^x in $\mathcal{D}'$ are discarded by $\Rightarrow_i^x$

with $h(\mathcal{D}) = h(\mathcal{D}') \setminus \{A^x\}$ and $c(\mathcal{D}) = A \Rightarrow B$

Idea. The label x determines how many occurrences of A are discharged by $\Rightarrow_i^x$.

Examples of derivations in ND, revisited (compare them with Lecture 1)

Notation. $A_1^{x_1}, \dots, A_n^{x_n} \vdash B$ (or $A_1^{x_1}, \dots, A_n^{x_n} \vdash_{\mathsf{ND}} B$) means that there is a derivation in ND with conclusion B and hypotheses *among* the labeled formulas $A_1^{x_n}, \dots, A_n^{x_n}$.

1. Prove that $A^x \vdash AA^x \vdash A$, and $\vdash A \Rightarrow A \vdash A \Rightarrow A$, and $B^y \vdash A \Rightarrow AB^y \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A \vdash A \Rightarrow B \Rightarrow A$.

$$A^x \qquad \frac{[A^x]}{A \Rightarrow A}\Rightarrow_i^x \qquad \frac{[A^x]}{A \Rightarrow A}\Rightarrow_i^x \qquad \frac{\dfrac{[A^x]}{B \Rightarrow A}\Rightarrow_i^y}{A \Rightarrow B \Rightarrow A}\Rightarrow_i^x$$

2. Give two distinct derivations of $A^x \vdash A \Rightarrow AA^x \vdash A \Rightarrow A$, and two distinct ones of $\vdash A \Rightarrow A \Rightarrow A \vdash A \Rightarrow A \Rightarrow A$.

$$\frac{A^x}{A \Rightarrow A}\Rightarrow_i^y \quad \neq \quad \frac{[A^y]}{A \Rightarrow A}\Rightarrow_i^y \qquad \frac{\dfrac{[A^x]}{A \Rightarrow A}\Rightarrow_i^y}{A \Rightarrow A \Rightarrow A}\Rightarrow_i^x \quad \neq \quad \frac{\dfrac{[A^x]}{A \Rightarrow A}\Rightarrow_i^x}{A \Rightarrow A \Rightarrow A}\Rightarrow_i^y$$

3. Prove $\vdash (A \Rightarrow B) \Rightarrow (B \Rightarrow C) \Rightarrow A \Rightarrow C \vdash (A \Rightarrow B) \Rightarrow (B \Rightarrow C) \Rightarrow A \Rightarrow C$ and $\vdash (A \Rightarrow (B \Rightarrow C)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow C) \vdash (A \Rightarrow (B \Rightarrow C)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow C)$.

$$\dfrac{\dfrac{\dfrac{[(B \Rightarrow C)^y] \quad \dfrac{[(A \Rightarrow B)^x] \quad [A^z]}{B}\Rightarrow_e}{C}\Rightarrow_e}{A \Rightarrow C}\Rightarrow_i^z}{(B \Rightarrow C) \Rightarrow A \Rightarrow C}\Rightarrow_i^y}{(A \Rightarrow B) \Rightarrow (B \Rightarrow C) \Rightarrow A \Rightarrow C}\Rightarrow_i^x$$

$$\dfrac{\dfrac{\dfrac{[(A \Rightarrow B \Rightarrow C)^x] \quad [A^z]}{B \Rightarrow C}\Rightarrow_e \quad \dfrac{[(A \Rightarrow B)^y] \quad [A^z]}{B}\Rightarrow_e}{C}\Rightarrow_e}{A \Rightarrow C}\Rightarrow_i^z}{(A \Rightarrow B) \Rightarrow (A \Rightarrow C)}\Rightarrow_i^y}{(A \Rightarrow (B \Rightarrow C)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow C)}\Rightarrow_i^x$$

$\mathsf{ND}_{\mathsf{seq}}$ revisited (refining the formulation from Lecture 1)

Def. An environment is a partial function with finite domain from labels to formulas. An environment Γ is denoted by $x_1 : A_1, \dots, x_n : A_n$ if $\mathrm{dom}(\Gamma) = \{x_1, \dots, x_n\}$ and $\Gamma(x_i) = A_i$.

A sequent is a pair $\Gamma \vdash A$ of an environment Γ and a formula A.

Def. A derivation in $\mathsf{ND}_{\mathsf{seq}}$ is a finite tree built up from the inference rules below.

$$\frac{}{\Gamma,\, x : A \vdash A}\,\mathsf{ax}^x \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\Rightarrow_e \qquad \frac{\Gamma,\, x : A \vdash B}{\Gamma \vdash A \Rightarrow B}\Rightarrow_i^x$$

Notations. $\mathcal{D} \triangleright_{\mathsf{ND}_{\mathsf{seq}}} \Gamma \vdash A$ means that $\mathcal{D}$ is a derivation in $\mathsf{ND}_{\mathsf{seq}}$ with conclusion $\Gamma \vdash A$.
$\Gamma \vdash_{\mathsf{ND}_{\mathsf{seq}}} A$ means that $\Gamma \vdash A$ is derivable in $\mathsf{ND}_{\mathsf{seq}}$ i.e. there is a derivation $\mathcal{D} \triangleright_{\mathsf{ND}_{\mathsf{seq}}} \Gamma \vdash A$.

Idea. In $\mathsf{ND}_{\mathsf{seq}}$ a derivation makes explicit:

1. the current hypotheses at each inference step;
2. if two occurrences of the same formula should be treated separately (different labels).

Rmk. $x : X, y : X$ is an environment, while $x : X, x : X \Rightarrow X$ is not an environment!

Examples of derivations in $\mathsf{ND}_{\mathsf{seq}}$, revisited (compare them with Lecture 1)

1. Prove $x : A \vdash Ax : A \vdash A$, and $\vdash A \Rightarrow A \vdash A \Rightarrow A$, and $y : B \vdash A \Rightarrow Ay : B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A \vdash A \Rightarrow B \Rightarrow A$.

$$\frac{}{x : A \vdash A}\,\mathsf{ax}^x \qquad \frac{\dfrac{}{x : A \vdash A}\,\mathsf{ax}^x}{\vdash A \Rightarrow A}\Rightarrow_i^x \qquad \frac{}{x : A,\, y : B \vdash A}\,\mathsf{ax}^x \qquad \frac{\dfrac{}{x : A,\, y : B \vdash A}\,\mathsf{ax}^x}{x : A \vdash B \Rightarrow A}\Rightarrow_i^y}{\vdash A \Rightarrow B \Rightarrow A}\Rightarrow_i^x$$

2. Give two distinct derivations of $x : A \vdash A \Rightarrow Ax : A \vdash A \Rightarrow A$, and two distinct ones of $\vdash A \Rightarrow A \Rightarrow A \vdash A \Rightarrow A \Rightarrow A$.

$$\frac{\dfrac{}{y : A,\, x : A \vdash A}\,\mathsf{ax}^x}{x : A \vdash A \Rightarrow A}\Rightarrow_i^y \quad \neq \quad \frac{\dfrac{}{y : A,\, x : A \vdash A}\,\mathsf{ax}^y}{x : A \vdash A \Rightarrow A}\Rightarrow_i^y \qquad \frac{\dfrac{}{y : A,\, x : A \vdash A}\,\mathsf{ax}^x}{\vdash A \Rightarrow A \Rightarrow A}\Rightarrow_i^x \quad \neq \quad \frac{\dfrac{}{y : A,\, x : A \vdash A}\,\mathsf{ax}^x}{y : A \vdash A \Rightarrow A}\Rightarrow_i^x}{\vdash A \Rightarrow A \Rightarrow A}\Rightarrow_i^y$$

3. Prove $\vdash (A \Rightarrow B) \Rightarrow (B \Rightarrow C) \Rightarrow A \Rightarrow C \vdash (A \Rightarrow B) \Rightarrow (B \Rightarrow C) \Rightarrow A \Rightarrow C$ and $\vdash (A \Rightarrow (B \Rightarrow C)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow C) \vdash (A \Rightarrow (B \Rightarrow C)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow C)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash B \Rightarrow C}\,\mathsf{ax}^y \quad \dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B}\,\mathsf{ax}^x \quad \dfrac{}{\Gamma \vdash A}\,\mathsf{ax}^z}{\Gamma \vdash B}\Rightarrow_e}{z : A,\, x : A \Rightarrow B,\, y : B \Rightarrow C \vdash C}\Rightarrow_e}{x : A \Rightarrow B,\, y : B \Rightarrow C \vdash A \Rightarrow C}\Rightarrow_i^z}{x : A \Rightarrow B \vdash (B \Rightarrow C) \Rightarrow A \Rightarrow C}\Rightarrow_i^y}{\vdash (A \Rightarrow B) \Rightarrow (B \Rightarrow C) \Rightarrow A \Rightarrow C}\Rightarrow_i^x$$

where $\Gamma = z : A,\, x : A \Rightarrow B,\, y : B \Rightarrow C$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B \Rightarrow C}\,\mathsf{ax}^x \quad \dfrac{}{\Gamma \vdash A}\,\mathsf{ax}^z}{\Gamma \vdash B \Rightarrow C}\Rightarrow_e \quad \dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B}\,\mathsf{ax}^y \quad \dfrac{}{\Gamma \vdash A}\,\mathsf{ax}^z}{\Gamma \vdash B}\Rightarrow_e}{z : A,\, y : A \Rightarrow B,\, x : A \Rightarrow B \Rightarrow C \vdash C}\Rightarrow_e}{y : A \Rightarrow B,\, x : A \Rightarrow B \Rightarrow C \vdash A \Rightarrow C}\Rightarrow_i^z}{x : A \Rightarrow B \Rightarrow C \vdash (A \Rightarrow B) \Rightarrow A \Rightarrow C}\Rightarrow_i^y}{\vdash (A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow C}\Rightarrow_i^x$$

where $\Gamma = z : A,\, y : A \Rightarrow B,\, x : A \Rightarrow B \Rightarrow C$

Substitution of derivations in $\mathsf{ND}_{\mathsf{seq}}$

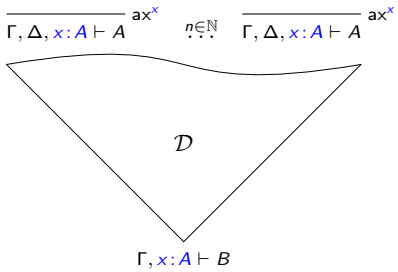
Def. Given the derivations $\mathcal{D} \triangleright_{\mathsf{ND}_{\mathsf{seq}}} \Gamma, x : A \vdash B$ and $\mathcal{D}' \triangleright_{\mathsf{ND}_{\mathsf{seq}}} \Gamma \vdash A$, the substitution of $\mathcal{D}'$ for the ax rules labeled by x in $\mathcal{D}$, denoted by $\mathcal{D}\{\mathcal{D}'/x\}$, is defined by induction on $\mathcal{D}$:

- if $\mathcal{D} = \dfrac{}{\Gamma, x : A \vdash A}\,\mathsf{ax}^x$ then $A = B$ and $\mathcal{D}\{\mathcal{D}'/x\} = \mathcal{D}'$;
- if $\mathcal{D} = \dfrac{}{\Gamma, x : A \vdash B}\,\mathsf{ax}^y$ where $x \neq y$, then $\Gamma(y) = B$ and $\mathcal{D}\{\mathcal{D}'/x\} = \dfrac{}{\Gamma \vdash B}\,\mathsf{ax}^y$;
- if $\mathcal{D} = \dfrac{\vdots\ \mathcal{D}_0}{\Gamma, x : A, y : C \vdash D}{\Gamma, x : A \vdash C \Rightarrow D}\Rightarrow_i^y$ with $y \notin \mathrm{dom}(\Gamma) \cup \{x\}$, then $\mathcal{D}\{\mathcal{D}'/x\} = \dfrac{\vdots\ \mathcal{D}_0\{\widehat{\mathcal{D}'}/x\}}{\Gamma, y : C \vdash D}{\Gamma \vdash C \Rightarrow D}\Rightarrow_i^y$
  where $\widehat{\mathcal{D}'}$ is obtained from $\mathcal{D}'$ by adding $y : C$ to the environment of each ax rule;
- if $\mathcal{D} = \dfrac{\vdots\ \mathcal{D}_0 \quad \vdots\ \mathcal{D}_1}{\Gamma, x : A \vdash C \Rightarrow B \quad \Gamma, x : A \vdash C}{\Gamma, x : A \vdash B}\Rightarrow_e$ then $\mathcal{D}\{\mathcal{D}'/x\} = \dfrac{\vdots\ \mathcal{D}_0\{\mathcal{D}'/x\} \quad \vdots\ \mathcal{D}_1\{\mathcal{D}'/x\}}{\Gamma \vdash C \Rightarrow B \quad \Gamma \vdash C}{\Gamma \vdash B}\Rightarrow_e$.

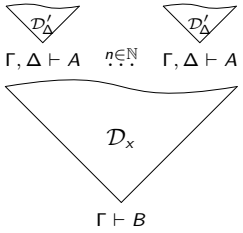
Rmk. If $\mathcal{D} \triangleright_{\mathsf{ND}_{\mathsf{seq}}} \Gamma, x : A \vdash B$ and $\mathcal{D}' \triangleright_{\mathsf{ND}_{\mathsf{seq}}} \Gamma \vdash A$, then $\mathcal{D}\{\mathcal{D}'/x\} \triangleright_{\mathsf{ND}_{\mathsf{seq}}} \Gamma \vdash B$.

Idea. Substitution in $\mathsf{ND}_{\mathsf{seq}}$ corresponds to composition of derivations in ND (see $\to_{\mathsf{cut}}$).

Substitution of derivations in $\mathrm{ND}_{\mathrm{seq}}$, graphically

Given $\mathcal{D} =$ 

and $\mathcal{D}' =$ 

then $\mathcal{D}\{\mathcal{D}'/x\} =$ 

where

- $\mathcal{D}_x$ is obtained from $\mathcal{D}$ by dropping $x : A$ from the environment in every sequent,
- $\mathcal{D}'_\Delta$ is obtained from $\mathcal{D}'$ by adding Δ to the environment in every sequent.

An example of substitution of derivations in $\mathrm{ND}_{\mathrm{seq}}$

Let $\Delta = z : A,\ y : A \Rightarrow B$ and $C = A \Rightarrow B \Rightarrow A$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Delta, x:C \vdash A \Rightarrow B \Rightarrow A}\,\mathrm{ax}^x \quad \dfrac{}{\Delta, x:C \vdash A}\,\mathrm{ax}^z}{\Delta, x:C \vdash B \Rightarrow A}\Rightarrow_e \quad \dfrac{\dfrac{}{\Delta, x:C \vdash A \Rightarrow B}\,\mathrm{ax}^y \quad \dfrac{}{\Delta, x:C \vdash A}\,\mathrm{ax}^z}{\Delta, x:C \vdash B}\Rightarrow_e}{\Delta,\ x : A \Rightarrow B \Rightarrow A \vdash A}\Rightarrow_e}{y : A \Rightarrow B,\ x : A \Rightarrow B \Rightarrow A \vdash A \Rightarrow A}\Rightarrow^z_i}{x : A \Rightarrow B \Rightarrow A \vdash (A \Rightarrow B) \Rightarrow A \Rightarrow A}\Rightarrow^y_i$$

$\mathcal{D}'$

$$\dfrac{\dfrac{\dfrac{}{a:A, b:B \vdash A}\,\mathrm{ax}^a}{a:A \vdash B \Rightarrow A}\Rightarrow^b_i}{\vdash A \Rightarrow B \Rightarrow A}\Rightarrow^a_i$$

$$\mathcal{D}\{\mathcal{D}'/x\} = \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Delta,\ a : A,\ b : B \vdash A}\,\mathrm{ax}^a}{\Delta,\ a : A \vdash B \Rightarrow A}\Rightarrow^b_i}{\Delta \vdash A \Rightarrow B \Rightarrow A}\Rightarrow^a_i \quad \dfrac{}{\Delta \vdash A}\,\mathrm{ax}^z}{\Delta \vdash B \Rightarrow A}\Rightarrow_e \quad \dfrac{\dfrac{}{\Delta \vdash A \Rightarrow B}\,\mathrm{ax}^y \quad \dfrac{}{\Delta \vdash A}\,\mathrm{ax}^z}{\Delta \vdash B}\Rightarrow_e}{\Delta \vdash A}\Rightarrow_e}{y : A \Rightarrow B \vdash A \Rightarrow A}\Rightarrow^z_i}{\vdash (A \Rightarrow B) \Rightarrow A \Rightarrow A}\Rightarrow^y_i$$

Cut-elimination in $\mathrm{ND}_{\mathrm{seq}}$

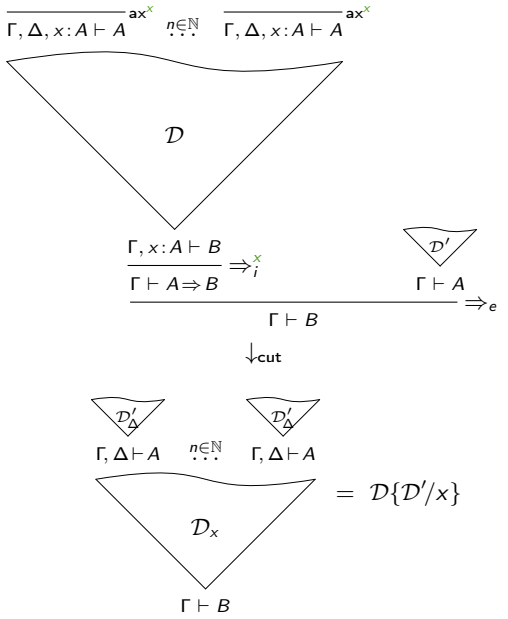
Def. A cut-formula in a derivation $\mathcal{D}$ is a formula occurrence on the right of $\vdash$ in a sequent in $\mathcal{D}$ that is the conclusion of a $\Rightarrow_i$ rule and the left premise of a $\Rightarrow_e$ rule.

Def. A redex in a derivation $\mathcal{D}$ is a pair of $\Rightarrow_i/\Rightarrow_e$ rules in $\mathcal{D}$ containing a cut-formula.

$$\dfrac{\dfrac{\vdots\ \mathcal{D}}{\Gamma, x : A \vdash B}}{\dfrac{\Gamma \vdash A \Rightarrow B}{\Gamma \vdash B}\Rightarrow_e}\Rightarrow^x_i \quad \vdots\ \mathcal{D}' \qquad \rightarrow_{\mathrm{cut}} \qquad \vdots\ \mathcal{D}\{\mathcal{D}'/x\}$$
$$\Gamma \vdash A \qquad\qquad\qquad \Gamma \vdash B$$

Rmk. A label on rule $\Rightarrow_i$ (resp. ax) is vital to define cut-elimination (resp. substitution).

The cut-elimination step in $\mathrm{ND}_{\mathrm{seq}}$, graphically



$= \mathcal{D}\{\mathcal{D}'/x\}$

An example of a cut-elimination step in $\mathsf{ND}_{\mathsf{seq}}$

Let $\Delta = z : A,\ y : A \Rightarrow B$ and $C = A \Rightarrow B \Rightarrow A$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Delta, x : C \vdash A \Rightarrow B \Rightarrow A}\,\mathsf{ax}^x \quad \dfrac{}{\Delta, x : C \vdash A}\,\mathsf{ax}^z}{\Delta, x : C \vdash B \Rightarrow A}\Rightarrow_e \quad \dfrac{\dfrac{}{\Delta, x : C \vdash A \Rightarrow B}\,\mathsf{ax}^y \quad \dfrac{}{\Delta, x : C \vdash A}\,\mathsf{ax}^z}{\Delta, x : C \vdash B}\Rightarrow_e}{\Delta,\ x : A \Rightarrow B \Rightarrow A \vdash B}\Rightarrow_e}{y : A \Rightarrow B,\ x : A \Rightarrow B \Rightarrow A \vdash A \Rightarrow A}\Rightarrow_i^z}{x : A \Rightarrow B \Rightarrow A \vdash (A \Rightarrow B) \Rightarrow A \Rightarrow A}\Rightarrow_i^y \quad \dfrac{\dfrac{\dfrac{}{a : A,\ b : B \vdash A}\,\mathsf{ax}^a}{a : A \vdash B \Rightarrow A}\Rightarrow_i^b}{\vdash A \Rightarrow B \Rightarrow A}\Rightarrow_i^a}{\vdash (A \Rightarrow B) \Rightarrow A \Rightarrow A}\Rightarrow_i^x$$

$$\downarrow \mathsf{cut}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Delta, a : A, b : B \vdash A}\,\mathsf{ax}^a}{\Delta, a : A \vdash B \Rightarrow A}\Rightarrow_i^b}{\Delta \vdash A \Rightarrow B \Rightarrow A}\Rightarrow_i^a \quad \dfrac{}{\Delta \vdash A}\,\mathsf{ax}^z}{\Delta \vdash B \Rightarrow A}\Rightarrow_e \quad \dfrac{\dfrac{}{\Delta \vdash A \Rightarrow B}\,\mathsf{ax}^y \quad \dfrac{}{\Delta \vdash A}\,\mathsf{ax}^z}{\Delta \vdash B}\Rightarrow_e}{\Delta \vdash B}\Rightarrow_e}{y : A \Rightarrow B \vdash A \Rightarrow A}\Rightarrow_i^z}{\vdash (A \Rightarrow B) \Rightarrow A \Rightarrow A}\Rightarrow_i^y$$

ND and $\mathsf{ND}_{\mathsf{seq}}$ are just different presentations of the same proof system

1. A derivation $\mathcal{D} \triangleright_{\mathsf{ND}} h(\mathcal{D}) \vdash A$ maps to a derivation $\mathcal{D}^\circ \triangleright_{\mathsf{ND}_{\mathsf{seq}}} h(\mathcal{D}) \vdash A$ defined by:

$$(A^x)^\circ = \dfrac{}{x : A \vdash A}\,\mathsf{ax}^x \qquad \left(\begin{array}{c}[A^x]\\ \vdots\,\mathcal{D}\\ \dfrac{B}{A \Rightarrow B}\Rightarrow_i^x\end{array}\right)^\circ = \dfrac{\begin{array}{c}\vdots\,\mathcal{D}^\circ\\ h(\mathcal{D}) \vdash B\end{array}}{h(\mathcal{D}) \smallsetminus \{A^x\} \vdash A \Rightarrow B}\Rightarrow_i^x$$

$$\left(\dfrac{\begin{array}{cc}\vdots\,\mathcal{D}_1 & \vdots\,\mathcal{D}_2\\ A \Rightarrow B & A\end{array}}{B}\Rightarrow_e\right)^\circ = \dfrac{\begin{array}{cc}\vdots\,(\mathcal{D}_1^\circ)_{h(\mathcal{D}_2)} & \vdots\,(\mathcal{D}_2^\circ)_{h(\mathcal{D}_1)}\\ \Delta \vdash A \Rightarrow B & \Delta \vdash A\end{array}}{\Delta \vdash A \Rightarrow B}\Rightarrow_e$$

where $\Delta = h(\mathcal{D}_1) \cup h(\mathcal{D}_2)$ and $(\mathcal{D}_1^\circ)_{h(\mathcal{D}_2)}$ is obtained from $\mathcal{D}_1^\circ$ by adding $h(\mathcal{D}_2)$ to the environment in every sequent. Similarly for $(\mathcal{D}_2^\circ)_{h(\mathcal{D}_1)}$.

2. A derivation $\mathcal{D} \triangleright_{\mathsf{ND}_{\mathsf{seq}}} \Gamma \vdash A$ maps to a derivation $\mathcal{D}^\bullet \triangleright_{\mathsf{ND}} \Gamma \vdash A$ defined by:

$$\left(\dfrac{}{\Gamma, x : A \vdash A}\,\mathsf{ax}^x\right)^\bullet = A^x \qquad \left(\begin{array}{c}\vdots\,\mathcal{D}\\ \dfrac{\Gamma, x : A \vdash B}{\Gamma \vdash A \Rightarrow B}\Rightarrow_i^x\end{array}\right)^\bullet = \begin{array}{c}[A^x]\\ \vdots\,\mathcal{D}^\bullet\\ \dfrac{B}{A \Rightarrow B}\Rightarrow_i^x\end{array} \qquad \left(\dfrac{\begin{array}{cc}\vdots\,\mathcal{D}_1 & \vdots\,\mathcal{D}_2\\ \Gamma \vdash A \Rightarrow B & \Gamma \vdash A\end{array}}{B}\Rightarrow_e\right)^\bullet = \dfrac{\begin{array}{cc}\vdots\,\mathcal{D}_1^\bullet & \vdots\,\mathcal{D}_2^\bullet\\ A \Rightarrow B & A\end{array}}{B}\Rightarrow_e$$

Rmk. The translation $(\cdot)^\bullet$ is not injective: $\left(\dfrac{}{y : B, x : A \vdash A}\,\mathsf{ax}^x\right)^\bullet = A^x = \left(\dfrac{}{x : A \vdash A}\,\mathsf{ax}^x\right)^\bullet$.

Proposition

$(\mathcal{D}^\circ)^\bullet = \mathcal{D}$ for every derivation $\mathcal{D}$ in ND. There is $\mathcal{D}$ in $\mathsf{ND}_{\mathsf{seq}}$ such that $(\mathcal{D}^\bullet)^\circ \neq \mathcal{D}$.

Proof. The first part is by induciton on $\mathcal{D}$. The second part follows from the remark. □

Outline

1. Natural deduction for minimal logic, revisited
2. The Curry-Howard correspondence, more formally
3. Strong normalization of the simply typed λ-calculus
4. Logic and/vs Computation
5. Conclusion, exercises and bibliography

The Curry-Howard correspondence in a nutshell, a recall from Lecture 2

minimal logic	simply typed λ-calculus	computer science
formula	type	specification
derivation	term	program
cut-elimination step	β-reduction	computation step
derivation without redexes	normal form	result
cut-elimination theorem	normalization	termination

Concerning the correspondence between derivations and terms:

derivation in minimal logic	term in simply typed λ-calculus
hypotheses	variable
$\Rightarrow_i$	abstraction λ
$\Rightarrow_e$	application @

Goal. We now want to see this correspondence more precisely mathematically.

The formalisms seen so far

minimal logic	STLC
ND	Curry-style
trees with formulas	λ-terms with internal type annotations
$\mathsf{ND}_{\mathsf{seq}}$	Church-style
trees with sequents	λ-terms without internal type annotations

The Curry–Howard correspondence between minimal logic and STLC can be expressed by

- relating ND and $\mathsf{STLC}_{\mathsf{Curry}}$;
- relating ND and $\mathsf{STLC}_{\mathsf{Church}}$;
- relating $\mathsf{ND}_{\mathsf{seq}}$ and $\mathsf{STLC}_{\mathsf{Curry}}$;
- relating $\mathsf{ND}_{\mathsf{seq}}$ and $\mathsf{STLC}_{\mathsf{Church}}$.

To express it in a precise, fine-grained and rigorous way, we choose $\mathsf{ND}_{\mathsf{seq}}$ — $\mathsf{STLC}_{\mathsf{Church}}$.

Mapping derivations in $\mathsf{ND}_{\mathsf{seq}}$ to λ-terms in Church-style

The inference rules for STLC in Church-style are the ones of $\mathsf{ND}_{\mathsf{seq}}$ plus *decoration*.
$\rightsquigarrow$ We can decorate each sequent in a derivation of $\mathsf{ND}_{\mathsf{seq}}$ with a λ-term in Church-style.

Def. A derivation $\mathcal{D}$ in $\mathsf{ND}_{\mathsf{seq}}$ maps to a unique λ-term $(\mathcal{D})_\lambda$ in Church-style defined by

$$\left(\dfrac{}{\Gamma, x:A \vdash A}\,\mathsf{ax}^x\right)_\lambda = x \qquad \left(\dfrac{\begin{array}{c}\vdots\,\mathcal{D}\\ \Gamma, x:A \vdash B\end{array}}{\Gamma \vdash A \Rightarrow B}\Rightarrow_i^x\right)_\lambda = \lambda x^A.(\mathcal{D})_\lambda \qquad \left(\dfrac{\begin{array}{cc}\vdots\,\mathcal{D}_0 & \vdots\,\mathcal{D}_1\\ \Gamma \vdash A \Rightarrow B & \Gamma \vdash A\end{array}}{\Gamma \vdash B}\Rightarrow_e\right)_\lambda = (\mathcal{D}_0)_\lambda (\mathcal{D}_1)_\lambda$$

Rmk. The variable labeling the rules ax and $\Rightarrow_i$ is crucial to uniquely determine $(\mathcal{D})_\lambda$.
Without labeling, it is unclear which λ-term is associated with a derivation $\mathcal{D}$ in $\mathsf{ND}_{\mathsf{seq}}$:

$$\left(\dfrac{}{A, A \vdash A}\,\mathsf{ax}\right)_\lambda = \begin{cases} x\ ? \\ y\ ? \end{cases} \qquad \left(\dfrac{\begin{array}{c}\vdots\,\mathcal{D}\\ A, A \vdash B\end{array}}{\Gamma \vdash A \Rightarrow B}\Rightarrow_i\right)_\lambda = \begin{cases} \lambda x^A.(\mathcal{D})_\lambda\ ? \\ \lambda y^A.(\mathcal{D})_\lambda\ ? \end{cases}$$

and $\dfrac{\dfrac{\dfrac{}{A, A \vdash A}\,\mathsf{ax}}{A \vdash A \Rightarrow A}\Rightarrow_i}{\vdash A \Rightarrow A \Rightarrow A}\Rightarrow_i = \dfrac{\dfrac{\dfrac{}{A, A \vdash A}\,\mathsf{ax}}{A \vdash A \Rightarrow A}\Rightarrow_i}{\vdash A \Rightarrow A \Rightarrow A}\Rightarrow_i$ but $\mathcal{D}_1 = \dfrac{\dfrac{\dfrac{}{y:A,\ x:A \vdash A}\,\mathsf{ax}^x}{x:A \vdash A \Rightarrow A}\Rightarrow_i^y}{\vdash A \Rightarrow A \Rightarrow A}\Rightarrow_i^x \neq \dfrac{\dfrac{\dfrac{}{y:A,\ x:A \vdash A}\,\mathsf{ax}^x}{y:A \vdash A \Rightarrow A}\Rightarrow_i^x}{\vdash A \Rightarrow A \Rightarrow A}\Rightarrow_i^y = \mathcal{D}_2$

and their computational behavior is different: $(\mathcal{D}_1)_\lambda = \lambda x^A.\lambda y^A.x$ and $(\mathcal{D}_2)_\lambda = \lambda y^A.\lambda x^A.x$

$$(\mathcal{D}_1)_\lambda t_1 t_2 = (\lambda x^A.\lambda y^A.x) t_1 t_2 \to_\beta^* t_1 \qquad (\mathcal{D}_2)_\lambda t_1 t_2 = (\lambda y^A.\lambda x^A.x) t_1 t_2 \to_\beta^* t_2.$$

The Curry–Howard correspondence: statics

Proposition

If $\mathcal{D} \triangleright_{\mathsf{ND}_{\mathsf{seq}}} \Gamma \vdash A$ then $\Gamma \vdash_{\mathsf{Church}} (\mathcal{D})_\lambda : A$.

Proof. Straightforward induction on $\mathcal{D}$. □

Proposition (Uniqueness of type and derivation in $\mathsf{STLC}_{\mathsf{Church}}$, recall from Lecture 2)

If $\mathcal{D} \triangleright_{\mathsf{Church}} \Gamma \vdash t : A$ and $\mathcal{D}' \triangleright_{\mathsf{Church}} \Gamma \vdash t : A'$, then $A = A'$ and $\mathcal{D} = \mathcal{D}'$.

Theorem (Bijection between $\mathsf{ND}_{\mathsf{seq}}$ and $\mathsf{STLC}_{\mathsf{Church}}$)

For every environment Γ and type A, the map $(\cdot)_\lambda$ defines a bijection from derivations of $\Gamma \vdash A$ in $\mathsf{ND}_{\mathsf{seq}}$ and λ-terms t in Church-style such that $\Gamma \vdash_{\mathsf{Church}} t : A$.

Proof. Use the two propositions above. □

The Curry–Howard correspondence: dynamics

The bijection lifts to a *dynamic* correspondence! $\rightsquigarrow$ As β-reduction and cut-elimination mimic each other, it is an isomorphism between STLC in Church-style and $\mathsf{ND}_{\mathsf{seq}}$.

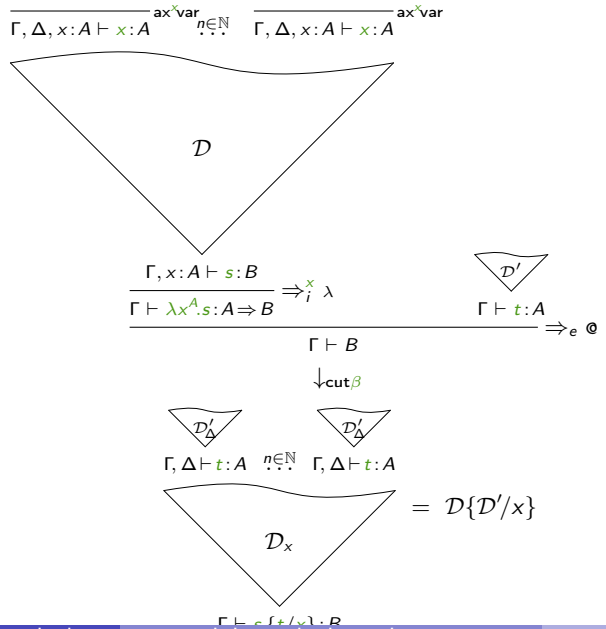
Theorem (Curry–Howard correspondence between $\mathsf{ND}_{\mathsf{seq}}$ and $\mathsf{STLC}_{\mathsf{Church}}$)

1. *Let $\mathcal{D}, \mathcal{D}'$ be derivations of $\Gamma \vdash A$ in $\mathsf{ND}_{\mathsf{seq}}$. If $\mathcal{D} \to_{\mathsf{cut}} \mathcal{D}'$ then $(\mathcal{D})_\lambda \to_\beta (\mathcal{D}')_\lambda$.*
2. *Let $\Gamma \vdash t : A$ and $\Gamma \vdash t' : A$ be derivable in STLC. If $t \to_\beta t'$ then $(t)_\lambda^{-1} \to_{\mathsf{cut}} (t')_\lambda^{-1}$.*

$$\begin{array}{lccc} \mathsf{ND}_{\mathsf{seq}} & \mathcal{D} & \xrightarrow{\quad\mathsf{cut}\quad} & \mathcal{D}' \\ & \Big\downarrow \mathsf{decoration} & (1) & \Big\downarrow \mathsf{decoration} \\ \mathsf{STLC} & (\mathcal{D})_\lambda & \xrightarrow{\quad\beta\quad} & (\mathcal{D}')_\lambda \end{array} \qquad \begin{array}{lccc} & (t)_\lambda^{-1} & \xrightarrow{\quad\mathsf{cut}\quad} & (t')_\lambda^{-1} \\ & \Big\uparrow \mathsf{forget} & (2) & \Big\uparrow \mathsf{forget} \\ & t & \xrightarrow{\quad\beta\quad} & t' \end{array}$$

Proof. 1) By induction on $\mathcal{D}$. 2) By induction on t. □

The $\mathrm{ND}_{\mathrm{seq}}$ cut-elimination/$\mathrm{STLC}_{\mathrm{Church}}$ β-reduction step in a picture



An example of $\mathrm{ND}_{\mathrm{seq}}$ cut-elimination/$\mathrm{STLC}_{\mathrm{Church}}$ β-reduction step

Let $\Delta = z : A$, $y : A \Rightarrow B$ and $C = A \Rightarrow B \Rightarrow A$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Delta, x:C \vdash x : C}\,\mathrm{ax}^x \quad \dfrac{}{\Delta, x:C \vdash z : A}\,\mathrm{ax}^z}{\Delta, x:C \vdash xz : B \Rightarrow A}\Rightarrow_e \quad \dfrac{}{\Delta, x:C \vdash y : A \Rightarrow B}\,\mathrm{ax}^y \quad \dfrac{}{\Delta, x:C \vdash z : A}\,\mathrm{ax}^z}{\Delta, x:C \vdash yz : B}\Rightarrow_e}{\Delta, x : A \Rightarrow B \Rightarrow A \vdash xz(yz) : A}\Rightarrow_e}{y : A \Rightarrow B, x : A \Rightarrow B \Rightarrow A \vdash \lambda z^A.xz(yz) : A \Rightarrow A}\Rightarrow_i^z}{x : A \Rightarrow B \Rightarrow A \vdash \lambda y^{A \Rightarrow B}.\lambda z^A.xz(yz) : (A \Rightarrow B) \Rightarrow A \Rightarrow A}\Rightarrow_i^y}{\vdash \lambda x^C.\lambda y^{A \Rightarrow B}.\lambda z^A.xz(yz) : (A \Rightarrow B \Rightarrow A) \Rightarrow (A \Rightarrow B) \Rightarrow A \Rightarrow A}\Rightarrow_i^x \quad \dfrac{\dfrac{\dfrac{}{a : A, b : B \vdash a : A}\,\mathrm{ax}^a}{a : A \vdash \lambda b^B.a : B \Rightarrow A}\Rightarrow_i^b}{\vdash \lambda a^A.\lambda b^B.a : A \Rightarrow B \Rightarrow A}\Rightarrow_i^a}{\vdash (\lambda x^C.\lambda y^{A \Rightarrow B}.\lambda z^A.xz(yz))\lambda a^A.\lambda b^B.a : (A \Rightarrow B) \Rightarrow A \Rightarrow A}\Rightarrow_e$$

$$\downarrow \mathrm{cut}\beta$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Delta, a : A, b : B \vdash a : A}\,\mathrm{ax}^a}{\Delta, a : A \vdash \lambda b^B.a : B \Rightarrow A}\Rightarrow_i^b}{\Delta \vdash \lambda a^A.\lambda b^B.a : A \Rightarrow B \Rightarrow A}\Rightarrow_i^a \quad \dfrac{}{\Delta \vdash z : A}\,\mathrm{ax}^z}{\Delta \vdash (\lambda a^A.\lambda b^B.a)z : B \Rightarrow A}\Rightarrow_e \quad \dfrac{\dfrac{}{\Delta \vdash y : A \Rightarrow B}\,\mathrm{ax}^y \quad \dfrac{}{\Delta \vdash z : A}\,\mathrm{ax}^z}{\Delta \vdash yz : B}\Rightarrow_e}{\Delta \vdash (\lambda a^A.\lambda b^B.a)z(yz) : A}\Rightarrow_e}{y : A \Rightarrow B \vdash \lambda z^A.(\lambda a^A.\lambda b^B.a)z(yz) : A \Rightarrow A}\Rightarrow_i^z}{\vdash \lambda y^{A \Rightarrow B}.\lambda z^A.(\lambda a^A.\lambda b^B.a)z(yz) : (A \Rightarrow B) \Rightarrow A \Rightarrow A}\Rightarrow_i^y$$

Indeed, $(\lambda x^C.\lambda y^{A \Rightarrow B}.\lambda z^A.xz(yz))\lambda a^A.\lambda b^B.a \to_\beta \lambda y^{A \Rightarrow B}.\lambda z^A.(\lambda a^A.\lambda b^B.a)z(yz)$

Computational understanding of logic / logical understanding of computation

Rmk. We presented the Curry–Howard correspondence as two distinct things:

- $\mathrm{STLC}_{\mathrm{Church}}$ as a programming language, and
- $\mathrm{ND}_{\mathrm{seq}}$ as a proof system for minimal logic,

which turn out to be isomorphic.

But they can be seen as two different views of the same thing $\rightsquigarrow$ a single underlying logical/computational system for reasoning about abstraction and hypotheticals:

- A proof of $A \Rightarrow B$ says "If I had a proof of A, I could prove B (without assuming A)".
- A program $: A \Rightarrow B$ says "If I had a value $: A$, I could compute a value $: B$".

That the underlying system are formalized as $\mathrm{ND}_{\mathrm{seq}}$ or $\mathrm{STLC}_{\mathrm{Church}}$ is just syntactic sugar.

Outline

1. Natural deduction for minimal logic, revisited
2. The Curry-Howard correspondence, more formally
3. Strong normalization of the simply typed λ-calculus
4. Logic and/vs Computation
5. Conclusion, exercises and bibliography

Abstract rewriting systems: normalization versus strong normalization

We have seen different sets (of derivations, λ-terms) and reductions (cut elimination, β).
$\rightsquigarrow$ Let us consider them abstractly, to study their common properties uniformly.

Def. An abstract rewriting system (ARS) is a set A and a relation $\to \subseteq A \times A$ (reduction); the reflexive-transitive closure of $\to$ is $\to^*$, i.e. $t \to^* s$ means $t \underbrace{\to \cdots \to}_{n \in \mathbb{N} \text{ times } \to} s$ ($t = s$ if $n = 0$).

- $t \in A$ is normal if there is no $s \in A$ such that $t \to s$.
- $t \in A$ is normalizing (or has a normal form) if $t \to^* u$ for some normal $u \in A$.
- $t \in A$ is strongly normalizing if there is no infinite sequence $(t_i)_{i \in \mathbb{N}}$ with $t_0 = t$ and $t_i \to t_{i+1}$ for all $i \in \mathbb{N}$, i.e. every reduction sequence eventually reaches a normal form.
- $\to$ is normalizing/strongly normalizing if so is every $t \in A$.
- $\to$ is confluent (resp. weakly confluent) if for all $t, r_1, r_2 \in A$ with $r_1 {}^*\!\leftarrow t \to^* r_2$ (resp. $r_1 \leftarrow t \to r_2$), there is $s \in A$ such that $r_1 \to^* s {}^*\!\leftarrow r_2$.

Rmk. Strong normalization implies normalization but the converse fails.
Rmk. Confluence implies weak confluence but the converse fails.

Proposition (Uniqueness of normal form)

If $\to$ is confluent, then for all $t \in A$ there is at most one normal $s \in A$ with $t \to^* s$.

Proof. If $r {}^*\!\leftarrow t \to^* s$ with r, s normal, by confluence $\exists\, u \in A$: $r \to^* u {}^*\!\leftarrow s$, so $r = u = s$. □

How to prove strong normalization: the combinatorial approach

Given a set A and a reduction $\to$ on A, we want to prove that $\to$ is strongly normalizing:
$\rightsquigarrow$ there is no (infinite) sequence $(t_i)_{i \in \mathbb{N}}$ such that $t_i \to t_{i+1}$ for all $i \in \mathbb{N}$.

Idea (combinatorial): For every $t \in A$, we define a measure $|t| \in S$ for some well-founded set $(S, <)$ — for instance $(\mathbb{N}, <)$ — such that: for *every* $s \in A$, if $t \to s$ then $|t| > |s|$.

Problem: It is doable for the simply typed λ-calculus, but it is very tricky.
$\rightsquigarrow$ After a single β-step the size ($\approx$ number of characters) of a term may not decrease.

$$(\lambda f^{X \Rightarrow X}.f(f(fx))\,(z(z(z(zf))))) \quad \to_\beta \quad (z(z(z(zf))))\Big((z(z(z(zf))))\Big((z(z(z(zf))))x\Big)\Big)$$

$\rightsquigarrow$ The measure should be defined independently of (or cannot rely on) the size of terms.

Rmk: The normalization theorem seen in Lecture 2 does not prove strong normalization.
$\rightsquigarrow$ The proof fires a specific redex (uppermost/innermost), otherwise the argument fails.

Reducibility candidates: a non-combinatorial approach

Idea: We define a set Red_A of terms (reducibility candidates) by induction on the type A:

- for any ground type X, Red_X is the set of strongly normalizing (SN) terms of type X;
- $\mathrm{Red}_{A \Rightarrow B}$ is the set of the terms s of type $A \Rightarrow B$ such that $st \in \mathrm{Red}_B$ for all $t \in \mathrm{Red}_A$.

Rmk: For every type A, every term in Red_A is SN (easy proof by induction on A).

Goal: For any type A, if $u : A$ then $u \in \mathrm{Red}_A$ (so u is SN). Proof by induction on u. Cases:

1. If $u = st : A$ then $s : B \Rightarrow A$ and $t : B$; by IH, $s \in \mathrm{Red}_{B \Rightarrow A}$ and $t \in \mathrm{Red}_B$, so $u \in \mathrm{Red}_A$.
2. If $u = x : X$, then u is SN, so $u \in \mathrm{Red}_X$. If $u = x : B \Rightarrow C$, to prove that $x \in \mathrm{Red}_{B \Rightarrow C}$ we have to show that $xt \in \mathrm{Red}_C$ for all $t \in \mathrm{Red}_B$ $\rightsquigarrow$ A stronger hypothesis is needed.
3. If $u = \lambda x^B.s : B \Rightarrow C$, to prove that $u \in \mathrm{Red}_{B \Rightarrow C}$ we have to show that $(\lambda x^B.s)t \in \mathrm{Red}_C$ for all $t \in \mathrm{Red}_B$. $\rightsquigarrow$ How to prove that?

Idea: Suppose $\lambda x^B.s : B \Rightarrow C$ and $t \in \mathrm{Red}_B$. Let us prove that $s\{t/x\} \in \mathrm{Red}_C$ and that if $s\{t/x\} \in \mathrm{Red}_C$ then $(\lambda x^B.s)t \in \mathrm{Red}_C$. This way, Point 3 above is done.

Problem. The environments for $\lambda x^B.s$ and t may be differ in some free variable.
$\rightsquigarrow$ The application of $\lambda x^B.s$ to t may not be possible, e.g. $\lambda x^B.f(fx)$ and $t = f$.

Fixing the reducibility candidate method

Solution: Let us take the environment into account when defining Red_A, for all types A.

$$\mathrm{Red}_X = \{\langle \Gamma; t\rangle \mid t \text{ is SN},\ \Gamma \vdash_{\mathsf{STLC}} t : X\}$$
$$\mathrm{Red}_{A \Rightarrow B} = \{\langle \Gamma; s\rangle \mid \Gamma \vdash_{\mathsf{STLC}} s : A \Rightarrow B,\ \langle \Gamma, \Delta;\ st\rangle \in \mathrm{Red}_B \text{ for all } \langle \Gamma, \Delta;\ t\rangle \in \mathrm{Red}_A\}$$

Lemma (Saturation)

1. If $\langle \Gamma; t\rangle \in \mathrm{Red}_B$ then t is SN.
2. If $\Gamma \vdash_{\mathsf{STLC}} xt_1 \dots t_n : B$ and $t_1, \dots, t_n$ are SN, then $\langle \Gamma; xt_1 \dots t_n\rangle \in \mathrm{Red}_B$.
3. If $\langle \Gamma; s\{t/x\}t_1 \dots t_n\rangle \in \mathrm{Red}_B$, $\Gamma \vdash_{\mathsf{STLC}} t : A$ and t is SN then $\langle \Gamma; (\lambda x^A.s)tt_1 \dots t_n\rangle \in \mathrm{Red}_B$.

Proof. Points 1–3 are proved simultaneously by induction on the type B. If $B = X$ then Point 1 is by definition of Red_X, for Points 2–3 see Exercise 10 on p. 36. Let $B = C \Rightarrow D$.

1. Let $z \notin \mathrm{dom}(\Gamma)$, so $\langle \Gamma, z : C;\ z\rangle \in \mathrm{Red}_C$ by the induction hypothesis of Point 2 applied to C. As $\langle \Gamma; t\rangle \in \mathrm{Red}_{C \Rightarrow D}$, then $\langle \Gamma, z : C;\ tz\rangle \in \mathrm{Red}_D$ and hence tz is SN by the induction hypothesis of Point 1 applied to D; thus t is SN too (why?).
2. Let $\langle \Gamma, \Delta;\ t\rangle \in \mathrm{Red}_C$, so t is SN by induction hypothesis of Point 1 applied to C; as $\Gamma, \Delta \vdash_{\mathsf{STLC}} xt_1 \dots t_n t : D$ (why?), $\langle \Gamma, \Delta; xt_1 \dots t_n t\rangle \in \mathrm{Red}_D$ by induction hypothesis of Point 2 applied to D; hence, $\langle \Gamma; xt_1 \dots t_n\rangle \in \mathrm{Red}_B$ by definition of $\mathrm{Red}_{C \Rightarrow D}$.
3. Let $\langle \Gamma, \Delta;\ r\rangle \in \mathrm{Red}_C$, so $\langle \Gamma, \Delta; s\{t/x\}t_1 \dots t_n r\rangle \in \mathrm{Red}_D$ and hence, by the induction hypothesis, $\langle \Gamma, \Delta; (\lambda x^A.s)tt_1 \dots t_n r\rangle \in \mathrm{Red}_D$; thus, $\langle \Gamma; (\lambda x^A.s)tt_1 \dots t_n\rangle \in \mathrm{Red}_B$. □

Strong normalization proved via reducibility candidates

Rmk. In the previous lemma, Point 1 needs Point 2 in its proof, and vice versa.
Point 3 is independent of Points 1–2 and is used in the proof of the lemma below.

Lemma (Substitution)

If $x_1 : B_1, \dots, x_n : B_n \vdash_{\mathsf{STLC}} t : A$ and $\langle \Gamma; s_i \rangle \in \mathrm{Red}_{B_i}$, then $\langle \Gamma; t\{s_1/x_1, \dots, s_n/x_n\} \rangle \in \mathrm{Red}_A$.

Proof. By structural induction on the term t, using Point 3 above (exercise!). □

Theorem (Strong normalization of the simply typed λ-calculus)

Every typable term in STLC (in Curry-style or Church-style) is SN.

Proof. Let $x_1 : B_1, \dots, x_n : B_n \vdash_{\mathsf{STLC}} t : A$. Let $\Gamma = x_1 : B_1, \dots, x_n : B_n$ and $s_i = x_i$ for all $1 \leq i \leq n$, hence $\langle \Gamma; s_i \rangle \in \mathrm{Red}_{B_i}$ by Point 2 of the saturation lemma on p. 28, for all $1 \leq i \leq n$. By the substitution lemma above, $\langle \Gamma; t \rangle = \langle \Gamma; t\{s_1/x_1, \dots, s_n/x_n\} \rangle \in \mathrm{Red}_A$. By Point 1 of the saturation lemma on p. 28, t is SN. □

An application of strong normalization

Lemma (Newman)

If an ARS $(A, \to)$ is SN and weakly confluent then it is confluent.

Proof. As $\to$ is SN, for all $x \in A$ there is $\mathrm{sn}(x) \in \mathbb{N}$ equal to the maximal number of $\to$-steps from x to a normal form. Let $s_1 {}^*\!\leftarrow t \to^* s_2$. We have to show that $s_1 \to^* r {}^*\!\leftarrow s_2$ for some r. We proceed by induction on $\mathrm{sn}(t)$. If $t = s_1$, we are done by taking $r = s_2$. Similarly if $t = s_2$. Otherwise, $s_1 {}^*\!\leftarrow t_1 \leftarrow t \to t_2 \to^* s_2$. By weak confluence, $t_1 \to^* s {}^*\!\leftarrow t_2$. By maximality, $\mathrm{sn}(t_1) < \mathrm{sn}(t)$, $\mathrm{sn}(t_2) < \mathrm{sn}(t)$ and $\mathrm{sn}(s) < \mathrm{sn}(t)$. By induction hypothesis applied to $s_1 {}^*\!\leftarrow t_1 \to^* s$ and $s {}^*\!\leftarrow t_2 \to^* s_2$, there are r_1, r_2 such that $s_1 \to^* r_1 {}^*\!\leftarrow s$ and $s \to^* r_2 {}^*\!\leftarrow s_2$. By induction hypothesis applied to $r_1 {}^*\!\leftarrow s \to^* r_2$, $s_1 \to^* r_1 \to^* r {}^*\!\leftarrow r_2 {}^*\!\leftarrow s_2$ for some r. □

Lemma

$(\mathsf{STLC}, \to_\beta)$ is weakly confluent.

Proof. It is a good exercise, by inspection of all possible cases where $s_1 \leftarrow t \to s_2$. □

Corollary

$(\mathsf{STLC}, \to_\beta)$ is confluent.

Proof. By Newman's lemma, and weak confluence and strong normalization of STLC. □

Moral. The order in which β-redexes are fired in a typable λ-term of STLC does not matter, it will eventually lead to a normal form by SN, and the same result by confluence.

Outline

1. Natural deduction for minimal logic, revisited
2. The Curry-Howard correspondence, more formally
3. Strong normalization of the simply typed λ-calculus
4. Logic and/vs Computation
5. Conclusion, exercises and bibliography

The expressiveness of the simply typed λ-calculus

The computational power of STLC is quite limited, far from Turing-completeness.

Theorem (Schwichtenberg)

The functions that are definable in STLC *are exactly the extended polynomials, that is, the smallest class of functions $f : \mathbb{N}^k \to \mathbb{N}$ for all $k \in \mathbb{N}$ containing the:*

- *projections $\pi_i^k(n_1, \dots, n_k) = n_i$ for all $1 \leq i \leq n$;*
- *constants $k(n) = n$ and signum $sg(0) = 0$ and $sg(n+1) = 1$;*

and closed under addition and multiplication.

A way to increase the computational power while keeping types is to enrich STLC with

- ground types `nat` for natural numbers and `bool` for Booleans, with their constants `true` : `bool`, `false` : `bool` and $\underline{n}$: `nat` (for all $n \in \mathbb{N}$) as axioms;
- some function symbols for basic functions such as predecessor, if-then-else and so on, with their appropriate types as axioms, and their reduction rules;
- for every type A, a fixpoint combinator $\mathtt{Y}_A$ with the type $(A \Rightarrow A) \Rightarrow A$ as an axiom and the reduction rule $\mathtt{Y}_A t \to_\beta t(\mathtt{Y}_A t)$.

$\rightsquigarrow$ PCF, a Turing-complete prototype of functional programming languages, but its logical meaning is gone: any type is inhabited ($\vdash \mathtt{Y}_A\, \lambda x^A.x : A$ is derivable for any A).

Curry–Howard beyond minimal logic

The Curry–Howard correspondence is not only for minimal logic, it can be extended to:

- full propositional intuitionistic logic, by adding conjunction (i.e. product types) with pairs/projections, disjunction (i.e. sum types) with injections/cases, ...;
- second order intuitionistic logic by adding a universal quantifier for polymorphism;
- some variants of classical logic and other logics;
- dependent type theory;
- ...

In such extensions, the computational power increases, keeping a logical meaning. E.g.

Theorem (Girard)

The functions that are definable in system F (second order intuitionistic logic) are the ones that can be proved to be total by second-order Peano arithmetic.

But these extensions cannot be Turing-complete: by cut-elimination/normalization they cannot represent partial functions.

There is an inherent trade-off between computational power and logical meaning.

Outline

1. Natural deduction for minimal logic, revisited
2. The Curry-Howard correspondence, more formally
3. Strong normalization of the simply typed λ-calculus
4. Logic and/vs Computation
5. Conclusion, exercises and bibliography

What have we learned today?

1. A refined way to present natural deduction in minimal logic.
2. The Curry–Howard isomorphism between natural deduction for minimal logic and the simply typed λ-calculus, in a very precise way.
3. The proof of strong normalization of STLC via reducibility candidates.
4. The limited expressiveness of STLC.

Questions?



Non-assessed exercises (assessed exercises are available on: click here)

1. For each the λ-terms $\lambda x^X.\lambda y^Y.x$ and $\lambda x^{Z\Rightarrow Y\Rightarrow X}.\lambda y^{Z\Rightarrow Y}.\lambda z^Z.xz(yz)$, write the unique derivation in $\mathrm{STLC}_{\mathrm{Church}}$ typing it with some type and the empty environment.
2. Define five ARSs $(A,\to)$: in the first $\to$ is normalizing but not strongly normalizing, in the second $\to$ is not normalizing, in the third $\to$ is strongly normalizing, in the fourth $\to$ is not confluent but every $t\in A$ has at most one normal form, in the fifth $\to$ is weakly confluent but not confluent (*hint:* it is not SN, by Newman's lemma).
3. In a ARS $(A,\to)$, prove that $t\in A$ is SN iff for every $t'\in A$, if $t\to t'$ then t' is SN.
4. Prove that (in Curry-style) if $t\to_\beta t'$ then $t\{s/x\}\to_\beta t'\{s/x\}$ and $s\{t/x\}\to_\beta^* s\{t'/x\}$.
5. Prove that if a λ-term t (in Curry-style) is SN then every subterm of t is SN.
6. Prove the following facts for λ-terms (in Curry-style):
   1. for all $n\in\mathbb{N}$, if $t_1,\dots,t_n$ are SN, then so is $xt_1\dots t_n$;
   2. if s is SN, then so is $\lambda x.s$;
   3. for all $n\in\mathbb{N}$, if t and $s\{t/x\}t_1\dots t_n$ are SN, then so is $(\lambda x.s)tt_1\dots t_n$;
   4. for all $n\in\mathbb{N}$, if $x\in\mathrm{fv}(s)$ and $s\{t/x\}t_1\dots t_n$ is SN then so is $(\lambda x.s)tt_1\dots t_n$.
7. Prove that if t can be built up from the rules below applied $n\in\mathbb{N}$ times, then t is SN and the maximal length $\mathrm{sn}(t)$ of the β-reduction sequences from t is $\leq n$.
$$\frac{(t_i)_{1\leq i\leq k}}{xt_1\dots t_k}\,@_k \qquad \frac{t}{\lambda x.t}\,\lambda \qquad \frac{u\quad s\{u/x\}t_1\dots t_k}{(\lambda x.s)ut_1\dots t_k}\,\beta_k \qquad \text{(for any } k\in\mathbb{N} \text{ in } @_k,\ \beta_k)$$
8. Prove that the SN terms (in Curry-style) are exactly those built up from $@_k,\lambda,\beta_k$.
9. Prove that if $\langle\Gamma;t\rangle\in\mathrm{Red}_B$, then $\langle\Gamma,x:A;\,t\rangle\in\mathrm{Red}_B$, by induction on the type B.
10. Prove rigorously the saturation lemma on p. 28 and the substitution lemma on p. 29.

Bibliography

- For more about the simply typed λ-calculus:

  
  Ralph Loader. *Notes on Simply Typed λ-Calculus*. Technical report ECS-LFCS-98-381, University of Edinburgh, 1998. `http://www.lfcs.inf.ed.ac.uk/reports/98/ECS-LFCS-98-381/ECS-LFCS-98-381.pdf`. [Chapters 1–3]

  
  Henk Barendregt. *Lambda Calculi with Types*. In Handbook of Logic in Computer Science; vol. 2, 117-309, 1992. `https://repository.ubn.ru.nl/bitstream/handle/2066/17231/17231.pdf`. [Chapter 3]

- For more about the Curry-Howard correspondence:

  
  Jean-Yves Girard, Yves Lafont, Paul Taylor. *Proofs and Types*. Cambridge Tracts in Theoretical Computer Science, Vol. 7, Cambridge University Press, 1989. `https://www.paultaylor.eu/stable/prot.pdf`. [Chapters 2–4, 6]

  
  Morten. H. Sørensen, Pawel Urzyczyin. *Lectures on the Curry-Howard Isomorphism*. In Studies in Logic and the Foundations of Mathematics, vol. 149, 2006. `https://disi.unitn.it/~bernardi/RSISE11/Papers/curry-howard.pdf`. [Chapters 2–4]

- For a combinatorial proof of normalization for the simply typed λ-calculus:

  
  Pablo Barenbaum, Cristian Sottile. Two Decreasing Measures for Simply Typed λ-Terms. FSCD 2023, LIPIcs, vol. 260, 11:1–11:19, 2023. `https://doi.org/10.4230/LIPIcs.FSCD.2023.11`.