Alexandre DIKOVSKY
Université de Nantes

Cours 2. Calculabilité et Lambda définissabilité des fonctions

1 Théorie λ

λ :
- (β) $(\lambda x.\ M)N = M[x := N]$
- (e_1) $M = M$
- (e_2) $M = N \to N = M$
- (e_3) $M = N \ \wedge\ N = L \to M = L$
- (c_1) $M = N \to MZ = NZ$
- (c_2) $M = N \to ZM = ZN$
- (c_3) $M = N \to \lambda x.\ M = \lambda x.\ N$
- (α) $\lambda x.\ M = \lambda y.\ M[x := y]$

Notation :
(i) Λ dénote l'ensemble des Lambda-termes. $M \equiv N$ veut dire : M et N sont *isomorphes* (coïncident) un renommage des variables liées près). C'est une équivalence sur Λ.
(ii) Soit R une relation binaire sur Λ (e.g. $=$ de λ).
$\to_R$ dénote la relation minimale telle que :
1) $(M, N) \in R \to M{\to_R}N$;
2) $(M, N) \in R \to ZM{\to_R}ZN$;
3) $(M, N) \in R \to MZ{\to_R}NZ$;
4) $(M, N) \in R \to \lambda x.\ M{\to_R}\lambda x.\ N$
(nous allons utiliser $\to_\beta$ à la place de $\to_=$ quand il s'agit de la relation $=$ de λ). On dit que la relation $\to_R$ est *conforme avec* λ.
$\twoheadrightarrow_R$ dénote la relation minimale telle que :
5) $M{\to_R}N \to M{\twoheadrightarrow_R}N$;
6) $M \equiv N \to M{\twoheadrightarrow_R}N$;
7) $M_1{\twoheadrightarrow_R}M_2 \ \wedge\ M_2{\twoheadrightarrow_R}M_3 \to M_1{\twoheadrightarrow_R}M_3$.
On dit que la relation $\twoheadrightarrow_R$ est *R-réduction* (*R-conversion*) des Lambda-termes. Donc, pour $=$ de λ c'est *β-conversion.*
$M\overset{*}{\leftrightarrow}_R N$ dénote la relation minimale (dite *R-équivalence*) telle que :
8) $M{\twoheadrightarrow_R}N \to M\overset{*}{\leftrightarrow}_R N$;
9) $M\overset{*}{\leftrightarrow}_R N \to N\overset{*}{\leftrightarrow}_R M$;
10) $M_1\overset{*}{\leftrightarrow}_R M_2 \ \wedge\ M_2\overset{*}{\leftrightarrow}_R M_3 \to M_1\overset{*}{\leftrightarrow}_R M_3$.

Exemple 1. *Reprenons l'exemple du cours 1 :*
$\mathbf{SKK}x \equiv (\lambda xyz.\ xz(yz))(\lambda xy.\ x)(\lambda xy.\ x)x \twoheadrightarrow_\beta (\lambda x_1yz.\ x_1z(yz))(\lambda x_2y_1.\ x_2)(\lambda x_3y_2.\ x_3)x \twoheadrightarrow_\beta$
$(\lambda yz.\ (\lambda x_2y_1.\ x_2)z(yz))(\lambda x_3y_2.\ x_3)x \twoheadrightarrow_\beta (\lambda z.\ (\lambda x_2y_1.\ x_2)z((\lambda x_3y_2.\ x_3)z))x \twoheadrightarrow_\beta$
$(\lambda z.\ (\lambda x_2y_1.\ x_2)z(\lambda y_2.\ z))x \twoheadrightarrow_\beta (\lambda x_2y_1.\ x_2)x(\lambda y_2.\ x) \twoheadrightarrow_\beta (\lambda x_2y_1.\ x_2)x(\lambda y_2.\ x) \twoheadrightarrow_\beta$

$(\lambda y_1.\, x)(\lambda y_2.\, x) \twoheadrightarrow_\beta x$

En utilisant la relation de β-réduction on peut définir le résultat de calcul d'un Lambda-terme.

Définition 1. *Un Lambda-terme M est* normal *s'il ne contient pas de rédex (c'est-à-dire, n'est pas β-réductible). Un Lambda-terme M est* normalisable *s'il existe un terme normal N auquel il se réduit : $M \twoheadrightarrow_\beta N$. N est valeur de M.*

Ainsi nous obtenons une sémantique opérationnele des Lambda-termes. Selon cette sémantique tout Lambda-terme définit un procesus. Seulement, pour l'utiliser, il faut prouver sa correction. En particulier, il faut prouver la fonctionnalité de cette sémantique.

En titre d'exemple, les termes $x, \mathbf{S}, \mathbf{K}, \mathbf{I}$ sont normaux, les termes $\mathbf{SKK}x$ et $\Omega \equiv (\lambda x.\, xx)(\lambda x.\, xx)$ ne le sont pas. Le théorème suivant central nous servira de base pour la sémantique de λ-calcul.

Théorème 1. *[Church, Rosser] Pour tout Lambda-terme M, s'il existent des termes normaux N_1, N_2 tels que $M \twoheadrightarrow_\beta N_1$ et $M \twoheadrightarrow_\beta N_2$, alors $N_1 \equiv N_2$.*

Ce théorème dit, que tout Lambda-terme M soit n'est pas normalisable, soit est réductible à un Lambda-terme normal unique N un isomorphisme près. Ainsi, dans ce cas la classe de l'isomorphisme de N est la ≪valeur≫ de M.

Par exemple, $\mathbf{SKK}x \twoheadrightarrow_\beta x$ montre que $\mathbf{SKK}x =_\beta x$. D'un autre côté, le terme Ω n'a pas de valeur, car il n'est réductible qu'à soi-même (un isomorphisme près). Selon le théorème 1 le processus défini par un Lambda-terme est fonctionnel (calcule une fonction sur Λ). En même temps, ce calcul est non-déterministe. Par exemple, le Lambda-terme $M \equiv \mathbf{K}x\Omega$ a un ensemble infini de calculs de sa valeur x : $M \twoheadrightarrow_\beta M \twoheadrightarrow_\beta ... \twoheadrightarrow_\beta x$.

Avant de discuter le théorème 1 reprenons les questions fondementales mais cette fois par rapport à la théorie λ.

QUESTIONS FONDAMENTALES.

Q_1 : La théorie λ , est-elle cohérente ?

Définition 2. λ *est* **cohérente** *s'il existe au moins une égalité $M_0 = N_0$ non prouvable à partir de λ :*

$$\lambda \not\models M_0 = N_0.$$

Parfois à la place de $\lambda \models M = N$ nous allons écrire $M =_\lambda N$.

Attention au traitement correct des variables !

Exemple 2. *Sophisme "λ est non-cohérente".*
Posons $F \equiv (\lambda xy.yx)$. Soit deux termes M, N. Alors,

$$FMN \equiv (\lambda xy.yx)MN.$$

En particulier, si $M \equiv y, N \equiv x$, alors
(1) $\quad Fyx \equiv (\lambda xy.\ yx)yx \twoheadrightarrow_\beta xy$
(2) $\quad Fyx \equiv (\lambda x.\ (\lambda y.\ yx))yx \twoheadrightarrow_\beta (\lambda y.\ yy)x \twoheadrightarrow_\beta xx$
Ainsi,
(3) $\quad (1), (2), (e_3) \models xy = xx.$

La proposition auxiliaire suivante à été pouvée dans le cours 1 :

Lemme 1. *Il existe des Lambda-termes* **cons**, π^1, π^2 *tels que*
(4) $\quad \pi^1(\mathbf{cons}\ UV) = U$ *et*
(5) $\quad \pi^2(\mathbf{cons}\ UV) = V$
pour tous Lambda-termes U, V.

Selon ce lemme et les axiomes (c_2),(e_3) (3) l'égalité
(6) $\quad (c_2), (e_3), (3) \models \pi^2(\mathbf{cons}\ xy) = \pi^2(\mathbf{cons}\ xx)$ et donc
(7) $\quad \lambda \models y = x.$
Ensuite,
(8) $\quad \lambda \models \lambda y.\ y = \lambda y.\ x$ (selon l'axiome (c_3))
On en déduit :
(9) $\quad M = \lambda(\lambda y.\ y)M =_\lambda (\lambda y.\ x)M =_\lambda x$ (par (c_1))
(10) $\quad N =_\lambda (\lambda y.\ y)N =_\lambda (\lambda y.\ x)N =_\lambda x$ (par (c_1))
Enfin,
(11) $\quad M =_\lambda N$ (par (e_3)).
Où est l'erreur ?

Dévoilement du sophisme : dans le point (2) on fait

$$(\lambda y.\ yx)[x := y]$$

Pourtant, x n'est pas libre pour la substitution $x := y$ dans le terme $(\lambda y.\ yx)$.
Morale : Renommez les variables liées avant de faire les $\beta-$réductions.

Convention de renommage : Les variables liées sont deux-à-deux différentes et sont distinctes des variables libres.

Q_2 : Le thérème de Church-Rosser, est-il vrai ? C'est-à-dire, pour

$$M \twoheadrightarrow_\beta N_1, \ M \twoheadrightarrow_\beta N_2$$

où N_1, N_2 sont en FN, est-il possible que

$$N_1 \not\equiv N_2?$$

2 Substitution et contextes

Propriétés de substitution (la convention de renommage étant respectée).

Lemme 2. *Si les termes $N_1, ..., N_n$ n'ont pas d'occurrence de $x_1, ..., x_n$, alors*

$$\lambda \models (\lambda x_1 ... x_n.\ M)N_1 ... N_n = M[x_1 := N_1]...[x_n := N_n].$$

Preuve : Par l'induction sur la syntaxe de M.

Corollaire 1. *Dans les mêmes conditions,*

(1) $M_1 =_\lambda M_2$ *implique* $M_1[x := N] =_\lambda M_2[x := N]$,

(2) $N_1 =_\lambda N_2$ *implique* $M[x := N_1] =_\lambda M[x := N_2]$

(3) $M_1 =_\lambda M_2$ *et* $N_1 = N_2$ *implique* $M_1[x := N_1] =_\lambda M_2[x := N_2]$.

On peut utiliser le lemme 2 et le corollaire 1 pour en déduire des égalités de λ. Certes, toute implication

$$E_1 =_\lambda E_2 \models E_1' =_\lambda E_2'$$

ainsi prouvée peut aussi être prouvée directement dans λ (c.-à-d., à l'aide des axiomes de λ).

Question : Est-ce que l'inverse est vrai aussi ?

Réponce : Non

Exemple 3. *Même s'il y a des occurrences libres de x dans N_1, N_2, la dérivation suivante est possible dans la théorie λ :*

$$\dfrac{\dfrac{\dfrac{(i)\quad N_1 =_\lambda N_2 \quad (C_3)}{(ii)\quad \lambda y.\ N_1 =_\lambda \lambda y.\ N_2 \quad (C_2)}}{(iii)\quad x\lambda y.\ N_1 =_\lambda x\lambda y.\ N_2 \quad (C_3)}}{(iv)\quad \lambda x.\ (x\lambda y.\ N_1) =_\lambda \lambda x.\ (x\lambda y.\ N_2)}$$

En même temps, on ne peut pas prouvér l'implication $(i) \models (iv)$ uniquement par le lemme de substitution parce que dans le cas où N_1 contient x libre, le terme $\lambda x.\ (x\lambda y.\ N_1)$ ne peut pas être représenté en forme

$$(\lambda x.\ (x(\lambda y.\ z))[z := N_1].$$

Nous voyons que ce sont les axiomes de congruence qui nous manquent. Pour les simuler, on utilise la notion suivante de **contexte**.

Définition 3.

(1) $x \in Var$ *et* [ ] *(un trou) sont* contextes .

(2) *Si* $C_1[\ ]$ *et* $C_2[\ ]$ *sont contextes, alors*
$C_1[\ ]C_2[\ ]$ *et* $\lambda x.\ C_1[\ ]$ *le sont aussi .*

(3) *Si* $C_1[\ ]$ *est un contexte et* M *est un terme, alors*
$C[M] =_{df} C[\ [\ ] := M]$ *(M remplace toute occurrence de [ ] dans C).*

Attention : une variable libre de M peut devenir liée dans $C[M]$!

Exemple 4. *Soit* $M \equiv xy$, $M_1 \equiv (\lambda z.\ z)xy$, *et* $C[\ ] \equiv \lambda x.\ x(\lambda y.\ [\ ])$. *Alors* $M =_\lambda M_1$, *ce qui donne* $C[M] =_\lambda C[M_1]$.

Théorème 2. *(De substitution dans les contextes)*
Si $M =_\lambda N$, alors $C[M] =_\lambda C[N]$ pour tout contexte $C[\ ]$.

Preuve : Par récurrence sur C et en utilisant les axiomes de congruence.

Remarque : La β-conversion n'est pas correcte dans les contextes qui ne sont pas libres pour le terme substitué (cf. le p.(2) du sophisme).

Notation et terminologie :

- $FV(N)$: l'ensemble des variables libres dans N.
- N est **combinateur**, si $FV(N) = 0$.
- Λ^0 est l'ensemble des combinateurs.
- $M(\overline{x})$: $\overline{x}$ sont les variables libres de M ;
  Pour $M(\overline{x})$, et $\overline{N} = N_1...N_k$, où $\overline{x} = x_1...x_k$,
  $M(\overline{N}) =_{df} M[x_1 := N_1]...[x_k := N_k]$ (simultainement !)
- $\overline{x}$ et $\bar{N}$ sont **compatibles**, si $|\ \overline{x}\ | = |\ \overline{N}\ |$ et $x_1, ..., x_k \notin FV(\overline{N})$.

Ex : $\overline{x} = x$ et $\overline{N} = \lambda y.\ xy$ ne sont pas compatibles).

Réformulation de la β-conversion : si $\overline{x}$ est compatible avec $\overline{N}$, alors

$$(\lambda\overline{x}.\ M)\overline{N} =_\lambda M(\overline{N}) \equiv M[\overline{x} := \overline{N}].$$

Formellement, $\overline{x}$ n'est pas compatible avec $\overline{x}$, et pourtant :

Lemme 3. $(\lambda\overline{x}.\ M)\overline{x} =_\lambda M$.

Exemple 5. $(\lambda x.\ M)x =_\lambda M[x := x] \equiv M$
Pour $\overline{x} = x_1, x_2$:

$$\begin{aligned}(\lambda\overline{x}.\ M)\overline{x} &\equiv ((\lambda x_1.\ (\lambda x_2.\ M))x_1)x_2 =_\lambda ((\lambda x_2.\ M)[x_1 := x_1])x_2\\ &=_\lambda (\lambda x_2.\ M)x_2 \quad \textit{(voir le cas précédant)}\\ &=_\lambda M.\end{aligned}$$

Le cas général prouvé par récurrence.

Corollaire 2. *(Complétude des combinateurs). Soit $M \equiv M(\overline{x})$. Alors, si on "compile" M en le combinateur $F \equiv \lambda\overline{x}.\ M$, on obtient :*
(1) $F\overline{x} =_\lambda M \equiv M(\overline{x})$;
(2) $F\overline{N} =_\lambda M(\overline{N})$ *pour tout* $\overline{N}$ *compatible avec* $\overline{x}$.

De plus, Schonfinkel a défini une compilation simple des Lambda-termes en combinateurs composés de **S** et **K**.

Lemme 4. *(i)* $\lambda x.\ x \equiv \mathbf{I} =_\lambda \mathbf{SKK}$.
(ii) $\lambda x.\ M =_\lambda \mathbf{K}M$, *si* $x \notin FV(M)$.
(iii) $\lambda x.\ MN =_\lambda \mathbf{S}(\lambda x.\ M)(\lambda x.\ N)$.

Preuve : (i) $\mathbf{SKK} \equiv (\lambda x_1x_2x_3.\ x_1x_3(x_2x_3))(\lambda x_4x_5.\ x_4)(\lambda x_4x_5.\ x_4)$
$\twoheadrightarrow_\beta \lambda x_3.\ (\lambda x_4x_5.\ x_4)x_3((\lambda x_4x_5.\ x_4)x_3)$
$\twoheadrightarrow_\beta \lambda x_3.\ x_3 \equiv \mathbf{I}$
(ii) $\mathbf{K}M \equiv (\lambda x_1x_2.\ x_1)M \twoheadrightarrow_\beta \lambda x_2.\ M$, si $x_2 \notin FV(M)$.
(iii) $\mathbf{S}(\lambda x.\ M)(\lambda x.\ M) \equiv (\lambda x_1x_2x_3.\ x_1x_3(x_2x_3))(\lambda x.\ M)(\lambda x.\ N) \twoheadrightarrow_\beta$

$$\lambda x_3.\ (\lambda x.\ M)x_3((\lambda x.\ N)x_3 \twoheadrightarrow_\beta \lambda x_3.\ M[x := x_3]N[x := x_3] \equiv$$
$$\lambda x_3.\ MN[x := x_3] \equiv \lambda x_3.\ MN$$

Ainsi on peut éliminer tous λ.

Exemple 6. *Compilation de $\lambda xy.yx$:*

$$\lambda xy.\ yx \equiv \lambda x.\ (\lambda y.\ yx) =_\lambda \lambda x.\ \mathbf{S}(\lambda y.\ y)(\lambda y.\ x) \equiv \lambda x.\ \mathbf{SI}(\lambda y.\ x) =_\lambda \lambda x.\ \mathbf{SI}(\mathbf{K}x) =_\lambda$$
$$\mathbf{S}(\lambda x.\ \mathbf{SI})(\lambda x.\ \mathbf{K}x) =_\lambda \mathbf{S}(\mathbf{K}(\mathbf{SI}))(\lambda x.\ \mathbf{K}x) =_\lambda \mathbf{S}(\mathbf{K}(\mathbf{S}(\mathbf{SKK})))(\lambda x.\ \mathbf{K}x) =_\lambda$$
$$\mathbf{S}(\mathbf{K}(\mathbf{S}(\mathbf{SKK})))(\mathbf{S}(\lambda x.\ \mathbf{K})(\lambda x.\ x)) =_\lambda \mathbf{S}(\mathbf{K}(\mathbf{S}(\mathbf{SKK})))(\mathbf{S}(\mathbf{KK})\mathbf{I}) =_\lambda$$
$$\mathbf{S}(\mathbf{K}(\mathbf{S}(\mathbf{SKK})))(\mathbf{S}(\mathbf{KK})(\mathbf{SKK})).$$

Morale : Dans la théorie λ, tout Lambda-terme T peut être transformé en un terme de la forme $C\overline{x}$, où C est un combinateur composé de **S** et **K**.

3 Calculs des fonctions arithmétiques

Supposons que la théorie λ soit fondée. Comment l'utiliser pour calculer les fonctions arithmétiques ? Il faut choisir un code des nombres par les Lambda-termes. Voici deux codes proposés par Church et par Barendregt :

Code de Church :

$\mathbf{0} =_{df} \lambda fx.\ x \equiv \mathbf{F}$

$\mathbf{1} =_{df} \lambda fx.\ fx$

$\mathbf{n+1} =_{df} \lambda fx.\ f(..fx\underbrace{)..)}_{n}$

Code de Barendregt :

$\lceil 0 \rceil =_{df} \mathbf{I} \equiv \lambda x.\ x$

$\lceil n+1 \rceil =_{df} \lambda z.\ z\mathbf{F}\lceil n \rceil$ $(\mathbf{F} \equiv \lambda fx.\ x)$.

Définition 4. *Une fonction totale f à k arguments sur N est λ-définissable s'il existe un Lambda-terme M_f tel que :*

$$\forall n_1, ..., n_k \in N\ (M_f\ \mathbf{code}(n_1)...\ \mathbf{code}(n_k) \twoheadrightarrow_\beta \mathbf{code}(f(n_1, ..., n_k)))$$

Commentaire : cela veut dire que le terme $\mathbf{code}(f(n_1, ..., n_k))$ est en FN et qu'il est **unique**.

Exemples des calculs.

A. Fonctions booléennes. $\mathbf{K} \equiv \lambda xy.\ x =_{df} \ll$vrai$\gg =_{df} \mathbf{T}$

$\mathbf{F} \equiv \lambda xy.\ y =_{df} \ll$faux$\gg$

A1. Conjonction : and $\equiv \lambda xy.\ xy\mathbf{F}$

A2. Disjonction : or $\equiv \lambda xy.\ x\mathbf{T}y$

A3. Négation : not $\equiv \lambda bxy.\ bxy$

A4. Implication : imp $\equiv$?

A5. sas : (si...alors...sinon...)

sas $\equiv \lambda bxy.\ bxy$

A6. sa : (si...alors...)

sa $\equiv \lambda bx.\ bx\mathbf{I}$

B. Paires :

$\mathbf{cons} \equiv \lambda xyz.\ zxy$
$\pi_1 \equiv \lambda p.\ p\mathbf{T}$
$\pi_2 \equiv \lambda p.\ p\mathbf{F}$

C. Prouvez pour les nombres de Barendregt :

Lemme 5. $\lceil n+1 \rceil \equiv \lambda z.\ z\mathbf{F}\lceil n \rceil = \mathbf{consF}\lceil n \rceil$

Ainsi $\lceil n+1 \rceil = \mathbf{consF}\lceil n \rceil$, c.à.d.,
$\mathbf{succ}_b \equiv \lambda n.\ \mathbf{consF}\lceil n \rceil$,
$\mathbf{zero}_b \equiv \lambda n.\ n\mathbf{T}$
$I_n^k \equiv \lambda x_1 ... x_n.\ x_k$
$\mathbf{pred}_b \equiv \lambda n.\ sas(\mathbf{zero}\ n)\mathbf{I}(\pi_2 n)$

D. Pour les nombres de Church :

$\mathbf{succ}_c \equiv \lambda nfx.\ f(nfx)$
$\mathbf{zero}_c \equiv \lambda n.\ \mathbf{sas}\ n(\lambda xyz.\ z)\mathbf{T}$.

E. Points fixes

Théorème 3. $[Kleene, Church]$
(i) Pour tout terme M, il existe un terme X tel que

$$\lambda \models MX = X.$$

(ii) Il existe un terme $\mathbf{Y}$ (combinateur du point fixe) tel que pour tout terme M :

$$\lambda \models M(\mathbf{Y}M) = \mathbf{Y}M$$

et

$$\mathbf{Y} \equiv \lambda f.\ (\lambda x.\ f(xx))(\lambda x.\ f(xx)).$$

Preuve. $(ii) \Longrightarrow (i)$. On prouve (ii) :
Soit M. Alors $\mathbf{Y}M \twoheadrightarrow_\beta (\lambda x.\ M(xx))(\lambda x.\ M(xx))$.
Notons $W \equiv \lambda x.\ M(xx)$. Alors,

$$\mathbf{Y}M \equiv WW \equiv \lambda x.\ M(xx)W \twoheadrightarrow_\beta M(WW) \equiv M(\mathbf{Y}M).$$

Corollaire 3. *Soit un contexte $C[f][x]$ (deux occurrences sélectonnées). Alors, il existe un terme M_c tel que*

$$M_c X = C[M_c][X].$$

pour tout terme X.

Exemple 7. *Soit*

$$Fact_b \equiv \lambda fx.\ \mathbf{sas}(\mathbf{zero}_b\ x)\lceil 1 \rceil(\mathbf{mult}_b\ x(f(\mathbf{prec}_b\ x))).$$

Alors $fact_b \equiv \mathbf{Y}Fact_b$ *est la factorielle :* $\left\{ \begin{array}{rcl} fact_b\lceil 0 \rceil & = & \lceil 1 \rceil \\ fact_b\lceil 1 \rceil & = & \lceil 1 \rceil \\ fact_b\lceil n+1 \rceil & = & \mathbf{mult}_b\ n(\mathbf{fact}_b\ n) \end{array} \right.$

Exemple 8. *Trouvez les Lambda-termes qui calculent* $\mathbf{add}_b$ *et* $\mathbf{mult}_b$.

Les théories des nombres naturels de Church et de Barendregt sont différentes. Dans la théorie de Barendregt les opérations de l'addition et de la multiplication sont exprimées par le moyen du PF, tandis que dans la théorie de Church on peut les exprimer sans récurrence :

Lemme 6. [*Rosser*] *Soient* $A_+ \equiv \lambda xypq.\ xp(ypq)$
$A_* \equiv \lambda xyz.\ x(yz)$
$A_{exp} \equiv \lambda xy.\ yx$

Alors, $A_+\mathbf{nm} =_\lambda \mathbf{n}+\mathbf{m}$ *(pour tout* $n, m \in N$ *)*
$A_*\mathbf{nm} =_\lambda \mathbf{n}*\mathbf{m}$ *(pour tout* $n, m \in N$ *)*
$A_{exp}\mathbf{nm} =_\lambda \mathbf{n}^{\mathbf{m}}$ *(pour tout* $n \in N$ *et tout* $m > 0$*).*

Par ailleurs, le prédécesseur est défini par un terme complèxe :

$$\mathbf{pred_c} \equiv \lambda xyz.\ x(\lambda pq.\ q(py))(\mathbf{K}z)\mathbf{I}.$$

F. Expression des fonctions récursives
Choisissons le code de Barendregt $\lceil n \rceil$.
Nous avons exprimé par les Lambda-termes les trois fonctions de base : I_n^k, $\mathbf{succ}_b$, $\mathbf{zero}_b$.
Exemple 9. $\mathbf{zero}_b \equiv \lambda x.\ x\mathbf{T}$.
Lemme 7. *Si* f *est définie par la récursion primitive*
$$\begin{cases} f(0,\overline{y}) &= g(\overline{y}) \\ f(x+1,\overline{y}) &= h(x,\overline{y},f(x,\overline{y})) \end{cases}$$
à partir des fonctions :
g *exprimée par un terme* $\lambda\overline{y}.\ G$ *et*
h *exprimée par un terme* $\lambda x\overline{y}z.\ H$,
alors f *est exprimée par le terme* $\mathbf{Y}F$, *où*

$$F \equiv \lambda fx\overline{y}.\ \mathbf{sas}(\mathbf{zero}_b\ x)(G\overline{y})(H(\mathbf{prec}_b\ x)\overline{y}(f(\mathbf{prec}_b\ x)\overline{y})).$$

Preuve : Théorème de PF.

Lemme 8. *Soit une fonction* f *définie par l'opérateur de minimisation :*

$$f(\overline{y}) = \mu x(g(x,\overline{y}) = 0)$$

à partir de g *qui est exprimée par un terme* $\lambda\overline{y}.\ G$*. Alors il existe un terme qui exprime* f.
Prouvez-le.

Théorème 4. [*Church, Turing*] *Une fonction arithmétique est récursive partielle ssi elle est exprimée par un Lambda-terme tant pour le code de Church que pour le code de Barendregt.*

Définition 5. *(i) Deux Lambda-termes sont équivalents s'ils calculent la même fonction arithmétique (pour un même code).*
(ii) Une propriété Q *de termes est* **invariante** *si* $Q(M)$ *est vrai ssi* $Q(N)$ *l'est aussi pour tous termes équivalents* M, N.

Exemple 10. *Si M, N sont équivalents et il s'existe M_0 tel que*

$$M \twoheadrightarrow_\beta M_0, \quad M_0 \text{ en } FN,$$

alors il existe aussi N_0 tel que :

$$N \twoheadrightarrow_\beta N_0, \quad N_0 \text{ en } FN$$

pour le terme N.

C.-à-d., le problème de terminaison des calculs est invariant.

(iii) Une propriété Q de termes est **non-triviale** *s'il existent au moins deux termes M_1, M_2 tels que*

$$Q(M_1) \quad et \quad \neg Q(M_2).$$

Exemple 11. *Le problème de terminaison est non-trivial.*

Théorème 5. [*Rice*] *Si une propriété de Lambda-termes est invariante et non-triviale, alors elle est non-décidable (c.-à-d. il n'y a pas de fonction récursive qui la calcule).*

Corollaire 4. *Le problème de terminaison est non-décidable.*

Tout cela est raisonable à condition que la théorie λ est cohérente !