# The Beta Cube

Álvaro García[1,2] Pablo Nogueira[2] Emilio Jesús Gallego Arias[2]

[1]IMDEA Software Institute

[2]Babel Research Group, Universidad Politécnica de Madrid

ITU 2011, Kobenham

# Context

- Reduction strategies for the pure (untyped) lambda calculus...

# Context

- Reduction strategies for the pure (untyped) lambda calculus...
  - Normal Order, Applicative (Standard) Order [Barendregt 1984].

# Context

- Reduction strategies for the pure (untyped) lambda calculus. . .
  - Normal Order, Applicative (Standard) Order [Barendregt 1984].
  - Call-by-name, Call-by-value [Plotkin 1975] [Sestoft 2002].

# Context

- Reduction strategies for the pure (untyped) lambda calculus. . .
  - Normal Order, Applicative (Standard) Order [Barendregt 1984].
  - Call-by-name, Call-by-value [Plotkin 1975] [Sestoft 2002].
  - Head spine (he) [Sestoft 2002] (headNF in [Paulson 1996]).

# Context

- Reduction strategies for the pure (untyped) lambda calculus...
  - Normal Order, Applicative (Standard) Order [Barendregt 1984].
  - Call-by-name, Call-by-value [Plotkin 1975] [Sestoft 2002].
  - Head spine (he) [Sestoft 2002] (headNF in [Paulson 1996]).
  - Hybrid applicative order (ha) [Sestoft 2002].

# Context

- Reduction strategies for the pure (untyped) lambda calculus...
  - Normal Order, Applicative (Standard) Order [Barendregt 1984].
  - Call-by-name, Call-by-value [Plotkin 1975] [Sestoft 2002].
  - Head spine (he) [Sestoft 2002] (headNF in [Paulson 1996]).
  - Hybrid applicative order (ha) [Sestoft 2002].
  - Hybrid normal order (hn) [Sestoft 2002].

# Context

- Reduction strategies for the pure (untyped) lambda calculus...
  - Normal Order, Applicative (Standard) Order [Barendregt 1984].
  - Call-by-name, Call-by-value [Plotkin 1975] [Sestoft 2002].
  - Head spine (he) [Sestoft 2002] (headNF in [Paulson 1996]).
  - Hybrid applicative order (ha) [Sestoft 2002].
  - Hybrid normal order (hn) [Sestoft 2002].
  - Head reduction (hr) [Barendregt 1984].

# Context

- Reduction strategies for the pure (untyped) lambda calculus...
  - Normal Order, Applicative (Standard) Order [Barendregt 1984].
  - Call-by-name, Call-by-value [Plotkin 1975] [Sestoft 2002].
  - Head spine (he) [Sestoft 2002] (headNF in [Paulson 1996]).
  - Hybrid applicative order (ha) [Sestoft 2002].
  - Hybrid normal order (hn) [Sestoft 2002].
  - Head reduction (hr) [Barendregt 1984].
  - ...

# Context

- Reduction strategies for the pure (untyped) lambda calculus...
  - Normal Order, Applicative (Standard) Order [Barendregt 1984].
  - Call-by-name, Call-by-value [Plotkin 1975] [Sestoft 2002].
  - Head spine (he) [Sestoft 2002] (headNF in [Paulson 1996]).
  - Hybrid applicative order (ha) [Sestoft 2002].
  - Hybrid normal order (hn) [Sestoft 2002].
  - Head reduction (hr) [Barendregt 1984].
  - ...
- ...defined by sets of big-step rules.

# What are strategies useful for?

- Program optimization via partial evaluation.

# What are strategies useful for?

- Program optimization via partial evaluation.
- $\beta$-equivalence testers for typing rules in dependent types.

# What are strategies useful for?

- Program optimization via partial evaluation.
- $\beta$-equivalence testers for typing rules in dependent types.
- Interpreting universes in structural generic programming with dependent types.

# What are strategies useful for?

- Program optimization via partial evaluation.
- $\beta$-equivalence testers for typing rules in dependent types.
- Interpreting universes in structural generic programming with dependent types.
- ...

# Pure lambda calculus reduction strategies (big-step)

Call-by-name (cbn):

$$\frac{}{x \stackrel{cbn}{\rightarrow} x}$$

$$\frac{}{\lambda x.B \stackrel{cbn}{\rightarrow} \lambda x.B}$$

$$\frac{M \stackrel{cbn}{\rightarrow} M' \equiv \lambda x.B \qquad [N/x]B \stackrel{cbn}{\rightarrow} S}{M\,N \stackrel{cbn}{\rightarrow} S}$$

$$\frac{M \stackrel{cbn}{\rightarrow} M' \not\equiv \lambda x.B}{M\,N \stackrel{cbn}{\rightarrow} M'\,N}$$

# Pure lambda calculus reduction strategies (big-step)

Call-by-name (cbn):

$$\frac{}{x \stackrel{cbn}{\to} x}$$

$$\frac{}{\lambda x.B \stackrel{cbn}{\to} \lambda x.B}$$

$$\frac{M \stackrel{cbn}{\to} M' \equiv \lambda x.B \qquad [N/x]B \stackrel{cbn}{\to} S}{M\,N \stackrel{cbn}{\to} S}$$

$$\frac{M \stackrel{cbn}{\to} M' \not\equiv \lambda x.B}{M\,N \stackrel{cbn}{\to} M'\,N}$$

Subsidiary

Normal order (nor):

$$\frac{}{x \stackrel{nor}{\to} x}$$

$$\frac{B \stackrel{nor}{\to} B'}{\lambda x.B \stackrel{nor}{\to} \lambda x.B'}$$

$$\frac{\boxed{M \stackrel{cbn}{\to} M' \equiv \lambda x.B} \qquad [N/x]B \stackrel{nor}{\to} S}{M\,N \stackrel{nor}{\to} S}$$

$$\frac{\boxed{M \stackrel{cbn}{\to} M' \not\equiv \lambda x.B} \qquad M' \stackrel{nor}{\to} M'' \qquad N \stackrel{nor}{\to} N''}{M\,N \stackrel{nor}{\to} M''\,N''}$$

Hybrid

# Pure lambda calculus reduction strategies (big-step)

Call-by-name (cbn):

$$\frac{}{x \overset{cbn}{\to} x}$$

$$\frac{}{\lambda x.B \overset{cbn}{\to} \lambda x.B}$$

$$\frac{M \overset{cbn}{\to} M' \equiv \lambda x.B \qquad [N/x]B \overset{cbn}{\to} S}{M\,N \overset{cbn}{\to} S}$$

$$\frac{M \overset{cbn}{\to} M' \not\equiv \lambda x.B}{M\,N \overset{cbn}{\to} M'\,N}$$

Subsidiary

Normal order (nor):

$$\frac{}{x \overset{nor}{\to} x}$$

$$\frac{\boxed{B \overset{nor}{\to} B'}}{\lambda x.B \overset{nor}{\to} \lambda x.B'}$$

$$\frac{\boxed{M \overset{cbn}{\to} M' \equiv \lambda x.B} \qquad [N/x]B \overset{nor}{\to} S}{M\,N \overset{nor}{\to} S}$$

$$\frac{\boxed{M \overset{cbn}{\to} M' \not\equiv \lambda x.B} \qquad M' \overset{nor}{\to} M'' \qquad \boxed{N \overset{nor}{\to} N''}}{M\,N \overset{nor}{\to} M''\,N''}$$

Hybrid

Hybrid reduces in more places than subsidiary!

# Pure lambda calculus reduction strategies (big-step)

Rule Template:

$$\text{VAR}\ \frac{}{x \xrightarrow{st} x} \qquad \text{ABS}\ \frac{B \xrightarrow{la} B'}{\lambda x.B \xrightarrow{st} \lambda x.B'}$$

$$\text{RED}\ \frac{M \xrightarrow{op_1} M' \equiv \lambda x.B \qquad N \xrightarrow{ar_1} N' \qquad [N'/x]B \xrightarrow{su} S}{M\,N \xrightarrow{st} S}$$

$$\text{APP}\ \frac{M \xrightarrow{op_1} M' \not\equiv \lambda x.B \qquad M' \xrightarrow{op_2} M'' \qquad N \xrightarrow{ar_2} N'}{M\,N \xrightarrow{st} M''\,N'}$$

# Pure lambda calculus reduction strategies (big-step)

Rule Template:

$$\text{VAR}\ \frac{}{x \xrightarrow{st} x} \qquad \text{ABS}\ \frac{B \xrightarrow{la} B'}{\lambda x.B \xrightarrow{st} \lambda x.B'}$$

$$\text{RED}\ \frac{M \xrightarrow{op_1} M' \equiv \lambda x.B \qquad N \xrightarrow{ar_1} N' \qquad [N'/x]B \xrightarrow{su} S}{M\,N \xrightarrow{st} S}$$

$$\text{APP}\ \frac{M \xrightarrow{op_1} M' \not\equiv \lambda x.B \qquad M' \xrightarrow{op_2} M'' \qquad N \xrightarrow{ar_2} N'}{M\,N \xrightarrow{st} M''\,N'}$$

# Pure lambda calculus reduction strategies (big-step)

Rule Template (cbn):

$$\text{VAR}\ \frac{}{x \xrightarrow{cbn} x} \qquad \text{ABS}\ \frac{B \xrightarrow{id} B}{\lambda x.B \xrightarrow{cbn} \lambda x.B}$$

$$\text{RED}\ \frac{M \xrightarrow{cbn} M' \equiv \lambda x.B \qquad N \xrightarrow{id} N \qquad [N\,/x]B \xrightarrow{cbn} S}{M\,N \xrightarrow{cbn} S}$$

$$\text{APP}\ \frac{M \xrightarrow{cbn} M' \not\equiv \lambda x.B \qquad M' \xrightarrow{id} M' \qquad N \xrightarrow{id} N}{M\,N \xrightarrow{cbn} M'\ N}$$

# Pure lambda calculus reduction strategies (big-step)

Rule Template (cbv):

$$\textrm{VAR}\ \frac{}{x \overset{cbv}{\rightarrow} x} \qquad \textrm{ABS}\ \frac{B \overset{id}{\rightarrow} B}{\lambda x.B \overset{cbv}{\rightarrow} \lambda x.B}$$

$$\textrm{RED}\ \frac{M \overset{cbv}{\rightarrow} M' \equiv \lambda x.B \qquad N \overset{cbv}{\rightarrow} N' \qquad [N'/x]B \overset{cbv}{\rightarrow} S}{M\,N \overset{cbv}{\rightarrow} S}$$

$$\textrm{APP}\ \frac{M \overset{cbv}{\rightarrow} M' \not\equiv \lambda x.B \qquad M' \overset{id}{\rightarrow} M' \qquad N \overset{cbv}{\rightarrow} N'}{M\,N \overset{cbv}{\rightarrow} M'\ N'}$$

# Pure lambda calculus reduction strategies (big-step)

Rule Template (aor):

$$\text{VAR}\ \frac{}{x \overset{aor}{\longrightarrow} x} \qquad \text{ABS}\ \frac{B \overset{aor}{\longrightarrow} B'}{\lambda x.B \overset{aor}{\longrightarrow} \lambda x.B'}$$

$$\text{RED}\ \frac{M \overset{aor}{\longrightarrow} M' \equiv \lambda x.B \qquad N \overset{aor}{\longrightarrow} N' \qquad [N'/x]B \overset{aor}{\longrightarrow} S}{M\,N \overset{aor}{\longrightarrow} S}$$

$$\text{APP}\ \frac{M \overset{aor}{\longrightarrow} M' \not\equiv \lambda x.B \qquad M' \overset{id}{\longrightarrow} M' \qquad N \overset{aor}{\longrightarrow} N'}{M\,N \overset{aor}{\longrightarrow} M'\ N'}$$

# Pure lambda calculus reduction strategies (big-step)

Rule Template (nor):

$$\text{VAR}\ \frac{}{x \xrightarrow{nor} x} \qquad \text{ABS}\ \frac{B \xrightarrow{nor} B'}{\lambda x.B \xrightarrow{nor} \lambda x.B'}$$

$$\text{RED}\ \frac{M \xrightarrow{cbn} M' \equiv \lambda x.B \qquad N \xrightarrow{id} N \qquad [N\ /x]B \xrightarrow{nor} S}{M\,N \xrightarrow{nor} S}$$

$$\text{APP}\ \frac{M \xrightarrow{cbn} M' \not\equiv \lambda x.B \qquad M' \xrightarrow{nor} M'' \qquad N \xrightarrow{nor} N'}{M\,N \xrightarrow{nor} M''\,N'}$$

Use of $op_1$ and $op_2$ to accomodate hybrid strategies!

# The Beta Cube

Parameters *la*, $ar1$ and $ar2$ are either recursive calls or identity.
Interpreted as boolean switches:

# The Beta Cube

Parameters *la*, *ar*1 and *ar*2 are either recursive calls or identity.
Interpreted as boolean switches:

$$(\sqsubseteq)\quad
\begin{tikzcd}
\mathcal{AOR} & & \mathcal{CBV} \arrow[ll] & \\
& 1\;0\;1 \arrow[ul] & & 0\;0\;1 \arrow[ll] \arrow[ul] \arrow[uu, crossing over] \\
1\;1\;0 \arrow[uu] & & 0\;1\;0 \arrow[ll] \arrow[uu] & \\
& \mathcal{HE} \arrow[ul] \arrow[uu, crossing over] & & \mathcal{CBN} \arrow[ll] \arrow[ul] \arrow[uu]
\end{tikzcd}
\qquad
\begin{tikzcd}
\; & \arrow[l, "la"'] \\
\; & \arrow[ul, "ar_1"'] \\
\; \arrow[from=d, "ar_2"'] & \\
\; &
\end{tikzcd}$$

# The Beta Cube

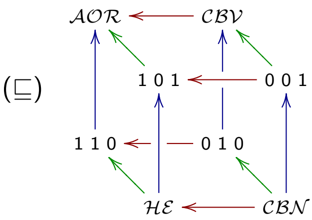
Parameters *la*, *ar*1 and *ar*2 are either recursive calls or identity.
Interpreted as boolean switches:

$$
(\sqsubseteq) \qquad
\begin{tikzcd}
\mathcal{AOR} & & \mathcal{CBV} \arrow[ll] & \\
& 1\ 0\ 1 \arrow[ul] & & 0\ 0\ 1 \arrow[ll] \arrow[ul] \\
1\ 1\ 0 \arrow[uu] & & 0\ 1\ 0 \arrow[ll] \arrow[uu] & \\
& \mathcal{HE} \arrow[ul] \arrow[uu] & & \mathcal{CBN} \arrow[ll] \arrow[ul] \arrow[uu]
\end{tikzcd}
$$

| | | |
|---|---|---|
| $\xleftarrow{la}$ | 0 weakness | 1 strength |
| $\nwarrow$ $ar_1$ | 0 non-strictness | 1 strictness |
| $\uparrow$ $ar_2$ | 0 headness | 1 non-headness |

# The Beta Cube

Parameters *la*, *ar*1 and *ar*2 are either recursive calls or identity.
Interpreted as boolean switches:




| | | |
|---|---|---|
| $\xleftarrow{la}$ | 0 weakness | 1 strength |
| $ar_1$ | 0 non-strictness | 1 strictness |
| $\uparrow ar_2$ | 0 headness | 1 non-headness |

$$\begin{tikzcd} nf & wnf \arrow[l] \\ hnf \arrow[u] & whnf \arrow[l] \arrow[u] \end{tikzcd}$$

# Axis of eval

$$\text{VAR}\ \frac{}{x \overset{cbn}{\longrightarrow} x} \qquad \text{ABS}\ \frac{}{\lambda x.B \overset{cbn}{\longrightarrow} \lambda x.B}$$

$$\text{RED}\ \frac{M \overset{cbn}{\longrightarrow} M' \equiv \lambda x.B \qquad [N\ /x]B \overset{cbn}{\longrightarrow} S}{M\ N \overset{cbn}{\longrightarrow} S}$$

$$\text{APP}\ \frac{M \overset{cbn}{\longrightarrow} M' \not\equiv \lambda x.B}{M\ N \overset{cbn}{\longrightarrow} M'\ N}$$

$$\begin{tikzcd}
\mathcal{AOR} & & \mathcal{CBV} \arrow[ll] & \\
& 1\ 0\ 1 \arrow[ul] & & 0\ 0\ 1 \arrow[ll] \arrow[ul] \\
1\ 1\ 0 \arrow[uu] & & 0\ 1\ 0 \arrow[ll] \arrow[uu] & \\
& \mathcal{HE} \arrow[uu] \arrow[ul] & & \boxed{\mathcal{CBN}} \arrow[ll] \arrow[uu] \arrow[ul]
\end{tikzcd}$$

$\xleftarrow{la}$ strength

$\nwarrow ar_1$ strictness

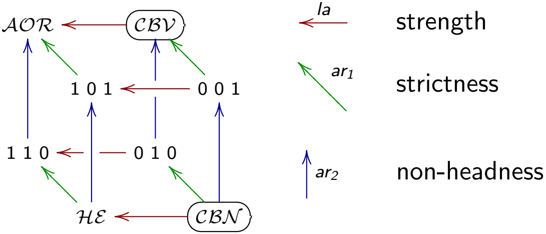
$\uparrow ar_2$ non-headness

# Axis of eval

$$\text{VAR}\ \frac{}{x \overset{cbv}{\to} x} \qquad \text{ABS}\ \frac{}{\lambda x.B \overset{cbv}{\to} \lambda x.B}$$

$$\text{RED}\ \frac{M \overset{cbv}{\to} M' \equiv \lambda x.B \qquad {\color{green} N \overset{cbv}{\to} N'} \qquad [N'/x]B \overset{cbv}{\to} S}{M\,N \overset{cbv}{\to} S}$$

$$\text{APP}\ \frac{M \overset{cbv}{\to} M' \not\equiv \lambda x.B \qquad {\color{blue} N \overset{cbv}{\to} N'}}{M\,N \overset{cbv}{\to} M'\,N'}$$

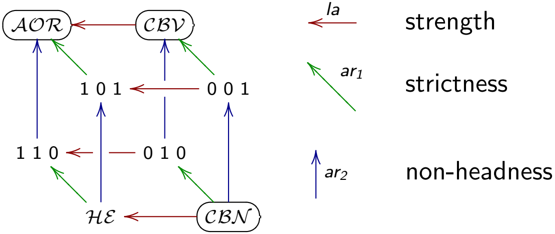


# Axis of eval

$$\text{VAR}\ \frac{}{x \xrightarrow{aor} x} \qquad\qquad \text{ABS}\ \frac{B \xrightarrow{aor} B'}{\lambda x.B \xrightarrow{aor} \lambda x.B'}$$

$$\text{RED}\ \frac{M \xrightarrow{aor} M' \equiv \lambda x.B \qquad N \xrightarrow{aor} N' \qquad [N'/x]B \xrightarrow{aor} S}{M\,N \xrightarrow{aor} S}$$

$$\text{APP}\ \frac{M \xrightarrow{aor} M' \not\equiv \lambda x.B \qquad N \xrightarrow{aor} N'}{M\,N \xrightarrow{aor} M'\,N'}$$



# Absorption

- Applying $s_2$ before applying $s_1$ doesn't change the result of $s_1$:

$$s_1 \text{ absorpts } s_2 \text{ iff } s_1(t) =_\alpha s_1(s_2(t)).$$

# Absorption

- Applying $s_2$ before applying $s_1$ doesn't change the result of $s_1$:

$$s_1 \text{ absorpts } s_2 \text{ iff } s_1(t) =_\alpha s_1(s_2(t)).$$

- $s_1$ absorpts $s_2$ iff $s_2$ is a left identity of $s_1$.

$$t \xrightarrow{s_1} t' \quad \text{iff} \quad t \xrightarrow{s_2} t'' \xrightarrow{s_1} t'$$

$$t \overset{s_1}{\not\rightarrow} \quad \text{iff} \quad \begin{cases} \quad t \overset{s_2}{\not\rightarrow} \\ \text{or} \\ \quad t \xrightarrow{s_2} t'' \overset{s_1}{\not\rightarrow} \end{cases}$$

# Absorption among uniform strategies

- We analyse the pairs of strategies in the order relation.

# Absorption among uniform strategies

- We analyse the pairs of strategies in the order relation.
- Counterexamples:

# Absorption among uniform strategies

- We analyse the pairs of strategies in the order relation.
- Counterexamples:
  - Any strict $st_s$ and non-strict $st$ strategies (differing at least in $ar_1$):

$$(st_s \circ st)((\lambda x.\lambda y.x)\, x\, \Omega) \neq_\alpha st_s((\lambda x.\lambda y.x)\, x\, \Omega).$$

# Absorption among uniform strategies

- We analyse the pairs of strategies in the order relation.
- Counterexamples:
  - Any strict $st_s$ and non-strict $st$ strategies (differing at least in $ar_1$):

$$(st_s \circ st)((\lambda x.\lambda y.x)\,x\,\Omega) \neq_\alpha st_s((\lambda x.\lambda y.x)\,x\,\Omega).$$

  - Any strong non-head strategy $st$ and its weak or head (or weak-head) counterpart $st_{wh}$ (differing in $la$ or $ar_2$ or both):

$$(st \circ st_{wh})((\lambda k.k\,\Omega)\,(\lambda x.y)) \neq_\alpha st((\lambda k.k\,\Omega)\,(\lambda x.y)).$$

# Absorption among uniform strategies

- We analyse the pairs of strategies in the order relation.
- Counterexamples:
  - Any strict $st_s$ and non-strict $st$ strategies (differing at least in $ar_1$):

    $$(st_s \circ st)((\lambda x.\lambda y.x)\,x\,\Omega) \neq_\alpha st_s((\lambda x.\lambda y.x)\,x\,\Omega).$$

  - Any strong non-head strategy $st$ and its weak or head (or weak-head) counterpart $st_{wh}$ (differing in $la$ or $ar_2$ or both):

    $$(st \circ st_{wh})((\lambda k.k\,\Omega)\,(\lambda x.y)) \neq_\alpha st((\lambda k.k\,\Omega)\,(\lambda x.y)).$$

  - Any strict strong strategy $st$ and any strict weak strategy $st_w$ (differing at least in $la$, with $ar_1 = True$):

    $$(st \circ st_w)(Z\ RecF\ Input\ d) \neq_\alpha st(Z\ RecF\ Input\ d).$$

# Absorption among uniform strategies

- We analyse the pairs of strategies in the order relation.
- Counterexamples:
  - Any strict $st_s$ and non-strict $st$ strategies (differing at least in $ar_1$):
    $$(st_s \circ st)((\lambda x.\lambda y.x)\,x\,\Omega) \neq_\alpha st_s((\lambda x.\lambda y.x)\,x\,\Omega).$$
  - Any strong non-head strategy $st$ and its weak or head (or weak-head) counterpart $st_{wh}$ (differing in $la$ or $ar_2$ or both):
    $$(st \circ st_{wh})((\lambda k.k\,\Omega)\,(\lambda x.y)) \neq_\alpha st((\lambda k.k\,\Omega)\,(\lambda x.y)).$$
  - Any strict strong strategy $st$ and any strict weak strategy $st_w$ (differing at least in $la$, with $ar_1 = True$):
    $$(st \circ st_w)(Z\ RecF\ Input\ d) \neq_\alpha st(Z\ RecF\ Input\ d).$$
  - The strategies cbv and 0 1 0 (weak strict strategies differing in $ar_2$):
    $$(cbv \circ 0\ 1\ 0)((\lambda x.\lambda y.x)\,x\,(x\,\Omega)) \neq_\alpha cbv((\lambda x.\lambda y.x)\,x\,(x\,\Omega))$$

# Absorption among uniform strategies

- We analyse the pairs of strategies in the order relation.
- Counterexamples:
  - Any strict $st_s$ and non-strict $st$ strategies (differing at least in $ar_1$):

$$(st_s \circ st)((\lambda x.\lambda y.x)\, x\, \Omega) \neq_\alpha st_s((\lambda x.\lambda y.x)\, x\, \Omega).$$

  - Any strong non-head strategy $st$ and its weak or head (or weak-head) counterpart $st_{wh}$ (differing in $la$ or $ar_2$ or both):

$$(st \circ st_{wh})((\lambda k.k\, \Omega)\, (\lambda x.y)) \neq_\alpha st((\lambda k.k\, \Omega)\, (\lambda x.y)).$$

  - Any strict strong strategy $st$ and any strict weak strategy $st_w$ (differing at least in $la$, with $ar_1 = True$):

$$(st \circ st_w)(Z\ RecF\ Input\ d) \neq_\alpha st(Z\ RecF\ Input\ d).$$

  - The strategies cbv and 0 1 0 (weak strict strategies differing in $ar_2$):

$$(cbv \circ 0\ 1\ 0)((\lambda x.\lambda y.x)\, x\, (x\, \Omega)) \neq_\alpha cbv((\lambda x.\lambda y.x)\, x\, (x\, \Omega))$$

- Proofs:

# Absorption among uniform strategies

- We analyse the pairs of strategies in the order relation.
- Counterexamples:
  - Any strict $st_s$ and non-strict $st$ strategies (differing at least in $ar_1$):

    $$(st_s \circ st)((\lambda x.\lambda y.x)\, x\, \Omega) \neq_\alpha st_s((\lambda x.\lambda y.x)\, x\, \Omega).$$

  - Any strong non-head strategy $st$ and its weak or head (or weak-head) counterpart $st_{wh}$ (differing in $la$ or $ar_2$ or both):

    $$(st \circ st_{wh})((\lambda k.k\, \Omega)\, (\lambda x.y)) \neq_\alpha st((\lambda k.k\, \Omega)\, (\lambda x.y)).$$

  - Any strict strong strategy $st$ and any strict weak strategy $st_w$ (differing at least in $la$, with $ar_1 = True$):

    $$(st \circ st_w)(Z\ RecF\ Input\ d) \neq_\alpha st(Z\ RecF\ Input\ d).$$

  - The strategies cbv and 0 1 0 (weak strict strategies differing in $ar_2$):

    $$(cbv \circ 0\ 1\ 0)((\lambda x.\lambda y.x)\, x\, (x\, \Omega)) \neq_\alpha cbv((\lambda x.\lambda y.x)\, x\, (x\, \Omega))$$

- Proofs:
  - 0 0 1 absorpts *cbn*. By induction on the structure of the derivations.

# Absorption among uniform strategies

- We still don't know if *he* absorpts *cbn*.

# Hybridisation: motivation

- Uniform strategies are not normalising (to NF).
- Standard reduction is neccesary for normalisation [Curry and Feys 1958]: Never reduce to the left of the residual of an already-reduced redex.
- A way to standardise: operators and operands in applications should be reduced to *values* (à la Plotkin).
- Hybridisation: produce new strategies that modify uniform strategies on this very point.

# Hybridisation template

Hybrid strategy from subsidiary $\mathcal{S}$ and base $\mathcal{B}$:

$$\frac{}{x \overset{sub}{\to} x}$$

$$\frac{B \xrightarrow[\mathcal{S}.la]{sub} B'}{\lambda x.B \overset{sub}{\to} \lambda x.B'}$$

$$\frac{M \overset{sub}{\to} \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_1]{sub} N' \quad [N'/x]B \overset{sub}{\to} S}{M\,N \overset{sub}{\to} S}$$

$$\frac{M \overset{sub}{\to} M' \not\equiv \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_2]{sub} N'}{M\,N \overset{sub}{\to} M'\,N'}$$

# Hybridisation template

Hybrid strategy from subsidiary $\mathcal{S}$ and base $\mathcal{B}$:

$$\frac{}{x \overset{sub}{\to} x}$$

$$\frac{B \xrightarrow[\mathcal{S}.la]{sub} B'}{\lambda x.B \overset{sub}{\to} \lambda x.B'}$$

$$\frac{M \overset{sub}{\to} \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_1]{sub} N' \qquad [N'/x]B \overset{sub}{\to} S}{M\,N \overset{sub}{\to} S}$$

$$\frac{M \overset{sub}{\to} M' \not\equiv \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_2]{sub} N'}{M\,N \overset{sub}{\to} M'\,N'}$$

# Hybridisation template

Hybrid strategy from subsidiary $\mathcal{S}$ and base $\mathcal{B}$:

$$\frac{}{x \overset{sub}{\to} x}$$

$$\frac{B \xrightarrow[\mathcal{S}.la]{sub} B'}{\lambda x.B \overset{sub}{\to} \lambda x.B'}$$

$$\frac{M \overset{sub}{\to} \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_1]{sub} N' \qquad [N'/x]B \overset{sub}{\to} S}{M\,N \overset{sub}{\to} S}$$

$$\frac{M \overset{sub}{\to} M' \not\equiv \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_2]{sub} N'}{M\,N \overset{sub}{\to} M'\,N'}$$

$$\frac{}{x \overset{hyb}{\to} x}$$

$$\frac{B \xrightarrow[\mathcal{B}.la]{hyb} B'}{\lambda x.B \overset{hyb}{\to} \lambda x.B'}$$

$$\frac{\boxed{M \overset{sub}{\to} \lambda x.B} \qquad N \xrightarrow[\mathcal{B}.ar_1]{sub} N' \qquad [N'/x]B \overset{hyb}{\to} S}{M\,N \overset{hyb}{\to} S}$$

$$\frac{\boxed{M \overset{sub}{\to} M' \not\equiv \lambda x.B} \qquad M' \overset{hyb}{\to} M'' \qquad N \xrightarrow[\mathcal{B}.ar_2]{hyb} N''}{M\,N \overset{hyb}{\to} M''\,N''}$$

# Hybridisation template

Hybrid strategy from subsidiary $\mathcal{S}$ and base $\mathcal{B}$:

$$\frac{}{x \overset{sub}{\to} x} \qquad\qquad \frac{}{x \overset{hyb}{\to} x}$$

$$\frac{B \xrightarrow[\mathcal{S}.la]{sub} B'}{\lambda x.B \overset{sub}{\to} \lambda x.B'} \qquad\qquad \frac{B \xrightarrow[\mathcal{B}.la]{hyb} B'}{\lambda x.B \overset{hyb}{\to} \lambda x.B'}$$

$$\frac{M \overset{sub}{\to} \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_1]{sub} N' \qquad [N'/x]B \overset{sub}{\to} S}{M\,N \overset{sub}{\to} S} \qquad\qquad \frac{\boxed{M \overset{sub}{\to} \lambda x.B} \qquad N \xrightarrow[\mathcal{B}.ar_1]{sub} N' \qquad [N'/x]B \overset{hyb}{\to} S}{M\,N \overset{hyb}{\to} S}$$

$$\frac{M \overset{sub}{\to} M' \not\equiv \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_2]{sub} N'}{M\,N \overset{sub}{\to} M'\,N'} \qquad\qquad \frac{\boxed{M \overset{sub}{\to} M' \not\equiv \lambda x.B} \qquad M' \overset{hyb}{\to} M'' \qquad N \xrightarrow[\mathcal{B}.ar_2]{hyb} N''}{M\,N \overset{hyb}{\to} M''\,N''}$$

# Hybridisation template

Hybrid strategy from subsidiary $\mathcal{S}$ and base $\mathcal{B}$:

$$\frac{}{x \overset{sub}{\to} x}$$

$$\frac{B \xrightarrow[\mathcal{S}.la]{sub} B'}{\lambda x.B \overset{sub}{\to} \lambda x.B'}$$

$$\frac{M \overset{sub}{\to} \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_1]{sub} N' \qquad [N'/x]B \overset{sub}{\to} S}{M\,N \overset{sub}{\to} S}$$

$$\frac{M \overset{sub}{\to} M' \not\equiv \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_2]{sub} N'}{M\,N \overset{sub}{\to} M'\,N'}$$

$$\frac{}{x \overset{hyb}{\to} x}$$

$$\frac{B \xrightarrow[\mathcal{B}.la]{hyb} B'}{\lambda x.B \overset{hyb}{\to} \lambda x.B'}$$

$$\frac{\boxed{M \overset{sub}{\to} \lambda x.B} \qquad \boxed{N \xrightarrow[\mathcal{B}.ar_1]{sub} N'} \qquad [N'/x]B \overset{hyb}{\to} S}{M\,N \overset{hyb}{\to} S}$$

$$\frac{\boxed{M \overset{sub}{\to} M' \not\equiv \lambda x.B} \qquad M' \overset{hyb}{\to} M'' \qquad N \xrightarrow[\mathcal{B}.ar_2]{hyb} N''}{M\,N \overset{hyb}{\to} M''\,N''}$$

*hyb* or *sub* for the operand?

# Hybridisation template

Hybrid strategy from subsidiary $\mathcal{S}$ and base $\mathcal{B}$:

$$\frac{}{x \overset{sub}{\to} x} \qquad\qquad \frac{}{x \overset{hyb}{\to} x}$$

$$\frac{B \xrightarrow[\mathcal{S}.la]{sub} B'}{\lambda x.B \overset{sub}{\to} \lambda x.B'} \qquad\qquad \frac{B \xrightarrow[\mathcal{B}.la]{hyb} B'}{\lambda x.B \overset{hyb}{\to} \lambda x.B'}$$

$$\frac{M \overset{sub}{\to} \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_1]{sub} N' \qquad [N'/x]B \overset{sub}{\to} S}{M\,N \overset{sub}{\to} S} \qquad\qquad \frac{\boxed{M \overset{sub}{\to} \lambda x.B} \qquad \boxed{N \xrightarrow[\mathcal{B}.ar_1]{sub} N'} \qquad [N'/x]B \overset{hyb}{\to} S}{M\,N \overset{hyb}{\to} S}$$

$$\frac{M \overset{sub}{\to} M' \not\equiv \lambda x.B \qquad N \xrightarrow[\mathcal{S}.ar_2]{sub} N'}{M\,N \overset{sub}{\to} M'\,N'} \qquad\qquad \frac{\boxed{M \overset{sub}{\to} M' \not\equiv \lambda x.B} \qquad M' \overset{hyb}{\to} M'' \qquad N \xrightarrow[\mathcal{B}.ar_2]{hyb} N''}{M\,N \overset{hyb}{\to} M''\,N''}$$

*hyb* or *sub* for the operand?

- Standardisation [Curry and Feys 1958] and absorption [Garcia et al. 2010] issues.

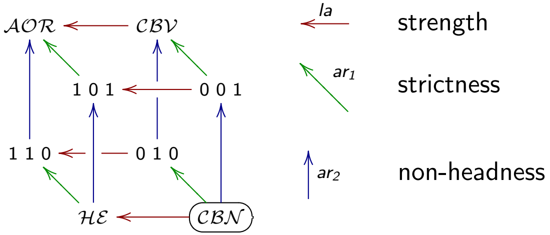
# Hybridisation and the Beta Cube

cbn:

$$\text{VAR}\ \frac{}{x \overset{cbn}{\to} x} \qquad \text{ABS}\ \frac{}{\lambda x.B \overset{cbn}{\to} \lambda x.B}$$

$$\text{RED}\ \frac{M \overset{cbn}{\to} M' \equiv \lambda x.B \qquad [N/x]B \overset{cbn}{\to} S}{M\,N \overset{cbn}{\to} S}$$

$$\text{APP}\ \frac{M \overset{cbn}{\to} M' \not\equiv \lambda x.B}{M\,N \overset{cbn}{\to} M'\ N}$$



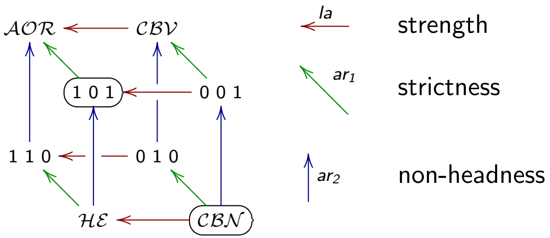
# Hybridisation and the Beta Cube

nor $\equiv$ hybridise(cbn,1 0 1):

$$\textrm{VAR}\ \frac{}{x \overset{nor}{\to} x} \qquad\qquad \textrm{ABS}\ \frac{B \overset{nor}{\to} B'}{\lambda x.B \overset{nor}{\to} \lambda x.B'}$$

$$\textrm{RED}\ \frac{\boxed{M \overset{cbn}{\to} M' \equiv \lambda x.B} \qquad [N/x]B \overset{nor}{\to} S}{M\,N \overset{nor}{\to} S}$$

$$\textrm{APP}\ \frac{\boxed{M \overset{cbn}{\to} M' \not\equiv \lambda x.B} \qquad M' \overset{nor}{\to} M'' \qquad N \overset{nor}{\to} N'}{M\,N \overset{nor}{\to} M''\,N'}$$



# Absorption theorem

## Theorem

*Let $\mathcal{S}$ and $\mathcal{B}$ be respectively a subsidiary and a base strategy considered as points in the cube which satisfy $\mathcal{S} \sqsubseteq \mathcal{B}$ and $\mathcal{S}.ar_1 = \mathcal{B}.ar_1$.*

*Let $\stackrel{sub}{\rightarrow}$ and $\stackrel{hyb}{\rightarrow}$ be the resulting instantiated strategies. Then*

$$(hyb \circ sub)(t) = hyb(t)$$

*for any term $t$.*

## Proof.

By induction on the structure of the derivations. □

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.
  - The operands of applications are reduced to normal forms before substitution.

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.
  - The operands of applications are reduced to normal forms before substitution.
- Standard reduction: never reduce to the left of the residual of an already-reduced redex, but...

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.
  - The operands of applications are reduced to normal forms before substitution.
- Standard reduction: never reduce to the left of the residual of an already-reduced redex, but...
  ...operands are reduced by the hybrid:

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.
  - The operands of applications are reduced to normal forms before substitution.
- Standard reduction: never reduce to the left of the residual of an already-reduced redex, but...
  ...operands are reduced by the hybrid:
  - If the operand is not a value then we still don't have a redex.

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.
  - The operands of applications are reduced to normal forms before substitution.
- Standard reduction: never reduce to the left of the residual of an already-reduced redex, but...
  ...operands are reduced by the hybrid:
  - If the operand is not a value then we still don't have a redex.
  - We reduce the operand to a value. Now the application is a redex.

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.
  - The operands of applications are reduced to normal forms before substitution.
- Standard reduction: never reduce to the left of the residual of an already-reduced redex, but...

  ...operands are reduced by the hybrid:
  - If the operand is not a value then we still don't have a redex.
  - We reduce the operand to a value. Now the application is a redex.
  - We keep reducing the operand up to a normal form.

    The value in the operand position is reduced before reducing the redex itself!

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.
  - The operands of applications are reduced to normal forms before substitution.
- Standard reduction: never reduce to the left of the residual of an already-reduced redex, but...
  ...operands are reduced by the hybrid:
  - If the operand is not a value then we still don't have a redex.
  - We reduce the operand to a value. Now the application is a redex.
  - We keep reducing the operand up to a normal form.
    The value in the operand position is reduced before reducing the redex itself!
- Consequently:

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.
  - The operands of applications are reduced to normal forms before substitution.
- Standard reduction: never reduce to the left of the residual of an already-reduced redex, but...
  ...operands are reduced by the hybrid:
  - If the operand is not a value then we still don't have a redex.
  - We reduce the operand to a value. Now the application is a redex.
  - We keep reducing the operand up to a normal form.
    The value in the operand position is reduced before reducing the redex itself!
- Consequently:
  - *ha* does not absorb *cbv* [Garcia et al. 2010].

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.
  - The operands of applications are reduced to normal forms before substitution.
- Standard reduction: never reduce to the left of the residual of an already-reduced redex, but...
  ...operands are reduced by the hybrid:
  - If the operand is not a value then we still don't have a redex.
  - We reduce the operand to a value. Now the application is a redex.
  - We keep reducing the operand up to a normal form.
    The value in the operand position is reduced before reducing the redex itself!
- Consequently:
  - *ha* does not absorb *cbv* [Garcia et al. 2010].
  - *ha* is not a standard $\beta_V$-reduction.

# Sestoft's hibrydisation

- *hybridiseSestoft* uses *hyb* for the selection of $ar_1$.
- Consider $ha = hybridiseSestoft(cbv, aor)$.
  - *ha* reduces to normal form.
  - The operands of applications are reduced to normal forms before substitution.
- Standard reduction: never reduce to the left of the residual of an already-reduced redex, but...
  ...operands are reduced by the hybrid:
  - If the operand is not a value then we still don't have a redex.
  - We reduce the operand to a value. Now the application is a redex.
  - We keep reducing the operand up to a normal form.
    The value in the operand position is reduced before reducing the redex itself!
- Consequently:
  - *ha* does not absorb *cbv* [Garcia et al. 2010].
  - *ha* is not a standard $\beta_V$-reduction.
  - *ha* is not normalising in $\lambda_V$.

# Implementations in OCaML and Haskell

- Rule Template:
  - Generic reducer: higher-order.
  - (Haskell) Monadic reducer: strict monads for strict semantics.
  - Particular strategies are fixed points.

# Implementations in OCaML and Haskell

- Rule Template:
  - Generic reducer: higher-order.
  - (Haskell) Monadic reducer: strict monads for strict semantics.
  - Particular strategies are fixed points.
- Beta Cube:
  - Boolean triple.
  - `cube2red` delivers a reducer from a point in the cube.

# Implementations in OCaML and Haskell

- Rule Template:
  - Generic reducer: higher-order.
  - (Haskell) Monadic reducer: strict monads for strict semantics.
  - Particular strategies are fixed points.
- Beta Cube:
  - Boolean triple.
  - `cube2red` delivers a reducer from a point in the cube.
- Hybridisation:

# Implementations in OCaML and Haskell

- Rule Template:
  - Generic reducer: higher-order.
  - (Haskell) Monadic reducer: strict monads for strict semantics.
  - Particular strategies are fixed points.
- Beta Cube:
  - Boolean triple.
  - `cube2red` delivers a reducer from a point in the cube.
- Hybridisation:
  - `hybridise` delivers a hybrid reducer from subsidiary and base from the cube.

# Contributions

- Rule template generalises pure lambda calculus reduction strategies. (introducing $op_1$ and $op_2$ to accommodate hybrids)
- Beta Cube + Hybridise systematise the strategy space.
- Studied absorption among vertices in the lattice.
- Hybridisation operator:
  1. Operands in applications reduced by hybrid: may not deliver strict normalising strategies.
  2. Operands in applications reduced by subsidiary: may deliver strict normalising strategies.
- Absorption among hybrids and their subsidiaries (Absorption theorem).
- Implementation in OCaML and Haskell.

# Future work

- Head strategies: using hnf instead of wnf as the notion of value.
  - Head thunks (reduction stops at the right of a free variable)

# Future work

- Head strategies: using hnf instead of wnf as the notion of value.
  - Head thunks (reduction stops at the right of a free variable)
- Strategies and CPS transformation.

# Future work

- Head strategies: using hnf instead of wnf as the notion of value.
  - Head thunks (reduction stops at the right of a free variable)
- Strategies and CPS transformation.
- Implementing efficient $\beta$-testers for typing rules in dependent types systems.

# Future work

- Head strategies: using hnf instead of wnf as the notion of value.
  - Head thunks (reduction stops at the right of a free variable)
- Strategies and CPS transformation.
- Implementing efficient $\beta$-testers for typing rules in dependent types systems.
- Strategies to interpret universes in structural generic programming for dependent types.

# Backup slides

# Generic reducer in OCaML

```
let genred la op1 ar1 su op2 ar2 = function
  | Var _ as v   -> v
  | Lam (x, b)   -> Lam (x, la b)
  | App (m, n)   -> let m’ = op1 m in match m’ with
                      | Lam (x, b) -> su (subst (ar1 n) x b)
                      | _          -> App (op2 m’ , ar2 n)
```

# Strategies are fixed point

```
(********************** la  op1 ar1 su  op2 ar2 *)
let rec cbn x = (genred id  cbn id  cbn id  id ) x
let rec cbv x = (genred id  cbv cbv cbv id  cbv) x
let rec nor x = (genred nor cbn id  nor nor nor) x
let rec aor x = (genred aor aor aor aor id  aor) x
...
```

# Beta Cube implementation

```
let sel p red = if p then red else id

let cube2red = function (la, ar1, ar2) ->
  let rec red x
          = (genred
               (sel la red) red (sel ar1 red) red red (sel ar2 red)) x
  in red
```

# Hybridisation operator

```
let hybridise s = function (la, ar1, ar2) ->
  let sub = cube2red s in
  let rec hyb x
          = (genred
               (sel la hyb) sub (sel ar1 sub) hyb hyb (sel ar2 hyb)) x
  in hyb
```

# Sestoft's hibrydisation

```
let hybridiseSestoft s = function (la, ar1, ar2) ->
  let sub = cube2red s in
  let rec hyb x
        = (genred
             (sel la hyb) sub (sel ar1 hyb) hyb hyb (sel ar2 hyb)) x
  in hyb
```

# Generic reducer in Haskell

```
data Term = Var String | Lam String Term | App Term Term

type Red = Monad m => Term -> m Term

genred :: Red -> Red -> Red -> Red -> Red -> Red -> Red
genred la op1 ar1 su op2 ar2 t =
  case t of
   v@(Var _) -> return v
   (Lam x b) -> do b' <- la b
                   return (Lam x b')
   (App m n) -> do m' <- op1 m
                   case m' of
                     (Lam x b) -> do n' <- ar1 n
                                     su (subst b n' x)
                     _         -> do m'' <- op2 m'
                                     n'' <- ar2 n
                                     return (App m'' n'')
```

# Strategies are fixed points

```
               la      op1  ar1      su   op2    ar2

cbn  = genred  return  cbn  return   cbn  return  return
cbv  = genred  return  cbv  cbv      cbv  return  cbv
aor  = genred  aor     aor  aor      aor  return  aor
nor  = genred  nor     cbn  return   nor  nor     return
...
```

# Beta Cube implementation

```
data BCube = BC Bool Bool Bool

cube2red :: Monad m => BCube -> Red m
cube2red (BC la ar1 ar2) =
    let red = genred
                (sel la red) red (sel ar1 red) red red (sel ar2 red)
    in red
    where sel par red = if par then red else return
```

# Hibrydisation operator

```
hybridise :: (BetaCube, BetaCube) -> Red
hybridise (sub, (BC lab ar1b ar2b)) =
  let s = cube2red sub
      h = genred (sel lab h) s (sel ar1b s) h h (sel ar2b h)
  in h
```

# Sestoft's hibrydisation operator

```
hybridiseSestoft :: (BetaCube, BetaCube) -> Red
hybridiseSestoft (sub, (BC lab ar1b ar2b)) =
  let s = cube2red sub
      h = genred (sel lab h) s (sel ar1b h) h h (sel ar2b h)
  in h
```