Travaux encadrés de Recherche
Théories des types et paradoxe de Girard

Jad Koleilat, sous la direction de Christine Paulin-Mohring

Mai 2022

Université Paris-Saclay

Je remercie Mme Paulin-Mohring pour l'opportunité qu'elle m'a donnée et pour le temps qu'elle m'a consacré.

Je remercie aussi mon ami et camarade Arthur Dallemagne pour sa relecture attentive.

Table des matières

Introduction

Le paradoxe de Russell est un des paradoxes les plus connus de la théorie des ensembles. Il apparaît lorsque l'on s'autorise trop de liberté dans la manière de définir de nouveaux ensembles. Ce paradoxe a un équivalent dans les théories des types : le paradoxe de Girard [**?**]. Le but de ce document est d'expliquer ce paradoxe à des lecteurs qui n'ont pas nécessairement de connaissances en logique. Ainsi, une grande partie du document est dédiée à la construction de structures et à l'explication de notions importantes pour la preuve du paradoxe. Une attention particulière a été donnée à l'explication intuitive des concepts abordés, car elle est cruciale à la compréhension des systèmes logiques que nous serons amenés à considérer.

On s'autorisera à utiliser le méta-symbole ":=", qui signifie que le membre de gauche sera utilisé comme raccourci pour écrire le membre de droite.

On rappelle les connecteurs logiques suivants ainsi que leur table de vérité :

- $\Rightarrow$ est le symbole logique pour "implique"
- $\wedge$ est le symbole logique pour "et"
- $\vee$ est le symbole logique pour "ou"
- $\neg$ est le symbole logique pour "non"
- $\bot$ est le symbole qui représente la proposition toujours fausse. Il se lit "bottom"

A	B	$A \wedge B$	$A \vee B$	$A \Rightarrow B$
Vrai	Vrai	Vrai	Vrai	Vrai
Vrai	Faux	Faux	Vrai	Faux
Faux	Vrai	Faux	Vrai	Vrai
Faux	Faux	Faux	Faux	Vrai

A	$\neg A$	$\bot$
Vrai	Faux	Faux
Faux	Vrai	Faux

On peut définir $\neg A := A \Rightarrow \bot$. Le lecteur peut s'en convaincre en utilisant les tables de vérité ci-dessus.

On rappelle que la convention de parenthésage pour $A \Rightarrow B \Rightarrow C$ est $A \Rightarrow (B \Rightarrow C)$.

1 Induction et relations biens fondées

1.1 Motivations

Cette section est consacrée à une généralisation de la notion de récurrence. En effet, il est possible de raisonner "par récurrence" sur des structures différentes de $\mathbb{N}$ et sans utiliser la notion classique de relation bien fondée. Les notions vues dans cette section sont indispensables à la compréhension de la suite de ce document.

1.2 Ensembles Inductifs

Définition 1.2.1 (Domaine d'une fonction partielle) *Soit $f : X \to Y$ une fonction partielle, $\mathcal{D}_f$ est l'ensemble des x dans X tel que $f(x)$ est défini.*
On appelle $\mathcal{D}_f$ le domaine de f.

Définition 1.2.2 *Soit X un ensemble, une définition inductive d'un sous-ensemble de X est :*

- *la donnée explicite d'une partie $B \subset X$, B est appelé l'ensemble de base.*
- *la donnée d'un ensemble de règles R (fini ou infini). Chaque règle $r_i \in R$ est une fonction $r_i : X^{n_i} \to X$ qui peut être partielle.*

On considère alors $\mathcal{F} \subset X$ le plus petit ensemble contenant B et stable par les règles de R. C'est-à-dire que $\forall r_i \in R, (x_1, \ldots, x_{n_i} \in \mathcal{D}_{r_i}) \wedge (x_1, \ldots, x_{n_i} \in \mathcal{F}) \Rightarrow (r_i(x_1, \ldots, x_{n_i}) \in \mathcal{F})$.

Théorème 1.2.3 (Point fixe) *Soit X un ensemble, $B \subset X$, R un ensemble de règles sur X, alors il existe un unique plus petit ensemble $\mathcal{F}$ qui vérifie :*

(B) $B \subset \mathcal{F}$
(I) *$\mathcal{F}$ est stable par les éléments de R*

On dit alors que $\mathcal{F}$ est défini inductivement.

Démonstration 1.2.3 Soit E l'ensemble des parties de X vérifiant (B) et (I). E est non vide car X vérifie (B) et (I). On définit

$$\mathcal{F} := \bigcap_{e \in E} e$$

B est inclus dans tous les $e \in E$ donc $B \subset \mathcal{F}$. Soit une règle $r_i \in R$, soit $x_1, \ldots, x_{n_i}$ des éléments de $\mathcal{F}$ qui sont dans le domaine de r_i. On a que :

$$\forall e \in E,\ x_1, \ldots, x_{n_i} \in e \Rightarrow r_i(x_1, \ldots, x_{n_i}) \in e$$

car tout les e vérifient (I). Puisque $\mathcal{F}$ est l'intersection des $e \in E$, on a bien que $r_i(x_1, \ldots, x_{n_i}) \in \mathcal{F}$. $\mathcal{F}$ vérifie donc (B) et (I). Par construction, on a bien la minimalité de $\mathcal{F}$.

Théorème 1.2.4 (Preuve par induction) *Soit $\mathcal{F} \subset X$ un ensemble défini inductivement à partir d'un ensemble de base B et d'un ensemble de règles R. Soit P un prédicat sur X (P est une fonction qui prend un élément de X et qui renvoie vrai ou faux). Si les propriétés suivantes sont vérifiées :*

- *$P(x)$ est vrai pour tout $x \in B$*
- *P est héréditaire : pour chaque $r_i \in R$, soit $x_1, \ldots, x_{n_i} \in \mathcal{F}$ et $x_1, \ldots, x_{n_i} \in \mathcal{D}_{r_i}$ si $P(x_1), \ldots, P(x_{n_i})$ sont tous vrais alors $P(r_i(x_1, \ldots, x_{n_i}))$ est vrai.*

Alors P est vrai pour tout élément de $\mathcal{F}$.

Démonstration 1.2.4 Soit G l'ensemble des éléments x de X tel que $P(x)$ est vrai. B est inclus dans G et G est stable par les règles R. Par minimalité de $\mathcal{F}$ on a donc $\mathcal{F} \subset G$ donc P est vrai pour tout élément de $\mathcal{F}$.

Remarque Il est usuel de présenter les règles sous forme de règles d'inférence. Une règle d'inférence se présente comme une fraction avec au numérateur les hypothèses ainsi que les éventuelles conditions correspondant au domaine de la règle et au dénominateur la conclusion. Le nom de chaque règle est indiqué entre parenthèses à coté du trait de fraction. Un ensemble de règles d'inférence s'appelle un système d'inférence.

Exemple On va définir par des règles d'inférence l'ensemble des entiers pairs P.

$$(1)\ \frac{}{0 \in P} \qquad (2)\ \frac{n \in P}{(n+2) \in P}$$

La règle d'inférence (2) nous dit que si n est un entier pair alors $n+2$ est aussi un entier pair. La règle d'inférence (1) nous dit que 0 est un entier pair. La règle d'inférence (1) correspond à se donner un ensemble de base $B := 0$ et la (2) une règle $r : n \mapsto n+2$. On considère alors le plus petit ensemble stable par ces règles qui est (sans surprise) l'ensemble des entiers pairs.

Définition 1.2.5 *Une relation binaire R sur un ensemble S est définie comme un sous-ensemble de S^2. Soit $x, y \in S$, on utilise alors la notation $x\,R\,y$ pour désigner que $(x, y) \in R$. On peut aussi voir une relation binaire comme une fonction qui va de $S \times S$ vers $0, 1$ (0 pour "Faux" et 1 pour "Vrai").*

Définition 1.2.6 (Clôture transitive) *Soit R une relation binaire sur un ensemble S, on appelle R' la clôture transitive de R définie par :*

$$(1)\ \frac{t\,R\,u}{t\,R'\,u} \qquad (2)\ \frac{u\,R'\,v \quad v\,R'\,t}{u\,R'\,t}$$

En regardant attentivement les règles, on peut remarquer qu'elles ne sont pas fonctionnelles. En effet, pour un élément u, il peut exister plusieurs éléments $t_1, t_2, \cdots \in S$ tels que $u\,R\,t_1$, $u\,R\,t_2$, ... Cela peut sembler problématique a priori car les règles sont des fonctions. En réalité, cela ne pose de problème. Une relation binaire est définie par un ensemble de couples. La règle (1) revient juste à se donner un ensemble de base $B \coloneqq (x, y) \in S^2 \mid x\,R\,y$. La règle (2) revient à se donner une fonction partiell $r : S^2 \times S^2 \to S^2$ définie par $r((x, y_1), (y_2, t)) = (x, t)$ si $y_1 = y_2$.

Exemple Soit R la relation binaire définie sur $\mathbb{N}$ par $(n\,R\,m) \coloneqq (n + 1 = m)$. On a que $2\,R\,3$ mais par contre $2\,R\,4$ est faux. Si on considère maintenant R', la règle (1) nous permet de déduire que $2\,R'\,3$ et $3\,R'\,4$. La règle (2) nous permet de déduire que $2\,R'\,4$. Ici, R' correspond à la relation d'ordre stricte usuelle sur les entiers.

Proposition 1.2.7 *Soit R une relation binaire, R' est la plus petite relation transitive qui contient R.*

Démonstration 1.2.7 Par définition, R' est transitive. Soit T une relation transitive qui contient R, alors T satisfait la règle (1) car elle contient R et la règle (2) car elle est transitive. R' étant la plus petite relation vérifiant (1) et (2), on a bien que R' est inclue dans T.

Définition 1.2.8 (Clôture transitive et réflexive) *Soit R une relation binaire sur un ensemble S, on appelle R^* la clôture transitive et réflexive de R définie par :*

$$(1)\ \frac{t\,R\,u}{t\,R^*\,u} \qquad (2)\ \frac{u\,R^*\,v \quad v\,R^*\,t}{u\,R^*\,t} \qquad (3)\ \frac{u \in S}{u\,R^*\,u}$$

Proposition 1.2.9 *Soit R une relation binaire, R^* est la plus petite relation réflexive et transitive qui contient R.*

On prouve cette proposition de la même manière que précédemment.

1.3 Relations biens fondées

Définition 1.3.1 (Tiers exclu) *Le tiers exclu est la propriété qui dit que pour toute proposition A, on a $A \vee \neg A$. C'est équivalent à $\neg\neg A \Rightarrow A$. Sans cette règle, il n'est pas possible de raisonner par l'absurde.*

Définition 1.3.2 (Accessibilité) *Soit S un ensemble, R une relation binaire sur S.* **acc** *est la plus petite relation unaire qui vérifie :*

$$(A)\ \forall x \in S,\ (\forall y \in S,\ y\,R\,x \Rightarrow \mathbf{acc}(y)) \Rightarrow \mathbf{acc}(x)$$

Si $x \in S$ vérifie **acc***, on dit alors que x est accessible (relativement à R).*

Définition 1.3.3 (Relation bien fondée) *On dit que R est bien fondée si tous les éléments de S sont accessibles.*

Il existe plusieurs définitions de relation bien fondée :

(1$*$) celle qu'on vient d'introduire

(2$*$) toute partie de S admet un élément R-minimale :
$\forall E \subset S,\ \exists m \in E,\ \forall e \in E,\ \neg(e\, R\, m)$

(3$*$) il n'existe pas de suite $(u_n)_{n\in\mathbb{N}} \in S^{\mathbb{N}}$ infiniment décroissante :
$\neg(\exists (u_n)_{n\in\mathbb{N}} \in S^{\mathbb{N}},\ \forall n \in \mathbb{N},\ u_{n+1}\, R\, u_n)$

En logique classique, ces trois définitions sont équivalentes. Dans la suite de ce document, on se plaçera dans des systèmes sans tiers exclu et sans axiome du choix. Ainsi, il n'y aura pas équivalence. On verra alors que la définition qui a été donnée est la plus appropriée.

Théorème 1.3.4 (Récurrence bien fondée) *Soit R une relation binaire bien fondée sur un ensemble S. Soit P une propriété telle que*

$$\forall x \in S, (\forall y \in S,\ y\, R\, x \Rightarrow P(y)) \Rightarrow P(x)$$

alors $\forall x \in S, P(x)$.

Démonstration 1.3.4 P satisfait (A) or **acc** est la plus petite relation qui satisfait (A) donc $\mathbf{acc} \subset P$. Par hypothèse, R est bien fondée donc **acc** est vrai pour tous les éléments de S donc P est vrai pour tous les éléments de S.

Cette notion de récurrence est plus appropriée que la notion usuelle de récurrence, dans le cadre qu'on verra à la section 5.

Théorème 1.3.5 ($(1*) \Rightarrow (3*)$) *Être bien fondé au sens* (1$*$) *implique qu'il n'existe pas de suite décroissante infinie.*

Démonstration 1.3.5 Soit R une relation bien fondée sur S un ensemble. Soit P la propriété, pour tout $x \in S$:

$$P(x) := \neg(\exists (u_n)_{n\in\mathbb{N}} \in S^{\mathbb{N}},\ (u_0\, R\, x) \wedge (\forall n \in \mathbb{N},\ u_{n+1}\, R\, u_n))$$

En language naturel : $P(x)$si et seulement si il n'existe pas de suite décroissante infinie à partir de x. Soit $x \in S$ tel que $\forall y \in S,\ y\, R\, x \Rightarrow P(y)$. On va montrer $P(x)$, ce qui nous permettra de conclure par récurrence bien fondée que $\forall x \in S,\ P(x)$. Supposons qu'il existe une suite $(u_n)_{n\in\mathbb{N}}$ infinie décroissante à partir de x. On en déduit qu'il existe une suite infinie décroissante à partir de u_0, or $u_0\, R\, x$. Donc, par hypothèse, on a $P(u_0)$. Contradiction, donc il n'existe pas de suite décroissante à partir de x. On vient donc de montrer que

$$\forall x \in S, (\forall y \in S,\ y\, R\, x \Rightarrow P(y)) \Rightarrow P(x)$$

Par récurrence bien fondée, on en déduit $\forall x \in S, P(x)$.

Remarque Dans cette démonstration, ni l'axiome du choix ni le tiers exclu n'interviennent. Ainsi, cette implication est vraie même dans des systèmes logiques sans axiome du choix ni tiers exclu (cadre dans lequel on se placera à la section 5).

2 Lambda-calcul

Une grande partie des définitions de cette section sont tirées du livre *Lambda-calcul types et modèles* de J.L. Krivine [**?**].

2.1 Motivations

Le but de cette section n'est pas d'étudier le λ-calcul en temps que sujet mais simplement d'en comprendre les bases car c'est un formalisme indispensable aux théories des types. Ainsi, on va énoncer des propriétés qui ne seront pas toutes démontrées. En effet, la preuve de beaucoup d'entre elles se fait par récurrence sur la structure des termes et n'apporte pas beaucoup à la compréhension globale. La partie 2.3 n'a pas besoin d'être lue en intégralité pour la compréhension du reste de ce document. Il faut cependant avoir compris que la notion de variable liée (ou muette), qui est considérée en mathématiques classiques comme une évidence, est un problème qui nécessite un traitement spécifique non trivial.

2.2 Termes et substitution

Définition 2.2.1 (λ-termes) *Les termes du λ-calcul sont des suites finies de symboles. Les symboles sont : "(", ")", "λ" , "." (qui jouent un rôle particulier) et un ensemble infini dénombrable de symboles (par exemple contenant l'alphabet) qu'on appelle les variables. "=" dans ce contexte représente l'égalité symbolique, deux termes sont égaux s'ils ont les mêmes symboles dans le même ordre. Soit L l'ensemble des termes, on le définit par induction :*

- *les variables sont des termes*
- *soit t et u des termes, alors $(u)t$ est un terme*
- *soit x une variable et t un terme, alors $\lambda x.t$ est un terme*

Les termes du λ-calcul sont appelés des λ-termes. Quand on dit "soit t un λ-terme" il faut le comprendre comme $t \in L$. On s'autorisera la notation xyz pour $((x)y)z$.

Au lieu de voir les termes comme des suites de symboles, on peut aussi les voir comme des arbres, ce qui est la manière standard de faire en informatique. Ici, on se contentera de les voir comme des suites de symboles.

Les termes de la forme $\lambda x.u$ sont appelés des abstractions. On peut les penser comme des fonctions. Par exemple $\lambda x.x$ "est" la fonction qui à x associe x. Cet aspect sera développé dans la partie 2.4.

Définition 2.2.2 (Occurrences libres) *Les occurrences libres d'une variable x sont définies par induction :*

- *soit t une variable, si $t = x$ l'occurence de x dans t est libre.*
- *soit $t = (u)v$ les occurrences libres de x dans t sont celles de x dans u et dans v*
- *soit $t = \lambda y.u$ (avec y une variable quelconque) les occurrences libres de x dans t sont celles de x dans u si $y \neq x$. Si $x = y$, alors x n'a pas d'occurrence libre dans t.*

Exemple Soit $t := \lambda x.u$ et $t' := \lambda y.u$, x n'a pas d'occurence libre dans t et les occurrences libres de x dans t' sont les occurrences libres de x dans u.

Définition 2.2.3 *Soit t un terme. On dit qu'une variable de t est libre si elle a au moins une occurence libre dans t. Une variable de t est liée si elle apparaît après un λ. Un terme sans variable libre est un terme clos.*

Exemples

- la variable x est liée dans $\lambda x.u$
- la variable x est libre dans $(x)y$
- la variable x est libre dans x
- la variable x est libre **et** liée dans $(\lambda x.u)x$

Définition 2.2.4 (Remplacement dans L) *Soit $t, u_1, \ldots, u_n$ des termes et $x_1, \ldots, x_n$ des variables distinctes. On note $t\langle u_1/x_1, \ldots, u_n/x_n\rangle$ le remplacement des variables $x_1, \ldots, x_n$ par les termes $u_1, \ldots, u_n$ dans le terme t. Si une variable x_{i_0} n'a pas d'occurence libre dans t alors la remplacer ne change pas le terme :*

$$t\langle u_1/x_1, \ldots, u_{i_0}/x_{i_0}, \ldots, u_n/x_n\rangle = t\langle u_1/x_1, \ldots, u_{i_0-1}/x_{i_0-1}, u_{i_0+1}/x_{i_0+1}, \ldots, u_n/x_n\rangle$$

Exemples

- $(xy)\langle z/y\rangle = xz$
- $(\lambda x.y)\langle z/y\rangle = \lambda x.z$
- $(\lambda x.y)\langle z/x\rangle = \lambda x.y$ (car x n'a pas d'occurence libre dans $\lambda x.y$)
- $(\lambda x.y)\langle z/a\rangle = \lambda x.y$ (car a n'a pas d'occurence libre dans $\lambda x.y$)
- $(xxy)\langle z/x, a/y\rangle = zza$
- $x\langle z/x\rangle\langle a/z\rangle = a$
- $(\lambda x.y)\langle x/y\rangle = \lambda x.x$

Le dernier exemple est une illustration du problème de capture. En effet les λ-termes $(\lambda x.z)$ et $(\lambda y.z)$ sont en un certain sens les mêmes, la variable z est libre dans les deux termes. Par contre, les λ-termes $(\lambda x.y)$ et $(\lambda x.x)$ sont différents, le premier a une variable libre alors que le second est clos. L'opération de remplacement du dernier exemple a changé une variable (en l'occurrence y) en une autre variable (x) qui à été capturée par la variable x sous le symbole λ. Le terme en a été fondamentalement changé, c'est ce qu'on appelle une capture.

2.3 Alpha-équivalence

Définition 2.3.1 (α-équivalence) *On définit une relation sur l'ensemble des termes du λ-calcul L, qu'on appelle α-équivalence et qui se note $\equiv$. L' α-équivalence est définie par induction :*

- *soit x et x' des variables alors $x \equiv x'$ si et seulement si $x = x'$*
- *soit $u := (w)v$ et u' des termes alors $u \equiv u'$ si et seulement si u' est de la forme $u' = (w')v'$ avec w' et v' des termes et $w \equiv w'$ et $v \equiv v'$*
- *soit $u := \lambda x.v$ et u' des termes alors $u \equiv u'$ si et seulement si u' est de la forme $u' = \lambda x'.v'$ avec v' un terme et $v\langle y/x\rangle \equiv v'\langle y/x'\rangle$ pour toute variable y sauf un nombre fini.*

La raison pour laquelle on demande "pour toute variable y sauf un nombre fini" est pour éviter les problèmes de capture. En effet, il ne peut y avoir de problème de capture que pour un nombre fini de variables, d'où la nécéssité d'avoir un ensemble de variables infini dénombrable.

En mathématiques usuelles, les fonctions $f : x \mapsto x$ et $f : y \mapsto y$ sont identiques tout comme les formules $\forall x,\ P(x)$ et $\forall y,\ P(y)$ puisque les variables x et y sont muettes. On aimerait formaliser cette notion de variable muette dans les termes du λ-calcul et dire que $\lambda x.x$ et $\lambda y.y$ sont égaux pour une certaine définition de l'égalité. L' α-équivalence correspond exactement à cette "certaine définition de l'égalité". Par exemple, $\lambda x.x \equiv \lambda y.y$, on le voit en utilisant la définition par induction avec $v = x$ et $v' = y$.

On peut aussi utiliser le formalisme du λ-calcul pour formaliser la notion de variable muette en mathématiques. Par exemple, $\forall x, P(x)$ devient $\forall(\lambda x.P(x))$. $\forall$ est alors une fonction et puisqu'on est à α-équivalence près on a bien $\forall(\lambda x.P(x)) \equiv \forall(\lambda y.P(y))$.

Proposition 2.3.2 *L'α-équivalence est une relation d'équivalence.*

Proposition 2.3.3 *Soit u et u' des termes, $u \equiv u'$ implique que u et u' ont les mêmes variables libres.*

Proposition 2.3.4 *Soit u, u' et t des termes, x une variable. Si $u \equiv u'$ et si aucune variable libre de t n'est liée dans u ou u' alors $u\langle t/x\rangle \equiv u'\langle t/x\rangle$*

Exemple Si $u := \lambda x.xz$ et $u' := \lambda y.yz$ on a bien $u \equiv u'$ mais

$$u\langle x/z\rangle = \lambda x.xx \not\equiv \lambda y.yx = u'\langle x/z\rangle$$

Car on a substitué la variable z par un terme qui est la variable x, qui est liée dans u.

Définition 2.3.5 (Passage au contexte) *Soit R une relation binaire sur L. On dit que R passe au contexte si :*
pour tout $t, t', u, u' \in L$ et pour toute variable x;

$$t\,R\,t' \Rightarrow \lambda x.t\,R\,\lambda x.t' \quad et \quad t\,R\,t' \wedge u\,R\,u' \Rightarrow (u)t\,R\,(u')t'$$

Proposition 2.3.6 *La relation $\equiv$ passe au contexte.*

Démonstration 2.3.6 Soit u et u' des termes et $u \equiv u'$. Montrer $\lambda x.u \equiv \lambda x.u'$ revient à montrer $u\langle y/x\rangle \equiv u'\langle y/x\rangle$ pour toute variable y sauf un nombre fini. Or u et u' étant des termes, ils ont un nombre fini de symboles et donc ont un nombre fini de variables liées. Donc pour toute variable y qui n'est pas une variable liée de u ou u', on a bien $u\langle y/x\rangle \equiv u'\langle y/x\rangle$ par la proposition 2.3.4. L'autre condition pour que $\equiv$ passe au contexte est vraie par définition de $\equiv$.

Définition 2.3.7 *On note $\Lambda = {}^{L}\!/_{\equiv}$ l'ensemble des termes quotienté par la relation d'α-équivalence.*

On remarque que la classe d'équivalence d'une variable x est x car deux variables sont α-équivalentes uniquement si elles sont les mêmes. De plus, on peut définir une notion de variable libre dans Λ car deux termes α-équivalents ont les mêmes variables libres. Les opérations $u, v \mapsto (u)v$ et $u, x \mapsto \lambda x.u$ sont compatibles avec $\equiv$. Par exemple dans le cas de la première opération, si $u \equiv u'$ et $v \equiv v'$ alors $(u)v \equiv (u')v'$. Il s'agit de la même notion de compatibilité que celle entre la loi d'un groupe et la loi d'un de ses groupes quotient. On peut donc définir ces opérations sur Λ.

Définition 2.3.8 (Substitution sur Λ) *Soit u, $t_1, \ldots, t_k$ des termes et soit $x_1, \ldots, x_k$ des variables distinctes : $u[t_1/x_1, \ldots, t_k/x_k] := u'\langle t_1/x_1, \ldots, t_k/x_k\rangle$ avec $u' \equiv u$ tel qu'aucune variable liée de u' ne soit libre dans $t_1, \ldots, t_k$ (on admet qu'il est toujours possible de trouver un tel u').*

Exemples

- $(\lambda x.y)[x/y] \equiv \lambda b.x$ alors que $(\lambda x.y)\langle x/y\rangle = \lambda x.x$
- $(\lambda x.xz)[(xy)/z] \equiv \lambda k.k(xy)$ alors que $(\lambda x.xz)\langle (xy)/z\rangle = \lambda x.x(xy)$
- $x[y/x] \equiv y$

En utilisant les notations de la dernière definition, on remarque que la classe d'équivalence de $u[t_1/x_1, \ldots, t_k/x_k]$ ne dépend pas du choix de u'. Cela s'illustre dans le premier exemple où on a choisi de remplacer x par k mais on aurait tout aussi bien pu choisir une autre variable différente de x et y.

2.4 Bêta-réduction et propriété de Church-Rosser

Dans la suite, on va confondre $=$ et $\equiv$ étant donné qu'on travaille dans Λ.

Proposition 2.4.1 *Soit u, u', t, t' des λ-termes, si $(\lambda x.u)t \equiv (\lambda x'.u')t'$ alors $u[t/x] \equiv u'[t'/x']$*

Définition 2.4.2 *Un terme de la forme $(\lambda x.u)t$ est appelé un redex, $u[t/x]$ est appelé son contracté. La proposition* 2.4.1 *garantit que cette notion est bien définie sur Λ.*

Définition 2.4.3 *On définit la relation binaire β_0 sur Λ. $t \beta_0 t'$ se lit : "t' est obtenu en contractant un redex de t" ou encore "t' est obtenu par β-réduction de t en 1 étape". On définit β_0 par induction :*

- *soit x une variable alors $x \beta_0 t$ est faux pour tout t*
- *soit $t := \lambda x.u$ un terme, alors $t \beta_0 t'$ si et seulement si $t' = \lambda x.u'$ avec $u \beta_0 u'$*
- *soit $t := (u)v$ un terme, alors $t \beta_0 t'$ si et seulement si:*
  - *ou bien $t' = (u)v'$ avec $v \beta_0 v'$*
  - *ou bien $t' = (u')v$ avec $u \beta_0 u'$*
  - *ou bien $u = \lambda x.w$ et $t' = w[v/x]$*

Exemples

- $(\lambda x.x)t$ se β-réduit en $x[t/x] = t$. On peut voir cela comme le résultat de l'application de la fonction identité à t
- $(\lambda x.xyz)t$ se β-réduit en $(xyz)[t/x] = tyz$
- $(\lambda x.(\lambda y.xy))t$ se β-réduit en $(\lambda y.xy)[t/x] = \lambda y.ty$ en supposant que la variable y ne soit pas libre dans t
- $((\lambda x.x)t)((\lambda x.x)t)$ se β-réduit en $(t)((\lambda x.x)t)$ ou en $((\lambda x.x)t)(t)$

La β-réduction, de par son nom et avec les exemples qui sont donnés ci-dessus, laisse supposer qu'on "réduit" une certaine quantité, en l'occurrence la longueur du terme. Cependant, l'exemple suivant démontre que ce n'est pas toujours le cas :

$$\delta := \lambda x.xx,\ \delta\delta \text{ se } \beta\text{-réduit en } (xx)[\delta/x] = \delta\delta$$

On a alors une suite de réductions infinie. La longueur du terme peut même augmenter :

$$t := \lambda x.((xx)x),\ tt \text{ se } \beta\text{-réduit en } ((xx)x)[t/x] = (tt)t \text{ qui se } \beta\text{-réduit en } ((tt)t)t$$

Définition 2.4.4 *On définit la relation binaire β sur Λ comme la clôture réflexive et transitive de β_0. On appelle cette relation la β-réduction.*

Définition 2.4.5 *Un terme t est dit normal ou en forme normale s'il ne contient aucun redex. Les termes normaux sont les termes obtenus en appliquant un nombre fini de fois les règles suivantes :*

- *si x est une variable alors x est normal*
- *si t est normal alors $\lambda x.t$ est normal*
- *si u et t sont normaux et si u ne commence pas par λ alors $(u)t$ est normal*

Un terme normal est donc un terme t tel que $t \beta t' \Rightarrow t \equiv t'$.
Il s'en suit que les termes normaux sont exactement ceux de la forme :

$$\lambda x_1 \ldots . \lambda x_k . x t_1 \ldots t_n$$

avec $k, n \geq 0$; $x_1, \ldots, x_k, x$ des variables et $t_1, \ldots, t_n$ des termes normaux.

Définition 2.4.6 (Normalisation) *Un terme t est dit normalisable s'il existe un terme normal t' tel que $t \beta t'$.*

Définition 2.4.7 (Normalisation forte) *Un terme t est dit fortement normalisable s'il n'existe aucune suite infinie $t_0 = t, t_1, \ldots$ telle que $t_i \beta t_{i+1}$ pour tout $i > 0$.*

Exemples

- le terme $\delta\delta$ n'est pas normalisable
- $t := (\lambda x.xx)\lambda x.x$ est fortement normalisable car t se β-réduit en $(\lambda x.x)\lambda x.x$ qui se β-réduit en $\lambda x.x$ qui est normal. Et on a exploré toutes les β-réductions possibles
- $(\lambda x.y)(\delta\delta)$ est normalisable mais pas fortement normalisable. En effet, il se β-réduit en $y[(\delta\delta)/x] = y$ (car x n'a pas d'occurence libre dans y) qui est normal. Il peut aussi se réduire en lui-même si on choisit de réduire la partie en $\delta\delta$

Définition 2.4.8 (PCR) *Soit R une relation binaire, on dit que R a la propriété de Church-Rosser (PCR) si :*

$$\forall t, u, u';\ (t\, R\, u)\ \wedge\ (t\, R\, u') \Rightarrow \exists v;\ (u\, R\, v)\ \wedge\ (u'\, R\, v)$$

Théorème 2.4.9 *La β-réduction a la Propriété de Church-Rosser.*

Théorème 2.4.10 *Il y a unicité de la forme normale lorsqu'elle existe.*

Démonstration 2.4.10 si t_1 et t_2 sont normaux et si $t \beta t_1$ et $t \beta t_2$ alors, d'après la PCR, il existe un terme t_3 tel que $t_1 \beta t_3$ et $t_2 \beta t_3$ or t_1 et t_2 sont normaux donc $t_1 \equiv t_3 \equiv t_2$

3 Déduction naturelle et logiques d'ordre supérieur

3.1 Déduction naturelle

Définition 3.1.1 (Formules propositionnelles) *Soit un ensemble $\mathcal{V}$ (qui contient par exemple l'alphabet en majuscules) qu'on appellera l'ensemble des variables propositionnelles. On définit inductivement l'ensemble $\mathcal{F}_\mathcal{P}$ des formules propositionnelles comme une partie de l'ensemble des suites finies de variables et de symboles "$\Rightarrow$", "$\wedge$", "$\bot$", (", ")", :*

$$\frac{A \in \mathcal{V}}{A \in \mathcal{F}_\mathcal{P}} \qquad \frac{A \in \mathcal{F}_\mathcal{P} \quad B \in \mathcal{F}_\mathcal{P}}{(A \Rightarrow B) \in \mathcal{F}_\mathcal{P}} \qquad \frac{A \in \mathcal{F}_\mathcal{P} \quad B \in \mathcal{F}_\mathcal{P}}{(A \wedge B) \in \mathcal{F}_\mathcal{P}} \qquad \frac{}{\bot \in \mathcal{F}_\mathcal{P}}$$

Avant d'aller plus loin dans l'abstraction on peut imaginer les variables propositionnelles comme pouvant valoir "vrai" ou "faux". Ainsi la formule $A \wedge B$ est vraie si A et B sont vrais, et fausse si A ou B est faux. La formule $A \Rightarrow A$ est tout le temps vraie peu importe la valeur de A (cf). Pour savoir si une formule est toujours vraie (on dit qu'elle est valide) il suffit alors de la tester pour toutes les valeurs possibles des variables. La formule $A \Rightarrow B$ n'est pas valide car, si A est vrai et si B est faux alors $A \Rightarrow B$ est faux. Voir les variables de manière binaire comme on vient de le décrire suppose le tiers exclu. En effet, la formule $A \vee (A \Rightarrow \bot)$ est valide, pour s'en rendre compte il suffit de la tester pour A vrai et A faux.

On va maintenant donner un autre sens à "A est vrai" que le sens calculatoire dont on vient de parler, en considérant qu'une formule est vraie si on peut la prouver.

Définition 3.1.2 (Séquent) *Un séquent est un couple formé d'un ensemble de formules Γ appelé les hypothèses et d'une formule P appelée la conclusion. On les notes $\Gamma \vdash P$.*

Remarque Si $A_1, \ldots, A_n$ sont des formules on note $A_1, \ldots, A_n \vdash P$ le séquent $A_1, \ldots, A_n \vdash P$. De plus l'ensemble des hypothèses peut être vide. On note alors $\vdash A$. On s'autorise à pouvoir avoir dans les hypothèses plusieurs fois la même formule.

Définition 3.1.3 (Déduction naturelle) *La déduction naturelle est un système d'inférence sur les séquents. Si un séquent appartient à l'ensemble défini par ces règles, on dit qu'il est prouvable. Dans la suite, Γ est un ensemble de formules propositionnelles quelconque.*

$$\frac{}{\Gamma, A \vdash A}\ (hyp) \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ (\wedge\mathcal{I}) \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ (\Rightarrow\mathcal{I})$$

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ (\wedge^1\mathcal{E}) \qquad \frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ (\wedge^2\mathcal{E}) \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\ (\Rightarrow\mathcal{E}) \qquad \frac{\Gamma \vdash \bot}{\Gamma \vdash C}\ (\bot\mathcal{E})$$

Explication intuitive des règles

(hyp) si A est dans notre ensemble d'hypothèses, alors, sous ces hypothèses, on peut conclure que A est vrai

($\wedge\mathcal{I}$) si, sous les hypothèses Γ, on a prouvé A et si, sous les hypothèses Γ, on a prouvé B, alors on peut en déduire que, sous les hypothèses Γ, on a prouvé $A \wedge B$

($\Rightarrow \mathcal{I}$) si, sous les hypothèses Γ et A, on a prouvé B, alors on peut en déduire que, sous les hypothèses Γ, on a prouvé $A \Rightarrow B$

($\wedge^1\mathcal{E}$) si, sous les hypothèses Γ, on a prouvé $A \wedge B$, alors on peut en déduire que, sous les hypothèses Γ, on a prouvé A

($\wedge^2\mathcal{E}$) si, sous les hypothèses Γ, on a prouvé $A \wedge B$, alors on peut en déduire que, sous les hypothèses Γ, on a prouvé B

($\Rightarrow \mathcal{E}$) si, sous les hypothèses Γ, on a prouvé $A \Rightarrow B$ et si, sous les hypothèses Γ, on a prouvé A, alors on peut en déduire que, sous les hypothèses Γ, on a prouvé B. Cette règle est aussi appelée *modus ponens*.

($\bot\mathcal{E}$) si, sous les hypothèses Γ, on a prouvé $\bot$, alors on peut en déduire que, sous les hypothèses Γ, on peut prouver n'importe quelle formule.

Lorsqu'on met de côté l'aspect très syntaxique, ces règles ne sont absolument pas surprenantes, elles vont de soi. C'est pour cela que ce système s'appelle déduction **naturelle**.

Les règles qui se terminent en $\mathcal{I}$ sont les règles d'introduction car elles introduisent un symbole dans la conclusion. Les règles qui se terminent en $\mathcal{E}$ sont les règles d'élimination car elles suppriment un symbole. Il y a une symétrie entre les règles d'introduction et les règles d'élimination.

Définition 3.1.4 (Arbre de dérivation) *Un arbre de dérivation est une succession de règles qui aboutissent à un séquent. Prouver un séquent (ou montrer qu'un séquent est prouvable), c'est écrire un arbre de dérivation qui aboutit à ce séquent et dont les feuilles sont des règles hypothèses.*

Exemple On va prouver en déduction naturelle le séquent $\vdash A \Rightarrow (B \Rightarrow A)$:

$$\dfrac{\dfrac{\dfrac{}{A, B \vdash A}\,(hyp)}{A \vdash B \Rightarrow A}\,(\Rightarrow \mathcal{I})}{\vdash A \Rightarrow (B \Rightarrow A)}\,(\Rightarrow \mathcal{I})$$

Si A est une formule, prouver le séquent $\vdash A$ revient à dire que la formule A est "vraie" sans aucune hypothèse. On vient donc de montrer que, sans aucune hypothèse et pour toutes formules A et B, la formule $A \Rightarrow (B \Rightarrow A)$ est "vraie". Les termes "vraie" sont entre guillemets car il ne s'agit que de la notion intuitive. Dans le contexte de la déduction naturelle, il n'y que la notion de preuve (ou de vérité) pour un séquent qu'on a définie avant. Cependant, si A est une formule, on peut montrer l'équivalence entre la validité de

A, au sens calculatoire évoqué précédemment, et la prouvabilité du séquent $\vdash A$. Il faut cependant rajouter la règle

$$\frac{A \Rightarrow \bot, \Gamma \vdash \bot}{\Gamma \vdash A}\ (abs)$$

qui correspond au raisonnement par l'absurde, c'est-à-dire le tiers exclu. En effet, on ne peut pas déduire $\Gamma \vdash A$ à partir du séquent $A \Rightarrow \bot, \Gamma \vdash \bot$ dans le système décrit précédemment. Si on veut pouvoir utiliser l'absurde, il faut nécessairement rajouter cette règle.

Théorème 3.1.5 (Affaiblissement des hypothèses) *Si $\Gamma \subset \Delta$ et si $\Gamma \vdash A$ est prouvable, alors $\Delta \vdash A$ est prouvable.*

Ce théorème traduit juste le fait que si l'on peut démontrer une formule A à partir d'hypothèses Γ, alors on peut rajouter des hypothèses en plus et toujours démontrer A. Il se prouve par récurrence sur la structure de l'arbre de dérivation.
Ce théorème n'est pas une règle de la déduction naturelle. Cependant, par souci de lisibilité, on notera son utilisation comme une règle notée (*aff*).

Dans ce système il n'y a pas de règles qui correspondent au symbole $\vee$. Il existe cependant une formule qui a des propriété similaires : $(A \Rightarrow \bot) \Rightarrow (B \Rightarrow \bot) \Rightarrow \bot$ (on peut s'en convaincre en utilisant la table de vérité).
Une propriété du $\vee$ est que, si on a A, on peut déduire $A \vee B$, et si on a B, on peut déduire $A \vee B$. On va montrer que, si on a $\Gamma \vdash A$, on peut déduire $\Gamma \vdash (A \Rightarrow \bot) \Rightarrow (B \Rightarrow \bot) \Rightarrow \bot$ et que, si on a $\Gamma \vdash B$, on peut déduire $\Gamma \vdash (A \Rightarrow \bot) \Rightarrow (B \Rightarrow \bot) \Rightarrow \bot$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, A \Rightarrow \bot, B \Rightarrow \bot \vdash A \Rightarrow \bot}\ (hyp) \quad \dfrac{\Gamma \vdash A}{\Gamma, A \Rightarrow \bot, B \Rightarrow \bot \vdash A}\ (aff)}{\Gamma, A \Rightarrow \bot, B \Rightarrow \bot \vdash \bot}\ (\Rightarrow \mathcal{E})}{\Gamma, A \Rightarrow \bot \vdash (B \Rightarrow \bot) \Rightarrow \bot}\ (\Rightarrow \mathcal{I})*}{\Gamma \vdash (A \Rightarrow \bot) \Rightarrow (B \Rightarrow \bot) \Rightarrow \bot}\ (\Rightarrow \mathcal{I})$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, A \Rightarrow \bot, B \Rightarrow \bot \vdash B \Rightarrow \bot}\ (hyp) \quad \dfrac{\Gamma \vdash B}{\Gamma, A \Rightarrow \bot, B \Rightarrow \bot \vdash B}\ (aff)}{\Gamma, A \Rightarrow \bot, B \Rightarrow \bot \vdash \bot}\ (\Rightarrow \mathcal{E})}{\Gamma, A \Rightarrow \bot \vdash (B \Rightarrow \bot) \Rightarrow \bot}\ (\Rightarrow \mathcal{I})*}{\Gamma \vdash (A \Rightarrow \bot) \Rightarrow (B \Rightarrow \bot) \Rightarrow \bot}\ (\Rightarrow \mathcal{I})$$

A l'étape $*$, en fonction de si on choisit de passer $A \Rightarrow \bot$ ou $B \Rightarrow \bot$ dans la conclusion, on peut déduire le séquent $\Gamma \vdash (A \Rightarrow \bot) \Rightarrow (B \Rightarrow \bot) \Rightarrow \bot$ ou $\Gamma \vdash (B \Rightarrow \bot) \Rightarrow (A \Rightarrow \bot) \Rightarrow \bot$. On a bien la symétrie à laquelle on s'attendait.

On pourrait se demander, sachant que $(A \Rightarrow \bot) \Rightarrow (B \Rightarrow \bot) \Rightarrow \bot$ semble se comporter comme un $\vee$, si on a aussi la règle de disjonction de cas :

$$\frac{\Gamma \vdash A \vee B \quad \Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma \vdash C}$$

On peut prouver qu'il n'est pas possible, à partir de $\Gamma \vdash (A \Rightarrow \bot) \Rightarrow (B \Rightarrow \bot) \Rightarrow \bot$; $\Gamma, A \vdash C$ et $\Gamma, B \vdash C$, de déduire $\Gamma \vdash C$. On n'a donc pas la disjonction de cas dans notre système. Cependant, cela devient le cas si on y ajoute la règle (abs).

On note $A \vee B \coloneqq (A \Rightarrow \bot) \Rightarrow (B \Rightarrow \bot) \Rightarrow \bot$ et, par souci de lisibilité, on sépare l'arbre en deux :

$$\dfrac{\dfrac{\Gamma \vdash A \vee B}{\Gamma, C \Rightarrow \bot \vdash A \vee B}\,(aff) \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, A, C \Rightarrow \bot \vdash C \Rightarrow \bot}\,(hyp) \quad \dfrac{\Gamma, A \vdash C}{\Gamma, A, C \Rightarrow \bot \vdash C}\,(aff)}{\Gamma, A, C \Rightarrow \bot \vdash \bot}\,(\Rightarrow \mathcal{E})}{\Gamma, C \Rightarrow \bot \vdash A \Rightarrow \bot}\,(\Rightarrow \mathcal{I})}{\Gamma, C \Rightarrow \bot \vdash (B \Rightarrow \bot) \Rightarrow \bot}\,(\Rightarrow \mathcal{E})$$

$$\dfrac{\dfrac{\vdots}{\Gamma, C \Rightarrow \bot \vdash (B \Rightarrow \bot) \Rightarrow \bot} \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, B, C \Rightarrow \bot \vdash C \Rightarrow \bot}\,(hyp) \quad \dfrac{\Gamma, B \vdash C}{\Gamma, B, C \Rightarrow \bot \vdash C}\,(aff)}{\Gamma, B, C \Rightarrow \bot \vdash \bot}\,(\Rightarrow \mathcal{E})}{\Gamma, C \Rightarrow \bot \vdash B \Rightarrow \bot}\,(\Rightarrow \mathcal{I})}{\Gamma, C \Rightarrow \bot \vdash \bot}\,(\Rightarrow \mathcal{E})}{\Gamma \vdash C}\,(abs)$$

3.2 Logique du premier ordre

Une théorie en logique du premier ordre est la donnée d'une signature et d'un ensemble de formules qu'on appelle axiomes. On suppose qu'on se trouve en logique classique, avec les quantificateurs et les règles habituelles.

Définition 3.2.1 (Signature) *Une signature, en logique du premier ordre, est un ensemble de symboles dits "de fonction" et de symboles dits "de prédicat". À chaque symbole on associe une arité qui correspond au "nombre d'arguments qu'il prend".*

Exemples

- $+$ est un symbole de fonction d'arité 2 (on le note de manière infixe, c'est-à-dire $a + b$ au lieu de $+(a, b)$).
- $=$ est un symbole de prédicat d'arité 2 (noté aussi de manière infixe).

Les symboles de fonction d'arité 0 sont appelés "constantes".

Définition 3.2.2 (Termes) *On définit les termes sur un ensemble infini de variables et une signature τ qui contient un ensemble de symboles de fonction $\mathcal{F}$ par induction :*

- *les constantes sont des termes*
- *les variables sont des termes*
- *si $f \in \mathcal{F}$ d'arité n et si $x_1, \ldots, x_n$ sont des termes, alors $f(x_1, \ldots, x_n)$ est un terme*

Définition 3.2.3 (Formule) *Une formule est une suite finie de symboles de prédicat et de symboles logiques, avec évidemment les règles usuelles de formation des formules.*

On peut intuitivement penser les symboles de fonction comme des fonctions sur les termes et les symboles de prédicat comme des fonctions qui vont des termes vers Vrai, Faux.

Exemple : théorie des groupes On peut modéliser la théorie des groupes en se donnant la signature τ, qui contient les symboles de fonctions $\mathcal{F} := \cdot, 1, ^{-1}$ d'arités respectives 2, 0 et 1, et le symbole de prédicat $\mathcal{P} := =$ d'arité 2. Ainsi que les axiomes :

- $\forall a,\ (a \cdot 1 = a) \wedge (1 \cdot a = a)$
- $\forall a, b, c,\ a \cdot (b \cdot c) = (a \cdot b) \cdot c$
- $\forall a,\ (a^{-1} \cdot a = 1) \wedge (a \cdot a^{-1} = 1)$
- $\forall a,\ a = a$
- $\forall a, b,\ a = b \Rightarrow b = a$
- $\forall a, b, c,\ (a = b \wedge b = c) \Rightarrow a = c$
- $\forall a, b,\ a = b \Rightarrow a^{-1} = b^{-1}$
- $\forall a, b, c, d,\ (a = b \wedge c = d) \Rightarrow (a \cdot c = b \cdot d)$

L'idée est la suivante : la signature nous donne des symboles et les axiomes décrivent comment ils se comportent. La théorie des ensembles est une théorie du premier ordre avec, entre autres, dans sa signature, les symboles de fonction $\cup, \cap, \varnothing$, d'arités respectives 2, 2, et 0, et le symbole de prédicat $\in$.

On remarque que quand on écrit $\forall$ ou $\exists$, on ne quantifie jamais sur les symboles de la signature, uniquement sur les variables représentant des termes. La signature est figée.

3.3 Logique d'ordre supérieur

En logique d'ordre supérieur, on ne se donne pas nécessairement de signature. À la place, on va s'autoriser à quantifier sur des variables qui ne représentent pas seulement des termes. Le fait de s'autoriser cette quantification va donner beaucoup de puissance aux systèmes que l'on va construire. Ce qui suit est un exemple de système de déduction en logique d'ordre supérieur.

Définition 3.3.1 (Formules propositionnelles du second ordre) *Soit un ensemble $\mathcal{V}$ que l'on appellera l'ensemble des variables. On définit inductivement l'ensemble ${\mathcal{F}_\mathcal{P}}^2$ des formules propositionnelles du second ordre comme une partie de l'ensemble des suites finies de variables et de symboles "$\Rightarrow$", "$\forall$", "(", ")", :*

$$\frac{X \in \mathcal{V}}{X \in {\mathcal{F}_\mathcal{P}}^2} \qquad \frac{A \in {\mathcal{F}_\mathcal{P}}^2 \quad B \in {\mathcal{F}_\mathcal{P}}^2}{A \Rightarrow B \in {\mathcal{F}_\mathcal{P}}^2} \qquad \frac{X \in \mathcal{V} \quad A \in {\mathcal{F}_\mathcal{P}}^2}{\forall X, A \in {\mathcal{F}_\mathcal{P}}^2}$$

Définition 3.3.2 (Déduction naturelle du second ordre) *La déduction naturelle du second ordre est le sytème d'inférence sur les séquents de ${\mathcal{F}_\mathcal{P}}^2$ donné par les règles suivantes :*

$$\frac{}{\Gamma, A \vdash A}\ \textit{(hyp)} \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ (\Rightarrow \mathcal{I}) \qquad \frac{\Gamma \vdash A}{\Gamma \vdash \forall X.A}\ (\forall^2 \mathcal{I})$$

$$\frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\ (\Rightarrow \mathcal{E}) \qquad \frac{\Gamma \vdash \forall X.A}{\Gamma \vdash A[B/X]}\ (\forall^2 \mathcal{E})$$

Pour la règle $(\forall^2 \mathcal{I})$, $X \in \mathcal{V}$ et X ne doit pas être libre dans Γ.
Pour la règle $(\forall^2 \mathcal{E})$, B est une formule quelconque et la substitution est celle sans capture, c'est-à-dire celle définie en 2.3.8.

La définition de variable libre ou liée est la même que dans le λ-calcul (avec $\forall$ à la place de λ). X est libre dans la formule $X \Rightarrow X$ et liée dans $\forall X.X \Rightarrow X$.

En regardant les règles $(\forall^2 \mathcal{I})$ et $(\forall^2 \mathcal{E})$, on peut se dire qu'au final, il n'y a pas beaucoup de différence avec un "pour tout" classique. En réalité, il y a une subtilité. La formule B dans la règle $(\forall^2 \mathcal{E})$ est quelconque, on peut très bien remplacer X par $\forall X.A$. Cela crée une circularité. On peut se demander (à juste titre) si cela ne crée pas de paradoxe. Dans ce cas précis, cette circularité ne cause pas de problème. Dans d'autres systèmes, elle cause en effet des problèmes et c'est cela qu'illustre le paradoxe de Girard.

On peut aussi se demander pourquoi les règles pour le "$\wedge$" et le "$\bot$" ont été enlevées. La réponse est que le système que l'on vient de définir peut exprimer la même chose que le système précédent (3.1.3) avec uniquement deux symboles : $\forall$ et $\Rightarrow$. En effet, les règles pour "$\wedge$" et "$\bot$" ont des équivalents :

$\forall X, X$ est l'équivalent de $\bot$: il y a un parallèle entre $(\forall^2 \mathcal{E})$ appliqué au séquent $\Gamma \vdash \forall X.X$ et $(\bot \mathcal{E})$

$$\frac{\Gamma \vdash \forall X.X}{\Gamma \vdash C}\ (\forall^2 \mathcal{E}) \qquad \frac{\Gamma \vdash \bot}{\Gamma \vdash C}\ (\bot \mathcal{E})$$

$\forall X, (A \Rightarrow (B \Rightarrow X)) \Rightarrow X$ est l'équivalent de $A \wedge B$. On va montrer qu'on peut déduire les séquents $\Gamma \vdash A$ et $\Gamma \vdash B$ à partir du séquent $\Gamma \vdash \forall X, (A \Rightarrow (B \Rightarrow X)) \Rightarrow X$:

$$\dfrac{\dfrac{\Gamma \vdash \forall X, (A \Rightarrow (B \Rightarrow X)) \Rightarrow X}{\Gamma \vdash (A \Rightarrow (B \Rightarrow A)) \Rightarrow A}\,(\forall^2\mathcal{E}) \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, A, B \vdash A}\,(hyp)}{\Gamma, A \vdash B \Rightarrow A}\,(\Rightarrow\mathcal{I})}{\Gamma \vdash A \Rightarrow (B \Rightarrow A)}\,(\Rightarrow\mathcal{I})}{\Gamma \vdash A}\,(\Rightarrow\mathcal{E})$$

$$\dfrac{\dfrac{\Gamma \vdash \forall X, (A \Rightarrow (B \Rightarrow X)) \Rightarrow X}{\Gamma \vdash (A \Rightarrow (B \Rightarrow B)) \Rightarrow B}\,(\forall^2\mathcal{E}) \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, A, B \vdash B}\,(hyp)}{\Gamma, A \vdash B \Rightarrow B}\,(\Rightarrow\mathcal{I})}{\Gamma \vdash A \Rightarrow (B \Rightarrow B)}\,(\Rightarrow\mathcal{I})}{\Gamma \vdash B}\,(\Rightarrow\mathcal{E})$$

Montrons maintenant qu'on peut déduire le séquent $\Gamma \vdash \forall X, (A \Rightarrow (B \Rightarrow X)) \Rightarrow X$ à partir des séquents $\Gamma \vdash A$ et $\Gamma \vdash B$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, A \Rightarrow (B \Rightarrow X) \vdash A \Rightarrow (B \Rightarrow X)}\,(hyp) \quad \dfrac{\Gamma \vdash A}{\Gamma, A \Rightarrow (B \Rightarrow X) \vdash A}\,(aff)}{\Gamma, A \Rightarrow (B \Rightarrow X) \vdash B \Rightarrow X}\,(\Rightarrow\mathcal{E}) \quad \dfrac{\Gamma \vdash B}{\Gamma, A \Rightarrow (B \Rightarrow X) \vdash B}\,(aff)}{\Gamma, A \Rightarrow (B \Rightarrow X) \vdash X}\,(\Rightarrow\mathcal{E})}{\Gamma \vdash A \Rightarrow (B \Rightarrow X) \Rightarrow X}\,(\Rightarrow\mathcal{E})}{\Gamma \vdash \forall X, A \Rightarrow (B \Rightarrow X) \Rightarrow X}\,(\forall^2\mathcal{I})$$

On a donc bien le même comportement qu'avec un $\wedge$.

Dans la section qui suit, nous allons construire le système F, qui est une théorie des types d'ordre supérieur. Ce système a des liens profonds avec la déduction naturelle.

4 Système F et isomorphisme de Curry-Howard

4.1 Théories des types : principes

En théorie des ensembles, les fonctions et les entiers sont des ensembles. La formule "$1_{\mathbb{N}} \in \sin$" est une formule qu'on a le droit d'écrire car le "1" des entiers naturels et la fonction sin sont des ensembles. Cela est un peu déroutant d'un point de vue intuitif car on "sent" bien que ce sont des entités de nature différente.

Dans les théories des types, chaque objet est associé à un type. Les types jouent en quelque sorte le même rôle que les ensembles. $A : \mathit{Type}$ est la notation pour dire que A est un type. $a : A$ est la notation pour dire que a est de type A. L'équivalent en théorie des ensembles est de dire que $a \in A$. On dit que a est un terme de type A. Si $A : \mathit{Type}$ et $B : \mathit{Type}$ alors $A \to B : \mathit{Type}$. C'est encore une fois purement syntaxique mais l'intuition est que cela correspond au type des fonctions de A vers B.

Si *int* est le type des entiers naturels et *real* le types des réels, les objets $1 : \mathit{int}$ et $\sin : \mathit{real} \to \mathit{real}$ sont des objets de nature vraiment différente (ce qui est intuitif), contrairement à ce qu'il se passe en théorie des ensembles.

Une théorie des types est constituée de règles d'inférence qui définissent comment sont construits les types, les termes et le typage des termes (c'est-à-dire l'association d'un terme à un type).

4.2 Système F : définition

Définition 4.2.1 (Règles du système F : types) *On se donne un ensemble de symboles $\mathcal{V} := X, Y, Z, \ldots$ que l'on appelle "variables de types". On se donne aussi les symboles $\to$ et Π. On définit l'ensemble des types par le système d'inférence suivant :*

$$\frac{X \in \mathcal{V}}{X \in \mathit{Type}} \qquad \frac{U \in \mathit{Type} \quad V \in \mathit{Type}}{(U \to V) \in \mathit{Type}} \qquad \frac{V \in \mathit{Type} \quad X \in \mathcal{V}}{\Pi X.V \in \mathit{Type}}$$

Dans la suite, on notera $T : \mathit{Type}$ pour $T \in \mathit{Type}$.

Définition 4.2.2 (Environnement de typage) *On se donne un ensemble de symboles $\mathcal{U} := x, y, z, \ldots$ (infini dénombrable) qu'on appelle des variables.*
Un environnement de typage est une fonction partielle de $\mathcal{U}$ vers les types, de domaine fini. Se donner un environnement de typage revient à assigner un type à un nombre fini de variables. Soit $x \in \mathcal{U}$, $T : \mathit{Type}$ et Γ un environnement de typage tel que $x \notin \mathcal{D}_\Gamma$, on note $\Gamma, x : T$ la fonction

$$\Gamma, x : T(x) = T \quad \textit{et si } y \neq x \textit{ et } y \in \mathcal{D}_\Gamma \textit{ alors } \Gamma, x : T(y) = \Gamma(y)$$

qui est aussi un environnement de typage.

Remarque Si Γ est un environnement de typage et que la notation $\Gamma, x : T$ est utilisée, alors on a implicitement supposé que $x \notin \mathcal{D}_\Gamma$.

Définition 4.2.3 (Règles du système F : termes) *Pour chaque terme t du système F, il existe un environnement Γ et un type U tels qu'on associe au couple (Γ, t) le type U. On note $\Gamma \vdash t : U$. Les règles de formation des termes ainsi que les règles de typage des termes sont :*

$$\frac{x \in \mathcal{D}_\Gamma \quad \Gamma(x) = T}{\Gamma \vdash x : T} \ (0) \qquad \frac{\Gamma \vdash t : U \to V \quad \Gamma \vdash u : U}{\Gamma \vdash tu : V} \ (1) \qquad \frac{\Gamma, x : U \vdash v : V}{\Gamma \vdash \lambda x^U.v : U \to V} \ (2)$$

$$\frac{\Gamma \vdash v : V \quad X \in \mathcal{V}}{\Gamma \vdash \Lambda X.v : \Pi X.V} \ (3)$$

Pour (3), il faut que X ne soit pas libre dans $\Gamma(z)$ pour tout $z \in \mathcal{D}_\Gamma$.

$$\frac{\Gamma \vdash t : \Pi X.V \quad U : \mathit{Type}}{\Gamma \vdash tU : V[U/X]} \ (4)$$

On se donne aussi la β-réduction pour les abstractions et pour les termes de la forme $\Lambda X.v$. Si t et u sont deux termes tels que $t \,\beta\, u$, on note $t \rightsquigarrow u$.

Explication intuitive

1. Les termes du système F sont des λ-termes (avec le symbole Λ en plus) auxquels on associe un type qui dépend de l'environnement de typage.
2. Le type $\Pi X.V$ correspond à une quantification : $\forall X, V$ sur tous les types, ce qui en fait un système d'ordre supérieur.
3. Un terme de la forme $\Lambda X.v$ correspond aussi à une fonction qui prend un type en argument et renvoie un terme dont le type dépend de l'argument. Autrement dit, v est paramétré par un type quelconque.

Remarques

1. On garde la même convention de parenthésage que dans la section sur le λ-calcul (cf 2.2).
2. Soit t un terme (du système F), la notion de variable libre est la même que dans le λ-calcul, mise à part la distinction entre les variables et les variables de types.
3. On peut étendre l'α-équivalence aux termes du système F, en distinguant variables et variables de types. Cela permet de définir la β-réduction sur les termes du système F. Tout comme pour le λ-calcul, c'est à cause de ces notions qu'il faut un ensemble infini de variables (cf 2.3).

4. Le type d'un terme dépend entièrement de l'environnement. Plus précisément, il ne dépend que du type des variables libres qu'il contient. Si Γ et Δ sont deux environnement de typage, $\mathcal{D}_\Gamma \subset \mathcal{D}_\Delta$, $\forall x \in \mathcal{D}_\Gamma, \Gamma(x) = \Delta(x)$ et si $\Gamma \vdash t : U$, alors $\Delta \vdash t : U$.
5. La règle (4) est circulaire. En effet, si $t : \Pi X.U$, alors $t(\Pi X.U)$ est un terme de type $U[(\Pi X.U)/X]$. Tout comme pour la déduction naturelle de second ordre, ici, cela ne pose pas de problème.

Exemples Dans cette série d'exemples $U : \mathit{Type}, V : \mathit{Type}$ et on se place implicitement dans un environnement où $u : U$ et $v : V$.

- $(\lambda x^U.v)u \rightsquigarrow v[u/x]$
- $(\lambda x^U.x)u \rightsquigarrow x[u/x] = u$
- $(\Lambda X.v)U \rightsquigarrow v[U/X]$
- $(\Lambda X.\lambda y^X.y)U \rightsquigarrow \lambda y^U.y$
- $(\Lambda Y.\Lambda X.\lambda x^{X\to Y}.y)UV \rightsquigarrow (\Lambda X.\lambda x^{X\to U}.y)V \rightsquigarrow \lambda x^{V\to U}.y$

Remarque Si $\Gamma \vdash t : U$ et $t \rightsquigarrow t'$, alors $\Gamma \vdash t' : U$. Cette propriété est appelée la subject-reduction.

4.3 Modélisation dans le système F

Le système F nous permet de représenter les entiers, une notion de type produit, les booléens et d'autres structures. On va présenter la construction du type produit et des entiers.

Définition 4.3.1 (Type produit) *Soit $U : \mathit{Type}$ et $V : \mathit{Type}$, on note :*

$$U \times V := \Pi X.(U \to V \to X) \to X$$

Soit $u : U$ et $v : V$, on note :

$$\langle u, v\rangle := \Lambda X.\lambda x^{U\to V\to X}.xuv$$

Soit $t : U \times V$, on définit alors les projections :

$$\pi^1 t := tU(\lambda x^U.\lambda y^V.x) \qquad \pi^2 t := tV(\lambda x^U.\lambda y^V.y)$$

Proposition 4.3.2 (Construction d'un terme de type produit) *Soit un environnement Γ où $u : U$ et $v : V$. On peut toujours construire le terme $\langle u, v\rangle : U \times V$.*

Démonstration 4.3.2 Soit Γ un environnement tel que $\Gamma \vdash u : U,\ \Gamma \vdash v : V$. Quitte à renommer les variables, on suppose que x n'apparaît ni dans u ni dans v et que $X \in \mathcal{V}$ n'est pas libre dans U et V. On note $\Gamma' \coloneqq \Gamma, x : U \to V \to X$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma' \vdash x : U \to V \to X}\,(0) \qquad \Gamma' \vdash u : U}{\Gamma' \vdash xu : V \to X}\,(1) \qquad \Gamma' \vdash v : V}{\Gamma' \vdash xuv : X}\,(1)}{\Gamma \vdash \lambda x^{U \to V \to X}.xuv : (U \to V \to X) \to X}\,(2) \qquad X \in \mathcal{V}}{\Gamma \vdash \Lambda X.\lambda x^{U \to V \to X}.xuv : \Pi X.(U \to V \to X) \to X}\,(3)$$

On peut donc toujours construire un terme de type $U \times V$.

Proposition 4.3.3 (Propriété du type produit) *Si $t \coloneqq \langle u, v \rangle$ alors*

$$\pi^1 t \rightsquigarrow u \text{ et } \pi^2 t \rightsquigarrow v$$

Démonstration 4.3.3 Soit un environnement dans lequel $u : U$ et $v : V$:

$$\begin{aligned}
\pi^1\langle u, v\rangle &= (\Lambda X.\lambda x_1^{U \to V \to X}.x_1 uv)\ U\ (\lambda x_2^U.\lambda y^V.x_2) \\
&\rightsquigarrow (\lambda x_1^{U \to V \to U}.x_1 uv)(\lambda x_2^U.\lambda y^V.x_2) \\
&\rightsquigarrow (\lambda x_2^U.\lambda y^V.x_2)uv \\
&\rightsquigarrow (\lambda y^V.u)v \\
&\rightsquigarrow u
\end{aligned}$$

$$\begin{aligned}
\pi^2\langle u, v\rangle &= (\Lambda X.\lambda x_1^{U \to V \to X}.x_1 uv)\ V\ (\lambda x_2^U.\lambda y^V.y) \\
&\rightsquigarrow (\lambda x_1^{U \to V \to V}.x_1 uv)(\lambda x_2^U.\lambda y^V.y) \\
&\rightsquigarrow (\lambda x_2^U.\lambda y^V.y)uv \\
&\rightsquigarrow (\lambda y^V.y)v \\
&\rightsquigarrow v
\end{aligned}$$

La propriété 4.3.3 montre bien que le type produit se comporte comme un produit cartésien.

Définition 4.3.4 (Type des entiers) *On définit le type*

$$\mathrm{Int} \coloneqq \Pi X.X \to (X \to X) \to X$$

On pose :

$$0 \coloneqq \Lambda X.\lambda x^X.\lambda y^{X \to X}.x$$

0 est bien de type Int. *Si $t : \mathrm{Int}$, on définit*

$$S\ t \coloneqq \Lambda X.\lambda x^X.\lambda y^{X \to X}.y(t\ X\ x\ y)$$

Le lecteur devinera que le terme 0 représente le 0 des entiers et S la fonction successeur. Pour comprendre l'intuition derrière cette définition, nous allons voir quelques exemples. Soit $U : \mathit{Type}$, $f : U \to U$ et $u : U$.

$$\begin{aligned} 0\ U\ u\ f &\coloneqq (\Lambda X.\lambda x^X.\lambda y^{X\to X}.x)\ U\ u\ f \\ &\rightsquigarrow (\lambda x^U.\lambda y^{U\to U}.x)\ u\ f \\ &\rightsquigarrow (\lambda y^{U\to U}.u)\ f \\ &\rightsquigarrow u \end{aligned}$$

Maintenant, regardons ce qu'il se passe pour le terme $1 \coloneqq S\ 0$:

$$\begin{aligned} S\ 0 &\coloneqq \Lambda X.\lambda x^X.\lambda y^{X\to X}.y((\Lambda X.\lambda x^X.\lambda y^{X\to X}.x)\ X\ x\ y) \\ &\rightsquigarrow \Lambda X.\lambda x^X.\lambda y^{X\to X}.y((\lambda x^X.\lambda y^{X\to X}.x)\ x\ y) \\ &\rightsquigarrow \Lambda X.\lambda x^X.\lambda y^{X\to X}.y((\lambda y^{X\to X}.x)\ y) \\ &\rightsquigarrow \Lambda X.\lambda x^X.\lambda y^{X\to X}.yx \end{aligned}$$

$$\begin{aligned} 1\ U\ u\ f &\rightsquigarrow (\Lambda X.\lambda x^X.\lambda y^{X\to X}.yx)\ U\ u\ f \\ &\rightsquigarrow (\lambda x^U.\lambda y^{U\to U}.yx)\ u\ f \\ &\rightsquigarrow (\lambda y^{U\to U}.y^{U\to U}u)\ f \\ &\rightsquigarrow f\ u \end{aligned}$$

De la même manière :

$$\begin{aligned} 2\ U\ u\ f &\rightsquigarrow f\ f\ u \\ 3\ U\ u\ f &\rightsquigarrow f\ f\ f\ u \\ 4\ U\ u\ f &\rightsquigarrow f\ f\ f\ f\ u \end{aligned}$$

Ainsi, le lecteur se convaincra aisément qu'un entier n est une fonction qui prend un argument un type: $U : \mathit{Type}$, un terme de ce type $u : U$ et une fonction de ce type vers lui-même $f : U \to U$ et renvoie $f^n(u)$ (avec $f^0 = \mathrm{id}$). On appelle ces entiers les entiers de Church.

4.4 Isomorphisme de Curry-Howard et cohérence

L'isomorphisme de Curry-Howard, tout d'abord, n'est pas un isomorphisme. Il s'agit d'une correspondance entre des règles de déduction (dans notre cas la déduction naturelle au second ordre) et des λ-termes typés (dans notre cas les termes du système F).
En effet, on établit les correspondances suivantes (cf 3.3.2,4.2.3 et 4.2.1) :

On remarque déjà que les types du système F sont construits de la même façon que les formules de ${\mathcal{F}_{\mathcal{P}}}^2$. Ainsi, on peut tout à fait faire correspondre une formule à un type. Maintenant, pour ce qui est de la correspondance entre les règles de déduction naturelle et les termes, on fait correspondre

$$\frac{}{\Gamma, A \vdash A}\ (hyp) \quad \text{avec} \quad \frac{x \in \mathcal{D}_\Gamma \quad \Gamma(x) = T}{\Gamma \vdash x : T}\ (0)$$

Avoir une formule A comme hypothèse correspond à se donner une variable de type A.

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ (\Rightarrow \mathcal{I}) \quad \text{avec} \quad \frac{\Gamma, x : U \vdash v : V}{\Gamma \vdash \lambda x^U.v : U \to V}\ (2)$$

Une preuve d'une implication $A \Rightarrow B$ correspond à un terme de type $A \to B$ (une fonction des preuves de A vers les preuves de B).

$$\frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\ (\Rightarrow \mathcal{E}) \quad \text{avec} \quad \frac{\Gamma \vdash t : U \to V \quad \Gamma \vdash u : U}{\Gamma \vdash tu : V}\ (1)$$

La règle $(\Rightarrow \mathcal{E})$ correspond à "l'évaluation" d'une fonction de $A \to B$ en un élément de A.

$$\frac{\Gamma \vdash A}{\Gamma \vdash \forall X.A}\ (\forall^2 \mathcal{I}) \quad \text{avec} \quad \frac{\Gamma \vdash v : V \quad X \in \mathcal{V}}{\Gamma \vdash \Lambda X.v : \Pi X.V}\ (3)$$

Et enfin

$$\frac{\Gamma \vdash \forall X.A}{\Gamma \vdash A[B/X]}\ (\forall^2 \mathcal{E}) \quad \text{avec} \quad \frac{\Gamma \vdash t : \Pi X.V \quad U : Type}{\Gamma \vdash tU : V[U/X]}\ (4)$$

Si A est une formule et Γ des hypothèses, prouver le séquent $\Gamma \vdash A$ revient à expliciter un terme de type A dans l'environnement Γ. S'il n'existe pas de tel terme, alors il n'existe pas de preuve du séquent et réciproquement. Un terme de type A peut être vu comme une preuve de A. Ainsi, même si nous sommes amenés à manipuler plusieurs termes de type A, au final, cela revient au même puisque ce sont tous des preuves de A.

Exemple

$$\cfrac{\cfrac{\cfrac{}{A, B \vdash A}\ (hyp)}{A \vdash B \Rightarrow A}\ (\Rightarrow \mathcal{I})}{\vdash A \Rightarrow (B \Rightarrow A)}\ (\Rightarrow \mathcal{I})$$

Cet arbre correspond au terme $\lambda x^A.\lambda y^B.x$. Ce terme peut être pensé comme une fonction qui prend une preuve de A et une preuve de B et renvoie une preuve de A.

Définition 4.4.1 (Coupure) *Une coupure dans un arbre de dérivation est la succession d'une règle d'introduction suivie d'une règle d'élimination du même type.*

Exemples

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Gamma, A \vdash B\end{array}}{\Gamma \vdash A \Rightarrow B}\ (\Rightarrow \mathcal{I}) \qquad \begin{array}{c}\vdots\\ \Gamma \vdash A\end{array}}{\Gamma \vdash B}\ (\Rightarrow \mathcal{E}) \qquad \dfrac{\dfrac{\begin{array}{c}\vdots\\ \Gamma \vdash A\end{array}}{\Gamma \vdash \forall X.A}\ (\forall^2 \mathcal{I})}{\Gamma \vdash A[B/X]}\ (\forall^2 \mathcal{E})$$

Théorème 4.4.2 *Le système F est fortement normalisable (cf 2.4.7).*

Théorème 4.4.3 *Le système F a la PCR (cf 2.4.8).*

Les preuves des deux théorèmes ci-dessus étant très techniques, on va les admettre.

Puisqu'on peut faire correspondre un λ-terme à un arbre, cela entraîne qu'on a une notion de réduction pour les arbres. Les deux théorèmes précédents garantissent que tout arbre est (fortement) normalisable et que la forme normale est unique.

Théorème 4.4.4 (Élimination des coupures) *Tout arbre de dérivation en déduction naturelle du second ordre se réduit en un arbre sans coupures. De manière équivalente, un arbre en forme normale n'a pas de coupures.*

En effet, les coupures correspondent à des redex. L'arbre étant en forme normale, il ne contient plus de redex. De manière plus détaillée:

- si on a une coupure en $(\Rightarrow)$, comme dans l'exemple, réduire cet arbre en forme normale revient à éliminer toutes les utilisations de l'hypothèse Γ, A en utilisant le séquent $\Gamma \vdash A$
- si on a une coupure en $(\forall^2)$, comme dans l'exemple, réduire cet arbre en forme normale revient à substituer dans toute la preuve X par B

Théorème 4.4.5 *Soit A une formule, la dernière règle d'une preuve en forme normale de $\vdash A$ est une règle d'introduction.*

Démonstration 4.4.5 Puisque les hypothèses sont vides, on ne peut pas déduire $\vdash A$ avec la règle *(hyp)*. La dernière règle n'est donc pas (*hyp*).

Supposons que la dernière règle est une règle d'élimination, l'arbre est donc de la forme:

$$\dfrac{\begin{array}{c}\vdots\\ \vdash B \Rightarrow A\end{array} \qquad \begin{array}{c}\vdots\\ \vdash B\end{array}}{\vdash A}\ (\Rightarrow \mathcal{E}) \quad \text{ou} \quad \dfrac{\begin{array}{c}\vdots\\ \vdash \forall X.B\end{array}}{\vdash A}\ (\forall^2 \mathcal{E})$$

avec B une certaine formule.

Les hypothèses étant vides, l'arbre doit contenir au moins une règle d'introduction. En effet, les règles d'introduction sont les seules qui permettent de réduire le nombre d'hypothèses et l'arbre ne peut pas commencer qu'avec des hypothèses non vides.

On remarque que si $(\Rightarrow \mathcal{I})$ est suivie par une règle d'élimination, alors c'est nécessairement la règle $(\Rightarrow \mathcal{E})$. En effet, $(\Rightarrow \mathcal{I})$ introduit le symbole "$\Rightarrow$". Le symbole $\forall$ ne peut donc pas être en tête de la formule. De même, si $(\forall^2\mathcal{I})$ est suivie par une règle d'élimination, alors c'est nécessairement la règle $(\forall^2\mathcal{E})$.

On sait donc qu'il y a au moins une règle d'introduction dans notre arbre qui est suivie par au moins une règle d'élimination (hypothèse). Donc, d'après ce qui vient d'être dit, il y a nécessairement une coupure. Cela contredit l'hypothèse que l'arbre est en forme normale. Donc la dernière règle ne peut pas être une règle d'élimination.

Définition 4.4.6 (Cohérence) *Un système est dit cohérent (ou consistant) s'il existe une formule qui ne peut pas être prouvée.*

Cette définition peut paraître troublante mais en fait elle ne l'est pas. Un système dans lequel tout est prouvable n'est pas intéressant puisqu'on peut prouver à la fois A et $\neg A$.

Proposition 4.4.7 *La déduction naturelle du second ordre est cohérentesi et seulement si le séquent $\vdash \forall X.X$ n'est pas prouvable.*

Démonstration 4.4.7 Si $\vdash \forall X.X$ n'est pas prouvable, alors le système est cohérent par définition.
Si $\vdash \forall X.X$ est prouvable, alors pour toute formule A et toutes hypothèses Γ:

$$\dfrac{\dfrac{\vdash \forall X.X}{\vdash A}\ (\forall^2\mathcal{E})}{\Gamma \vdash A}\ \textit{(aff)}$$

et donc tout séquent est prouvable.

Théorème 4.4.8 *La déduction naturelle du second ordre est cohérente.*

Démonstration 4.4.8 Supposons qu'on puisse démontrer $\vdash \forall X.X$. Cela signifie qu'il existe un arbre de dérivation en forme normale qui se termine par le séquent $\vdash \forall X.X$. La dernière règle est donc une règle d'introduction. Cela ne peut évidement pas être la règle $(\Rightarrow \mathcal{I})$. L'arbre est donc de la forme:

$$\dfrac{\begin{array}{c}\vdots\\ \vdash X\end{array}}{\vdash \forall X.X}\ (\forall^2\mathcal{I})$$

Par le même raisonnement, la règle avant $\vdash X$ est aussi une règle d'introduction, ce qui n'est pas possible car X est une variable. Contradiction, donc on ne peut pas démontrer le séquent $\vdash \forall X.X$. On a donc bien la cohérence.

Remarque On vient de raisonner sur la déduction naturelle du second ordre depuis la théorie des ensembles. En effet, on ne peut pas exprimer la preuve de la cohérence de la déduction naturelle dans la déduction naturelle. De manière générale, sous certaines hypothèses, si un système est cohérent alors il ne peut pas exprimer la preuve de sa cohérence: c'est le théorème d'incomplétude de Gödel.

5 Paradoxe de Girard

Le paradoxe de Girard est l'équivalent du paradoxe de Russell en théorie des types. On va construire une théorie des types et montrer qu'elle est incohérente. En plus de systèmes d'inférence pour définir les types et les termes, on va se donner un système de déduction qui permet de définir si un terme de type *Prop* est prouvable.

5.1 Calcul intuitionniste de Church

Définition 5.1.1 (Calcul intuitionniste de Church: types)

$$\frac{}{Prop : Type} \qquad \frac{A : Type \quad B : Type}{A \to B : Type}$$

Prop (comme son nom l'indique) est le type des propositions. Intuitivement, on peut penser un terme de type $A \to Prop$ comme une fonction qui va de A vers Vrai, Faux. *Prop* est une constante de type, contrairement au système F où on n'a que des variables de types (cf 4.2.1). Tous les types sont donc des chaînes finies de "*Prop*" et de "$\to$".

Définition 5.1.2 (Calcul intuitionniste de Church: termes) *On se donne un ensemble de symbole $\mathcal{U}$ infini dénombrable qui est l'ensemble des variables.*

$$\frac{x \in \mathcal{D}_\gamma \quad \gamma(x) = A}{\gamma \vdash x : A}\ (0) \qquad \frac{}{\vdash \Rightarrow : Prop \to (Prop \to Prop)}\ (1) \qquad \frac{\gamma \vdash t : A \to Prop}{\gamma \vdash \forall(t) : Prop}\ (2)$$

$$\frac{\gamma \vdash t : A \to B \quad \gamma \vdash u : A}{\gamma \vdash (tu) : B}\ (3) \qquad \frac{\gamma, x : A \vdash t : B}{\gamma \vdash \lambda x^A.t : A \to B}\ (4)$$

On se donne aussi la β-réduction.

Les règles $(0), (3)$ et (4) sont des règles qui ont déjà été abordées. La règle (1) dit simplement que $\Rightarrow$ est une constante, de type $Prop \to (Prop \to Prop)$ (son type ne dépend pas de l'environnement). On note $A \Rightarrow B$ au lieu de $\Rightarrow A\ B$. La règle (4) doit être comprise comme un "pour tout" usuel.

Définition 5.1.3 (Calcul intuitionniste de Church: déduction) *Une γ-formule est un terme de type Prop dans un environnement γ. On définit inductivement l'ensemble des γ-formules prouvables $\mathfrak{P}$:*

$$\frac{\gamma \vdash t \in \mathfrak{P} \quad t \rightsquigarrow u}{\gamma \vdash u \in \mathfrak{P}} \qquad \frac{\gamma \vdash t : Prop \quad \gamma \vdash u : Prop}{\gamma \vdash t \Rightarrow (u \Rightarrow t) \in \mathfrak{P}}$$

$$\frac{\gamma \vdash t : Prop \quad \gamma \vdash u : Prop \quad \gamma \vdash v : Prop}{\gamma \vdash (t \Rightarrow (u \Rightarrow v)) \Rightarrow ((t \Rightarrow u) \Rightarrow (t \Rightarrow v)) \in \mathfrak{P}}$$

$$\frac{\gamma \vdash \varphi : A \to Prop \quad \gamma \vdash \forall(\varphi) \in \mathfrak{P} \quad \gamma \vdash t : A}{\gamma \vdash (\varphi t) \in \mathfrak{P}}$$

$$\frac{\gamma, x : A \vdash \psi \Rightarrow \varphi \in \mathfrak{P} \quad x \text{ n'apparait pas dans } \psi}{\gamma \vdash \psi \Rightarrow \forall(\lambda x^A.\varphi) \in \mathfrak{P}}$$

$$\frac{\gamma \vdash \psi \Rightarrow \varphi \in \mathfrak{P} \quad \gamma \vdash \psi \in \mathfrak{P}}{\gamma \vdash \varphi \in \mathfrak{P}}$$

Dans ce système, contrairement à la théorie des ensembles, les propositions sont "au même niveau" que les objets de la théorie: ce sont tous des termes d'un certain type. Bien sûr, le type *Prop* joue un rôle un peu différent mais cela reste un type comme les autres. Les règles pour les types et les termes permettent de définir les objets que l'on va manipuler et les règles de déduction permettent de prouver des propriétés sur ces objets. Ce système de déduction est le système de Hilbert (à quelques modifications près). Il correspond à la logique usuelle sans tiers exclu ni axiome du choix. On pourrait ajouter la règle

$$\frac{\gamma \vdash \varphi : Prop}{\gamma \vdash ((\varphi \Rightarrow \bot) \Rightarrow \bot) \Rightarrow \varphi \in \mathfrak{P}}$$

où $\bot$ est le terme $\forall(\lambda x^{Prop}.x)$. Se donner cette règle (l'absurde) en plus ne changerait rien à la suite mais on ne va pas l'inclure par souci de généralité.
Il est important de garder en tête la première règle de déduction car elle est cruciale.

Dans la suite on utilisera les notations:

- si $\varphi : Prop$ on note $(\forall x : A)\varphi$ pour $\forall(\lambda x^A.\varphi)$
- si $\varphi : Prop$ on note $(\exists x : A)\varphi$ pour $(\forall \delta : Prop)((\forall x : A)\varphi \Rightarrow \delta) \Rightarrow \delta)$
- si $\varphi : Prop$ et $\psi : Prop$ on note $\varphi \wedge \psi$ pour $(\forall \delta : Prop)(\varphi \Rightarrow \psi \Rightarrow \delta) \Rightarrow \delta$
- si $\varphi : Prop$ et $\psi : Prop$ on note $\varphi \vee \psi$ pour $(\forall \delta : Prop)(\varphi \Rightarrow \delta) \Rightarrow (\psi \Rightarrow \delta) \Rightarrow \delta$

Les notations $\forall, \exists, \wedge$ et $\vee$ se comportent exactement de la même façon que d'ordinaire. Le lecteur pourra se référer aux parties sur la déduction naturelle et le système F (cf 3.1.3, 3.3.2, 4.2.3) pour essayer de s'en convaincre.

On remarque qu'on ne s'est pas donné de notion d'égalité entre deux termes de même type. En effet, ce système ne vient pas avec une notion primitive d'égalité. Soit $t, u : A$, on va définir ce qui s'appelle l'égalité intentionnelle:

$$t =_A u \text{ est la notation pour } (\forall P : A \to \mathit{Prop})Pt \Rightarrow Pu$$

En langage naturel: si t satisfait une propriété P, alors u satisfait aussi P.

$=_A$ est évidemment réflexive. Montrons qu'elle est symétrique:

On suppose $t =_A u$. Soit

$$P_0 \coloneqq \lambda x^A.(x =_A t) : A \to \mathit{Prop}$$

On a $(\forall P : A \to \mathit{Prop})Pt \Rightarrow Pu$ donc en particulier pour P_0:

$$P_0 t \Rightarrow P_0 u$$

$P_0 t = (\lambda x^A.(x =_A t))t \rightsquigarrow t =_A t$. Donc $(P_0 t \Rightarrow P_0 u) \rightsquigarrow ((t =_A t) \Rightarrow P_0 u)$. $t =_A t$ est prouvable par hypothèse, donc $P_0 u$ est prouvable.
$P_0 u = (\lambda x^A.(x =_A t))u \rightsquigarrow u =_A t$, donc $u =_A t$ est prouvable.

Montrons maintenant qu'elle est transitive:

On suppose $t =_A u$ et $u =_A v$. Soit

$$P_1 \coloneqq \lambda x^A.(t =_A x) : A \to \mathit{Prop}$$

On a $(\forall P : A \to \mathit{Prop})Pu \Rightarrow Pv$ donc en particulier, pour P_1:

$$P_1 u \Rightarrow P_1 v$$

$P_1 u = (\lambda x^A.(t =_A x))u \rightsquigarrow t =_A u$. Donc $(P_1 u \Rightarrow P_1 v) \rightsquigarrow ((t =_A u) \Rightarrow P_1 v)$. $t =_A u$ est prouvable par hypothèse, donc $P_1 v$ est prouvable.
$P_1 v = (\lambda x^A.(t =_A x))v \rightsquigarrow t =_A v$ donc $t =_A v$ est prouvable, ce qui conclut.

5.2 Types comme ensembles et relations

On aimerait bien pouvoir manipuler les types comme des ensembles, cependant ce n'est pas si facile. En effet, les types représentent des propriétés moins fines que les ensembles. Tout d'abord on aimerait, à partir d'un type, pouvoir définir un sous-type. Si $A : \mathit{Type}$, ce qui va jouer le rôle de sous-type est un terme $P : A \to \mathit{Prop}$. L'idée est que si $x : A$ et Px est prouvable, alors "$x \in P$".

On peut aussi définir la notion d'inclusion. Si $P : A \to \mathit{Prop}$ et $Q : A \to \mathit{Prop}$, l'inclusion de P dans Q est le prédicat $\lambda P^{A \to \mathit{Prop}}.\lambda Q^{A \to \mathit{Prop}}.(\forall x : A)(Px) \Rightarrow (Qx)$.

Si l'on souhaite munir un type d'une relation binaire, la première solution qui vient en tête est de dire que si $A : \mathit{Type}$, une relation binaire sur A est un terme $R : A \to A \to \mathit{Prop}$. Cette définition est la bonne mais soulève une subtilité. Si $A : \mathit{Type}$, $R : A \to A \to \mathit{Prop}$, $B : \mathit{Type}$ et $S : B \to B \to \mathit{Prop}$, un plongement de (A, R) [1] dans (B, S) est un terme $f : A \to B$ tel que :

$$(\forall x : A)(\forall y : A)(R\ x\ y) \Rightarrow (S\ (fx)\ (fy)) \text{ et } (\exists b : B)(\forall x : A)(S\ (fx)\ b)$$

Cependant, la condition $(\exists b : B)(\forall x : A)(S\ (fx)\ b)$ est en pratique trop forte car S n'est pas nécessairement définie pour tout $b : B$. Pour faire fonctionner cette définition, il faut se restreindre au domaine de S.

Définition 5.2.1 (Domaine d'une relation binaire) *Soit $A : \mathit{Type}$ et R une relation binaire sur A, le domaine de R est le prédicat $\lambda x^A.(\exists y : A)((R\ x\ y) \vee (R\ y\ x))$ noté $\mathcal{D}_R$.*

Définition 5.2.2 (Morphisme) *Soit $A, B : \mathit{Type}$ et R,S respectivement des relations binaire sur ces types, on dit que $f : A \to B$ est un morphisme de (A, R) vers (B, S) si :*

$$(\forall x : A)(\forall y : A)(R\ x\ y) \Rightarrow (S\ (fx)\ (fy))$$

Définition 5.2.3 (Plongement) *Soit $A, B : \mathit{Type}$ et R,S respectivement des relations binaire sur ces types, on dit que (A, R) se plonge dans (B, S) s'il existe $f : A \to B$ tel que :*

$$(\forall x : A)(\forall y : A)(R\ x\ y) \Rightarrow (S\ (fx)\ (fy))$$

et

$$(\exists b : B)[(\mathcal{D}_S\ b) \wedge (\forall x : A)(\mathcal{D}_R\ x) \Rightarrow (S\ (fx)\ b)]$$

On dit que f est un plongement de (A, R) vers (B, S).

[1] la notation (A, R) est une méta-notation pour désigner le type A et sa relation binaire R

5.3 Extension au second ordre

La nécessité d'étendre ce système au second ordre vient du fait qu'il ne capture pas certaines généralités. Par exemple, la notion d'inclusion qu'on a définie :

$$\lambda P^{A\to Prop}.\lambda Q^{A\to Prop}.(\forall x : A)(Px) \Rightarrow (Qx)$$

dépend du type A. Si on veut parler de l'inclusion dans un autre type, on a alors besoin d'un autre prédicat :

$$\lambda P^{B\to Prop}.\lambda Q^{B\to Prop}.(\forall x : B)(Px) \Rightarrow (Qx)$$

On voudrait définir des prédicats indépendamment du type des objets sur lesquels il agit.

Définition 5.3.1 (Extension au second ordre : types) *On se donne un ensemble $\mathcal{V}$ qui est l'ensemble des variables de type. On rajoute au système précédent les règles :*

$$\frac{X \in \mathcal{V}}{X : Type} \qquad \frac{X \in \mathcal{V} \quad A : Type}{(\Pi X)A : Type}$$

Définition 5.3.2 (Extension au second ordre : termes) *On rajoute au système précédent les règles :*

$$\frac{\gamma \vdash t : A \quad \gamma \text{ ne contient pas } X \text{ libre}}{\gamma \vdash \Lambda X.t : (\Pi X)A} \qquad \frac{\gamma \vdash t : (\Pi X)A \quad B : Type}{\gamma \vdash (tB) : A[B/X]}$$

$$\frac{\gamma \vdash t : (\Pi X)Prop}{\gamma \vdash (\forall t) : Prop}$$

On étend aussi la β-réduction aux termes de la forme $\Lambda X.t$.

Définition 5.3.3 (Extension au second ordre : déduction) *On étend les règles de déduction existantes pour inclure les nouveaux termes.*

Encore une fois, ce système est circulaire pour les mêmes raisons que les autres systèmes d'ordre supérieur que l'on a déjà vus. Cette fois, cela va mal se passer. Avant de voir cela, on va utiliser la force du second ordre pour (re)définir plusieurs prédicats.

Définition 5.3.4 (Égalité intentionnelle) $=_{Le}$[2] *est la notation pour*

$$\Lambda X.\lambda x^X.\lambda y^X.(\forall P : X \to Prop)(Px) \Rightarrow (Py)$$

On la note de manière infixe et sans donner l'argument de type, car on peut facilement le déduire du types des termes. Il n'y a donc pas de confusion possible. $=_{Le}$ est de type $(\Pi X)X \to X \to Prop$.

[2]Cette égalité est aussi appelée : égalité de Leibniz, d'où la notation

Définition 5.3.5 (Transitivité) Tran *est la notation pour*

$$\Lambda X.\lambda R^{X\to X\to Prop}.(\forall x:X)(\forall y:X)(\forall z:X)(R\ x\ y)\wedge(R\ y\ z)\Rightarrow(R\ x\ z)$$

Tran *est de type* $(\Pi X)(X\to X\to Prop)\to Prop$

Définition 5.3.6 (Domaine d'une relation binaire) $\mathcal{D}$ *est la notation pour*

$$\Lambda X.\lambda R^{X\to X\to Prop}.\lambda x^X.((\exists y:X)(R\ x\ y)\vee(R\ y\ x))$$

On note $\mathcal{D}_R\ x$ *pour* $\mathcal{D}\ X\ R\ x$*. Encore une fois, il n'y a pas de confusion possible car on connaît le type de* R*.* $\mathcal{D}$ *est de type* $(\Pi X)(X\to X\to Prop)\to X\to Prop$*.*

Définition 5.3.7 (EMB) EMB *est la notation pour*

$$\Lambda A.\lambda R^{A\to A\to Prop}.\Lambda B.\lambda S^{B\to B\to Prop}.(\exists f:A\to B)$$
$$[(\forall x:A)(\forall y:A)((R\ x\ y)\Rightarrow(S\ (fx)\ (fy)))]$$
$$\wedge$$
$$[(\exists b:B)((\mathcal{D}_S\ b)\wedge(\forall x:A)(D\,R\,x)\Rightarrow(S\ fx\ b))]$$

EMB *est de type* $(\Pi A)(A\to A\to Prop)\to(\Pi B)(B\to B\to Prop)\to Prop$*.*

C'est le prédicat qui correspond à : il existe un plongement de (A,R) dans (B,S).

Définition 5.3.8 (Accessibilité) acc *est la notation pour le prédicat d'accessibilité (cf 1.3.2)*

$$\Lambda A.\lambda R^{A\to A\to Prop}.\lambda x^A.(\forall P:A\to Prop)(\forall y:A)(R\ y\ x)\Rightarrow[((\forall z:A)(R\ z\ y)\Rightarrow Pz)\Rightarrow Py]$$

On note $\mathbf{acc}_R\ x$ *au lieux de* $\mathbf{acc}\ A\ R\ x$ *pour les mêmes raisons que précédemment.* **acc** *est de type* $(\Pi A)(A\to A\to Prop)\to A\to Prop$*.*

acc est exactement le même prédicat que vu en 1.3.2 traduit dans cette théorie. La partie $(\forall P:A\to Prop)$ permet d'exprimer l'intersection de toutes les relations.

Définition 5.3.9 (Relation bien fondée) WF *est la notation pour le prédicat de bonne fondation (cf 1.3.3)*

$$\Lambda A.\lambda R^{A\to A\to Prop}.(\forall x:A)(\mathbf{acc}_R\ x)$$

WF *est de type* $(\Pi A)(A\to A\to Prop)\to Prop$

Les propriétés comme la récurrence bien fondée, qu'on a montrées en 1.3, sont prouvables dans ce système. On les a montrées sans utiliser l'axiome du choix ni le tiers exclu, on s'est donc placé exactement dans les conditions de ce système. Pour prouver ces propriétés, il suffit donc de traduire ces preuves dans le formalisme de ce système.

Avant de passer à la suite il nous reste à définir la composition de fonctions. En effet, si $f : A \to B$ et $g : B \to C$ on ne peut pas écrire $(g)f$. Ce n'est pas un terme car f n'est pas de type B. Cependant, on peut tout à fait écrire $g(fa)$ avec $a : A$.

Définition 5.3.10 (Composition de fonctions) $\circ$ *est la notation pour la composition*

$$\Lambda A.\Lambda B.\Lambda C.\lambda f^{A\to B}.\lambda g^{B\to C}.\lambda x^{A}.(g(fx))$$

On la note de manière infixe sans donner les arguments de types.
$\circ$ *est de type* $(\Pi A)(\Pi B)(\Pi C)(A \to B) \to (B \to C) \to (A \to C)$

5.4 Paradoxe de Girard

Théorème 5.4.1 (Paradoxe de Girard) *Le calcul de Church intuitionniste étendu au second ordre n'est pas cohérent.*

Pour le démontrer, on a besoin de plusieurs résultats.

Proposition 5.4.2 EMB *est une relation transitive.*

Démonstration 5.4.2 Soit $A, B, C : Type$, R_A, R_B et R_C des relations binaires respectivement sur A, B et C. On suppose (**EMB** $A\ R_A\ B\ R_B$) et (**EMB** $B\ \ R_B\ C\ R_C$). On a donc qu'il existe $f : A \to B$ et $g : B \to C$ des plongements. On va montrer que $g \circ f$ est un plongement de (A, R_A) vers (C, R_C).

Soit $x, y : A$ tel que $(R_A\ x\ y)$, on a donc $(R_B\ (fx)\ (fy))$ et donc $(R_C\ (g \circ f\ x)\ (g \circ f\ y))$. On a aussi qu'il existe $c : C$ tel que $(\mathcal{D}_{R_C}\ c) \wedge (\forall y : B)(\mathcal{D}_{R_B}\ y) \Rightarrow (R_C\ (gy)\ c)$. Or $(\forall x : A)(\mathcal{D}_{R_A}\ x) \Rightarrow (\mathcal{D}_{R_B}(fx))$. Donc $(\forall x : A)(\mathcal{D}_{R_A}\ x) \Rightarrow (R_C\ (g \circ f\ x)\ c)$

Définition 5.4.3 (Système universel de notation) *Soit $A_0 : Type$, A_0 est un système de notation universel s'il existe un terme $i_0 : (\Pi X)(X \to X \to Prop) \to A_0$ tel que, si $(i_0\ A\ R) =_{Le} (i_0\ B\ S)$, alors il existe un morphisme de (A, R) vers (B, S).*

L'idée est que tous les types munis d'une relation binaire (qui satisfait certaines propriétés) se plongent dans A_0. Ainsi, si on munit A_0 d'une relation binaire (qui satisfait ces propriétés), alors A_0 va se plonger dans A_0. Cela crée un paradoxe. On aura donc montré que, s'il existe un système de notation universel, alors le système est incohérent. On construira ensuite un tel système dans le calcul de Church intuitionniste étendu au second ordre.

On suppose qu'il existe un tel A_0 et i_0.

Définition 5.4.4 (emb) *On définit la relation binaire* **emb** *par:*

$$\mathbf{emb} \coloneqq \lambda x^{A_0}.\lambda y^{A_0}.(\exists A : \mathit{Type})(\exists R : A \to A \to \mathit{Prop})(\exists B : \mathit{Type})(\exists S : B \to B \to \mathit{Prop})$$
$$(x =_{Le} (i_0\ A\ R) \wedge y =_{Le} (i_0\ B\ S) \wedge (\mathbf{EMB}\ A\ R\ B\ S))$$

emb *est une relation binaire sur* A_0 *(ie* $\mathbf{emb} : A_0 \to A_0 \to \mathit{Prop}$*)*

En language courant : si $x, y : A_0$ alors **emb** x y si x et y sont l'image par i_0 de deux types munis chacun d'une relation binaire tel que le premier se plonge dans le deuxième.

Proposition 5.4.5 **emb** *est transitive.*

Démonstration 5.4.5 Soit $x, y, z : A_0$ tels que (**emb** x y) et (**emb** y z). Par définition de **emb**, on a donc qu'il existe les types et relations binaires tels que:

- $x = (i_0\ A_x\ R_x)$
- $y = (i_0\ A_y^1\ R_y^1)$ et $y = (i_0\ A_y^2\ R_y^2)$
- $z = (i_0\ A_z\ R_z)$
- **EMB** $A_x\ R_x\ A_y^1\ R_y^1$
- **EMB** $A_y^2\ R_y^2\ A_z\ R_z$

On a donc $(i_0\ A_y^1\ R_y^1)(i_0\ A_y^2\ R_y^2)$, donc il existe un morphisme f de (A_y^1, R_y^1) vers (A_y^2, R_y^2). On a qu'il existe un plongement g de (A_y^2, R_y^2) dans (A_z, R_z). $g \circ f$ est un plongement de (A_y^1, R_y^1) vers (A_z, R_z). De plus, on a un plongement h de $(A_x\ R_x)$ dans (A_y^1, R_y^1). $(g \circ f) \circ h$ est un plongement de $(A_x\ R_x)$ dans (A_z, R_z). Donc on a bien **emb** x z.

Définition 5.4.6 (wf) *On définit le prédicat* **wf** *par :*

$$\mathbf{wf} \coloneqq \lambda x^{A_0}.(\exists A : \mathit{Type})(\exists R : A \to A \to \mathit{Prop})[(\mathbf{WF}\ A\ R) \wedge x =_{Le} (i_0\ A\ R)]$$

wf *est de type* $A_0 \to \mathit{Prop}$*.*

Définition 5.4.7 ($\mathbf{emb_{wf}}$) *On définit la relation binaire* $\mathbf{emb_{wf}}$ *par :*

$$\mathbf{emb_{wf}} \coloneqq \lambda x^{A_0}.\lambda y^{A_0}.(\mathbf{emb}\ x\ y) \wedge (\mathbf{wf}\ x) \wedge (\mathbf{wf}\ y)$$

$\mathbf{emb_{wf}}$ *est une relation binaire sur* A_0*.*

Proposition 5.4.8 $\mathbf{emb_{wf}}$ *est transitive.*

Démonstration 5.4.8 C'est une conséquence directe de la transitivité de **emb**

Théorème 5.4.9 *On a* **WF** A_0 $\mathbf{emb_{wf}}$*.*

Démonstration 5.4.9 Soit $S : Type$, R_S une relation binaire sur S.
Soit $P : S \to Prop$ définie par :
Px si et seulement si pour tout $A : Type$ et R_A relation binaire sur A si
[$\mathbf{emb_{wf}}$ $(i_0\ A\ R_A)$ $(i_0\ S\ R_S)$ et les plongemens de (A, R_A) vers (S, R_S) sont majorés par x] alors $\mathbf{acc}_{A_0}(i_0\ A\ R_A)$.

La définition de $\mathbf{emb_{wf}}$ nous permet de supposer $\mathbf{WF}\ S\ R_S$. Dans un premier temps, on va montrer par récurrence bien fondée qu'on a Px pour tout $x : S$. Soit $x : S$ et supposons que P est prouvable pour tout $y : S$ tel que $R_S\ y\ x$, montrons qu'elle est vraie pour x:

Soit $A : Type$ et R_A tel que $\mathbf{emb_{wf}}$ $(i_0\ A\ R_A)$ $(i_0\ S\ R_S)$ et les plongements de A dans S sont dominés par x.
Soit $B : Type$ et R_B tel que $\mathbf{emb_{wf}}$ $(i_0\ B\ R_B)$ $(i_0\ A\ R_A)$. Il existe donc deux plongements $f : B \to A$ et $g : A \to S$. Il existe aussi $[a : A$ et $\mathcal{D}_{R_A} a]$ tel que $(\forall b : B)(\mathcal{D}_{R_B}\ b) \Rightarrow (R_A\ (fb)\ a)$ d'où $(\forall b : B)(\mathcal{D}_{R_B}\ b) \Rightarrow (R_S\ (g \circ f\ b)\ (ga))$. ga est dans l'image de A par g donc $(R_S\ (ga)\ x)$. L'image de B par $g \circ f$ est dominée par (ga) et $(R_S\ (ga)\ x)$. Par hypothèse de récurrence, on a $\mathbf{acc}_{\mathbf{emb_{wf}}}\ (i_0\ B\ R_B)$.

Comme le choix de (B, R_B) est arbitraire, on a que $(\forall x : A_0)(\mathbf{emb_{wf}}\ x\ (i_0\ A\ R_A)) \Rightarrow \mathbf{acc}_{\mathbf{emb_{wf}}}\ x$ donc $\mathbf{acc}_{\mathbf{emb_{wf}}}\ (i_0\ A\ R_A)$.

On vient de montrer que si P est prouvable pour tout $y : S$ tel que $R_S\ y\ x$, alors Px est prouvable. On a supposé $\mathbf{WF}\ S\ R_S$, donc par récurrence bien fondée, on a P pour tout $x : S$.

On a que $(\forall x : A_0)(\mathbf{emb_{wf}}\ x\ (i_0\ S\ R_S)) \Rightarrow \mathbf{acc}_{\mathbf{emb_{wf}}}\ x$, car un plongement dans S est (par définition) majoré par un terme de type S. Donc $\mathbf{acc}_{\mathbf{emb_{wf}}}\ (i_0\ S\ R_S)$. Comme le choix de (S, R_S) est arbitraire, on en déduit que pour tout (S, R_S), on a $\mathbf{acc}_{\mathbf{emb_{wf}}}\ (i_0\ S\ R_S)$, ce qui est la définition de $\mathbf{WF}\ A_0\ \mathbf{emb_{wf}}$

Théorème 5.4.10 *On a*

$$(\forall A : Type)(\forall R : A \to A \to Prop)((\mathbf{WF}\ A\ R) \wedge (\mathbf{Tran}\ A\ R)) \Rightarrow (\mathbf{EMB}\ A\ R\ A_0\ \mathbf{emb_{wf}})$$

Démonstration 5.4.10 Soit $\mathbf{WF}\ A\ R$ et $\mathbf{Tran}\ A\ R$. On va construire un plongement $f : A \to A_0$.

Soit $a : A$ on va définir $R_a : A \to A \to Prop$ par

$$R_a := \lambda x^A.\lambda y^A.(R\ x\ y) \wedge ((R\ y\ a) \vee (y =_{Le} a))$$

On pose maintenant $f := \lambda a.(i_0\ A\ R_a) : A \to A_0$, on note qu'on a $\mathbf{WF}\ A\ R_a$ et $\mathbf{Tran}\ A\ R_a$, R_a est simplement une restriction de R.
Soit $a, b : A$ tel que $(R\ a\ b)$, on a bien $\mathbf{emb_{wf}}\ (fa)\ (fb)$ car l'identité de A est un plongement

de (A, R_a) vers (A, R_b). f est donc un morphisme de A vers A_0. Pour montrer que f est un plongement, il reste à montrer que l'image de A par f est dominée par un élément de A_0.

L'idée est de construire un type "$A \cup \infty$" et de définir une nouvelle relation R_∞ qui est la même que R avec ∞ l'élément maximum. Ensuite, par construction, on aura bien que $(\forall x : A)(\mathcal{D}_R\ x) \Rightarrow \mathbf{EMB}\ A\ R_x\ A \cup \infty\ R_\infty$, ce qui permettra de conclure.

On va construire ce type:

On définit

$$S \coloneqq (\Pi X)(A \to X) \to ((X \to X) \to X) \to X$$
$$h \coloneqq \lambda a^A.\Lambda X.\lambda x^{A\to X}.\lambda y^{(X\to X)\to X}.xa : A \to S$$
$$\infty \coloneqq \Lambda X.\lambda x^{A\to X}.\lambda y^{(X\to X)\to X}.y(\lambda z^X.z) : S$$

h est une injection de A dans S. On définit $R_\infty : S \to S \to Prop$ par:

$$R_\infty \coloneqq \lambda x^S.\lambda y^S.[(\exists a : A)(\exists b : A)((ha =_{Le} x) \wedge (hb =_{Le} y)) \wedge (R\ a\ b)] \vee [y =_{Le} \infty]$$

Ce qui conclut.

Théorème 5.4.11 *On a*

$$(\forall A : Type)(\forall R : A \to A \to Prop)((\mathbf{WF}\ A\ R) \wedge (\mathbf{Tran}\ A\ R)) \Rightarrow (\mathbf{EMB}\ A\ R\ A\ R) \Rightarrow \bot$$

Démonstration 5.4.11 Soit $A : Type$ et R une relation binaire sur A tel que : $(\mathbf{WF}\ A\ R)$, $(\mathbf{Tran}\ A\ R)$ et $(\mathbf{EMB}\ A\ R\ A\ R)$.

Il existe un plongement $f : A \to A$. On suppose que l'image de A par f est dominée par $a : A$. On va construire une suite infinie décroissante à partir de a :

On a $(R\ (fa)\ a)$ donc $(R\ (f \circ f\ a)\ (fa))$ car f est un plongement. De plus, on a que pour tout n $\mathcal{D}_R\ (f^n a)$. On a donc construit une suite infinie décroissante dans A. Or, d'après 1.3.5, il ne peut pas y avoir de suite infinie décroissante. On vient donc de déduire $\bot$.

Théorème 5.4.12 *Si un système universel de notation existe, alors le système est incohérent.*

Démonstration 5.4.12 D'après le théorème 5.4.9, on a $\mathbf{WF}\ A_0\ \mathbf{emb_{wf}}$. Le théorème 5.4.10 permet de déduire $(\mathbf{EMB}\ A_0\ \mathbf{emb_{wf}}\ A_0\ \mathbf{emb_{wf}})$. Enfin, le théorème 5.4.11 permet de déduire $\bot$.

Pour démontrer le paradoxe de Girard il ne reste plus qu'à construire un système universel de notation
On pose

$$A_0 \coloneqq ((\Pi B)((B \to B \to \mathit{Prop}) \to \mathit{Prop})) \to \mathit{Prop}$$

et

$$i_0 \coloneqq \Lambda B.\lambda R^{B\to B\to \mathit{Prop}}.\lambda x^{(\Pi B)(B\to B\to \mathit{Prop})\to \mathit{Prop}}.xBR : (\Pi B)(B \to B \to \mathit{Prop}) \to A_0$$

On suppose que $(i_0\ B\ S) =_{Le} (i_0\ C\ T)$ et on veut montrer qu'il existe un morphisme de (B, S) vers (C, T). On définit

$$\mathbf{MOR} \coloneqq \Lambda B.\lambda S^{B\to B\to \mathit{Prop}}.\Lambda C.\lambda T^{C\to C\to \mathit{Prop}}.$$

$$(\exists f : B \to C)(\forall x : B)(\forall y : B)(S\ x\ y) \Rightarrow (T\ (fx)\ (fy))$$

$$\mathbf{MOR} : (\Pi B)(B \to B \to \mathit{Prop}) \to (\Pi C)(C \to C \to \mathit{Prop}) \to \mathit{Prop}$$

MOR est le prédicat pour "il existe un morphisme de (B, S) dans (C, T)".

$$F \coloneqq \Lambda D.\lambda H^{D\to D\to \mathit{Prop}}.\mathbf{MOR}\ B\ S\ D\ H : (\Pi D)(D \to D \to \mathit{Prop}) \to \mathit{Prop}$$

$$Q \coloneqq \lambda x^{A_0}.xF : A_0 \to \mathit{Prop}$$

$Q(i_0\ X\ R_X) \rightsquigarrow \mathbf{MOR}\ B\ S\ X\ R_X.\ (i_0\ B\ S) =_{Le} (i_0\ C\ T)$ donc, par définition :

$$(\forall P : A_0 \to \mathit{Prop})P(i_0\ B\ S) \Rightarrow P(i_0\ C\ T)$$

Donc, en particulier:

$$Q(i_0\ B\ S) \Rightarrow Q(i_0\ C\ T)$$

$Q(i_0\ B\ S) \Rightarrow Q(i_0\ C\ T) \rightsquigarrow (\mathbf{MOR}\ B\ S\ B\ S) \Rightarrow Q(i_0\ C\ T)$, **MOR** est réflexive donc $(\mathbf{MOR}\ B\ S\ B\ S)$ est prouvable. On en déduit que $Q(i_0\ C\ T)$ est prouvable. $Q(i_0\ C\ T) \rightsquigarrow \mathbf{MOR}\ B\ S\ C\ T$, donc $\mathbf{MOR}\ B\ S\ C\ T$ est prouvable, ce qui conclut.

Conclusion

L'intérêt d'étudier les propriétés de divers systèmes logiques est qu'on peut y représenter des structures qui nous sont utiles : algorithmes, bases de données, etc ... Connaître les propriétés de ces systèmes permet alors d'en déduire des propriétés sur les objets qu'on y représente. Par exemple, si on peut modéliser un algorithme dans le système T (qui est semblable au système F), alors on peut en déduire que cet algorithme termine.

On peut aussi vouloir créer un système qui nous permet de faire des mathématiques, par exemple, la théorie des ensembles et toutes ses variantes. On peut aussi s'intéresser à la vérification automatique de preuves : un programme qui prend en entrée une preuve et nous dit si elle est correcte ou non. Il faut alors se demander quels systèmes logiques (dans lesquels on peut faire des mathématiques) sont les plus appropriés pour ce genre d'automatisation. Certains assistants de preuve se basent sur une théorie des types. Coq et Lean sont parmi les assistants de preuve les plus connus et se basent chacun sur une théorie des types qui est une extension du calcul des constructions.

Cependant, créer un système qui a les propriétés souhaitées est difficile. Se donner trop de liberté peut mener à des paradoxes et au contraire, trop se restreindre ne permet pas de faire ce que l'on voudrait. Le paradoxe de Girard en est un exemple : il paraît naturel de vouloir étendre le système au second ordre, pourtant, cela fait perdre la cohérence.

Références

[1] Thierry Coquand. An analysis of Girard's paradox. In *Proceedings of the Symposium on Logic in Computer Science (LICS '86), Cambridge, Massachusetts, USA, June 16-18, 1986*, pages 227–236. IEEE Computer Society, 1986.

[2] J.L. Krivine. *Lambda-calcul types et modèles.* Masson, 1990.

[3] Jean-Yves Girard (translated and with appendices by Y. Lafont & P. Taylor). *Proofs and Types.* Number 7 in Cambridge Tracts in Theoretical Computer Science. Cambridge university press, 1989.