Proofs and Programs

Phillipe Audebaud [*]
ENS de Lyon

Contents

[*] https://perso.ens-lyon.fr/philippe.audebaud/PnP/

Basis

- Lecture: Tue 8h-10h (Philippe Audebaud)
- Tutorial: We 8h-10h (Aurore)

10 Weeks of courses (3x3), which is really low.

$$Final\ mark = 50\% \cdot CC + 50\% \cdot Exam$$

No mid-time exam, but weekly homework.

Warning Presence at the courses and tutorial will have an impact on the marks.

Prerequisites

- L2.2 $\rightarrow$ Logical (Natacha P., Chapter 1 & 2):
  - Proof theory
  - Formal system for logic inference.
- λ-calculus
- Category theory

Part I
(Pure) λ-Calculus

0.1 Computing with functions ?

How do we do mathematics ?

A. Having *structures:* numbers, spaces (points, vectors, functions) $\rightarrow$ Eilenberg-Mac Lane ($\sim$ 1942) Category theory

B. Build, explore, transform structures $\rightarrow$ Church ($\sim$ 1930) λ-*Calculus*

C. Compare "stuff": *equality* $\rightarrow$ Voevoski ($\sim$ 2006) Algebraic topology $\rightarrow$ search HoTT (Hight order Type Theory)

D. Provide a framework (*rules*) to reasoning on all that! $\rightarrow$ 1st point

0.2 Church λ-calculus (informally)

$$\begin{aligned} f :& A \rightarrow B \\ & x \mapsto e \end{aligned}$$

Given $a \in A$, $f(a)$ is the "replacement of the occurrence of x in e by a"

$$f \stackrel{\text{def}}{=} \lambda x.e \qquad (\lambda\text{-abstraction})$$
$$f\ a = (\lambda a.e)\ a \qquad (\text{Application})$$

Notation

$$e\langle a/x\rangle$$

is the replacement in e of all the occurrences of a by x.

Example

1. $$\lambda x.x$$
   $$x \mapsto x$$

   is the identity function

2. $$\lambda x.y$$
   $$x \mapsto y$$

   Here x and y are variables, $x \neq y$. $(\lambda x.y)\ a$ leads to $y\langle a/x\rangle \equiv y$

   $$(\lambda x.a)\ b \to_\beta a\langle b/x\rangle$$

   $\to_\beta$ is a binary relation on lambda-terms $\Rightarrow$ idea of computation on terms.

Notion of α-equivalence

$$\lambda x.a \stackrel{?}{=}_\alpha \lambda y.b$$

Pick a *fresh* variable, let say z,

$$a\langle z/x\rangle =_\alpha b\langle z/y\rangle$$

All the results and proofs will be done up to α-equivalence (no difference made between $\lambda x.x$ and $\lambda y.y$).

1 A toolbox on λ-calculus

λ-calculus: Syntax and Semantics, Herk Barendregt (1977)

Let $\mathcal{X}$ be a measurable set of variables, ranged over by $x, y, z, ...$

Definition 1. *A λ-term e is generated by the grammar:*

$$a, b, e... ::= x \in \mathcal{X} \mid \lambda x.e \mid a\ b$$

The set of λ-terms is denoted Λ.

Definition 2 (Free variable)**.** *The set of free variables in e, denoted $FV(e)$ is defined inductively:*

- *if* $e \equiv x \in \mathcal{X},\ FV(x) \equiv \{x\}$
- *if* $e \equiv \lambda x.a_0,\ FV(\lambda x.a_0) \equiv FV(a_0) \setminus \{x\}$
- *if* $e \equiv a_1\ a_2,\ FV(a_1\ a_2) \equiv FV(a_1) \cup FV(a_2)$

A term e is closed *if* $FV(e) = \emptyset$

Definition 3 (Substitution)**.** *Given $x \in \mathcal{X}$, $a \in \Lambda$, the substitution of (all the) occurrences of a in $e \in \Lambda$, denoted $e\langle a/x\rangle$ is:*

- *if* $y \in \mathcal{X} \setminus \{x\},\ y\langle a/x\rangle \equiv y$ *and* $x\langle a/x\rangle \equiv a$
- $(\lambda y.e)\langle a/x\rangle = \lambda y.e\langle a/x\rangle$
- $(e\ f)\langle a/x\rangle = (e\langle a/x\rangle)\ f\langle a/x\rangle$

Definition 4 ($\rightarrow_\beta$ reduction)**.**

$$\rightarrow_\beta \subseteq \Lambda \times \Lambda$$
$$\Big\{\big(\underbrace{(\lambda x.a)\ b}_{redex},\ \underbrace{a\langle b/x\rangle}_{contraction}\ \big)\Big|x \in \mathcal{X},\ a, v \in \Lambda\Big\}$$

Example

1.

$$(\underbrace{\lambda x.(\lambda y.y)\ a}_{\twoheadrightarrow_\beta (\lambda y.y)\ b})\ b) \rightarrow_\beta ((\lambda y.y)\ a)\ \langle b/x\rangle \equiv ((\lambda y.y)\ \langle b/x\rangle)a\langle b/x\rangle$$
$$\equiv (\lambda y.y)\ a\langle b/x\rangle$$

2.

$$(\lambda x.y)\ a \rightarrow_\beta y$$

3.

$$(\lambda x.x\ x)(\lambda x.x\ x) \rightarrow_\beta (x\ x)\langle \lambda x.x\ x/x\rangle \text{ or } (x\ x)\langle \lambda y.y\ y/x\rangle$$
$$(\lambda x.x\ x)(\lambda x.x\ x)$$

Russell paradox: we get an infinite β-reduction!

$$\rightarrow_\beta \subseteq\ \beta_0\ \subseteq \underbrace{\beta}_{\beta-\text{reduction}} = \beta_0^*$$
$$\rightarrow_\beta^* \text{ is the } \beta\text{-reduction, noted } \twoheadrightarrow_\beta$$

Definition 5 (β_0-contraction)**.** *Let* $a, b \in \Lambda$. $a\ \beta_0\ b$ *is defined by cases:*

- $x\ \beta_0\ x$
- $(\lambda x.u)v\ \beta_0\ u\langle v/x\rangle$
- $(\lambda x.u)\ \beta_0\ (\lambda x.v)$ *if* $u\ \beta_0\ v$
- $(u\ v)\ \beta_0\ (u'\ v)$ *if* $u\ \beta_0\ u'$
- $(u\ v)\ \beta_0\ (u\ v')$ *if* $v\ \beta_0\ v'$

Maintenant en français !

Remarque: β_0 est réflexive.

Definition 6. *La* β-réduction *est la clôture transitive de* β_0*:*

$$\beta = \beta_0^*$$

Remarque Si $a, b \in \Lambda$, alors $a \; \beta \; b$ si il existe $n \geq 0$ et $(e_k)_{0 \leq k \leq n}$ λ-termes tels que :

- $a = e_0$ et $b = e_n$
- pour tout $k < n$, $e_k \; \beta_0 \; e_{k+1}$

Definition 7. *Soit $\mathcal{R}$ une relation binaire sur Λ. On dit que $\mathcal{R}$ est* λ-compatible *si elle satisfait les propriétés suivantes :*

- $x \; \mathcal{R} \; x$
- *si $a \; \mathcal{R} \; b$ et $c \; \mathcal{R} \; d$ alors $a \, c \; \mathcal{R} \; b \, d$*
- *si $a \; \mathcal{R} \; b$ alors $\lambda x.a \; \mathcal{R} \; \lambda x.b$*

Propriety 1. *La β-réduction est la plus petite relation λ-compatible et transitive contenant $\rightarrow_\beta$*

Proof. $\star$ On vérifie d'abord :

$$\rightarrow_\beta \subseteq \beta_0 \subseteq \beta_0^* = \beta$$

D'autre part, β_0 est λ-compatible :

- par réflexivité, $x \; \beta \; x$
- soit $a \; \beta \; b$; par définition,il existe $n \geq 0$, $(e_k)_{0 \leq k \leq n}$ tel que $a = e_0$, $b = e_n$ et pour tout $k < n$, $e_k \; \beta_0 \; e_{k+1}$. Du coup, par définition de β_0, pour tout $k < n$, $\lambda x.e_k \; \beta_0 \; \lambda x.e_{k+1}$.

  Ainsi, $\lambda x.a \; \beta \; \lambda x.b$.

$\star$ Soit $\mathcal{R}$ une autre relation λ-compatible et transitive contenant $\rightarrow_\beta$. Montrons que $\beta \subseteq \mathcal{R}$. Il "suffit" de vérifier que $\beta_0 \subseteq \mathcal{R}$ (laissé en exercice). ☐

Propriétés essentielles de la β-réduction

Remarque (Λ, β_0) est un *système de réduction abstrait*[1].

Definition 8 (Forme normale, Relation normalisante)**.** *Soit $\mathcal{R}$ une relation binaire sur Λ,*

- *On dit que a* est *une forme normale (relativement à $\mathcal{R}$) s'il n'existe pas $b \in \Lambda$ tel que $a \; \mathcal{R} \; b$.*
- *On dit que a* a *une forme normale (relativement à $\mathcal{R}$) s'il existe $b \in \Lambda$ tel que b* est *une forme normale et $a \; \mathcal{R}^* \; b$*
- *On dit que $\mathcal{R}$ est normalisante si tout $a \in \Lambda$ a une forme normale*

Exemple

- $\lambda x.x$ est une forme normale relativement à β_0
- β_0 n'est pas normalisante !

$$\Omega \equiv (\lambda x.x \; x) \; (\lambda x.x \; x)$$
$$\Omega \rightarrow_\beta \Omega$$

Definition 9 (Confluence)**.** *Soit $\mathcal{R}$ une relation binaire sur Λ. On dit que $\mathcal{R}$ est confluente si pour tout $(a, b, c) \in \Lambda^3$ tel que*

$$a \; \mathcal{R}^* \; b \text{ et } a \; \mathcal{R}^* \; c$$

alors il existe $d \in \Lambda$, tel que

$$b \; \mathcal{R}^* \; d \text{ et } c \; \mathcal{R}^* \; d$$

Theorem 1. *La β_0-réduction est confluente.*

Proof. En semaine 3 ou 4. ☐

Corollary 1. *Tout λ-terme admet au plus une forme normale, relativement à β_0*

[1] Cf ThPr

Notion d'égalité sur les λ-termes

Definition 10. *La β-équivalence sur Λ est la relation binaire notée $=_\beta$, définie comme la clôture réflexive symétrique transitive de β_0 :*
$a =_\beta b$ s'il existe $n \geq 0$ et $(e_k)_{0 \leq k \leq n}$ tel que $a = e_0$ et $b = e_n$ et $\forall k < n$, soit $e_k\ \beta_0\ e_{k+1}$ soit $e_{k+1}\ \beta_0\ e_k$

Definition 11 (λ-congruence)**.** *Une relation binaire $\mathcal{R}$ (sur Λ) est une λ-congruence si c'est une relation d'équivalence et qu'elle est λ-compatible.*

Theorem 2 (Church-Rosser)**.** *Pour tout $(a, b) \in \Lambda^2$, $a =_\beta b$ si et seulement si il existe $c \in \Lambda$ tel que $a\ \beta\ b$ et $b\ \beta\ c$*

Proof. La condition est suffisante

Réciproquement, pour la condition nécessaire, on introduit $R \subseteq \Lambda \times A$ défini par :
$a\ \mathcal{R}\ b$ s'il existe c tel que $a\ \beta\ c$ et $b\ \beta\ c$.
On remarque, par définition de $\mathcal{R}$,

- $\mathcal{R}$ est réflexive et symétrique
- $\mathcal{R}$ est transitive

De plus, $\mathcal{R}$ contient β (ou β_0). Donc, si $a =_\beta b$, alors $a\ R\ b$. □

Theorem 3. *La relation d'équivalence $=_\beta$ est la plus petite λ-congruence contenant $\rightarrow_\beta$*

Proof. En exo. □

Notation On note $\equiv$ pour une définition ($\stackrel{\text{def}}{=}$), mais aussi pour l'α-équivalence ($=_\alpha$). On peut utiliser la notation de Bruijn (cf références).

2 Calcul propositionnel et correspondance de Curry-Howard

2.1 Éléments de langage (informels)

- Théorie de la *démonstration* (prouvabilité)
- Thème des modèles

Quelques "ingrédients" :

- *énoncés* (logiques) : ici les familles du calcul propositionnel:
$$A ::= x \mid \top \mid \bot \mid A \Rightarrow B \mid A \wedge B \mid A \vee B \mid \neg A^2 \qquad (\star)$$
La notation "A propriété" signifie que A est engendrée par la grammaire ($\star$)
- On parle de *jugements* sur ces énoncés : "A true"
- On introduit aussi des jugements hypothétiques : A_1 true, A_2 true, ... ,A_n true $\vdash B$ true

Commentaires sur les différentes règles de (NJ):

- Le vrai
- L'implication (/!\: $A \Rightarrow B \neq \neg A \vee B$ dans (NJ))
- Le faux
- La négation
- La disjonction

[2] $\neg A$ signifie en fait $A \Rightarrow \bot$

2.2 Fragments $\lambda_\rightarrow$

On peut associer des règles au typages de λ-termes en raisonnant sur $\lambda x.t : T$

Theorem 4 (Curry-Howard)**.** *Le fragment $NJ_\rightarrow$ et $\lambda_\rightarrow$ sont en correspondance via:*

1. Si $\Delta \vdash t : T$ dans $\lambda_\rightarrow$

$\lambda_\rightarrow$	$NJ_\rightarrow$
type	*proposition*
variable de type type flèche	proposition atomique implication
terme	*dèrivation*
variable de terme λ-abstraction application β-redex β-réduction	hypothèse règle d'introduction règle d'élimination coupure transformation sur les dérivations
forme normale	*dérivation sans coupure*

Figure 1: Correspondance de Curry-Howard

2.3 Interprétation BHK

L'interprétation de Brouwer-Heyting-Kolmogorov consiste à construire un témoin (une preuve) d'une proposition selon le protocole suivant :

- Un témoin pour $A \wedge B$ est une paire formée par un témoin pour A et un témoin pour B
- Il y a un témoin unique pour $\top$
- Un témoin pour $A \vee B$ est soit un témoin pour A, soit un témoin pour B
- Il n'y a pas de témoin pour $\bot$
- Un témoin pour $A \Rightarrow B$ est une application de témoins pour A vers des témoins pour B
- Un témoin pour $\neg A$ est un témoin de $A \Rightarrow \bot$

Avec A, B engendrés par la grammaire

$$A ::= X \mid A \Rightarrow A \mid A \vee A \mid A \wedge A \mid \top \mid \bot$$

Definition 12 (Produit (paire))**.** *Soit A,B. Le produit de A par B est le damier de $A \times B$, et de la propriété universelle suivante :*
Pour tout $f : D \to A$ et $g : D \to B$, il existe $h : D \to A \times B$ tel que $\pi_1 \circ h = f$ et $\pi_2 \circ h = g$[3]*.*

De plus, h est unique *et ne dépend que de f et de g,*

$$h = \langle f, g \rangle$$

[3]Ces égalité correspondent à des β-réduction dans le λ-calcul

Par ailleurs, si $e : D \to A \times B$, alors
$$\begin{cases} \pi_1 \circ e : D \to A \\ \pi_2 \circ e : D \to B \end{cases}$$
Pour ce couple, il existe $\langle \pi_1 \circ e, \pi_2 \circ e \rangle : D \to A \times B$

Du coup, par unicité, on a nécessairement
$$\langle \pi_1 \circ e, \pi_2 \circ e \rangle = e$$

Cette observation donne lieu à :

- une transformation sur les dérivations
- une autre forme de réduction sur les λ-termes

On parle alors d'η-réduction.

On rajoute alors les règles de typage du produit $(\times_i)$ et $(\times_{E,k})$ pour $k \in \{1, 2\}$.

Definition 13 (Somme (coproduit))**.** *Soit A, B. C'est la donné de $A + B$ avec la propriété universelle suivante :*
Si $f : A \to C$, et $g : B \to C$, il existe $k : A + B \to C$ unique, ne dépendant que de f et de g noté $\{f, g\}$, tel que
$$\begin{cases} k \circ in_l = f \\ k \circ in_r = g \end{cases}$$

Par ailleurs, si on se donne
$$e : A + B \to C$$
Alors
$$\begin{cases} e \circ in_l : A \to C \\ e \circ in_r : B \to C \end{cases}$$
Donc
$$\{e \circ \in_l, e \circ in_r\} = e$$

On rajoute alors trois règles : (+Ig), (+Id) et (+E)

3 λ-calcul simplement typé

3.1 Quelques Lemmes

Lemma 1. *Si $\Delta \vdash t : T$ clos, $FV(t) \subseteq FV(\Delta)$, où $FV(\emptyset) = \emptyset$, et $FV(\Delta, x : S) = FV(\Delta) \cup \{x\}$.*

Attention *: Un contexte de typage $\Delta \equiv x_1 : S, ..., x_p : S_p$ où $p \geq 0$ est valide si les variables $x_1, ..., x_p$ sont distinctes deux à deux.*

On peut rajouter des règles sur la validité de Δ en tant que contexte.

$$\frac{}{\emptyset \text{ contexte valide}}$$

$$\frac{\Delta \text{ contexte valide} \qquad T \text{ type} \qquad x \notin FV(\Delta)}{\Delta, x : T \text{ contexte valide}}$$

Et on augmente (Hyp).

$$\frac{\Delta \text{ contexte valide} \qquad x : T \in \Delta}{\Delta, x : T \text{ contexte valide}} \text{(Hyp)}$$

Lemma 2 (Affaiblissement)**.** *Si $\Delta \vdash t : T$ et si $\Delta \subseteq \Delta'$, avec Δ' contexte valide, alors $\Delta' \vdash t : T$.*

Proof. Par induction sur la dérivation principale, c'est-à-dire $\Delta \vdash t : T$. Le seul cas "délicat" est lorsque

$$\frac{\Delta, x : U \vdash a : V}{} \rightarrow_I$$

□

Theorem 5. *Si $\Delta \vdash t : T$, alor t est fortement normalisant.*

Proof. Deux parties : poser la notation générale, puis l'adapter à $\rightarrow_\lambda$.

1. *Définition générale :* Si $e \in \Lambda$, $e \equiv \lambda\overline{x}.\Delta\overline{u}$ avec $|\overline{x}| > 0$, $|\overline{u}| > 0$ et $\Delta \in \mathcal{X}$ ou bien Δ est un β-redex
   - e est en forme normale si $\Delta \in \mathcal{X}$ et chaque u_i est en forme normale
   - e est une forme normale de tête (HNF) si $\Delta \in \mathcal{X}$
   - si e n'est pas en HNF, c'est-à-dire Δ est un β-redex, Δ est appelé *redex de tête.*

Definition 14. *$e \in \Lambda$ est fortement normalisant (SN) s'il n'existe pas de β-réduction infinie issue de e*

Exemple

- Ω n'est pas SN
- $(\lambda x.\lambda y.y\Omega$ n'est pas SN (il existe une dérivation infinie) $\rightarrow$ *attention* : la β-équivalence n'est pas compatible avec la propriété d'être fortement normalisant.

Par contre, si $a =_\beta b$ et b a une NF (resp HNF), alors a a une NF (resp HNF)
De plus :

- Si e a une NF (resp HNF), $\lambda x.e$ a une NF (resp HNF)
- Si e est SN, $\lambda x.e$ est SN

Soit $\mathcal{N}$ l'ensemble des termes SN, et $\mathcal{N}_0 \equiv \{x\overline{u} \mid x\overline{u} \in \mathcal{N}\} \subseteq \mathcal{N}$

Notation

- $e \in \Lambda$, $Succ(e) = \{e' \in \Lambda \mid e\beta_0 e'\}$, et cet ensemble est *fini* (réduction à branchements fini)
- Lemme de Koenig : si un arbre est infini et que cet arbre est a branchement fini, alors il existe un chemin infini

Si $e \in \mathcal{N}$, $\bigcup_{p \geq 0} Succ^p(e)$ est *fini*, de sorte que la définition suivante est bien fondée :

Definition 15. *Pour $e \in \mathcal{N}$, $\ell(e)$ désigne la somme des longueurs des chemins de tout réduction issue de e.*

Lemma 3. *Sont immédiats :*

- *Si $e \in \mathcal{N}$, alors $\lambda x.e \in \mathcal{N}$*
- *Si de plus $e' \in \mathcal{N}$ et $e\beta e'$, alors $e' \in \mathcal{N}$*
- *Si $e \in \Lambda$ tel que $Succ(e) \subseteq \mathcal{N}$, alors $e \in \mathcal{N}$*

Proof. Pour le troisième point, soit $e \in \Lambda$ tel que $Succ(e) \subseteq \mathcal{N}$, pour tout $e' \in Succ(e)$, $\ell(e') < \ell(e) \rightarrow$ une récurrence simple sur $\ell(e)$ permet d'établir $\mathcal{P}(e) \equiv$ "$Succ(e) \subseteq \mathcal{N}$ implique $e \in \mathcal{N}$" □

□

Soit $\mathcal{N}_0 \equiv \{ \underbrace{x\ \bar{u}}_{(((x\ u_1)\ u_2)\ ...)\ u_n} \mid x\ \bar{u} \in \mathcal{N} \subseteq \mathcal{N}\}$

Lemma 4. *i) Si $e \in \mathcal{N}$, alors $\lambda x.e \in \mathcal{N}$*

ii) Si $e \in \mathcal{N}$, et $e\ \beta'\ e$, alors $e' \in \mathcal{N}$

iii) Si $e \in \Lambda$ tel que $Succ(e) \subseteq \mathcal{N}$, alors $e \in \mathcal{N}$

iv) Si $e \in \Lambda$ et $x \in \mathcal{V}$, $e\ x \in \mathcal{N}$ implique $e \in \mathcal{N}$

Lemma 5. *Si $b \in \mathcal{N}$ et $a\langle b/x\rangle\ \bar{u} \in \mathcal{N}$*

Proof. Par récurrence sir $l(b) + l(a\langle b/x\rangle\ \bar{u})$ Il suffit de montrer que $\underset{\beta_0}{\underset{\downarrow}{Suc}}((\lambda x.a\ b\ \bar{u}) \subseteq \mathcal{N}$ □

3.2 Parties saturées de Λ

Definition 16. *Soit $S \subseteq \Lambda$. Ont dit que S est* saturée *si elle satisfait les conditions suivantes:*

$$\mathcal{N}_0 \subseteq S \subseteq \mathcal{N} \tag{Sat 1}$$

$$Si\ e \in S\ et\ e\ \beta_0^*\ e' \tag{Sat 2}$$

$$Si\ e \in \Lambda\ et\ e\ n'est\ pas\ une\ \lambda\text{-}abstraction,\ et\ si\ Succ(e) \subseteq S, e \in S \tag{Sat 3}$$

Propriety 2. *i) $\mathcal{N}_0$ est saturée*
ii) $\mathcal{N}$ est saturée
iii) Si X, Y sont des parties saturées, alors $X \to Y = \{e \in \Lambda \mid \forall a \in X, e\ a \in Y\}$

Proof. i) En exercice

ii) Il suffit de vérifier (Sat 3) En clair, soit $e \in \Lambda$ qui n'est pas une λ-abstraction, et tel que $Succ(e) \in \mathcal{N}$. On procède par induction structurelle sur e.

* $e \in \mathcal{X}$ trivial
* $e \equiv e_0\ a$, avec $Succ(e_0\ a) \in \mathcal{N}$

$$\left.\begin{array}{r} e\ \beta_0\ e_0'\ a,\ \text{avec}\ e_0\ \beta_0\ e_0' \\ e\ \beta_0\ e_0\ a',\ \text{avec}\ a\ \beta_0\ a' \\ e\ \beta_0\ e_1\langle a/x\rangle,\ \text{si}\ e_0 \equiv \lambda x.e_1 \end{array}\right] \begin{array}{l} e_0'\ a \in \mathcal{N} \\ e_0\ a' \in \mathcal{N} \\ e_1\langle a/x\rangle \in \mathcal{N} \end{array}$$

iii) $X, Y \in Sat(\Lambda)$, montrons que $X \to Y \in Sat(\Lambda)$
(Sat 1)

* $\mathcal{N}_0 \subseteq X \to Y$?
  Si $x\ \bar{u} \in \mathcal{N}$, et $a \in X \subseteq \mathcal{N}$
* $X \to Y \subseteq \mathcal{N}$
  Si $e \in X \to Y$, c'est-à-dire pour tout $a \in X$, $e\ a \in Y \subseteq \mathcal{N}$

(Sat 2) facile : $e \in X \to Y$ et $e\ \beta_0\ e'$
Si $a \in X$, $(e\ a)\ \beta_0\ (e'\ a)$ donc $e'\ a \in Suc(e\ a)$. Par hypothèse, $e \in X \to Y$, c'est-à-dire $e\ a \in Y$. Par (Sat 2) appliqué à Y, $e'\ a \in Y$.

(Sat 3) Soit e qui n'est pas un λ-abstraction et tel que $Succ(e) \subseteq X \to Y$; montrons que $e \in X \to Y$. Cela revient à établir que pour tout $a \in X, e\ a \in Y$. On montre ça en appliquant (Sat 3) à Y, car $e\ a$ n'est pas un λ-abstraction.

Il suffit de vérifier que $Succ(e\ a) \subseteq Y$:

$$e\ a\ \beta_0\ e'\ a \text{ avec } e\ \beta_0\ e'$$
$$e\ a\ \beta_0\ e'\ a \text{ avec } a\ \beta_0\ a'$$

Et c'est tout ! On remarque qu'il suffit de faire une démonstration par récurrence sur $\ell(a) \rightarrow$ à faire ! □

3.3 Normalisation pour $\lambda_\rightarrow$

Theorem 6 (SN). *Si $\Delta \vdash_{\lambda_\rightarrow} e : T$, alors $e \in \mathcal{N}$*

Proof. * Une "interprétation" des types commes parties saturées
* Un lemme d'"étiquetage"
* Le théorème apparaît comme corollaire.

□

Definition 17. *Soit $\rho \in \mathcal{V} \rightarrow Sat(\Lambda)$. On définit par induction structurelle l'interprétation d'un type T selon ρ notée $[T]_\rho$:*

* *Si $T \in \mathcal{V}, [T]_\rho \equiv \rho(T)$*
* *Si $T \equiv U \rightarrow V$, $[T]_\rho \equiv [U]_\rho \rightarrow [V]_\rho$*

Proof. D'après la proposition précédente :

- ρ existe
- $[T]_\rho \in Suc(\Lambda)$

□

Remarque $[T]_\rho$ ne dépend que de $\rho \upharpoonright FT(T)$

Definition 18. *$x_1, ..., x_n \in \mathcal{X}$ et $t_1, ..., t_n \in \Lambda$. La substitution $\sigma \equiv \langle t1/x_1, ..., t_n/x_n\rangle$ donne lieu à $\sigma(t)$ oú $t \in \Lambda$:*

- *Si $t \equiv x_i \in \{x_1, ...x_n\}$, alors $\sigma(t) = t_i$*
- *Si $t \in \mathcal{X}\{x_1, ..., x_n\}, \sigma(t) \equiv t$*
- *Si $t \equiv \lambda x.t_0$, on peut supposer que $x \notin \{x_1, ..., x_n\} \cup \{\cup_{i \leq n} FV(t_i)\}$ et $\sigma(t) \equiv \lambda x.\sigma(t_0)$*
- *Si $t \equiv a\ b$, $\sigma(t) = \sigma(a)\ \sigma(b)$*

Propriety 3. *Soit ρ une interprétation des types et $\Delta \vdash_{\lambda_t o} t : T$*
Pour tout substitution σ de domaine $\subseteq FV(\Delta)$ telle que pour tout $x : S \in \Delta, \sigma(x) \in [S]_\rho$, on a:

$$\sigma(t) \in [T]_\rho$$

Proof. Par induction sur ka hauteur de la dérivation, le seul cas intéréssant est $\rightarrow_i$

$$\frac{\Delta, x : U \vdash a : V}{\Delta \vdash t : U \rightarrow V}$$

Où $t \equiv \lambda x.a$.

Soit σ de domaine $\subseteq FV(\Delta)$; montrons que si $\forall x : S \in \Delta, \sigma(x) \subseteq [S]_\rho$, alors $\sigma(\lambda x.a) \in [U \rightarrow V]_\rho = [U]_rho \rightarrow [V]_\rho$. Par définition, cela revient à montrer que pour tout $b \in [U]_\rho$, $\sigma(\lambda x.a)\ b \in [V]_\rho$. On peut faire en sorte que

$$x \notin dom(\Delta) \cup \{\bigcup_{y:S\in\Delta} FV(\sigma(y))\} \cup FV(b)$$

Du coup, $\sigma' \equiv \langle t_1/x_1, ..., t_n/x_n, b/x\rangle$ où $\sigma \equiv \langle t_1/x_1, ..., t_n/x_n\rangle$ avec $\{x_1, ..., x_n\} \subseteq dom(\Delta)$

Et σ' satisfait :

$$\sigma(\lambda x.a)\ b \equiv \sigma'(a)$$

σ' satisfait les hypothèses relatives à $\Delta, x : U$ car $\sigma'(x) = b \in [U]_\rho$. Du coup, par hypothèse d'induction,

$$\sigma'(a) \in [V]_\rho$$

Finalement,

$$(\lambda x.\sigma(a)\ b\ \beta_0\ \sigma'(a) \in [V]_\rho$$

En résumé, $\lambda x.\sigma(a) \equiv \sigma(\lambda x.a) \in [U \to V]_\rho$ □

Part II
Polymorphisme

4 Abstraction des types

Programmation	Théorie des types	Raisonnement
λ-calcul	$\lambda_\to$	NJ($\Rightarrow$)
λ-calcul enrichie	$\lambda_{\to,\times,\top\bot,...}$	NJ
	λ_μ ($\sim$ 1990)	NK
Calcul de combinateurs (S,K,I)		Système de Hilbert (1900)
*	Système F	//
//	++	//

Figure 2: Correspondance programmation - langage de preuve

4.1 Motivations

Prenons quelques exemples:

1. L'identité :
   Dans $\lambda_\to$, $\vdash \lambda x.x : T \to T$ pour n'importe quel type T.
   On voudrait donner à $\lambda x.x$ un type "polymorphe"

2. Un entier de Church :
   $\bar{z} = \lambda x.\lambda f.f(fx)$. Dans $\lambda_\to$, $\vdash \lambda x.\lambda f.f(fx) : A \to (A \to A) \to A$
   $x : A, f : B \vdash f\ \underbrace{(f\ x)}_{:V} : C$

   (a) $f : b \equiv U \to V$ et $x : U$ donc $A \equiv U$
   (b) $B \equiv V \to W$ et $W \equiv C$ et $U \to V \equiv V \to W$ d'où $U \equiv V$ et $V \equiv W$
   (c) $\Delta \equiv \lambda x.x\ x$ qui nécessite $W = W \to V$.

Observations

⋆ On veut donner à un terme "plusieurs types" "d'un coup "

  En C $\to$ template

En O'Caml $\rightarrow$ `fun x -> x` : $'\alpha \rightarrow \alpha$

- $\star$ On peut avoir besoin de gérer plusieurs occurences d'une même variable
- $\star$ Observation de J.Reunolds ('74)
- $\star$ Introduit par JY Girard ('70) (cadre logique)

4.2 Le système F à la Church

Que veut-on ?

- $\star$ Une notion de généralisation (d'abstraction) sur les types

$$T ::= X \in \mathcal{V} \mid T \rightarrow T \mid \forall X.T$$

- $\star$ Une notion *d'instanciation* de type

$$\forall X.T \rightsquigarrow T\langle S/X\rangle$$

La conséquence sur les termes :

$$t ::= x \in \mathcal{X} \mid \lambda x^T.t \mid t\ t \mid \Lambda X.t \mid t\ T$$

(Hyp), $(\rightarrow_i)$[4] et $(\rightarrow_E)$ de $\lambda_\rightarrow$ plus :

$$\frac{\Delta \vdash t : T \qquad X \notin FT(\delta)}{\Delta \vdash \Lambda X.t : \forall X.T}\ (\forall_i) \qquad\qquad \frac{\Delta \vdash t : \forall X.T}{\Delta \vdash t\langle S/X\rangle : T\langle S/X\rangle}\ (\forall_e)$$

Retour sur les exemples

1. $\vdash \lambda x.x : X \rightarrow X$ devient $\Lambda X.\lambda x^X.x : \forall X\ X \rightarrow X$
2. $\bar{z} \equiv \lambda x.\lambda f.f\ (f\ x)$

$$\frac{\vdash \lambda x^X.\lambda f^{X\rightarrow X}.f\ (f\ x) : X \rightarrow (X \rightarrow X) \rightarrow X}{\vdash \Lambda X.\lambda x.\lambda f.f\ (f\ x) : \forall X.X \rightarrow (X \rightarrow X) \rightarrow X}$$

3. $\lambda x.x\ x$
   - $\star$ $x\ U : U \rightarrow U$
   - $\star$ $x\ V : V \rightarrow V$
     avec $V \rightarrow V \equiv U$

   On en déduit que $W \equiv V \rightarrow V$.

$$\frac{\dfrac{\dfrac{x\ U : U \rightarrow U}{x : \forall X.S \vdash x\ U : (V \rightarrow V) \rightarrow W} \qquad \dfrac{x\ V : V \rightarrow V}{x : \forall X.S \vdash x\ V : V \rightarrow V}}{x : \forall x : S \vdash x\ U\ (x\ V) : W^{V\rightarrow V}}}{\vdash \lambda x^{\forall X.S}.x\ U\ (X\ V) : (\forall X.S) \rightarrow W}$$

On trouve un type $((\forall X.(X \rightarrow X) \rightarrow \forall X.(X \rightarrow X)) \rightarrow ((\forall X.(X \rightarrow X) \rightarrow \forall X.(X \rightarrow X))$

4.2.1 Système F à la Curry

- $\star$ λ-term pur
- $\star$ Même types que dans le système à la Church
- $\star$ (Hyp), $(\rightarrow_i)$, $(\rightarrow_e)$

[4] où $\lambda x^S.t$ précise le type de x

⋆ Plus

$$\frac{\Delta \vdash t : T \qquad X \notin FT(\Delta)}{\Delta \vdash t : \forall X.T}\ (\forall_i) \qquad\qquad \frac{\Delta \vdash t : \forall X.T}{\Delta \vdash t : T\langle S/X\rangle}\ (\forall_e)$$

4.2.2 Aspects dynamiques

a. A la Church :

  ⋆ $(\lambda x^T.a)\ b \to_\beta a\langle b/x\rangle$

  ⋆ $(\Lambda X.t)\ S \to_\beta t\langle S/X\rangle$

b. A la Curry :

5 Propriétés méta du système F

	F-Church	F-Cury
Confluence (Church-Rosser)	×	Déjà fait
SR (Subject Reduction)	Facile	Plus dure
SN (Strong Normalisation)	En TD	A partir de $\lambda_\to$

Figure 3: Propriétés de confluence de systèmes à la Curry et à la Church

Lemma 6 (Dans les deux versions du système F). *i) Si $\Delta \vdash t : T$ et Δ' contexte de typage tel que $\Delta_{\restriction FV(t)} = \Delta_{\restriction FV(t}$, alors $\Delta' \vdash t : T$*

ii) Si $\Delta \vdash t : T$, alors pour tout S et pour tout $X \in \mathcal{V}$, $\Delta\langle S/X\rangle \vdash \underbrace{t\langle S/X\rangle}_{t} : T\langle S/X\rangle$

iii) Si $\Delta, x : S \vdash t : T$ et $\Delta \vdash s : S$ alors $\Delta \vdash t\langle s/x\rangle : T$

5.1 Propriétés de F-Church

Propriety 4 (Subject reduction). *Si $\Delta \vdash_{Church} t : T$ et $t \to t'$ alors $\Delta \vdash t' : T$*

Proof. En exercice.

Pour rappel, $\to$ est l'analogue de β_0:

$$\to_\beta \subseteq \to$$

également dans un terme. □

Propriety 5 (Strong Normalization). *Si $\Delta \vdash_{Church} t : T$, alors t est fortement normalisant.*

Proof. Fiche d'exercice (semaine 7), basée sur le résultat pour $\lambda_\to$ □

Propriety 6 (Confluence faible). *Si $\Delta \vdash_{Church} t : T$ et si $t \to t_1$, $t \to t_2$, il existe t_3 tel que $t_1 \to^* t_3$ et $t_1 \to^* t_3$.*

Corollary 2 (Confluence). *La relation $\to$ est confluente sur les termes bien typés.*

Proof. On s'appuie sur SN et la confluence faible. □

5.2 Propriétés de F-Curry

On veut établir le résultat suivant:

Propriety 7. *Si* $\Delta \vdash_{Curry} t : T$ *et* $t \underset{\beta_0}{\twoheadrightarrow} t'$, *alors* $\Delta \vdash t' : T$.

Le résultat est vrai, mais sa démonstration nécessite des détours...

5.2.1 Système alternatif

On introduit la notion de *séquence de types*.

$$\Delta \rightarrow_n T_0, ..., T_n \ \text{ seq}$$

avec les règles suivantes :

$$\text{(Ax)}\ \frac{}{\Delta \vdash_0 T \text{ seq}} \qquad (\rightarrow_i)\ \frac{\Delta, x : S \vdash_n t : T}{\Delta \vdash_0 \lambda x.t : S \rightarrow T}$$

$$\text{(Gen)}\ \frac{\Delta \rightarrow_n T_0, ..., T_n \ \text{ seq} \qquad X \notin FT(\Delta)}{\Delta \vdash_{n+1} T_0, ..., T_n, \forall X.T_n \text{ seq}}$$

$$(\rightarrow_e)\ \frac{\Delta \vdash_p e : S \rightarrow T \qquad \Delta \vdash_q s : S}{\Delta \vdash_0 e\ s : T}$$

$$\text{(Inst)}\ \frac{\Delta \rightarrow_n T_0, ..., T_n \ \text{ seq} \qquad T_n \equiv \forall X.T}{\Delta \vdash_{n+1} T_0, ..., T_n, T\langle S/X\rangle \text{ seq}}$$

$$\text{(Hyp)}\ \frac{x : T \in \Delta}{\Delta \vdash_0 x : T} \qquad \text{(Sub)}\ \frac{\Delta \vdash_0 t : T \qquad \Delta \vdash_n T_0, ..., T_n \text{ seq}}{\Delta \vdash_n t : T_n}$$

Propriety 8. $\Delta \vdash_{Curry} t : T$ *si et seulement si il existe* $n \geq 0$ *tel que* $\Delta \vdash_n t : T$

Proof. Dans chacun des sens, par induction. □

Exo Si $\Delta \vdash_p t : T$ et $\Delta \vdash_p T_o, ..., T_q$ seq avec $T_0 \equiv T$, alors

$$\Delta \rightarrow_{p+q-1} t : T_q$$

Exo Si $\Delta \rightarrow_n T_0, ..., T_n$ seq avec $T_0 \equiv U \rightarrow V$ et $T_n \equiv A \rightarrow B$ alors il existe $\bar{X} \subseteq \mathcal{V}$ et $\bar{S}$ de même longueur tel que

$$A \equiv U\langle \bar{S}/\bar{X}\rangle, B \equiv V\langle \bar{S}/\bar{X}\rangle$$

Lemma 7. *i) Si* $\Delta \vdash_n t : T$, *alors* $\Delta\langle S/X\rangle \vdash_n t : TSX$

ii) Si $\Delta, x : S \vdash_n t : T$ *et* $\Delta \rightarrow_r s : S$ *alors il existe* $n' \geq 0$ *tel que*

$$\Delta \vdash_{n'} t\langle s/x\rangle : T$$

Proof. i) facile

ii) Considérons le cas où t est une variable. Soit:

$$\frac{}{\Delta, x : S \vdash_0 t : T}$$

- Si $t \equiv x$, $S \equiv T$
  Par ailleurs, $\Delta \vdash_r s : T$, du coup $\Delta \vdash_r t\langle s/x\rangle : T$
- Si $t \not\equiv x$, $t : T \in \Delta$ et

$$\text{(Hyp)}\ \frac{x : T \in \Delta}{\Delta \vdash_0 x : T}$$

Soit $n \geq 0$:

$$\frac{\Delta \vdash_0 t : T_0 \qquad \Delta \vdash_n T_0, ..., T_n \text{ seq}}{\Delta \vdash_n t : T_n}$$

Par hypothèse d'induction, il existe $n'' \geq 0$ tel que $\Delta \rightarrow_{n''} t : T_0$. On en conclut $\Delta \vdash_{n''+n-1} t : T_n$ □

→ La démonstration "séquence" est différente !

Lemma 8. *Si $\Delta \vdash_n (\lambda x.a)\ b : T$, alors il existe $n, \geq 0$ tel que $\Delta \vdash_{n'} a\langle b/x\rangle : T$*

Proof. On se limite ici au cas où $n = 0$:

$$\frac{\Delta \vdash_p \lambda x.a : S \rightarrow T \qquad \Delta \vdash_q b : S}{\Delta \vdash_0 (\lambda x.a)\ b : T}$$

$$\frac{\dfrac{\dfrac{\square}{\Delta, x : U \vdash_r a : V}}{\Delta \vdash_0 \lambda x.a : U \rightarrow V} \qquad \Delta \vdash_p F_0, ..., F_p \text{ seq}}{\Delta \vdash_p \lambda x.a : S \rightarrow T}$$

Avec $F_0 \equiv U \rightarrow V$ et $F_p \equiv S \rightarrow T$, donc il existe $\bar{X}, \bar{S}$ tels que $S \equiv U\langle\bar{S}/\bar{X}\rangle$ et $T \equiv \langle\bar{S}/\bar{X}\rangle$ et $\bar{X} \cap FT(\Delta) = \emptyset$.

Par ailleurs,

$$\frac{\Delta \vdash_o b : S \qquad \Delta \vdash_q S_0, ..., S_q seq}{\Delta \vdash_q b : S \qquad (S_q \equiv S)}$$

On en déduit :

* $\Delta\langle\bar{S}/\bar{X}\rangle, x : U\langle\bar{S}/\bar{X}\rangle \vdash_r a : V\langle\bar{S}/\bar{X}\rangle$
  mais, puisque $\bar{X} \cap FV(\Delta) = \emptyset$

$$\Delta, x : S \vdash_r a : T$$

* Finalement,

$$\left.\begin{cases} \Delta, x : S \vdash_r a : T \\ \Delta \vdash_q b : S \end{cases}\right\} \Rightarrow \Delta \vdash_{r'} a\langle b/x\rangle : T$$

Pour un certain $r' \geq 0$ □

Propriety 9 (SR)**.** *Si $\Delta \vdash_n t : T$ et si $t\ \beta_0\ t'$, alors il existe $n' \geq 0$ tel que $\Delta \vdash_{n'} t' : T$*

Proof. On a vu le cas de base:

* $\rightarrow_\beta \subseteq \beta_0$
* On regarde les autres cas qui définissent β_0

□

Theorem 7 (SR)**.** *Si $\Delta \vdash_{Curry} t : T$ et si $t\ \beta_0\ t'$, alors $\Delta \vdash_{Curry} t' : T$*

Remarque Dans (SU)[5], on utilise $\sigma \preceq \tau$ avec σ, τ types.

Part III
Égalité

0 Usage des inductifs dans Coq

0.1 Les booléens

$\mathbf{bool} \equiv \{\mathbf{true}, \mathbf{false}\}$

En Coq,

```
Inductive bool : Type :=
| true                                                    (⋆)
| false
```

correspond à la déclaration de **true** et **false** comme éléments de type **bool**.

Ce qui "correspond" à l'énoncé d'un principe d'induction sur **bool** :

Pour tout $P : \mathbf{bool} \to \mathbf{True}$, si $\begin{cases} (P\ \mathbf{true}) \text{ est habitable} \\ (P\ \mathbf{false}) \text{ est habitable} \end{cases}$, alors pour tout b **bool**, $P\ b$ est habitable.

Dans Coq, $(\star)$ engendre:

- un nouveau type **bool**
- Les constructeurs **true** et **false**
- Le principle d'induction, de type $\Pi\ P : \mathbf{bool} \to Type$

0.2 Les entiers de Peano

```
 nat : Type
0 : nat
S : nat → nat

   Inductive nat : Type :=
| 0 : nat
| S : nat → nat
```

Ce qui correspond à l'introduction d'un entier, et au principe de récurrence "standard".

0.3 Le produit $A \times B$

```
 Inductive prod (A B : Type) : Type :=
pair : A → B → prod A B


   Inductive sum (A B : Type) : Type :=
| inl : A → sum A B
| inr : B → sum A B
```

[5]Bouquin de référence

1 Types dépendants

$\rightarrow (\Pi\ x : A)\ B(x)$ "$\forall x \in A.B(x)$"
$\rightarrow (\Sigma\ x.A)\ B(X)$ "$\exists x \in A.B(x)$"

2 Egalité (prélude)

Dans Coq : égalité de Leibniz.
$a, b : A$
"$a = b$" si pour tout $P : A \rightarrow Type$, $P\ a \rightarrow P\ b$.

3 Définition inductive

Idée $x, y : A \vdash Id_a(x, y) : Type$ "$x =_A y$" d'où la règle de formation :

$$\frac{A : Type \qquad a : A \qquad b : A}{Id_A(a, b) : Type}\ \text{(Id-form)} \qquad\qquad \frac{a : A}{refl_a : Id_A(a, a)}\ \text{(Id-intro)}$$

Et rien d'autre ! Dans Coq:

```
Inductive Id (A : Type) :  A → A → Type :=
IA_reflexive :  forall x:A Id A x x
```

$$\frac{\text{C : }(\Pi\ \text{x,y : A})(\Pi\ \text{p : Id}_A\ \text{x y})\ \text{Type} \qquad \text{x : A} \vdash \text{c : C x x }(\text{refl}_x) \qquad \text{a : A} \qquad \text{b : A} \qquad \text{p : Id}_A\ \text{a b}}{\text{Ind}_{Id_A}\ \text{C }(\lambda\text{x.c})\text{ a b p : C a b p}}$$