The λ-Calculus, from Minimal to Classical Logic

Webpage of the course

Davide Barbarossa
db2437@bath.ac.uk
Dept of Computer Science

Giulio Guerrieri
g.guerrieri@sussex.ac.uk
Dept of Ìnformatics

ESSLLI Summer School, Bochum (Germany) *28/07/2025 – 01/08/2025*

Previously...

- What does denotational semantics is and is for.
- Some abstract properties that an algebraic structure has to fulfill to be a denotational semantics of the untyped λ-calculus.
- A taste of category theory.
- The notions of Cartesian closed category and reflexive object.
- How to interpret the untyped λ-calculus in a reflexive object of a Cartesian closed category.

The λ-Calculus, from Minimal to Classical Logic

Lecture 4:

Curry-Howard and Minimal Logic

Read the notes: they are full of details, proofs, explanations, exercises, bibliography!

Giulio Guerrieri
g.guerrieri@sussex.ac.uk
Dept of Informatics
UNIVERSITY OF SUSSEX

ESSLLI Summer School, Bochum (Germany) *31/07/2025*

Outline

The "philosophy" behind the untyped λ-calculus:

1. Everything is a function, including values such as Booleans and natural numbers.

$$\underline{true} = \lambda x.\lambda y.x = x \mapsto (y \mapsto x) \qquad \underline{2} = \lambda f.\lambda x.f(fx) = f \mapsto (x \mapsto f(f(x)))$$

The "philosophy" behind the untyped λ-calculus:

1. Everything is a function, including values such as Booleans and natural numbers.

$$\underline{true} = \lambda x.\lambda y.x = x \mapsto (y \mapsto x) \qquad \underline{2} = \lambda f.\lambda x.f(fx) = f \mapsto (x \mapsto f(f(x)))$$

2. Functions are treated anonymously, that is, without giving them a name.

$$\mathrm{id}(x) = x \quad \rightsquigarrow \quad x \mapsto x \qquad\qquad \mathrm{proj}_1^2(x, y) = x \quad \rightsquigarrow \quad (x, y) \mapsto x$$

The "philosophy" behind the untyped λ-calculus:

1. Everything is a function, including values such as Booleans and natural numbers.

$$\underline{\mathit{true}} = \lambda x.\lambda y.x = x \mapsto (y \mapsto x) \qquad \underline{2} = \lambda f.\lambda x.f(fx) = f \mapsto (x \mapsto f(f(x)))$$

2. Functions are treated anonymously, that is, without giving them a name.

$$\mathrm{id}(x) = x \quad \rightsquigarrow \quad x \mapsto x \qquad\qquad \mathrm{proj}_1^2(x, y) = x \quad \rightsquigarrow \quad (x, y) \mapsto x$$

3. Functions of several arguments are transformed into ones of a single argument:

$$(x, y) \mapsto x \qquad \rightsquigarrow \qquad x \mapsto (y \mapsto x) = \lambda x.\lambda y.x \qquad\qquad (\text{currying})$$

The "philosophy" behind the untyped λ-calculus:

1. Everything is a function, including values such as Booleans and natural numbers.

$$\underline{true} = \lambda x.\lambda y.x = x \mapsto (y \mapsto x) \qquad \underline{2} = \lambda f.\lambda x.f(fx) = f \mapsto (x \mapsto f(f(x)))$$

2. Functions are treated anonymously, that is, without giving them a name.

$$\mathrm{id}(x) = x \quad \rightsquigarrow \quad x \mapsto x \qquad \mathrm{proj}_1^2(x, y) = x \quad \rightsquigarrow \quad (x, y) \mapsto x$$

3. Functions of several arguments are transformed into ones of a single argument:

$$(x, y) \mapsto x \quad \rightsquigarrow \quad x \mapsto (y \mapsto x) = \lambda x.\lambda y.x \qquad \text{(currying)}$$

4. Functions can be applied to functions and can return functions (higher-order).

$$(x \mapsto x)5 = 5 \qquad (x \mapsto x)(y \mapsto y^2) = y \mapsto y^2$$

The "philosophy" behind the untyped λ-calculus:

1. Everything is a function, including values such as Booleans and natural numbers.

$$\underline{true} = \lambda x.\lambda y.x = x \mapsto (y \mapsto x) \qquad \underline{2} = \lambda f.\lambda x.f(fx) = f \mapsto (x \mapsto f(f(x)))$$

2. Functions are treated anonymously, that is, without giving them a name.

$$\mathrm{id}(x) = x \quad \rightsquigarrow \quad x \mapsto x \qquad\qquad \mathrm{proj}_1^2(x, y) = x \quad \rightsquigarrow \quad (x, y) \mapsto x$$

3. Functions of several arguments are transformed into ones of a single argument:

$$(x, y) \mapsto x \quad \rightsquigarrow \quad x \mapsto (y \mapsto x) = \lambda x.\lambda y.x \qquad \text{(currying)}$$

4. Functions can be applied to functions and can return functions (higher-order).

$$(x \mapsto x)5 = 5 \qquad\qquad (x \mapsto x)(y \mapsto y^2) = y \mapsto y^2$$

5. There are no restrictions when applying functions to other functions (untyped).

The "philosophy" behind the untyped λ-calculus:

1. Everything is a function, including values such as Booleans and natural numbers.

$$\underline{true} = \lambda x.\lambda y.x = x \mapsto (y \mapsto x) \qquad \underline{2} = \lambda f.\lambda x.f(fx) = f \mapsto (x \mapsto f(f(x)))$$

2. Functions are treated anonymously, that is, without giving them a name.

$$\mathrm{id}(x) = x \quad \rightsquigarrow \quad x \mapsto x \qquad \mathrm{proj}_1^2(x, y) = x \quad \rightsquigarrow \quad (x, y) \mapsto x$$

3. Functions of several arguments are transformed into ones of a single argument:

$$(x, y) \mapsto x \quad \rightsquigarrow \quad x \mapsto (y \mapsto x) = \lambda x.\lambda y.x \qquad \text{(currying)}$$

4. Functions can be applied to functions and can return functions (higher-order).

$$(x \mapsto x)5 = 5 \qquad (x \mapsto x)(y \mapsto y^2) = y \mapsto y^2$$

5. There are no restrictions when applying functions to other functions (untyped).

The untyped feature sounds suspicious, it looks too wild (see Curry's paradox).

Question: Can we drop it and keep all the other features?

People seem very unhappy about the *untyped* λ-calculus!

The Working Mathematician: Is the untyped λ-calculus a real theory of (computable) functions? Are you kidding me? In mathematics functions have domain and codomains, their arguments can't live outside their domain.

People seem very unhappy about the *untyped* λ-calculus!

The Working Mathematician: Is the untyped λ-calculus a real theory of (computable) functions? Are you kidding me? In mathematics functions have domain and codomains, their arguments can't live outside their domain.

The Working Computer Scientist: I can't take the untyped λ-calculus as a serious programming language! There are no types! I can't give any specifications to my programs. I can write nonsensical programs that return nonsensical outputs.

$$\underline{2}\,\underline{true} = (\lambda f.\lambda x.f(fx))(\lambda z.\lambda y.z) \twoheadrightarrow_\beta^* \lambda x.\lambda y.\lambda z.x = \mathrm{proj}_1^3$$

People seem very unhappy about the *untyped* λ-calculus!

The Working Mathematician: Is the untyped λ-calculus a real theory of (computable) functions? Are you kidding me? In mathematics functions have domain and codomains, their arguments can't live outside their domain.

The Working Computer Scientist: I can't take the untyped λ-calculus as a serious programming language! There are no types! I can't give any specifications to my programs. I can write nonsensical programs that return nonsensical outputs.

$$\underline{2}\,\underline{true} = (\lambda f.\lambda x.f(fx))(\lambda z.\lambda y.z) \twoheadrightarrow^*_\beta \lambda x.\lambda y.\lambda z.x = \mathrm{proj}^3_1$$

The Working Logician: I want my money back! You promised me a course about logic. After three days, I haven't seen any logic yet! You scammer!

People seem very unhappy about the *untyped* λ-calculus!

The Working Mathematician: Is the untyped λ-calculus a real theory of (computable) functions? Are you kidding me? In mathematics functions have domain and codomains, their arguments can't live outside their domain.

The Working Computer Scientist: I can't take the untyped λ-calculus as a serious programming language! There are no types! I can't give any specifications to my programs. I can write nonsensical programs that return nonsensical outputs.

$$\underline{2}\,\underline{true} = (\lambda f.\lambda x.f(fx))(\lambda z.\lambda y.z) \twoheadrightarrow_\beta^* \lambda x.\lambda y.\lambda z.x = \mathrm{proj}_1^3$$

The Working Logician: I want my money back! You promised me a course about logic. After three days, I haven't seen any logic yet! You scammer!

Let us try to make the working mathematician, computer scientist and logician happy.

Let $\mathtt{M} \in \Lambda$ with $\vec{x} = (x_1, \ldots, x_n)$ adequate for $\mathtt{M}$.

- In the untyped λ-calculus, the interpretation of $\mathtt{M}$ is $[\![\mathtt{M}]\!]_{\vec{x}} \colon U^n \to U$, given a reflexive object $(U, \lambda, \mathrm{fun})$ in a CCC (see Day 3) $\rightsquigarrow$ $\mathtt{M}$ can be seen as a function

$$
\begin{aligned}
[\![\mathtt{M}]\!]_{\vec{x}} \colon U \times \overset{n}{\cdots} \times U &\longrightarrow U \\
(a_1, \ldots, a_n) &\mapsto \mathtt{M}\{x_1 \coloneqq a_1, \ldots, x_n \coloneqq a_n\}
\end{aligned}
$$

- Because of the retraction $(\lambda \colon U \Rightarrow U \to U, \mathrm{fun} \colon U \to U \Rightarrow U)$, every function lives in U and can be applied to any other function (including itself!).

Let $\mathtt{M} \in \Lambda$ with $\vec{x} = (x_1, \dots, x_n)$ adequate for $\mathtt{M}$.

- In the untyped λ-calculus, the interpretation of $\mathtt{M}$ is $[\![\mathtt{M}]\!]_{\vec{x}} \colon U^n \to U$, given a reflexive object $(U, \lambda, \mathrm{fun})$ in a CCC (see Day 3) $\leadsto$ $\mathtt{M}$ can be seen as a function

$$\begin{aligned} [\![\mathtt{M}]\!]_{\vec{x}} \colon U \times \overset{n}{\cdots} \times U &\longrightarrow U \\ (a_1, \dots, a_n) &\mapsto \mathtt{M}\{x_1 \coloneqq a_1, \dots, x_n \coloneqq a_n\} \end{aligned}$$

- Because of the retraction $(\lambda \colon U \Rightarrow U \to U, \mathrm{fun} \colon U \to U \Rightarrow U)$, every function lives in U and can be applied to any other function (including itself!).

Let us restrict the domain of our functions $\leadsto$ We would like to see $\mathtt{M}$ as a function

$$\begin{aligned} [\![\mathtt{M}]\!]_{\vec{x}} \colon A_1 \times \cdots \times A_n &\longrightarrow B \\ (a_1, \dots, a_n) &\mapsto \mathtt{M}\{x_1 \coloneqq a_1, \dots, x_n \coloneqq a_n\} \end{aligned}$$

Let $\mathtt{M} \in \Lambda$ with $\vec{x} = (x_1, \ldots, x_n)$ adequate for $\mathtt{M}$.

- In the untyped λ-calculus, the interpretation of $\mathtt{M}$ is $[\![\mathtt{M}]\!]_{\vec{x}} \colon U^n \to U$, given a reflexive object $(U, \lambda, \mathrm{fun})$ in a CCC (see Day 3) $\leadsto$ $\mathtt{M}$ can be seen as a function

$$\begin{aligned} [\![\mathtt{M}]\!]_{\vec{x}} \colon U \times \overset{n}{\cdots} \times U &\longrightarrow U \\ (a_1, \ldots, a_n) &\mapsto \mathtt{M}\{x_1 := a_1, \ldots, x_n := a_n\} \end{aligned}$$

- Because of the retraction $(\lambda \colon U \Rightarrow U \to U, \mathrm{fun} \colon U \to U \Rightarrow U)$, every function lives in U and can be applied to any other function (including itself!).

Let us restrict the domain of our functions $\leadsto$ We would like to see $\mathtt{M}$ as a function

$$\begin{aligned} [\![\mathtt{M}]\!]_{\vec{x}} \colon A_1 \times \cdots \times A_n &\longrightarrow B \\ (a_1, \ldots, a_n) &\mapsto \mathtt{M}\{x_1 := a_1, \ldots, x_n := a_n\} \end{aligned}$$

1. What are the domains $A_1, \ldots, A_n$?
2. What is the codomain B? Should it depend on $\mathtt{M}$?
3. Which type discipline we should follow?
4. Does it restrict the λ-terms that can be built?

Let us introduce types to get these restrictions. What are some basic rules for typing? They should express the type of a term depending on the types of its *free variables*.

1. If $x_1 : A_1, \ldots, x_n : A_n$ then $x_i : A_i$ for every $i \in \{1, \ldots, n\}$.

Let us introduce types to get these restrictions. What are some basic rules for typing? They should express the type of a term depending on the types of its *free variables*.

1. If $x_1 : A_1, \ldots, x_n : A_n$ then $x_i : A_i$ for every $i \in \{1, \ldots, n\}$.
2. If $\mathtt{M} : B$ under the environment $x_1 : C_1, \ldots, x_n : C_n, y : A$,

   $$(\text{that is, } [\![\mathtt{M}]\!]_{\vec{x},y} : C_1 \times \cdots \times C_n \times A \longrightarrow B\ )$$

   then $\lambda x.\mathtt{M} : A \Rightarrow B$ under the environment $x_1 : C_1, \ldots, x_n : C_n$.

   $$(\text{that is, } [\![\lambda \mathtt{y}.\mathtt{M}]\!]_{\vec{x}} : C_1 \times \cdots \times C_n \longrightarrow A \Rightarrow B\ )$$

Let us introduce types to get these restrictions. What are some basic rules for typing? They should express the type of a term depending on the types of its *free variables*.

1. If $x_1 : A_1, \ldots, x_n : A_n$ then $x_i : A_i$ for every $i \in \{1, \ldots, n\}$.
2. If $\mathtt{M} : B$ under the environment $x_1 : C_1, \ldots, x_n : C_n, y : A$,

   $$(\text{that is, } [\![\mathtt{M}]\!]_{\vec{x},y} : C_1 \times \cdots \times C_n \times A \longrightarrow B\ )$$

   then $\lambda x.\mathtt{M} : A \Rightarrow B$ under the environment $x_1 : C_1, \ldots, x_n : C_n$.

   $$(\text{that is, } [\![\lambda \mathtt{y}.\mathtt{M}]\!]_{\vec{x}} : C_1 \times \cdots \times C_n \longrightarrow A \Rightarrow B\ )$$

3. If $\mathtt{M} : A \Rightarrow B$ and $\mathtt{N} : A$ under the common environment $x_1 : C_1, \ldots, x_n : C_n$,

   $$(\text{that is, } [\![\mathtt{M}]\!]_{\vec{x}} : C_1 \times \cdots \times C_n \longrightarrow A \Rightarrow B \ \text{ and } \ [\![\mathtt{N}]\!]_{\vec{x}} : C_1 \times \cdots \times C_n \longrightarrow A\ )$$

   then $\mathtt{M}\,\mathtt{N} : B$ under the same environment.

   $$(\text{that is, } [\![\mathtt{M}\,\mathtt{N}]\!]_{\vec{x}} : C_1 \times \cdots \times C_n \longrightarrow B\ )$$

Let us introduce types to get these restrictions. What are some basic rules for typing? They should express the type of a term depending on the types of its *free variables*.

1. If $x_1 : A_1, \ldots, x_n : A_n$ then $x_i : A_i$ for every $i \in \{1, \ldots, n\}$.
2. If $\mathtt{M} : B$ under the environment $x_1 : C_1, \ldots, x_n : C_n, y : A$,

$$(\text{that is, } [\![\mathtt{M}]\!]_{\vec{x},y} : C_1 \times \cdots \times C_n \times A \longrightarrow B\ )$$

   then $\lambda x.\mathtt{M} : A \Rightarrow B$ under the environment $x_1 : C_1, \ldots, x_n : C_n$.

$$(\text{that is, } [\![\lambda \mathtt{y}.\mathtt{M}]\!]_{\vec{x}} : C_1 \times \cdots \times C_n \longrightarrow A \Rightarrow B\ )$$

3. If $\mathtt{M} : A \Rightarrow B$ and $\mathtt{N} : A$ under the common environment $x_1 : C_1, \ldots, x_n : C_n$,

$$(\text{that is, } [\![\mathtt{M}]\!]_{\vec{x}} : C_1 \times \cdots \times C_n \longrightarrow A \Rightarrow B \text{ and } [\![\mathtt{N}]\!]_{\vec{x}} : C_1 \times \cdots \times C_n \longrightarrow A\ )$$

   then $\mathtt{M}\,\mathtt{N} : B$ under the same environment.

$$(\text{that is, } [\![\mathtt{M}\,\mathtt{N}]\!]_{\vec{x}} : C_1 \times \cdots \times C_n \longrightarrow B\ )$$

Rmk: This naive approach seems to make sense.

- Types only require a connective $\Rightarrow$.
- The rules above sound similar to inference rules.
- Rule 2 does something similar to curry$(\cdot)$ in a CCC.
- Rule 3 does something similar to $\mathrm{ev}_{A,B}$ in a CCC.

Let us introduce the simply typed λ-calculus in Church-style (STLC).

Types: $A, B ::= X \mid A \Rightarrow B$ given a set of *ground* types ranged over by $X, Y, Z \dots$

(λ-)Terms: $s, t ::= x \mid \lambda x^{A}.t \mid st$

From the Untyped to the Simply Typed λ-Calculus

Let us introduce the simply typed λ-calculus in Church-style (STLC).

Types: $A, B ::= X \mid A \Rightarrow B$ given a set of *ground* types ranged over by $X, Y, Z \dots$

(λ-)Terms: $s, t ::= x \mid \lambda x^A.t \mid st$

Environment: function from finitely many variables to types, noted $x_1 : A_1, \dots, x_n : A_n$.
The well-typed terms are the ones that can be constructed via the *typing rules* below.

$$\frac{}{\Gamma, x : A \vdash x : A}\,\mathsf{var} \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x^A.t : A \Rightarrow B}\,\lambda \qquad \frac{\Gamma \vdash s : B \Rightarrow A \quad \Gamma \vdash t : B}{\Gamma \vdash st : A}\,@$$

From the Untyped to the Simply Typed λ-Calculus

Let us introduce the simply typed λ-calculus in Church-style (STLC).

Types: $A, B ::= X \mid A \Rightarrow B$ given a set of *ground* types ranged over by $X, Y, Z \ldots$

(λ-)Terms: $s, t ::= x \mid \lambda x^A.t \mid st$

Environment: function from finitely many variables to types, noted $x_1 : A_1, \ldots, x_n : A_n$.
The well-typed terms are the ones that can be constructed via the *typing rules* below.

$$\frac{}{\Gamma, x : A \vdash x : A}\,\mathsf{var} \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x^A.t : A \Rightarrow B}\,\lambda \qquad \frac{\Gamma \vdash s : B \Rightarrow A \quad \Gamma \vdash t : B}{\Gamma \vdash st : A}\,@$$

The free variables of a term t are the variables that are not bound to a λ. Formally,

$$\mathsf{fv}(x) = \{x\} \qquad \mathsf{fv}(st) = \mathsf{fv}(s) \cup \mathsf{fv}(t) \qquad \mathsf{fv}(\lambda x.t) = \mathsf{fv}(t) \setminus \{x\}$$

From the Untyped to the Simply Typed λ-Calculus

Let us introduce the simply typed λ-calculus in Church-style (STLC).

Types: $A, B ::= X \mid A \Rightarrow B$ given a set of *ground* types ranged over by $X, Y, Z \ldots$

(λ-)Terms: $s, t ::= x \mid \lambda x^A.t \mid st$

Environment: function from finitely many variables to types, noted $x_1 : A_1, \ldots, x_n : A_n$.
The well-typed terms are the ones that can be constructed via the *typing rules* below.

$$\frac{}{\Gamma, x : A \vdash x : A}\,\mathsf{var} \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x^A.t : A \Rightarrow B}\,\lambda \qquad \frac{\Gamma \vdash s : B \Rightarrow A \quad \Gamma \vdash t : B}{\Gamma \vdash st : A}\,@$$

The free variables of a term t are the variables that are not bound to a λ. Formally,

$$\mathsf{fv}(x) = \{x\} \qquad \mathsf{fv}(st) = \mathsf{fv}(s) \cup \mathsf{fv}(t) \qquad \mathsf{fv}(\lambda x.t) = \mathsf{fv}(t) \setminus \{x\}$$

Proposition (If $\Gamma \vdash t : A$ is derivable , Γ is essentially a type assignment for $\mathsf{fv}(t)$)

1. If $\Gamma \vdash t : A$ is derivable, then so is $\Gamma, x : B \vdash t : A$, for any type B and $x \notin \mathsf{dom}(\Gamma)$.
2. If $\Gamma \vdash t : A$ is derivable, then $\mathsf{fv}(t) \subseteq \mathsf{dom}(\Gamma)$ and $\Gamma\restriction_{\mathsf{fv}(t)} \vdash t : A$ is derivable.

Let us introduce the simply typed λ-calculus in Church-style (STLC).

Types: $A, B ::= X \mid A \Rightarrow B$ given a set of *ground* types ranged over by $X, Y, Z \ldots$

(λ-)Terms: $s, t ::= x \mid \lambda x^A.t \mid st$

Environment: function from finitely many variables to types, noted $x_1 : A_1, \ldots, x_n : A_n$.
The well-typed terms are the ones that can be constructed via the *typing rules* below.

$$\frac{}{\Gamma, x : A \vdash x : A}\,\mathsf{var} \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x^A.t : A \Rightarrow B}\,\lambda \qquad \frac{\Gamma \vdash s : B \Rightarrow A \quad \Gamma \vdash t : B}{\Gamma \vdash st : A}\,@$$

The free variables of a term t are the variables that are not bound to a λ. Formally,

$$\mathsf{fv}(x) = \{x\} \qquad \mathsf{fv}(st) = \mathsf{fv}(s) \cup \mathsf{fv}(t) \qquad \mathsf{fv}(\lambda x.t) = \mathsf{fv}(t) \setminus \{x\}$$

Proposition (If $\Gamma \vdash t : A$ is derivable , Γ is essentially a type assignment for $\mathsf{fv}(t)$)

1. If $\Gamma \vdash t : A$ is derivable, then so is $\Gamma, x : B \vdash t : A$, for any type B and $x \notin \mathsf{dom}(\Gamma)$.
2. If $\Gamma \vdash t : A$ is derivable, then $\mathsf{fv}(t) \subseteq \mathsf{dom}(\Gamma)$ and $\Gamma\restriction_{\mathsf{fv}(t)} \vdash t : A$ is derivable.

β-reduction ($t\{s/x\}$ is the capture-avoiding substitution of s for the free occurrences of x in t):

$$(\lambda x^A.t)s \to_\beta t\{s/x\} \qquad \text{(possibly not well-typed)}$$

Rmk: $\lambda x^X.x$ and $\lambda x^{X \Rightarrow X}.x$ are different terms in Church-style, because $X \neq X \Rightarrow X$.
Idea: In Church-style STLC, types are intrinsic to terms (static typing, *a priori*).

Rmk: $\lambda x^X.x$ and $\lambda x^{X \Rightarrow X}.x$ are different terms in Church-style, because $X \neq X \Rightarrow X$.
Idea: In Church-style STLC, types are intrinsic to terms (static typing, *a priori*).

Syntax-directed: The search for a derivation is uniquely determined by the λ-term.
$\rightsquigarrow$ To build a derivation $\mathcal{D}$ of $\Gamma \vdash t : A$, just look at t to know the last rule (if any).

Rmk: $\lambda x^X.x$ and $\lambda x^{X \Rightarrow X}.x$ are different terms in Church-style, because $X \neq X \Rightarrow X$.
Idea: In Church-style STLC, types are intrinsic to terms (static typing, *a priori*).

Syntax-directed: The search for a derivation is uniquely determined by the λ-term.
$\leadsto$ To build a derivation $\mathcal{D}$ of $\Gamma \vdash t : A$, just look at t to know the last rule (if any).

Lemma (Typability of subterms, Substitution)

1. *Let t a term. If t is well-typed then so is every subterm of t.*
2. *If $\Gamma, x : A \vdash t : B$ and $\Gamma \vdash s : A$ are derivable, then so is $\Gamma \vdash t\{s/x\} : B$.*

Proof. Both points are proved by induction on t. □

Theorem (Subject reduction)

Let $t \to_\beta t'$. If $\Gamma \vdash t : A$ is derivable then so is $\Gamma \vdash t' : A$.

Proof. By induction on the definition of $t \to_\beta t'$, using the substitution lemma. □

Rmk: $\lambda x^X.x$ and $\lambda x^{X \Rightarrow X}.x$ are different terms in Church-style, because $X \neq X \Rightarrow X$.
Idea: In Church-style STLC, types are intrinsic to terms (static typing, *a priori*).

Syntax-directed: The search for a derivation is uniquely determined by the λ-term.
$\rightsquigarrow$ To build a derivation $\mathcal{D}$ of $\Gamma \vdash t : A$, just look at t to know the last rule (if any).

Lemma (Typability of subterms, Substitution)

1. *Let t a term. If t is well-typed then so is every subterm of t.*
2. *If $\Gamma, x : A \vdash t : B$ and $\Gamma \vdash s : A$ are derivable, then so is $\Gamma \vdash t\{s/x\} : B$.*

Proof. Both points are proved by induction on t. □

Theorem (Subject reduction)

Let $t \to_\beta t'$. If $\Gamma \vdash t : A$ is derivable then so is $\Gamma \vdash t' : A$.

Proof. By induction on the definition of $t \to_\beta t'$, using the substitution lemma. □

This means that well-typed terms are closed under β-reduction. But well-typed terms are not closed under β-expansion: Consider $(\lambda z^Z.x)(\delta\delta) \to_\beta x$ where $\delta = \lambda y^Y.yy$.

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x : A \vdash x : A}\,\mathsf{ax}^{x}$$

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x : A \vdash x : A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x : A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x$$

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{y : B \vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^B.x : B \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^B.x : A \Rightarrow B \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x$$

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x : A \vdash x : A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x : A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x : A, y : B \vdash x : A}\,\mathsf{ax}^x}{y : B \vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{\dfrac{}{x : A, y : B \vdash x : A}\,\mathsf{ax}^x}{x : A \vdash \lambda y^B.x : B \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^B.x : A \Rightarrow B \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x$$

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x : A \vdash x : A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x : A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x : A, y : B \vdash x : A}\,\mathsf{ax}^x}{y : B \vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{\dfrac{}{x : A, y : B \vdash x : A}\,\mathsf{ax}^x}{x : A \vdash \lambda y^B.x : B \Rightarrow A}\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^B.x : A \Rightarrow B \Rightarrow A}\Rightarrow_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{}{x : A, y : A \vdash x : A}\,\mathsf{ax}^x}{y : A \vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x$$

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{y : B \vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^B.x : B \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^B.x : A \Rightarrow B \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x:A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x:A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y$$

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{y : B \vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^B.x : B \Rightarrow A}\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^B.x : A \Rightarrow B \Rightarrow A}\Rightarrow_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^y \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x}{\vdash \lambda y^A.\lambda x^A.x : A \Rightarrow A \Rightarrow A}\Rightarrow_{\mathsf{i}}^y$$

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\frac{}{x:A \vdash x : A}\,\mathsf{ax}^x \qquad \frac{\frac{}{x:A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \frac{\frac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{y : B \vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \frac{\frac{\frac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^B.x : B \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^B.x : A \Rightarrow B \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x$$

$$\frac{\frac{}{x:A, y:A \vdash x : A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \frac{\frac{}{x:A, y:A \vdash x : A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y \qquad \frac{\frac{\frac{}{x:A, y:A \vdash x : A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x}{\vdash \lambda y^A.\lambda x^A.x : A \Rightarrow A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y \qquad \frac{\frac{\frac{}{x:A, y:A \vdash x : A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^A.x : A \Rightarrow A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x$$

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x$$

$$\dfrac{\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{y : B \vdash \lambda x^A.x : A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^B.x : B \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^B.x : A \Rightarrow B \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^y$$

$$\dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^x}{\vdash \lambda y^A.\lambda x^A.x : A \Rightarrow A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^y$$

$$\dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^A.x : A \Rightarrow A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash z : B \Rightarrow C}\,\mathsf{ax}^z \quad \dfrac{\dfrac{}{\Gamma \vdash x : A}\,\mathsf{ax}^x \quad \dfrac{}{\Gamma \vdash y : A \Rightarrow B}\,\mathsf{ax}^y}{x:A, y:A \Rightarrow B, z:B \Rightarrow C \vdash xy : B}\,{\Rightarrow}_{\mathsf{e}}}{x:A, y:A \Rightarrow B, z:B \Rightarrow C \vdash z(xy) : C}\,{\Rightarrow}_{\mathsf{e}}}{y:A \Rightarrow B, z:B \Rightarrow C \vdash \lambda x^A.z(xy) : A \Rightarrow C}\,{\Rightarrow}_{\mathsf{i}}^x}{y:A \Rightarrow B \vdash \lambda z^{B \Rightarrow C}.\lambda x^A.z(xy) : (B \Rightarrow C) \Rightarrow A \Rightarrow C}\,{\Rightarrow}_{\mathsf{i}}^z}{\vdash \lambda y^{A \Rightarrow B}.\lambda z^{B \Rightarrow C}.\lambda x^A.z(xy) : (A \Rightarrow B) \Rightarrow (B \Rightarrow C) \Rightarrow A \Rightarrow C}\,{\Rightarrow}_{\mathsf{i}}^y$$

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

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{y : B \vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^B.x : B \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^B.x : A \Rightarrow B \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x}{\vdash \lambda y^A.\lambda x^A.x : A \Rightarrow A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^A.x : A \Rightarrow A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash x : A \Rightarrow B \Rightarrow C}\,\mathsf{ax}^x \quad \dfrac{}{\Gamma \vdash z : A}\,\mathsf{ax}^z}{z : A,\ y : A \Rightarrow B,\ x : A \Rightarrow B \Rightarrow C \vdash xz : B \Rightarrow C}\,\Rightarrow_{\mathsf{e}} \qquad \dfrac{\dfrac{}{\Gamma \vdash y : A \Rightarrow B}\,\mathsf{ax}^y \quad \dfrac{}{\Gamma \vdash z : A}\,\mathsf{ax}^z}{z : A,\ y : A \Rightarrow B,\ x : A \Rightarrow B \Rightarrow C \vdash yz : B}\,\Rightarrow_{\mathsf{e}}}{A,\ A \Rightarrow B,\ A \Rightarrow (B \Rightarrow C) \vdash xz(yz) : C}\,\Rightarrow_{\mathsf{e}}}{A \Rightarrow B,\ A \Rightarrow (B \Rightarrow C) \vdash \lambda z.xz(yz) : A \Rightarrow C}\,\Rightarrow_{\mathsf{i}}^z}{A \Rightarrow (B \Rightarrow C) \vdash \lambda y.\lambda z.xz(yz) : (A \Rightarrow B) \Rightarrow (A \Rightarrow C)}\,\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x.\lambda y.\lambda z.xz(yz) : (A \Rightarrow (B \Rightarrow C)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow C)}\,\Rightarrow_{\mathsf{i}}^x$$

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

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x:A\vdash x:A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x:A\vdash x:A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A, y:B\vdash x:A}\,\mathsf{ax}^x}{y:B\vdash \lambda x^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:B\vdash x:A}\,\mathsf{ax}^x}{x:A\vdash \lambda y^B.x : B\Rightarrow A}\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^B.x : A\Rightarrow B\Rightarrow A}\Rightarrow_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{}{x:A, y:A\vdash x:A}\,\mathsf{ax}^x}{y:A\vdash \lambda x^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A, y:A\vdash x:A}\,\mathsf{ax}^x}{x:A\vdash \lambda y^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^y \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A\vdash x:A}\,\mathsf{ax}^x}{y:A\vdash \lambda x^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^x}{\vdash \lambda y^A.\lambda x^A.x : A\Rightarrow A\Rightarrow A}\Rightarrow_{\mathsf{i}}^y \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A\vdash x:A}\,\mathsf{ax}^x}{x:A\vdash \lambda y^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^A.x : A\Rightarrow A\Rightarrow A}\Rightarrow_{\mathsf{i}}^x$$

Examples

1. Prove that xx cannot be well-typed with any type A and any environment Γ.

From the Untyped to the Simply Typed λ-Calculus

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\,{\Rightarrow}_i^x \qquad \dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{y : B \vdash \lambda x^A.x : A \Rightarrow A}\,{\Rightarrow}_i^x \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^B.x : B \Rightarrow A}\,{\Rightarrow}_i^y}{\vdash \lambda x^A.\lambda y^B.x : A \Rightarrow B \Rightarrow A}\,{\Rightarrow}_i^x$$

$$\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\,{\Rightarrow}_i^x \qquad \dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\,{\Rightarrow}_i^y \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\,{\Rightarrow}_i^x}{\vdash \lambda y^A.\lambda x^A.x : A \Rightarrow A \Rightarrow A}\,{\Rightarrow}_i^y \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\,{\Rightarrow}_i^y}{\vdash \lambda x^A.\lambda y^A.x : A \Rightarrow A \Rightarrow A}\,{\Rightarrow}_i^x$$

Examples

1. Prove that xx cannot be well-typed with any type A and any environment Γ.
   By syntax-direction, a derivation of $xx : A$ must have the form below

$$\dfrac{\dfrac{}{x : B \Rightarrow A \vdash x : B \Rightarrow A}\,\mathsf{var}^x \qquad \dfrac{}{x : B \vdash x : B}\,\mathsf{var}^x}{\Gamma \vdash xx : A}\,@$$

   but this is not a derivation because $B = B \Rightarrow A$ should hold, which is impossible.

From the Untyped to the Simply Typed λ-Calculus

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\frac{}{x:A \vdash x : A}\,\mathsf{ax}^x \qquad \frac{\dfrac{}{x:A \vdash x : A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \frac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{y : B \vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \frac{\dfrac{\dfrac{}{x:A, y:B \vdash x : A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^B.x : B \Rightarrow A}\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A\lambda y^B.x : A \Rightarrow B \Rightarrow A}\Rightarrow_{\mathsf{i}}^x$$

$$\frac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \frac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^y \qquad \frac{\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{y:A \vdash \lambda x^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x}{\vdash \lambda y^A\lambda x^A.x : A \Rightarrow A \Rightarrow A}\Rightarrow_{\mathsf{i}}^y \qquad \frac{\dfrac{\dfrac{}{x:A, y:A \vdash x:A}\,\mathsf{ax}^x}{x:A \vdash \lambda y^A.x : A \Rightarrow A}\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A\lambda y^A.x : A \Rightarrow A \Rightarrow A}\Rightarrow_{\mathsf{i}}^x$$

Examples

1. Prove that xx cannot be well-typed with any type A and any environment Γ.
   By syntax-direction, a derivation of $xx : A$ must have the form below
   $$\frac{\dfrac{}{x : B \Rightarrow A \vdash x : B \Rightarrow A}\,\mathsf{var}^x \qquad \dfrac{}{x : B \vdash x : B}\,\mathsf{var}^x}{\Gamma \vdash xx : A}\,@$$
   but this is not a derivation because $B = B \Rightarrow A$ should hold, which is impossible.
2. Prove that $\delta_A = \lambda x^A.xx$ and $\delta_A\delta_A$ cannot be well-typed with any A, Γ.

Some examples of derivations in STLC ($\Rightarrow$ associates to the right).

$$\dfrac{}{x:A\vdash x:A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x:A\vdash x:A}\,\mathsf{ax}^x}{\vdash \lambda x^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A,y:B\vdash x:A}\,\mathsf{ax}^x}{y:B\vdash \lambda x^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{\dfrac{}{x:A,y:B\vdash x:A}\,\mathsf{ax}^x}{x:A\vdash \lambda y^B.x : B\Rightarrow A}\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^B.x : A\Rightarrow B\Rightarrow A}\Rightarrow_{\mathsf{i}}^x$$

$$\dfrac{\dfrac{}{x:A,y:A\vdash x:A}\,\mathsf{ax}^x}{y:A\vdash \lambda x^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A,y:A\vdash x:A}\,\mathsf{ax}^x}{x:A\vdash \lambda y^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^y \qquad \dfrac{\dfrac{\dfrac{}{x:A,y:A\vdash x:A}\,\mathsf{ax}^x}{y:A\vdash \lambda x^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^x}{\vdash \lambda y^A.\lambda x^A.x : A\Rightarrow A\Rightarrow A}\Rightarrow_{\mathsf{i}}^y \qquad \dfrac{\dfrac{\dfrac{}{x:A,y:A\vdash x:A}\,\mathsf{ax}^x}{x:A\vdash \lambda y^A.x : A\Rightarrow A}\Rightarrow_{\mathsf{i}}^y}{\vdash \lambda x^A.\lambda y^A.x : A\Rightarrow A\Rightarrow A}\Rightarrow_{\mathsf{i}}^x$$

Examples

1. Prove that xx cannot be well-typed with any type A and any environment Γ.
   By syntax-direction, a derivation of $xx : A$ must have the form below

$$\dfrac{\dfrac{}{x:B\Rightarrow A\vdash x:B\Rightarrow A}\,\mathsf{var}^x \qquad \dfrac{}{x:B\vdash x:B}\,\mathsf{var}^x}{\Gamma\vdash xx : A}\,@$$

   but this is not a derivation because $B = B\Rightarrow A$ should hold, which is impossible.
2. Prove that $\delta_A = \lambda x^A.xx$ and $\delta_A\delta_A$ cannot be well-typed with any A,Γ.
   If δ_A and $\delta_A\delta_A$ were well-typed for some type A and environment Γ, so would be their subterm xx by Lemma 1 (p. 11), but this is impossible by the previous point.

Natural Deduction for Minimal Logic

We introduce natural deduction for minimal (= implicative intuitionistic) logic (ND). The types used in the STLC are exactly the formulas of minimal logic.

A sequent is a pair $\Gamma \vdash A$ where Γ is an environment and A is a type of STLC.

A derivation in ND is a tree built up from the inference rules below.

$$\frac{}{\Gamma, x\!:\!A \vdash A}\,\mathsf{ax}^x \qquad \frac{\Gamma, x\!:\!A \vdash B}{\Gamma \vdash A \Rightarrow B}\,\Rightarrow_{\mathsf{i}}^{x} \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\,\Rightarrow_{\mathsf{e}}$$

Natural Deduction for Minimal Logic

We introduce natural deduction for minimal (= implicative intuitionistic) logic (ND).
The types used in the STLC are exactly the formulas of minimal logic.

A sequent is a pair $\Gamma \vdash A$ where Γ is an environment and A is a type of STLC.

A derivation in ND is a tree built up from the inference rules below.

$$\frac{}{\Gamma, x\!:\!A \vdash A}\,\mathsf{ax}^x \qquad \frac{\Gamma, x\!:\!A \vdash B}{\Gamma \vdash A \Rightarrow B}\Rightarrow_{\mathsf{i}}^{x} \qquad \frac{\Gamma \vdash A \Rightarrow B \qquad \Gamma \vdash A}{\Gamma \vdash B}\Rightarrow_{\mathsf{e}}$$

Theorem (Soundness and completeness)

A sequent $x_1 : A_1, \ldots, x_n : A_n \vdash B$ is derivable in ND *if and only if $A_1 \Rightarrow \cdots \Rightarrow A_n \Rightarrow B$ is valid in minimal logic.*

Natural Deduction for Minimal Logic

We introduce natural deduction for minimal (= implicative intuitionistic) logic (ND).
The types used in the STLC are exactly the formulas of minimal logic.

A sequent is a pair $\Gamma \vdash A$ where Γ is an environment and A is a type of STLC.

A derivation in ND is a tree built up from the inference rules below.

$$\frac{}{\Gamma, x : A \vdash A}\,\mathsf{ax}^x \qquad \frac{\Gamma, x : A \vdash B}{\Gamma \vdash A \Rightarrow B}\Rightarrow_{\mathsf{i}}^x \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\Rightarrow_{\mathsf{e}}$$

Theorem (Soundness and completeness)

A sequent $x_1 : A_1, \ldots, x_n : A_n \vdash B$ is derivable in ND if and only if $A_1 \Rightarrow \cdots \Rightarrow A_n \Rightarrow B$ is valid in minimal logic.

Natural Deduction for Minimal Logic

We introduce natural deduction for minimal (= implicative intuitionistic) logic (ND). The types used in the STLC are exactly the formulas of minimal logic.

A sequent is a pair $\Gamma \vdash A$ where Γ is an environment and A is a type of STLC.

A derivation in ND is a tree built up from the inference rules below.

$$\frac{}{\Gamma, x : A \vdash A}\,\mathsf{ax}^x \qquad \frac{\Gamma, x : A \vdash B}{\Gamma \vdash A \Rightarrow B}\Rightarrow_{\mathsf{i}}^{x} \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\Rightarrow_{\mathsf{e}}$$

Theorem (Soundness and completeness)

A sequent $x_1 : A_1, \ldots, x_n : A_n \vdash B$ is derivable in ND if and only if $A_1 \Rightarrow \cdots \Rightarrow A_n \Rightarrow B$ is valid in minimal logic.

$\mathcal{D}\{\mathcal{D}'/x\}$ stands for the substitution of derivation $\mathcal{D}'$ for ax rules labeled by x in $\mathcal{D}$. The definition is by induction on $\mathcal{D}$, when $\mathcal{D}$ proves $\Gamma, x : A \vdash B$ and $\mathcal{D}'$ proves $\Gamma \vdash A$.

- $\mathcal{D} = \frac{}{\Gamma,\ x : A \vdash A}\,\mathsf{ax}^x$: then $\mathcal{D}\{\mathcal{D}'/x\} = \mathcal{D}'$.
- $\mathcal{D} = \frac{}{\Gamma,\ x : A \vdash B}\,\mathsf{ax}^y$ where $x \neq y$: then $y \in \mathsf{dom}(\Gamma)$ and $\mathcal{D}\{\mathcal{D}'/x\} = \frac{}{\Gamma \vdash B}\,\mathsf{ax}^y$.

Natural Deduction for Minimal Logic

We introduce natural deduction for minimal (= implicative intuitionistic) logic (ND). The types used in the STLC are exactly the formulas of minimal logic.

A sequent is a pair $\Gamma \vdash A$ where Γ is an environment and A is a type of STLC.

A derivation in ND is a tree built up from the inference rules below.

$$\frac{}{\Gamma, x\!:\!A \vdash A}\,\mathsf{ax}^x \qquad \frac{\Gamma, x\!:\!A \vdash B}{\Gamma \vdash A \Rightarrow B}\,\Rightarrow_{\mathsf{i}}^x \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\,\Rightarrow_{\mathsf{e}}$$

Theorem (Soundness and completeness)

A sequent $x_1 : A_1, \ldots, x_n : A_n \vdash B$ is derivable in ND *if and only if $A_1 \Rightarrow \cdots \Rightarrow A_n \Rightarrow B$ is valid in minimal logic.*

$\mathcal{D}\{\mathcal{D}'/x\}$ stands for the substitution of derivation $\mathcal{D}'$ for ax rules labeled by x in $\mathcal{D}$. The definition is by induction on $\mathcal{D}$, when $\mathcal{D}$ proves $\Gamma, x : A \vdash B$ and $\mathcal{D}'$ proves $\Gamma \vdash A$.

- $\mathcal{D} = \frac{}{\Gamma,\ x : A \vdash A}\,\mathsf{ax}^x$: then $\mathcal{D}\{\mathcal{D}'/x\} = \mathcal{D}'$.
- $\mathcal{D} = \frac{}{\Gamma,\ x : A \vdash B}\,\mathsf{ax}^y$ where $x \neq y$: then $y \in \mathsf{dom}(\Gamma)$ and $\mathcal{D}\{\mathcal{D}'/x\} = \frac{}{\Gamma \vdash B}\,\mathsf{ax}^y$.
- $\mathcal{D} = \dfrac{\begin{matrix}\vdots\,\mathcal{D}_1 \\ \Gamma,\ x\!:\!A \vdash C \Rightarrow B\end{matrix} \quad \begin{matrix}\vdots\,\mathcal{D}_2 \\ \Gamma,\ x\!:\!A \vdash C\end{matrix}}{\Gamma,\ x\!:\!A \vdash B}\,@$: then $\mathcal{D}\{\mathcal{D}'/x\} = \dfrac{\begin{matrix}\vdots\,\mathcal{D}_1\{\mathcal{D}'/x\} \\ \Gamma \vdash C \Rightarrow B\end{matrix} \quad \begin{matrix}\vdots\,\mathcal{D}_2\{\mathcal{D}'/x\} \\ \Gamma \vdash C\end{matrix}}{\Gamma \vdash B}\,@$.

Natural Deduction for Minimal Logic

We introduce natural deduction for minimal (= implicative intuitionistic) logic (ND). The types used in the STLC are exactly the formulas of minimal logic.

A sequent is a pair $\Gamma \vdash A$ where Γ is an environment and A is a type of STLC.

A derivation in ND is a tree built up from the inference rules below.

$$\frac{}{\Gamma, x\!:\!A \vdash A}\,\mathsf{ax}^x \qquad \frac{\Gamma, x\!:\!A \vdash B}{\Gamma \vdash A \Rightarrow B}\,\Rightarrow_{\mathsf{i}}^x \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\,\Rightarrow_{\mathsf{e}}$$

Theorem (Soundness and completeness)

A sequent $x_1 : A_1, \ldots, x_n : A_n \vdash B$ is derivable in ND *if and only if $A_1 \Rightarrow \cdots \Rightarrow A_n \Rightarrow B$ is valid in minimal logic.*

$\mathcal{D}\{\mathcal{D}'/x\}$ stands for the substitution of derivation $\mathcal{D}'$ for ax rules labeled by x in $\mathcal{D}$. The definition is by induction on $\mathcal{D}$, when $\mathcal{D}$ proves $\Gamma, x : A \vdash B$ and $\mathcal{D}'$ proves $\Gamma \vdash A$.

- $\mathcal{D} = \frac{}{\Gamma,\ x : A \vdash A}\,\mathsf{ax}^x$: then $\mathcal{D}\{\mathcal{D}'/x\} = \mathcal{D}'$.
- $\mathcal{D} = \frac{}{\Gamma,\ x : A \vdash B}\,\mathsf{ax}^y$ where $x \neq y$: then $y \in \mathsf{dom}(\Gamma)$ and $\mathcal{D}\{\mathcal{D}'/x\} = \frac{}{\Gamma \vdash B}\,\mathsf{ax}^y$.
- $\mathcal{D} = \dfrac{\begin{matrix}\vdots\ \mathcal{D}_0 \\ \Gamma, x\!:\!A, y\!:\!C \vdash D\end{matrix}}{\Gamma, x\!:\!A \vdash C \Rightarrow D}\,\Rightarrow_{\mathsf{i}}^y$ with $y \notin \{x\} \cup \mathsf{dom}(\Gamma)$: so $\mathcal{D}\{\mathcal{D}'/x\} = \dfrac{\begin{matrix}\vdots\ \mathcal{D}_0\{\widehat{\mathcal{D}'}/x\} \\ \Gamma,\ y\!:\!C \vdash D\end{matrix}}{\Gamma \vdash C \Rightarrow D}\,\Rightarrow_{\mathsf{i}}^y$

  where $\widehat{\mathcal{D}'}$ is obtained from $\mathcal{D}'$ by adding $y\!:\!C$ to the environment of each ax rule.

Natural Deduction for Minimal Logic

We introduce natural deduction for minimal (= implicative intuitionistic) logic (ND). The types used in the STLC are exactly the formulas of minimal logic.

A sequent is a pair $\Gamma \vdash A$ where Γ is an environment and A is a type of STLC.

A derivation in ND is a tree built up from the inference rules below.

$$\frac{}{\Gamma, x : A \vdash A}\,\mathsf{ax}^x \qquad \frac{\Gamma, x : A \vdash B}{\Gamma \vdash A \Rightarrow B}\Rightarrow_{\mathsf{i}}^x \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\Rightarrow_{\mathsf{e}}$$

Theorem (Soundness and completeness)

A sequent $x_1 : A_1, \ldots, x_n : A_n \vdash B$ is derivable in ND if and only if $A_1 \Rightarrow \cdots \Rightarrow A_n \Rightarrow B$ is valid in minimal logic.

$\mathcal{D}\{\mathcal{D}'/x\}$ stands for the substitution of derivation $\mathcal{D}'$ for ax rules labeled by x in $\mathcal{D}$.

Rmk: If $\mathcal{D}$ proves $\Gamma, x : A \vdash B$ and $\mathcal{D}'$ proves $\Gamma \vdash A$, then $\mathcal{D}\{\mathcal{D}'/x\}$ proves $\Gamma \vdash B$.

Natural Deduction for Minimal Logic

We introduce natural deduction for minimal (= implicative intuitionistic) logic (ND). The types used in the STLC are exactly the formulas of minimal logic.

A sequent is a pair $\Gamma \vdash A$ where Γ is an environment and A is a type of STLC.

A derivation in ND is a tree built up from the inference rules below.

$$\frac{}{\Gamma, x : A \vdash A}\,\mathsf{ax}^x \qquad \frac{\Gamma, x : A \vdash B}{\Gamma \vdash A \Rightarrow B}\,\Rightarrow_{\mathsf{i}}^x \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\,\Rightarrow_{\mathsf{e}}$$

Theorem (Soundness and completeness)

A sequent $x_1 : A_1, \ldots, x_n : A_n \vdash B$ is derivable in ND if and only if $A_1 \Rightarrow \cdots \Rightarrow A_n \Rightarrow B$ is valid in minimal logic.

$\mathcal{D}\{\mathcal{D}'/x\}$ stands for the substitution of derivation $\mathcal{D}'$ for ax rules labeled by x in $\mathcal{D}$.

Rmk: If $\mathcal{D}$ proves $\Gamma, x : A \vdash B$ and $\mathcal{D}'$ proves $\Gamma \vdash A$, then $\mathcal{D}\{\mathcal{D}'/x\}$ proves $\Gamma \vdash B$.

cut-elimination:

$$\frac{\dfrac{\begin{array}{c}\vdots\,\mathcal{D}_1\\ \Gamma, x : A \vdash B\end{array}}{\Gamma \vdash A \Rightarrow B}\,\Rightarrow_{\mathsf{i}}^x \qquad \begin{array}{c}\vdots\,\mathcal{D}_2\\ \Gamma \vdash A\end{array}}{\Gamma \vdash B}\,\Rightarrow_{\mathsf{e}} \quad \to_{\mathsf{cut}} \quad \begin{array}{c}\vdots\,\mathcal{D}_1\{\mathcal{D}_2/x\}\\ \Gamma \vdash B\end{array}$$

Rmk: Variable labels on ax and $\Rightarrow_{\mathsf{i}}$ are crucial to define cut-elimin. and substitution.

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

$$\frac{}{x : A \vdash A}\,\mathsf{ax}^{x}$$

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

$$\dfrac{}{x : A \vdash A}\,\mathsf{ax}^x \qquad \dfrac{\dfrac{}{x : A \vdash A}\,\mathsf{ax}^x}{\vdash A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x$$

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

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

Natural Deduction for Minimal Logic

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

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

2. Give two (distinct) derivations of $A \vdash A \Rightarrow A$ and two ones of $\vdash A \Rightarrow A \Rightarrow A$.

Natural Deduction for Minimal Logic

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

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

2. Give two (distinct) derivations of $A \vdash A \Rightarrow A$ and two ones of $\vdash A \Rightarrow A \Rightarrow A$.

$$\dfrac{\dfrac{}{x : A, y : A \vdash A}\,\mathsf{ax}^x}{y : A \vdash A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x$$

Natural Deduction for Minimal Logic

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

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

2. Give two (distinct) derivations of $A \vdash A \Rightarrow A$ and two ones of $\vdash A \Rightarrow A \Rightarrow A$.

$$\dfrac{\dfrac{}{x : A, y : A \vdash A}\,\mathsf{ax}^x}{y : A \vdash A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^{x} \qquad \dfrac{\dfrac{}{x : A, y : A \vdash A}\,\mathsf{ax}^x}{x : A \vdash A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^{y}$$

Natural Deduction for Minimal Logic

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

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

2. Give two (distinct) derivations of $A \vdash A \Rightarrow A$ and two ones of $\vdash A \Rightarrow A \Rightarrow A$.

$$\dfrac{\dfrac{}{x : A, y : A \vdash A}\,\mathsf{ax}^x}{y : A \vdash A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x : A, y : A \vdash A}\,\mathsf{ax}^x}{x : A \vdash A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y \qquad \dfrac{\dfrac{\dfrac{}{x : A, y : A \vdash A}\,\mathsf{ax}^x}{y : A \vdash A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^x}{\vdash A \Rightarrow A \Rightarrow A}\,\Rightarrow_{\mathsf{i}}^y$$

Natural Deduction for Minimal Logic

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

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

2. Give two (distinct) derivations of $A \vdash A \Rightarrow A$ and two ones of $\vdash A \Rightarrow A \Rightarrow A$.

$$\dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{y:A \vdash A \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^x} \qquad \dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{x:A \vdash A \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^y} \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{y:A \vdash A \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^x}}{\vdash A \Rightarrow A \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^y} \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{x:A \vdash A \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^y}}{\vdash A \Rightarrow A \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^x}$$

Natural Deduction for Minimal Logic

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

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

2. Give two (distinct) derivations of $A \vdash A \Rightarrow A$ and two ones of $\vdash A \Rightarrow A \Rightarrow A$.

$$\dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{y:A \vdash A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^x \qquad \dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{x:A \vdash A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^y \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{y:A \vdash A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^x}{\vdash A \Rightarrow A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^y \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{x:A \vdash A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^y}{\vdash A \Rightarrow A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^x$$

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

Natural Deduction for Minimal Logic

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

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

2. Give two (distinct) derivations of $A \vdash A \Rightarrow A$ and two ones of $\vdash A \Rightarrow A \Rightarrow A$.

$$\dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{y:A \vdash A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^{x} \qquad \dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{x:A \vdash A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^{y} \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{y:A \vdash A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^{x}}{\vdash A \Rightarrow A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^{y} \qquad \dfrac{\dfrac{\dfrac{}{x:A, y:A \vdash A}\,\mathsf{ax}^x}{x:A \vdash A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^{y}}{\vdash A \Rightarrow A \Rightarrow A}\,{\Rightarrow}_{\mathsf{i}}^{x}$$

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

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x:A, y:A \Rightarrow B, z:B \Rightarrow C \vdash B \Rightarrow C}\,\mathsf{ax}^z \quad \dfrac{\dfrac{}{x:A, y:A \Rightarrow B, z:B \Rightarrow C \vdash A}\,\mathsf{ax}^x \quad \dfrac{}{x:A, y:A \Rightarrow B, z:B \Rightarrow C \vdash A \Rightarrow B}\,\mathsf{ax}^y}{x:A, y:A \Rightarrow B, z:B \Rightarrow C \vdash B}\,{\Rightarrow}_{\mathsf{e}}}{x:A, y:A \Rightarrow B, z:B \Rightarrow C \vdash C}\,{\Rightarrow}_{\mathsf{e}}}{y:A \Rightarrow B, z:B \Rightarrow C \vdash A \Rightarrow C}\,{\Rightarrow}_{\mathsf{i}}^{x}}{y:A \Rightarrow B \vdash (B \Rightarrow C) \Rightarrow A \Rightarrow C}\,{\Rightarrow}_{\mathsf{i}}^{z}}{\vdash (A \Rightarrow B) \Rightarrow (B \Rightarrow C) \Rightarrow A \Rightarrow C}\,{\Rightarrow}_{\mathsf{i}}^{y}$$

Some examples of derivations in ND ($\Rightarrow$ associates to the right).

1. Prove that $A \vdash A$, and $\vdash A \Rightarrow A$, and $B \vdash A \Rightarrow A$, and $\vdash A \Rightarrow B \Rightarrow A$.

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

2. Give two (distinct) derivations of $A \vdash A \Rightarrow A$ and two ones of $\vdash A \Rightarrow A \Rightarrow A$.

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

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

$$\frac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B \Rightarrow C}\,\mathsf{ax}^x \quad \dfrac{}{\Gamma \vdash A}\,\mathsf{ax}^z}{A, A \Rightarrow B, A \Rightarrow B \Rightarrow C \vdash B \Rightarrow C}\Rightarrow_{\mathsf{e}} \quad \dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B}\,\mathsf{ax}^y \quad \dfrac{}{\Gamma \vdash A}\,\mathsf{ax}^z}{A, A \Rightarrow B, A \Rightarrow B \Rightarrow C \vdash B}\Rightarrow_{\mathsf{e}}}{A, A \Rightarrow B, A \Rightarrow (B \Rightarrow C) \vdash C}\Rightarrow_{\mathsf{e}}}{A \Rightarrow B, A \Rightarrow (B \Rightarrow C) \vdash A \Rightarrow C}\Rightarrow_{\mathsf{i}}^{z}}{A \Rightarrow (B \Rightarrow C) \vdash (A \Rightarrow B) \Rightarrow (A \Rightarrow C)}\Rightarrow_{\mathsf{i}}^{y}}{\vdash (A \Rightarrow (B \Rightarrow C)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow C)}\Rightarrow_{\mathsf{i}}^{x}$$

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

An example of cut-elimination step in ND, where $\Gamma = z : A,\ y : A \Rightarrow B,\ x : A \Rightarrow B \Rightarrow A$.

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{ax^{x}}
\UnaryInfC{$\Gamma \vdash A \Rightarrow B \Rightarrow A$}
\AxiomC{}
\RightLabel{ax^{z}}
\UnaryInfC{$\Gamma \vdash A$}
\RightLabel{$\Rightarrow_{\mathsf{e}}$}
\BinaryInfC{$\Gamma \vdash B \Rightarrow A$}
\AxiomC{}
\RightLabel{ax^{y}}
\UnaryInfC{$\Gamma \vdash A \Rightarrow B$}
\AxiomC{}
\RightLabel{ax^{z}}
\UnaryInfC{$\Gamma \vdash A$}
\RightLabel{$\Rightarrow_{\mathsf{e}}$}
\BinaryInfC{$\Gamma \vdash B$}
\RightLabel{$\Rightarrow_{\mathsf{e}}$}
\BinaryInfC{$\Gamma \vdash A$}
\RightLabel{$\Rightarrow_{\mathsf{i}}^{z}$}
\UnaryInfC{$x : A \Rightarrow B,\ y : A \Rightarrow (B \Rightarrow A) \vdash A \Rightarrow A$}
\RightLabel{$\Rightarrow_{\mathsf{i}}^{y}$}
\UnaryInfC{$x : A \Rightarrow (B \Rightarrow A) \vdash (A \Rightarrow B) \Rightarrow (A \Rightarrow A)$}
\RightLabel{$\Rightarrow_{\mathsf{i}}^{x}$}
\UnaryInfC{$\vdash (A \Rightarrow (B \Rightarrow A)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow A)$}
\AxiomC{}
\RightLabel{ax^{a}}
\UnaryInfC{$a : A,\ b : B \vdash A$}
\RightLabel{$\Rightarrow_{\mathsf{i}}^{b}$}
\UnaryInfC{$a : A \vdash B \Rightarrow A$}
\RightLabel{$\Rightarrow_{\mathsf{i}}^{a}$}
\UnaryInfC{$\vdash A \Rightarrow B \Rightarrow A$}
\RightLabel{$\Rightarrow_{\mathsf{e}}$}
\BinaryInfC{$\vdash (A \Rightarrow B) \Rightarrow A \Rightarrow A$}
\end{prooftree}
$$

An example of cut-elimination step in ND, where $\Gamma = z : A,\ y : A \Rightarrow B,\ x : A \Rightarrow B \Rightarrow A$.

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B \Rightarrow A}\,\mathsf{ax}^{x} \quad \dfrac{}{\Gamma \vdash A}\,\mathsf{ax}^{z}}{\Gamma \vdash B \Rightarrow A}\,{\Rightarrow_{\mathsf{e}}}
\qquad
\dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B}\,\mathsf{ax}^{y} \quad \dfrac{}{\Gamma \vdash A}\,\mathsf{ax}^{z}}{\Gamma \vdash B}\,{\Rightarrow_{\mathsf{e}}}
}{\Gamma \vdash A}\,{\Rightarrow_{\mathsf{e}}}
}{x : A \Rightarrow B,\ y : A \Rightarrow (B \Rightarrow A) \vdash A \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^{z}}
}{x : A \Rightarrow (B \Rightarrow A) \vdash (A \Rightarrow B) \Rightarrow (A \Rightarrow A)}\,{\Rightarrow_{\mathsf{i}}^{y}}
}{\vdash (A \Rightarrow (B \Rightarrow A)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow A)}\,{\Rightarrow_{\mathsf{i}}^{x}}
\qquad
\dfrac{\dfrac{\dfrac{}{a : A,\ b : B \vdash A}\,\mathsf{ax}^{a}}{a : A \vdash B \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^{b}}}{\vdash A \Rightarrow B \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^{a}}
}{}
}{\vdash (A \Rightarrow B) \Rightarrow A \Rightarrow A}\,{\Rightarrow_{\mathsf{e}}}
$$

$$\downarrow_{\mathsf{cut}}$$

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{\dfrac{\dfrac{}{\Delta, a : A,\ b : B \vdash A}\,\mathsf{ax}^{a}}{\Delta, a : A \vdash B \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^{b}}}{\Delta \vdash A \Rightarrow B \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^{a}}
\quad \dfrac{}{\Delta \vdash A}\,\mathsf{ax}^{z}
}{\Delta \vdash B \Rightarrow A}\,{\Rightarrow_{\mathsf{e}}}
\qquad
\dfrac{\dfrac{}{\Delta \vdash A \Rightarrow B}\,\mathsf{ax}^{y} \quad \dfrac{}{\Delta \vdash A}\,\mathsf{ax}^{z}}{\Delta \vdash B}\,{\Rightarrow_{\mathsf{e}}}
}{\Delta \vdash A}\,{\Rightarrow_{\mathsf{e}}}
}{y : A \Rightarrow B \vdash A \Rightarrow A}\,{\Rightarrow_{\mathsf{i}}^{z}}
}{\vdash (A \Rightarrow B) \Rightarrow (A \Rightarrow A)}\,{\Rightarrow_{\mathsf{i}}^{y}}
$$

where $\Delta = z : A,\ y : A \Rightarrow B$.

The Curry-Howard Correspondence between ND and STLC

The inference rules for the simply typed λ-calculus are the ones of ND plus *decoration*.
$\rightsquigarrow$ We can decorate each sequent in a derivation of ND with a well-typed term.

The inference rules for the simply typed λ-calculus are the ones of ND plus *decoration*.

$\rightsquigarrow$ We can decorate each sequent in a derivation of ND with a well-typed term.

$\rightsquigarrow$ Each derivation $\mathcal{D}$ in ND corresponds to a unique well-typed λ-term $(\mathcal{D})_\lambda$ defined by

$$\left(\frac{}{\Gamma, x : A \vdash A}\,\mathsf{ax}^x\right)_\lambda = x \qquad \left(\frac{\begin{array}{c}\vdots\ \mathcal{D}\\ \Gamma, x : A \vdash B\end{array}}{\Gamma \vdash A \Rightarrow B}\Rightarrow_{\mathsf{i}}^x\right)_\lambda = \lambda x.(\mathcal{D})_\lambda \qquad \left(\frac{\begin{array}{c}\vdots\mathcal{D}\\ \Gamma \vdash A \Rightarrow B\end{array} \quad \begin{array}{c}\vdots\mathcal{D}'\\ \Gamma \vdash A\end{array}}{\Gamma \vdash B}\Rightarrow_{\mathsf{e}}\right)_\lambda = (\mathcal{D})_\lambda(\mathcal{D}')_\lambda$$

Rmk. The variable labeling the rules ax and $\Rightarrow_{\mathsf{i}}$ is crucial to uniquely determine $(\mathcal{D})_\lambda$.

The inference rules for the simply typed λ-calculus are the ones of ND plus *decoration*.
$\leadsto$ We can decorate each sequent in a derivation of ND with a well-typed term.
$\leadsto$ Each derivation $\mathcal{D}$ in ND corresponds to a unique well-typed λ-term $(\mathcal{D})_\lambda$ defined by

$$\left(\frac{}{\Gamma, x : A \vdash A}\,\mathsf{ax}^x\right)_\lambda = x \qquad \left(\dfrac{\begin{array}{c}\vdots\,\mathcal{D}\\ \Gamma, x : A \vdash B\end{array}}{\Gamma \vdash A \Rightarrow B}\Rightarrow_\mathsf{i}^x\right)_\lambda = \lambda x.(\mathcal{D})_\lambda \qquad \left(\dfrac{\begin{array}{c}\vdots\,\mathcal{D}\\ \Gamma \vdash A \Rightarrow B\end{array} \quad \begin{array}{c}\vdots\,\mathcal{D}'\\ \Gamma \vdash A\end{array}}{\Gamma \vdash B}\Rightarrow_\mathsf{e}\right)_\lambda = (\mathcal{D})_\lambda(\mathcal{D}')_\lambda$$

Rmk. The variable labeling the rules ax and $\Rightarrow_\mathsf{i}$ is crucial to uniquely determine $(\mathcal{D})_\lambda$.

Rmk. If $\mathcal{D}$ derives $\Gamma \vdash A$ in ND, then $\Gamma \vdash (\mathcal{D})_\lambda : A$ is derivable in STLC.

Proposition (Uniqueness of type and derivation for well-typed terms)

In STLC, if $\mathcal{D}$ derives $\Gamma \vdash t : A$ and $\mathcal{D}'$ derives $\Gamma \vdash t : A'$, then $A = A'$ and $\mathcal{D} = \mathcal{D}'$.

Proof. By structural induction on t (exercise!). □

The inference rules for the simply typed λ-calculus are the ones of ND plus *decoration*.
$\rightsquigarrow$ We can decorate each sequent in a derivation of ND with a well-typed term.
$\rightsquigarrow$ Each derivation $\mathcal{D}$ in ND corresponds to a unique well-typed λ-term $(\mathcal{D})_\lambda$ defined by

$$\left(\overline{\Gamma, x : A \vdash A}\ \mathsf{ax}^x\right)_\lambda = x \qquad \left(\dfrac{\begin{array}{c}\vdots\ \mathcal{D}\\ \Gamma, x : A \vdash B\end{array}}{\Gamma \vdash A \Rightarrow B} \Rightarrow_{\mathsf{i}}^x\right)_\lambda = \lambda x.(\mathcal{D})_\lambda \qquad \left(\dfrac{\begin{array}{c}\vdots\ \mathcal{D}\\ \Gamma \vdash A \Rightarrow B\end{array} \quad \begin{array}{c}\vdots\ \mathcal{D}'\\ \Gamma \vdash A\end{array}}{\Gamma \vdash B} \Rightarrow_{\mathsf{e}}\right)_\lambda = (\mathcal{D})_\lambda(\mathcal{D}')_\lambda$$

Rmk. The variable labeling the rules ax and $\Rightarrow_{\mathsf{i}}$ is crucial to uniquely determine $(\mathcal{D})_\lambda$.

Rmk. If $\mathcal{D}$ derives $\Gamma \vdash A$ in ND, then $\Gamma \vdash (\mathcal{D})_\lambda : A$ is derivable in STLC.

Proposition (Uniqueness of type and derivation for well-typed terms)

In STLC, if $\mathcal{D}$ derives $\Gamma \vdash t : A$ and $\mathcal{D}'$ derives $\Gamma \vdash t : A'$, then $A = A'$ and $\mathcal{D} = \mathcal{D}'$.

Proof. By structural induction on t (exercise!). □

Theorem (Bijection between ND and Church-style STLC)

For every environment Γ *and type* A, *the map* $(\cdot)_\lambda$ *defines a bijection from derivations of* $\Gamma \vdash A$ *in* ND *and well-typed* λ*-terms of type* A *and environment* Γ *in* STLC.

Proof. Use Proposition and the second Remark above. □

We proved $()\cdot_\lambda$ is a bijection between well-typed terms in STLC and derivations in ND.
$\rightsquigarrow$ Well-typed λ-terms in STLC are proof-terms, that is, concise (and linear) representations of derivations (trees) in ND (a *static* correspondence).

The Curry-Howard Correspondence between ND and STLC

We proved $(\cdot)_\lambda$ is a bijection between well-typed terms in STLC and derivations in ND.
$\rightsquigarrow$ Well-typed λ-terms in STLC are proof-terms, that is, concise (and linear) representations of derivations (trees) in ND (a *static* correspondence).

The bijection lifts to a *dynamic* correspondence! $\rightsquigarrow$ As β-reduction and cut-elimination mimic each other, it is an isomorphism between STLC and ND.

The Curry-Howard Correspondence between ND and STLC

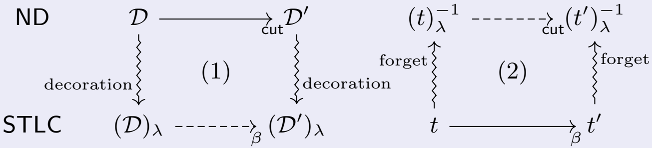
We proved $()\cdot_\lambda$ is a bijection between well-typed terms in STLC and derivations in ND.

$\rightsquigarrow$ Well-typed λ-terms in STLC are proof-terms, that is, concise (and linear) representations of derivations (trees) in ND (a *static* correspondence).

The bijection lifts to a *dynamic* correspondence! $\rightsquigarrow$ As β-reduction and cut-elimination mimic each other, it is an isomorphism between STLC and ND.

Theorem (Curry-Howard correspondence)

1. *Let* $\mathcal{D}.\mathcal{D}'$ *be derivations of* $\Gamma \vdash A$ *in* ND. *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* $(t)_\lambda^{-1} \to_{\mathsf{cut}} (t')_\lambda^{-1}$.



An example of cut-elimination step in ND, where $\Gamma = z : A,\ y : A \Rightarrow B,\ x : A \Rightarrow B \Rightarrow A$.

$$\mathcal{D} = \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B \Rightarrow A}\,\mathsf{ax}^x \quad \dfrac{}{\Gamma \vdash A}\,\mathsf{ax}^z}{\Gamma \vdash B \Rightarrow A}\,{\Rightarrow_\mathsf{e}} \qquad \dfrac{\dfrac{}{\Gamma \vdash A \Rightarrow B}\,\mathsf{ax}^y \quad \dfrac{}{\Gamma \vdash A}\,\mathsf{ax}^z}{\Gamma \vdash B}\,{\Rightarrow_\mathsf{e}}}{\Gamma \vdash A}\,{\Rightarrow_\mathsf{e}}}{x : A \Rightarrow B,\ y : A \Rightarrow (B \Rightarrow A) \vdash A \Rightarrow A}\,{\Rightarrow_\mathsf{i}^z}}{x : A \Rightarrow (B \Rightarrow A) \vdash (A \Rightarrow B) \Rightarrow (A \Rightarrow A)}\,{\Rightarrow_\mathsf{i}^y}}{\vdash (A \Rightarrow (B \Rightarrow A)) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow A)}\,{\Rightarrow_\mathsf{i}^x} \qquad \dfrac{\dfrac{\dfrac{}{a : A,\ b : B \vdash A}\,\mathsf{ax}^a}{a : A \vdash B \Rightarrow A}\,{\Rightarrow_\mathsf{i}^b}}{\vdash A \Rightarrow B \Rightarrow A}\,{\Rightarrow_\mathsf{i}^a}}{\vdash (A \Rightarrow B) \Rightarrow A \Rightarrow A}\,{\Rightarrow_\mathsf{e}}$$

$$\downarrow_{\mathsf{cut}}$$

$$\mathcal{D}' = \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Delta, a : A,\ b : B \vdash A}\,\mathsf{ax}^a}{\Delta, a : A \vdash B \Rightarrow A}\,{\Rightarrow_\mathsf{i}^b}}{\Delta \vdash A \Rightarrow B \Rightarrow A}\,{\Rightarrow_\mathsf{i}^a} \quad \dfrac{}{\Delta \vdash A}\,\mathsf{ax}^z}{\Delta \vdash B \Rightarrow A}\,{\Rightarrow_\mathsf{e}} \qquad \dfrac{\dfrac{}{\Delta \vdash A \Rightarrow B}\,\mathsf{ax}^y \quad \dfrac{}{\Delta \vdash A}\,\mathsf{ax}^z}{\Delta \vdash B}\,{\Rightarrow_\mathsf{e}}}{\Delta \vdash A}\,{\Rightarrow_\mathsf{e}}}{y : A \Rightarrow B \vdash A \Rightarrow A}\,{\Rightarrow_\mathsf{i}^z}}{\vdash (A \Rightarrow B) \Rightarrow (A \Rightarrow A)}\,{\Rightarrow_\mathsf{i}^y}$$

where $\Delta = z : A,\ y : A \Rightarrow B$.

Observe that $(\mathcal{D})_\lambda = (\lambda x.\lambda y.\lambda z.xz(yz))\lambda a.\lambda b.\lambda a.a \rightarrow_\beta \lambda y.\lambda z.(\lambda a.\lambda b.a)z(yz) = (\mathcal{D}')_\lambda$.

Two examples of derivation in STLC, where $\Gamma = z : A,\ y : A \Rightarrow B,\ x : A \Rightarrow B \Rightarrow A$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash x : A \Rightarrow B \Rightarrow A}\,\mathsf{ax}^x \quad \dfrac{}{\Gamma \vdash z : A}\,\mathsf{ax}^z}{\Gamma \vdash xz : B \Rightarrow A}\,\Rightarrow_\mathsf{e} \qquad \dfrac{\dfrac{}{\Gamma \vdash y : A \Rightarrow B}\,\mathsf{ax}^y \quad \dfrac{}{\Gamma \vdash z : A}\,\mathsf{ax}^z}{\Gamma \vdash yz : B}\,\Rightarrow_\mathsf{e}}{\Gamma \vdash xz(yz) : A}\,\Rightarrow_\mathsf{e}}{x : A \Rightarrow B,\ y : A \Rightarrow B \Rightarrow A \vdash \lambda z.xz(yz) : A \Rightarrow A}\,\Rightarrow_\mathsf{i}^z}{x : A \Rightarrow B \Rightarrow A \vdash \lambda y.\lambda z.xz(yz) : (A \Rightarrow B) \Rightarrow (A \Rightarrow A)}\,\Rightarrow_\mathsf{i}^y}{\vdash \lambda x.\lambda y.\lambda z.xz(yz) : (A \Rightarrow B \Rightarrow A) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow A)}\,\Rightarrow_\mathsf{i}^x \quad \dfrac{\dfrac{\dfrac{}{a : A,\ b : B \vdash a : A}\,\mathsf{ax}^a}{a : A \vdash \lambda b.a : B \Rightarrow A}\,\Rightarrow_\mathsf{i}^b}{\vdash \lambda a.\lambda b.\lambda a.a : A \Rightarrow B \Rightarrow A}\,\Rightarrow_\mathsf{i}^a}{\vdash (\lambda x.\lambda y.\lambda z.xz(yz))\lambda a.\lambda b.\lambda a.a : (A \Rightarrow B) \Rightarrow A \Rightarrow A}\,\Rightarrow_\mathsf{e}$$

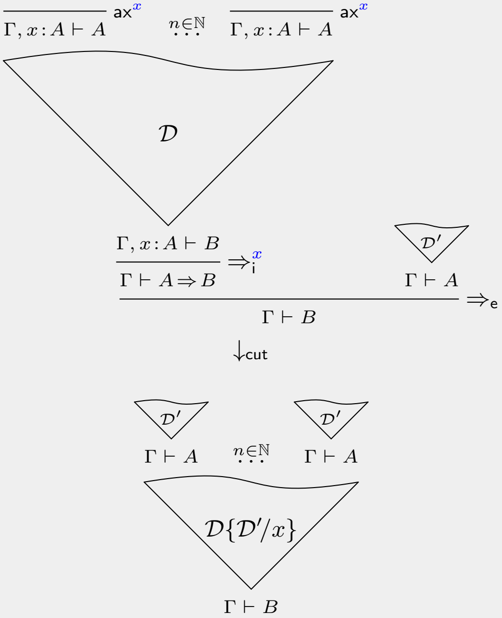
(Types on the abstracted variables are omitted for the sake of readability.)

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Delta, a : A,\ b : a : B \vdash A}\,\mathsf{ax}^a}{\Delta, a : A \vdash \lambda b.a : B \Rightarrow A}\,\Rightarrow_\mathsf{i}^b}{\Delta \vdash \lambda a.\lambda b.a : A \Rightarrow B \Rightarrow A}\,\Rightarrow_\mathsf{i}^a \quad \dfrac{}{\Delta \vdash z : A}\,\mathsf{ax}^z}{\Delta \vdash (\lambda a.\lambda b.a)z : B \Rightarrow A}\,\Rightarrow_\mathsf{e} \qquad \dfrac{\dfrac{}{\Delta \vdash y : A \Rightarrow B}\,\mathsf{ax}^y \quad \dfrac{}{\Delta \vdash z : A}\,\mathsf{ax}^z}{\Delta \vdash yz : B}\,\Rightarrow_\mathsf{e}}{\Delta \vdash (\lambda a.\lambda b.a)z(yz) : A}\,\Rightarrow_\mathsf{e}}{y : A \Rightarrow B \vdash \lambda z.(\lambda a.\lambda b.a)z(yz) : A \Rightarrow A}\,\Rightarrow_\mathsf{i}^z}{\vdash \lambda y.\lambda z.(\lambda a.\lambda b.a)z(yz) : (A \Rightarrow B) \Rightarrow (A \Rightarrow A)}\,\Rightarrow_\mathsf{i}^y$$

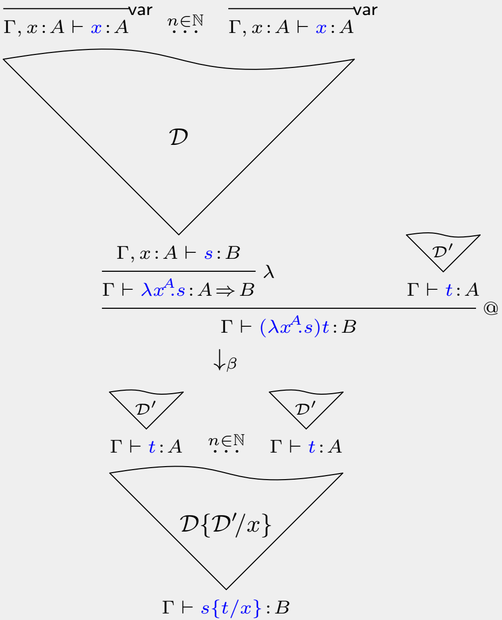
where $\Delta = z : A,\ y : A \Rightarrow B$.

Observe that $(\mathcal{D})_\lambda = (\lambda x.\lambda y.\lambda z.xz(yz))\lambda a.\lambda b.\lambda a.a \rightarrow_\beta \lambda y.\lambda z.(\lambda a.\lambda b.a)z(yz) = (\mathcal{D}')_\lambda$.

The Curry-Howard Correspondence between ND and STLC



The Curry-Howard Correspondence between ND and STLC



The Curry-Howard Correspondence between ND and STLC

ND and STLC are strictly related!

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
provability	inhabitation	$\exists$ program meeting spec.

The Curry-Howard Correspondence between ND and STLC

ND and STLC are strictly related!

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
provability	inhabitation	$\exists$ program meeting spec.

Concerning the correspondence between derivations and terms:

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

Cartesian Closed Categories strike back!

The simply typed λ-calculus can be interpreted in *any* CCC (see Day 3 for its definition).

- Simple types (that is, formulas of minimal logic) are interpreted by objects.
- Well-typed terms in STLC (i.e., derivations in ND) are interpreted by morphisms.

The simply typed λ-calculus can be interpreted in *any* CCC (see Day 3 for its definition).

- Simple types (that is, formulas of minimal logic) are interpreted by objects.
- Well-typed terms in STLC (i.e., derivations in ND) are interpreted by morphisms.

Definition (Categorical semantics/interpretation of well-typed λ-terms)

Let $\boldsymbol{C}$ be a CCC. The *interpretation* $[\![A]\!]$ of a type A in $\boldsymbol{C}$ is an object defined by:

$$[\![X]\!] = \text{an arbitrary object of } \boldsymbol{C} \qquad [\![A \Rightarrow B]\!] = [\![A]\!] \Rightarrow [\![B]\!].$$

Let $\Gamma \vdash t : B$ be derivable in STLC with $\Gamma = x_1 : A_1, \ldots x_n : A_n$ and $\vec{x} = (x_1, \ldots, x_n)$.
The categorical semantics of $\Gamma \vdash t : B$ wrt $\vec{x}$ in $\boldsymbol{C}$ is a morphism
$[\![\Gamma \vdash t : B]\!]_{\vec{x}} \colon [\![A_1]\!] \times \cdots \times [\![A_n]\!] \to [\![B]\!]$ defined by:

$$
\begin{aligned}
[\![\Gamma \vdash x_i : A_i]\!]_{\vec{x}} &= \pi_i && \text{where } i \in \{1, \ldots, n\} \\
[\![\Gamma \vdash st : B]\!]_{\vec{x}} &= \mathrm{ev}_{A,B} \circ \langle [\![\Gamma \vdash s : A \Rightarrow B]\!]_{\vec{x}}, [\![\Gamma \vdash t : A]\!]_{\vec{x}} \rangle && \\
[\![\Gamma \vdash \lambda y.t : A \Rightarrow B]\!]_{\vec{x}} &= \mathrm{curry}([\![\Gamma, y : A \vdash t : B]\!]_{\vec{x},y}) && \text{we assume wlog } y \notin \{x_1, \ldots, x_n\}.
\end{aligned}
$$

The simply typed λ-calculus can be interpreted in *any* CCC (see Day 3 for its definition).

- Simple types (that is, formulas of minimal logic) are interpreted by objects.
- Well-typed terms in STLC (i.e., derivations in ND) are interpreted by morphisms.

Definition (Categorical semantics/interpretation of well-typed λ-terms)

Let $\boldsymbol{C}$ be a CCC. The *interpretation* $[\![A]\!]$ of a type A in $\boldsymbol{C}$ is an object defined by:

$$[\![X]\!] = \text{an arbitrary object of } \boldsymbol{C} \qquad [\![A \Rightarrow B]\!] = [\![A]\!] \Rightarrow [\![B]\!].$$

Let $\Gamma \vdash t : B$ be derivable in STLC with $\Gamma = x_1 : A_1, \ldots x_n : A_n$ and $\vec{x} = (x_1, \ldots, x_n)$. The categorical semantics of $\Gamma \vdash t : B$ wrt $\vec{x}$ in $\boldsymbol{C}$ is a morphism $[\![\Gamma \vdash t : B]\!]_{\vec{x}} : [\![A_1]\!] \times \cdots \times [\![A_n]\!] \to [\![B]\!]$ defined by:

$$[\![\Gamma \vdash x_i : A_i]\!]_{\vec{x}} = \pi_i \qquad \text{where } i \in \{1, \ldots, n\}$$

$$[\![\Gamma \vdash st : B]\!]_{\vec{x}} = \mathrm{ev}_{A,B} \circ \langle [\![\Gamma \vdash s : A \Rightarrow B]\!]_{\vec{x}}, [\![\Gamma \vdash t : A]\!]_{\vec{x}} \rangle$$

$$[\![\Gamma \vdash \lambda y.t : A \Rightarrow B]\!]_{\vec{x}} = \mathrm{curry}([\![\Gamma, y : A \vdash t : B]\!]_{\vec{x},y}) \qquad \text{we assume wlog } y \notin \{x_1, \ldots, x_n\}.$$

Rmk: Formally, the definition of semantics is for *derivations*, not for their conclusion, and by induction on derivations. By uniqueness of the derivation (Proposition on p. 18) there is no ambiguity if we only write the conclusion of the derivation to interpret.

Lemma (Substitution)

Let $\Gamma, x\!:\!A \vdash s\!:\!B$ *and* $\Gamma \vdash t\!:\!A$ *be derivable in* STLC *with* $\vec{y} = \mathsf{dom}(\Gamma)$. *Then,*
$[\![\Gamma \vdash s\{t/x\} : B]\!]_{\vec{y}} = [\![\Gamma, x\!:\!A \vdash s\!:\!B]\!]_{\vec{y},x} \circ \langle \mathrm{id}_{[\![\Gamma]\!]}, [\![\Gamma \vdash s\!:\!A\ ]\!]_{\vec{y}} \rangle$.

Proof. By induction on s. Exercise!

Lemma (Substitution)

Let $\Gamma, x : A \vdash s : B$ *and* $\Gamma \vdash t : A$ *be derivable in* STLC *with* $\vec{y} = \mathsf{dom}(\Gamma)$*. Then,*
$[\![\Gamma \vdash s\{t/x\} : B]\!]_{\vec{y}} = [\![\Gamma, x : A \vdash s : B]\!]_{\vec{y},x} \circ \langle \mathrm{id}_{[\![\Gamma]\!]}, [\![\Gamma \vdash s : A\ ]\!]_{\vec{y}} \rangle$.

Proof. By induction on s. Exercise!

Theorem (Invariance/Soundness)

Let $\Gamma \vdash t : B$ *and* $\Gamma \vdash t' : B$ *be derivable in* STLC *with* $\vec{y} = \mathsf{dom}(\Gamma)$*. If* $t \to_\beta t'$ *then*
$[\![\Gamma \vdash t : B]\!]_{\vec{y}} = [\![\Gamma \vdash t' : B]\!]_{\vec{y}}$.

Lemma (Substitution)

Let $\Gamma, x\!:\!A \vdash s\!:\!B$ *and* $\Gamma \vdash t\!:\!A$ *be derivable in* STLC *with* $\vec{y} = \mathsf{dom}(\Gamma)$*. Then,*
$[\![\Gamma \vdash s\{t/x\} : B]\!]_{\vec{y}} = [\![\Gamma, x\!:\!A \vdash s\!:\!B]\!]_{\vec{y},x} \circ \langle \mathrm{id}_{[\![\Gamma]\!]}, [\![\Gamma \vdash s\!:\!A\ ]\!]_{\vec{y}} \rangle$.

Proof. By induction on s. Exercise!

Theorem (Invariance/Soundness)

Let $\Gamma \vdash t\!:\!B$ *and* $\Gamma \vdash t'\!:\!B$ *be derivable in* STLC *with* $\vec{y} = \mathsf{dom}(\Gamma)$*. If* $t \to_\beta t'$ *then*
$[\![\Gamma \vdash t\!:\!B]\!]_{\vec{y}} = [\![\Gamma \vdash t'\!:\!B]\!]_{\vec{y}}$.

Proof. The key case is $t = (\lambda x^A.t_1)t_2 \to_\beta t_1\{t_2/x\} = t'$. We assume wlog $x \notin \vec{y}$.

$$
\begin{aligned}
[\![\Gamma \vdash t\!:\!B]\!]_{\vec{y}} &= [\![\Gamma \vdash (\lambda x^A.t_1)t_2 : B]\!]_{\vec{y}} \\
&= \mathrm{ev}_{A,B} \circ \langle [\![\Gamma \vdash \lambda x^A.t_1 : A \Rightarrow B]\!]_{\vec{y}},\ [\![\Gamma \vdash t_2 : A]\!]_{\vec{y}} \rangle && \text{(def. of } [\![\cdot]\!]) \\
&= \mathrm{ev}_{A,B} \circ \langle \mathrm{curry}([\![\Gamma, x\!:\!A \vdash t_1 : B]\!]_{\vec{y},x}),\ [\![\Gamma \vdash t_2 : A]\!]_{\vec{y}} \rangle && \text{(def. of } [\![\cdot]\!]) \\
&= [\![\Gamma, x\!:\!A \vdash t_1 : B]\!]_{\vec{y},x} \circ \langle \mathrm{id}_{[\![\Gamma]\!]},\ [\![\Gamma \vdash t_2 : A]\!]_{\vec{y}} \rangle && \text{(rule } \beta_e) \\
&= [\![\Gamma \vdash t_1\{t_2/x\} : B]\!]_{\vec{y}} = [\![\Gamma \vdash t' : B]\!]_{\vec{y}} && \text{(substitution)}
\end{aligned}
$$

The other cases follow from the IH (proof by induction on the defn. of $t \to_\beta t'$). □

Lemma (Substitution)

Let $\Gamma, x\!:\!A \vdash s\!:\!B$ *and* $\Gamma \vdash t\!:\!A$ *be derivable in* STLC *with* $\vec{y} = \mathsf{dom}(\Gamma)$*. Then,* $[\![\Gamma \vdash s\{t/x\} : B]\!]_{\vec{y}} = [\![\Gamma, x\!:\!A \vdash s\!:\!B]\!]_{\vec{y},x} \circ \langle \mathrm{id}_{[\![\Gamma]\!]}, [\![\Gamma \vdash s\!:\!A\ ]\!]_{\vec{y}} \rangle$.

Proof. By induction on s. Exercise!

Theorem (Invariance/Soundness)

Let $\Gamma \vdash t\!:\!B$ *and* $\Gamma \vdash t'\!:\!B$ *be derivable in* STLC *with* $\vec{y} = \mathsf{dom}(\Gamma)$*. If* $t \to_\beta t'$ *then* $[\![\Gamma \vdash t\!:\!B]\!]_{\vec{y}} = [\![\Gamma \vdash t'\!:\!B]\!]_{\vec{y}}$.

Proof. The key case is $t = (\lambda x^A.t_1)t_2 \to_\beta t_1\{t_2/x\} = t'$. We assume wlog $x \notin \vec{y}$.

$$
\begin{aligned}
[\![\Gamma \vdash t\!:\!B]\!]_{\vec{y}} &= [\![\Gamma \vdash (\lambda x^A.t_1)t_2\!:\!B]\!]_{\vec{y}} \\
&= \mathrm{ev}_{A,B} \circ \langle [\![\Gamma \vdash \lambda x^A.t_1 : A \Rightarrow B]\!]_{\vec{y}},\ [\![\Gamma \vdash t_2 : A]\!]_{\vec{y}} \rangle && \text{(def. of } [\![\cdot]\!]) \\
&= \mathrm{ev}_{A,B} \circ \langle \mathrm{curry}([\![\Gamma, x\!:\!A \vdash t_1 : B]\!]_{\vec{y},x}),\ [\![\Gamma \vdash t_2 : A]\!]_{\vec{y}} \rangle && \text{(def. of } [\![\cdot]\!]) \\
&= [\![\Gamma, x\!:\!A \vdash t_1 : B]\!]_{\vec{y},x} \circ \langle \mathrm{id}_{[\![\Gamma]\!]},\ [\![\Gamma \vdash t_2 : A]\!]_{\vec{y}} \rangle && \text{(rule } \beta_e) \\
&= [\![\Gamma \vdash t_1\{t_2/x\} : B]\!]_{\vec{y}} = [\![\Gamma \vdash t' : B]\!]_{\vec{y}} && \text{(substitution)}
\end{aligned}
$$

The other cases follow from the IH (proof by induction on the defin. of $t \to_\beta t'$). □

Even contextuality holds. Consistency depends on the specific CCC.

It can be proved that the **STLC** is the *internal language* of a CCC $\rightsquigarrow$ a deep link.

Cartesian Closed Categories strike back!

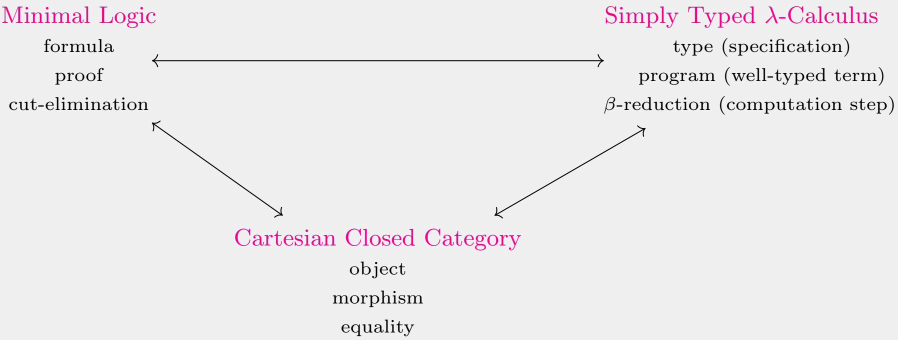
It can be proved that the STLC is the *internal language* of a CCC $\rightsquigarrow$ a deep link.

The Curry-Howard correspondence: a link between logic, programming.

Minimal Logic		Simply Typed λ-Calculus
formula	$\longleftrightarrow$	type (specification)
proof		program (well-typed term)
cut-elimination		β-reduction (computation step)

It can be proved that the **STLC** is the *internal language* of a CCC ⇝ a deep link.

The Curry-Howard-*Lambek* correspondence: a link between logic, programming, maths



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

Given a reduction $\to$ on a set A, we aim to prove that $\to$ is strongly normalizing (SN):
$\rightsquigarrow$ 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 any $t \in A$, 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 $|t| > |s|$.

Given a reduction $\to$ on a set A, we aim to prove that $\to$ is strongly normalizing (SN):
$\rightsquigarrow$ 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 any $t \in A$, 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.

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

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

Given a reduction $\to$ on a set A, we aim to prove that $\to$ is strongly normalizing (SN):
$\leadsto$ 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 any $t \in A$, 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.
$\leadsto$ After a single β-step the size ($\approx$ number of characters) of a term may not decrease.

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

$\leadsto$ The measure should be defined independently of/cannot rely on the size of terms.

Question: How to prove SN for STLC using a non-combinatorial approach?
Answer: Use the reducibility candidates method.

Strong Normalization for the Simply Typed λ-Calculus

Idea: 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 SN terms of type X;
- $\mathsf{Red}_{A \Rightarrow B}$ is the set of the terms $s : A \Rightarrow B$ such that $st \in \mathsf{Red}_B$ for all $t \in \mathsf{Red}_A$.

Idea: 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 SN terms of type X;
- $\mathsf{Red}_{A \Rightarrow B}$ is the set of the terms $s : A \Rightarrow B$ such that $st \in \mathsf{Red}_B$ for all $t \in \mathsf{Red}_A$.

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

Idea: 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 SN terms of type X;
- $\mathsf{Red}_{A \Rightarrow B}$ is the set of the terms $s : A \Rightarrow B$ such that $st \in \mathsf{Red}_B$ for all $t \in \mathsf{Red}_A$.

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 $u \in \mathsf{Red}_A$ (so u is SN). Proof by induction on u.

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

Idea: 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 SN terms of type X;
- $\mathsf{Red}_{A \Rightarrow B}$ is the set of the terms $s : A \Rightarrow B$ such that $st \in \mathsf{Red}_B$ for all $t \in \mathsf{Red}_A$.

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 $u \in \mathsf{Red}_A$ (so u is SN). Proof by induction on u.

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

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

Strong Normalization for the Simply Typed λ-Calculus

Idea: 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 SN terms of type X;
- $\mathsf{Red}_{A \Rightarrow B}$ is the set of the terms $s : A \Rightarrow B$ such that $st \in \mathsf{Red}_B$ for all $t \in \mathsf{Red}_A$.

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 $u \in \mathsf{Red}_A$ (so u is SN). Proof by induction on u.

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

Idea: Suppose $\lambda x^B.s : B \Rightarrow C$ and $t \in \mathsf{Red}_B$. Let us prove that $s\{t/x\} \in \mathsf{Red}_C$ and that if $s\{t/x\} \in \mathsf{Red}_C$ then $(\lambda x^B.s)t \in \mathsf{Red}_C$. 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.

Solution: Take the environment into account when defining Red_A, for all types A.

$$\mathsf{Red}_X = \{\langle \Gamma; t \rangle \mid t \text{ is SN},\ \Gamma \vdash t : X\}$$

$$\mathsf{Red}_{A \Rightarrow B} = \{\langle \Gamma; s \rangle \mid \Gamma \vdash s : A \Rightarrow B,\ \langle \Gamma, \Delta;\, st \rangle \in \mathsf{Red}_B \text{ for all } \langle \Gamma, \Delta;\, t \rangle \in \mathsf{Red}_A\}$$

Solution: Take the environment into account when defining Red_A, for all types A.

$$\mathsf{Red}_X = \{\langle\Gamma; t\rangle \mid t \text{ is SN},\ \Gamma \vdash t : X\}$$

$$\mathsf{Red}_{A \Rightarrow B} = \{\langle\Gamma; s\rangle \mid \Gamma \vdash s : A \Rightarrow B,\ \langle\Gamma, \Delta; st\rangle \in \mathsf{Red}_B \text{ for all } \langle\Gamma, \Delta; t\rangle \in \mathsf{Red}_A\}$$

Lemma

1. If $\langle\Gamma; t\rangle \in \mathsf{Red}_B$ then t is SN.
2. If $\Gamma \vdash x t_1 \ldots t_n : B$ is derivable and $t_1, \ldots, t_n$ are SN, then $\langle\Gamma; x t_1 \ldots t_n\rangle \in \mathsf{Red}_B$.
3. (Closure under β-expansion) If $\langle\Gamma; s\{t/x\} t_1 \ldots t_n\rangle \in \mathsf{Red}_B$, $\Gamma \vdash t : A$ is derivable and t is SN, then $\langle\Gamma; (\lambda x^A\!.s) t t_1 \ldots t_n\rangle \in \mathsf{Red}_B$.

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, Points 2–3 are a good exercise! Let $B = C \Rightarrow D$.

Solution: Take the environment into account when defining Red_A, for all types A.

$$\mathsf{Red}_X = \{\langle\Gamma; t\rangle \mid t \text{ is SN}, \ \Gamma \vdash t : X\}$$

$$\mathsf{Red}_{A\Rightarrow B} = \{\langle\Gamma; s\rangle \mid \Gamma \vdash s : A \Rightarrow B, \ \langle\Gamma, \Delta; st\rangle \in \mathsf{Red}_B \text{ for all } \langle\Gamma, \Delta; t\rangle \in \mathsf{Red}_A\}$$

Lemma

1. If $\langle\Gamma; t\rangle \in \mathsf{Red}_B$ then t is SN.
2. If $\Gamma \vdash xt_1 \ldots t_n : B$ is derivable and $t_1, \ldots, t_n$ are SN, then $\langle\Gamma; xt_1 \ldots t_n\rangle \in \mathsf{Red}_B$.
3. (Closure under β-expansion) If $\langle\Gamma; s\{t/x\}t_1 \ldots t_n\rangle \in \mathsf{Red}_B$, $\Gamma \vdash t : A$ is derivable and t is SN, then $\langle\Gamma; (\lambda x^A. s)tt_1 \ldots t_n\rangle \in \mathsf{Red}_B$.

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, Points 2–3 are a good exercise! Let $B = C \Rightarrow D$.

1. Let $z \notin \mathsf{dom}(\Gamma)$, so $\langle\Gamma, z : C; z\rangle \in \mathsf{Red}_C$ by the induction hypothesis of Point 2 applied to C. As $\langle\Gamma; t\rangle \in \mathsf{Red}_{C\Rightarrow D}$, then $\langle\Gamma, z : C; tz\rangle \in \mathsf{Red}_D$ and hence tz is SN by the induction hypothesis of Point 1 applied to D; thus t is SN too.

Solution: Take the environment into account when defining Red_A, for all types A.

$$\mathsf{Red}_X = \{\langle\Gamma; t\rangle \mid t \text{ is SN},\ \Gamma \vdash t : X\}$$

$$\mathsf{Red}_{A\Rightarrow B} = \{\langle\Gamma; s\rangle \mid \Gamma \vdash s : A \Rightarrow B,\ \langle\Gamma, \Delta; st\rangle \in \mathsf{Red}_B \text{ for all } \langle\Gamma, \Delta; t\rangle \in \mathsf{Red}_A\}$$

Lemma

1. If $\langle\Gamma; t\rangle \in \mathsf{Red}_B$ then t is SN.
2. If $\Gamma \vdash xt_1 \dots t_n : B$ is derivable and $t_1, \dots, t_n$ are SN, then $\langle\Gamma; xt_1 \dots t_n\rangle \in \mathsf{Red}_B$.
3. (Closure under β-expansion) If $\langle\Gamma; s\{t/x\}t_1 \dots t_n\rangle \in \mathsf{Red}_B$, $\Gamma \vdash t : A$ is derivable and t is SN, then $\langle\Gamma; (\lambda x^A.s)tt_1 \dots t_n\rangle \in \mathsf{Red}_B$.

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, Points 2–3 are a good exercise! Let $B = C \Rightarrow D$.

1. Let $z \notin \mathsf{dom}(\Gamma)$, so $\langle\Gamma, z{:}C;\ z\rangle \in \mathsf{Red}_C$ by the induction hypothesis of Point 2 applied to C. As $\langle\Gamma; t\rangle \in \mathsf{Red}_{C\Rightarrow D}$, then $\langle\Gamma, z{:}C; tz\rangle \in \mathsf{Red}_D$ and hence tz is SN by the induction hypothesis of Point 1 applied to D; thus t is SN too.
2. Let $\langle\Gamma, \Delta;\ t\rangle \in \mathsf{Red}_C$, so t is SN by induction hypothesis of Point 1 applied to C; as $\Gamma, \Delta \vdash xt_1 \dots t_n t : D$ is derivable, $\langle\Gamma, \Delta; xt_1 \dots t_n t\rangle \in \mathsf{Red}_D$ by induction hypothesis of Point 2 applied to D; so $\langle\Gamma; xt_1 \dots t_n\rangle \in \mathsf{Red}_B$ by defin. of $\mathsf{Red}_{C\Rightarrow D}$.

Solution: Take the environment into account when defining Red_A, for all types A.

$$\mathsf{Red}_X = \{\langle \Gamma; t\rangle \mid t \text{ is SN},\ \Gamma \vdash t : X\}$$

$$\mathsf{Red}_{A\Rightarrow B} = \{\langle \Gamma; s\rangle \mid \Gamma \vdash s : A \Rightarrow B,\ \langle \Gamma, \Delta;\ st\rangle \in \mathsf{Red}_B \text{ for all } \langle \Gamma, \Delta;\ t\rangle \in \mathsf{Red}_A\}$$

Lemma

1. If $\langle \Gamma; t\rangle \in \mathsf{Red}_B$ then t is SN.
2. If $\Gamma \vdash x t_1 \ldots t_n : B$ is derivable and $t_1, \ldots, t_n$ are SN, then $\langle \Gamma; x t_1 \ldots t_n\rangle \in \mathsf{Red}_B$.
3. (Closure under β-expansion) If $\langle \Gamma; s\{t/x\} t_1 \ldots t_n\rangle \in \mathsf{Red}_B$, $\Gamma \vdash t : A$ is derivable and t is SN, then $\langle \Gamma; (\lambda x^A. s) t t_1 \ldots t_n\rangle \in \mathsf{Red}_B$.

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, Points 2–3 are a good exercise! Let $B = C \Rightarrow D$.

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

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.

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

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

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

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

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

Every well-typed term in the simply typed λ-calculus is SN.

Proof. Let $x_1 : B_1, \ldots, x_n : B_n \vdash t : A$ be derivable. Let $\Gamma = x_1 : B_1, \ldots, x_n : B_n$ and $s_i = x_i$ for all $1 \leq i \leq n$, hence $\langle \Gamma; s_i \rangle \in \mathsf{Red}_{B_i}$ by Point 2 of the lemma on p. 29 (as $\Gamma \vdash x_i : B_i$ is derivable), for all $1 \leq i \leq n$. By the substitution lemma above, $\langle \Gamma; t \rangle = \langle \Gamma; t\{s_1/x_1, \ldots, s_n/x_n\} \rangle \in \mathsf{Red}_A$. By Point 1 of the lemma on p. 29, t is SN. □

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

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

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

Every well-typed term in the simply typed λ-calculus is SN.

Proof. Let $x_1 : B_1, \ldots, x_n : B_n \vdash t : A$ be derivable. Let $\Gamma = x_1 : B_1, \ldots, x_n : B_n$ and $s_i = x_i$ for all $1 \leq i \leq n$, hence $\langle \Gamma; s_i \rangle \in \mathsf{Red}_{B_i}$ by Point 2 of the lemma on p. 29 (as $\Gamma \vdash x_i : B_i$ is derivable), for all $1 \leq i \leq n$. By the substitution lemma above, $\langle \Gamma; t \rangle = \langle \Gamma; t\{s_1/x_1, \ldots, s_n/x_n\} \rangle \in \mathsf{Red}_A$. By Point 1 of the lemma on p. 29, t is SN. □

Moral: It does not matter the order in which β-redexes are fired in a well-typed term of STLC, it will eventually lead to a normal form (the same result by confluence).

By Curry-Howard, normalization of STLC can be seen as a cut-elimination theorem in ND $\rightsquigarrow$ Let us see some proof-theoretic consequences in ND.

Strong Normalization for the Simply Typed λ-Calculus

By Curry-Howard, normalization of STLC can be seen as a cut-elimination theorem in ND $\rightsquigarrow$ Let us see some proof-theoretic consequences in ND.

Rmk: If $\mathcal{D}$ proves $\Gamma \vdash A$ in ND without detours (a detour is a formula occurrence being conclusion of $\Rightarrow_{\mathsf{i}}$ and left premise of $\Rightarrow_{\mathsf{e}}$), then $\mathcal{D}$ only contains subformulas of Γ or A.

By Curry-Howard, normalization of STLC can be seen as a cut-elimination theorem in ND $\rightsquigarrow$ Let us see some proof-theoretic consequences in ND.

Rmk: If $\mathcal{D}$ proves $\Gamma \vdash A$ in ND without detours (a detour is a formula occurrence being conclusion of $\Rightarrow_\mathsf{i}$ and left premise of $\Rightarrow_\mathsf{e}$), then $\mathcal{D}$ only contains subformulas of Γ or A.

Corollary (Subformula property)

If $\Gamma \vdash A$ is provable in ND, then there is a derivation $\mathcal{D}$ of $\Gamma \vdash A$ only containing subformulas of Γ or A.

Proof. By cut-elimination, there is $\mathcal{D}$ with no detours. Rmk. above concludes. □

Moral: When searching for a derivation of $\Gamma \vdash A$, just look at the subformulas of Γ, A.

By Curry-Howard, normalization of STLC can be seen as a cut-elimination theorem in ND $\rightsquigarrow$ Let us see some proof-theoretic consequences in ND.

Rmk: If $\mathcal{D}$ proves $\Gamma \vdash A$ in ND without detours (a detour is a formula occurrence being conclusion of $\Rightarrow_{\mathsf{i}}$ and left premise of $\Rightarrow_{\mathsf{e}}$), then $\mathcal{D}$ only contains subformulas of Γ or A.

Corollary (Subformula property)

If $\Gamma \vdash A$ is provable in ND, then there is a derivation $\mathcal{D}$ of $\Gamma \vdash A$ only containing subformulas of Γ or A.

Proof. By cut-elimination, there is $\mathcal{D}$ with no detours. Rmk. above concludes. □

Moral: When searching for a derivation of $\Gamma \vdash A$, just look at the subformulas of Γ, A.

Corollary (Consistency of ND)

Some sequents (e.g. all ground types without hypotheses) are not provable in ND.

By Curry-Howard, normalization of STLC can be seen as a cut-elimination theorem in ND $\rightsquigarrow$ Let us see some proof-theoretic consequences in ND.

Rmk: If $\mathcal{D}$ proves $\Gamma \vdash A$ in ND without detours (a detour is a formula occurrence being conclusion of $\Rightarrow_{\mathsf{i}}$ and left premise of $\Rightarrow_{\mathsf{e}}$), then $\mathcal{D}$ only contains subformulas of Γ or A.

Corollary (Subformula property)

If $\Gamma \vdash A$ is provable in ND, then there is a derivation $\mathcal{D}$ of $\Gamma \vdash A$ only containing subformulas of Γ or A.

Proof. By cut-elimination, there is $\mathcal{D}$ with no detours. Rmk. above concludes. □

Moral: When searching for a derivation of $\Gamma \vdash A$, just look at the subformulas of Γ, A.

Corollary (Consistency of ND)

Some sequents (e.g. all ground types without hypotheses) are not provable in ND.

Proof. If $\vdash X$ were provable in ND, there would be a derivation $\mathcal{D}$ of $\vdash X$ with the subformula property by Coroll. above, hence the last rule of $\mathcal{D}$ could neither be $\Rightarrow_{\mathsf{e}}$ nor $\Rightarrow_{\mathsf{i}}$ (as X is not an implication) nor ax (as there are no hypotheses). □

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* $f\colon \mathbb{N}^k \to \mathbb{N}$ *for all* $k \in \mathbb{N}$ *containing the:*

- *projections* $\pi_i^k(n_1, \ldots, n_k) = n_i$ *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.

Logic and/vs Computation

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 $f\colon \mathbb{N}^k \to \mathbb{N}$ for all $k \in \mathbb{N}$ containing the:

- *projections $\pi_i^k(n_1, \ldots, n_k) = n_i$ 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 $\mathtt{true} : \mathtt{bool}$, $\mathtt{false} : \mathtt{bool}$ and $\underline{n} : \mathtt{nat}$ (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;
- 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).

Logic and/vs Computation

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 (see more in Day 5);
- dependent type theory;
- ...

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 (see more in Day 5);
- 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.

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 (see more in Day 5);
- 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.

- The simply typed λ-calculus in Church-style.
- The proof of strong normalization for the simply typed λ-calculus via reducibility candidates.
- Natural deduction for minimal logic.
- The Curry–Howard–Lambek correspondence:
  - formula = type = object in a CCC;
  - proof = program = morphism in a CCC;
  - cut-elimination = β-reduction = equality.
- Computational understanding of logic and logical understanding of computation.

- The simply typed λ-calculus in Church-style.
- The proof of strong normalization for the simply typed λ-calculus via reducibility candidates.
- Natural deduction for minimal logic.
- The Curry–Howard–Lambek correspondence:
  - formula = type = object in a CCC;
  - proof = program = morphism in a CCC;
  - cut-elimination = β-reduction = equality.
- Computational understanding of logic and logical understanding of computation.

Rmk: We presented the Curry–Howard correspondence as two *distinct* things, STLC as a programming language and ND as a proof system, that turn out to be *isomorphic*.

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

- The simply typed λ-calculus in Church-style.
- The proof of strong normalization for the simply typed λ-calculus via reducibility candidates.
- Natural deduction for minimal logic.
- The Curry–Howard–Lambek correspondence:
  - formula = type = object in a CCC;
  - proof = program = morphism in a CCC;
  - cut-elimination = β-reduction = equality.
- Computational understanding of logic and logical understanding of computation.

Rmk: We presented the Curry–Howard correspondence as two *distinct* things, STLC as a programming language and ND as a proof system, that 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 formula $A \Rightarrow B$ says "If I had an A, I could prove B".
- A program : $A \Rightarrow B$ says "If I had a value : A, I could compute a value : B".

That the underlying system can be formalized as ND or STLC is just syntactic sugar.

- Do the proofs of the statements on the slides.
- Look at our **notes** on the webpage of the course, there are plenty of **details**, **proofs** and **exercises**. Today's notes are under construction!
- The exercises will have **solutions** (but try to do them by yourself before looking at them!).
- Don't hesitate to **ask** us questions in person or on Discord about lectures, exercises, solutions, further reading.

- Chapter 4 of:
  Amadio R., Curien P-L.: Domains and lambda-calculi, 1996,
  https://www.cambridge.org/core/books/domains-and-lambdacalculi/4C6AB6938E436CFA8D5A8533B76A7F23
- Chapters 2 to 4 of:
  Sørensen M. H., Urzyczyin P.: Lectures on the Curry-Howard Isomorphism, 2006,
  https://www.sciencedirect.com/bookseries/studies-in-logic-and-the-foundations-of-mathematics/vol/149/
  (A draft is available on
  https://disi.unitn.it/~bernardi/RSISE11/Papers/curry-howard.pdf)
- Chapters 1 to 3 of:
  Barendregt H. P.: Lambda Calculi with Types. In Handbook of Logic in Computer Science, vol. 2, 1993,
  https://www.cs.rhul.ac.uk/~zhaohui/Barendregt92.pdf
- Chapters 1 to 3 and 6 of:
  Girard J. Y., Lafont Y., Taylor P.: Proof and Types, 1989,
  https://www.paultaylor.eu/stable/prot.pdf