SwissMAP
The Mathematics of Physics
National Centre of Competence in Research

Présentation SPRING

λ-Calcul

Joé Mighali & Théo von Düring

23 juin 2023

Le λ-Calcul

Le λ-calcus dans son contexte

But du lambda calcul : Fonder alternativement les mathématiques sur les concepts de fonctions et d'opérations plutôt que celui d'ensemble et d'objets.

- Inventé par Alonzo Church dans les années 30.
- Introduction du premier système de typage par Church en 1940
- 1958 : McCarthy crée le langage de programmation LISP inspiré du lambda calcul
- 1969 : Première interprétation du lambda calcul par Dana Scott dans la théorie des ensembles

Intuitions à avoir pour le λ-calcul

Dans le lambda calcul, **Tout est fonctions !**

Au lieu d'avoir une fonction identité par ensemble :

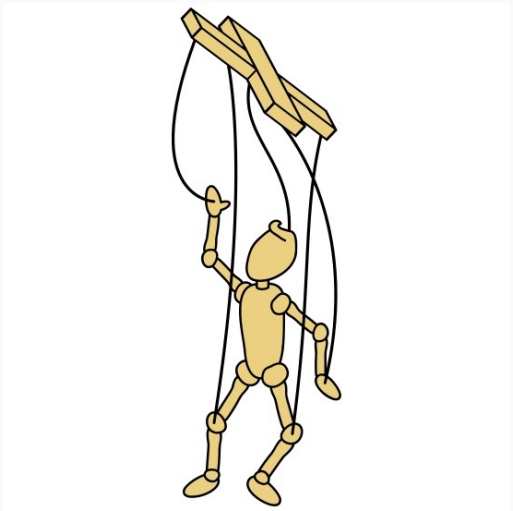
- $\mathbf{I}_{\mathbb{N}} : \mathbb{N} \to \mathbb{N}, \quad x \mapsto x$
- $\mathbf{I}_{\mathbb{Z}} : \mathbb{Z} \to \mathbb{Z}, \quad x \mapsto x$
- $\mathbf{I}_{\mathbb{Q}} : \mathbb{Q} \to \mathbb{Q}, \quad x \mapsto x$
- ...

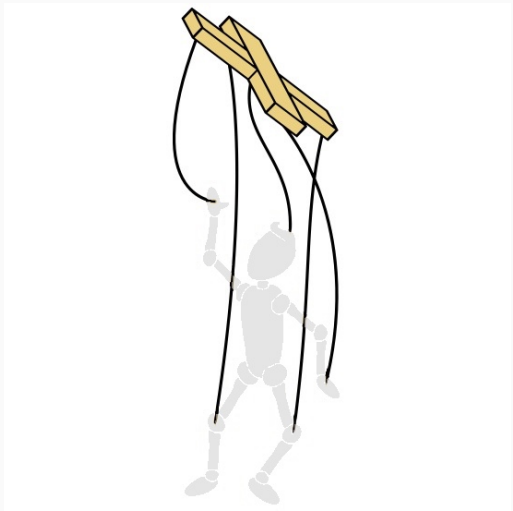
Nous avons une fontion générale $\mathbf{I}$: $\quad x \mapsto x$.

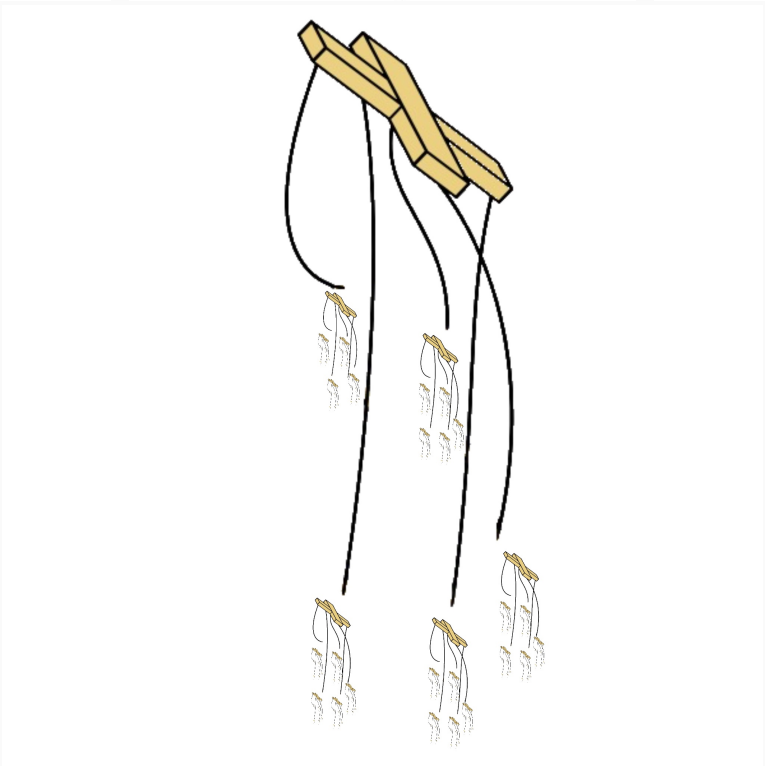
Cela signifie aussi que $\mathbf{I}(\mathbf{I}) = \mathbf{I}$

Church (1941) : "L'étude des propriétés générales des fonctions, indépendemment de leur emploi dans un domain particulier des mathématiques"

Intutions à avoir pour le λ-calcul







Notation du λ-calcul

Pour définir la fonction

$$f(x, y) = x - y$$

nous écrirons à la place

$$\lambda x.(\lambda y.x - y)$$

Quand on évalue la fonction

$$f(1, y) = 1 - y \quad \text{ou} \quad f(x, 1) = x - 1$$

nous écrirons :

$$(\lambda x.(\lambda y.x - y))(1) = \lambda y.(1 - y) \quad \text{et} \quad \lambda x.((\lambda y.x - y)(1)) = \lambda x.x - 1$$

λ-termes

Définition (λ-termes)

(a) toutes les variables et constantes atomiques sont des λ-termes

(b) si M et N sont des λ-termes, alors (MN) est un λ-terme

(c) si M est un λ-terme et x une variable, alors $(\lambda x.M)$ est un λ-terme

Exemple :

- Soit x une variable et f_1 une constante, alors x et f_1 sont des λ-termes (def : a).
- Ainsi, (xf_1) et $\lambda x.f_1$ sont des λ-termes (def : b,c)

Les termes suivants sont-ils des lambdas termes ?

- $\lambda f_1.x$
- $\lambda x.(\lambda y.x)$
- $(\lambda x.x)(\lambda x.x)$
- $\lambda(\lambda x.y).z$

Variables libres et liées

$$\lambda x.\lambda y.(z(xu)y)$$

et

$$VL(\lambda x.\lambda y.(z(xu)y)) = \{z, u\}$$

la β contraction

Tout à l'heure, nous avons vu l'exemple suivant :

$$(\lambda x.(\lambda y.x - y))(1) = \lambda y.(1 - y)$$

Définition

Soit $\lambda x.M$ et N des λ-termes. Un appelle tout terme de la forme $(\lambda x.M)N$ une redex. Et l'opération suivante est une β contraction.

$$(\lambda x.M)N \quad \triangleright_{1\beta} \quad [x \to N]M$$

Si λ-terme ne contient plus de redex, alors ont dit que le terme est une forme normale (β-*nf*).

Si l'on change juste la variable liée par une autre dans un λ-terme, alors on dit que ceux-ci sont α équivalents (noté $\equiv_\alpha$). Exemple :

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

Quelques exemples

1. $(\lambda x.x)y \quad \triangleright_{1\beta} \quad y$
2. $(\lambda y.x)y \quad \triangleright_{1\beta} \quad x$
3. $(\lambda x.(\lambda y.x))ab \quad \triangleright_{1\beta} \quad (\lambda y.a)b \quad \triangleright_{1\beta} \quad a$
4. $(\lambda x.(\lambda y.y))ab \quad \triangleright_{1\beta} \quad (\lambda y.y)b \quad \triangleright_{1\beta} \quad b$
5. $(\lambda x.(\lambda y.xxy))(\lambda x.\lambda y.x)(\lambda x.\lambda y.y) \triangleright_{\beta}$
   $(\lambda x.\lambda y.x)(\lambda x.\lambda y.x)(\lambda x.\lambda y.y) \quad \triangleright_{\beta} \quad (\lambda x.\lambda y.x)$
6. $(\lambda x.xx)(\lambda x.xx) \quad \triangleright_{\beta} \quad (\lambda x.xx)(\lambda x.xx) \quad \triangleright_{\beta} \quad (\lambda x.xx)(\lambda x.xx)...$

Appelons par définition $(\lambda x.(\lambda y.x)) = \overline{true}$ et $(\lambda x.(\lambda y.y)) = \overline{false}$.
Ainsi :

1. $(\lambda x.(\lambda y.xxy))\overline{true}\ \overline{true} \ \triangleright_{\beta} \overline{true}\ \overline{true}\ \overline{true} \ \triangleright_{\beta} \overline{true}$
2. $(\lambda x.(\lambda y.xxy))\overline{true}\ \overline{false} \ \triangleright_{\beta} \overline{true}\ \overline{true}\ \overline{false} \ \triangleright_{\beta} \overline{true}$
3. $(\lambda x.(\lambda y.xxy))\overline{false}\ \overline{true} \ \triangleright_{\beta} \overline{false}\ \overline{false}\ \overline{true} \ \triangleright_{\beta} \overline{true}$
4. $(\lambda x.(\lambda y.xxy))\overline{false}\ \overline{false} \ \triangleright_{\beta} \overline{false}\ \overline{false}\ \overline{false} \ \triangleright_{\beta} \overline{false}$

Plus d'exemples

1. $(\lambda x.(\lambda y.xyx))\overline{true}\ \overline{true} \rhd_\beta \overline{true}\ \overline{true}\ \overline{true} \rhd_\beta \overline{true}$
2. $(\lambda x.(\lambda y.xyx))\overline{true}\ \overline{false} \rhd_\beta \overline{true}\ \overline{false}\ \overline{true} \rhd_\beta \overline{false}$
3. $(\lambda x.(\lambda y.xyx))\overline{false}\ \overline{true} \rhd_\beta \overline{false}\ \overline{true}\ \overline{false} \rhd_\beta \overline{false}$
4. $(\lambda x.(\lambda y.xyx))\overline{false}\ \overline{false} \rhd_\beta \overline{fasle}\ \overline{false}\ \overline{false} \rhd_\beta \overline{false}$

On appelera donc $\overline{and}$ le λ-terme $\lambda x.(\lambda y.xyx)$ et $\overline{or}$ le λ-terme $\lambda x.(\lambda y.xxy)$

On peut reproduire de la même manière avec le λ-terme $((\lambda x.x)\ \overline{false}\ \overline{true})$ la table de vérité de la négation. On appelera ce terme $\overline{not}$

Church-Rosser

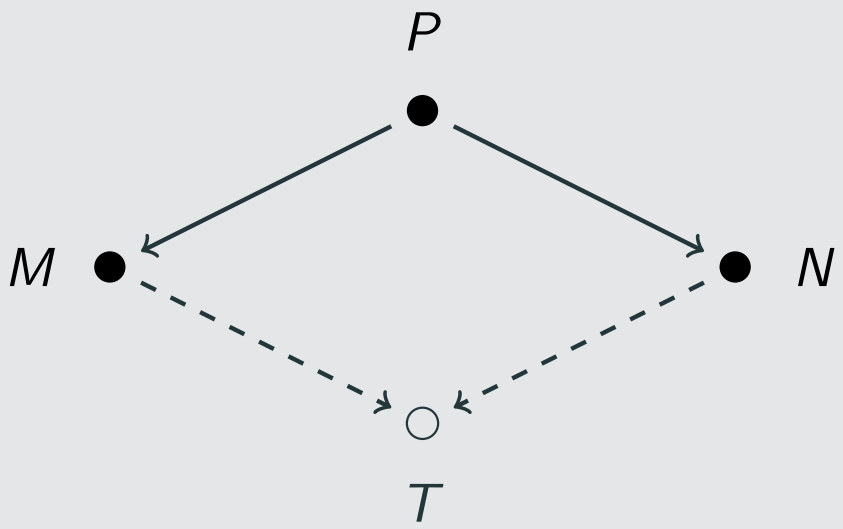
Church-Rosser $\triangleright_\beta$

Prouvé par Alonzo Church et Barkley Rosser en 1935

Théorème (Church-Rosser $\triangleright_\beta$)

Pour tous termes P,M,N

$P \triangleright_\beta M,\ P \triangleright_\beta N \implies (\exists T)\ M \triangleright_\beta T,\ N \triangleright_\beta T$



Church-Rosser $\triangleright_\beta$

Corollaire (Unicité des formes normales $\triangleright_\beta$)

Si P possède une β-nf, elle est unique modulo $\equiv_\alpha$
Autrement dit, si P possèdes les β-nf M et N, alors $M \equiv_\alpha N$

Preuve :
soit $P \triangleright_\beta M$ et $P \triangleright_\beta N$, M et N des β-*nf*
Par Church-Rosser, $(\exists T)$ $M \triangleright_\beta T$ et $N \triangleright_\beta T$
Comme M et N ne contiennent pas de redex, $M \equiv_\alpha N$

β-égalité

Définition (β-égalité)

Pour tous termes P, Q

$P =_\beta Q$ ssi Q peut être obtenu à partir de P par une suite finie de β-contractions, β-contractions inverse et α-conversions.

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$
$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

$\triangleright_\beta \lambda yz.y((\lambda xu.x(x(xu)))yz)$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

$\triangleright_\beta \lambda yz.y((\lambda xu.x(x(xu)))yz)$

$\triangleright_\beta \lambda yz.y(y(y(yz)))$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

$\triangleright_\beta \lambda yz.y((\lambda xu.x(x(xu)))yz)$

$\triangleright_\beta \lambda yz.y(y(y(yz)))$

$(\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

$\triangleright_\beta \lambda yz.y((\lambda xu.x(x(xu)))yz)$

$\triangleright_\beta \lambda yz.y(y(y(yz)))$

$(\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta (\lambda xy.x(xy))(\lambda xy.x(xy))$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

$\triangleright_\beta \lambda yz.y((\lambda xu.x(x(xu)))yz)$

$\triangleright_\beta \lambda yz.y(y(y(yz)))$

$(\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta (\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))((\lambda xu.x(xu))y)$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

$\triangleright_\beta \lambda yz.y((\lambda xu.x(x(xu)))yz)$

$\triangleright_\beta \lambda yz.y(y(y(yz)))$

$(\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta (\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))((\lambda xu.x(xu))y)$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))(\lambda u.y(yu))$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

$\triangleright_\beta \lambda yz.y((\lambda xu.x(x(xu)))yz)$

$\triangleright_\beta \lambda yz.y(y(y(yz)))$

$(\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta (\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))((\lambda xu.x(xu))y)$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))(\lambda u.y(yu))$

$\triangleright_\beta \lambda y.(\lambda u.(\lambda v.y(yv))((\lambda v.y(yv))u))$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

$\triangleright_\beta \lambda yz.y((\lambda xu.x(x(xu)))yz)$

$\triangleright_\beta \lambda yz.y(y(y(yz)))$

$(\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta (\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))((\lambda xu.x(xu))y)$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))(\lambda u.y(yu))$

$\triangleright_\beta \lambda y.(\lambda u.(\lambda v.y(yv))((\lambda v.y(yv))u))$

$\triangleright_\beta \lambda y.(\lambda u.(\lambda v.y(yv))(y(yu)))$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

$\triangleright_\beta \lambda yz.y((\lambda xu.x(x(xu)))yz)$

$\triangleright_\beta \lambda yz.y(y(y(yz)))$

$(\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta (\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))((\lambda xu.x(xu))y)$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))(\lambda u.y(yu))$

$\triangleright_\beta \lambda y.(\lambda u.(\lambda v.y(yv))((\lambda v.y(yv))u))$

$\triangleright_\beta \lambda y.(\lambda u.(\lambda v.y(yv))(y(yu)))$

$\triangleright_\beta \lambda y.(\lambda u.y(y(y(yu)))) \quad \equiv \quad \lambda yu.y(y(y(yu)))$

β-égalité

Exemple

Montrer

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$(\lambda xyz.y(xyz))(\lambda xy.x(x(xy)))$

$\triangleright_\beta \lambda yz.y((\lambda xu.x(x(xu)))yz)$

$\triangleright_\beta \lambda yz.y(y(y(yz)))$

$(\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta (\lambda xy.x(xy))(\lambda xy.x(xy))$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))((\lambda xu.x(xu))y)$

$\triangleright_\beta \lambda y.(\lambda xu.x(xu))(\lambda u.y(yu))$

$\triangleright_\beta \lambda y.(\lambda u.(\lambda v.y(yv))((\lambda v.y(yv))u))$

$\triangleright_\beta \lambda y.(\lambda u.(\lambda v.y(yv))(y(yu)))$

$\triangleright_\beta \lambda y.(\lambda u.y(y(y(yu)))) \quad \equiv \quad \lambda yu.y(y(y(yu)))$

$\lambda yz.y(y(y(yz))) \quad \equiv_\alpha \quad \lambda yu.y(y(y(yu)))$

$\Rightarrow (\lambda xyz.y(xyz))(\lambda xy.x(x(xy))) =_\beta (\lambda xy.yx)(\lambda xy.x(xy))(\lambda xy.x(xy))$

Church-Rosser $=_\beta$

Théorème (Church-Rosser $=_\beta$)

Pour tous termes P, Q

$P =_\beta Q \implies (\exists T)\ P \triangleright_\beta T,\ Q \triangleright_\beta T$

Corollaire (1)

Si $P =_\beta Q$ et Q est une β-nf, alors $P \triangleright_\beta Q$

Corollaire

Si $P =_\beta Q$, alors soit P et Q possèdent la **même** *β-nf soit P et Q ne possède pas de β-nf*

Corollaire (Unicité des formes normales $=_\beta$)

Un λ-terme est β-égal à au plus une β-nf modulo $\equiv_\alpha$

Preuve : par (1) et l'unicité des formes normales pour $\triangleright_\beta$

Nombres naturels et opérations

Nombres de Church

Intuition pour les nombres naturels

On peut exprimer tous les nombres naturels avec :

- le nombre 0
- la fonction $succ(x) = x + 1$

en appliquant plusieurs fois la fonction *succ* au nombre 0.

Dans cette partie on notera $f^n(x)$ (et $f^n x$ en λ-calcul) pour parler de $\underbrace{f(f(...f}_{n\ 'f'}(x)...))$

ainsi $0 = 0,\ 1 = succ(0),\ 2 = succ(succ(0)),\ ...,\ n = succ^n(0)$

Nombres de Church

Définition

Pour tout $n \in \mathbb{N}$, le nombre de church pour n est un terme appelé $\overline{n}$ défini comme suit :

$$\overline{n} \quad \equiv \quad \lambda xy.x^n y$$

On peut remarquer :

- pour tous termes F et X $\quad \overline{n}FX \quad \triangleright_\beta \quad F^n X$
- $(\forall n \in \mathbb{N})$ $\overline{n}$ est une β-nf

On utilisera la β-égalité pour comparer l'égalité entre les nombres de Church et les résultats des opérations

Opérations

Succésseur

$$\overline{succ} \equiv \lambda nxy.x(nxy)$$

$$\begin{aligned}
\overline{succ}\ \overline{a} &\equiv (\lambda nxy.x(nxy))(\lambda xy.x^a y) \\
&\triangleright_\beta (\lambda xy.x((\lambda uv.u^a v)xy)) \\
&\triangleright_\beta (\lambda xy.x(x^a y)) \\
&\equiv (\lambda xy.x^{a+1} y) \\
&=_\beta \overline{a+1}
\end{aligned}$$

Opérations

Succésseur

$\overline{succ} \quad \equiv \quad \lambda nxy.x(nxy)$

Addition

On cherche une fonction $f(x, y) = x + y$

$\overline{add} \quad \equiv \quad \lambda mn.m\ \overline{succ}\ n$

$$\begin{aligned}
\overline{add}\ \overline{a}\ \overline{b} &\equiv (\lambda mn.m\ \overline{succ}\ n)(\lambda xy.x^a y)\overline{b} \\
&\triangleright_\beta (\lambda n.(\lambda xy.x^a y)\ \overline{succ}\ n)\overline{b} \\
&\triangleright_\beta (\lambda n.\overline{succ}^a n)\overline{b} \\
&\triangleright_\beta \overline{succ}^a\ \overline{b} \\
&=_\beta \overline{a + b}
\end{aligned}$$

Opérations

Succésseur

$\overline{succ} \equiv \lambda nxy.x(nxy)$

Addition

On cherche une fonction $f(x, y) = x + y$

$\overline{add} \equiv \lambda mn.m\ \overline{succ}\ n$

Multiplication

On cherche une fonction $f(x, y) = x \cdot y$

$\overline{mul} \equiv \lambda mnx.m(nx)$

$\overline{mul} \equiv \lambda mn.m(\overline{add}\ n)\overline{0}$

$$
\begin{aligned}
\overline{mul}\ \overline{a}\ \overline{b} &\equiv (\lambda mn.m(\overline{add}\ n)\overline{0})(\lambda xy.x^a y)\overline{b} \\
&\triangleright_\beta (\lambda n.(\lambda xy.x^a y)(\overline{add}\ n)\overline{0})\overline{b} \\
&\triangleright_\beta (\lambda xy.x^a y)(\overline{add}\ \overline{b})\overline{0} \\
&\triangleright_\beta (\overline{add}\ \overline{b})^a\ \overline{0} \\
&=_\beta \overline{a \cdot b}
\end{aligned}
$$

Opérations

Succésseur

$\overline{succ} \quad \equiv \quad \lambda nxy.x(nxy)$

Addition

On cherche une fonction $f(x, y) = x + y$

$\overline{add} \quad \equiv \quad \lambda mn.m\ \overline{succ}\ n$

Multiplication

On cherche une fonction $f(x, y) = x \cdot y$

$\overline{mul} \quad \equiv \quad \lambda mnx.m(nx)$

$\overline{mul} \quad \equiv \quad \lambda mn.m(\overline{add}\ n)\overline{0}$

Exponentiation

On cherche une fonction $f(x, y) = x^y$

$\overline{pow} \quad \equiv \quad \lambda mn.nm$

Opérations

Succésseur

$\overline{succ} \equiv \lambda nxy.x(nxy)$

Addition

On cherche une fonction $f(x, y) = x + y$

$\overline{add} \equiv \lambda mn.m\ \overline{succ}\ n$

Multiplication

On cherche une fonction $f(x, y) = x \cdot y$

$\overline{mul} \equiv \lambda mnx.m(nx)$

$\overline{mul} \equiv \lambda mn.m(\overline{add}\ n)\overline{0}$

Exponentiation

On cherche une fonction $f(x, y) = x^y$

$\overline{pow} \equiv \lambda mn.(\overline{isZero}\ n)(\overline{1})(nm)$

$\overline{pow} \equiv \lambda mn.n(\overline{mul}\ m)\overline{1}$

Opérations

Prédécesseur

On cherche une fonction $f(x) = max(0, x - 1)$

$\overline{pred} \quad \equiv \quad \lambda nxy.n(\lambda uv.v(ux))(\lambda u.y)(\lambda u.u)$

Opérations

Prédécesseur

On cherche une fonction $f(x) = max(0, x - 1)$

$\overline{pred} \quad \equiv \quad \lambda nxy.n(\lambda uv.v(ux))(\lambda u.y)(\lambda u.u)$

Soustraction

On cherche une fonction $f(x, y) = max(0, x - y)$

$\overline{sub} \quad \equiv \quad \lambda mn.n\ \overline{pred}\ m$

Opérations

Prédécesseur

On cherche une fonction $f(x) = max(0, x-1)$

$$\overline{pred} \quad \equiv \quad \lambda nxy.n(\lambda uv.v(ux))(\lambda u.y)(\lambda u.u)$$

Soustraction

On cherche une fonction $f(x,y) = max(0, x-y)$

$$\overline{sub} \quad \equiv \quad \lambda mn.n\ \overline{pred}\ m$$

Fonction caractéristique 0

On cherche une fonction $f(x) = \begin{cases} \overline{true} & \text{si } x =_\beta \overline{0} \\ \overline{false} & \text{sinon} \end{cases}$

$$\overline{isZero} \quad \equiv \quad \lambda n.n\ \overline{false}\ \overline{not}\ \overline{false}$$

$$\begin{aligned} \overline{isZero}\ \overline{a} \quad &\equiv \quad (\lambda n.n\ \overline{false}\ \overline{not}\ \overline{false})(\lambda xy.x^a y) \\ &\rhd_\beta \quad \overline{false}^a\ \overline{not}\ \overline{false} \\ &\rhd_\beta \quad \begin{cases} \overline{not}\ \overline{false} & \rhd_\beta \quad \overline{true} \quad a = 0 \\ \overline{false}(\overline{false}^{a-1}\ \overline{not})\ \overline{false} & \rhd_\beta \quad \overline{false} \quad a \neq 0 \end{cases} \end{aligned}$$

Opérations

Plus grand ou égal

On cherche une fonction $f(x, y) = x \geq y$

$\overline{geq} \quad \equiv \quad \lambda mn.\overline{isZero}(\overline{sub}\ nm)$

Opérations

Plus grand ou égal

On cherche une fonction $f(x, y) = x \geq y$

$\overline{geq} \quad \equiv \quad \lambda mn.\overline{isZero}(\overline{sub}\ nm)$

Plus petit ou égal

On cherche une fonction $f(x, y) = x \leq y$

$\overline{leq} \quad \equiv \quad \lambda mn.\overline{isZero}(\overline{sub}\ mn)$

Opérations

Plus grand ou égal

On cherche une fonction $f(x, y) = x \geq y$

$\overline{geq} \quad \equiv \quad \lambda mn.\overline{isZero}(\overline{sub}\ nm)$

Plus petit ou égal

On cherche une fonction $f(x, y) = x \leq y$

$\overline{leq} \quad \equiv \quad \lambda mn.\overline{isZero}(\overline{sub}\ mn)$

Plus grand

On cherche une fonction $f(x, y) = x > y$

$\overline{greater} \quad \equiv \quad \lambda mn.\overline{and}(\overline{geq}\ mn)(\overline{not}(\overline{leq}\ mn))$

Opérations

Plus grand ou égal

On cherche une fonction $f(x, y) = x \geq y$

$\overline{geq} \quad \equiv \quad \lambda mn.\overline{isZero}(\overline{sub}\ nm)$

Plus petit ou égal

On cherche une fonction $f(x, y) = x \leq y$

$\overline{leq} \quad \equiv \quad \lambda mn.\overline{isZero}(\overline{sub}\ mn)$

Plus grand

On cherche une fonction $f(x, y) = x > y$

$\overline{greater} \quad \equiv \quad \lambda mn.\overline{and}(\overline{geq}\ mn)(\overline{not}(\overline{leq}\ mn))$

Plus petit

On cherche une fonction $f(x, y) = x < y$

$\overline{less} \quad \equiv \quad \lambda mn.\overline{and}(\overline{leq}\ mn)(\overline{not}(\overline{geq}\ mn))$

Opérations

Plus grand ou égal

On cherche une fonction $f(x, y) = x \geq y$

$\overline{geq} \quad \equiv \quad \lambda mn.\overline{isZero}(\overline{sub}\ nm)$

Plus petit ou égal

On cherche une fonction $f(x, y) = x \leq y$

$\overline{leq} \quad \equiv \quad \lambda mn.\overline{isZero}(\overline{sub}\ mn)$

Plus grand

On cherche une fonction $f(x, y) = x > y$

$\overline{greater} \quad \equiv \quad \lambda mn.\overline{and}(\overline{geq}\ mn)(\overline{not}(\overline{leq}\ mn))$

Plus petit

On cherche une fonction $f(x, y) = x < y$

$\overline{less} \quad \equiv \quad \lambda mn.\overline{and}(\overline{leq}\ mn)(\overline{not}(\overline{geq}\ mn))$

Egalité

On cherche une fonction $f(x, y) = x$ is y

$\overline{equal} \quad \equiv \quad \lambda mn.\overline{and}(\overline{leq}\ mn)(\overline{geq}\ mn)$

Théorème du point fixe

Un point point fixe d'une fonction ou opération est un objet qui ne change pas quand l'opération lui est appliquée.
ex. $f(x) = x^2$ possède 2 points fixes 0 et 1 car $f(0) = 0$ et $f(1) = 1$
Le théorème du point fixe montre que

$$(\forall X)\ (\exists P)\ \text{t.q.}\ XP =_\beta P$$

Théorème (Théorème du point fixe)

Dans le λ-calcul il exist un terme $\mathbf{Y}$ *tel que*

$$\mathbf{Y}x \quad \triangleright_\beta \quad x(\mathbf{Y}x)$$

Un $\mathbf{Y}$ *satisfaisant cette condition a été trouvé par Alan Turing en 1937*

$$Y \equiv UU, \quad \text{où} \quad U \equiv \lambda ux.x(uux)$$

Théorème du point fixe

$$
\begin{aligned}
\mathbf{Y}x &\equiv (\lambda ux.x(uux))Ux \\
&\rhd_\beta [u \to U](\lambda x.x(uux))x \\
&\equiv (\lambda x.x(UU))x \\
&\rhd_\beta x(UUx) \\
&\equiv x(\mathbf{Y}x)
\end{aligned}
$$

Théorème du point fixe

Corollaire

Pour tout terme Z et $n \geq 0$, l'équation

$$xy_1 \ldots y_n \quad = \quad Z$$

peut être résolue pour x. Il existe donc un terme X tel que

$$Xy_1 \ldots y_n \quad =_\beta \quad [x \to X]Z$$

Tout simplement $X \equiv \mathbf{Y}(\lambda x y_1 \ldots y_n.Z)$. On a donc :

$$\begin{aligned} Xy_1 \ldots y_n \equiv\ & \mathbf{Y}(\lambda x y_1 \ldots y_n.Z) y_1 \ldots y_n \\ \rhd_\beta\ & (\lambda x y_1 \ldots y_n.Z)(\mathbf{Y}(\lambda x y_1 \ldots y_n.Z)) y_1 \ldots y_n \\ \rhd_\beta\ & (\lambda y_1 \ldots y_n.[x \to \mathbf{Y}(\lambda x y_1 \ldots y_n.Z)]Z) y_1 \ldots y_n \\ \equiv\ & (\lambda y_1 \ldots y_n.[x \to X]Z) y_1 \ldots y_n \\ \rhd_\beta\ & [x \to X]Z \end{aligned}$$

Opération factorielle

Factorielle

On cherche une fonction $f(x) = x!$

Notre fonction devrait se comporter comme suit :

$fact(n) \ = \ n \ is \ 0 \ ? \ 1 \ : \ n \cdot (fact(n-1))$

Opération factorielle

Factorielle

On cherche une fonction $f(x) = x!$

Notre fonction devrait se comporter comme suit :

$\overline{fact}\ n\ =\ (\overline{isZero}\ n)(\overline{1})(\overline{mul}\ n\ (\overline{fact}(\overline{pred}\ n)))$

Opération factorielle

Factorielle

On cherche une fonction $f(x) = x!$

Notre fonction devrait se comporter comme suit :

$$\underbrace{\overline{fact}}_{X}\underbrace{n}_{y_1} = \underbrace{(\overline{isZero}\ n)(\overline{1})(\overline{mul}\ n\ (\overline{fact}(\overline{pred}\ n)))}_{[x \to X]Z}$$

Point fixe :

$$\overline{fact} \equiv \mathbf{Y}(\lambda xn.(\overline{isZero}\ n)(\overline{1})(\overline{mul}\ n\ (x(\overline{pred}\ n))))$$

Rappel

L'équation $xy_1...y_n = Z$ peut être résolue pour x
$(Xy_1...y_n =_\beta [x \to X]Z)$
avec $X \equiv \mathbf{Y}(\lambda xy_1...y_n.Z)$

Opération Division

Division entière

On cherche une fonction $f(x, y) = \lfloor x/y \rfloor$

Notre fonction devrait se comporter comme suit :

$div(m, n) \ = \ m \ \geq \ n \ ? \ 1 + div(m - n, n) \ : \ 0$

Opération Division

Division entière

On cherche une fonction $f(x, y) = \lfloor x/y \rfloor$

Notre fonction devrait se comporter comme suit :

$\overline{div}\ mn\ =\ (\overline{geq}\ mn)(\overline{add}\ \overline{1}(\overline{div}(\overline{sub}\ mn)n))(\overline{0})$

Opération Division

Division entière

On cherche une fonction $f(x, y) = \lfloor x/y \rfloor$

Notre fonction devrait se comporter comme suit :

$$\underbrace{\overline{div}}_{X}\underbrace{mn}_{y_1 y_2} = \underbrace{(\overline{geq}\ mn)(\overline{add}\ \overline{1}(\overline{div}(\overline{sub}\ mn)n))(\overline{0})}_{[x \to X]Z}$$

Point fixe :

$$\overline{div} \equiv \mathbf{Y}(\lambda xmn.(\overline{geq}\ mn)(\overline{add}\ \overline{1}(x(\overline{sub}\ mn)n))(\overline{0}))$$

Rappel

L'équation $xy_1...y_n = Z$ peut être résolue pour x
$(Xy_1...y_n =_\beta [x \to X]Z)$
avec $X \equiv \mathbf{Y}(\lambda xy_1...y_n.Z)$

Indécidabilité

Le théorème d'indécidabilité

La forme normale d'un λ-terme = résultat du calcul.

Chaque réduction = une étape du calcul.

Ainsi si un lambda terme n'a pas de forme normale, c'est que la fonction n'est pas calculable. (Exemple : la formule $\Omega = (\lambda x.xx)(\lambda x.xx)$.)

Il est possible d'attribuer par codage à chaque λ-terme un nombre code unique. Soit C un l'ensemble de ces nombres codes. Ainsi on peut définir $\overline{C} = \{\overline{n} \mid n \in C\}$

On dit qu'un ensemble E est algorithmiquement décidable ssi il existe pour son ensemble code $\overline{C_E}$ un λ-terme L t.q

$$L\overline{n} = \begin{cases} \lambda x.\lambda y.x = \overline{true} & \text{if } \overline{n} \in \overline{C_E} \\ \lambda x.\lambda y.y = \overline{false} & \text{if } \overline{n} \notin \overline{C_E} \end{cases}$$

Le théorème d'indécidabilité

Théorème

L'ensemble des tous les λ-termes sous forme normale n'est pas algorithmiquement décidable. Ainsi, aucun λ-terme n'est capable décider si un λ-terme possède une forme normale ou non.

Remarques

1. On peut attribuer aux lambda termes des **types** qui restreint certaines fonction à ne pas s'appliquer à d'autre. Très utile dans la programmation. (Par exemple : on définira différement la multiplication dans les nombres naturels que dans les nombres complexes)
2. Il existe un système appelé la logique combinatoire qui est composé uniquement de 3 combinateurs (i.e uniquement 3 fonctions initiale) qui sont **I**, **K**, **S** et qui possèdes les mêmes porpriétés du lambda calcul de manière analogue.
3. La logique combinatoire, le λ-calcul, les machines de Turing et les fonctions récursives de Gödel se sont avérés être des systèmes équivalents entre eux. La thèse de Church-Turing postule que ces systèmes définissent la notion de calculabilité.