Faculté des Sciences

Département de Mathématiques

Le lambda calcul vu comme monade initiale

Mémoire de Recherche

master 2

année 2005/06

Directeur : André Hirschowitz
Auteur : Julianna Zsidó

Remerciements

J'aimerais remercier tous ceux qui m'ont rendu ce stage de master 2 et ce mémoire de recherche possibles :

Tout d'abord M. Hirschowitz, mon directeur de stage, pour avoir dispensé le cours de lambda calcul au premier trimestre et éveillé mon interêt, pour m'avoir proposé un sujet de stage et pour l'avoir encadré.

Les autres professeurs de master 2 pour leur enseignement, en particulier M. Pottier pour le cours de preuves formelles et l'introduction à Coq. M. Maggesi pour ses «Coq–libraries».

Claudine pour avoir été là chaque jour pendant le troisième trimestre et pour m'avoir écoutée, conseillée et aidée. Pierre qui a toujours été interessé à discuter de mon sujet. Thu pour sa disponibilité à répondre à mes petites questions sur les catégories.

Mme Laurent et M. Thomin de la bibliothèque du laboratoire, Mme Lachkar, la secrétaire du master 2 et M. Lacroix pour m'avoir facilité l'accès aux ressources dont j'avais besoin.

Mes parents, en particulier pour m'avoir soutenu pendant mes études en France.

Julianna Zsidó

Table des matières

1 Introduction **1**

2 Lambda Calcul non-typé **3**
2.1 Monades et modules sur monades 3
2.2 Pull–back module 9
2.3 Monade exponentielle 11

3 Lambda calcul simplement typé **14**

4 Perspective **18**

Bibliographie **20**

Chapitre 1

Introduction

Quand on cherche dans les références de la littérature (par exemple [Bar85], [Bar00], [Har02], [Ber02]) la définition du lambda calcul pur, on trouve qu'il s'agit d'une théorie des fonctions. Celles–ci sont vues comme une règle de correspondance sans types : une fonction peut être appliquée même à elle–même parce qu'il n'y a pas de notions de domaine et de codomaine. Dans [Har02] ce point de vue est appelé «intentionnel» par contraste avec le point de vue «extensionnel» dans lequel on considère une fonction comme son graphe, c'est–à–dire comme un ensemble de couples.

Puis on trouve des remarques sur l'histoire du lambda calcul, contenant les noms des chercheurs associés. Les personnes liées fortement au lambda calcul sont premièrement son inventeur A. Church et ses deux élèves J. B. Rosser et S. C. Kleene qui ont contribué au développement, par exemple en prouvant l'inconsistance du système initial publié par Church. Après s'être restreint à un sous–système, la consistance était démontrée par le théorème de Church–Rosser. Les noms de H. B. Curry et A. Turing ne peuvent pas être omis dans cette liste car la logique combinatoire de Curry est reliée au lambda calcul. Ainsi que les fonctions calculables par une machine de Turing sont équivalentes à la classe des fonctions λ–calculables, définies formellement par le lambda calcul.

Ensuite, plusieurs aspects et intérêts du lambda calcul sont donnés. Par exemple l'intention de développer un fondement des mathématiques, le modèle de la calculabilité, des aspects informatiques, etc. Dans ce contexte la simplicité[1] et la puissance de la construction sont soulignées. Parfois il y a une référence au lambda calcul simplement typé qui limite la notion générale de fonction en ajoutant domaine et codomaine par l'intermédiaire du typage. Son intérêt : les termes typables sont les fonctions calculables[2]. De plus il permet d'établir la correspondance de Curry–Howard entre les propositions logiques et les termes typés.

Mais on ne trouve pas une caractérisation ou une définition conceptuelle du lambda calcul dans l'environnement mathématique classique. Un des buts de [HM05] est d'y pallier. Comme la référence de base de ce mémoire est [HM05], il essaie de contribuer à cette caractérisation. Les structures d'une monade et d'un module sur une monade sont les bonnes notions, en particulier pour la liaison d'une variable par l'abstraction. Un outil employé pour vérifier les preuves est le logiciel Coq développé à INRIA, un outil d'aide à la preuve. Ces preuves peuvent être consultées

[1] l'application, l'abstraction et la beta–réduction, qui est la règle principal du calcul

[2] C'est la thèse de Church.

sur l'internet à l'adresse suivant : http ://math.unice.fr/∼zsido/st.

Ce mémoire consiste surtout en deux parties. La première traite du lambda calcul non–typé et la deuxième du lambda calcul simplement typé. Le deuxième chapitre est basé sur [HM05]. On y introduit et étudie les structures catégorielles, les monades, les modules et les morphismes afin de caractériser le lambda calcul pur ou non–typé comme l'objet initial de la catégorie des monades exponentielles. Les preuves des deux premières sections ont été réalisées par le logiciel Coq. Le troisième chapitre est un analogue du chapitre précédent, où les différences de structure avec le cas non–typé sont soulignées. Toutes les preuves ont été réalisées par Coq. Enfin, dans le quatrième chapitre, on donne une perspective pour unifier et généraliser les deux cas précédents.

Chapitre 2

Lambda Calcul non-typé

Nous supposons un certain degré de familiarité avec le lambda calcul, en particulier avec les constructeurs des lambda termes, var, app et abs ainsi qu'avec les notions de la substition, de l'équivalence α, β et η.

2.1 Monades et modules sur monades

Nous admettons aussi la connaissance des notions générales des foncteurs et des transformations naturelles de la théorie de catégories.

Définition 2.1 (Monade) *Une* monade *d'une catégorie $\mathcal{C}$ est un triplet (T, η, μ) consistant en un endofoncteur $T : \mathcal{C} \to \mathcal{C}$ et deux transformations naturelles $\eta : \mathrm{Id} \stackrel{\cdot}{\to} T$ et $\mu : T^2 \stackrel{\cdot}{\to} T$, où* Id *désigne le foncteur identité sur $\mathcal{C}$. Ils font commuter les diagrammes suivants pour tout $X \in Ob(\mathcal{C})$:*

$$\begin{array}{ccccc} TX & \xrightarrow{T\eta_X} & T^2X & \xleftarrow{\eta_{TX}} & TX \\ & \searrow^{Id_{TX}} & \downarrow{\scriptstyle \mu_X} & \swarrow_{Id_{TX}} & \\ & & TX & & \end{array} \qquad (2.1)$$

$$\begin{array}{ccc} T^3X & \xrightarrow{\mu_{TX}} & T^2X \\ {\scriptstyle T\mu_X}\downarrow & & \downarrow{\scriptstyle \mu_X} \\ T^2X & \xrightarrow[\mu_X]{} & TX \end{array} \qquad (2.2)$$

Dans ce chapitre nous allons considérer des monades sur la catégorie des ensembles, notée Ens. Dans la suite nous allons dire que «T est une monade», quand T est le foncteur de la monade, au lieu de préciser le triplet (T, η, μ). Commençons par introduire quelques notations.

Notation 2.2

- *Pour un ensemble $X \in$ Ens on note $X^* \in$ Ens la réunion disjointe de X et d'un élément ∞_X.*
- *Pour un morphisme d'ensembles $f : X \to Y$ on note $f^* : X^* \to Y^*$ tel que $f^* \mid_X = f$ et $f(\infty_X) = \infty_Y$.*
- *L'injection canonique de $X \in$ Ens : $X \hookrightarrow X^*$ sera noté par $\imath_X$.*

Exemple 2.3 *Le foncteur* SLC*, associant à chaque ensemble de variables* $X \in \mathrm{Ens}$ *l'ensemble des lambda termes non–typés* $\mathsf{SLC}\,X$*, est une monade avec* $\eta = \mathrm{var}$ *et* μ *correspondant à la substitution. Pour vérifier la fonctorialité de* SLC*, on se donne un morphisme* $f : X \to Y$ *avec* $X, Y \in \mathrm{Ens}$ *et on construit* $\mathsf{SLC}\,f : \mathsf{SLC}\,X \to \mathsf{SLC}\,Y$ *récursivement :*

$$T \in \mathsf{SLC}\,X \mapsto \begin{cases} \mathrm{var}(f(x)) & \text{si } T = \mathrm{var}\,x \text{ avec } x \in X \\ \mathrm{app}(\mathsf{SLC}\,f(T_1), \mathsf{SLC}\,f(T_2)) & \text{si } T = \mathrm{app}(T_1, T_2) \text{ avec } T_1, T_2 \in \mathsf{SLC}\,X \\ \mathrm{abs}(\mathsf{SLC}(f^*)(T')) & \text{si } T = \mathrm{abs}(T') \text{ avec } T' \in \mathsf{SLC}(X^*) \end{cases}$$

D'après cette construction, on a clairement $\mathsf{SLC}\,\mathrm{Id}_X = \mathrm{Id}_{\mathsf{SLC}\,X}$ *et* $\mathsf{SLC}(f_1 \circ f_2) = \mathsf{SLC}\,f_1 \circ \mathsf{SLC}\,f_2$ *pour deux morphismes* f_1 *et* f_2 *composables. La fonctorialité de* SLC *est une sorte de renommage des variables selon l'application* f*.*

D'une manière équivalente, le foncteur LC *associant à chaque ensemble de variables* $X \in \mathrm{Ens}$ *l'ensemble des lambda termes modulo* α*–,* β*– et* η*–équivalence,* $\mathsf{LC}\,X$*, est une monade.*

Définition 2.4 (Monade derivée) *Étant donnée une monade* R*, on construit sa* derivée R' *en posant pour tout* $X \in \mathrm{Ens}$ *:*

$$R'X := R(X^*)$$

Pour tout morphisme d'ensembles $f : X \to Y$ *on pose :*

$$R'f := R(f^*)$$

Remarque 2.5 *La monade derivée* R' *d'une monade* R *est une monade.*

Preuve. On pose pour tout $X \in \mathrm{Ens}$:

$$\eta'_X := R\imath_X \circ \eta_X$$

$$\mu'_X := \mu_{X^*} \circ Rd_X$$

où $d_X : (R'X)^* \to R'X$ tel que $d_X \mid_{R'X} = \mathrm{Id}_{R'X}$ et $d_X(\infty_{R'X}) = \eta_{X^*}(\infty_X)$. Pour démontrer II. de (2.1) pour R', on remplace les définitions de η' et de μ' et on trouve le diagramme suivant :

$$\begin{array}{ccccc} R((R'X)^*) & \xleftarrow{R\imath_{R(X^*)}} & RR(X^*) & \xleftarrow{\eta_{R(X^*)}} & R(X^*) \\ \downarrow{\scriptstyle Rd_X} & \swarrow{\scriptstyle \mathrm{Id}_{RR(X^*)}} & \downarrow{\scriptstyle \mu_{X^*}} & \swarrow{\scriptstyle \mathrm{Id}_{R(X^*)}}{\scriptstyle I.} & \\ RR(X^*) & & \xrightarrow[\mu_{X^*}]{} R(X^*) & & \end{array} \tag{2.3}$$

Nous remarquons que la composition $Rd_X \circ R\imath_{R(X^*)} = R(d_X \circ \imath_{R(X^*)})$ est $R\,\mathrm{Id}_{R(X^*)} = \mathrm{Id}_{RR(X^*)}$ car les restrictions de d_X et de $\imath_{R(X^*)}$ sur $R(X^*)$ sont l'identité sur $R(X^*)$ d'après les définitions. Donc le carré dans (2.3) commute ; le triangle I. aussi, car il est la partie II. de (2.1) appliquée à R et X^*. Pour I. de (2.1) pour R' on trouve le diagramme :

$$\begin{array}{ccc} R(X^*) & \xrightarrow{R((\eta'_X)^*)} & R((R(X^*))^*) \\ \downarrow{\scriptstyle \mathrm{Id}_{R(X^*)}} & \searrow{\scriptstyle R\eta_{X^*}} & \downarrow{\scriptstyle Rd_X} \\ R(X^*) & \xleftarrow[\mu_{X^*}]{} & RR(X^*) \end{array} \tag{2.4}$$

Le triangle à gauche dans (2.4) commute grâce à (2.1) appliqué à R et X^*. L'autre commute parce que $d_X \circ (\eta'_X)^* = \eta_{X^*}$. Cette égalité est vraie :

$$\infty_X \xrightarrow{(\eta'_X)^*} \infty_{R(X^*)} \xrightarrow{d_X} \eta_{X^*}(\infty_X)$$

et pour $x \in X$ on a

$$x \xrightarrow{\eta_X} \eta_X(x) \xrightarrow{R\imath_X} \eta_{X^*}(x) \xrightarrow{d_X} \eta_{X^*}(x)$$

Il reste à voir (2.2) pour R'.

$$\begin{array}{ccccc} R'R'R'X & \xrightarrow{Rd_{R'X}} & RR'R'X & \xrightarrow{\mu_{(R'X)^*}} & R'R'X \\ \Big\downarrow{\scriptstyle R((\mu'_x)^*)} & & \Big\downarrow{\scriptstyle RRd_X} \quad I. & & \Big\downarrow{\scriptstyle Rd_X} \\ & III. & RRR'X & \xrightarrow{\mu_{R'X}} & RR'X \\ & & \Big\downarrow{\scriptstyle R\mu_{X^*}} & & \Big\downarrow{\scriptstyle \mu_{X^*}} \\ R'R'X & \xrightarrow[Rd_X]{} & RR'X & \xrightarrow[\mu_{X^*}]{} & R'X \end{array} \tag{2.5}$$

Le carré I. de (2.5) commute parce que μ est une transformation naturelle et le carré II. parce que on a (2.2) pour R et X^*. La côté gauche, III. commute car le diagramme suivant commute :

$$\begin{array}{ccc} (R'R'X)^* & \xrightarrow{d_{R'X}} & R((R'X)^*) \\ & & \Big\downarrow{\scriptstyle Rd_X} \\ \Big\downarrow{\scriptstyle (\mu'_X)^*} & & RR(X^*) \\ & & \Big\downarrow{\scriptstyle \mu_{X^*}} \\ (R'X)^* & \xrightarrow[d_X]{} & R(X^*) \end{array}$$

On suit les flèches dans les deux cas $x = \infty_{R'R'X}$ ou $x \in R'R'X$:

$$\infty_{R'R'X} \xrightarrow{d_{R'X}} \eta_{(R'X)^*}(\infty_{R'X}) \xrightarrow{Rd_X} R\eta_{X^*}(\infty_X) \xrightarrow{\mu_{X^*}} \eta_{X^*}(\infty_X)$$

$$\infty_{R'R'X} \xrightarrow{(\mu'_X)^*} \infty_{R'X} \xrightarrow{d_X} \eta_{X^*}(\infty_X)$$

Pour tout $x \in R'R'X$:

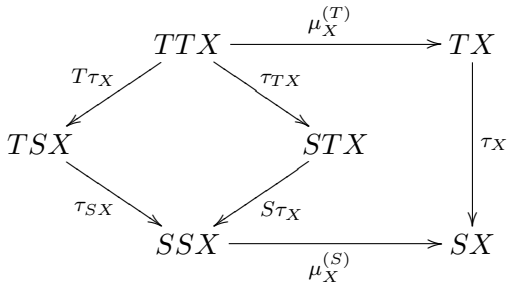
$$x \xrightarrow{d_{R'X}} x \xrightarrow{\mu'_X} \mu'_X(x)$$

$$x \xrightarrow{(\mu'_X)^*} \mu'_X(x) \xrightarrow{d_X} \mu'_X(x)$$

□

Définition 2.6 (Morphisme de monade) *Soient $(T, \eta^{(T)}, \mu^{(T)})$ et $(S, \eta^{(S)}, \mu^{(S)})$ deux monades. Un* morphisme de monades *τ est une transformation naturelle $T \xrightarrow{\cdot} S$ qui est compatible avec $\eta^{(T)}$, $\eta^{(S)}$, $\mu^{(T)}$ et $\mu^{(S)}$. Il fait alors commuter les diagrammes suivants pour tout $X \in$ Ens :*

$$\begin{array}{ccc} X & \xrightarrow{\eta^{(T)}_X} & TX \\ & \searrow^{\eta^{(S)}_X} & \Big\downarrow{\scriptstyle \tau_X} \\ & & SX \end{array} \tag{2.6}$$



(2.7)

Définition 2.7 (Module) *Soit R une monade. Un* module *sur R est un couple (M, σ) où M est un endofoncteur Ens $\to$* Ens *muni d'une transformation naturelle, appelé* substitution*, $\sigma : MR \dot{\to} M$ vérifiant le diagramme commutatif suivant pour tout $X \in$* Ens *:*

$$\begin{array}{ccc} MRRX & \xrightarrow{\sigma_{RX}} & MRX \\ {\scriptstyle M\mu_X}\downarrow & & \downarrow{\scriptstyle \sigma_X} \\ MRX & \xrightarrow[\sigma_X]{} & MX \end{array} \tag{2.8}$$

Remarque 2.8 (Module tautologique) *Une monade R est un module sur elle–même, appelé* module tautologique*.*

Preuve. $\sigma^{(R)} := \mu$. (2.8) est donné par (2.2). □

Un module est une sorte de généralisation d'une monade. Il est moins structuré, et la notion de substitution est plus faible que la substitution μ d'une monade.

Définition 2.9 (Morphisme de module) *Soient R une monade, $(M, \sigma^{(M)})$ et $(N, \sigma^{(N)})$ deux R–modules. Un* morphisme de module *F est une transformation naturelle compatible avec $\sigma^{(M)}$ et $\sigma^{(N)}$. Il fait alors commuter le diagramme suivant pour tout $X \in$* Ens *:*

$$\begin{array}{ccc} MRX & \xrightarrow{\sigma_X^{(M)}} & MX \\ {\scriptstyle F_{RX}}\downarrow & & \downarrow{\scriptstyle F_X} \\ NRX & \xrightarrow[\sigma_X^{(N)}]{} & NX \end{array} \tag{2.9}$$

Définition 2.10 (Module derivé) *Soient R une monade et M un R–module. On construit le* module derivé *M' en posant pour tout $X \in$* Ens *:*

$$M'(X) := M(X^*)$$

Pour tout $f : X \to Y$ morphisme d'ensembles on pose :

$$M'f := M(f^*)$$

Remarque 2.11 *Le module derivé M' d'un R–module M est un R–module.*

Preuve. On pose pour tout $X \in \mathrm{Ens}$:

$$\sigma'_X := \sigma_{X^*} \circ Mg_X$$

où $g : (RX)^* \to R(X^*)$ tel que $g \mid_{RX} = R\imath_X$ et $g(\infty_{RX}) = \eta_{X^*}(\infty_X)$. Il reste à voir le diagramme analogue à (2.8).

$$\begin{array}{ccccc}
M((RRX)^*) & \xrightarrow{Mg_{RX}} & MR((RX)^*) & \xrightarrow{\sigma_{(RX)^*}} & M((RX)^*) \\
\Big\downarrow{\scriptstyle M(\mu_X^*)} & & \Big\downarrow{\scriptstyle MRg_X} \quad I. & & \Big\downarrow{\scriptstyle Mg_X} \\
 & III. & MRR(X^*) & \xrightarrow{\sigma_{R(X^*)}} & MR(X^*) \\
 & & \Big\downarrow{\scriptstyle M\mu_{X^*}} & II. & \Big\downarrow{\scriptstyle \sigma_{X^*}} \\
M((RX)^*) & \xrightarrow[Mg_X]{} & MR(X^*) & \xrightarrow[\sigma_{X^*}]{} & M(X^*)
\end{array} \tag{2.10}$$

Le carré I. de (2.10) commute car σ est une transformation naturelle et le carré II. car on a (2.8) pour M et X^*. Le carré III. commute parce que le diagramme suivant commute :

$$\begin{array}{ccc}
(RRX)^* & \xrightarrow{g_{RX}} & R((RX)^*) \\
\Big\downarrow{\scriptstyle (\mu_X)^*} & & \Big\downarrow{\scriptstyle Rg_X} \\
 & & RR(X^*) \\
 & & \Big\downarrow{\scriptstyle \mu_{X^*}} \\
(RX)^* & \xrightarrow[g_X]{} & R(X^*)
\end{array}$$

On suit les flèches dans les deux cas $x = \infty_{RRX}$ ou $x \in RRX$:

$$\infty_{RRX} \xrightarrow{g_{RX}} \eta_{(RX)^*}(\infty_{RX}) \xrightarrow{Rg_X} R\eta_{X^*}(\infty_X) \xrightarrow{\mu_{X^*}} \eta_{X^*}(\infty_X)$$

$$\infty_{RRX} \xrightarrow{(\mu_X)^*} \infty_{RX} \xrightarrow{g_X} \eta_{X^*}(\infty_X)$$

Pour tout $x \in RRX$:

$$x \xrightarrow{g_{RX}} x \xrightarrow{Rg_X} x \xrightarrow{\mu_{X^*}} \mu_{X^*}(x)$$

$$x \xrightarrow{(\mu_X)^*} \mu_X(x) \xrightarrow{g_X} \mu_{X^*}(x)$$

□

Lemme 2.12 *L'injection canonique $M\imath : M \hookrightarrow M'$ est un morphisme de module.*

Preuve. Il faut voir que le diagramme analogue à (2.9) commute pour tout $X \in \mathrm{Ens}$:

$$\begin{array}{ccccc}
MRX & & \xrightarrow{\sigma_X} & & MX \\
\Big\downarrow{\scriptstyle M\imath_{RX}} & \searrow^{MR\imath_X} & & & \Big\downarrow{\scriptstyle M\imath_X} \\
 & I. & & II. & \\
M'RX & \xrightarrow[Mg_X]{} & MR(X^*) & \xrightarrow[\sigma_{X^*}]{} & M(X^*)
\end{array} \tag{2.11}$$

En remarquant que pour tout $x \in RX$ on a $g_X \circ \imath_{RX}(x) = R\imath_X(x)$, car

$$x \xrightarrow{\imath_{RX}} x \xrightarrow{g_X} R\imath_X(x)$$

le triangle I. de (2.11) commute ; le II. commute aussi, puisque σ est une transformation naturelle. □

Remarque 2.13 (produit de modules) *Soient R une monade, $(M, \sigma^{(M)})$ et $(N, \sigma^{(N)})$ deux R–modules. Le produit (cartésien) de foncteurs $M \times N$ est aussi un R–module.*

Preuve. On a pour tout $X \in \text{Ens}$: $M \times NX = MX \times NX$ et pour tout $f : X \to Y$: $M \times Nf = Mf \times Nf$. On pose pour tout $X \in \text{Ens}$:

$$\sigma_X^{(M\times N)} := \sigma_X^{(M)} \times \sigma_X^{(N)}$$

Le diagramme suivant commute pour tout $X \in \text{Ens}$ car on a (2.8) pour M et N.

$$\begin{array}{ccc} M \times NRRX & \xrightarrow{\sigma_{RX}^{(M\times N)}} & M \times NRX \\ \Big\downarrow{\scriptstyle M\times N\mu_X} & & \Big\downarrow{\scriptstyle \sigma_X^{(M\times N)}} \\ M \times NRX & \xrightarrow[\sigma_X^{(M\times N)}]{} & M \times NX \end{array}$$

□

Exemple 2.14 *Le constructeur* app *des lambda termes est un morphisme de LC–modules $\mathsf{LC} \times \mathsf{LC} \to \mathsf{LC}$. On rappelle la définition de la substitution sur le constructeur* app *; pour tout $M, N, T \in \mathsf{LC}\,X$ et $x \in X$ on a :*

$$(\text{app}(M, N))[x \leftarrow T] = \text{app}(M[x \leftarrow T], N[x \leftarrow T])$$

c'est–à–dire que le diagramme analogue à (2.9) commute pour tout $X \in \text{Ens}$:

$$\begin{array}{ccc} \mathsf{LC} \times \mathsf{LC}(\mathsf{LC}\,X) & \longrightarrow & \mathsf{LC} \times \mathsf{LC}\,X \\ \Big\downarrow{\scriptstyle \text{app}_{\mathsf{LC}\,X}} & & \Big\downarrow{\scriptstyle \text{app}_X} \\ \mathsf{LC}\,\mathsf{LC}\,X & \longrightarrow & \mathsf{LC}\,X \end{array}$$

Exemple 2.15 *Le constructeur* abs *des lambda termes est un morphisme de LC–modules $\mathsf{LC}' \to \mathsf{LC}$. On rappelle la définition de la substitution sur le constructeur* abs *; pour tout $M \in \mathsf{LC}(X^*)$, $T \in \mathsf{LC}\,X$ et $x \in X$ on a*

$$(\text{abs}(M))[x \leftarrow T] = \text{abs}(M[x \leftarrow T'])$$

où $T' = T$ mais $T' \in \mathsf{LC}(X^*)$ d'après l'injection canonique d'un module dans son derivé. Donc le diagramme analogue à (2.9) commute :

$$\begin{array}{ccc} \mathsf{LC}'\,\mathsf{LC}\,X & \longrightarrow & \mathsf{LC}'\,X \\ \Big\downarrow{\scriptstyle \text{abs}_{\mathsf{LC}\,X}} & & \Big\downarrow{\scriptstyle \text{abs}_X} \\ \mathsf{LC}\,\mathsf{LC}\,X & \longrightarrow & \mathsf{LC}\,X \end{array}$$

Nous avons caracterisé les trois constructeurs des lambda termes et la substitution dans le langage des monades et modules.

2.2 Pull–back module

Définition 2.16 (pull–back module) *Soient A, B deux monades, $f : A \to B$ un morphisme de monades et $(M, \sigma^{(B)})$ un B–module. On construit le* pull–back module *de M en posant pour tout $X \in$ Ens :*

$$f^*MX := MX$$

et la substitution :

$$\sigma^{(A)}_X := \sigma^{(B)}_X \circ Mf_X$$

Lemme 2.17 *Le pull–back f^*M du B–module M est un A–module, avec les notations de la définition 2.16.*

Preuve. Montrons que le diagramme analogue à (2.8) pour f^*M et A commute pour tout $X \in$ Ens. On remplace f^*M et $\sigma^{(A)}$ par les définitions et on obtient le diagramme suivant :

$$\begin{array}{ccccc} MAAX & \xrightarrow{Mf_{AX}} & MBAX & \xrightarrow{\sigma^{(B)}_{AX}} & MAX \\ \downarrow{\scriptstyle M\mu^{(A)}_X} & & \downarrow{\scriptstyle MBf_X} \quad I. & & \downarrow{\scriptstyle Mf_X} \\ & & MBBX & \xrightarrow{\sigma^{(B)}_{BX}} & MBX \\ & III. & \downarrow{\scriptstyle M\mu^{(B)}_X} & II. & \downarrow{\scriptstyle \sigma^{(B)}} \\ MAX & \xrightarrow[Mf_X]{} & MBX & \xrightarrow[\sigma^{(B)}_X]{} & MX \end{array} \tag{2.12}$$

Le carré I. de (2.12) commute parce que $\sigma^{(B)}$ est une transformation naturelle et II. parce que (2.8) est vrai pour M et B. Le carré à gauche, III. commute grâce à (2.7). □

Remarque 2.18 *Soit R une monade. La classe des R–modules et les morphismes entre les R–modules forment une sous–catégorie de* Funct(Ens, Ens)*, la catégorie des foncteurs sur* Ens*. La catégorie des R–modules sera notée* Mod_R*.*

Preuve. On va démontrer que le foncteur $F : \mathrm{Mod}_R \to \mathrm{Funct}(\mathrm{Ens}, \mathrm{Ens})$:

$$(M, \sigma^{(M)}) \mapsto M$$

$$((N, \sigma^{(N)}) \stackrel{\cdot}{\to} (P, \sigma^{(P)})) \mapsto (N \stackrel{\cdot}{\to} P)$$

vérifie la propriété suivante pour tout φ, ψ morphismes de R–modules :

$$F\varphi = F\psi \Rightarrow \varphi = \psi$$

Soient donc $\varphi : (M, \sigma^{(M)}) \stackrel{\cdot}{\to} (N, \sigma^{(N)})$ et $\psi : (T, \sigma^{(T)}) \stackrel{\cdot}{\to} (S, \sigma^{(S)})$ deux morphismes de R–modules, c'est–à–dire $M \stackrel{\cdot}{\to} N$ et $T \stackrel{\cdot}{\to} S$ deux transformations naturelles qui vérifient chacune (2.9). On suppose que $F\varphi = F\psi$ ou encore, pour tout $X \in$ Ens : $(F\varphi)_X = (F\psi)_X$. D'après la définition de F on a $\varphi_X = \psi_X$ vus comme des transformations naturelles (sans la compatibilité

avec les substitutions). Donc $M = T$ et $N = T$. En remplaçant φ_X par ψ_X dans le carré commutatif (2.9) pour φ, on trouve pour tout $X \in \mathrm{Ens}$:

$$\begin{array}{ccc} MRX & \xrightarrow{\sigma_X^{(M)}} & MX \\ {\scriptstyle \psi_{RX}}\downarrow & & \downarrow{\scriptstyle \psi_X} \\ NRX & \xrightarrow[\sigma_X^{(N)}]{} & NX \end{array}$$

commute et analoguement le carré (2.9) pour ψ :

$$\begin{array}{ccc} MRX & \xrightarrow{\sigma_X^{(T)}} & MX \\ {\scriptstyle \varphi_{RX}}\downarrow & & \downarrow{\scriptstyle \varphi_X} \\ NRX & \xrightarrow[\sigma_X^{(S)}]{} & NX \end{array}$$

Ces carrés commutatifs impliquent $\varphi = \psi$ vus comme morphismes de R–modules. □

Définition 2.19 *Soient A, B deux monades, $f : A \to B$ un morphisme de monade et M un B–module. On définit le foncteur pull-back $f^* : \mathrm{Mod}_B \to \mathrm{Mod}_A$.*

En effet, d'après la définition 2.16, $f^* : M \to f^*M$ pour chaque $M \in \mathrm{Mod}_B$. Pour $g : M_1 \to M_2$ un morphisme de B–modules, où M_1 et M_2 sont munis d'une substitution par rapport à B, on a $f^*g : M_1 \to M_2$ comme $f^*M_i = M_i$ pour $i = 1, 2$. Donc $f^*g = f$ mais ici, M_1 et M_2 sont munis d'une substitution par rapport à A et on demande à f^*g la compatibilité avec la substitution par rapport à A. f^*g est un morphisme de A–modules car le diagramme analogue à (2.9) commute pour tout $X \in \mathrm{Ens}$:

$$\begin{array}{ccccc} M_1AX & \xrightarrow{M_1 f_X} & M_1BX & \xrightarrow{\sigma_X^{(M_1,B)}} & M_1X \\ {\scriptstyle g^*f_{AX}}\downarrow & II. & \downarrow{\scriptstyle g^*f_{BX}} & I. & \downarrow{\scriptstyle g^*f_X} \\ M_2AX & \xrightarrow[M_2 f_X]{} & M_2BX & \xrightarrow[\sigma_X^{(M_2,B)}]{} & M_2X \end{array} \tag{2.13}$$

Le carré I. de (2.13) commute parce que f est un morphisme de B–modules et le carré II. commute parce que f est une transformation naturelle. Les deux axiomes de fonctorialité sont évidemment satisfaits : Pour $M \in \mathrm{Mod}_B$ on a $f^* \mathrm{Id}_M = \mathrm{Id}_{f^*M}$ et pour g_1, g_2 deux morphismes de B–modules composables on a $f^*(g_2 \circ g_1) = f^*g_2 \circ f^*g_1$. Cela justifie d'avoir défini le pull-back comme un foncteur.

Proposition 2.20 *Soient A et B deux monades. Pour chaque $f : A \to B$ morphisme de monade, on a un morphisme de A–modules $f : A \to f^*B$ où on identifie A et B avec leurs modules tautologiques.*

Preuve. Pour vérifier que le carré (2.9) commute, on remplace les définitions de f^*B, $\sigma^{(A)}$ du module B et on utilise le fait que pour le module tautologique $\sigma = \mu$, on trouve le diagramme

suivant.

$$\begin{array}{ccccc} AAX & & \xrightarrow{\mu_X^{(A)}} & & AX \\ \downarrow{\scriptstyle f_{AX}} & & & & \downarrow{\scriptstyle f_X} \\ BAX & \xrightarrow[Bf_X]{} & BBX & \xrightarrow[\mu_X^{(B)}]{} & BX \end{array}$$

Ce diagramme commute grâce à (2.7). □

2.3 Monade exponentielle

Définition 2.21 (monade exponentielle) *Une* monade exponentielle R *est le couple* (R, α) *où* R *est une monade et* α *est un isomorphisme entre le module tautologique* R *et son derivé* R'.

Notation 2.22 *On note pour tout* $x \in \mathsf{LC}\,X$ *:* $\mathrm{app1}(x) := \mathrm{app}(x, \mathrm{var}(\infty_X))$.

Remarque 2.23 $\mathrm{app1} : \mathsf{LC} \to \mathsf{LC}'$ *est un morphisme de* LC*–modules.*

Preuve. Pour tout $M, T \in \mathsf{LC}\,X$ et $x \in X$ on a :

$$\begin{aligned} (\mathrm{app1}(M))[x \leftarrow T] &= (\mathrm{app}(M, \mathrm{var}(\infty_X)))[x \leftarrow T] \\ &= \mathrm{app}(M[x \leftarrow T], \mathrm{var}(\infty_X)[x \leftarrow T]) \\ &= \mathrm{app}(M[x \leftarrow T], \mathrm{var}(\infty_X)) \\ &= \mathrm{app1}(M[x \leftarrow T]) \end{aligned}$$

c'est–à–dire que le diagramme suivant commute :

$$\begin{array}{ccc} \mathsf{LC}\,\mathsf{LC}\,X & \longrightarrow & \mathsf{LC}\,X \\ \downarrow{\scriptstyle \mathrm{app1}_{\mathsf{LC}\,X}} & & \downarrow{\scriptstyle \mathrm{app1}_X} \\ \mathsf{LC}'\,\mathsf{LC}\,X & \longrightarrow & \mathsf{LC}'\,X \end{array}$$

□

Exemple 2.24 *La monade* LC *est une monade exponentielle avec* $\alpha = \mathrm{app1}$ *et inverse* abs. *D'après l'équivalence* β *on a pour tout* $x \in \mathsf{LC}'\,X$ *:* $\mathrm{app1}(\mathrm{abs}(x)) = x$ *et d'après l'équivalence* η *on a pour tout* $x \in \mathsf{LC}\,X$ *:* $\mathrm{abs}(\mathrm{app1}(x)) = x$.

Définition 2.25 (morphisme de monades exponentielles) *Soient* $(T, \alpha^{(T)})$ *et* $(S, \alpha^{(S)})$ *deux monades exponentielles. Un* morphisme de monades exponentielles $\nu : T \dot{\to} S$ *est un morphisme de monades qui est compatible avec* $\alpha^{(T)}$, $\alpha^{(S)}$ *et leurs inverses.* ν *fait alors commuter les diagrammes suivants pour tout* $X \in \mathrm{Ens}$ *:*

$$\begin{array}{ccc} TX & \xrightarrow{\nu_X} & SX \\ {\scriptstyle \alpha_X^{(T)-1}}\uparrow\downarrow{\scriptstyle \alpha_X^{(T)}} & & {\scriptstyle \alpha_X^{(S)-1}}\uparrow\downarrow{\scriptstyle \alpha_X^{(S)}} \\ T'X & \xrightarrow[\nu_{X^*}]{} & S'X \end{array} \tag{2.14}$$

Théorème 2.26 *La monade exponentielle* LC *est l'objet initial dans la catégorie des monades exponentielles.*

Preuve. Ici nous allons voir les idées générales de la preuve, pour les détails il est conseillé de consulter la preuve réalisée par Coq.

Les monades exponentielles avec les morphismes de monades exponentielles forment clairement une catégorie. Pour chaque monade exponentielle, on a le morphisme unité et il existe une composition des morphismes, qui est la composition verticale des transformations naturelles. Cette composition est trivialement compatible avec les α. L'associativité de la composition découle de l'associativité des morphismes dans la catégorie des ensembles et de même la composition avec l'unité.

Soit $X \in \mathrm{Ens}$ un ensemble fixé de variables et $((M, \eta, \mu), \alpha)$ une monade exponentielle. Pour μ, la substitution on utilisera la notation usuelle de la substitution du lambda calcul. Nous allons construire un morphisme de monade exponentielle $\varphi : \mathsf{LC} \to M$ et voir qu'il est unique.

D'abord nous construisons inductivement une application $\psi_X : \mathsf{SLC}\, X \to MX$. Pour tout $T \in \mathsf{SLC}\, X$ on pose :

$$\psi_X(T) := \begin{cases} \eta_X(x) & \text{si } T = \mathrm{var}(x) \text{ avec } x \in X \\ \alpha_X(\psi_X(M))[\infty_X \leftarrow \psi_X(N)] & \text{si } T = \mathrm{app}(M, N) \text{ avec } M, N \in \mathsf{SLC}\, X \\ \alpha_X^{-1}(\psi_{X^*}(M)) & \text{si } T = \mathrm{abs}(M) \text{ avec } M \in \mathsf{SLC}'\, X \end{cases}$$

Nous allons voir que pour tout $M, N \in \mathsf{SLC}\, X$ tels que $M =_{\alpha\beta\eta} N$, on a $\psi_X(M) = \psi_X(N)$.

Pour un terme M contenant un β–redex, c'est–à–dire un sous–terme de la forme $\mathrm{app}(\mathrm{abs}(T_1), T_2)$ avec $T_1 \in \mathsf{SLC}'\, Y$, $T_2 \in \mathsf{SLC}\, Y$, où $Y \subseteq X$, on a :

$$\begin{aligned} \psi_Y(\mathrm{app}(\mathrm{abs}(T_1), T_2)) &= \alpha_X(\psi_Y(\mathrm{abs}(T_1)))[\infty_Y \leftarrow \psi_Y(T_2)] \\ &= \alpha_X(\alpha_X^{-1}(\psi_{Y^*}(T_1)))[\infty_Y \leftarrow \psi_Y(T_2)] \\ &= \psi_{Y^*}(T_1)[\infty_Y \leftarrow \psi_Y(T_2)] \\ &= \psi_Y(T_1[\infty_Y \leftarrow \psi_Y(T_2)]) \end{aligned}$$

Dans le calcul on a utilisé la compatibilité entre ψ et la substitution qui est prouvé en Coq. Donc si $M \to_\beta N$, on a $\psi_X(M) = \psi_X(N)$ et $\psi_X(N) = \psi_X(M)$. Également si $M_1 \to_\beta N$ et $M_2 \to_\beta N$, on a $\psi_X(M_1) = \psi_X(N)$ et $\psi_X(M_2) = \psi_X(N)$ et aussi $\psi_X(M_1) = \psi_X(M_2)$. Cela montre la compatibilité avec l'équivalence β.

Pour un terme M contenant un η–redex, c'est–à–dire un sous–terme de la forme $\mathrm{abs}(\mathrm{app}(T, \mathrm{var}(\infty_Y)))$ avec $T \in \mathsf{SLC}\, Y$ où $Y \subseteq X$, on a :

$$\begin{aligned} \psi_{Y^*}(\mathrm{abs}(\mathrm{app}(T, \mathrm{var}(\infty_Y)))) &= \alpha_{Y^*}^{-1}(\psi_{Y^*}(\mathrm{app}(T, \mathrm{var}(\infty_Y)))) \\ &= \alpha_{Y^*}^{-1}(\alpha_{Y^*}(\psi_{Y^*}(T))[\infty_Y \leftarrow \psi_{Y^*}(\mathrm{var}(\infty_Y))]) \\ &= \alpha_{Y^*}^{-1}(\alpha_{Y^*}(\psi_{Y^*}(T))[\infty_Y \leftarrow \eta_{Y^*}(\infty_Y)]) \\ &= \alpha_{Y^*}^{-1}(\alpha_{Y^*}(\psi_{Y^*}(T))) \\ &= \psi_{Y^*}(T) \end{aligned}$$

Donc si $M \to_\eta N$, on a $\psi_X(M) = \psi_X(N)$. Cela montre par une raisonnement analogue la compatibilité avec l'équivalence η.

On pose $\varphi_X : \mathsf{LC}\, X \to MX, \overline{T} \mapsto \psi_X(T)$, où T est un représentant de la classe $\overline{T}$. φ_X est bien défini car on vient de voir que ψ est constant sur chaque classe d'équivalence par rapport à l'équivalence β et η. Pour l'équivalence α c'est trivial. φ est compatible avec η et var d'après

la définition de ψ, il est également compatible avec la substitution, donc φ est un morphisme de monades. Pour voir qu'il est même un morphisme de monades exponentielles, la compatibilité de ψ et φ avec α et app1 :

$$\psi_{X^*}(\mathrm{app1}(T)) = \alpha_X(\psi_X(T))$$

était verifiée en Coq ; la compatibilité avec α^{-1} et abs est déjà donnée par la définition de ψ.

Soit $\rho : \mathsf{LC} \to M$ un morphisme de monades exponentielles et $\overline{T} \in \mathsf{LC}\, X$. Il existe alors un représentant $T \in \mathsf{SLC}\, X$ de $\overline{T}$. On va voir par induction structurelle que $\rho_X(\overline{T}) = \psi_X(T)$.

- Si $T = \mathrm{var}\, x$ avec $x \in X$, alors $\psi_X(\mathrm{var}\, x) = \eta_X(x)$. Comme ρ est en particulier un morphisme de monades, d'après (2.6) on a $\rho_X(\overline{\mathrm{var}\, x}) = \eta_X(x)$.
- Si $T = \mathrm{app}(T_1, T_2)$ avec $T_1, T_2 \in \mathsf{SLC}\, X$, alors on a deux hypothèses de récurrence : $\psi_X(T_i) = \rho_X(\overline{T_i})$ pour $i = 1, 2$. Il faut voir :

  $$\rho_X(\overline{\mathrm{app}(T_1, T_2)}) = \psi_X(\mathrm{app}(T_1, T_2))$$

  ou une forme équivalente en reformulant app par l'aide de app1 :

  $$\rho_X(\overline{\mathrm{app1}(T_1)[\infty_X \leftarrow T_2]}) = \psi_X(\mathrm{app1}(T_1)[\infty_X \leftarrow T_2])$$

  en utilisant les propriétés (prouvées en Coq) des classes d'équivalences et la compatibilité entre ρ et la substitution, donnée par (2.7), on trouve :

  $$\begin{aligned}\rho_X(\overline{\mathrm{app1}(T_1)[\infty_X \leftarrow T_2]}) &= \rho_X(\overline{\mathrm{app1}(T_1)}[\infty_X \leftarrow \overline{T_2}]) \\ &= \rho_{X^*}(\overline{\mathrm{app1}(T_1)})[\infty_X \leftarrow \rho_X(\overline{T_2})] \\ &= \rho_{X^*}(\mathrm{app1}(\overline{T_1}))[\infty_X \leftarrow \rho_X(\overline{T_2})]\end{aligned}$$

  Comme ρ est un morphisme de monades exponentielles on a grâce à (2.14) :

  $$\rho_{X^*}(\mathrm{app1}(\overline{T_1}))[\infty_X \leftarrow \rho_X(\overline{T_2})] = \alpha_X(\rho_X(\overline{T_1}))[\infty_X \leftarrow \rho_X(\overline{T_2})]$$

  En utilisant les hypothèses de récurrence :

  $$\alpha_X(\rho_X(\overline{T_1}))[\infty_X \leftarrow \rho_X(\overline{T_2})] = \alpha_X(\psi_X(T_1))[\infty_X \leftarrow \psi_X(T_2)]$$

  En utilisant les compatibilités entre ψ et app1 d'une part, entre ψ et la substitution d'autre part :

  $$\begin{aligned}\alpha_X(\psi_X(T_1))[\infty_X \leftarrow \psi_X(T_2)] &= \psi_{X^*}(\mathrm{app1}(T_1))[\infty_X \leftarrow \psi_X(T_2)] \\ &= \psi_X(\mathrm{app1}(T_1)[\infty_X \leftarrow T_2])\end{aligned}$$

- Si $T = \mathrm{abs}(T')$ avec $T' \in \mathsf{SLC}'\, X$, alors on a un hypothèse de récurrence : $\psi_{X^*}(T') = \rho_{X^*}(\overline{T'})$. Montrons :

  $$\rho_X(\overline{\mathrm{abs}(T')}) = \psi_X(\mathrm{abs}(T'))$$

  En utilisant des propriétés des classes d'équivalences, (2.14), l'hypothèse de récurrence et le définition de ψ, on trouve :

  $$\begin{aligned}\rho_X(\overline{\mathrm{abs}(T')}) &= \rho_X(\mathrm{abs}(\overline{T'})) \\ &= \alpha_X^{-1}(\rho_{X^*}(\overline{T'})) \\ &= \alpha_X^{-1}(\psi_{X^*}(T')) \\ &= \psi_X(\mathrm{abs}(T'))\end{aligned}$$

□

Chapitre 3

Lambda calcul simplement typé

Dans le lambda calcul simplement typé on considère des lambda termes typés, c'est–à–dire des lambda termes munis de types.

L'ensemble de types $\mathcal{T} \in \mathrm{Ens}$ est défini inductivement par un type de base $*$ et un constructeur $\Rightarrow$:

$$\mathcal{T} = * \mid \mathcal{T} \Rightarrow \mathcal{T}$$

À partir d'un ensemble de variables typés, $\Gamma_X = ((x_i : t_i))_{x_i \in X}$, où $X \in \mathrm{Ens}$ est l'ensemble des variables et $t_i \in \mathcal{T}$, on associe aux lambda termes leurs types d'après les règles suivantes :

$$\frac{(x : t) \in \Gamma_X}{\Gamma_X \vdash x : t}\ \mathrm{var}$$

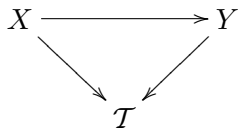
$$\frac{\Gamma_X \vdash x_1 : t_1 \Rightarrow t_2; \Gamma_X \vdash x_2 : t_1}{\Gamma_X \vdash \mathrm{app}(x_1, x_2) : t_2}\ \mathrm{app}$$

$$\frac{\Gamma_{X^*} \vdash x : t}{\Gamma_X \vdash \mathrm{abs}(x) : u \Rightarrow t}\ \mathrm{abs}$$

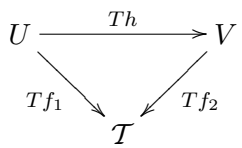
où $\Gamma_{X^*} = \Gamma_X \cup (\infty_X : u)$ et $u \in \mathcal{T}$.

En considérant Γ_X comme le graphe d'une application $X \to \mathcal{T}$, on peut voir le lambda calcul simplement typé comme un foncteur $\mathsf{SST} : (\mathrm{Ens} \downarrow \mathcal{T}) \to (\mathrm{Ens} \downarrow \mathcal{T})$, qui associe à tout ensemble des variables typés $(X \to \mathcal{T})$ l'application $(\mathsf{SLC}\, X \to \mathcal{T})$, les lambda termes typés.

Nous nous trouvons donc dans une nouvelle catégorie de base : $(\mathrm{Ens} \downarrow \mathcal{T})$, au lieu de Ens dans le cas du lambda calcul non–typé. Les objets sont des flèches $X \to \mathcal{T}$ et les morphismes sont les flèches $X \to Y$ telles que :



commute, où $X \to \mathcal{T}$ et $Y \to \mathcal{T}$ désignent deux objets. En particulier, le morphisme identité sur un objet $X \to \mathcal{T}$ coïncide avec Id_X. Un endofoncteur $T : (\mathrm{Ens} \downarrow \mathcal{T}) \to (\mathrm{Ens} \downarrow \mathcal{T})$ transforme un objet $f : X \to \mathcal{T}$ en $Tf : U \to \mathcal{T}$ et un morphisme $h : f_1 \to f_2$ entre deux objets f_1 et f_2, en $Th : U \to V$ tel que :



commute. Sur cette catégorie on définit une monade selon la définition 2.1 avec $\mathcal{C} = (\text{Ens} \downarrow \mathcal{T})$. Dans ce chapitre nous allons considérer des monades et modules sur cette catégorie $(\text{Ens} \downarrow \mathcal{T})$.

La maniement des objets de cette catégorie en Coq est plus convenable d'un point de vue «inverse» : au lieu d'associer à chaque $x \in X$ un type $f(x) \in \mathcal{T}$, où f désigne un objet de $(\text{Ens} \downarrow \mathcal{T})$, on associe à chaque type $t \in \mathcal{T}$ le sous–ensemble de X tel que $f(x) = t$. On manipule en Coq donc une autre forme de $f \in (\text{Ens} \downarrow \mathcal{T}) : \bar{f} : \mathcal{T} \to \mathcal{P}(X), t \mapsto \{x \in X : f(x) = t\}$ où $\mathcal{P}(X)$ désigne l'ensemble des parties de X.

Exemple 3.1 *Le lambda calcul simplement typé* SST *et le lambda calcul simplement typé modulo α–, β– et η–équivalence* ST *sont des monades. Comme dans le cas non–typé, $\eta = \text{var}$, μ est donné par la substitution et la fonctorialité peut être vue comme un renommage des variables. Les preuves de ces assertions réalisées en Coq sont les preuves des lemmes varT_subsT, subst_expl_var, subsT_subsT, tt_var_subst, tt_subst_var et tt_subst_subst. La construction des deux monades correspond aux définitions SLCT et LCT.*

Les définitions d'un morphisme de monades, d'un module sur une monade et d'un morphisme de modules sont analogues aux définitions 2.6, 2.7 et 2.9.

Définition 3.2 (module derivé typé) *Soient R une monade et M un R–module. On construit le* module derivé typé par rapport à $u \in \mathcal{T}$ *de M en posant pour tout $f \in (\text{Ens} \downarrow \mathcal{T})$*

$$\partial_u M f := M f_u$$

où $f : X \to \mathcal{T}$ et $f_u : X^* \to \mathcal{T}$ avec $f_u \mid_X = f$ et $f_u(\infty_X) = u$; et pour tout h morphisme de $(\text{Ens} \downarrow \mathcal{T})$:

$$\partial_u M h := M h^*$$

Remarque 3.3 *Pour tout $u \in \mathcal{T}$ le module derivé typé $\partial_u M$ d'un R–module M est un R–module.*

Preuve. Cette preuve est analogue à la preuve de la remarque 2.5 du cas non–typé. La preuve réalisée en Coq est la preuve du lemme mbind_der_bind. □

La différence avec le cas non–typé est donc l'existence des «derivées partielles» par rapport à chaque type $u \in \mathcal{T}$.

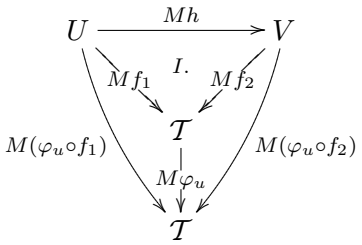
Définition 3.4 (u–module) *Soit R une monade, M un R–module et $u \in \mathcal{T}$. On construit le u–module de M en posant pour tout $f \in (\text{Ens} \downarrow \mathcal{T})$:*

$$M^u f := M(\varphi_u \circ f)$$

où $\varphi_u : \mathcal{T} \to \mathcal{T}, t \mapsto (u \Rightarrow t)$; et pour tout h morphisme de $(\text{Ens} \downarrow \mathcal{T})$:

$$M^u h := M h$$

Soit $h : f_1 \to f_2$ avec $f_1, f_2 \in (\text{Ens} \downarrow \mathcal{T})$, tel que $f_2 \circ h = f_1$, alors on sait que $Mh : Mf_1 \to Mf_2$ est tel que $Mf_2 \circ Mh = Mf_1$. Si on note $Mf_1 : U \to \mathcal{T}$ et $Mf_2 : V \to \mathcal{T}$, alors le triangle I. du diagramme suivant commute et on peut compléter le diagramme par $M\varphi_u$:



Donc Mh fait commuter le triangle extérieur ; ce qui justifie d'avoir posé $M^u h = Mh$.

Remarque 3.5 *Le u–module M^u d'un R–module M est un R–module.*

Preuve. On pose

$$\sigma^u := \sigma$$

Comme $\sigma_f : MRf \to Mf$ est un morphisme de (Ens $\downarrow \mathcal{T}$), et d'après la réflexion précédente, on a bien $\sigma^u_f = \sigma_f : M(\varphi_u \circ Rf) \to M(\varphi_u \circ f)$. Une conséquence de $\sigma^u = \sigma$ est que le carré analogue à (2.8) pour σ^u commute pour tout $f \in$ (Ens $\downarrow \mathcal{T}$) grâce au diagramme commutatif pour σ. Cette preuve correspond à la preuve du lemme mbind_int_bind. □

Le u–module d'un module est donc presque le même module sauf que les types sont «décalés» par u. Cette construction est possible pour tout $u \in \mathcal{T}$.

Notation 3.6 *On note* abs_u *le constructeur* abs *tel que :*

$$\frac{\Gamma_{X^*} \vdash x : t}{\Gamma_X \vdash \mathrm{abs}_u(x) : u \Rightarrow t}$$

On note $\mathrm{app1}_u$ *le constructeur* app1 *tel que :*

$$\frac{\Gamma_X \vdash x : u \Rightarrow t}{\Gamma_{X^*} \vdash \mathrm{app1}_u(x) : t}$$

où $\Gamma_{X^*} = \Gamma_X \cup (\infty_X : u)$ *et* $u \in \mathcal{T}$.

Exemple 3.7 *Le constructeur* abs_u *de* ST *est un morphisme de* ST*–modules* $\partial_u \mathsf{ST} \to \mathsf{ST}^u$ *pour tout* $u \in \mathcal{T}$. *La preuve de cette assertion est essentiellement la preuve du lemme tt_subst_lam réalisée en Coq. La formulation explicite est la définition Lam.*

Exemple 3.8 *Le constructeur* $\mathrm{app1}_u$ *est un morphisme de* ST*–modules* $\mathsf{ST}^u \to \partial_u \mathsf{ST}$ *pour tout* $u \in \mathcal{T}$. *La preuve de cette assertion est essentiellement la preuve du lemme tt_subst_app1 réalisée en Coq. La formulation explicite est la définition App1.*

Définition 3.9 (monade exponentielle typée) *Une* monade exponentielle typée R *est une monade munie pour chaque* $u \in \mathcal{T}$ *d'un isomorphisme de modules* $\alpha_u : M^u \to \partial_u M$ *où* M *désigne le module tautologique de* R.

Une monade exponentielle typée est donc une monade munie d'un isomorphisme de modules pour chaque type, alors qu'une monade exponentielle est munie d'un seul isomorphisme de modules. Il y a aussi une différence entre les modules concernant les isomorphismes : comme dans le cas non–typé il n'existe pas de types, le u–module «devient» le module tautologique.

Exemple 3.10 *La monade* ST *est une monade exponentielle typée avec* $\alpha_u = \mathrm{app1}_u$ *et* $\alpha_u^{-1} = \mathrm{abs}_u$ *pour tout* $u \in \mathcal{T}$*. Les équivalences* η *et* β *garantissent que* $\mathrm{app1}_u$ *et* abs_u *sont inverses. La preuve de cette assertion se retrouve essentiellement dans les preuves des lemmes tt_beta et tt_eta, mais la construction explicite est la définition LCT_Exp.*

Définition 3.11 (morphisme de monades exponentielles typées) *Soient* $(T, \alpha^{(T)})$ *et* $(S, \alpha^{(S)})$ *deux monades exponentielles typées. Un* morphisme de monades exponentielles typées $\nu : T \dot{\to} S$ *est un morphisme de monades qui est compatible pour chaque* $u \in \mathcal{T}$ *avec* $\alpha_u^{(T)}$, $\alpha_u^{(S)}$ *et leurs inverses.* ν *vérifie alors les diagrammes commutatifs suivantes pour tout* $u \in \mathcal{T}$ *et* $f \in (\mathrm{Ens} \downarrow \mathcal{T})$ *:*

$$\begin{array}{ccc} T^u f & \xrightarrow{\nu_{fu}} & S^u \\ {\scriptstyle \alpha_X^{(T)-1}}\uparrow\downarrow{\scriptstyle \alpha_{uf}^{(T)}} & & {\scriptstyle \alpha_{uf}^{(S)-1}}\uparrow\downarrow{\scriptstyle \alpha_{uf}^{(S)}} \\ \partial_u T f & \xrightarrow[\nu_{f_u}]{} & \partial_u S f \end{array}$$

Théorème 3.12 *La monade exponentielle* ST *est l'objet initial dans la catégorie des monades exponentielles typées.*

Preuve. Cette preuve est analogue à la preuve du théorème 2.26. La réalisation de cette preuve est le contenu du fichier Coq tip_thm.v. □

Chapitre 4

Perspective

Dans les deux chapitres précédents, on a caractérisé le lambda calcul pur et le lambda calcul simplement typé comme des objets initiaux dans certaines catégories. Ces catégories et les structures se ressemblent, mais elles ne sont pas aussi reliées que le lambda calcul pur l'est au lambda calcul simplement typé.

Si on munit le lambda calcul pur d'un type, qui est égal pour chaque terme possible, on peut réunir la catégorie de base du lambda calcul pur et du lambda calcul simplement typé dans une nouvelle catégorie $\mathcal{C}$ généralisant (Ens $\downarrow \mathcal{T}$) et Ens. Les objets de $\mathcal{C}$ sont des couples de flèches (f, arr) où $f : X \to T$ pour $X, T \in \mathrm{Ens}$ et $\mathrm{arr} : T \times T \to T$. Les morphismes sont aussi des couples de flèches (ψ, φ) tels que :

$$\begin{array}{ccc} X & \xrightarrow{\psi} & X' \\ {\scriptstyle f_1}\downarrow & & \downarrow{\scriptstyle f_2} \\ T & \xrightarrow[\varphi]{} & T' \end{array}$$

commute et pour tout $u, t \in T$:

$$\varphi(\mathrm{arr}_1(u, t)) = \mathrm{arr}_2(\varphi(u), \varphi(t))$$

pour un morphisme $(f_1, \mathrm{arr}_1) \to (f_2, \mathrm{arr}_2)$. En particulier le morphisme identité sur un objet (f, arr) est donné par $(\mathrm{Id}_X, \mathrm{Id}_T)$. La loi de composition des morphismes est la composition des morphismes d'ensembles dans chaque composant :

$$(\psi_2, \varphi_2) \circ (\psi_1, \varphi_1) := (\psi_2 \circ \psi_1, \varphi_2 \circ \varphi_1)$$

L'associativité et la compatibilité avec l'unité de la composition découlent de la définition.

Une sous–catégorie de $\mathcal{C}$ est (Ens $\downarrow \mathcal{T}$) : l'ensemble $T := \mathcal{T}$ est fixé pour les objets, arr sur $\mathcal{T}$ est le constructeur $\Rightarrow$, les objets de cette sous–catégorie sont donc les couples des flèches $(f, \Rightarrow)$ où $f : X \to \mathcal{T}$ pour un $X \in \mathrm{Ens}$. Les morphismes convenables sont de la forme $(h, \mathrm{Id}_{\mathcal{T}})$. Au chapitre précédent on a vu que ST est l'objet initial de la catégorie des monades exponentielles typées sur cette sous–catégorie.

Une autre sous–catégorie de $\mathcal{C}$ est Ens. Comme $\{*\}$ est l'objet terminal dans Ens, il existe exactement un morphisme $X \to \{*\}$ pour chaque $X \in \mathrm{Ens}$ et on peut identifier chaque flèche de (Ens $\downarrow \{*\}$) avec son domaine. Ens et (Ens $\downarrow \{*\}$) sont donc isomorphes. L'ensemble $T := \{*\}$ est fixé, $\mathrm{arr}_{\{*\}}(*, *) := *$, les objets sont donc de la forme $(f, \mathrm{arr}_{\{*\}})$ où $f : X \to \{*\}$. Mais la partie concernant $\{*\}$ est négligable. Quant aux morphismes, ils sont de la forme $(h, \mathrm{Id}_{\{*\}})$ où h est un morphisme d'ensembles.

En relisant les différences entre les structures typées et non–typées décrites dans les deux chapitres précédents de ce point de vue, on trouve les point suivants :

- Comme $T = \{*\}$ est un singleton dans le cas non–typé, on peut dire, comme dans le cas typé, qu'il existe un module derivé par rapport à chaque type.
- Comme arr dans le cas non–typé est défini comme plus haut, le u–module du module tautologique coïncide clairement avec le module tautologique pour tout $u \in \{*\}$.
- Une monade exponentielle est une monade munie d'un isomorphisme pour chaque type de même qu'une monade exponentielle typée. Mais comme la cardinalité de l'ensemble des types est un, on a un seul isomorphisme.

Il est bien possible de réunir les différences. Logiquement à la suite de ces remarques, il faudrait continuer d'étudier en particulier ces notions et d'essayer de formuler un théorème analogue à (2.26) et (3.12). Ce projet dépasse malheureusement les possibilités de ce mémoire, mais on peut quand même espérer se frayer un chemin pour atteindre ce but.

Bibliographie

[HM05] A. Hirschowitz, M. Maggesi. *Modules over Monads, Monadic Syntax and the Category of untyped Lambda-Calculi*, http ://math.unice.fr/∼ah/rech/marco.html, 2005

[Hir05] A. Hirschowitz. Notes du cours *Lambda Calcul* du master 2, Faculté des Sciences, Université de Nice Sophia–Antipolis, année 2005/06

[Mcl98] S. Mac Lane. *Categories for the Working Mathematician*, Second Edition, Springer Verlag New York, 1998

[Bor94] F. Borceux. *Handbook of categorical algebra 2, Categories and structures*, Cambridge University Press, 1994

[Bar85] H. Barendregt. *The Lambda Calculus – It's Syntax and Semantics*, North–Holland, Amsterdam, New York, Oxford, 1985

[Bar00] H. Barendregt, E. Barendsen. *Introduction to Lambda Calculus*, http ://ling.ucsd.edu/∼barker/Lambda, 1998, 2000

[Har02] T. Hardin–Accart. *Cours de Lambda–calcul* du DEA Sémantique, Preuves et Langages, Laboratoire d'Informatique de Paris 6, Université Pierre et Marie Curie, Paris, http ://www-spi.lip6.fr/∼hardin, année 2002–2003

[Ber02] C. Berline. *Une introduction au λ–calcul*, http ://www.pps.jussieu.fr/∼berline/Cours.html, 1995,2002