Lambda Calculus

Week 1

The systems CL and $\boldsymbol{\lambda}$

Henk Barendregt, Freek Wiedijk

assisted by Andrew Polonsky

Lambda calculus (and the related combinatory logic)

is a language expressing

- algorithms functional programming
- proofs machine verification

with applications

The system comes in several versions

- untyped
- typed

Types are like dimensions of constants in physics

$c = 3.10^8$m/s for the speed of light

giving important partial correctness check

$$E = mc^2 \quad \text{both having dimension } gm^2/s^2$$

An alphabet Σ is a set of symbols

A word over Σ is a finite string of elements in Σ

Σ^* consists of all strings over Σ (possibly nonsensical)

A language L over Σ is a subset $L \subseteq \Sigma^*$

$L \subseteq \Sigma^*$ chooses in some way *meaningful* strings called *formulas*

often such a language is given by a grammar

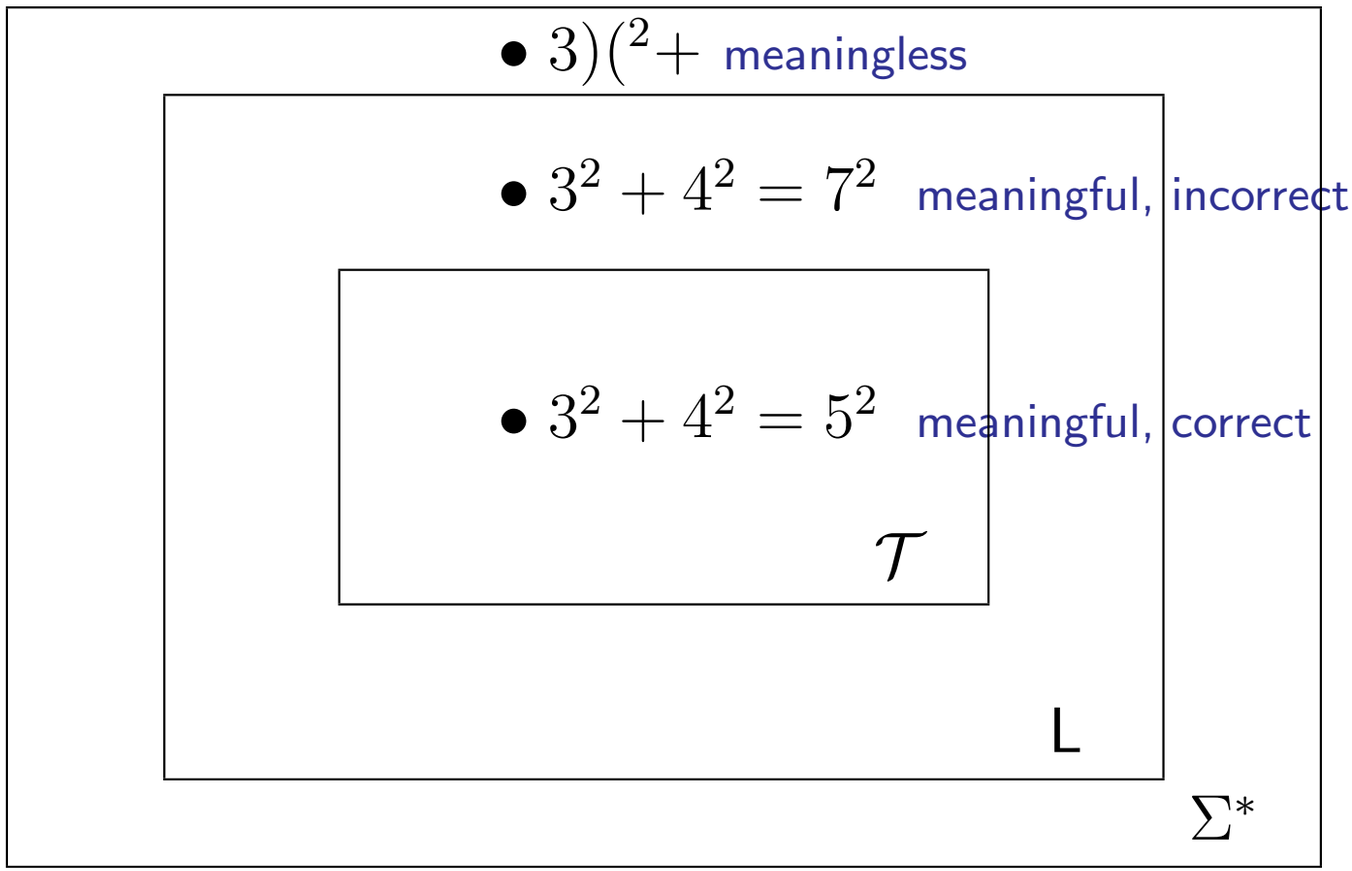
With a theory $\mathcal{T}$ we go one step further: it is a subset $\mathcal{T} \subseteq L$

selecting a set of *correct* (true, valid, intended) sentences

often such a theory is given by an axiomatic system

In an equational theory sentences are of the form $t_1 = t_2$

with t_1, t_2 belonging to a set of terms $T \subseteq \Sigma^*$



$$\Sigma \longmapsto \Sigma^* \longmapsto L \longmapsto \mathcal{T}$$

Alphabet

$$\Sigma_{\mathsf{CL}} = \{\mathbf{I}, \mathbf{K}, \mathbf{S}, x, ',), (, =\}$$

We introduce several simple regular grammars over Σ_{CL}.

(i) $\boxed{\texttt{constant} := \mathbf{I} \mid \mathbf{K} \mid \mathbf{S}}$

(ii) $\boxed{\texttt{variable} := x \mid \texttt{variable}'}$

(iii) $\boxed{\texttt{term} := \texttt{constant} \mid \texttt{variable} \mid (\texttt{term}\,\texttt{term})}$

(iv) $\boxed{\texttt{formula} := \texttt{term} = \texttt{term}}$

Intuition:

in (FA) the term F stands for a *function* and A for an *argument*

Axioms

$$\begin{array}{rcll} \mathbf{I}P & = & P & (\mathbf{I}) \\ \mathbf{K}PQ & = & P & (\mathbf{K}) \\ \mathbf{S}PQR & = & PR(QR) & (\mathbf{S}) \end{array}$$

Deduction rules (equational logic)

$$\begin{array}{rcl} & & P = P \\ P = Q & \Rightarrow & Q = P \\ P = Q, Q = R & \Rightarrow & P = R \\ P = Q & \Rightarrow & PR = QR \\ P = Q & \Rightarrow & RP = RQ \end{array}$$

Here P, Q, R denote arbitrary terms

$\mathbf{I}P$ stands for $(\mathbf{I}P)$, $\mathbf{K}PQ$ for $((\mathbf{K}P)Q)$ and $\mathbf{S}PQR$ for $(((\mathbf{S}P)Q)R)$

In general $PQ_1 \dots Q_n \equiv (..((PQ_1)Q_2) \dots Q_n)$ (association to the left)

PROPOSITION.

(i) Let $\mathbf{D} \triangleq \mathbf{SII}$. Then (doubling)

$$\mathbf{D}x =_{\mathsf{CL}} xx.$$

(ii) Let $\mathbf{B} \triangleq \mathbf{S}(\mathbf{KS})\mathbf{K}$. Then (composition)

$$\mathbf{B}fgx =_{\mathsf{CL}} f(gx).$$

(iii) Let $\mathbf{L} \triangleq \mathbf{D}(\mathbf{BDD})$. Then (self-doubling, life!)

$$\mathbf{L} =_{\mathsf{CL}} \mathbf{LL}.$$

PROOF.

(i)
$$\begin{array}{lcl} \mathbf{D}x & \triangleq & \mathbf{SII}x \\ & = & \mathbf{I}x(\mathbf{I}x) \\ & = & xx. \end{array}$$

(ii)
$$\begin{array}{lcl} \mathbf{B}fgx & \triangleq & \mathbf{S}(\mathbf{KS})\mathbf{K}fgx \\ & = & \mathbf{KS}f(\mathbf{K}f)gx \\ & = & \mathbf{S}(\mathbf{K}f)gx \\ & = & \mathbf{K}fx(gx) \\ & = & f(gx). \end{array}$$

(iii)
$$\begin{array}{lcl} \mathbf{L} & \triangleq & \mathbf{D}(\mathbf{BDD}) \\ & = & \mathbf{BDD}(\mathbf{BDD}) \\ & = & \mathbf{D}(\mathbf{D}(\mathbf{BDD})) \\ & \triangleq & \mathbf{DL} \\ & = & \mathbf{LL}. \end{array}$$

We want to understand and preferably also to control this!

The meaning of

$$\lambda x.3x$$

is the function

$$x \longmapsto 3x$$

that assigns to x the value $3x$ (3 times x). Therefore

$$(\lambda x.3x)(6) = 18.$$

The parentheses around the 6 are usually not written:

$$(\lambda x.3x)6 = 18$$

Principal axiom

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

Alphabet

$$\Sigma = \{x, ', (,), \lambda, =\}$$

Language

$$\begin{aligned} \texttt{variable} &:= \texttt{x} \mid \texttt{variable}' \\ \texttt{term} &:= \texttt{variable} \mid (\texttt{term}\ \texttt{term}) \mid (\lambda\, \texttt{variable}\ \texttt{term}) \\ \texttt{formula} &:= \texttt{term} = \texttt{term} \end{aligned}$$

Theory 1. Axioms $(\lambda x\, M)N = M[x := N]$

2. Deduction rules: equational logic

$$\begin{aligned} & & M &= M \\ M &= N & \Rightarrow \quad N &= M \\ M &= N, N = L & \Rightarrow \quad M &= N \\ M &= N & \Rightarrow \quad ML &= NL \\ M &= N & \Rightarrow \quad LM &= LN \\ M &= N & \Rightarrow \quad \lambda x\, M &= \lambda x\, N \end{aligned}$$

Substitution

M	$M[x:=N]$
x	N
$y(\not\equiv x)$	y
PQ	$(P[x:=N])(Q[x:=N])$
$\lambda x\, P$	$\lambda x\, P$
$\lambda y\, P$	$\lambda y\, (P[x:=N])$

'Association to the left'

$$PQ_1 \dots Q_n \equiv (..((PQ_1)Q_2) \dots Q_n).$$

'Associating to the right'

$$\lambda x_1 \dots x_n.M \equiv (\lambda x_1(\lambda x_2(..(\lambda x_n(M))..)))).$$

Outer parentheses are usually omitted. For example

$$(\lambda x.x)y \equiv ((\lambda xx)y)$$

$\lambda x.x$ and $\lambda y.y$ acting on M both give M

We write

$$\lambda x.x \equiv_\alpha \lambda y.y$$

"Names of *bound variables* may be changed"

NB

$$\begin{aligned} \mathsf{K}MN &\equiv (\lambda xy.x)MN \\ &\equiv (((\lambda x(\lambda y\ x))M)N) \\ &= ((\lambda yM)N) \\ &= M \qquad \text{assuming that } y \text{ not in } M. \end{aligned}$$

But

$$\begin{aligned} \mathsf{K}yz &\equiv (((\lambda x(\lambda y\ x))y)z) \\ &=_? ((\lambda y\ y)z) \\ &= z?? \end{aligned}$$

better:

$$\begin{aligned} \mathsf{K}yz &\equiv (((\lambda x'(\lambda y'\ x'))y)z) \\ &= (\lambda y'\ y)z \\ &= y \qquad \text{as it should.} \end{aligned}$$

There is an asymmetry in

$$(\lambda x\, M)N = M[x := N]$$

We express this by (*contraction* rule)

$$(\lambda x\, M)N \mapsto_\beta M[x := N]$$

This gives rise to $\to_\beta$ (*one step reduction*)

and $\twoheadrightarrow_\beta$ (*many step reduction*)

$$\begin{array}{rcl}
M \mapsto_\beta N & \Rightarrow & M \to_\beta N \\
M \to_\beta N & \Rightarrow & ML \to_\beta NL \\
M \to_\beta N & \Rightarrow & LM \to_\beta LN \\
M \to_\beta N & \Rightarrow & \lambda x\, M \to_\beta \lambda x\, N \\
M \to_\beta N & \Rightarrow & M \twoheadrightarrow_\beta N \\
 & & M \twoheadrightarrow_\beta M \\
M \twoheadrightarrow_\beta N, N \twoheadrightarrow_\beta L & \Rightarrow & M \twoheadrightarrow_\beta L
\end{array}$$

$\twoheadrightarrow_\beta$ is the transitive reflexive closure of $\to_\beta$

$\to_\beta$ is the 'compatible closure' of $\mapsto_\beta$

Set of lambda terms: Λ. Examples:

$$\begin{array}{rcllrcl}
\mathsf{I} & \triangleq & \lambda x.x & \Rightarrow & \mathsf{I}X & \to_\beta & X \\
\mathsf{K} & \triangleq & \lambda xy.x & \Rightarrow & \mathsf{K}XY & \twoheadrightarrow_\beta & X \\
\mathsf{S} & \triangleq & \lambda xyz.xz(yz) & \Rightarrow & \mathsf{S}XYZ & \twoheadrightarrow_\beta & XZ(YZ) \\
\mathsf{D} & \triangleq & \lambda x.xx & \Rightarrow & \mathsf{D}X & \twoheadrightarrow_\beta & XX \\
\mathsf{B} & \triangleq & \lambda xyz.x(yz) & \Rightarrow & \mathsf{B}XYZ & \twoheadrightarrow_\beta & X(YZ)
\end{array}$$

Indeed, $\mathsf{K}XY$

$$\begin{array}{rl}
\triangleq & (\lambda xy.x)XY \\
\triangleq & (\underline{(\lambda x(\lambda yx))X})Y \\
\to_\beta & \underline{(\lambda yX)Y} \\
\to_\beta & X
\end{array}$$

Variable convention: assume that the bound and free variables in a situation differ

The set of free (bound) variables of M, notation $\mathrm{FV}(M)$ (resp. $\mathrm{BV}(M)$), is

$\mathrm{FV}(x) = \{x\}$, $\mathrm{FV}(MN) = \mathrm{FV}(M) \cup FV(N)$, $\mathrm{FV}(\lambda x.M) = \mathrm{FV}(M) - \{x\}$

$\mathrm{BV}(x) = \emptyset$, $\mathrm{BV}(MN) = \mathrm{BV}(M) \cup BV(N)$, $\mathrm{BV}(\lambda x.M) = \mathrm{BV}(M) \cup \{x\}$.

Exercises

THEOREM. For all $F \in \Lambda$ there is an $M \in \Lambda$ such that

$$FM =_\beta M$$

PROOF. Defines $\mathsf{W} \equiv \lambda x.F(xx)$ and $M \equiv \mathsf{WW}$. Then

$$\begin{aligned} M &\equiv \mathsf{WW} \\ &\equiv (\lambda x.F(xx))\mathsf{W} \\ &= F(\mathsf{WW}) \\ &\equiv FM. \blacksquare \end{aligned}$$

COROLLARY. For any 'context' $C[\vec{x}, m]$ there exists a M such that

$$M\vec{X} = C[\vec{X}, M].$$

PROOF. M can be taken the fixed point of $\lambda m\vec{x}.C[\vec{x}, m]$.
Then $M\vec{X} = (\lambda m\vec{x}.C[\vec{x}, m])M\vec{X} = C[\vec{X}, M]$. $\blacksquare$

We can construct terms $\mathsf{Y}, \mathsf{L}, \mathsf{O}, \mathsf{P}$ such that

$$\begin{aligned} \mathsf{Y}f &= f(\mathsf{Y}f) && \text{producing fixed points;} \\ \mathsf{L} &= \mathsf{LL} && \text{take } \mathsf{L} \equiv \mathsf{Y}\,\mathsf{D}; \\ \mathsf{O}x &= \mathsf{O} && \text{take } \mathsf{O} \equiv \mathsf{Y}\,\mathsf{K}; \\ \mathsf{P} &= \mathsf{P}x. \end{aligned}$$

LEMMA. For every CL-term P and every variable x,

there is a CL-term Q such that x does not occur in Q and

$$Qx =_{\mathsf{CL}} P.$$

We denote this term Q constructed in the proof as $[x]P$

PROOF. Induction on the complexity of P

Case 1. P is a constant or a variable

Subcase 1.1 $P \equiv x$. Take $[x]x \equiv \mathbf{I}$, then

$$([x]x)x \equiv \mathbf{I}x =_{\mathsf{CL}} x.$$

Subcase 1.2 $P \equiv y \not\equiv x$. Take $[x]y \equiv \mathbf{K}x$, then indeed

$$([x]y)x \equiv \mathbf{K}yx =_{\mathsf{CL}} y.$$

Subcase 1.3 $P \equiv \mathbf{C}$ with $\mathbf{C} \in \{\mathbf{I}, \mathbf{K}, \mathbf{S}\}$. Take $[x]\mathbf{C} \equiv \mathbf{K}\mathbf{C}$, then indeed

$$([x]\mathbf{C})x =_{\mathsf{CL}} K\mathbf{C}x =_{\mathsf{CL}} \mathbf{C}.$$

Case 2. $P \equiv UV$. Take $[x](UV) \equiv \mathbf{S}([x]U)([x]V)$, then indeed

$$([x](UV))x \equiv \mathbf{S}([x]U)([x]V)x =_{\mathsf{CL}} (([x]U)x)(([x]V)x) =_{\mathsf{CL}} UV. \blacksquare$$

The previous proof gave

P	$[x]P$	$([x]P)x = P$?
x	$\mathbf{I}$	$\mathbf{I}x = x$
$y \not\equiv x$	$\mathbf{K}y$	$\mathbf{K}yx = y$
$\mathbf{C}$	$\mathbf{K}\mathbf{C}$	$\mathbf{K}\mathbf{C}x = \mathbf{C}$
UV	$\mathbf{S}([x]U)([x]V)$	$\mathbf{S}([x]U)([x]V)x =$ $(([x]U)x)(([x]V)x) = UV$

More efficient algorithm

P	$[x]P$
x	$\mathbf{I}$
P with $x \notin P$	$\mathbf{K}P$
UV	$\mathbf{S}([x]U)([x]V)$

Define $=_\beta$ as the symmetric transitive relation containing $\twoheadrightarrow_\beta$

$$\begin{aligned} M \twoheadrightarrow_\beta N &\Rightarrow M =_\beta N \\ M =_\beta N &\Rightarrow N =_\beta M \\ M =_\beta N \;\&\; N =_\beta L &\Rightarrow M =_\beta L \end{aligned}$$

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

PROOF. ($\Rightarrow$) By induction on the proof of $\lambda \vdash M = N$.
($\Leftarrow$) Show (except for the first case by a similar induction)

$$\begin{aligned} M \mapsto_\beta N &\Rightarrow \lambda \vdash M = N \\ M \to_\beta N &\Rightarrow \lambda \vdash M = N \\ M \twoheadrightarrow_\beta N &\Rightarrow \lambda \vdash M = N \\ M =_\beta N &\Rightarrow \lambda \vdash M = N \Box \end{aligned}$$

CHURCH-ROSSER THEOREM
$M =_\beta N \Leftrightarrow \exists Z.[M \twoheadrightarrow_\beta Z \;\&\; N \twoheadrightarrow_\beta Z]$

Let $M \in \Lambda$ with $x \notin \mathrm{FV}(M)$. Then

$$\lambda x.Mx \mapsto_\eta M \qquad (\eta)$$

From this one gets $\to_\eta, \twoheadrightarrow_\eta, \twoheadrightarrow_{\beta\eta}$