A puzzle to ponder on α-conversion

- give an upperbound on the #α-renamings needed to β-reduce $((\underline{2}\,\underline{8})\,(\underline{4}\,\underline{9}))\,(\underline{5}\,\underline{7})\,(\underline{4}\,\underline{2})$ to normal form?
- note 1: application of Church-numerals is exponentiation; $\underline{k}\,\underline{n} \twoheadrightarrow_\beta \underline{n^k}$
- note 2: whether α-conversion is needed in a β-reduction is undecidable

On naïvely implementing the $\lambda\beta$-calculus

Vincent van Oostrom

λ-calculus naïvely

$$\underline{2} := \lambda sz.s\,(s\,z)$$

Church numeral 2

running example, reduces to four

$$(\lambda sz.s\,(s\,z))\,\lambda sz.s\,(s\,z)$$

λ-calculus naïvely

$\underline{2} := \lambda sz.s\,(s\,z)$

$\underline{2}\,\underline{2}$

λ-calculus naïvely

$$\underline{2} := \lambda sz.s\,(s\,z)$$

$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

β-reduction with naïve substitution

(not in λx; indiscriminantly in λy)

$$\underline{2}\,\underline{2}$$

λ-calculus naïvely

$$\underline{2} := \lambda sz.s\,(s\,z)$$

$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

$$\underline{2}\,\underline{2}$$

combinator system

$\underline{2} := \lambda sz.s\,(s\,z)$

$(\lambda x.M)\,N \to_\beta M[x{:=}N]$

lifting $\lambda z.s\,(s\,z)$

$\underline{2}\,\underline{2}$

combinator system

$\underline{2} := \lambda sz.s\,(s\,z)$

$(\lambda x.M)\,N \to_\beta M[x{:=}N]$

lifting $\lambda z.s\,(s\,z)$

skeleton $\lambda z.\square\,(\square\,z) \mapsto$ f-symbol Z

maximal free subexpressions s, s

$\underline{2}\,\underline{2}$

combinator system

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$Z[x,y]$ represents $\lambda z.x\,(y\,z)$

$$\underline{2}\,\underline{2}$$

combinator system

$\underline{2} := \lambda sz.s\,(s\,z)$

$(\lambda x.M)\,N \to_\beta M[x{:=}N]$

$Z[x,y]\,z \to_\kappa x\,(y\,z)$

$\underline{2}\,\underline{2}$

combinator system

$\underline{2} := \lambda sz.s\,(s\,z)$

$(\lambda x.M)\,N \to_\beta M[x{:=}N]$

$\underline{2}\,\underline{2}$

$Z[x,y]\,z \to_\kappa x\,(y\,z)$

lifting $\lambda s.Z[s,s]$

its own skeleton $\mapsto$ f-symbol S

no maximal free subexpressions

combinator system

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
S represents $\underline{2} := \lambda sz.s\,(s\,z)$

$$\underline{2}\;\underline{2}$$

combinator system

$\underline{2} := \lambda sz.s\,(s\,z)$

$(\lambda x.M)\,N \to_\beta M[x{:=}N]$

$Z[x,y]\,z \to_\kappa x\,(y\,z)$

$S\,z \to_\kappa Z[z,z]$

running example, reduces to four

$\underline{2}\,\underline{2}$

$S\,S$

combinator system

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

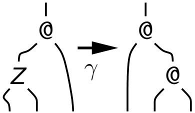
$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$

$$\underline{2}\,\underline{2}$$

$$S\,S$$

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$



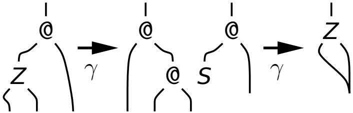
$$\underline{2}\,\underline{2}$$

$$S\,S$$

TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$



$\underline{2}\,\underline{2}$

$S\,S$

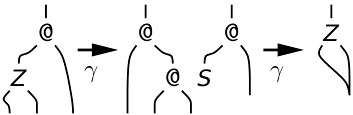
duplication by sharing in rhs

$\underline{2} := \lambda sz.s\,(s\,z)$

$(\lambda x.M)\,N \to_\beta M[x{:=}N]$

$Z[x, y]\,z \to_\kappa x\,(y\,z)$

$S\,z \to_\kappa Z[z, z]$



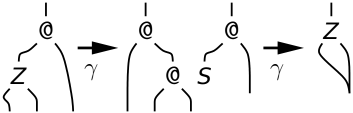
$\underline{2}\,\underline{2}$

$S\,S$

TGRS

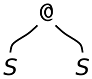
$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_{\beta} M[x{:=}N]$$

$$Z[x,y]\,z \to_{\kappa} x\,(y\,z)$$
$$S\,z \to_{\kappa} Z[z,z]$$



$$\underline{2}\,\underline{2}$$

$$S\,S$$

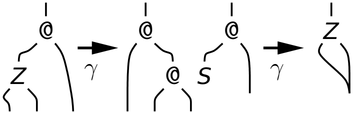


running example, reduces to four

λ-calculus $\iff$ combinator system $\iff$ TGRS

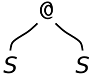
$\underline{2} := \lambda sz.s\,(s\,z)$
$(\lambda x.M)\,N \to_\beta M[x{:=}N]$

$Z[x,y]\,z \to_\kappa x\,(y\,z)$
$S\,z \to_\kappa Z[z,z]$



$\underline{2}\,\underline{2}$

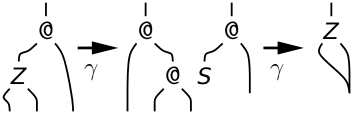
$S\,S$



λ-calculus $\iff$ combinator system $\iff$ TGRS

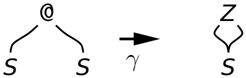
$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$



$$\underline{2}\,\underline{2} \xrightarrow{\beta} \lambda z.\underline{2}\,(\underline{2}\,z)$$

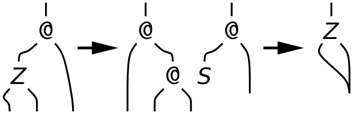
$$S\,S \xrightarrow{\kappa} Z[S,S]$$



λ-calculus $\iff$ combinator system $\iff$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$



$$\lambda z.\underline{2}\,(\underline{2}\,z)$$

$$Z[S,S]$$

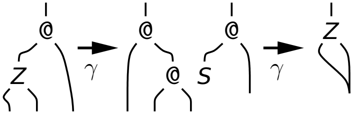
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

$\underline{2} := \lambda sz.s\,(s\,z)$

$(\lambda x.M)\,N \to_\beta M[x{:=}N]$

$Z[x, y]\,z \to_\kappa x\,(y\,z)$

$S\,z \to_\kappa Z[z, z]$



$\lambda z.\underline{2}\,(\underline{2}\,z)$

$Z[S, S]$

weak head normal form (under λ)

normal form (Z is stuck)

normal form (Z is stuck)

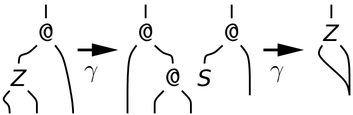
λ-calculus $\iff$ combinator system $\iff$ TGRS

$\underline{2} := \lambda sz.s\,(s\,z)$

$(\lambda x.M)\,N \to_{\beta} M[x{:=}N]$

$Z[x,y]\,z \to_{\kappa} x\,(y\,z)$

$S\,z \to_{\kappa} Z[z,z]$



$\lambda z.\underline{2}\,(\underline{2}\,z)$

$Z[S,S]$

root-introduce fresh constant

root-introduce fresh constant

root-introduce fresh constant

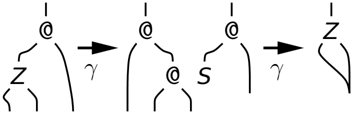
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$
(z' fresh; think of as constant)
$$\lambda z.\underline{2}\,(\underline{2}\,z)$$

factor α through β (at root)

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$
$$Z[S,S]$$

unstuck combinator (at root)



Z,S-rules as expected (at root)

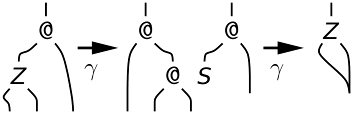
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

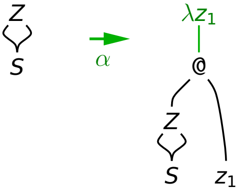
$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda z.\underline{2}\,(\underline{2}\,z)$$
$$\Big\downarrow \alpha$$
$$\lambda z_1.(\lambda z.\underline{2}\,(\underline{2}\,z))\,z_1$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$

$$Z[S,S]$$
$$\Big\downarrow \alpha$$
$$\lambda z_1.Z[S,S]\,z_1$$



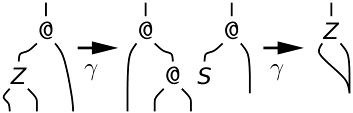


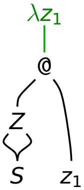
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda z_1.(\lambda z.\underline{2}\,(\underline{2}\,z))\,z_1$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$
$$\lambda z_1.Z[S,S]\,z_1$$





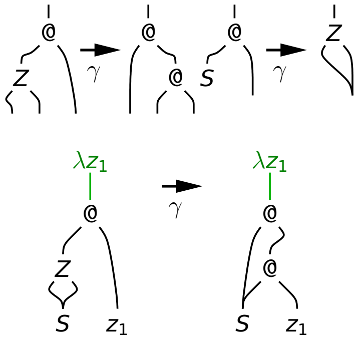
λ-calculus $\iff$ combinator system $\iff$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda z_1.(\lambda z.\underline{2}\,(\underline{2}\,z))\,z_1$$
$$\Big\downarrow \beta$$
$$\lambda z_1.\underline{2}\,(\underline{2}\,z_1)$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$

$$\lambda z_1.Z[S,S]\,z_1$$
$$\Big\downarrow \kappa$$
$$\lambda z_1.S\,(S\,z_1)$$



λ-calculus $\iff$ combinator system $\iff$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$

$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda z_1.\underline{2}\,(\underline{2}\,z_1)$$

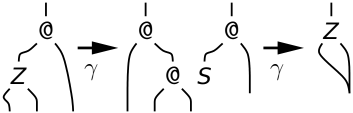
$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$

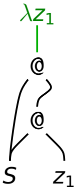
$$S\,z \to_\kappa Z[z,z]$$

$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$

$$S \to_\alpha \lambda z'.S\,z'$$

$$\lambda z_1.S\,(S\,z_1)$$



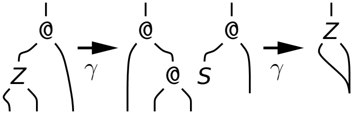


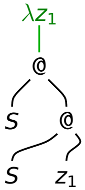
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda z_1.\underline{2}\,(\underline{2}\,z_1)$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$
$$\lambda z_1.S\,(S\,z_1)$$





unshare constructor of redex

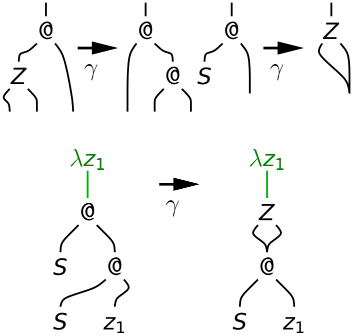
λ-calculus $\iff$ combinator system $\iff$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda z_1.\underline{2}\,(\underline{2}\,z_1)$$
$$\Big\downarrow \beta$$
$$\lambda z_1.\lambda z.\underline{2}\,z_1\,(\underline{2}\,z_1\,z)$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$

$$\lambda z_1.S\,(S\,z_1)$$
$$\Big\downarrow \kappa$$
$$\lambda z_1.Z[S\,z_1, S\,z_1]$$



λ-calculus $\iff$ combinator system $\iff$ TGRS

$\underline{2} := \lambda sz.s\,(s\,z)$

$(\lambda x.M)\,N \to_\beta M[x{:=}N]$

$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$

$\lambda z_1.\lambda z.\underline{2}\,z_1\,(\underline{2}\,z_1\,z)$

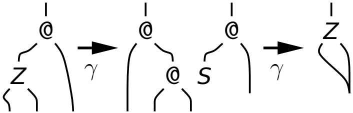
$Z[x,y]\,z \to_\kappa x\,(y\,z)$

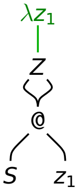
$S\,z \to_\kappa Z[z,z]$

$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$

$S \to_\alpha \lambda z'.S\,z'$

$\lambda z_1.Z[S\,z_1, S\,z_1]$





λ-calculus $\iff$ combinator system $\iff$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$

$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$

$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda z_1.\lambda z.\underline{2}\,z_1\,(\underline{2}\,z_1\,z)$$

$$\downarrow \alpha$$

$$\lambda z_1 z_2.(\lambda z.\underline{2}\,z_1\,(\underline{2}\,z_1\,z))\,z_2$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$

$$S\,z \to_\kappa Z[z,z]$$

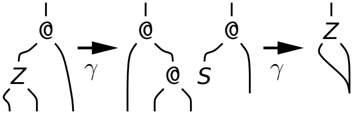
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$

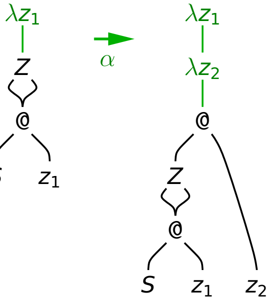
$$S \to_\alpha \lambda z'.S\,z'$$

$$\lambda z_1.Z[S\,z_1, S\,z_1]$$

$$\downarrow \alpha$$

$$\lambda z_1 z_2.Z[S\,z_1, S\,z_1]\,z_2$$



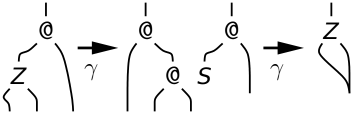


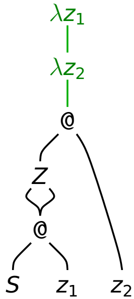
λ-calculus $\iff$ combinator system $\iff$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda z_1 z_2.(\lambda z.\underline{2}\,z_1\,(\underline{2}\,z_1\,z))\,z_2$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$
$$\lambda z_1 z_2.Z[S\,z_1, S\,z_1]\,z_2$$





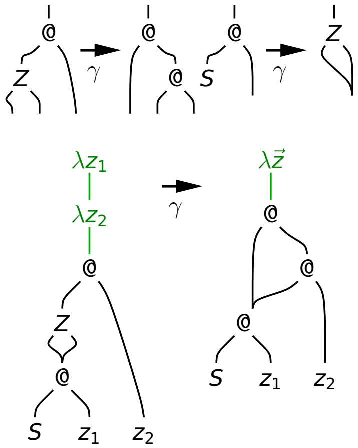
λ-calculus $\iff$ combinator system $\iff$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda z_1 z_2.(\lambda z.\underline{2}\,z_1\,(\underline{2}\,z_1\,z))\,z_2$$
$$\Big\downarrow \beta$$
$$\lambda \vec{z}.\underline{2}\,z_1\,(\underline{2}\,z_1\,z_2)$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$

$$\lambda z_1 z_2.Z[S\,z_1, S\,z_1]\,z_2$$
$$\Big\downarrow \kappa$$
$$\lambda \vec{z}.S\,z_1\,(S\,z_1\,z_2)$$

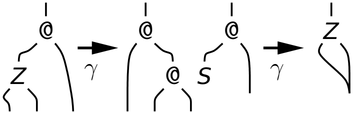


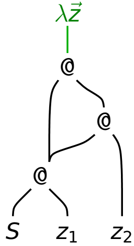
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda\vec{z}.\underline{2}\,z_1\,(\underline{2}\,z_1\,z_2)$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$
$$\lambda\vec{z}.S\,z_1\,(S\,z_1\,z_2)$$





λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

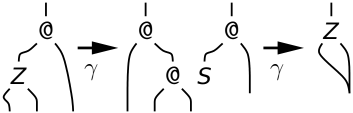
$$\lambda\vec{z}.\underline{2}\,z_1\,(\underline{2}\,z_1\,z_2)$$
$$\Big\downarrow \mathrm{f}\beta$$
$$\lambda\vec{z}.(\lambda z.z_1\,(z_1\,z))\,((\lambda z.z_1\,(z_1\,z))\,z_2)$$

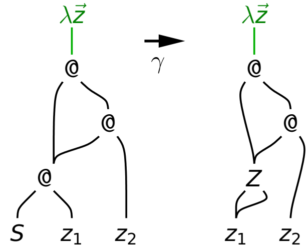
parallel β (weak family)

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$

$$\lambda\vec{z}.S\,z_1\,(S\,z_1\,z_2)$$
$$\Big\downarrow \mathrm{f}\kappa$$
$$\lambda\vec{z}.Z[z_1,z_1]\,(Z[z_1,z_1]\,z_2)$$

parallel κ (family)





contract shared S-redex

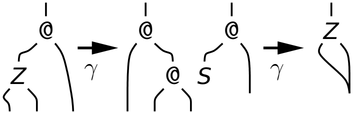
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

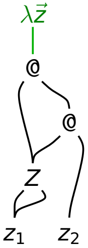
$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \rightarrow_\beta M[x{:=}N]$$
$$\lambda z.M \rightarrow_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda\vec{z}.(\lambda z.z_1\,(z_1\,z))\,((\lambda z.z_1\,(z_1\,z))\,z_2)$$

$$Z[x,y]\,z \rightarrow_\kappa x\,(y\,z)$$
$$S\,z \rightarrow_\kappa Z[z,z]$$
$$Z[x,y] \rightarrow_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \rightarrow_\alpha \lambda z'.S\,z'$$

$$\lambda\vec{z}.Z[z_1,z_1]\,(Z[z_1,z_1]\,z_2)$$





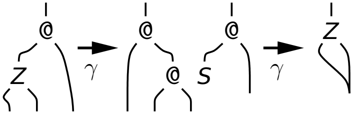
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

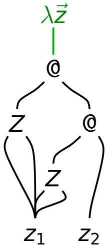
$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda\vec{z}.(\lambda z.z_1\,(z_1\,z))\,((\lambda z.z_1\,(z_1\,z))\,z_2)$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$

$$\lambda\vec{z}.Z[z_1,z_1]\,(Z[z_1,z_1]\,z_2)$$





unshare constructor of redex

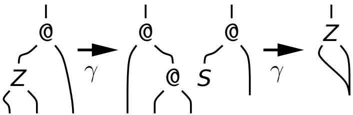
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

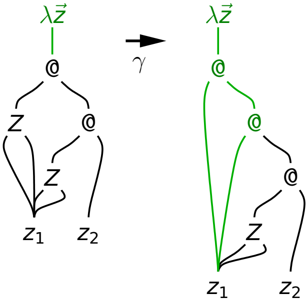
$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \rightarrow_\beta M[x{:=}N]$$
$$\lambda z.M \rightarrow_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda\vec{z}.(\lambda z.z_1\,(z_1\,z))\,((\lambda z.z_1\,(z_1\,z))\,z_2)$$
$$\big\downarrow \beta$$
$$\lambda\vec{z}.z_1\,(z_1\,((\lambda z.z_1\,(z_1\,z))\,z_2))$$

$$Z[x,y]\,z \rightarrow_\kappa x\,(y\,z)$$
$$S\,z \rightarrow_\kappa Z[z,z]$$
$$Z[x,y] \rightarrow_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \rightarrow_\alpha \lambda z'.S\,z'$$

$$\lambda\vec{z}.Z[z_1,z_1]\,(Z[z_1,z_1]\,z_2)$$
$$\big\downarrow \kappa$$
$$\lambda\vec{z}.z_1\,(z_1\,(Z[z_1,z_1]\,z_2))$$



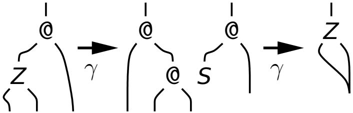


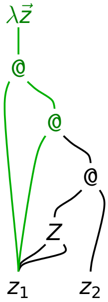
λ-calculus $\iff$ combinator system $\iff$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda\vec{z}.z_1\,(z_1\,((\lambda z.z_1\,(z_1\,z))\,z_2))$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$
$$\lambda\vec{z}.z_1\,(z_1\,(Z[z_1,z_1]\,z_2))$$





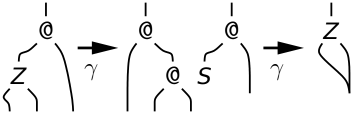
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

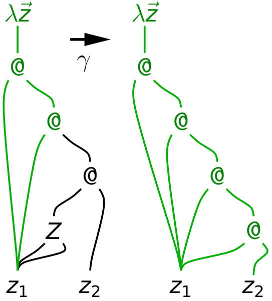
$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda\vec{z}.z_1\,(z_1\,((\lambda z.z_1\,(z_1\,z))\,z_2))$$
$$\Big\downarrow \beta$$
$$\lambda\vec{z}.z_1\,(z_1\,(z_1\,(z_1\,z_2)))$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$

$$\lambda\vec{z}.z_1\,(z_1\,(Z[z_1,z_1]\,z_2))$$
$$\Big\downarrow \kappa$$
$$\lambda\vec{z}.z_1\,(z_1\,(z_1\,(z_1\,z_2)))$$





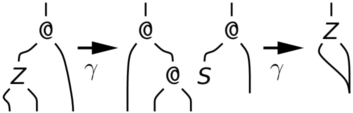
λ-calculus $\iff$ combinator system $\iff$ TGRS

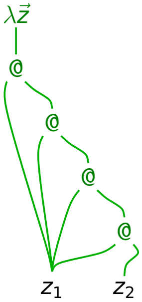
$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

$$\lambda\vec{z}.z_1\,(z_1\,(z_1\,(z_1\,z_2)))$$

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$

$$\lambda\vec{z}.z_1\,(z_1\,(z_1\,(z_1\,z_2)))$$





λ-calculus $\iff$ combinator system $\iff$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

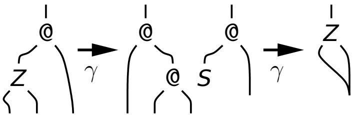
$$\lambda\vec{z}.z_1\,(z_1\,(z_1\,(z_1\,z_2)))$$

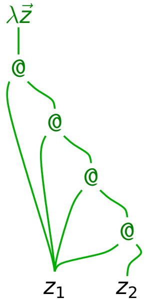
normal form

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$

$$\lambda\vec{z}.z_1\,(z_1\,(z_1\,(z_1\,z_2)))$$

normal form





normal form

λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS

$$\underline{2} := \lambda sz.s\,(s\,z)$$
$$(\lambda x.M)\,N \to_\beta M[x{:=}N]$$
$$\lambda z.M \to_\alpha \lambda z'.(\lambda z.M)\,z'$$

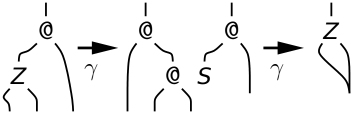
$\underline{4}$

normal form

$$Z[x,y]\,z \to_\kappa x\,(y\,z)$$
$$S\,z \to_\kappa Z[z,z]$$
$$Z[x,y] \to_\alpha \lambda z'.Z[x,y]\,z'$$
$$S \to_\alpha \lambda z'.S\,z'$$

$\underline{4}$

normal form



$\dagger_4$

normal form

Concrete results

- a graph implementation of $\ell o \beta$ where every step is bounded (by sizes of liftings of λ-subterms)

Concrete results

- a graph implementation of ℓoβ where every step is bounded
- leftmost–outermost (ℓo) γ-step corresponds to parallel ℓoβ-step (#ℓoβ's is upperbound on space,time; ℓoβ cost-model)

Concrete results

- a graph implementation of $\ell o\beta$ where every step is bounded
- leftmost–outermost (ℓo) γ-step corresponds to parallel $\ell o\beta$-step
- only naïve substitution (no renaming) in $\ell o\beta$; explicit α-steps (fresh) (1 per λ in output term; derived variable name, no de Bruin-indices)

Concrete results

- a graph implementation of $\ell o\beta$ where every step is bounded
- leftmost–outermost (ℓo) γ-step corresponds to parallel $\ell o\beta$-step
- only naïve substitution in $\ell o\beta$; explicit α-steps
- based on classical TGR techniques
  (previous millennium)

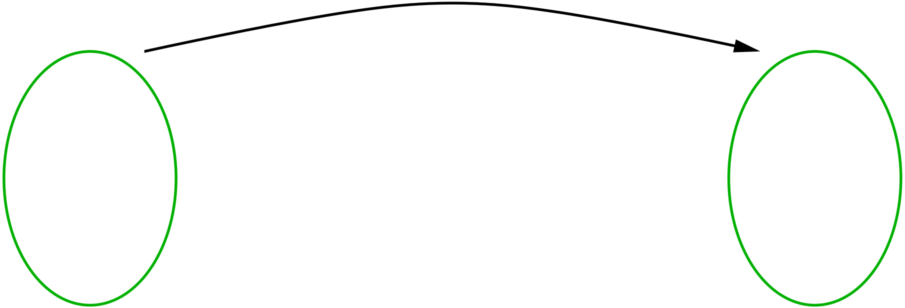
Concrete results

- a graph implementation of $\ell o\beta$ where every step is bounded
- leftmost–outermost (ℓo) γ-step corresponds to parallel $\ell o\beta$-step
- only naïve substitution in $\ell o\beta$; explicit α-steps
- based on classical TGR techniques

caution: all this is strategy ($\ell o\beta$) specific; combinator translations will hide β-redexes in general (the naïve abstraction algorithm maps any λ-abstraction to a CL-normal form). In the case of $\ell o\beta$ explicit α at the root suffices to make such come unstuck. In general, that's not sufficient.

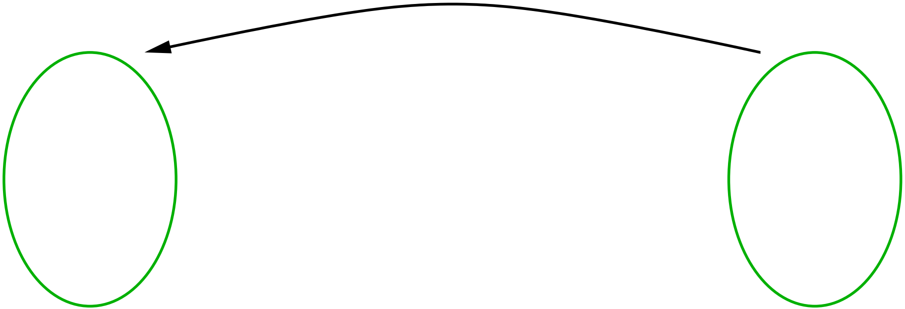
λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS, weak

a sharing graph implementation of β (Wadsworth 71)

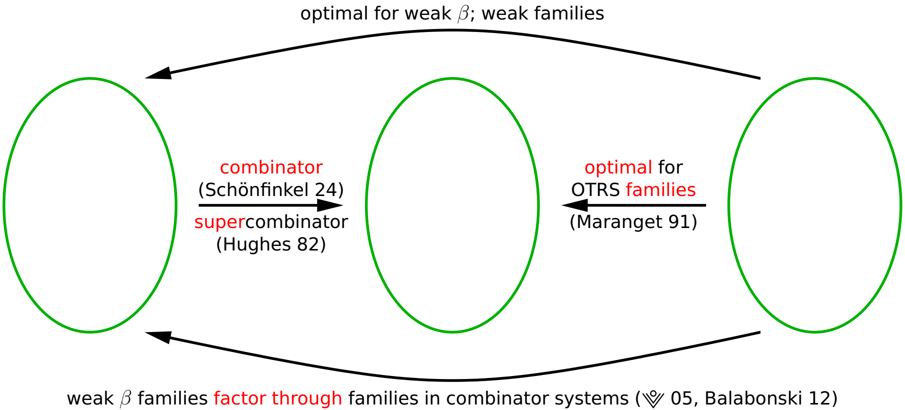


λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS, weak

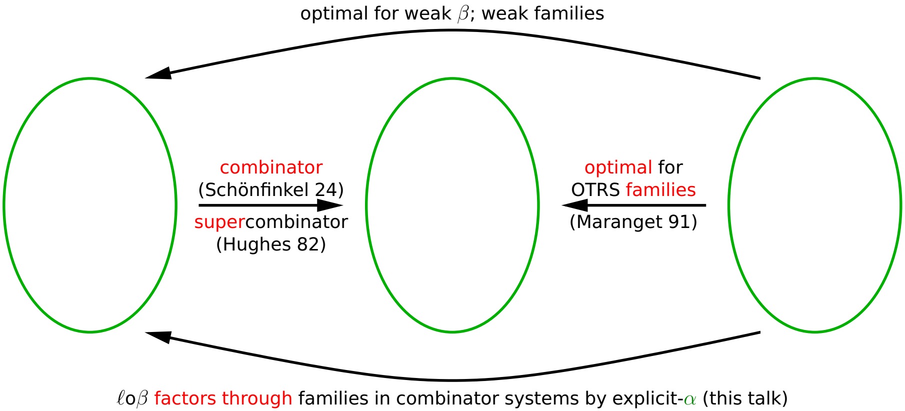
optimal (Blanc, Lévy, Maranget 05) for weak β (Çağman, Hindley 98); weak families



λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS, weak



λ-calculus $\Longleftrightarrow$ combinator system $\Longleftrightarrow$ TGRS, $\ell o \beta$



Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf (only naïve substitution; here $\ell o \beta$ but e.g. also call-by-value)

Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of $\ell o\beta$
  (optimal (Maranget) for supercombinators (weak β), not for β (Lévy)!)

Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of ℓoβ
- cost-model for ℓoβ (Dal Lago,Accatoli)
  (space,time linear in #TGRS-steps and bounded by #ℓoβ-steps)

Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of $\ell o \beta$
- cost-model for $\ell o \beta$ (Dal Lago,Accatoli)

implemented (rapid Haskell prototype) $\ell o \beta$ to CL; from CL to TGRS future work

Meta-Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of $\ell o \beta$
- cost-model for $\ell o \beta$ (Dal Lago,Accatoli)

implemented $\ell o \beta$ to CL; from CL to TGRS future work

Meta-Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of ℓoβ
- cost-model for ℓoβ (Dal Lago,Accatoli)

implemented ℓoβ to CL; from CL to TGRS future work

- rewriting useful both for simple description and efficient implementation (no intermediate abstract machines (Krivine))

Meta-Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of ℓoβ
- cost-model for ℓoβ (Dal Lago,Accatoli)

implemented ℓoβ to CL; from CL to TGRS future work

- rewriting useful both for simple description and efficient implementation
- higher-order rewriting useful to bridge λ-calculus $\iff$ supercombinators (rid of binders, no intermediate let-calculus; combinator system novel?)

Meta-Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of ℓoβ
- cost-model for ℓoβ (Dal Lago,Accatoli)

implemented ℓoβ to CL; from CL to TGRS future work

- rewriting useful both for simple description and efficient implementation
- higher-order rewriting useful to bridge λ-calculus $\iff$ supercombinators
- substitution calculus (⩔, van Raamsdonk) useful to modularise TGR
  (ж makes matching and substitution explicit; see paper)

Meta-Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of ℓoβ
- cost-model for ℓoβ (Dal Lago,Accatoli)

implemented ℓoβ to CL; from CL to TGRS future work

- rewriting useful both for simple description and efficient implementation
- higher-order rewriting useful to bridge λ-calculus $\iff$ supercombinators
- substitution calculus (, van Raamsdonk) useful to modularise TGR
- classical techniques (Schönfinkel, Wadsworth, Hughes,...) for complexity

Meta-Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of ℓoβ
- cost-model for ℓoβ (Dal Lago,Accatoli)

implemented ℓoβ to CL; from CL to TGRS future work

- rewriting useful both for simple description and efficient implementation
- higher-order rewriting useful to bridge λ-calculus $\iff$ supercombinators
- substitution calculus (, van Raamsdonk) useful to modularise TGR
- classical techniques (Schönfinkel, Wadsworth, Hughes,...) for complexity
- Gödel not convinced by $\lambda\beta$ / TRS; me neither because no unit-time steps (abstract from replication; cf. Java abstracting from garbage collection)

Meta-Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of $\ell o\beta$
- cost-model for $\ell o\beta$ (Dal Lago,Accatoli)

implemented $\ell o\beta$ to CL; from CL to TGRS future work

- rewriting useful both for simple description and efficient implementation
- higher-order rewriting useful to bridge λ-calculus $\iff$ supercombinators
- substitution calculus (, van Raamsdonk) useful to modularise TGR
- classical techniques (Schönfinkel, Wadsworth, Hughes,$\ldots$) for complexity
- Gödel not convinced by $\lambda\beta$ / TRS; me neither because no unit-time steps

current work: amortised complexity via term rewriting (derivation monoids)

Meta-Conclusions

- unstucking CL (Schönfinkel, Curry, Reynolds, Hughes) for reduction to β-nf
- optimal term graph rewriting implementation of $\ell o \beta$
- cost-model for $\ell o \beta$ (Dal Lago,Accatoli)

implemented $\ell o \beta$ to CL; from CL to TGRS future work

- rewriting useful both for simple description and efficient implementation
- higher-order rewriting useful to bridge λ-calculus $\iff$ supercombinators
- substitution calculus (, van Raamsdonk) useful to modularise TGR
- classical techniques (Schönfinkel, Wadsworth, Hughes,...) for complexity
- Gödel not convinced by $\lambda\beta$ / TRS; me neither because no unit-time steps

current work: amortised complexity via term rewriting (derivation monoids)

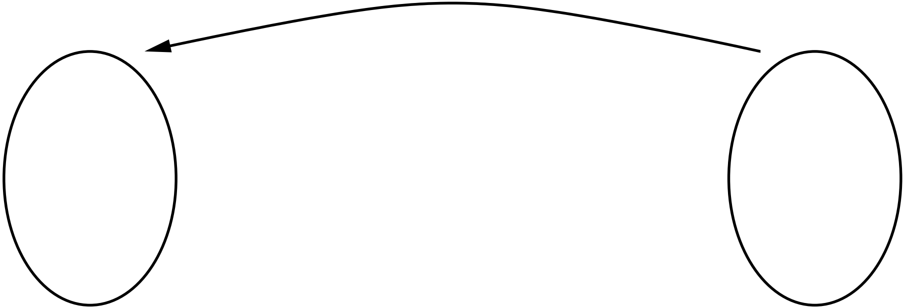
thanks: students, co-workers (Zwitserlood, Hendriks, Grabmayer, Heijltjes,...)

λ-calculus $\implies$ combinator system in Haskell

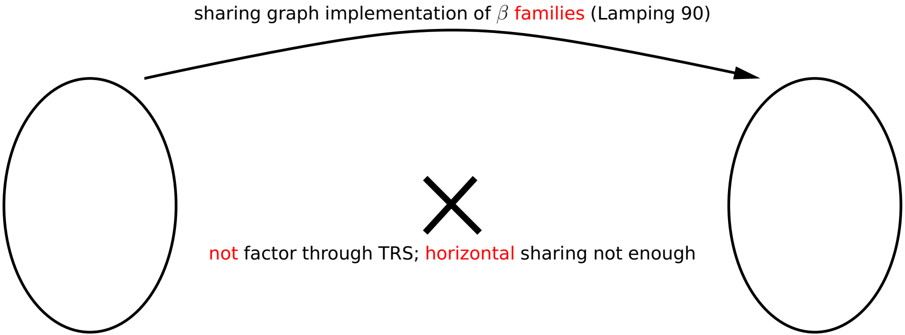
```
data Lam = Lam Head [Lam] deriving (Show)
data Head = Var String | Abs String Lam deriving (Show)
subst x s (Lam h l) = let
  (Lam h' l') = case h of
     (Var y)   | x == y -> s
     (Abs y u) | x /= y -> Lam (Abs y (subst x s u)) []
     _                  -> Lam h [] in (Lam h' (l'++(map (subst x s) l)))
whnf (Lam (Abs x t) (u:l)) = let Lam h s = subst x u t in whnf (Lam h (s++l))
whnf t = t
nf = rnf (\x -> 1)
rnf f t = let
  (Lam h l) = whnf t
  f' x      = \y -> f y + (if (x==y) then 1 else 0)
  v x       = x++"_"++show (f x) in case h of
    (Abs x _) -> Lam (Abs (v x) (rnf (f' x) (Lam h [Lam (Var (v x)) []]))) []
    _         -> Lam h (map (rnf f) l)
```

λ-calculus $\Longleftrightarrow$ interaction nets (Lafont 90), strong

characterisation of optimal β (Lévy 78); families



λ-calculus $\Longleftrightarrow$ interaction nets (Lafont 90), strong



A puzzle to ponder on α-conversion

- give an upperbound on the #α-renamings needed to β-reduce $((\underline{2}\,\underline{8})\,(\underline{4}\,\underline{9}))\,(\underline{5}\,\underline{7})\,(\underline{4}\,\underline{2})$ to normal form?
- note 1: application of Church-numerals is exponentiation; $\underline{k}\,\underline{n} \twoheadrightarrow_\beta \underline{n^k}$
- note 2: whether α-conversion is needed in a β-reduction is undecidable