Rachunek lambda - ciąg dalszy

18 marca 2013

Siła wyrazu: logika zdaniowa

$$\textbf{true} = \lambda xy.x \qquad\qquad \textbf{false} = \lambda xy.y$$

$$\textbf{if } P \textbf{ then } Q \textbf{ else } R = PQR.$$

It works:

$$\textbf{if true then } Q \textbf{ else } R \twoheadrightarrow_\beta Q$$

$$\textbf{if false then } Q \textbf{ else } R \twoheadrightarrow_\beta R.$$

○

Ordered pair

Pair = Boolean selector:

$$
\begin{aligned}
\langle M, N \rangle &= \lambda x. x M N; \\
\pi_i &= \lambda x_1 x_2 . x_i \qquad (i = 1, 2); \\
\Pi_i &= \lambda p.\, p \pi_i \qquad (i = 1, 2).
\end{aligned}
$$

Ordered pair

Pair = Boolean selector:

$$
\begin{array}{rcll}
\langle M, N\rangle & = & \lambda x.xMN; & \\
\pi_i & = & \lambda x_1 x_2.x_i & (i = 1, 2); \\
\Pi_i & = & \lambda p.\, p\pi_i & (i = 1, 2).
\end{array}
$$

It works:

$$\Pi_1\langle M, N\rangle \to_\beta \langle M, N\rangle\pi_1 \twoheadrightarrow_\beta M.$$

○

Church's numerals

$$c_n = \mathbf{n} = \lambda fx.f^n(x),$$

$$\begin{aligned}
\mathbf{0} &= \lambda fx.x; \\
\mathbf{1} &= \lambda fx.fx; \\
\mathbf{2} &= \lambda fx.f(fx); \\
\mathbf{3} &= \lambda fx.f(f(fx)), \text{ etc.}
\end{aligned}$$

Some definable functions

- Successor: $\mathbf{succ} = \lambda nfx.f(nfx)$;

Some definable functions

- Successor: $\mathbf{succ} = \lambda nfx.f(nfx);$
- Addition: $\mathbf{add} = \lambda mnfx.mf(nfx);$

Some definable functions

- Successor: $\mathbf{succ} = \lambda nfx.f(nfx)$;
- Addition: $\mathbf{add} = \lambda mnfx.mf(nfx)$;
- Multiplication: $\mathbf{mult} = \lambda mnfx.m(nf)x$;

Some definable functions

- Successor: $\mathbf{succ} = \lambda nfx.f(nfx)$;
- Addition: $\mathbf{add} = \lambda mnfx.mf(nfx)$;
- Multiplication: $\mathbf{mult} = \lambda mnfx.m(nf)x$;
- Exponentiation: $\mathbf{exp} = \lambda mnfx.mnfx$;

Some definable functions

- Successor: $\mathbf{succ} = \lambda nfx.f(nfx)$;
- Addition: $\mathbf{add} = \lambda mnfx.mf(nfx)$;
- Multiplication: $\mathbf{mult} = \lambda mnfx.m(nf)x$;
- Exponentiation: $\mathbf{exp} = \lambda mnfx.mnfx$;
- Test for zero: $\mathbf{zero} = \lambda m.m(\lambda y.\mathbf{false})\mathbf{true}$;

Predecessor is definable too

$$p(n+1) = n, \quad p(0) = 0$$

Predecessor is definable too

$$p(n+1) = n, \quad p(0) = 0$$

$$\mathbf{Step} = \lambda p. \langle \mathbf{succ}(p\pi_1), p\pi_1 \rangle$$

$$\mathbf{pred} = \lambda n. (n\,\mathbf{Step}\langle \mathbf{0}, \mathbf{0} \rangle)\pi_2$$

Predecessor is definable too

$$p(n+1) = n, \quad p(0) = 0$$

$$\mathbf{Step} = \lambda p.\langle \mathbf{succ}(p\pi_1), p\pi_1 \rangle$$
$$\mathbf{pred} = \lambda n.\,(n\,\mathbf{Step}\langle \mathbf{0}, \mathbf{0} \rangle)\pi_2$$

How it works:

Predecessor is definable too

$$p(n+1) = n, \quad p(0) = 0$$

$$\mathbf{Step} = \lambda p. \langle \mathbf{succ}(p\pi_1), p\pi_1 \rangle$$

$$\mathbf{pred} = \lambda n. (n\, \mathbf{Step} \langle \mathbf{0}, \mathbf{0} \rangle)\pi_2$$

How it works:

$$\mathbf{Step}\langle \mathbf{0}, \mathbf{0} \rangle \twoheadrightarrow_\beta \langle \mathbf{1}, \mathbf{0} \rangle$$

Predecessor is definable too

$$p(n+1) = n, \quad p(0) = 0$$

$$\mathbf{Step} = \lambda p. \langle \mathbf{succ}(p\pi_1), p\pi_1 \rangle$$

$$\mathbf{pred} = \lambda n. (n\, \mathbf{Step} \langle \mathbf{0}, \mathbf{0} \rangle) \pi_2$$

How it works:

$$\mathbf{Step}\langle \mathbf{0}, \mathbf{0} \rangle \twoheadrightarrow_\beta \langle \mathbf{1}, \mathbf{0} \rangle$$
$$\mathbf{Step}\langle \mathbf{1}, \mathbf{0} \rangle \twoheadrightarrow_\beta \langle \mathbf{2}, \mathbf{1} \rangle$$

Predecessor is definable too

$$p(n+1) = n, \quad p(0) = 0$$

$$\mathbf{Step} = \lambda p.\langle \mathbf{succ}(p\pi_1), p\pi_1\rangle$$

$$\mathbf{pred} = \lambda n.\,(n\,\mathbf{Step}\langle \mathbf{0}, \mathbf{0}\rangle)\pi_2$$

How it works:

$$\mathbf{Step}\langle \mathbf{0}, \mathbf{0}\rangle \twoheadrightarrow_\beta \langle \mathbf{1}, \mathbf{0}\rangle$$
$$\mathbf{Step}\langle \mathbf{1}, \mathbf{0}\rangle \twoheadrightarrow_\beta \langle \mathbf{2}, \mathbf{1}\rangle$$
$$\mathbf{Step}\langle \mathbf{2}, \mathbf{1}\rangle \twoheadrightarrow_\beta \langle \mathbf{3}, \mathbf{2}\rangle,$$

Predecessor is definable too

$$p(n+1) = n, \quad p(0) = 0$$

$$\mathbf{Step} = \lambda p. \langle \mathbf{succ}(p\pi_1), p\pi_1 \rangle$$

$$\mathbf{pred} = \lambda n. (n\, \mathbf{Step} \langle \mathbf{0}, \mathbf{0} \rangle)\pi_2$$

How it works:

$$\begin{aligned} &\mathbf{Step}\langle \mathbf{0}, \mathbf{0} \rangle \twoheadrightarrow_\beta \langle \mathbf{1}, \mathbf{0} \rangle \\ &\mathbf{Step}\langle \mathbf{1}, \mathbf{0} \rangle \twoheadrightarrow_\beta \langle \mathbf{2}, \mathbf{1} \rangle \\ &\mathbf{Step}\langle \mathbf{2}, \mathbf{1} \rangle \twoheadrightarrow_\beta \langle \mathbf{3}, \mathbf{2} \rangle, \end{aligned}$$

and so on.

Undecidability

The following are undecidable problems:

- Given M and N, does $M \twoheadrightarrow_\beta N$ hold?

Undecidability

The following are undecidable problems:

- Given M and N, does $M \twoheadrightarrow_\beta N$ hold?
- Given M and N, does $M =_\beta N$ hold?

Undecidability

The following are undecidable problems:

- Given M and N, does $M \twoheadrightarrow_\beta N$ hold?
- Given M and N, does $M =_\beta N$ hold?
- Given M, does M normalize?

Undecidability

The following are undecidable problems:

- Given M and N, does $M \twoheadrightarrow_\beta N$ hold?
- Given M and N, does $M =_\beta N$ hold?
- Given M, does M normalize?
- Given M, does M strongly normalize?

The standard theory

Adding equational axioms

Example

Add the axiom $\mathbf{K} = \mathbf{S}$ to the equational theory of λ-calculus.

Adding equational axioms

Example

Add the axiom $\mathbf{K} = \mathbf{S}$ to the equational theory of λ-calculus. Then, for every M, one proves:

$$M = \mathbf{SI}(\mathbf{K}M)\mathbf{I} = \mathbf{KI}(\mathbf{K}M)\mathbf{I} = \mathbf{I}.$$

This extension is inconsistent.

Adding equational axioms

Example

Add the axiom $\mathbf{K} = \mathbf{S}$ to the equational theory of λ-calculus. Then, for every M, one proves:

$$M = \mathbf{SI}(\mathbf{K}M)\mathbf{I} = \mathbf{KI}(\mathbf{K}M)\mathbf{I} = \mathbf{I}.$$

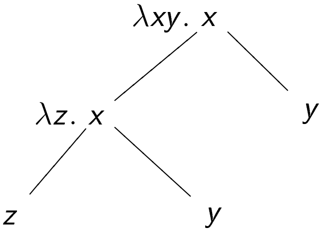
This extension is inconsistent.

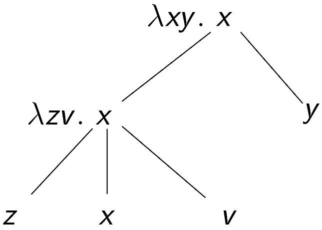
Böhm Theorem

*Let M, N be β-normal combinators with $M \neq_{\beta\eta} N$. Then $M\vec{P} =_\beta$ **true** and $N\vec{P} =_\beta$ **false**, for some $\vec{P}$.*

○

Böhm Trees (finite case)

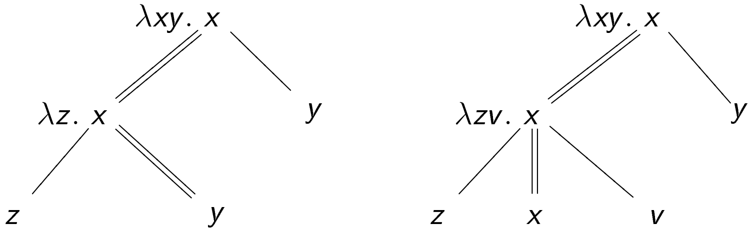




$$M = \lambda xy.x(\lambda z.xzy)y$$

$$N = \lambda xy.x(\lambda zv.xzxv)y$$

Böhm Trees: the difference



$M = \lambda xy.x(\lambda z.xzy)y$ $\qquad$ $N = \lambda xy.x(\lambda zv.xzxv)y$

Trick: Applying M to $\lambda uv.\langle u, v\rangle$ gives $\lambda y.\langle \lambda z.\langle z, y\rangle, y\rangle$.
And components can be extracted from a pair.

Discriminating terms

$M = \lambda xy.x(\lambda z.xzy)y$ $\qquad$ $N = \lambda xy.x(\lambda zv.xzxv)y$

Discriminating terms

$M = \lambda xy.x(\lambda z.xzy)y$ $\qquad$ $N = \lambda xy.x(\lambda zv.xzxv)y$

Applying M and N to $P = \lambda uv.\langle u, v\rangle$, then to any Q yields:

$\langle \lambda z.\langle z, Q\rangle, Q\rangle$ $\qquad$ $\langle \lambda zv.\langle z, P\rangle v, Q\rangle$

Discriminating terms

$M = \lambda xy.x(\lambda z.xzy)y$ $\qquad$ $N = \lambda xy.x(\lambda zv.xzxv)y$

Applying M and N to $P = \lambda uv.\langle u, v\rangle$, then to any Q yields:

$\langle \lambda z.\langle z, Q\rangle, Q\rangle$ $\qquad$ $\langle \lambda zv.\langle z, P\rangle v, Q\rangle$

Next appply both to **true**, **I**, **false** to obtain:

Q $\qquad$ $P = \lambda uv.\langle u, v\rangle$

Discriminating terms

$M = \lambda xy.x(\lambda z.xzy)y$ $\qquad$ $N = \lambda xy.x(\lambda zv.xzxv)y$

Applying M and N to $P = \lambda uv.\langle u, v\rangle$, then to any Q yields:

$\langle \lambda z.\langle z, Q\rangle, Q\rangle$ $\qquad$ $\langle \lambda zv.\langle z, P\rangle v, Q\rangle$

Next appply both to **true**, **I**, **false** to obtain:

Q $\qquad$ $P = \lambda uv.\langle u, v\rangle$

Choose $Q = \lambda uvw.$ **true** and apply both sides to **false**, **I**, **true**:

true $\qquad$ **false**.

The Meaning of “Value” and “Undefined”

First idea: *Value = Normal form.*
Undefined = without normal form.

The Meaning of “Value” and “Undefined”

First idea: *Value = Normal form.*
Undefined = without normal form.

Can we identify all such terms?

The Meaning of “Value” and “Undefined”

First idea: *Value = Normal form.*
Undefined = without normal form.

Can we identify all such terms?

No: for instance $\lambda x.x\mathbf{K}\mathbf{\Omega} = \lambda x.x\mathbf{S}\mathbf{\Omega}$ implies $\mathbf{K} = \mathbf{S}$ (apply both to $\mathbf{K}$).

The Meaning of “Value” and “Undefined”

First idea: *Value = Normal form.*
Undefined = without normal form.

Can we identify all such terms?

No: for instance $\lambda x.x\mathbf{K}\boldsymbol{\Omega} = \lambda x.x\mathbf{S}\boldsymbol{\Omega}$ implies $\mathbf{K} = \mathbf{S}$ (apply both to $\mathbf{K}$).

Moral: A term without normal form can still behave in a well-defined way. In a sense it has a „value”.

The Meaning of “Value” and “Undefined”

First idea: *Value = Normal form.*
Undefined = without normal form.

Can we identify all such terms?

No: for instance $\lambda x.x\mathbf{K}\boldsymbol{\Omega} = \lambda x.x\mathbf{S}\boldsymbol{\Omega}$ implies $\mathbf{K} = \mathbf{S}$ (apply both to $\mathbf{K}$).

Moral: A term without normal form can still behave in a well-defined way. In a sense it has a „value”.

Better idea: *Value = Head normal form.*
Undefined = without head normal form.

Solvability

A closed term is *solvable* iff $M\vec{P} =_\beta \mathbf{I}$, for some closed $\vec{P}$.

Solvability

A closed term is *solvable* iff $M\vec{P} =_\beta \mathbf{I}$, for some closed $\vec{P}$.

If $\mathsf{FV}(M) = \vec{x}$ then M is *solvable* iff $\lambda\vec{x}\, M$ is solvable.

Solvability

A closed term is *solvable* iff $M\vec{P} =_\beta \mathbf{I}$, for some closed $\vec{P}$.

If $\mathsf{FV}(M) = \vec{x}$ then M is *solvable* iff $\lambda\vec{x}\,M$ is solvable.

Theorem

A term is solvable iff it has a head normal form.

Solvability

A closed term is *solvable* iff $M\vec{P} =_\beta \mathbf{I}$, for some closed $\vec{P}$.

If $\mathsf{FV}(M) = \vec{x}$ then M is *solvable* iff $\lambda\vec{x}\,M$ is solvable.

Theorem

A term is solvable iff it has a head normal form.

Proof for closed terms:

$(\Rightarrow)$ If $M\vec{P} =_\beta \mathbf{I}$ then $M\vec{P} \twoheadrightarrow_\beta \mathbf{I}$. If $M\vec{P}$ head normalizes then also M must head normalize.

Solvability

A closed term is *solvable* iff $M\vec{P} =_\beta \mathbf{I}$, for some closed $\vec{P}$.

If $\mathrm{FV}(M) = \vec{x}$ then M is *solvable* iff $\lambda\vec{x}\,M$ is solvable.

Theorem

A term is solvable iff it has a head normal form.

Proof for closed terms:

($\Rightarrow$) If $M\vec{P} =_\beta \mathbf{I}$ then $M\vec{P} \twoheadrightarrow_\beta \mathbf{I}$. If $M\vec{P}$ head normalizes then also M must head normalize.

($\Leftarrow$) If $M =_\beta \lambda x_1 x_2 \ldots x_n.x_i R_1 \ldots R_m$ then $MP \ldots P = \mathbf{I}$, for $P = \lambda y_1 \ldots y_m.\mathbf{I}$.

○

The standard theory

We identify all unsolvable terms as “undefined”.

The standard theory

We identify all unsolvable terms as “undefined”.

Which solvable terms may be now be consistently identified?

The standard theory

We identify all unsolvable terms as “undefined”.

Which solvable terms may be now be consistently identified?

We cannot classify terms by their head normal forms.
Too many of them!

The standard theory

We identify all unsolvable terms as “undefined”.

Which solvable terms may be now be consistently identified?

We cannot classify terms by their head normal forms.
Too many of them!

We can only *observe* their behaviour.

○

Observational equivalence

Terms M, N with $\mathsf{FV}(M) \cup \mathsf{FV}(N) = \vec{x}$, are *observationally equivalent* ($M \equiv N$) when, for all closed P:

$$P(\lambda\vec{x}.M) \text{ is solvable} \quad \Longleftrightarrow \quad P(\lambda\vec{x}.N) \text{ is solvable}$$

Observational equivalence

Terms M, N with $\mathsf{FV}(M) \cup \mathsf{FV}(N) = \vec{x}$, are *observationally equivalent* ($M \equiv N$) when, for all closed P:

$$P(\lambda \vec{x}.M) \text{ is solvable} \iff P(\lambda \vec{x}.N) \text{ is solvable}$$

Put it differently:

$$C[M] \text{ is solvable} \iff C[N] \text{ is solvable}$$

Observational equivalence

Terms M, N with $\mathsf{FV}(M) \cup \mathsf{FV}(N) = \vec{x}$, are *observationally equivalent* ($M \equiv N$) when, for all closed P:

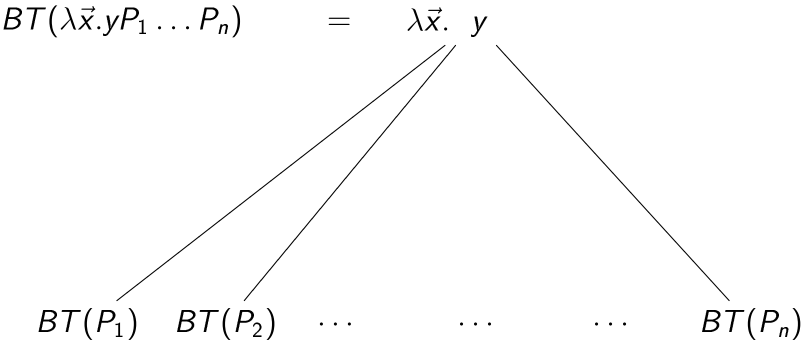
$$P(\lambda\vec{x}.M) \text{ is solvable} \iff P(\lambda\vec{x}.N) \text{ is solvable}$$

Put it differently:

$$C[M] \text{ is solvable} \iff C[N] \text{ is solvable}$$

Note: If $M =_\eta N$ then $M \equiv N$.

Böhm Trees



If M has a hnf N then $BT(M) = BT(N)$.

If M is unsolvable then $BT(M) = \bot$.

○

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

Write Φ for $\lambda fxy.\, x(fy))$. Then:

$\mathbf{J} = \mathbf{Y}\Phi =_\beta \Phi\mathbf{J}$

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

Write Φ for $\lambda fxy.\, x(fy))$. Then:

$\mathbf{J} = \mathbf{Y}\Phi =_\beta \Phi\mathbf{J} =_\beta \lambda xy.\, x(\mathbf{J}y)$

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

Write Φ for $\lambda fxy.\, x(fy))$. Then:

$$\mathbf{J} = \mathbf{Y}\Phi =_\beta \Phi\mathbf{J} =_\beta \lambda xy.\, x(\mathbf{J}y) =_\beta \lambda xy_0.\, x(\Phi\mathbf{J}y_0)$$

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

Write Φ for $\lambda fxy.\, x(fy))$. Then:

$$\mathbf{J} = \mathbf{Y}\Phi =_\beta \Phi\mathbf{J} =_\beta \lambda xy.\, x(\mathbf{J}y) =_\beta \lambda xy_0.\, x(\Phi\mathbf{J}y_0)$$
$$=_\beta \lambda xy_0.\, x(\lambda y_1.\, y_0(\mathbf{J}y_1))$$

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

Write Φ for $\lambda fxy.\, x(fy))$. Then:

$$\mathbf{J} = \mathbf{Y}\Phi =_\beta \Phi\mathbf{J} =_\beta \lambda xy.\, x(\mathbf{J}y) =_\beta \lambda xy_0.\, x(\Phi\mathbf{J}y_0)$$
$$=_\beta \lambda xy_0.\, x(\lambda y_1.\, y_0(\mathbf{J}y_1)) =_\beta \lambda xy_0.\, x(\lambda y_1.\, y_0(\Phi\mathbf{J}y_1)) =_\beta \ldots$$

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

Write Φ for $\lambda fxy.\, x(fy))$. Then:

$$\mathbf{J} = \mathbf{Y}\Phi =_\beta \Phi\mathbf{J} =_\beta \lambda xy.\, x(\mathbf{J}y) =_\beta \lambda xy_0.\, x(\Phi\mathbf{J}y_0)$$
$$=_\beta \lambda xy_0.\, x(\lambda y_1.\, y_0(\mathbf{J}y_1)) =_\beta \lambda xy_0.\, x(\lambda y_1.\, y_0(\Phi\mathbf{J}y_1)) =_\beta \ldots$$

The tree $BT(\mathbf{J})$ consists of one infinite path:

$$\lambda xy_0.\, x \text{———} \lambda y_1.\, y_0 \text{———} \lambda y_2.\, y_1 \text{———} \lambda y_3.\, y_2 \text{———} \cdots$$

○

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

The tree $BT(\mathbf{J})$ consists of one infinite path:

$$\lambda xy_0.\, x \text{———} \lambda y_1.\, y_0 \text{———} \lambda y_2.\, y_1 \text{———} \lambda y_3.\, y_2 \text{———} \cdots$$

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

The tree $BT(\mathbf{J})$ consists of one infinite path:

$$\lambda xy_0.\, x \text{———} \lambda y_1.\, y_0 \text{———} \lambda y_2.\, y_1 \text{———} \lambda y_3.\, y_2 \text{———} \cdots$$

The tree $BT(\mathbf{I})$ consists of a single node: $\lambda x\, x$

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

The tree $BT(\mathbf{J})$ consists of one infinite path:

$$\lambda xy_0.\, x \text{———} \lambda y_1.\, y_0 \text{———} \lambda y_2.\, y_1 \text{———} \lambda y_3.\, y_2 \text{———} \cdots$$

The tree $BT(\mathbf{I})$ consists of a single node: $\lambda x\, x$

The first can be obtained from the second by means of an infinite sequence of η-expansions:

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

The tree $BT(\mathbf{J})$ consists of one infinite path:

$$\lambda xy_0.\, x \text{——} \lambda y_1.\, y_0 \text{——} \lambda y_2.\, y_1 \text{——} \lambda y_3.\, y_2 \text{——} \cdots$$

The tree $BT(\mathbf{I})$ consists of a single node: $\lambda x\, x$

The first can be obtained from the second by means of an infinite sequence of η-expansions:

$$\lambda x\, x \quad {}_{\eta}\!\leftarrow \quad \lambda xy_0.\, x \text{——} y_0$$

Example: $\mathbf{J} = \mathbf{Y}(\lambda fxy.\, x(fy))$

The tree $BT(\mathbf{J})$ consists of one infinite path:

$$\lambda xy_0 .\, x \text{———} \lambda y_1 .\, y_0 \text{———} \lambda y_2 .\, y_1 \text{———} \lambda y_3 .\, y_2 \text{———} \cdots$$

The tree $BT(\mathbf{I})$ consists of a single node: $\lambda x\, x$

The first can be obtained from the second by means of an infinite sequence of η-expansions:

$$\lambda x\, x \quad {}_{\eta}\!\leftarrow \quad \lambda xy_0 .\, x \text{———} y_0 \quad {}_{\eta}\!\leftarrow \quad \lambda xy_0 .\, x \text{———} \lambda y_1 .\, y_0 \text{———} y_1$$

○

When are terms observationally equivalent?

Böhm trees B i B' are *η-equivalent* ($B \approx_\eta B'$), if there are two (possibly infinite) sequences of η-expansions:

$$B = B_0 \;{}_\eta\!\leftarrow B_1 \;{}_\eta\!\leftarrow B_2 \;{}_\eta\!\leftarrow B_3 \;{}_\eta\!\leftarrow \cdots$$

$$B' = B'_0 \;{}_\eta\!\leftarrow B'_1 \;{}_\eta\!\leftarrow B'_2 \;{}_\eta\!\leftarrow B'_3 \;{}_\eta\!\leftarrow \cdots$$

converging to the same (possibly infinite) tree.

When are terms observationally equivalent?

Böhm trees B i B' are *η-equivalent* ($B \approx_\eta B'$), if there are two (possibly infinite) sequences of η-expansions:

$$B = B_0 \;{}_\eta\!\leftarrow B_1 \;{}_\eta\!\leftarrow B_2 \;{}_\eta\!\leftarrow B_3 \;{}_\eta\!\leftarrow \cdots$$

$$B' = B'_0 \;{}_\eta\!\leftarrow B'_1 \;{}_\eta\!\leftarrow B'_2 \;{}_\eta\!\leftarrow B'_3 \;{}_\eta\!\leftarrow \cdots$$

converging to the same (possibly infinite) tree.

Theorem

Terms M and N are observationally equivalent if and only if $BT(M) \approx_\eta BT(N)$.

Semantics

Goal: Interpret any term M as an element $[\![M]\!]$ of some structure A, so that $M =_\beta N$ implies $[\![M]\!] = [\![N]\!]$.

Semantics

Goal: Interpret any term M as an element $[\![M]\!]$ of some structure A, so that $M =_\beta N$ implies $[\![M]\!] = [\![N]\!]$.

More precisely, $[\![M]\!]$ may depend on a *valuation*:

$$v : Var \rightarrow A.$$

Write $[\![M]\!]_v$, for the value of M under v.

Lambda-interpretation: $\mathcal{A} = \langle A, \cdot, [\![\ ]\!] \rangle$

Application $\cdot$ is a binary operation in A;

Lambda-interpretation: $\mathcal{A} = \langle A, \cdot, [\![\ ]\!] \rangle$

Application $\cdot$ is a binary operation in A;

$$[\![\ ]\!] : \Lambda \times A^{Var} \to A.$$

Write $[\![M]\!]_v$ instead of $[\![\ ]\!](M, v)$.

Lambda-interpretation: $\mathcal{A} = \langle A, \cdot, [\![\ ]\!] \rangle$

Application $\cdot$ is a binary operation in A;

$$[\![\ ]\!] : \Lambda \times A^{Var} \to A.$$

Write $[\![M]\!]_v$ instead of $[\![\ ]\!](M, v)$.

Postulates:

(a) $[\![x]\!]_v = v(x)$;

(b) $[\![PQ]\!]_v = [\![P]\!]_v \cdot [\![Q]\!]_v$;

(c) $[\![\lambda x.P]\!]_v \cdot a = [\![P]\!]_{v[x \mapsto a]}$, for any $a \in A$;

(d) If $v|_{\mathrm{FV}(P)} = u|_{\mathrm{FV}(P)}$, then $[\![P]\!]_v = [\![P]\!]_u$.

$\circ$

Extensionality

Write $a \approx b$ when $a \cdot c = b \cdot c$, for all c.

Extensionality

Write $a \approx b$ when $a \cdot c = b \cdot c$, for all c.

Extensional interpretation: $a \approx b$ implies $a = b$, for all a, b.

Extensionality

Write $a \approx b$ when $a \cdot c = b \cdot c$, for all c.

Extensional interpretation: $a \approx b$ implies $a = b$, for all a, b.

Weakly extensional interpretation:

$[\![\lambda x.M]\!]_v \approx [\![\lambda x.N]\!]_v$ implies $[\![\lambda x.M]\!]_v = [\![\lambda x.N]\!]_v$, for all N, v.

Extensionality

Write $a \approx b$ when $a \cdot c = b \cdot c$, for all c.

Extensional interpretation: $a \approx b$ implies $a = b$, for all a, b.

Weakly extensional interpretation:

$[\![\lambda x.M]\!]_v \approx [\![\lambda x.N]\!]_v$ implies $[\![\lambda x.M]\!]_v = [\![\lambda x.N]\!]_v$, for all N, v.

Meaning: Abstraction makes sense algebraically.

(N.B. $[\![\lambda x.M]\!]_v \approx [\![\lambda x.N]\!]_v$ iff $[\![M]\!]_{v[x \mapsto a]} = [\![N]\!]_{v[x \mapsto a]}$, all a.)

○

Lambda-model

Lambda-model: Weakly extensional lambda-interpretation:

$$[\![\lambda x.M]\!]_v \approx [\![\lambda x.N]\!]_v \quad \text{implies} \quad [\![\lambda x.M]\!]_v = [\![\lambda x.N]\!]_v$$

$\circ$

Very Important Lemma

Lemma

In every lambda-model,

$$[\![M[x := N]]\!]_v = [\![M]\!]_{v[x \mapsto [\![N]\!]_v]}.$$

Very Important Lemma

Lemma

In every lambda-model,

$$[\![M[x := N]]\!]_v = [\![M]\!]_{v[x \mapsto [\![N]\!]_v]}.$$

Proof: Induction wrt M. Case of λ with $x \notin \mathrm{FV}(N)$.

$[\![(\lambda y\, P)[x := N]]\!]_{v[x \mapsto [\![N]\!]_v]} \cdot a = [\![\lambda y . P[x := N]]\!]_v \cdot a$

$= [\![P[x := N]]\!]_{v[y \mapsto a]} = [\![P]\!]_{v[y \mapsto a][x \mapsto [\![N]\!]_{v[y \mapsto a]}]}$

$= [\![P]\!]_{v[y \mapsto a][x \mapsto [\![N]\!]_v]} = [\![\lambda y . P]\!]_{v[x \mapsto [\![N]\!]_v]} \cdot a$, for all a.

Therefore $[\![(\lambda y\, P)[x := N]]\!]_{v[x \mapsto [\![N]\!]_v]} = [\![(\lambda y . P)]\!]_{v[x \mapsto [\![N]\!]_v]}$.

○

Soundness

Proposition

Every lambda-model is a "lambda-algebra":

$$M =_\beta N \quad \textit{implies} \quad [\![M]\!]_v = [\![N]\!]_v$$

Proof: Induction wrt $M =_\beta N$. Non-immediate cases are two:

(Beta)

$$[\![(\lambda x.P)Q]\!]_v = [\![\lambda x.P]\!]_v \cdot [\![Q]\!]_v = [\![P]\!]_{v[x \mapsto [\![Q]\!]_v]} = [\![P[x := Q]]\!]_v.$$

Soundness

Proposition

Every lambda-model is a "lambda-algebra":

$$M =_\beta N \quad \textit{implies} \quad [\![M]\!]_v = [\![N]\!]_v$$

Proof: Induction wrt $M =_\beta N$. Non-immediate cases are two:

(Beta)

$[\![(\lambda x.P)Q]\!]_v = [\![\lambda x.P]\!]_v \cdot [\![Q]\!]_v = [\![P]\!]_{v[x \mapsto [\![Q]\!]_v]} = [\![P[x := Q]]\!]_v$.

(Xi)

Let $P =_\beta Q$ and let $M = \lambda x.P$, $N = \lambda x.Q$. Then

$[\![M]\!]_v \cdot a = [\![P]\!]_{v[x \mapsto a]} = [\![Q]\!]_{v[x \mapsto a]} = [\![N]\!]_v \cdot a$, for all a.

○

Completeness

Theorem

The following are equivalent:

1) $M =_\beta N$;
2) $\mathcal{A} \models M = N$, *for every lambda-model* $\mathcal{A}$.

Proof.

(1)$\Rightarrow$(2) By soundness.

(2)$\Rightarrow$(1) Because term model is a lambda-model. □

Complete partial orders

Let $\langle A, \leq \rangle$ be a partial order.

A subset $B \subseteq A$ is *directed* when for every $a, b \in B$ there is $c \in B$ with $a, b \leq c$.

Complete partial orders

Let $\langle A, \leq \rangle$ be a partial order.

A subset $B \subseteq A$ is *directed* when for every $a, b \in B$ there is $c \in B$ with $a, b \leq c$.

The set A is a *complete partial order (cpo)* when every directed subset has a supremum.

Complete partial orders

Let $\langle A, \leq \rangle$ be a partial order.

A subset $B \subseteq A$ is *directed* when for every $a, b \in B$ there is $c \in B$ with $a, b \leq c$.

The set A is a *complete partial order (cpo)* when every directed subset has a supremum.

It follows that every cpo has a least element $\bot = \sup \varnothing$.

Complete partial orders

Let $\langle A, \leq \rangle$ and $\langle B, \leq \rangle$ be cpos, and $f : A \rightarrow B$.

Then f is *monotone* if $a \leq a'$ implies $f(a) \leq f(a')$.

Complete partial orders

Let $\langle A, \leq \rangle$ and $\langle B, \leq \rangle$ be cpos, and $f : A \to B$.

Then f is *monotone* if $a \leq a'$ implies $f(a) \leq f(a')$.

And f is *continuous* if $\sup f(C) = f(\sup C)$
for every **nonempty** directed $C \subseteq A$.

Complete partial orders

Let $\langle A, \leq \rangle$ and $\langle B, \leq \rangle$ be cpos, and $f : A \to B$.

Then f is *monotone* if $a \leq a'$ implies $f(a) \leq f(a')$.

And f is *continuous* if $\sup f(C) = f(\sup C)$
for every **nonempty** directed $C \subseteq A$.

Fact: *Every continuous function is monotone.*

Complete partial orders

Let $\langle A, \leq \rangle$ and $\langle B, \leq \rangle$ be cpos, and $f : A \to B$.

Then f is *monotone* if $a \leq a'$ implies $f(a) \leq f(a')$.

And f is *continuous* if $\sup f(C) = f(\sup C)$
for every **nonempty** directed $C \subseteq A$.

Fact: *Every continuous function is monotone.*

$[A \to B]$ is the set of all continuous functions from A to B

○

Complete partial orders

If $\langle A, \leq \rangle$ and $\langle B, \leq \rangle$ are cpos then:

- The product $A \times B$ is a cpo with
  $$\langle a, b \rangle \leq \langle a', b' \rangle \text{ iff } a \leq a' \text{ and } b \leq b'.$$
- The function space $[A \rightarrow B]$ is a cpo with
  $$f \leq g \text{ iff } \forall a.\, f(a) \leq g(a).$$

○

Continuous functions

Lemma

A function $f : A \times B \to C$ *is continuous iff it is continuous wrt both arguments, i.e. all functions of the form* $\lambda a.\, f(a, b)$ *and* $\lambda b.\, f(a, b)$ *are continuous.*

Continuous functions

Lemma

A function $f : A \times B \to C$ is continuous iff it is continuous wrt both arguments, i.e. all functions of the form $\lambda a.\, f(a, b)$ and $\lambda b.\, f(a, b)$ are continuous.

Proof.

($\Leftarrow$) Take $X \subseteq A \times B$ directed. Let $X_i = \pi_i(X)$ for $i = 1, 2$.

Continuous functions

Lemma

A function $f : A \times B \to C$ is continuous iff it is continuous wrt both arguments, i.e. all functions of the form $\lambda\!\!\lambda a.\, f(a,b)$ and $\lambda\!\!\lambda b.\, f(a,b)$ are continuous.

Proof.

($\Leftarrow$) Take $X \subseteq A \times B$ directed. Let $X_i = \pi_i(X)$ for $i = 1, 2$.

Step 1: If $\langle a, b\rangle \in X_1 \times X_2$ then $\langle a, b\rangle \leq \langle a', b'\rangle \in X$.

Continuous functions

Lemma

A function $f : A \times B \to C$ is continuous iff it is continuous wrt both arguments, i.e. all functions of the form $\lambda a.\, f(a,b)$ and $\lambda b.\, f(a,b)$ are continuous.

Proof.

($\Leftarrow$) Take $X \subseteq A \times B$ directed. Let $X_i = \pi_i(X)$ for $i = 1, 2$.

Step 1: If $\langle a, b \rangle \in X_1 \times X_2$ then $\langle a, b \rangle \leq \langle a', b' \rangle \in X$.

Step 2: Therefore $\sup X = \langle \sup X_1, \sup X_2 \rangle = \langle a_0, b_0 \rangle$.
We show that $\langle f(a_0), f(b_0) \rangle$ is the supremum of $f(X)$.

Continuous functions

Lemma

A function $f : A \times B \to C$ is continuous iff it is continuous wrt both arguments, i.e. all functions of the form $\lambda a.\, f(a,b)$ and $\lambda b.\, f(a,b)$ are continuous.

Proof.

($\Leftarrow$) Take $X \subseteq A \times B$ directed. Let $X_i = \pi_i(X)$ for $i = 1, 2$.

Step 1: If $\langle a, b\rangle \in X_1 \times X_2$ then $\langle a, b\rangle \leq \langle a', b'\rangle \in X$.

Step 2: Therefore $\sup X = \langle \sup X_1, \sup X_2\rangle = \langle a_0, b_0\rangle$.
We show that $\langle f(a_0), f(b_0)\rangle$ is the supremum of $f(X)$.
Let $c \geq f(X)$, then $c \geq f\langle a, b\rangle$ for all $\langle a, b\rangle \in X_1 \times X_2$.

Continuous functions

Lemma

A function $f : A \times B \to C$ is continuous iff it is continuous wrt both arguments, i.e. all functions of the form $\lambda a.\, f(a,b)$ and $\lambda b.\, f(a,b)$ are continuous.

Proof.

($\Leftarrow$) Take $X \subseteq A \times B$ directed. Let $X_i = \pi_i(X)$ for $i = 1, 2$.

Step 1: If $\langle a, b\rangle \in X_1 \times X_2$ then $\langle a, b\rangle \leq \langle a', b'\rangle \in X$.

Step 2: Therefore $\sup X = \langle \sup X_1, \sup X_2\rangle = \langle a_0, b_0\rangle$.
We show that $\langle f(a_0), f(b_0)\rangle$ is the supremum of $f(X)$.
Let $c \geq f(X)$, then $c \geq f\langle a, b\rangle$ for all $\langle a, b\rangle \in X_1 \times X_2$.
Fix a, to get $c \geq \sup_b f(a, b) = f(a, b_0)$.

Continuous functions

Lemma

A function $f : A \times B \to C$ is continuous iff it is continuous wrt both arguments, i.e. all functions of the form $\lambda a.\, f(a, b)$ and $\lambda b.\, f(a, b)$ are continuous.

Proof.

($\Leftarrow$) Take $X \subseteq A \times B$ directed. Let $X_i = \pi_i(X)$ for $i = 1, 2$.

Step 1: If $\langle a, b\rangle \in X_1 \times X_2$ then $\langle a, b\rangle \leq \langle a', b'\rangle \in X$.

Step 2: Therefore $\sup X = \langle \sup X_1, \sup X_2\rangle = \langle a_0, b_0\rangle$.
We show that $\langle f(a_0), f(b_0)\rangle$ is the supremum of $f(X)$.
Let $c \geq f(X)$, then $c \geq f\langle a, b\rangle$ for all $\langle a, b\rangle \in X_1 \times X_2$.
Fix a, to get $c \geq \sup_b f(a, b) = f(a, b_0)$.
Fix b_0, to get $c \geq \sup_a f(a, b_0) = f(a_0, b_0)$. $\square$

Continuous functions

Lemma

The application $App : [A \to B] \times A \to B$ *is continuous.*

Proof: Uses the previous lemma.

Lemma

The abstraction $Abs : [(A \times B) \to C] \to [A \to [B \to C]]$, *given by* $Abs(F)(a)(b) = F(a, b)$, *is continuous.*

Reflexive cpo

The cpo D is *reflexive* iff there are continuous functions $F : D \to [D \to D]$ and $G : [D \to D] \to D$,

with $F \circ G = \mathrm{id}_{[D \to D]}$.

Reflexive cpo

The cpo D is *reflexive* iff there are continuous functions
$F : D \rightarrow [D \rightarrow D]$ and $G : [D \rightarrow D] \rightarrow D$,
with $F \circ G = \mathrm{id}_{[D \rightarrow D]}$.

Then F must be onto and G is injective.

Reflexive cpo

The cpo D is *reflexive* iff there are continuous functions
$F : D \to [D \to D]$ and $G : [D \to D] \to D$,
with $F \circ G = \mathrm{id}_{[D \to D]}$.

Then F must be onto and G is injective.

The following are equivalent conditions:

"$G \circ F = \mathrm{id}_D$", "$G$ onto", "F injective".

∘

Reflexive cpo

$$F : D \to [D \to D], \quad G : [D \to D] \to D, \quad F \circ G = \mathrm{id}.$$

Reflexive cpo

$$F : D \to [D \to D], \quad G : [D \to D] \to D, \quad F \circ G = \mathrm{id}.$$

Define application as $a \cdot b = F(a)(b)$ so that $G(f) \cdot a = f(a)$.

Reflexive cpo

$$F : D \to [D \to D], \quad G : [D \to D] \to D, \quad F \circ G = \mathrm{id}.$$

Define application as $a \cdot b = F(a)(b)$ so that $G(f) \cdot a = f(a)$.

Define interpretation as

- $[\![x]\!]_v = v(x)$;
- $[\![PQ]\!]_v = [\![P]\!]_v \cdot [\![Q]\!]_v$;
- $[\![\lambda x.P]\!]_v = G(\lambda\!\!\lambda a.[\![P]\!]_{v[x \mapsto a]})$.

Reflexive cpo

$$F : D \to [D \to D], \quad G : [D \to D] \to D, \quad F \circ G = \mathrm{id}.$$

Define application as $a \cdot b = F(a)(b)$ so that $G(f) \cdot a = f(a)$.

Define interpretation as

- $[\![x]\!]_v = v(x)$;
- $[\![PQ]\!]_v = [\![P]\!]_v \cdot [\![Q]\!]_v$;
- $[\![\lambda x.P]\!]_v = G(\lambda\!\!\lambda a.[\![P]\!]_{v[x \mapsto a]})$.

Fact: This is a (well-defined) lambda interpretation.
(Use continuity of *App* and *Abs*.)

○

Reflexive cpo

Theorem

A reflexive cpo is a lambda-model.

Reflexive cpo

Theorem

A reflexive cpo is a lambda-model.

Proof.

Prove weak extensionality: let $[\![\lambda x.M]\!]_v \cdot a = [\![\lambda x.N]\!]_v \cdot a$, all a. Note that $[\![\lambda x.M]\!]_v \cdot a = G(\lambda\!\!\lambda a.[\![M]\!]_{v[x\mapsto a]}) \cdot a = [\![M]\!]_{v[x\mapsto a]}$, and thus $\lambda\!\!\lambda a.[\![M]\!]_{v[x\mapsto a]} = \lambda\!\!\lambda a.[\![N]\!]_{v[x\mapsto a]}$. By the injectivity of G, it follows that $[\![\lambda x.M]\!]_v = [\![\lambda x.N]\!]_v$. □