Motivation

Concieved by Church in 1932/33

A set of postulates for the foundation of logic, *Annals of Mathematics* (2) 33, pp. 346-366 and 34, pp. 839-864

as model for computable fucntions. This notes are excerpt from

Abramsky S. et al., Handbook of Logic in Computer Science: Volume 2. Background: Computational Structures, Claredon Press, 1992

in particular from chapter *Lambda Calculi with Types* by Henk Barendregt

Motivation - cont.

- **Application** use data (expresion) **F** as an algorithm on data **A**

$$(\mathbf{FA})$$

- **Abstraction** for expresion $\mathbf{M} \equiv \mathbf{M}[\mathbf{x}]$ possibly depending on **x** the map

$$x \mapsto M[x]$$

  is denoted by expression

$$\lambda\mathbf{x}.\mathbf{M}[\mathbf{x}] \quad \text{or} \quad \lambda\mathbf{x}.\mathbf{M}$$

Motivation - cont.

- For example
$$(\lambda x.x + 1)3 = 3^2 + 1$$
- In general we have
$$(\lambda x.M[x])N = M[N]$$
or preferable written as
$$(\lambda x.M[x])N = M[x := N] \qquad (\beta)$$

Formal description

Definition (Lambda Calculus)

The set of λ-*terms* Λ built up from an infinite set of variables $V = \{v, v', v'', \ldots\}$ is a set:

$$
\begin{aligned}
x \in V &\quad\Rightarrow\quad x \in \Lambda \\
M, N \in \Lambda &\quad\Rightarrow\quad (MN) \in \Lambda \\
M \in \Lambda, x \in V &\quad\Rightarrow\quad (\lambda x M) \in \Lambda
\end{aligned}
$$

i. e. in abstract syntax

$$
\begin{aligned}
V &::= v \mid v' \\
\Lambda &::= V \mid (\Lambda\Lambda) \mid (\lambda V \Lambda)
\end{aligned}
$$

Formal description - cont.

Example

Following are λ-*terms*

$$v$$
$$(vv'')$$
$$(\lambda v(vv''))$$
$$((\lambda v'((\lambda v(vv''))v'))v''')$$

Conventions

- $z, y, z, \ldots$ denotes variables
- $M, N, L, \ldots$ denotes lambda terms
- $FM_1M_2 \ldots M_n$ stands for $(\ldots((FM_1)M_2)\ldots M_n)$
- $\lambda x_1x_2 \ldots x_n.M$ stands for $(\lambda x_1(\lambda x2(\ldots((x_n(M)))\ldots)))$

Formal description - cont.

Definition

1. the set of *free variables* of M

$$FV(x) = \{x\}$$
$$FV(MN) = FV(M) \cup FV(N)$$
$$FV(\lambda x.M) = FV(M) \setminus \{x\}$$

   other variables are bound
2. M is a *closed term (combinator)* iff $FV(M) = \emptyset$

Definition (Equivalence up to renaming)

$M \equiv N$ denotes that terms can be obtained from each other by renaming bound variables

Formal description - cont.

Definition

1. The principal axiom scheme called β-conversion: for all $M, N \in \Lambda$

$$(\lambda x.M[x])N = M[x := N] \qquad (\beta)$$

2. logical axioms and rules

$$M = N$$
$$M = N \Rightarrow N = M$$
$$M = N, N = L \Rightarrow M = L$$
$$M = M' \Rightarrow MZ = M'Z$$
$$M = M' \Rightarrow ZM = ZM'$$
$$M = M' \Rightarrow \lambda x.M = \lambda x.M'$$

3. If $M = N$ is provable from axioms than we write $\lambda \vdash M = N$

Fixed point theorem

Theorem (Fixed point theorem)

1. $\forall F \in \Lambda\, \exists X \in \Lambda \quad \lambda \vdash FX = X$
2. *There is a fixed point combinator*

$$\mathbf{Y} \equiv \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$$

s. t.

$$\forall F \quad F(\mathbf{Y}F) = \mathbf{Y}F$$

Proof.

1. Define $W \equiv \lambda x.F(xx)$ and $X \equiv WW$. Then $X \equiv WW \equiv (\lambda x.F(xx))W = F(WW) \equiv FX$.
2. By the proof of (1). $\mathbf{Y} = (\lambda x.F(xx))(\lambda x.F(xx)) \equiv X$

□

Booleans

Definition (Booleans, conditional)

1. **true** $\equiv \lambda xy.x$, **false** $\equiv \lambda xy.y$
2. If B is either **true** ot **false**

$$\textbf{if } B \textbf{ then } P \textbf{ else } Q$$

   can be represented as BPQ

It holds that **true**$PQ = P$ and **false**$PQ = Q$

Church numerals

Definition

1. $F^n(M)$ with $n \in \mathbb{N}$ and $F,\ M \in \Lambda$, is defined:

$$F^0(M) \equiv M$$
$$F^{n+1}(M) \equiv F(F^n(M))$$

2. The *Church numerals* $c_0, c_1, \ldots$ are defined:

$$c_n \equiv \lambda fx.f^n(x)$$

Church numerals - cont.

Lemma (Rosser)

Define

$$A_+ \equiv \lambda xypq.xp(ypq)$$
$$A_* \equiv \lambda xyz.x(yz)$$
$$A_{exp} \equiv \lambda xy.yx$$

then for all $n, m \in \mathbb{N}$

1. $A_+ c_m c_n = c_{m+n}$
2. $A_* c_m c_n = c_{mn}$
3. $A_{exp} c_m c_n = c_{(m^n)}$, *except for* $m = 0$

Operational semantics

In order to program in the language we need to equip it with a semantics. We use operational semantics here. This computatinal aspect is expressed as

$$(\lambda x.x^2 + 1)3 \twoheadrightarrow 10$$

and reads ,,$(\lambda x.x^2 + 1)3$ *reduces to 10*".

β-reduction

Definition

The binary relations $\rightarrow_\beta$, $\twoheadrightarrow_\beta$, and $=_\beta$ are defined

1. (a) $(\lambda x.M)N \rightarrow_\beta M[x := N]$
   (b) $M \rightarrow_\beta N \Rightarrow ZM \rightarrow_\beta ZN, MZ \rightarrow_\beta NZ$ and $\lambda x.M \rightarrow_\beta \lambda x.N$
2. (a) $M \twoheadrightarrow_\beta M$
   (b) $M \rightarrow_\beta N \rightarrow M \twoheadrightarrow_\beta N$
   (c) $M \twoheadrightarrow_\beta N, N \twoheadrightarrow_\beta L \Rightarrow M \twoheadrightarrow_\beta L$
3. (a) $M \twoheadrightarrow_\beta M \Rightarrow M =_\beta N$
   (b) $M =_\beta M \Rightarrow N =_\beta M$
   (c) $M =_\beta N, N =_\beta L \Rightarrow M =_\beta L$

and read *β-reduces in one step to*, *β-reduces to*, and *is β convertible to*.

Lemma

$$M =_\beta N \Leftrightarrow \lambda \vdash M = N$$

β-reduction - cont.

Definition

1. A *β-redex* is a term of the form

$$(\lambda x.M)N$$

   and in this case

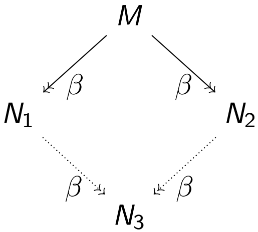
$$M[x := N]$$

   is its *contractum*

2. A λ-term M is in a *β-normal form* it it does not have a β-redex as subexpression
3. A term M *has* a β-normal form if $M =_\beta N$ and N is in a β-nf, for some N

Theorem (Church-Rosser)

If $M \twoheadrightarrow_\beta N_1$ and $M \twoheadrightarrow_\beta N_2$ then for some N_3 one has $N_1 \twoheadrightarrow_\beta N_3$ and $N_2 \twoheadrightarrow_\beta N_3$:



Corollary

1. *If $M =_\beta N$ then there is an L s. t. $M \twoheadrightarrow_\beta L$ and $N \twoheadrightarrow_\beta L$*
2. *If M has N as β-nf then $M \twoheadrightarrow_\beta N$*
3. *A λ-term has at most one β-nf*

Definition

1. The *main symbol* of $(\lambda x.M)N$ is the first λ.
2. Let R_1,R_2 be two redexes in M. Then R_1 *is to the left of* R_2 if the main symbol of R_1 is to the left of R_2
3. We write $M \rightarrow_l N$ if N results from M by contracting the leftmost redex M. The reflexive transitive closure is denoted $\twoheadrightarrow_l$

Theorem (Curry)

If M *has a* β*-normal form then* $M \twoheadrightarrow_l N$