
Lambda Calculus

Alonzo Church (1903 - 1995)

2

-calculus : syntax

Grammar for terms :

$$\begin{aligned}
 t, u ::= & \ x && \text{(variable)} && / \\
 & \ t \ u && \text{(application)} && / \\
 & \ x.t && \text{(abstraction)}
 \end{aligned}$$

Notation :

Application is left-associative so that $t_1 \ t_2 \ \dots \ t_n$ means $(\dots (t_1 \ t_2) \ \dots \ t_n)$.

$x_1 \ \dots \ x_n.t$ means $x_1.\dots \ x_n.t$.

3

Inductive definition for -terms

$$\frac{x \text{ is a variable}}{x \text{ is a -term}} \text{ (Var)}$$

$$\frac{t \text{ is a -term} \quad u \text{ is a -term}}{t \ u \text{ is a -term}} \text{ (App)} \qquad \frac{t \text{ is a -term}}{x.t \text{ is a -term}} \text{ (Lamb)}$$

4

Free and bound variables

$$x.(z\ x\ y\ (z.z\ y))$$

The set of **free** and **bound** variables are defined as follows

$$\begin{aligned} \text{fv}(x) &= \{x\} & \text{bv}(x) &= \\ \text{fv}(t\ u) &= \text{fv}(t) \cup \text{fv}(u) & \text{bv}(t\ u) &= \text{bv}(t) \cup \text{bv}(u) \\ \text{fv}(x.t) &= \text{fv}(t) \cup \{x\} & \text{bv}(x.t) &= \text{bv}(t) \cup \{x\} \end{aligned}$$

But for $t = x\ (x.x)$

$$\text{fv}(x\ (x.x)) = \{x\} = \text{bv}(x\ (x.x))$$

5

Defining the notion of fresh replacement

The operation $t[x//y]$ means the **replacement** of all the **free** occurrences of x in t by a **fresh** variable y .

$$\begin{aligned} x[x//y] &= y \\ z[x//y] &= z \\ (t\ u)[x//y] &= t[x//y]\ u[x//y] \\ (x.t)[x//y] &= x.t \\ (z.t)[x//y] &= z.t[x//y] \end{aligned}$$

7

Alpha-conversion

The relation $=_\alpha$, called **alpha-conversion**, is the **congruence** generated by the axiom α . Formally,

$$\frac{}{x.t\ \alpha\ y.t[x//y] \text{ for } y \text{ fresh}}$$

$$\frac{t =_\alpha t}{x.t =_\alpha x.t} \quad \frac{t =_\alpha t \quad u =_\alpha u}{t\ u =_\alpha t\ u}$$

6

Examples

$$\begin{aligned} x.(x.x\ z) &\alpha\ y.(x.x\ z) =_\alpha\ y.(y.y\ z) \\ x\ (x.x) &=_\alpha\ x\ (z.z) \end{aligned}$$

8

Barendregt variable convention

From now on we assume the following variable convention :

1. No variable is both free and bound.
2. Bound variables have all different names.

Example : $x (z.z)$ is OK, $x (x.x)$ is not OK, $x. y.x z$ is OK but $x. x.x z$ is not OK.

Theorem : For every λ -term t there is λ -term u verifying the Barendregt convention such that $t =_{\alpha} u$.

Indeed, $x (x.x) =_{\alpha} x (z.z)$ and $x. x.x z =_{\alpha} y. x.x z$.

9

Towards a notion of substitution : warning!

$$(x.(y.x)) y \beta (y.x) \{x/y\} = y.y$$

Incorrect

$$(x.(y.x)) y =_{\alpha} (x.(z.x)) y \beta (z.x) \{x/y\} = z.y$$

Correct

11

Operational semantics of λ -calculus

A **one-step** β -reduction is given inductively by

$$\frac{}{(x.t) u \beta t \{x/u\}} \quad \frac{t \beta t}{x.t \beta x.t}$$

$$\frac{t \beta t}{t u \beta t u} \quad \frac{u \beta u}{t u \beta t u}$$

What is exactly $_ \{ _ / _ \}$?

10

A simple notion of higher-order substitution

$t \{x/u\}$ means replace all the **free** occurrences of x in t by u .

This operation is defined **modulo** β -conversion as follows :

$$x \{x/u\} = u$$

$$y \{x/u\} = y$$

$$(y.v) \{x/u\} = y.v \{x/u\} \text{ if } x = y \text{ and no capture holds}$$

$$(t v) \{x/u\} = (t \{x/u\} v \{x/u\})$$

12

A terminating β -reduction sequences

$$\begin{aligned}
 (x. f.x f y) (z.z) (w.w w) & \quad \beta \\
 (f.x f y) \{x/z.z\} (w.w w) & = \\
 (f.(z.z) f y) (w.w w) & \quad \beta \\
 (f.f y) (w.w w) & \quad \beta \\
 (f y) \{f/w.w w\} & = \\
 (w.w w) y & \quad \beta \\
 (w w) \{w/y\} & = \\
 (y y) &
 \end{aligned}$$

13

A non terminating β -reduction sequences

Let $\Pi = x.f(x x)$.

$$\begin{aligned}
 \Pi \Pi & \quad \beta \\
 (f(x x)) \{x/\Pi\} & = \\
 f(\Pi \Pi) & \quad \beta \\
 f(f(x x)) \{x/\Pi\} & = \\
 f(f(\Pi \Pi)) & \quad \beta \\
 \vdots &
 \end{aligned}$$

14

Properties of β -calculus

[Confluence] The reduction relation β is confluent.

[Free variables decrease] If $t \beta t'$, then $fV(t) \supseteq fV(t')$.

15

Curry-Howard Isomorphism

Logical system

Propositions

Proofs

Proof normalisation

Language

Types

Programs

Program Evaluation

16

Curry'58, Howard'68

17

Typing Environment

A **typing environment** Γ is a **finite** function from variables to types, usually written $x_1 : A_1, \dots, x_n : A_n$.

Thus for example, $x : A, y : B$ and $y : B, x : A$ are two different notations for the same typing environment.

The **domain** of $\Gamma = x_1 : A_1, \dots, x_n : A_n$, written $\text{dom}(\Gamma)$, is the set $\{x_1, \dots, x_n\}$.

We write $\Gamma, x : A$ for the typing environment extending Γ with the pair $x : A$. It is only defined iff $x \notin \text{dom}(\Gamma)$.

19

Adding (simply) types to λ -calculus

Grammar for types :

$$A, B ::= \quad \text{(base types)} \quad / \quad A \rightarrow B \quad \text{(functional types)}$$

Notation : \rightarrow is right-associative so that for example $A_1 \rightarrow A_2 \rightarrow A_3$ means $A_1 \rightarrow (A_2 \rightarrow A_3)$.

18

Natural deduction as **typed** λ -calculus

$$\frac{}{\Gamma, x : A \quad x : A} \quad (ax)$$

$$\frac{\Gamma, x : A \quad t : B}{\Gamma \quad x.t : A \rightarrow B} \quad (i) \quad \frac{\Gamma \quad t : A \rightarrow B \quad \Gamma \quad u : A}{\Gamma \quad t u : B} \quad (e)$$

We denote by $\Gamma \vdash t : A$ the derivability/typing relation. We say that t is **typable** iff there is Γ and A such that $\Gamma \vdash t : A$.

Remark : $\Gamma \vdash t : A$ denotes also a sequent!

20

Examples

Example of a typable term : $(\lambda x.x)(\lambda y.y)$.

$$\frac{\frac{y:}{y.y: (\quad)} \quad \frac{y:}{(\quad)}}{\quad} \quad \frac{x: \quad x:}{x.x:} \\ \hline (\lambda y.y)(\lambda x.x) :$$

Example of non-typable term : $\lambda x.xx$.

21

Typed Properties

[Subject Reduction] If $\Gamma \vdash t : A$ and $t \rightarrow_{\beta} t'$, then $\Gamma \vdash t' : A$.

[Strong Normalization] Every **typed** term is normalising : if $\Gamma \vdash t : A$, then $t \rightarrow_{\beta}^* SN_{\beta}$.

22

Some General Remarks

- When using typed terms, the notation $t\{x/u\}$ means that x and u have the same type.
- $t \rightarrow_{\beta}^* SN_{\beta}$ iff there is no infinite \rightarrow_{β} -reduction sequence starting at t
iff every \rightarrow_{β} -reduction sequence starting at t is finite
iff $t \rightarrow_{\beta}^* [(t \rightarrow_{\beta} t') \text{ implies } t' \rightarrow_{\beta}^* SN_{\beta}]$.
- A particular case is : $t \rightarrow_{\beta}^* SN_{\beta}$ if t is in β -normal form.
- If $t \rightarrow_{\beta}^* SN_{\beta}$, then every subterm of t is also $\rightarrow_{\beta}^* SN_{\beta}$.
- SN_{β} is not stable by substitution. Example : $x \rightarrow_{\beta}^* SN_{\beta}$, $y.y \rightarrow_{\beta}^* SN_{\beta}$, but $(x \rightarrow_{\beta}^* SN_{\beta})\{x/y.y\} = \Delta \rightarrow_{\beta}^* \Delta \not\rightarrow_{\beta}^* SN_{\beta}$.
- $u \rightarrow_{\beta}^* SN_{\beta}$ iff $y.u \rightarrow_{\beta}^* SN_{\beta}$.
- $u_1, \dots, u_n \rightarrow_{\beta}^* SN_{\beta}$ iff $x u_1 \dots u_n \rightarrow_{\beta}^* SN_{\beta}$.

23

- Given $t \rightarrow_{\beta}^* SN_{\beta}$ we define $\mu_{\beta}(t)$ as the maximal length of a reduction sequence starting at t . We observe that $t \rightarrow_{\beta} t'$ implies $\mu_{\beta}(t) > \mu_{\beta}(t')$.
- Every type A can be written as $A_1 \rightarrow \dots \rightarrow A_n$, where A_1, \dots, A_n ($n \geq 0$) are arbitrary types and \rightarrow is a base type.
- The standard order between types is given by $A < A \rightarrow B$ and $B < A \rightarrow B$.
Thus base types are minimal with respect this order.

24

First Proof of the SN property

Définition : Let M be of type $A = A_1 \dots A_n$. Then $M \text{ SC}$ iff $R_i : A_i \text{ SC} \implies M R_1 \dots R_n \text{ SN}_\beta$.

The definition implies

1. $\text{SC} \implies \text{SN}$.
2. SC is closed under \dots .
3. $x \text{ SC}$ for every variable x (using 1).

25

Lemma : Let M be a typed term. Let σ be a type preserving substitution mapping all the free variables of M to terms in SC . Then $M \sigma \text{ SC}$.

Proof. We proceed by **induction** on the typed term M .

- If $M = x$, then $x \sigma = (x) \text{ SC}$ by hypothesis.
- If $M = NL$, then let $R_n \text{ SC}$. We have N and L in SC by **i.h.** Then $(N L) R_n = N L R_n \text{ SN}_\beta$ by definition.
- If $M = \lambda x. N$, then $(\lambda x. N) \sigma = \lambda x. N \sigma$. Since $\sigma \{x/x\}$ verifies the hypothesis of the lemma, then by the **i.h.** $N(\sigma \{x/x\}) = N \sigma \text{ SC} \text{ SN}_\beta$. To show $\lambda x. N \sigma \text{ SC}$ we show $(\lambda x. N \sigma) P_1 \dots P_n \text{ SN}_\beta$ for $P_1 \dots P_n \text{ SC} \text{ SN}_\beta$.

This follows from the Basic Lemma.

27

Lemma : [Basic Lemma] If t, R_1, \dots, R_n ($n \geq 1$) SN_β and $t\{x/R_1\}R_2 \dots R_n \text{ SN}_\beta$, then $(\lambda x. t)R_1 R_2 \dots R_n \text{ SN}_\beta$.

Proof. It is sufficient to show that all the reducts of

$(\lambda x. t)R_1 \dots R_n$ are in SN_β . We reason by **induction** on

$\mu(t) + \sum_i \mu(R_i)$. The reducts are

- $(\lambda x. t)R_1 \dots R_n$, where $t \rightarrow t'$. Then $\mu(t') < \mu(t)$, we conclude by the **i.h.**
- $(\lambda x. t)R_1 \dots R_i \dots R_n$, where $R_i \rightarrow R_i'$. Then $\mu(R_i') < \mu(R_i)$, we conclude by the **i.h.**
- $t\{x/R_1\}R_2 \dots R_n$. We conclude by the hypothesis.

26

Proof. Using the previous lemma the th.

Every typed term is

Lemma : Every typed term is SC .

Proof. Using the previous lemma with the identity substitution (which verifies the hypothesis of the lemma).

Second proof of the SN property

1. Define SN inductively :
 - $M_1, \dots, M_n \text{ SN}$ implies $x M_1 \dots M_n \text{ SN}$.
 - $M \text{ SN}$ implies $x.M \text{ SN}$.
 - $M\{x/N\}P_n \text{ SN}$ and $N \text{ SN}$ implies $(x.M) N P_n \text{ SN}$.
2. Define Λ_A inductively :
 - If x is a variable of type A , then $x \in \Lambda_A$.
 - If $t \in \Lambda_C$ and x is a variable of type B , then $x.t \in \Lambda_{B \rightarrow C}$.
 - If $t \in \Lambda_{B \rightarrow A}$ and $u \in \Lambda_B$, then $t u \in \Lambda_A$.
3. Define $SN_A = \{M \mid M \text{ SN} \wedge M \in \Lambda_A\}$.
4. Define $X \rightarrow Y = \{M \mid N \rightarrow X \wedge (MN) \rightarrow Y\}$

29

5. Show $SN_\beta = SN$.
6. Show $\Lambda_{A \rightarrow B} = \Lambda_A \rightarrow \Lambda_B$
7. Show $SN_A \rightarrow SN_B = SN_{A \rightarrow B}$ (easy).
8. If $N \in SN_{A_1} \rightarrow SN_{A_2} \rightarrow \dots \rightarrow SN_{A_m}$ with A_m a base type and $P \in SN_B$, then $P\{x/N\} \in SN_B$ (induction on SN using 7).
9. Show $SN_{A \rightarrow B} = SN_A \rightarrow SN_B$ (using 8).
10. Show that $\Lambda_A = SN_A$ (by induction using 9).

30

Third Proof of the SN property

Lemma : If t and u are typed and belong to SN_β , then $t\{x/u\} \in SN_\beta$.

Proof. By induction on $type(u), \mu(t), size(t)$.

- The base case base type, 0, 1 is trivial.
- Case $t = y.v$ is straightforward ($size(t)$ strictly decreases).
- Case $t = y c_n$ with $x = y$ is straightforward ($\mu(t)$ decreases and $size(t)$ strictly decreases.).
- Case $t = x$. We have $x\{x/u\} = u \in SN_\beta$ by hypothesis.
- Case $t = x b c_n$. By i.h. $B = b\{x/u\}$ and $C_i = c_i\{x/u\}$ are in SN_β . We want to show that $u B C_n \in SN_\beta$. It is sufficient to show that all its reducts are in SN_β . We reason by induction on

31

- $\mu(u) + \mu(B) + \sum_i \mu(C_i)$. The reducts are
 - $u B C_n$, where $u \rightarrow u'$. Apply the i.h.
 - $u B C_n$, where $B \rightarrow B'$. Apply the i.h.
 - $u B C_1 \dots C_i \dots C_n$, where $C_i \rightarrow C'_i$. Apply the i.h.
 - $u_1\{y/B\}C_n$, where $u = y.u_1$. But $u_1\{y/B\}C_n = (zC_n)\{z/u_1\{y/B\}\}$ and $type(u_1\{y/B\}) < type(u)$. We thus conclude by the i.h. since zC_n and $u_1\{y/B\}$ are typed and in SN_β by the i.h..
- Case $t = (z.b) c d$. By i.h. $B = b\{x/u\}$ and $C = c\{x/u\}$ and $D_i = d_i\{x/u\}$ are in SN_β . Suppose $t\{x/u\} = (z.B) C D_n \notin SN_\beta$. Then $B\{z/C\} D_n \notin SN_\beta$. But $B\{z/C\} D_n = (b\{z/c\}d_n)\{x/u\}$ and $\mu(b\{z/c\}d_n) < \mu(t)$. Thus $B\{z/C\} D_n \in SN_\beta$ by the i.h. Contradiction. Thus $t\{x/u\} = (z.B) C D_n \in SN_\beta$.

32

Theorem : If t is typable, then $t \rightarrow^* SN_\beta$.

Proof. By induction on the typing derivation of t .

- Case $t = x$ is trivial.
- Case $t = \lambda y. u$ holds by the i.h.
- For the case $t = u \ v$ use the fact that $t = (z \ v) \{z/u\}$ and apply previous lemma (verification of the hypothesis is easy).

33

34

Fourth proof of the SN property

See for example Gandy's proof by Alexandre Miquel.

A combinatorial proof of strong normalisation for the simply typed lambda-calculus.

<http://www.pps.univ-paris-diderot.fr/~miquel/publis/snlam.pdf>

35