06/02/2018

λ-CALCUL

(1) CM

Suite typage - Amphi λ-calcul

Typage

- controle à l'ecriture.
- aide à la conception.

Plusieurs systemes de type

- typage simple.
- typage polymorphe.

typage simple -

On type toujours dans un contexte (environement)

|| contexte (environement) : assignation d'un type à des variables libres.

On se donne un langage exprimant les types : τ_1 τ_2 τ_3 ....

$$\left.\begin{array}{l} x_1 : \tau_1 \\ x_2 : \tau_2 \\ \vdots \\ x_m : \tau_m \end{array}\right\} \text{Contexte } \Gamma$$

Pour le typage simple, les types sont de la forme suivante :

- soit un nom abstrait α, β, τ ...
- soit $\tau_1 \rightarrow \tau_2$ où τ_1 et τ_2 sont des types.

$(\alpha \rightarrow \alpha) \rightarrow (\alpha \rightarrow (\alpha \rightarrow \alpha))$

intuitivement: fonction de τ_1 vers τ_2.

- soit un type prédéfini : bool, nat ... (dependant du langage)

Regles de typage.

Definition : jugement de typage. $\underbrace{\Gamma}_{\text{contexte}} \vdash \underbrace{U}_{\lambda\text{-terme}} : \underbrace{\tau}_{\text{type}}$

(1) variable : $\sigma \vdash x : \tau$

ssi $x : \tau \in \sigma$

(2) application : $\dfrac{\sigma \vdash u : \tau_1 \rightarrow \tau_2 \quad \sigma \vdash v : \tau_1}{\sigma \vdash uv : \tau_2}$

③ abstraction.

$$\frac{\sigma, (x : \tau_1) \vdash u : \tau_2}{\sigma \vdash \lambda x . u : \tau_1 \to \tau_2}$$

exemple :

Pas de sens si $x \neq$ nat

$$\frac{x : \text{nat} \vdash \overbrace{x + 2}^{} : \text{nat}}{\lambda x.\ x + 2 : \text{nat} \to \text{nat}}$$

exemple ③ et ②

$$\frac{\dfrac{\dfrac{\sigma_{\text{nat}}, x : \text{nat}, + : \text{nat} \to (\text{nat} \to \text{nat}) \qquad \sigma_{\text{nat}}, x : \text{nat} \vdash x : \text{nat}}{\sigma_{\text{nat}}, x : \text{nat} \vdash (+, x) : \text{nat} \to \text{nat} \qquad \sigma_{\text{nat}}, x : \text{nat} \vdash 2 : \cancel{\text{int}}\ \text{nat}}}{\sigma_{\text{nat}}, x : \text{nat} \vdash x + 2 : \text{nat} \qquad \text{LIRE } (+ \quad x)\, 2}}{\dfrac{\sigma_{\text{nat}} \vdash \lambda x^{\text{nat}}. x + 2 : \text{nat} \to \text{nat} \qquad \sigma_{\text{nat}} \vdash 5 : \text{nat}}{\sigma_{\text{nat}} \vdash (\lambda x^{\text{nat}}. x + 2)\, 5 : \text{nat}}}$$

Remarque : on prend $\sigma_{\text{nat}} = (0_{\text{nat}}), (1\ _{\text{nat}}), \dots, (N_{\text{nat}})$

Illustration en Coq

```
Definition pl2 := fun (x:nat) => x+2
        check pl2

        section
        variable x : bool (nat)
        definition a = 3 + 2      (x)
        check (3+2)
        Definition pl2 := fun x:nat => x+2
        Definition pl := fun y:nat => y+x
```

Codage des structures de données en λ-calcul pur

Booleens : On se donne un type T

$\text{bool} \overset{=}{\text{def}} T \to T \to T.$

$\text{true} \overset{=}{\text{def}} \lambda x^T. \lambda y^T. x$

$\text{false} \overset{=}{\text{def}} \lambda x^T. \lambda y^T. y$

$\text{neg} \overset{=}{\text{def}} \lambda b^{T \to T \to T}. \lambda x^T. \lambda y^T. b\, y\, x$

06/02/2018

λ-CALCUL

② CM $\text{neg} . \text{true} = (\lambda b . \lambda x y . \, b y x)(\lambda x y . x)$

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

$\lambda x y . ((\lambda x' y' x') \, y \, x)$

y

31/01/2018

λ-CALCUL

1 TD

TD2 - Scéance 2 - typage elementaire

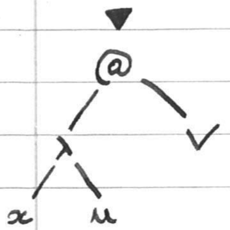
1. $x^{mat} + 2$
2. $\lambda x^{mat}. x + 2$
3. $(\lambda x^{mat}. x + 2)\, 3$
4. $(\lambda x^{mat}. \lambda y^{mat}. x - y)\, 108$
5. $f^{mat \to mat}\, 3 + 1$
6. $\lambda f^{mat \to mat}. (f\ 3+1)^{mat}$
7. $\lambda f^{mat \to mat}. (f\ 108 + 1)$
8. $(\lambda x^{mat\ \cancel{mat}}. x + 7)\ y^{mat}$
9. $(\lambda x^{mat \to mat}. x + 7)\ y^{mat}\ 4$
10. $(\lambda v.\ u\ v\ w)\ 4$
11. $(\lambda u.\ u\ v\ w)\ 4$
12. $(\lambda f^{mat \to mat}. (f\ 1))(\lambda x^{nat}. x + 2)^{nat}$

30/01/2018

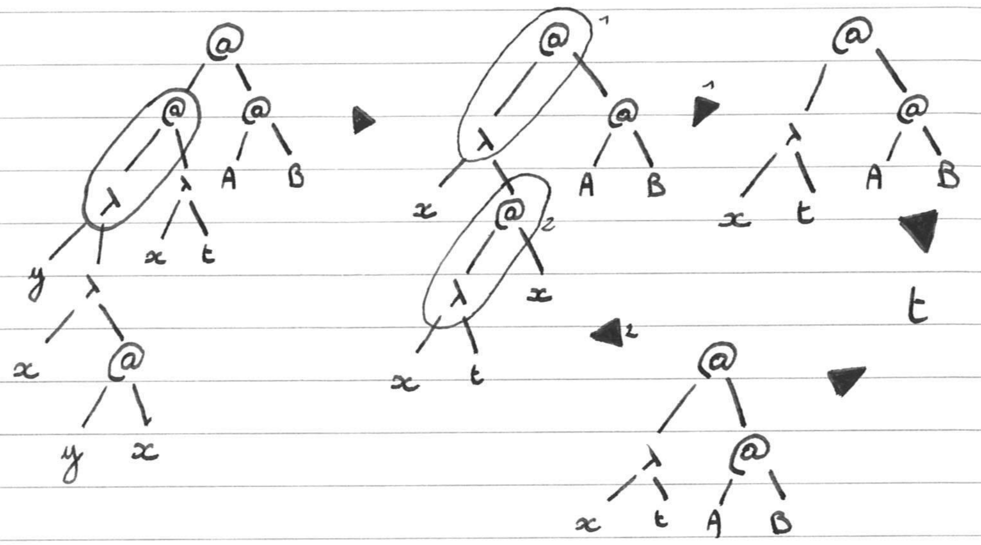
λ-CALCUL

① CM

$(\lambda x.u)V \blacktriangleright u[x:=V]$



$(\lambda yx.\ yx)\, T\, (AB)$ T peut être : $\lambda x.t$, $\lambda x.x.\lambda_u.t, \lambda_{u}.y$, $\lambda u.u$, $\lambda u.x$



α-convention (renommage de variable)

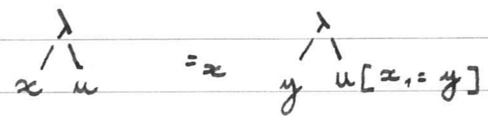
renommage variable libre (y) renommage variable liée (x)

$\lambda x.x + x$	$\lambda x.x + x$	$\lambda z.z + z$
$\lambda x.x + y$	$\lambda x.x + z$	$\lambda z.z + y$

pas identique

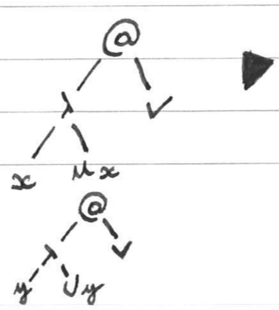
identique

Dans un terme



c'est "normalement" la même.

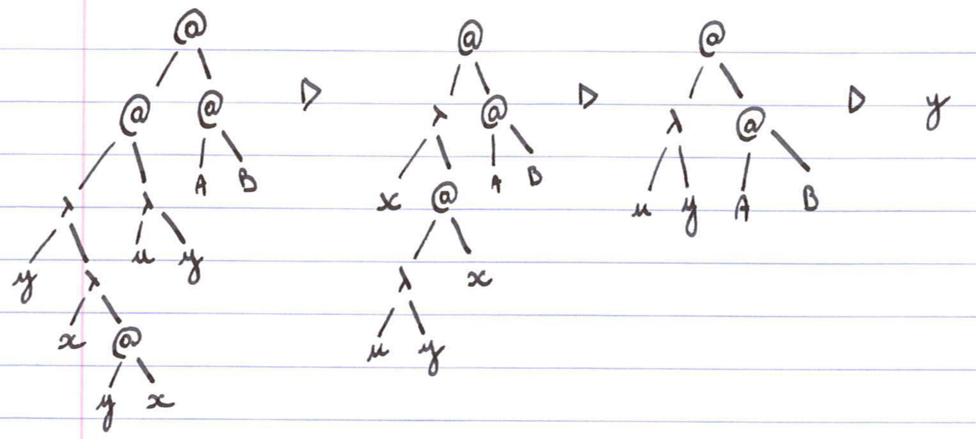
Supposons un redex

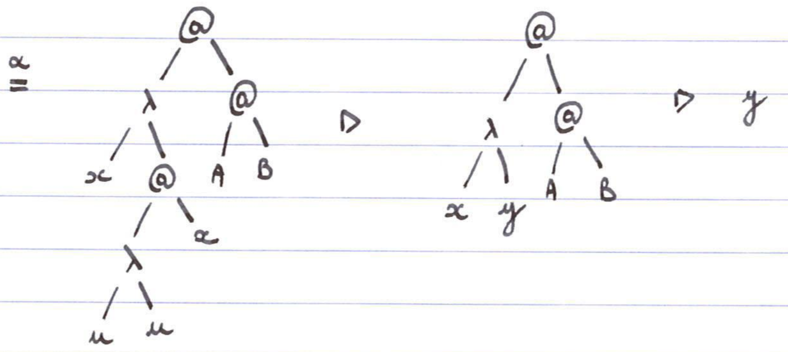


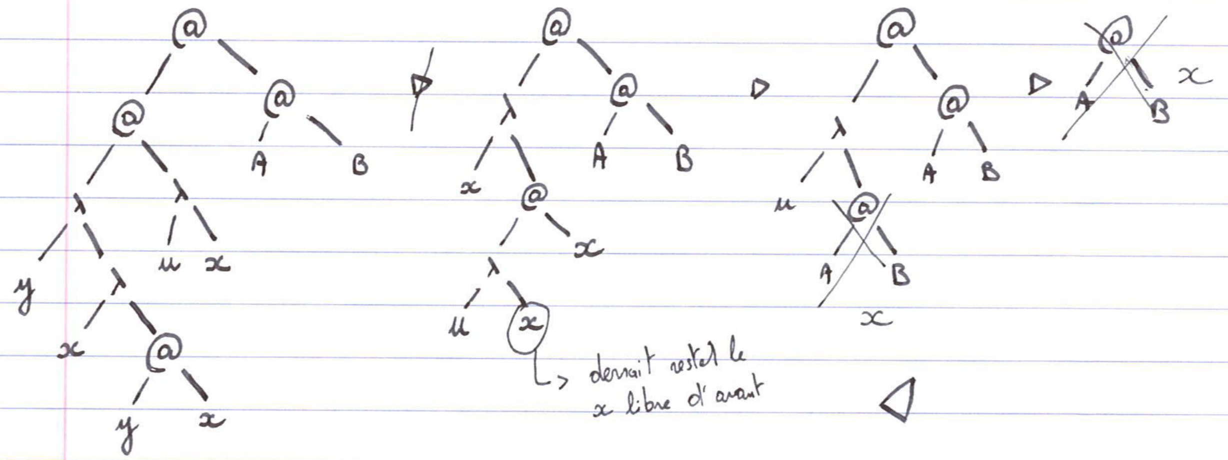
$\blacktriangleright$ $U[x:=V]$

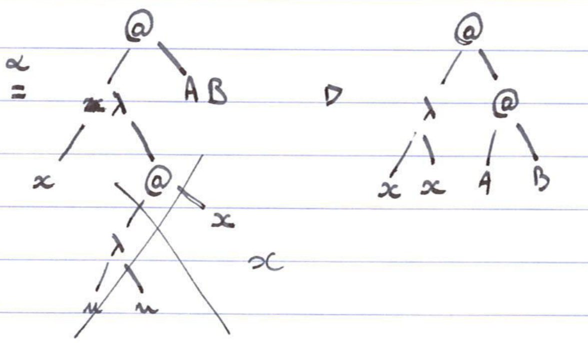
identique

$U_y[y:=V]$









Definition definitive de $U[x := V]$

① effectuer des α-conversions dans U pour remplacer toutes ses variables liées par des variables fraiches.

② remplacer les occurences libres de x dans U par V.

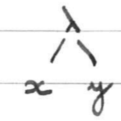
30/01/2018

λ-CALCUL

2 CM

introduction au typage.

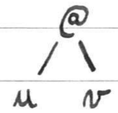
intuition :



x a un certain type A par α

u a un certain type B

Le type de $\lambda x\, u$ est alors $A \to B$



u doit être de type de la forme $A \to B$

V doit être de type de la forme A.

alors UV est de type B.

$$\frac{U : A \to B \qquad V : A}{UV : B}$$

$$\frac{x : A \qquad U : B}{\lambda x . u : A \to B}$$

$\lambda x^{mat} . x \qquad mat \to mat$

$\lambda x^{mat} . \lambda y^{mat} . x + y \quad : \quad mat \to (mat \to mat)$

$\qquad mat \to mat \to mat$

$\lambda f^{mat \to mat} . (\lambda x^{m} . f x) \quad : \quad (mat \to mat) \to (mat \to mat)$

$\qquad (mat \to mat) \to mat \to mat$

en coq : fun(f : mat -> mat) => fun (x . mat) => f x

23/01/2018

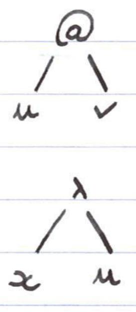
λ-calcul

CM

① **Rappel syntaxe.**

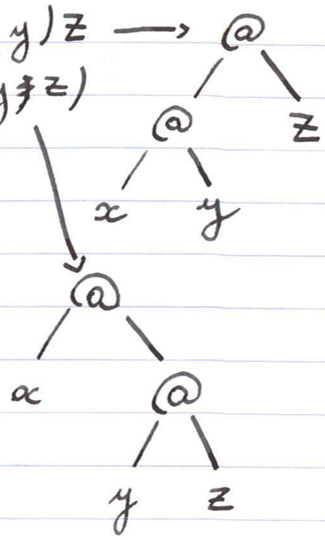
- variable x, u ...
- Application (uv)
- Abstraction $(\lambda x.\ u)$

@ : application de gauche avec les parenthèses à droite.



$(x\ y)z \longrightarrow$

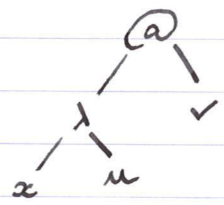
$\{x(y\}z)$



Réduction β-reduction

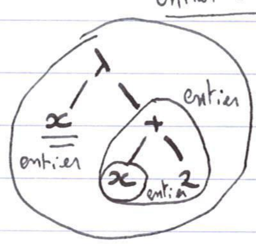
$(\lambda x.\ u)\,v$

$\triangleright\ u[x := v]$

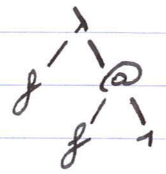


entier → entier

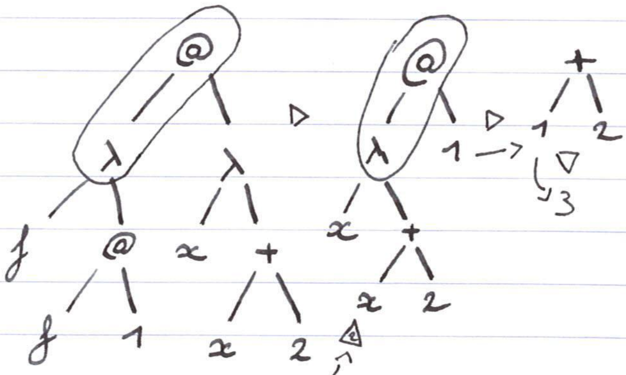
$\lambda x.\ x+2$



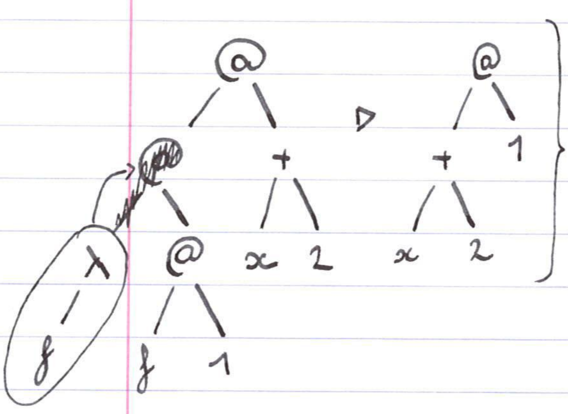
$\lambda f.(f\ 1)$

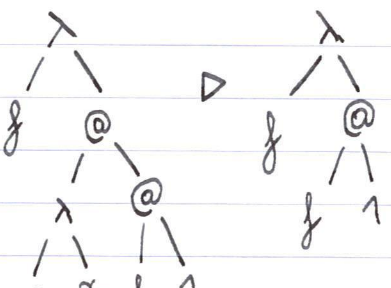


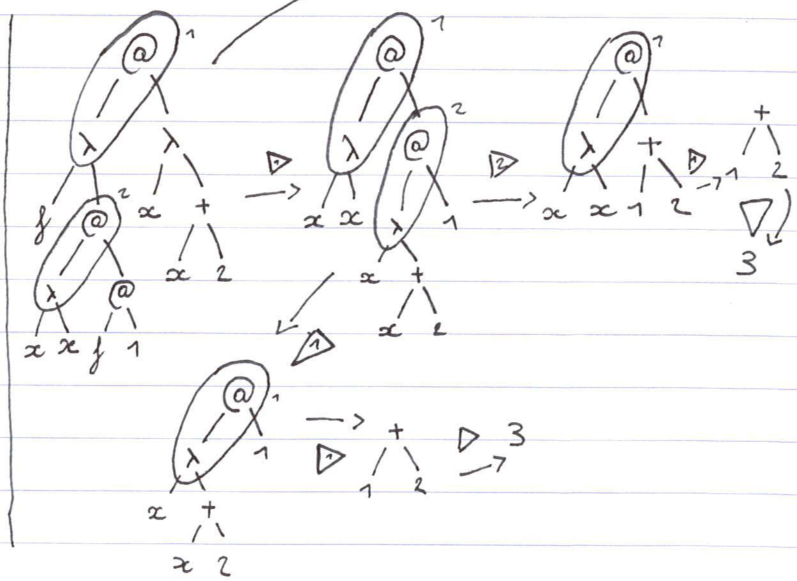
$(\lambda f.(f\ 1))(\lambda x.\ x+2)$



pas de sens dans calcul typé







$$(\lambda f.(\lambda x.x)(f\,1))(\lambda x.(x+2))$$

$$\triangleright\ (\lambda x.x)(f\,1)\,[f := \lambda x.x+2]$$

$$(\lambda x.x)((\lambda x.x+2)\,1)$$

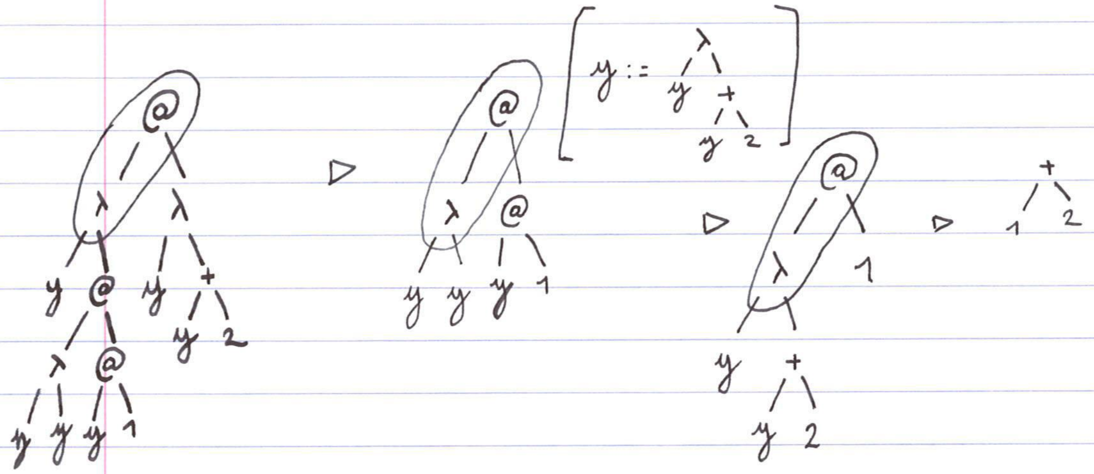
$$\triangleright\ (\lambda x.x)((x+2)\,[x:=1])$$

$$(\lambda x.x)(1+2)$$

$$\triangleright\ x\,[x:=1+2]$$

$$1+2$$

$$3$$



2 Abréviations

Donner la notation complètement parenthésée et non abrégée des λ-termes suivants :

1/ $(x\ y\ z)$

2/ $\lambda v.u\ x\ z$

3/ $\lambda vxy.u$

4/ $\lambda vtr.(u\ x\ z)$

5/ $\lambda x.(x\ (\lambda y.y\ x))$

6/ $(u\ x\ (y\ z)(\lambda v.v\ y))$

7/ $(\lambda xyz.(x\ z\ (y\ z))\ u\ v\ w)$

8/ $(w(\lambda xyz.(x\ z\ (y\ z))\ u\ v))$

3 Définitions de fonctions

Rendre à chaque terme sa définition en français.

1/ $\lambda x.x$

2/ $\lambda x.y$

3/ $\lambda x.\lambda v.x$

4/ $\lambda x.\lambda y.(x\ y)$

5/ $\lambda x.(x\ x)$

1/ Fonction qui, à deux arguments, associe le premier.

2/ Fonction identité.

3/ Fonction qui applique son argument à lui-même.

4/ Fonction qui, à deux arguments, associe l'application du premier au deuxième.

5/ Fonction qui, à un argument, associe la valeur d'une variable.

4 Variables libres / liées

Pour chaque λ-terme, donner les occurrences libres et liées, puis marquer la portée de chaque variable x.

1/ $(\lambda x.x\ y)$

2/ $(\lambda vxy.u\ v\ y)$

3/ $(\lambda x.(x\ (\lambda y.y\ x)))$

4/ $((\lambda xyz.x\ z\ (y\ z))\ u\ x\ w)$

5/ $(u\ x\ (y\ z)\ (\lambda x.x\ y))$

5 Redexes et Réduction

Souligner les redexes des λ-termes suivants puis les réduire jusqu'à leur forme normale.

1/ $(\lambda x.x\,x)\ (\lambda y.x)\ (\lambda y.x)$

2/ $(\lambda y.x)\ ((\lambda y.x)\ (\lambda y.x))$

3/ $(\lambda y.x\,(\lambda y.x)\ (\lambda y.x))$

4/ $(\lambda y.(\lambda y.x)\ (\lambda y.x)x)$

Modèles de Calcul [Lambda-Calcul]
Initiation, syntaxe & portées

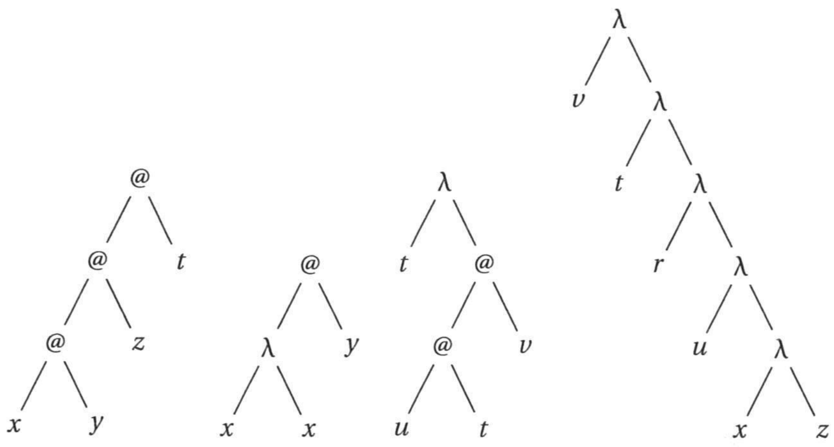
Pascal Fradet, Jean-François Monin, Catherine Parent-Vigouroux

Rappels

Soit V un ensemble infini dénombrable de variables. On définit inductivement l'ensemble Λ des λ-termes par $\Lambda := x|(\lambda x.\Lambda)|(\Lambda\Lambda)$ avec $x \in V$.

1 Arbres syntaxiques

1. Donner le λ-terme correspondant à chacun de ces arbres syntaxiques :



2. Pour chacun des termes suivants, donner l'arbre qui lui correspond :

1/ $(\lambda x.(\lambda v.x))$	**3/** $(x\ (y\ t))$	**5/** $((x\ y)\ (((z\ t)\ t)\ v))$
2/ $(\lambda x.(x\ y))$	**4/** $(\lambda v.(\lambda t.(\lambda r.((u\ x)\ z))))$	**6/** $(((\lambda v.(((u\ t)\ v)\ r))\ (y\ z))\ (\lambda x.x))$

3. Simplifier au maximum le parenthèsage des termes de cet exercice.

24/01/2018

λ-calcul

① TD

Lambda calcul - TD séance 1

Exercice 1 - Arbres syntaxiques

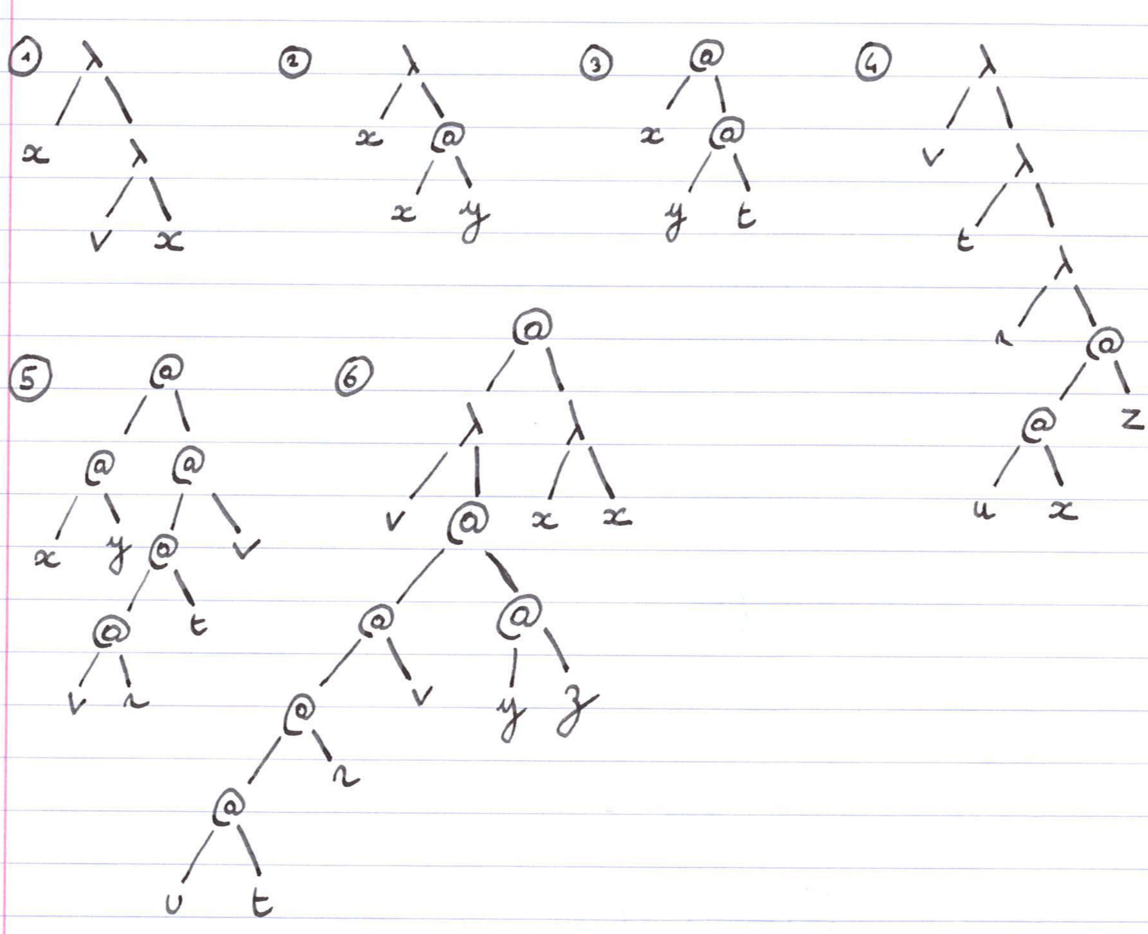
① $(((x\ y)\ z)\ t)$

② $((\lambda x.x)\ y)$

③ $\lambda t.(ut)\ v$

④ $\lambda v.(\lambda t.(\lambda r.(\lambda u.(\lambda x.z))))$

Exercice 2 -



Exercice 3

① $\lambda x.\lambda v.\lambda x$

② $\lambda x.x\ y$

③ $x(y\,t)$

④ $\lambda v.\lambda t.\lambda r.\,u\,x\,z$

⑤ $(x\ y)(z\,t\,t\,v)$

⑥ $(\lambda v.(u\,t\,v\,r)(y\ z))\ \lambda x.x$

Exercice 2 - Abreviation

① $(((x)\ y)\ z)$
② $((\lambda v.(u))(x\ z))$
③ $(\lambda v.(\lambda x.(\lambda y.u)))$
④ $(\lambda v.(\lambda t.(\lambda r.((u\ x)\ z))))$
⑤ $(\lambda x.(x\ (\lambda y.(y\ x))))$
⑥ $((u\ x)(y\ z)(\lambda v.(v\ y)))$
⑦ $(\lambda x.(\lambda y.(\lambda z.(((((x\ z)(y\ z))\ u)\ v)\ w))))$
⑧ $(w(\lambda x.(\lambda y.(\lambda z.((((x\ y)(y\ z))\ u)\ v)))))$

Exercice 3 - Definition de fonction

1 -> 2
2 -> 5
3 -> 1
4 -> 4
5 -> 3

Exercice 4 - Variables libres / liées

vert : libre
rouge : lié

Reponses non contractuelles

① $(\lambda x.x\ y)$
② $(\lambda x.(x\ (\lambda y.y\ x)))$
③ ~~$(\lambda x.(x\ (\lambda y.y\ x$~~$)))$
② $(\lambda v\ x\ y.u\ v\ y)$
④ $((\lambda x\ y\ z.x\ z\ (y\ z))\ u\ x\ w)$
⑤ $(u\ x\ (y\ z)\ (\lambda x.x\ y))$

24/01/2018

λ-calcul

2 TD

Exercice 5 - Redexes et reduction

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

16/01/2018

λ

① CM **Le λ-calcul traite des expressions:**

- Les expressions sont constitués de fonctions et de données : On applique les fonctions aux données. Les opérateurs sont considérés en tant que fonction.

Structure du cours

- Representation et manipulation du λ-calcul
- Codage en λ-calcul
- pratique du λ-calcul : Coq <-> coqide / proofgeneral / emacs
- Projet

Equipe pédagogique

- Pascal X
- J.F Monin

Plusieurs λ-calculs. (4)

λ-calcul pur(1)

λ-calcul avec constantes

λ-calcul typé(2)

λ-calcul avec constantes typées

type : aide à la conception

(1) : Contient toutes les fonctions recursives possible

(2) :

Definition du λ-calcul:

- Un programme
  - λ-expression
  - λ-terme
  - terme
- 3 constructions
  - variable "x", y, x_1, x'
    - coq : identificateurs long.:
  - application (d'une fonction à un argument)
  - abstraction λx U

    Coq -> fun x => U

Exemple : application de f à x s'écrit $f\ x$

On peut avoir des fonctions qui prennent des fonctions en argument, ainsi que des fonctions qui renvoient une fonction.

$\lambda x\ u$:

- x : paramètre de la fonction
- u : corps de la fonction

} fonction identité

Evaluation (calcul), intuitivement :

② Comment évaluer un terme ?

① Quand évaluer -> quand on a une abstraction appliquée à un argument.

<u>Exemple</u> : $(\lambda x . x)(g\ y) \triangleright g\ y$

$$(\lambda x . (\dots) x \dots x \dots)(g\ y) \triangleright (\dots)(g\ y) \dots (g\ y) \dots$$

Etapes de réduction :

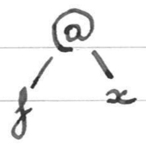
Soient U et V deux termes

$(\lambda x . U)\ V$ se réduit en U où tout les x ont été remplacés par V

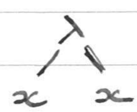
$$(\lambda x . U)\ V \triangleright U \underbrace{[x := V]}_{\text{substitution}}$$

presentation du λ-calcul avec des arbres

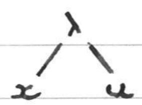
- application $f x$



- $\lambda x\ x$,

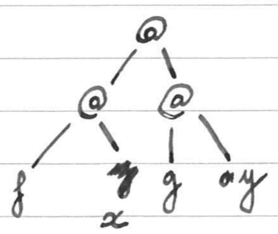


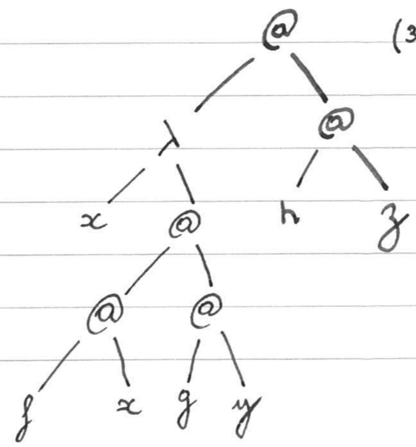
- abstraction $\lambda\ x, u$



- $(\lambda x . (f x)(g\ y))(h\ z)$

- $(f x)\ (g\ y)$





(3)

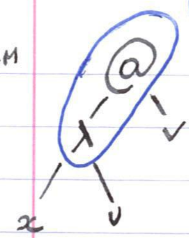
16/01/2018

λ

②

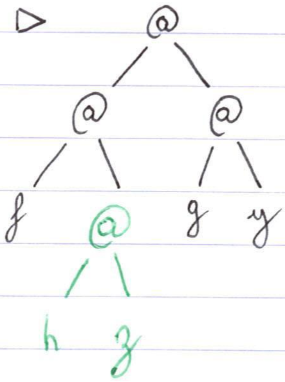
CM

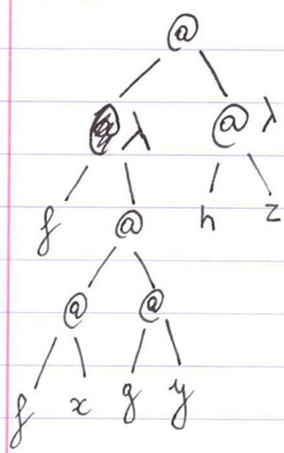
REDEX



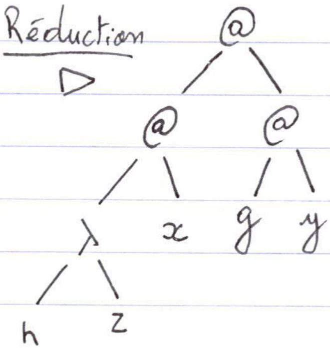
▷ U[x:=v]

(3) ▷





Réduction ▷



▷