Lambda-calcul
et sémantique des langages de programmation

Thibaut Balabonski @ *UPSay*
https://www.lri.fr/~blsk/Semantique/
Hiver 2021

Chapitre 5 - λ-calculabilité

1 Données et opérations de base

Fonctions
Fonction identité

$$\mathsf{I} \equiv \lambda x.x$$

Composition de fonctions

$$g \circ f \equiv \lambda x.g(f(x))$$

Exemple

$$\begin{array}{rcl} \mathsf{I} \circ \mathsf{I} & \equiv & \lambda x.\mathsf{I}(\mathsf{I}x) \\ & \equiv & \lambda x.(\lambda y.y)\ ((\lambda z.z)\ x) \\ & \to_\beta & \lambda x.(\lambda z.z)\ x \\ & \to_\beta & \lambda x.x \end{array}$$

Booléens et conditionnelles
Valeurs booléennes

$$\begin{array}{rcl} \mathsf{T} & \equiv & \lambda xy.x \\ \mathsf{F} & \equiv & \lambda xy.y \end{array}$$

Expression conditionnelle

$$\mathsf{if}\ c\ \mathsf{then}\ a\ \mathsf{else}\ b \equiv c\ a\ b$$

Exemple

$$\begin{array}{rcl} \mathsf{if}\ \mathsf{T}\ \mathsf{then}\ a\ \mathsf{else}\ b & \equiv & \mathsf{T}\ a\ b \\ & \equiv & (\lambda xy.x)\ a\ b \\ & \to_\beta & (\lambda y.a)\ b \\ & \to_\beta & a \end{array}$$

Exercice : opérateurs booléens
Le terme suivant représente un opérateur booléen, mais lequel ?

$$\lambda ab.ab\mathsf{F}$$

Trouver des termes définissant les autres opérateurs

Paires et projections
Paire

$$\langle a, b\rangle \equiv \lambda s.s\ a\ b$$

Projections

$$\begin{array}{lcll} \pi_1 & \equiv & \lambda p.p\ (\lambda ab.a) & (\equiv \lambda p.p\ \mathsf{T}) \\ \pi_2 & \equiv & \lambda p.p\ (\lambda ab.b) & (\equiv \lambda p.p\ \mathsf{F}) \end{array}$$

Exemple

$$\begin{array}{rcl} \pi_2\ \langle A, B\rangle & \equiv & (\lambda p.p\ (\lambda ab.b))\ \langle A, B\rangle \\ & \to_\beta & \langle A, B\rangle\ \lambda ab.b \\ & \equiv & (\lambda s.s\ A\ B)\ \lambda ab.b) \\ & \to_\beta & (\lambda ab.b)\ A\ B \\ & \to_\beta & (\lambda b.b)\ B \\ & \to_\beta & B \end{array}$$

Nombres entiers
Pour chaque $n \in \mathbb{N}$ on définit un λ-terme $[n]$

$$\begin{array}{rcl} [0] & \equiv & \mathsf{I} \\ [n+1] & \equiv & \langle \mathsf{F}, [n]\rangle \end{array}$$

Quelques opérations de base

$$\begin{array}{rcll} \mathsf{S} & \equiv & \lambda x.\langle \mathsf{F}, x\rangle & \text{successeur} \\ \mathsf{P} & \equiv & \lambda x.x\mathsf{F} & \text{prédécesseur} \\ \mathsf{isZ} & \equiv & \lambda x.x\mathsf{T} & \text{test à zéro} \end{array}$$

Exercice : manipulation des entiers
Rappel

$$\begin{array}{rcl} [0] & \equiv & \mathsf{I} \\ [n+1] & \equiv & \langle \mathsf{F}, [n]\rangle \end{array} \qquad \begin{array}{rcl} \mathsf{S} & \equiv & \lambda x.\langle \mathsf{F}, x\rangle \\ \mathsf{P} & \equiv & \lambda x.x\mathsf{F} \\ \mathsf{isZ} & \equiv & \lambda x.x\mathsf{T} \end{array} \qquad \begin{array}{rcl} \langle a, b\rangle & \equiv & \lambda c.cab \\ \mathsf{T} & \equiv & \lambda ab.a \\ \mathsf{F} & \equiv & \lambda ab.b \end{array}$$

Vérifier les égalités suivantes

$$\begin{array}{rcl} \mathsf{S}\ [n] & =_\beta & [n+1] \\ \mathsf{P}\ [n+1] & =_\beta & [n] \\ \mathsf{P}\ [0] & =_\beta & \mathsf{F} \\ \mathsf{isZ}\ [0] & =_\beta & \mathsf{T} \\ \mathsf{isZ}\ [n+1] & =_\beta & \mathsf{F} \end{array}$$

Définir un terme add tel que

$$\mathsf{add}\ [n]\ [m] \quad = \quad [n+m]$$

Addition
On aimerait pouvoir écrire une fonction récursive

$$\mathsf{add}\ n\ m \quad = \quad \mathsf{if\ isZ}\ n\ \mathsf{then}\ m\ \mathsf{else\ add}\ (\mathsf{P}\ n)\ (\mathsf{S}\ m)$$

Souci : trouver un tel λ-terme add consiste à résoudre une équation

2 Points fixes

Point fixe, en arithmétique
Un point fixe d'une fonction f est un x tel que

$$f(x) = x$$

Trouver un point fixe de f, c'est résoudre l'équation $x = f(x)$
Les fonctions arithmétiques ont plus ou moins de points fixes

$x \mapsto x$	une infinité
$x \mapsto x+1$	aucun
$x \mapsto x^2$	deux
$f : [0;1] \to [0;1]$	au moins un si croissante et continue

Point fixe, en λ-calcul
En λ-calcul, t est un point fixe de f si

$$f\ t\ =_\beta\ t$$

Théorème de point fixe

Tout λ-terme f admet un point fixe

Le théorème de point fixe affirme que dans le λ-calcul on peut toujours résoudre l'équation $t =_\beta f\ t$

Combinateur de point fixe de Turing
Combinateur de point fixe

$$\begin{array}{rcl} \Theta & \equiv & A\ A \\ A & \equiv & \lambda xy.y(xxy) \end{array}$$

Vérification de $f(\Theta f) =_\beta \Theta f$

$$\begin{array}{rcl} \Theta\ f & \equiv & (\lambda xy.y(xxy))\ A\ f \\ & \to_\beta & (\lambda y.y(AAy))\ f \\ & \equiv & (\lambda y.y(\Theta y))\ f \\ & \to_\beta & f(\Theta f) \end{array}$$

Pour tout f, le terme $\Theta\ f$ est un point fixe de f

Récursion mutuelle
Théorème du double point fixe

$$\forall f, g \quad \exists a, b \qquad a =_\beta f\ a\ b \quad \wedge \quad b =_\beta g\ a\ b$$

Preuve : on pose

$$\begin{array}{rcl} d & \equiv & \Theta\ (\lambda x.\langle f\ (\pi_1 x)\ (\pi_2 x), g\ (\pi_1 x)\ (\pi_2 x)\rangle) \\ a & \equiv & \pi_1\ d \\ b & \equiv & \pi_2\ d \end{array}$$

Alors

$$\begin{array}{rcrcl} & & d & \to^* & \langle f\ (\pi_1 d)\ (\pi_2 d), g\ (\pi_1 d)\ (\pi_2 d)\rangle \\ a & \equiv & \pi_1 d & \to^* & f\ (\pi_1 d)\ (\pi_2 d)\ \equiv\ f\ a\ b \\ b & \equiv & \pi_2 d & \to^* & g\ (\pi_1 d)\ (\pi_2 d)\ \equiv\ g\ a\ b \end{array}$$

Extensible à un point fixe n-aire

Retour sur l'addition

$$\begin{array}{rcl} \mathsf{add}\ n\ m & = & \mathsf{if}\ \mathsf{isZ}\ n\ \mathsf{then}\ m\ \mathsf{else}\ \mathsf{add}\ (\mathsf{P}\ n)\ (\mathsf{S}\ m) \\ \mathsf{add} & = & \lambda nm.\mathsf{if}\ \mathsf{isZ}\ n\ \mathsf{then}\ m\ \mathsf{else}\ \mathsf{add}\ (\mathsf{P}\ n)\ (\mathsf{S}\ m) \\ \mathsf{add} & = & (\lambda fnm.\mathsf{if}\ \mathsf{isZ}\ n\ \mathsf{then}\ m\ \mathsf{else}\ \mathsf{f}\ (\mathsf{P}\ n)\ (\mathsf{S}\ m))\ \mathsf{add} \end{array}$$

On construit donc add comme un point fixe avec

$$\mathsf{add}\ \equiv\ \Theta\ (\lambda fnm.\mathsf{if}\ \mathsf{isZ}\ n\ \mathsf{then}\ m\ \mathsf{else}\ \mathsf{f}\ (\mathsf{P}\ n)\ (\mathsf{S}\ m))$$

Exercice : suite de Fibonacci
Définir un λ-terme représentant la fonction de Fibonacci, définie par

$$\begin{array}{rcl} f(0) & = & 0 \\ f(1) & = & 1 \\ f(n+2) & = & f(n+1) + f(n) \end{array}$$

Exercice : quel est ce point fixe ?
On a dit dans cette section :

— $f : x \mapsto x + 1$ est une fonction qui n'a pas de point fixe
— $F = \lambda x.\mathsf{S}\ x$ est un λ-terme, et a donc un point fixe

Comment ces deux faits peuvent-ils cohabiter ?

Exercice : entiers/itérateurs de Church
Définition alternative pour $[n]$

$$[n] \equiv \lambda fx.f^n\ x$$

Interprétation : $[n]$ prend en argument une fonction f et renvoie la fonction itérant n fois f
Montrer que $\lambda nfx.f(nfx)$ représente la fonction successeur
Trouver des termes représentant l'addition, la multiplication et le prédécesseur

Exercice : combinateur Y de Curry
Combinateur

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

Vérifier que pour tout terme t on a

$$\mathsf{Y}\ t\ =_\beta\ t\ (\mathsf{Y}\ t)$$

A-t-on aussi $\mathsf{Y}\ t\ \to^*_\beta\ t\ (\mathsf{Y}\ t)$?

3 Le λ-calcul est un modèle de calcul

Définissabilité
Une fonction mathématique $\varphi : \mathbb{N}^p \to \mathbb{N}$ est *définissable* s'il existe un λ-terme $f \in \Lambda$ tel que

$$\forall n_1, \ldots, n_p \in \mathbb{N}, \quad f[n_1] \ldots [n_p] =_\beta [\varphi(n_1, \ldots, n_p)]$$

Par la propriété de Church-Rosser on aurait aussi pu donner la condition

$$\forall n_1, \ldots, n_p \in \mathbb{N}, \quad f[n_1] \ldots [n_p] \to^*_\beta [\varphi(n_1, \ldots, n_p)]$$

Propriété : les fonctions λ-définissables sont exactement les fonctions récursives

Fonctions récursives initiales
Zéro $Z(n) = 0$

— $\mathsf{Z} = \lambda x.[0]$

Successeur $S(n) = n + 1$

— $\mathsf{S} = \lambda x.\langle \mathsf{F}, x\rangle$

Projection $U_i^p(n_0, \ldots, n_p) = n_i$ avec $0 \leqslant i \leqslant p$

— $\mathsf{U}_i^p = \lambda x_0 \ldots x_p.x_i$

Composition de fonctions récursives
Si F, G_1, ..., G_m sont récursives alors la fonction H définie par

$$H(\vec{n}) = F(G_1(\vec{n}), \ldots, G_m(\vec{n}))$$

est récursive
Supposons F, G_1, ..., G_m définies respectivement par f, g_1, ..., g_m alors H peut être définie par

$$h \equiv \lambda \vec{x}.F\ (G_1\ \vec{x}) \ldots (G_m\ \vec{x})$$

Récursion primitive
Si F et G sont récursives alors la fonction H définie par

$$\begin{array}{rcl} H(0,\vec{n}) & = & F(\vec{n}) \\ H(k+1,\vec{n}) & = & G(H(k,\vec{n}),k,\vec{n}) \end{array}$$

est récursive
Supposons F et G définies respectivement par f et g, on cherche à définir un h tel que

$$h \quad \equiv \quad \lambda x\vec{y}.\mathsf{if}\ \mathsf{isZ}\ x\ \mathsf{then}\ f\ \vec{y}\ \mathsf{else}\ g\ (h\ (\mathsf{P}x)\ \vec{y})\ (\mathsf{P}x)\ \vec{y}$$

Un tel h existe par théorème de point fixe

Minimisation
Si F est récursive et telle que

$$\forall \vec{n}\ \exists m\ F(\vec{n},m) = 0$$

alors la fonction M définie par

$$M(\vec{n}) \quad = \quad \text{le plus petit entier } m \text{ tel que } F(\vec{n},m) = 0$$

est récursive
Supposons F définie par f, alors on pose

$$m \quad \equiv \quad \lambda\vec{x}.(\Theta\ (\lambda hy.\mathsf{if}\ \mathsf{isZ}\ (f\vec{x}y)\ \mathsf{then}\ y\ \mathsf{else}\ h(\mathsf{S}y))\ [0])$$

Bilan
On a pu coder en λ-calcul

- les fonctions initiales Z, S et U_i^p
- la composition
- la récursion primitive
- la minimisation

Toutes les fonctions récursives sont λ-définissables
Le λ-calcul est Turing-complet

4 Décidabilité

Numéroter les λ-termes
On se donne une injection $\varphi : \mathbb{N}^2 \to \mathbb{N}$, par exemple $\varphi(x,y) \equiv 2^x(2y+1) - 1$
On numérote l'ensemble des variables : $\{x_0, x_1, x_2, \ldots\}$
On en déduit une fonction $\sharp : \Lambda \to \mathbb{N}$ donnant un numéro unique à chaque λ-terme

$$\begin{array}{rcl} \sharp x_i & = & \varphi(0,\ i) \\ \sharp(t\ u) & = & \varphi(1,\ \varphi(\sharp t,\ \sharp u)) \\ \sharp(\lambda x_i.t) & = & \varphi(2,\ \varphi(i,\ \sharp t)) \end{array}$$

Codage

$$[t] \quad \equiv \quad [\sharp t]$$

Théorème d'énumération (admis)

$$\text{Il existe } \mathsf{E} \text{ tel que pour tout } \lambda\text{-terme clos } t,\ \mathsf{E}\ [t] \to_\beta^* t$$

Deuxième théorème de point fixe
Encodage des λ-termes en λ-calcul, on combine numérotation et représentation des entiers

$$[t] \equiv [\sharp t]$$

Théorème

$$\forall f\ \exists t \quad f\ [t] =_\beta t$$

Preuve du deuxième théorème de point fixe
On a des fonctions récursives φ_A et φ_N telles que

$$\begin{array}{rcl} \varphi_A(\sharp t, \sharp u) & = & \sharp(t\ u) \\ \varphi_N(\sharp t) & = & \sharp[t] \end{array}$$

Elles sont définies par des λ-termes A et N tels que

$$\begin{array}{rcl} \mathsf{A}\ [t]\ [u] & =_\beta & [t\ u] \\ \mathsf{N}\ [t] & =_\beta & [[t]] \end{array}$$

On pose

$$\begin{array}{rcl} w & \equiv & \lambda x.f\ (\mathsf{A}\ x\ (\mathsf{N}\ x)) \\ z & \equiv & w\ [w] \end{array}$$

Alors z est le point fixe cherché

$$\begin{array}{rcll} z \equiv w\ [w] & =_\beta & f\ (\mathsf{A}\ [w]\ (\mathsf{N}\ [w])) & \\ & =_\beta & f\ (\mathsf{A}\ [w]\ [[w]]) & \\ & =_\beta & f\ [w\ [w]] & \equiv\ f\ [z] \end{array}$$

Théorème d'indécidabilité de Scott
Théorème

1. deux ensembles $A, B \subseteq \Lambda$ non vides et β-clos ne sont pas récursivement séparables
2. un ensemble $A \subseteq \Lambda$ non trivial et β-clos n'est pas récursif

Définitions

— E est β-clos si $\forall x, y \in \Lambda\ x \in E \wedge x =_\beta y \implies y \in E$

— E non trivial s'il existe $x \in E$ et $y \notin E$

— A et B sont récursivement séparables s'il existe C récursif tel que $A \subseteq C$ et $B \cap C = \emptyset$

— C est récursif si sa fonction caractéristique est récursive

Preuve du théorème de Scott
Deux ensembles $A, B \subseteq \Lambda$ non vides et β-clos ne sont pas récursivement séparables
Supposons qu'il existe C récursif avec $A \subseteq C$ et $B \cap C = \emptyset$ Sa fonction caractéristique est réalisée par un terme f tel que

$$\begin{array}{lll} t \in C & \Longrightarrow & f\ [t] =_\beta [1] \\ t \notin C & \Longrightarrow & f\ [t] =_\beta [0] \end{array}$$

A et B étant non vides, on prend $a \in A$ et $b \in B$ et on définit

$$g \equiv \lambda x.\mathsf{if\ isZ}\ (f\ x)\ \mathsf{then}\ a\ \mathsf{else}\ b$$

On a donc

$$\begin{array}{lll} t \in C & \Longrightarrow & g\ [t] =_\beta b \\ t \notin C & \Longrightarrow & g\ [t] =_\beta a \end{array}$$

Du deuxième théorème de point fixe il existe z tel que $g\ [z] = z$

$$\begin{array}{lllll} z \in C & \Longrightarrow & z =_\beta g\ [z] =_\beta b \in B & \Longrightarrow & z \notin C \\ z \notin C & \Longrightarrow & z =_\beta g\ [z] =_\beta a \in A & \Longrightarrow & z \in C \end{array}$$

Contradiction !

Indécidabilité de la β-équivalence
Le problème de la β-égalité entre deux lambda-termes est indécidable
Supposons que f est un λ-terme tel que $f\ [a]\ [b]$ vaut $[1]$ si a et b sont équivalent et $[0]$ sinon
On définit $A = \{x \mid x =_\beta a\}$

- par définition, A est β-clos
- A est non vide, car il contient a
- $\Lambda \setminus A$ est non vide, car
  - si a a une forme normale, alors $\Omega \notin A$
  - si a n'a pas de forme normale, alors $\lambda x.x \notin A$

Donc A n'est pas récursif
Or $f\ [a]$ calcule la fonction caractéristique de A *Contradiction*

Exercice : indécidabilité de la terminaison
Le problème de l'existence d'une forme normale pour un λ-terme est indécidable

Indécidabilité de la stratégie optimale
Stratégie : fonction $F : \Lambda \to \Lambda$ telle que

$$\forall t \in \Lambda \quad t \to_\beta F(t)$$

Stratégie *optimale* : qui décrit systématiquement un chemin le plus court vers l'éventuelle forme normale
Il n'existe pas de stratégie optimale calculable

Indécidabilité de la stratégie optimale : plan
On considère l'ensemble de λ-termes

$$t_n \equiv (\lambda x.x\mathsf{E}x)\ (\lambda y.y[n](\mathsf{II}))$$

où E énumère les λ-termes avec au plus une variable libre a
Avec E en forme normale il faut choisir, pour chaque n, entre :

— réduire $t_n \to_\beta (\lambda y.y[n](\mathsf{II}))\ \mathsf{E}\ (\lambda y.y[n](\mathsf{II}))$

— réduire $t_n \to_\beta (\lambda x.x\mathsf{E}x)\ (\lambda y.y[n]\mathsf{I})$

Mais le meilleur choix dépend de la forme normale de $\mathsf{E}\ [n]$

Stratégie optimale : premier cas
Si $\mathsf{E}\ [n] \to_\beta^* \lambda xyz.z$ en k étapes alors

$$\begin{array}{rcl}(\lambda y.y[n](\mathsf{II}))\ \mathsf{E}\ (\lambda y.y[n](\mathsf{II})) & \to_\beta & \mathsf{E}\ [n]\ (\mathsf{II})\ (\lambda y.y[n](\mathsf{II})) \\ & \to_\beta^* & (\lambda xyz.z)\ (\mathsf{II})\ (\lambda y.y[n](\mathsf{II})) \\ & \to_\beta^2 & \lambda z.z\end{array}$$

optimalement en $k+3$ étapes et

$$\begin{array}{rcl}(\lambda x.x\mathsf{E}x)\ (\lambda y.y[n]\mathsf{I}) & \to_\beta & (\lambda y.y[n]\mathsf{I})\ \mathsf{E}\ (\lambda y.y[n]\mathsf{I}) \\ & \to_\beta & \mathsf{E}\ [n]\ \mathsf{I}\ (\lambda y.y[n]\mathsf{I}) \\ & \to_\beta^* & (\lambda xyz.z)\ \mathsf{I}\ (\lambda y.y[n]\mathsf{I}) \\ & \to_\beta^2 & \lambda z.z\end{array}$$

optimalement en $k+4$ étapes

Stratégie optimale : deuxième cas
Si $\mathsf{E}\ [n] \to_\beta^* a$ en k étapes alors

$$\begin{array}{rcl}(\lambda y.y[n](\mathsf{II}))\ \mathsf{E}\ (\lambda y.y[n](\mathsf{II})) & \to_\beta & \mathsf{E}\ [n]\ (\mathsf{II})\ (\lambda y.y[n](\mathsf{II})) \\ & \to_\beta^* & a\ (\mathsf{II})\ (\lambda y.y[n](\mathsf{II})) \\ & \to_\beta^2 & a\ \mathsf{I}\ (\lambda y.y[n]\mathsf{I})\end{array}$$

optimalement en $k+3$ étapes et

$$\begin{array}{rcl}(\lambda x.x\mathsf{E}x)\ (\lambda y.y[n]\mathsf{I}) & \to_\beta & (\lambda y.y[n]\mathsf{I})\ \mathsf{E}\ (\lambda y.y[n]\mathsf{I}) \\ & \to_\beta & \mathsf{E}\ [n]\ \mathsf{I}\ (\lambda y.y[n]\mathsf{I}) \\ & \to_\beta^* & a\ \mathsf{I}\ (\lambda y.y[n]\mathsf{I})\end{array}$$

optimalement en $k+2$ étapes

Stratégie optimale : bilan

$$t_n \equiv (\lambda x.x\mathsf{E}x)\ (\lambda y.y[n](\mathsf{II}))$$

Si F est une stratégie optimale, alors

— si $\mathsf{E}\ [n] \to_\beta^* \lambda xyz.z$ alors $F(t_n) = (\lambda y.y[n](\mathsf{II}))\ \mathsf{E}\ (\lambda y.y[n](\mathsf{II}))$

— si $\mathsf{E}\ [n] \to_\beta^* a$ alors $F(t_n) = (\lambda x.x\mathsf{E}x)\ (\lambda y.y[n]\mathsf{I})$

Une stratégie optimale sépare donc les ensembles

$$\{n \mid \mathsf{E}\ [n] \to_\beta^* \lambda xyz.z\} \qquad \text{et} \qquad \{n \mid \mathsf{E}\ [n] \to_\beta^* a\}$$

Or ces ensembles ne sont pas récursivement séparables, car par théorème de Scott

$$\{t \mid t \to_\beta^* \lambda xyz.z\} \qquad \text{et} \qquad \{t \mid t \to_\beta^* a\}$$

ne sont pas récursivement séparables

DM - *à rédiger et à me renvoyer avant le prochain cours*

Codage du λ-calcul en λ-calcul

— proposer une manière de construire un λ-terme $[t]$, représentant l'arbre de syntaxe du λ-terme t

— définir un λ-terme qui, appliqué à une représentation $[t]$, produit la représentation $[[t]]$ du λ-terme représentant t

— en déduire une preuve du théorème d'indécidabilité de Scott qui n'utilise aucun codage numérique de Gödel