LOGIQUE FORMELLE

λ calcul : Histoire
- 1901 - Hilberg - les 10 pb du siècle
- 1930s' - Church & Curry - Théorie des fonctions
- 1930s' - Turing - Effective computability
- 1920s' - Brouwer, Heyting & Kolmogorov - Rpz de preuve formelle
- 1960s' - McCarthy & Scott - Base de pgm° fct^L
- 1960s' - Montague - Sémantique pour les lgg naturels.

Goedel → Thm d'incomplétude : On ne peut pas automatiser toutes les preuves.
⇒ pt de départ de l'informatique ⇒ distinct° entre prouvable et non prouvable, calculable et non calculable.

Mais art de mecaniser une preuve → Qu'est ce qu'une preuve?

λ-calcul = | théorie des fct math°
| 1° lgg de prog funct^L
→ sémantique operationnelle (sem° des petits pas)

En maths plutot sem° des gds pas ⇒ sem° dénotationnelle + abstrait.

Variables : lettres min
Funct° : λ var . Autre
Application : $\lambda t_1 \lambda t_2$
associative à gauche
} λtermes

Fct° ¬
Abstract° assoc à dte

Pls arg autorisés → curryfication.

fct° : $x \longmapsto 2x+1 \Leftrightarrow \lambda x . 2x+1$

$$\frac{}{x \in \Lambda}\, x \in \mathcal{V} \qquad \frac{M \in \Lambda \quad N \in \Lambda}{(MN) \in \Lambda} \qquad \frac{M \in \Lambda}{(\lambda x . M) \in \Lambda}\, x \in \mathcal{V}$$

avec Λ l'ensemble des λ termes

var — app — fct°

→ On peut contruire avec ça des λtermes complexes
On peut voir la construct° comme un arbre "à l'envers" la racine est la ccl° et les feuilles les hypothèses de base de la preuve (que l'expr finale est un λterm)

Sous terme :
- sub $(x) := \{x\}$
- sub $(\lambda x \cdot M) := \{\lambda x \cdot M\} \cup$ sub (M)
- sub $(MN) := \{MN\} \cup$ sub $(M) \cup$ sub (N)

Variable libre :
- $FV(x) := \{x\}$
- $FV(\lambda x \cdot M) := FV(M) \setminus \{x\}$
- $FV(MN) := FV(M) \cup FV(N)$

Variable liée = ! Variable libre
↳ liée par l'abstract°

Un λterm sans variable libre est un terme clos.

M et N sont α convertible (noté $M \equiv_\alpha N$) ssi leur seule diff est le nom des var d'extrémité sans introduire de capture

ex : $\lambda x \cdot x \equiv_\alpha \lambda y \cdot y \equiv_\alpha \lambda z \cdot z$ (On notera $\equiv_\alpha$ par $=$)
$x \lambda x \cdot x \equiv_\alpha x \lambda y \cdot y$
$x \lambda x \cdot x \not\equiv_\alpha y \lambda y \cdot y$
$\lambda x \cdot \lambda y \cdot xy \not\equiv_\alpha \lambda x \cdot \lambda x \cdot xx$ (⚠ scope) → capture

Substitut° de x par M dans N : $[M/x]N$
toutes les occ libres de x sont remplacée par M.

ex : $[\lambda z \cdot zz / x] \lambda y \cdot xy = \lambda y \cdot (\lambda z \cdot zz) y$.

On peut aussi écrire $N[x \leftarrow M]$

La β conversion entre 2 termes est la relation def par:
$$(\lambda x \cdot M)N \;\beta\; [N/x]M \quad \forall M, N \in \Lambda$$

λβ formal system

$$\frac{}{M = M} \qquad \frac{M = N}{N = M} \qquad \frac{M = N \quad N = L}{M = L} \qquad \frac{M = M' \quad N = N'}{MN = M'N'} \qquad \frac{M = N}{\lambda x \cdot M = \lambda x \cdot N}$$

$$\frac{}{(\lambda x \cdot M)N = [N/x]M}$$

One step R-Reduction:
la relat° $\xrightarrow[R]{}$ est la + petite relat° tq

$$\frac{(M,N) \in R}{M \xrightarrow[R]{} N} \qquad \frac{M \xrightarrow[R]{} N}{ML \xrightarrow[R]{} NL} \qquad \frac{M \xrightarrow[R]{} N}{LM \xrightarrow[R]{} LN} \qquad \frac{M \xrightarrow[R]{} N}{\lambda x \cdot M \xrightarrow[R]{} \lambda x \cdot N}$$

Multiple step
la relat° $\xrightarrow[R]{*}$ est la + petite relat° tq

$$\frac{M \xrightarrow[R]{} N}{M \xrightarrow[R]{*} N} \qquad \frac{}{M \xrightarrow[R]{*} M} \qquad \frac{M \xrightarrow[R]{*} N \quad N \xrightarrow[R]{*} L}{M \xrightarrow[R]{*} L}$$

$\xrightarrow[R]{*}$ est $(\xrightarrow[R]{})$ 0 ou plus de fois.

Une β-redex est un terme de la forme $(\lambda x \cdot M) N$

One step β-Reduct° : $\overline{(\lambda x \cdot M)N \xrightarrow[\beta]{} [N/x]M}$

$\xrightarrow[\beta]{*}$ est la fermeture transitive & reflexive de $\xrightarrow[\beta]{}$

$\equiv_\beta$ est la fermeture transitive, reflexive & symetrique de $\xrightarrow[\beta]{}$

ex : $(\lambda x \cdot x) y \rightarrow y \qquad (\lambda x \cdot xx) y \rightarrow yy$
$(\lambda x \cdot xx)(\lambda x \cdot xx) \rightarrow (\lambda x \cdot xx)(\lambda x \cdot xx)$

Combinateurs (termes clos) :

- $\omega \equiv \lambda x \cdot xx$
- $\Omega \equiv \omega\omega$
- $\tilde{\Omega} \equiv \lambda x \cdot x(xx)$
- $K \equiv \lambda x \cdot (\lambda y \cdot x)$

$$\Omega = \omega\omega \xrightarrow{\beta} \omega\omega = \Omega$$

$$\tilde{\Omega}\tilde{\Omega} \xrightarrow{\beta} \tilde{\Omega}(\tilde{\Omega}\tilde{\Omega}) \xrightarrow{\beta} \tilde{\Omega}(\tilde{\Omega}(\tilde{\Omega}\tilde{\Omega})) \xrightarrow{\beta} \dots$$:c

ex : $(\lambda x \cdot x)((\lambda y \cdot y)\, x) \xrightarrow{*} x$
$(\lambda x \cdot xx)((\lambda x \cdot xx) y) \xrightarrow{*} yy(yy)$

$(\lambda x \cdot xx)((\lambda x \cdot x) y) \xrightarrow{*} yy$
$(\lambda x \cdot x)((\lambda x \cdot xx) y) \xrightarrow{*} yy$

$\lambda\beta \vdash (\lambda x \cdot xx)((\lambda x \cdot x) y) = (\lambda x \cdot x)((\lambda x \cdot xx) y)$

η réduct° : $\lambda x \cdot Mx \xrightarrow{\eta} M$ (M ne contient pas de x)

η expansion : $M \xrightarrow{\eta_{exp}} \lambda x \cdot Mx$

R-Normal Form (R-NF) : un terme en RNF ne contient plus de N tq $M \xrightarrow{R} N$

R-Strongly Normalizat° Term : un terme M est R-Strongly Norm^able si il n'y a pas une infinité de one step réduct° partant de M

ex : $I = \lambda x . x$ est β-NF

II a une β-NF : c'est I
II est R-stronly Norm^able

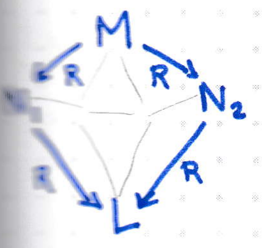
Ω n'est pas normalisable.

$KI\Omega$ est faiblement normalisable $\to I$ mais pas fortement normalisable
($KI\Omega$ peut se réduire en I ou en $KI\Omega$)

R est forteu^t norm^ble si ts les term sont R-forteu^t norm^ble
" " faibleu^t " " " " " " R-norm^ble

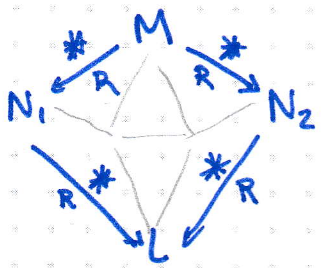
Stratégie de réduct° : fct spécifiant quel est la prochaine one step réduct° à faire.

$\xrightarrow{R}$ satisfait la propriété du diamand si $M \xrightarrow{R} N_1$, $M \xrightarrow{R} N_2$ implique $\exists L$ tq $N_1 \xrightarrow{R} L$, $N_2 \xrightarrow{R} L$



Prop du diamand

$\xrightarrow{R}$ est Church Rosser si $\xrightarrow[R]{*}$ satisfait la prop du diamand.

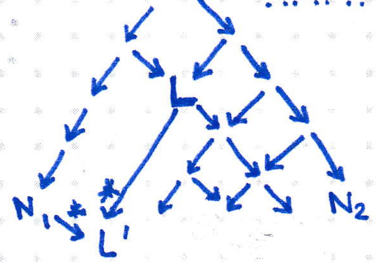


Church Rosser

$\xrightarrow{R}$ a la prop de la forme normale unique si $M \xrightarrow[R]{*} N_1$, $M \xrightarrow[R]{*} N_2$ avec N_1 et N_2 en forme normale implique $N_1 \equiv N_2$

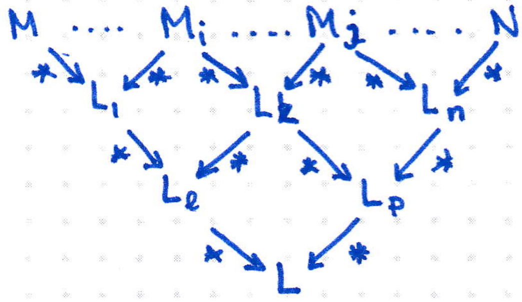
Diamand implique Church Rosser

Diamand



Si R est Church Rosser alors $M =_R N$ ssi $\exists L$ tq $M \xrightarrow[R]{*} L$ et $N \xrightarrow[R]{*} L$

$M =_R N$



Si R est Church Rosser alors il a la prop de la forme normale unique.

la β reduction a la prop de Church Rosser
→ Any term has (at most) a unique NF

Si → est une strat de reduct° alors cf slides.

- Head reduct° $\xrightarrow{h}$: $\lambda\vec{x}.(\lambda y.M)N\vec{L} \xrightarrow{h} \lambda\vec{x}.[N/y]M\vec{L}$
  tous les termes sont de la forme $\lambda\vec{x}.(\lambda y.M)\vec{L}$ ou $\lambda\vec{x}.y\vec{L}$

ex : $K I \Omega \xrightarrow{h} I$ $K I \Omega \xrightarrow{h} \omega I \xrightarrow{h} I I \xrightarrow{h} I$ $x I x \not\xrightarrow{h} xx$

- leftmost reduct° $\xrightarrow{l}$ single step of β convers° le + à gauche.
  elle est normalisante.

- Evaluat° paresseuse

→ λ → crée des abstract°
→ Variables x y z   λ (abstract°) $\lambda x.x$
→ Applicat° MN (j'applique M à N) $(\lambda x.M)N \longrightarrow_\beta M[x \leftarrow N]$
→ Si $M \not\rightarrow_\beta N$ alors M est en forme normale.
→ $\omega = \lambda x.xx$ $\Omega = \omega\omega$

Booleen dans le λ calcul : if M then N else L
→ if MNL. Mais y a-t-il vraiment besoin du if ? Non.
MNL avec M booleen fct appliquée à N et L
$T := \lambda xy.x$ $F := \lambda xy.y$ (Booleen de Church)

Entiers de Church
$\underline{n} := \lambda f.\lambda x.f^n x = \lambda f.\lambda x.(f\dots(f(x))\dots)$
ex : $\underline{2} = \lambda f.\lambda x.f(f(x))$ $\underline{3} = \lambda f.\lambda x.f(f(f(x)))$

Opérat° :
- successeur $succ := \lambda n.\lambda f.\lambda x.f(nfx)$
- addition $plus := \lambda m.\lambda n.\lambda f.\lambda x.mf(nfx)$
  $plus := \lambda m.\lambda n.n\, succ\, m$
  $plus := \lambda n.n\, succ$

$succ \xrightarrow[\beta]{*} \lambda f.\lambda x.f^{n+1}x.$

Paires pair := $\lambda xy . \lambda f . fxy$
first := $\lambda p . pT$
second := $\lambda p . pF$

Combinateur de point fixe

- Combinateur Y de Curry
  $Y := \lambda f . (\lambda x . f(xx))(\lambda x . f(xx))$
- Combinateur Θ de Turing
  $\Theta := (\lambda xy . y(xxy))(\lambda xy . y(xxy))$

Il y a une infinité de combinateur de pt fixe

Stratégie de réduct° ds les lgg de pgmt°
→ Full beta reduct°
→ Applicative order : leftmost innermost
= appel par valeur → réduit les args en 1°
→ Normal order : leftmost outermost
= appel par nom (sauf redex ds abstract°)
= appel par besoin (evaluat° paresseuse) (pas de duplicat°)

LOGIQUE COMBINATOIRE

Alternative au λ calcul : Moses Ilyich Schönfinkel
Presque tous ses papiers ont été brulés par ses voisins pour se chauffer pendant l'hiver : il nous reste peu de chose.

Logique combinatoire : inventée par Moses Schönfinkel en 1920's
dev en 1925 par Haskell Curry. Simple.
λ reduction : inventée par Alonzo Church en 1936
complexe, implem pleine de pièges subtils.

- $S := \lambda x . (\lambda y . (\lambda z . ((xz)(yz))))$
- $K := \lambda x . (\lambda y . x)$
- $I := \lambda x . x$

} Classic combinators.

- $SXYZ \rightarrow XZ(YZ)$ • $KXY \rightarrow X$ • $IX \rightarrow X$ • $IX \rightarrow X$ • $SKKX \rightarrow KX(KX) \rightarrow X$
- $I = SKK$

Combinat° is left associative : $SKKX = (((SK)K)X) \rightarrow KX(KX) \rightarrow X$

Booléens : $T = K$ $F = KI$ $TXY \rightarrow X$ $FXY \rightarrow Y$

Combinateurs : bcp trop longs et moches.
- $Y = S(K(SII))(S(S(KS)K)(K(SII)))$
- le + simple combinateur de pt fixe en SK : $Y = SSK(S(K(SS(S(SSK))))K)$
- Celui de Jan Willem Klop
  $Y_K = (LLLLLLLLLLLLLLLLLLLLLLLLLL)$ (26 L !)
  où $L = \lambda abcdefghijklmnopqrstuvwxyzr\,(r(\text{this is a fixed point combinator}))$

Toutefois, la logique combinatoire a été abandonnée.

λ CALCUL TYPÉ

1° apparit° : • Curry 1934 logique combinatoire
• Church 1940

Types : obj syntaxique
$M : A$ = M est de type A.
ex : $IA : A$.

→ On arrête de considerer que tout peut être appliqué à tout.

Set of type var : $\alpha, \beta, \ldots$
Symbol → for funct° : $\alpha \to \alpha$, $\alpha \to (\beta \to \gamma) \ldots$
→ est associatif à dte : $\alpha \to \beta \to \gamma = \alpha \to (\beta \to \gamma)$

Church $\dfrac{x : \alpha}{\lambda x^{\alpha} . x : \alpha \to \alpha}$ Curry $\dfrac{x : \alpha}{\lambda x . x : \alpha \to \alpha}$

$\mathcal{TV}$: set of var $\alpha, \beta \ldots$ $\dfrac{}{\alpha \in \mathcal{T}} \alpha \in \mathcal{TV}$ $\dfrac{\sigma \in \mathcal{T} \quad \tau \in \mathcal{T}}{(\sigma \to \tau) \in \mathcal{T}}$

Statement $M : \sigma$ paire avec $M \in \Lambda$ et $\sigma \in \mathcal{T}$
Type context Γ set fini de statement over distinct var $\{x_1 : \sigma_1, \ldots\}$
la var x est assignée au type σ dans Γ ssi $x : \sigma \in \Gamma$.

$\Gamma - x$ est Γ sans tous les assignem$^+$ $x : \sigma$ $\Rightarrow \Gamma \upharpoonright M = \Gamma - FV(M)$

type derivat° : $\dfrac{M : \sigma \to \tau \quad N : \sigma}{MN : \tau}$
autres construits à partir de ces nodes ↑

$$\dfrac{\begin{matrix}[x : \sigma] \\ \vdots \\ M : \tau\end{matrix}}{\lambda x . M : \sigma \to \tau}$$

Statement $M : \sigma$ dérivable à partir du type context Γ, $\Gamma \vdash M : \sigma$
si il y a une dérivat° de $M : \sigma$ dont les non canceled assympt° sont ds Γ

Type dérivat°

- $\dfrac{}{\{x : \sigma\} \mapsto x : \sigma}$
- $\dfrac{\Gamma \mapsto M : \sigma \to \tau \quad \Delta \mapsto N : \sigma}{\Gamma \cup \Delta \mapsto MN : \tau}$ Γ, Δ consistent
- $\dfrac{\Gamma \mapsto M : \tau}{\Gamma \setminus \{x : \sigma\} \mapsto \lambda x . M : \sigma \to \tau}$ $\Gamma, \{x : \sigma\}$ consistent
- $\dfrac{\Gamma \mapsto M : \tau}{\Gamma, \Delta \vdash M : \tau}$

Type $\omega = \lambda x . xx$? On ne trouve pas de derivat° → pas de type

Un terme M est typable si $\exists \sigma$ tq $\vdash M : \sigma$
ex: $\omega\ \Omega$ pas typable, S K I typable, Y pas typable

On consider la derivat° Π pour $M : \sigma$
- Si $M = x$ alors $\Gamma = \{x : \sigma\}$ et $\Pi = \dfrac{}{\{x:\sigma\} \mapsto x : \sigma}$
- Si $M = NL$ alors $\Pi = \dfrac{\Gamma \upharpoonright N \to N : \tau \to \sigma \qquad \Gamma \upharpoonright L \to L : \tau}{\Gamma \mapsto NL : \sigma}$
- Si $M = \lambda x . N$ alors $\sigma = \sigma_1 \mapsto \sigma_1$ et
  - → si $x \in FV(N)$ $\Pi = \dfrac{\Gamma \cup \{x : \sigma_1\} \to N : \sigma_2}{\Gamma \mapsto \lambda x . N : \sigma_1 \to \sigma_2}$
  - → si $x \notin FV(N)$ $\Pi = \dfrac{\Gamma \to N : \sigma_2}{\Gamma \mapsto \lambda x . N : \sigma_1 \mapsto \sigma_2}$

α invariance : Si $\Gamma \mapsto M : \sigma$ et $M \equiv_\alpha N$ alors $\Gamma \mapsto N : \sigma$

Substitut° : Si $\Gamma \cup \{x : \tau\} \vdash M : \sigma$ et $\Delta \vdash N : \tau$
et Γ, Δ sont consistent et $x \notin FV(N)$
alors $\Gamma \cup \Delta \vdash [N/x] M : \sigma$

THM sujet reduct° : si $\Gamma \vdash M : \sigma$ et $M \longrightarrow_\beta N$ alors $\Gamma \vdash N : \sigma$

THM Expans° du sujet : Si $\Gamma \vdash N : \sigma$ et $M \longrightarrow_\beta N$ alors $\Gamma \vdash M : \sigma$

Tous les termes typables sont normalisant β-strongly.

$\Gamma = \{x_1 : \sigma_1, \dots, x_n : \sigma_n\}$ $\Gamma \vdash M : \tau \iff \vdash \lambda x_1 \dots x_n . M : \sigma_1 \to \dots \to \sigma_n \to \tau$

→ Type checking : Soit M, σ, $\vdash M : \sigma$?
→ Typabilité : Soit M, $\exists ? \sigma$ tq $\vdash M : \sigma$? $\vdash M : ?$ } decidable

Les types ne sont pas uniques !

LOGIQUE CLASSIQUE

Constantes a, b, c ...
Variables A, B, C ...
Connective

Ind Var x, y, z ...
Fct f, g, h ...
predicat P, Q, R ...
Quantifier $\forall, \exists$
ponctuat° . :

formule ::= propositional var
| ¬ formula
| formula ∧ formula
| formula ∨ formula
| formula ⇒ formula
| predicate (term)
| ∀ ind var . formula
| ∃ ind var . formula

term ::= cst
| fct (term)

Système de preuve :
- Hilbertian $
- Deduct° naturelle
- Calcul sequentiel
- Deduct° naturelle en calcul sequentiel

Les axiomes sont des formules considérées vraies a priori
ex : $\forall x \cdot x + 0 = x$

Schéma d'axiome : meta variables ex : $X + Y = Y + X$
utilisés qd on ne veut pas de quantificateurs.

$$\frac{H_1 \quad H_2 \; \cdots \; H_n}{C} \text{ Rule name} \qquad \frac{}{A} \text{ axiome name.}$$

$$\frac{A \qquad A \Rightarrow B}{B} \text{ modus ponens}$$

Une déduct° est un arbre dont la racine A est la conclusion et dont les feuilles Γ sont des hypothèses.

$$\begin{array}{c} \Gamma \\ \vdots \\ A \end{array}$$

Une preuve est une déduct° sans hypothèse.