

Programmation Fonctionnelle
Fondaments de la programmation
fonctionnelle :
le λ -calcul non typé

Luigi Santocanale
LIF, Aix-Marseille Université
Marseille, FRANCE

14 octobre 2014

Plan

Le λ -calcul

Confluence

Stratégies d'évaluations

Plan

Le λ -calcul

Confluence

Stratégies d'évaluations

Le λ -calcul

Ensemble de termes (ou expressions) définis par la grammaire suivante :

$$T = x \mid T T \mid \lambda x. T$$

Exemple. Les suivants sont des termes :

$$\lambda x. xx, (\lambda x. (xx))y, \dots$$

Un terme de la forme $\lambda x. ts$ est appelé *application* ;
il est noté $\lambda x. ts$.

Le λ -calcul

Ensemble de termes (ou expressions) définis par la grammaire suivante :

$$T = x \mid T T \mid \lambda x. T$$

Exemple. Les suivants sont des termes :

$$\lambda x. xx, (\lambda x. (xx))y, \dots$$

Un terme de la forme $\lambda x. t$ est appelé *application* ;
abstraction .

Le λ -calcul

Ensemble de termes (ou expressions) définis par la grammaire suivante :

$$T = x \mid T T \mid \lambda x. T$$

Exemple. Les suivants sont des termes :

$$\lambda x. xx, (\lambda x. (xx))y, \dots$$

Un terme de la forme $t s$ est appelé *application* ;
un terme de la forme $\lambda x. t$ est appelé *abstraction* .

Le λ -calcul

Ensemble de termes (ou expressions) définis par la grammaire suivante :

$$T = x \mid T T \mid \lambda x. T$$

Exemple. Les suivants sont des termes :

$$\lambda x. xx, (\lambda x. (xx))y, \dots$$

Un terme de la forme $\lambda x. t s$ est appelé *application* ;
abstraction .

Variables libres et liées

Variable libres (Free) :

$$FV(x) = \{x\},$$

$$FV(t s) = FV(t) \cup FV(s),$$

$$FV(\lambda x.t) = FV(t) \setminus \{x\}.$$

Variables liées (Bound) :

$$BV(x) = \emptyset,$$

$$BV(t s) = BV(t) \cup BV(s),$$

$$BV(\lambda x.t) = BV(t) \cup \{x\}.$$

Substitution (I)

$$x[r/y] = \begin{cases} x, & \text{si } x \neq y, \\ r, & \text{si } x = y; \end{cases}$$

$$(t s)[r/y] = t[r/y] s[r/y];$$

$$(\lambda x.t)[r/y] = \begin{cases} \lambda x.t, & \text{si } x = y, \\ \lambda x.(t[r/y]), & \text{si } x \neq y (**). \end{cases}$$

(**): Cette règle s'applique sous la contrainte

$$BV(t) \cap FV(r) = \emptyset,$$

pour éviter le « variable capture », c'est-à-dire qu'une variable libre dans r devient liée après substitution—comme par exemple dans :

$$(\lambda x.xy)[x/y] \rightsquigarrow \lambda x.xx.$$

Substitution (I)

$$x[r/y] = \begin{cases} x, & \text{si } x \neq y, \\ r, & \text{si } x = y; \end{cases}$$

$$(t s)[r/y] = t[r/y] s[r/y];$$

$$(\lambda x.t)[r/y] = \begin{cases} \lambda x.t, & \text{si } x = y, \\ \lambda x.(t[r/y]), & \text{si } x \neq y (***) . \end{cases}$$

(***) : Cette règle s'applique sous la contrainte

$$BV(t) \cap FV(r) = \emptyset,$$

pour éviter le « variable capture », c'est-à-dire qu'une variable libre dans r devient liée après substitution—comme par exemple dans :

$$(\lambda x.xy)[x/y] \rightsquigarrow \lambda x.xx.$$

Substitution (I)

$$x[r/y] = \begin{cases} x, & \text{si } x \neq y, \\ r, & \text{si } x = y; \end{cases}$$

$$(t s)[r/y] = t[r/y] s[r/y];$$

$$(\lambda x.t)[r/y] = \begin{cases} \lambda x.t, & \text{si } x = y, \\ \lambda x.(t[r/y]), & \text{si } x \neq y (**). \end{cases}$$

(**): Cette règle s'applique sous la contrainte

$$BV(t) \cap FV(r) = \emptyset,$$

pour éviter le « variable capture », c'est-à-dire qu'une variable libre dans r devient liée après substitution—comme par exemple dans :

$$(\lambda x.xy)[x/y] \rightsquigarrow \lambda x.xx.$$

Substitution (I)

$$x[r/y] = \begin{cases} x, & \text{si } x \neq y, \\ r, & \text{si } x = y; \end{cases}$$

$$(t s)[r/y] = t[r/y] s[r/y];$$

$$(\lambda x.t)[r/y] = \begin{cases} \lambda x.t, & \text{si } x = y, \\ \lambda x.(t[r/y]), & \text{si } x \neq y (***) . \end{cases}$$

(***) : Cette règle s'applique sous la contrainte

$$BV(t) \cap FV(r) = \emptyset,$$

pour éviter le « variable capture », c'est-à-dire qu'une variable libre dans r devient liée après substitution—comme par exemple dans :

$$(\lambda x.xy)[x/y] \overset{?}{\rightsquigarrow} \lambda x.xx.$$

α -équivalence

La substitution est partiellement définie :

comment définir $(\lambda x.(xy))[x/y]$?

On peut répondre en déclarant équivalentes les abstractions par rapport à variables différentes (invariance d'une fonction par rapport aux paramètres formels). Par exemple :

$$\lambda x.xy \sim \lambda z.zy \quad (\text{mais } \lambda x.xy \not\sim \lambda y.yy).$$

Règle générale :

$$\begin{array}{l} X \sim_{\alpha} X, \\ t \sim_{\alpha} t' \text{ et } s \sim_{\alpha} s' \text{ implique } ts \sim_{\alpha} t's', \\ t \sim_{\alpha} t' \text{ et } z \notin \text{FV}(t') \cup \text{BV}(t') \text{ implique } \lambda x.t \sim_{\alpha} \lambda z.t'[z/x]. \end{array}$$

Si $t \sim_{\alpha} s$, alors dit que t et s sont α -équivalentes,

ou bien que s est une α -variante de t .

α -équivalence

La substitution est partiellement définie :

comment définir $(\lambda x.(xy))[x/y]$?

On peut répondre en déclarant équivalentes les abstractions par rapport à variables différentes (invariance d'une fonction par rapport aux paramètres formels). Par exemple :

$$\lambda x.xy \sim \lambda z.zy \quad (\text{mais } \lambda x.xy \not\sim \lambda y.yy).$$

Règle générale :

$$\begin{array}{l} X \sim_{\alpha} X, \\ t \sim_{\alpha} t' \text{ et } s \sim_{\alpha} s' \text{ implique } ts \sim_{\alpha} t's', \\ t \sim_{\alpha} t' \text{ et } z \notin \text{FV}(t') \cup \text{BV}(t') \text{ implique } \lambda x.t \sim_{\alpha} \lambda z.t'[z/x]. \end{array}$$

Si $t \sim_{\alpha} s$, alors dit que t et s sont α -équivalentes,

ou bien que s est une α -variante de t .

α -équivalence

La substitution est partiellement définie :

comment définir $(\lambda x.(xy))[x/y]$?

On peut répondre en déclarant équivalentes les abstractions par rapport à variables différentes (invariance d'une fonction par rapport aux paramètres formels). Par exemple :

$$\lambda x.xy \sim \lambda z.zy \quad (\text{mais } \lambda x.xy \not\sim \lambda y.yy).$$

Règle générale :

$$\begin{array}{l} X \sim_{\alpha} X, \\ t \sim_{\alpha} t' \text{ et } s \sim_{\alpha} s' \text{ implique } ts \sim_{\alpha} t' s', \\ t \sim_{\alpha} t' \text{ et } z \notin \text{FV}(t') \cup \text{BV}(t') \text{ implique } \lambda x.t \sim_{\alpha} \lambda z.t'[z/x]. \end{array}$$

Si $t \sim_{\alpha} s$, alors dit que t et s sont α -équivalentes,
ou bien que s est une α -variante de t .

α -équivalence

La substitution est partiellement définie :

comment définir $(\lambda x.(xy))[x/y]$?

On peut répondre en déclarant équivalentes les abstractions par rapport à variables différentes (invariance d'une fonction par rapport aux paramètres formels). Par exemple :

$$\lambda x.xy \sim \lambda z.zy \quad (\text{mais } \lambda x.xy \not\sim \lambda y.yy).$$

Règle générale :

$$\begin{array}{l} X \sim_{\alpha} X, \\ t \sim_{\alpha} t' \text{ et } s \sim_{\alpha} s' \text{ implique } ts \sim_{\alpha} t' s', \\ t \sim_{\alpha} t' \text{ et } z \notin \text{FV}(t') \cup \text{BV}(t') \text{ implique } \lambda x.t \sim_{\alpha} \lambda z.t'[z/x]. \end{array}$$

Si $t \sim_{\alpha} s$, alors dit que t et s sont **α -équivalentes**,
ou bien que s est une **α -variante** de t .

Substitution (II)

$$x[r/y] = \begin{cases} x, & \text{si } x \neq y, \\ r, & \text{si } x = y; \end{cases}$$

$$(t s)[r/y] = t[r/y] s[r/y];$$

$$(\lambda x.t)[r/y] = \begin{cases} \lambda x.t, & \text{si } x = y, \\ \lambda x.(t[r/y]), & \text{si } x \neq y \text{ et } \text{BV}(t) \cap \text{FV}(r) = \emptyset, \\ v[r/y], & \text{sinon,} \\ & \text{où } v \sim_{\alpha} \lambda x.t \text{ et} \\ & \text{BV}(v) \cap \text{FV}(r) = \emptyset. \end{cases}$$

Exemple : $\lambda x.xy \sim_{\alpha} \lambda z.zy$, donc :

$$(\lambda x.xy)[x/y] = (\lambda z.zy)[x/y] = \lambda z.(z[x/y]y[x/y]) = \lambda z.zx.$$

Substitution (II)

$$x[r/y] = \begin{cases} x, & \text{si } x \neq y, \\ r, & \text{si } x = y; \end{cases}$$

$$(t s)[r/y] = t[r/y] s[r/y];$$

$$(\lambda x.t)[r/y] = \begin{cases} \lambda x.t, & \text{si } x = y, \\ \lambda x.(t[r/y]), & \text{si } x \neq y \text{ et } \text{BV}(t) \cap \text{FV}(r) = \emptyset, \\ v[r/y], & \text{sinon,} \\ & \text{où } v \sim_{\alpha} \lambda x.t \text{ et} \\ & \text{BV}(v) \cap \text{FV}(r) = \emptyset. \end{cases}$$

Exemple : $\lambda x.xy \sim_{\alpha} \lambda z.zy$, donc :

$$(\lambda x.xy)[x/y] = (\lambda z.zy)[x/y] = \lambda z.(z[x/y]y[x/y]) = \lambda z.zx.$$

Substitution (II)

$$x[r/y] = \begin{cases} x, & \text{si } x \neq y, \\ r, & \text{si } x = y; \end{cases}$$

$$(t s)[r/y] = t[r/y] s[r/y];$$

$$(\lambda x.t)[r/y] = \begin{cases} \lambda x.t, & \text{si } x = y, \\ \lambda x.(t[r/y]), & \text{si } x \neq y \text{ et } \text{BV}(t) \cap \text{FV}(r) = \emptyset, \\ v[r/y], & \text{sinon,} \\ & \text{où } v \sim_{\alpha} \lambda x.t \text{ et} \\ & \text{BV}(v) \cap \text{FV}(r) = \emptyset. \end{cases}$$

Exemple : $\lambda x.xy \sim_{\alpha} \lambda z.zy$, donc :

$$(\lambda x.xy)[x/y] = (\lambda z.zy)[x/y] = \lambda z.(z[x/y]y[x/y]) = \lambda z.zx.$$

Substitution (II)

$$x[r/y] = \begin{cases} x, & \text{si } x \neq y, \\ r, & \text{si } x = y; \end{cases}$$

$$(t s)[r/y] = t[r/y] s[r/y];$$

$$(\lambda x.t)[r/y] = \begin{cases} \lambda x.t, & \text{si } x = y, \\ \lambda x.(t[r/y]), & \text{si } x \neq y \text{ et } \text{BV}(t) \cap \text{FV}(r) = \emptyset, \\ v[r/y], & \text{sinon,} \\ & \text{où } v \sim_{\alpha} \lambda x.t \text{ et} \\ & \text{BV}(v) \cap \text{FV}(r) = \emptyset. \end{cases}$$

Exemple : $\lambda x.xy \sim_{\alpha} \lambda z.zy$, donc :

$$(\lambda x.xy)[x/y] = (\lambda z.zy)[x/y] = \lambda z.(z[x/y]y[x/y]) = \lambda z.zx.$$

β -réduction

Loi fondamentale du λ -calcul :

$$(\lambda x.t) s \rightarrow_{\beta} t[x/s]$$

*La fonction $\lambda x.t$ appliquée à son argument s ,
s'évalue à t , le corps de la fonction,
avec le paramètre formel x remplacé par l'argument s .*

Évaluation

Extension de la β -réduction en contexte :

$$\begin{array}{ll} (\lambda x.t) s \rightsquigarrow_{\beta} t[s/x], & (\beta\text{-réduction}) \\ t s \rightsquigarrow_{\beta} t' s, & \text{si } t \rightsquigarrow_{\beta} t', \\ t s \rightsquigarrow_{\beta} t s', & \text{si } s \rightsquigarrow_{\beta} s', \\ \lambda x.t \rightsquigarrow_{\beta} \lambda x.t', & \text{si } t \rightsquigarrow_{\beta} t'. \end{array}$$

Definition

Un **valeur** est un terme t tel que, pour tout terme s , on a $t \not\rightsquigarrow_{\beta} s$.

Exemple. Une variable x est un valeur.

Évaluation

Extension de la β -réduction en contexte :

$$\begin{array}{ll} (\lambda x.t) s \rightsquigarrow_{\beta} t[s/x], & (\beta\text{-réduction}) \\ t s \rightsquigarrow_{\beta} t' s, & \text{si } t \rightsquigarrow_{\beta} t', \\ t s \rightsquigarrow_{\beta} t s', & \text{si } s \rightsquigarrow_{\beta} s', \\ \lambda x.t \rightsquigarrow_{\beta} \lambda x.t', & \text{si } t \rightsquigarrow_{\beta} t'. \end{array}$$

Definition

Un **valeur** est un terme t tel que, pour tout terme s , on a $t \not\rightsquigarrow_{\beta} s$.

Exemple. Une variable x est un valeur.

Notation

On pose :

$$t \overset{n}{\rightsquigarrow}_{\beta} s,$$

s'il existe une suite

$$t = t_0 \rightsquigarrow_{\beta} t_1 \rightsquigarrow_{\beta} \dots t_{n-1} \rightsquigarrow_{\beta} t_n = s.$$

On pose :

$$t \overset{*}{\rightsquigarrow}_{\beta} s,$$

s'il existe $n \geq 0$ tel que $t \overset{n}{\rightsquigarrow}_{\beta} s$.

Notation

On pose :

$$t \overset{n}{\rightsquigarrow}_{\beta} s,$$

s'il existe une suite

$$t = t_0 \rightsquigarrow_{\beta} t_1 \rightsquigarrow_{\beta} \dots t_{n-1} \rightsquigarrow_{\beta} t_n = s.$$

On pose :

$$t \overset{*}{\rightsquigarrow}_{\beta} s,$$

s'il existe $n \geq 0$ tel que $t \overset{n}{\rightsquigarrow}_{\beta} s$.

Exemple : codage des nombres

$$\hat{0} = \lambda f. \lambda x. x$$

$$\hat{1} = \lambda f. \lambda x. fx$$

$$\hat{2} = \lambda f. \lambda x. f(fx)$$

$$\hat{3} = \dots$$

$$SUCC = \lambda n. \lambda f. \lambda x. f(nfx)$$

$$PLUS = \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx)$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fxfx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Examples

$$SUCC \hat{i} = \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx)$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x)$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2},$$

$$PLUS \hat{i} \hat{i} = \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx$$

$$\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx$$

$$\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx)$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx)$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}.$$

Exemples

$$SUCC \hat{i} = \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx)$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x)$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2},$$

$$PLUS \hat{i} \hat{i} = \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx$$

$$\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx$$

$$\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx)$$

$$\overset{2}{\rightsquigarrow}_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx)$$

$$\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}.$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{i} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{i} \hat{i} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta}^2 \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta}^2 \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta}^2 \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta}^2 \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Exemples

$$\begin{aligned} \text{SUCC } \hat{1} &= \lambda n. \lambda f. \lambda x. f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda f. \lambda x. fx)fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f((\lambda x. fx)x) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}, \end{aligned}$$

$$\begin{aligned} \text{PLUS } \hat{1} \hat{1} &= \lambda m. \lambda n. \lambda f. \lambda x. mf(nfx) \quad \lambda f. \lambda x. fx \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda f. \lambda x. fx) f(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda n. \lambda f. \lambda x. (\lambda x. fx)(nfx) \quad \lambda f. \lambda x. fx \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. (\lambda x. fx)(\lambda f. \lambda x. fx fx) \\ &\rightsquigarrow_{\beta}^2 \lambda f. \lambda x. (\lambda x. fx)(fx) \\ &\rightsquigarrow_{\beta} \lambda f. \lambda x. f(fx) = \hat{2}. \end{aligned}$$

Des théorèmes

Theorem

Une fonction

$$f : \mathbb{N} \longrightarrow \mathbb{N}$$

est calculable par une machine de Turing si et seulement si il existe un λ -terme t tel que

$$t \hat{n} \overset{*}{\rightsquigarrow}_{\beta} \hat{m}, \quad \text{pour tous entiers } n \text{ et } m \text{ tels que } f(n) = m.$$

Plan

Le λ -calcul

Confluence

Stratégies d'évaluations

L'évaluation (β -réduction en contexte) n'est pas déterministe.

Par exemple, mais :

$$\begin{array}{ccc} & & (\lambda y. wy)z \\ & \rightsquigarrow \beta & \\ (\lambda x. (\lambda y. xy)z)w & & \\ & \rightsquigarrow \beta & \\ & & (\lambda x. xz)w \end{array} \quad \begin{array}{ccc} & & wZ \\ & \rightsquigarrow \beta & \\ & & \end{array}$$

L'évaluation (β -réduction en contexte) n'est pas déterministe.

Par exemple, mais :

$$\begin{array}{ccc} & & (\lambda y. wy)z \\ & \rightsquigarrow \beta & \\ (\lambda x. (\lambda y. xy)z)w & & \\ & \rightsquigarrow \beta & \\ & & (\lambda x. xz)w \end{array} \quad \begin{array}{ccc} & & wZ \\ & \rightsquigarrow \beta & \\ & & \end{array}$$

L'évaluation (β -réduction en contexte) n'est pas déterministe.

Par exemple, mais :

$$\begin{array}{ccc} & & (\lambda y. wy)z \\ & \rightsquigarrow_{\beta} & \\ (\lambda x. (\lambda y. xy)z)w & & \\ & \rightsquigarrow_{\beta} & \\ & & (\lambda x. xz)w \end{array} \quad \begin{array}{ccc} & & wZ \\ & \rightsquigarrow_{\beta} & \\ & & wZ \end{array}$$

Confluence du λ -calcul

Theorem

Soient t_0, t_1, t_2 trois termes tels que $t_0 \xrightarrow{*}_{\beta} t_1$ et $t_0 \xrightarrow{*}_{\beta} t_2$.

Il existe alors un terme t_3 tel que $t_1 \xrightarrow{\beta} t_3$ et $t_2 \xrightarrow{\beta} t_3$.



Corollary

Soient t_0, t_1, t_2 trois termes tels que $t_0 \xrightarrow{*}_{\beta} t_1$ et $t_0 \xrightarrow{*}_{\beta} t_2$.

Si t_1 et t_2 sont des valeurs, alors $t_1 = t_2$.

Confluence du λ -calcul

Theorem

Soient t_0, t_1, t_2 trois termes tels que $t_0 \xrightarrow{*}_{\beta} t_1$ et $t_0 \xrightarrow{*}_{\beta} t_2$.
Il existe alors un terme t_3 tel que $t_1 \rightsquigarrow_{\beta} t_3$ et $t_2 \rightsquigarrow_{\beta} t_3$.



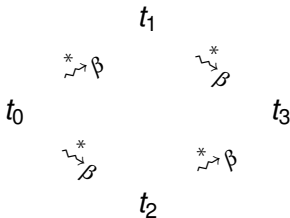
Corollary

Soient t_0, t_1, t_2 trois termes tels que $t_0 \xrightarrow{*}_{\beta} t_1$ et $t_0 \xrightarrow{*}_{\beta} t_2$.
Si t_1 et t_2 sont des valeurs, alors $t_1 = t_2$.

Confluence du λ -calcul

Theorem

Soient t_0, t_1, t_2 trois termes tels que $t_0 \xrightarrow{*}_{\beta} t_1$ et $t_0 \xrightarrow{*}_{\beta} t_2$.
Il existe alors un terme t_3 tel que $t_1 \xrightarrow{*}_{\beta} t_3$ et $t_2 \xrightarrow{*}_{\beta} t_3$.



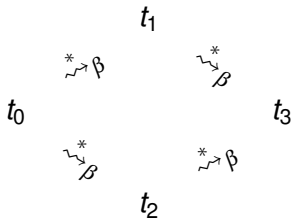
Corollary

Soient t_0, t_1, t_2 trois termes tels que $t_0 \xrightarrow{*}_{\beta} t_1$ et $t_0 \xrightarrow{*}_{\beta} t_2$.
Si t_1 et t_2 sont des valeurs, alors $t_1 = t_2$.

Confluence du λ -calcul

Theorem

Soient t_0, t_1, t_2 trois termes tels que $t_0 \xrightarrow{*}_{\beta} t_1$ et $t_0 \xrightarrow{*}_{\beta} t_2$.
Il existe alors un terme t_3 tel que $t_1 \xrightarrow{*}_{\beta} t_3$ et $t_2 \xrightarrow{*}_{\beta} t_3$.



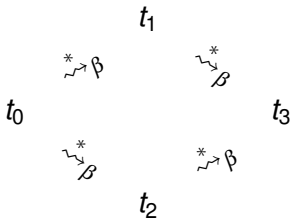
Corollary

Soient t_0, t_1, t_2 trois termes tels que $t_0 \xrightarrow{*}_{\beta} t_1$ et $t_0 \xrightarrow{*}_{\beta} t_2$.
Si t_1 et t_2 sont des valeurs, alors $t_1 = t_2$.

Confluence du λ -calcul

Theorem

Soient t_0, t_1, t_2 trois termes tels que $t_0 \xrightarrow{*}_{\beta} t_1$ et $t_0 \xrightarrow{*}_{\beta} t_2$.
Il existe alors un terme t_3 tel que $t_1 \xrightarrow{*}_{\beta} t_3$ et $t_2 \xrightarrow{*}_{\beta} t_3$.



Corollary

Soient t_0, t_1, t_2 trois termes tels que $t_0 \xrightarrow{*}_{\beta} t_1$ et $t_0 \xrightarrow{*}_{\beta} t_2$.
Si t_1 et t_2 sont des valeurs, alors $t_1 = t_2$.

Un terme un peu bizarre

Soit

$$\omega := (\lambda x.xx)(\lambda x.xx)$$

Il n'existe aucune valeur t tel que $\omega \rightsquigarrow_{\beta}^* t$.

Plan

Le λ -calcul

Confluence

Stratégies d'évaluations

Comment rendre l'évaluation déterministe ?

Definition

Soit t un λ -terme. Un **redex** de t est un sous-terme t' de t de la forme

$$t' = (\lambda x.s) r.$$

Réduire un redex : substituer un tel t' par $s[r/x]$ dans t .

Par exemple, dans

$$t = (\lambda x.(\lambda y.xy)z) w$$

nous avons deux redexes :

• $(\lambda x.(\lambda y.xy)z)$

Comment rendre l'évaluation déterministe ?

Definition

Soit t un λ -terme. Un **redex** de t est un sous-terme t' de t de la forme

$$t' = (\lambda x.s) r.$$

Réduire un redex : substituer un tel t' par $s[r/x]$ dans t .

Par exemple, dans

$$t = (\lambda x.(\lambda y.xy)z) w$$

nous avons deux redexes :

1. le terme t lui-même :

$$(\lambda x.(\lambda y.xy)z) w$$

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

Comment rendre l'évaluation déterministe ?

Definition

Soit t un λ -terme. Un **redex** de t est un sous-terme t' de t de la forme

$$t' = (\lambda x.s) r.$$

Réduire un redex : substituer un tel t' par $s[r/x]$ dans t .

Par exemple, dans

$$t = (\lambda x.(\lambda y.xy)z) w$$

nous avons deux redexes :

1. le terme t lui-même :

$$(\lambda x.(\lambda y.xy)z) w$$

2. le sous-terme $f = (\lambda y.xy)z$:

$$(\lambda y.xy)z.$$

Comment rendre l'évaluation déterministe ?

Definition

Soit t un λ -terme. Un **redex** de t est un sous-terme t' de t de la forme

$$t' = (\lambda x.s) r.$$

Réduire un redex : substituer un tel t' par $s[r/x]$ dans t .

Par exemple, dans

$$t = (\lambda x.(\lambda y.xy)z) w$$

nous avons deux redexes :

1. le terme t lui-même :

$$(\lambda x.(\lambda y.xy)z) w$$

2. le sous-terme $t' = (\lambda y.xy)z$:

$$(\lambda y.xy) z.$$

Comment rendre l'évaluation déterministe ?

Definition

Soit t un λ -terme. Un **redex** de t est un sous-terme t' de t de la forme

$$t' = (\lambda x.s) r.$$

Réduire un redex : substituer un tel t' par $s[r/x]$ dans t .

Par exemple, dans

$$t = (\lambda x.(\lambda y.xy)z) w$$

nous avons deux redexes :

1. le terme t lui-même :

$$(\lambda x.(\lambda y.xy)z) w$$

2. le sous-terme $t' = (\lambda y.xy)z$:

$$(\lambda y.xy) z.$$

Dans un même terme, deux redexes peuvent être :

1. l'un plus extérieur (resp. plus intérieur) que l'autre.

$(\lambda x. (\lambda y. xy)z) w$ est plus extérieur que $(\lambda y. xy)z$;

2. l'un plus à gauche que l'autre. Dans

$(\lambda g. \lambda x. gx) h ((\lambda y. fy) z),$

$(\lambda g. \lambda x. gx) h$ est plus à gauche que $(\lambda y. fy) z$.

Les stratégies d'évaluation

Ordre applicatif (évaluation par l'intérieur) : le redex le plus à l'intérieur, le plus à gauche est réduit.

Ordre normal (évaluation par l'extérieur) : le redex le plus à l'extérieur, le plus à gauche est réduit.

Exemple :

$(\lambda x.(\lambda y.xy)z) w \rightsquigarrow_{\beta} \dots$

Les stratégies d'évaluation

Ordre applicatif (évaluation par l'intérieur) : le redex le plus à l'intérieur, le plus à gauche est réduit.

Ordre normal (évaluation par l'extérieur) : le redex le plus à l'extérieur, le plus à gauche est réduit.

Exemple :

$$(\lambda x.(\lambda y.xy)z) w \rightsquigarrow_{\beta} \dots$$

Call by name : comme pour l'ordre normal, mais pas de réduction sous une abstraction.

Call by value : Seulement les redexes extérieurs sont réduits, une fois que l'argument a été réduit à un valeur. C'est-à-dire : comme pour l'ordre applicatif, mais pas de réduction sous une abstraction.

Call by need : comme pour l'ordre normal, mais en utilisant le partage.