Ça dépend, ça dépasse

Une monade dépendante de contrôle délimité

Pierre-Marie Pédrot

PPS/πr^2

14 novembre 2012

Plan

Motivations

- Le manège enchanté de la correspondance de Curry-Howard
  - $\rightsquigarrow$ Prendre le contenu calculatoire des preuves au sérieux
  - $\rightsquigarrow$ Transposer les structures des langages de programmation à la logique

Motivations

- Le manège enchanté de la correspondance de Curry-Howard
  - $\rightsquigarrow$ Prendre le contenu calculatoire des preuves au sérieux
  - $\rightsquigarrow$ Transposer les structures des langages de programmation à la logique

- Des systèmes intuitionnistes puissants
  - $\rightsquigarrow$ Des vrais morceaux de mathématiques !
  - $\rightsquigarrow$ Renouveau des fondations (MLTT, CIC, etc.)

Motivations

- Le manège enchanté de la correspondance de Curry-Howard
  - $\rightsquigarrow$ Prendre le contenu calculatoire des preuves au sérieux
  - $\rightsquigarrow$ Transposer les structures des langages de programmation à la logique
- Des systèmes intuitionnistes puissants
  - $\rightsquigarrow$ Des vrais morceaux de mathématiques !
  - $\rightsquigarrow$ Renouveau des fondations (MLTT, CIC, etc.)
- ... mais les mathématiciens vivent dans un monde classique
  - $\rightsquigarrow$ On y travaille depuis Griffin '90...

Nos outils

- Les sages enseignements de la **logique linéaire**
  - $\rightsquigarrow$ Po–la–ri–sa–tion
  - $\rightsquigarrow$ Analyse de la CPS en style direct
  - $\rightsquigarrow$ ~~Linéarité~~

Nos outils

- Les sages enseignements de la **logique linéaire**
  - $\rightsquigarrow$ Po–la–ri–sa–tion
  - $\rightsquigarrow$ Analyse de la CPS en style direct
  - $\rightsquigarrow$ ~~Linéarité~~
- Des types **riches**
  - $\rightsquigarrow$ Ordre supérieur
  - $\rightsquigarrow$ (Co-)Inductifs
  - $\rightsquigarrow$ **Élimination dépendante**

Slogan

Typi adfingendi sunt

Dans cet exposé

Partir d'un encodage monadique bien connu : la non-non-traduction

↓

Introduire la délimitation

↓

Introduire la dépendance

↓

Retrouver la linéarité ?

Contrôle non-délimité

Au commencement était le contrôle non-délimité (« double négation »).

$$TA := (A \Rightarrow R) \Rightarrow R$$

avec les opérations monadiques habituelles :

$$\begin{aligned} \texttt{return} : &\quad A \Rightarrow TA \\ := &\quad \lambda x.\lambda k.kx \end{aligned}$$

$$\begin{aligned} \texttt{bind} : &\quad TA \Rightarrow (A \Rightarrow TB) \Rightarrow TB \\ := &\quad \lambda m.\lambda f.\lambda k.m(\lambda x.fxk) \end{aligned}$$

Opérateurs non-délimités

Peu d'intérêt, parce qu'un seul type de retour.

$$\begin{aligned}
\texttt{callcc} : &\quad ((A \Rightarrow TB) \Rightarrow TA) \Rightarrow TA \\
:= &\quad \lambda f.\lambda k.f(\lambda x.\lambda - .kx)
\end{aligned}$$

$$\begin{aligned}
\texttt{em} : &\quad T(A \oplus (A \Rightarrow TB)) \\
:= &\quad \lambda k.\texttt{inr}\,(\lambda x.\lambda - .k(\texttt{inl}\ x))
\end{aligned}$$

$$\begin{aligned}
\texttt{absurd} : &\quad ((A \Rightarrow T0) \Rightarrow T0) \Rightarrow TA \\
:= &\quad \lambda f.\lambda k.f\,(\lambda x.\lambda - .kx)\,(\lambda p.\texttt{case}\,p\,\texttt{with}\,[])
\end{aligned}$$

Expressivité du contrôle non-délimité

Peu d'expressivité dans cette monade :

- Dans le cas monomorphe, type de retour essentiel
- Dans le cas polymorphe, aucune garantie sur le contenu
  - Pas de propriété de la disjonction ni de l'existentielle
  - Voire existence de monstres

Contrôle délimité

Le contrôle délimité peut se percevoir de deux manières :

- Du côté dynamique :
  - Les continuations ne rendent pas la main
  - Elles sont réutilisables et composables
  - Présence d'une évaluation locale (`run`)
- Du côté statique :
  - Des types de retour différents
  - Une différence nette entre calculs et valeurs

Contrôle délimité (définitions)

Un nouvel encodage de « monade indicée » :

$$TA :: I \triangleright O := (A \Rightarrow I) \Rightarrow O$$

et des combinateurs plus fins :

$$\begin{aligned} \texttt{return} : &\quad A \Rightarrow TA :: R \triangleright R \\ := &\quad \lambda x.\lambda k.kx \\ \texttt{bind} : &\quad TA :: M \triangleright O \Rightarrow (A \Rightarrow TB :: I \triangleright M) \Rightarrow TB :: I \triangleright O \\ := &\quad \lambda m.\lambda f.\lambda k.m(\lambda x.fxk) \end{aligned}$$

Opérateurs délimités

Quelques opérateurs sympathiques absents du cas non-délimité :

$$\begin{aligned}
\texttt{run} : &\quad (TA :: A \triangleright R) \Rightarrow R \\
:= &\quad \lambda m.m(\lambda x.x) \\
\texttt{catch} : &\quad TA :: I \triangleright (TA :: I \triangleright O) \Rightarrow TA :: I \triangleright O \\
:= &\quad \lambda m.\lambda k.mkk \\
\texttt{abort} : &\quad O \Rightarrow (TA :: I \triangleright O) \\
:= &\quad \lambda v.\lambda - .v
\end{aligned}$$

Et les opérateurs délimités fondateurs :

$$\begin{aligned}
\texttt{reset} : &\quad TI :: I \triangleright M \Rightarrow TM :: O \triangleright O \\
:= &\quad \lambda m.\texttt{return}\,(\texttt{run}\ m) \\
\texttt{shift} : &\quad \ldots \\
:= &\quad \ldots
\end{aligned}$$

Opérateurs délimités

Les grands classiques :

$$
\begin{array}{rl}
\texttt{callcc} : & ((A \Rightarrow TB :: I \triangleright M) \Rightarrow TA :: M \triangleright O) \Rightarrow TA :: M \triangleright O \\
:= & \ldots \\
\texttt{em} : & T(A \oplus (A \Rightarrow TB :: I \triangleright O)) :: O \triangleright O \\
:= & \ldots \\
\texttt{absurd} : & ((A \Rightarrow T0 :: M \triangleright I) \Rightarrow T0 :: R \triangleright O) \Rightarrow TA :: I \triangleright O \\
:= & \ldots
\end{array}
$$

Expressivité du contrôle délimité

Nettement meilleure expressivité que le contrôle non-délimité :

- Capacité de sortir de la monade ;
- Informations plus riches sur ce que renvoie le calcul
  - ... mais ça n'est pas aussi puissant que les types linéaires
- Grâce à la paramétricité, résultats métas
  - ... mais ça n'est pas interne

Prendre la polarisation au sérieux I

En logique linéaire polarisée, dualité fondamentale :

- Positifs : constructeurs $\rightsquigarrow$ valeurs

$$P ::= 1 \mid P \otimes Q \mid 0 \mid P \oplus Q \mid !N$$

- Négatifs : destructeurs $\rightsquigarrow$ calculs

$$N ::= \bot \mid M \parr N \mid \top \mid M \,\&\, N \mid ?P$$

Cependant, l'orthogonalité est asymétrique.

- $P^\bot := P \multimap \bot$ est bien polarisé
- $N^\bot := N \multimap \bot$ ne l'est pas

Prendre la polarisation au sérieux II

- Dans CIC, l'élimination dépendante provient des inductifs (i.e. positifs)
- Le type associé à un connecteur négatif mime son terme de preuve

Terme	Type
$\lambda x : A.M$	$\forall x : A.B$
$\texttt{case}\,(p : A \otimes B)\,\texttt{with}\,(x, y) \Rightarrow M$	$\texttt{case}\,(p : A \otimes B)\,\texttt{with}\,(x, y) \Rightarrow C$
$\texttt{case}\,(s : A \oplus B)\,\texttt{with}$ $\mid \texttt{inl}\,x \Rightarrow M_1$ $\mid \texttt{inr}\,y \Rightarrow M_2$	$\texttt{case}\,(s : A \oplus B)\,\texttt{with}$ $\mid \texttt{inl}\,x \Rightarrow C_1$ $\mid \texttt{inr}\,y \Rightarrow C_2$

Prendre la polarisation au sérieux III

À cause de la nature de l'élimination dépendante, on restreint la dépendance au **positifs** ($\cong$ *value restriction*).

La monade de continuation délimitée devient :

$$\frac{I : A \Rightarrow \mathtt{Type} \qquad O : \mathtt{Type} \qquad A \text{ positif}}{T\,A :: I \triangleright O := (\forall x : A.\, I\,x) \Rightarrow O}$$

Et le `return` passe naturellement :

$$\frac{R : A \Rightarrow \mathtt{Type} \qquad A \text{ positif}}{\mathtt{return}\,(v : A) : (\forall x : A.\, R\,x) \Rightarrow R\,v}$$

Le problème du `bind`

Comment raffiner dans un monde dépendant :

$$\texttt{bind} : TA \Rightarrow (A \Rightarrow TB) \Rightarrow TB$$

En effet, on voudrait :

$$\frac{A : \texttt{Type} \qquad B : A \Rightarrow \texttt{Type} \qquad A \text{ positif}}{\texttt{bind}\,(m : TA) : (\forall x : A.\, T\,(B\,x)) \Rightarrow T\,(B\,\star)}$$

Mais m est une monade, et on ne peut pas en sortir aussi facilement !
Que met-on dans $\star$?

Une monade généralisée

La solution consiste à généraliser le type de la monade délimitée en y incluant un **télescope** entier.

$$\mathcal{T} ::= \emptyset : \mathcal{T} \mid (::) : \forall A : \mathtt{Type}.\, (A \Rightarrow \mathcal{T}) \Rightarrow \mathcal{T}$$

On définit inductivement l'habitant d'un télescope :

$$\frac{I : \mathtt{Type}}{I \in \emptyset} \qquad \frac{A : \mathtt{Type} \qquad \mathfrak{t} : A \Rightarrow \mathcal{T} \qquad \forall x : A.\, (Ix \in \mathfrak{t}\, x)}{\forall x : A.\, Ix \in A :: \mathfrak{t}}$$

En pratique, l'habitant d'un télescope est un type de la forme :

$$\forall x_1 : A_1.\, \ldots\, x_n : A_n.\, I x_1 \ldots x_n$$

C'est un peu plus compliqué...

Enfin, la monade généralisée

La forme complète de la monade dépendante délimitée devient :

$$\frac{\mathtt{t} : \mathcal{T} \qquad I \in \mathtt{t} \qquad O : \mathtt{Type}}{I \triangleright O := I \Rightarrow O}$$

et les combinateurs de base deviennent :

$$\frac{R \in A :: \emptyset}{\mathtt{return}\,(v : A) : R \triangleright Rv}$$

$$\frac{I \in A :: \mathtt{t} \qquad M \in A :: \emptyset \qquad O : \mathtt{Type}}{\mathtt{bind}\,(m : M \triangleright O)(f : \forall x : A.\,(I \triangleright Mx)) : I \triangleright O}$$

if not understood then raise WTF

Une version lisible par les humains[1] de la slide précédente :

$$\frac{R : A \Rightarrow \texttt{Type}}{\texttt{return}\,(v : A) : (\forall x : A. R\,x) \Rightarrow R\,v}$$

$$\frac{\begin{array}{c} O : \texttt{Type} \qquad M : A \Rightarrow \texttt{Type} \\ I : \forall x : A. \forall \vec{y} : \vec{B}\,x. \texttt{Type} \\ m : (\forall x : A.\, M\,x) \Rightarrow O \qquad f : \forall x : A.\, [(\forall \vec{y} : \vec{B}.\, I\,x\,\vec{y}) \Rightarrow M\,x] \end{array}}{\texttt{bind}\,m\,f : (\forall x : A. \forall \vec{y} : \vec{B}\,x.\, I\,x\,\vec{y}) \Rightarrow O}$$

1. Photo non contractuelle.

D'autres combinateurs

Le fait d'utiliser des monades variadiques élimine certains artefacts :

$$\begin{array}{lll} \texttt{callcc} : & ((\forall x : A.\, (\forall y : B.\, M\, x\, y) \Rightarrow I\, x) & \Rightarrow (\forall x : A.\, I\, x) \Rightarrow O) \Rightarrow (\forall x : A.\, I\, x) \Rightarrow O \\ \texttt{callcc}' : & ((\forall x : A.\, I\, x) & \Rightarrow (\forall x : A.\, I\, x) \Rightarrow O) \Rightarrow (\forall x : A.\, I\, x) \Rightarrow O \end{array}$$

$$\begin{array}{lll} \texttt{run} : & ((A \Rightarrow A) & \Rightarrow O) \Rightarrow O \\ \texttt{run}' : & (\forall p : 1.\, \texttt{case}\ p\ \texttt{with}\ () & \Rightarrow O) \Rightarrow O \end{array}$$

Quelques remarques

- Cette définition fait un usage implicite de règles de conversions
- On peut tout à fait l'internaliser en CIC
- Sur le fragment $\forall$, les termes de preuve sont **inchangés**
  - En particulier `return` et `bind`
  - Mais aussi `run`, `abort` et `callcc`
- Le cas du type de m dans `bind` peut être aisément généralisé
- Le type de `bind` ressemble furieusement au *double negation shift*

$$\frac{O : \mathtt{Type} \qquad M : A \Rightarrow \mathtt{Type}}{I : \forall x : A.\, \forall \vec{y} : \vec{B}.\, \mathtt{Type}}$$

$$\frac{m : (\forall x : A.\, M\,x) \Rightarrow O \qquad f : \forall x : A.\, [(\forall \vec{y} : \vec{B}\,x.\, I\,x\,\vec{y}) \Rightarrow M\,x]}{\mathtt{bind}\,m\,f : (\forall x : A.\, \forall \vec{y} : \vec{B}\,x.\, I\,x\,\vec{y}) \Rightarrow O}$$

Deux combinateurs

On peut déjà construire deux combinateurs sympathiques sur les habitants de télescopes.

$$\frac{}{\top : \forall s : 0.\, \texttt{case}\ s\ \texttt{with}\ [] \in 0 :: \emptyset}$$

$$\frac{M := \forall p : P.\, \hat{M} p \qquad N := \forall q : Q.\, \hat{N} q}{M \,\&\, N := \forall s : P \oplus Q.\, \texttt{case}\ s\ \texttt{with}\ [\texttt{inl}\, x \Rightarrow \hat{M}\, x \mid \texttt{inl}\, y \Rightarrow \hat{N}\, y]}$$

Les principes classiques

Voici enfin deux principes classiques auquels on est habitués :

$$\begin{aligned}
\texttt{em} : &\quad ((\forall x : A.\, I\, x)) \,\&\, ((\forall x : A.\, I\, x) \Rightarrow O) \Rightarrow O \\
\texttt{em} : &\quad (\forall (s : A \oplus (\forall x : A.\, I\, x)).\, \texttt{case}\ s\ \texttt{with} \\
&\qquad [\texttt{inl}\, x \Rightarrow I\, x \mid \texttt{inr}\, - \Rightarrow O]) \Rightarrow O
\end{aligned}$$

$$\texttt{absurd}' : \quad ((\forall x : A.\, I\, x) \Rightarrow \top \Rightarrow O) \Rightarrow (\forall x : A.\, I\, x) \Rightarrow O$$

Termes de preuve laissés au lecteur.

Inversibilité du &

Les règles sur les positifs en logique linéaire sont problématiques d'un point de vue de la focalisation.

$$\frac{\vdash \Gamma, A}{\vdash \Gamma, A \oplus B}$$

Dans cette règle, rien ne garantit en effet que les règles suivantes porteront uniquement sur A (i.e. la preuve de A n'est pas nécessairement une valeur). Au niveau de la CPS, cela correspond à une commutation :

$$TA \oplus TB \Rightarrow T(A \oplus B)$$

Inversibilité du &

En calcul délimité non-dépendant, cette commutation existe, mais seulement si le type d'entrée coincide. De plus, elle perd de l'information sur le type de sortie : la fonction n'est pas **réversible**.

$$(T\,A_1 :: I \triangleright O_1) \oplus (T\,A_2 :: I \triangleright O_2) \Rightarrow T(A_1 \oplus A_2) :: I \triangleright (O_1 \oplus O_2)$$

$$\lambda s.\lambda k.\texttt{case}\ s\ \texttt{with}\ [\texttt{inl}\,x \Rightarrow \texttt{inl}(x\,k) \mid \texttt{inr}\,y \Rightarrow \texttt{inr}(y\,k)]$$

Problème similaire avec $\top$: il faut choisir un type de retour.

$$\top_O := 0 \Rightarrow O$$

Inversibilité du &

Ici, grâce aux types dépendants :

$$M_i := (\forall x_i : A_i.\, I_i\, x_i) \Rightarrow O_i$$
$$I := \lambda p : (A_1 \oplus A_2).\, \texttt{case}\ p\ \texttt{with}\ [\texttt{inl}\, x \Rightarrow I_1\, x \mid \texttt{inr}\, y \Rightarrow I_2\, y])$$
$$O := \lambda r : (M_1 \oplus M_2).\, \texttt{case}\ r\ \texttt{with}\ [\texttt{inl}\, - \Rightarrow O_1 \mid \texttt{inr}\, - \Rightarrow O_2])$$

$$\forall r : (M_1 \oplus M_2).\, (\forall p : (A_1 \oplus A_2).\, I\, p) \Rightarrow (O\, r)$$

On récupère une commutation réversible.

Quid du $\mathcal{Y}$?

On voudrait faire la même chose avec la règle :

$$\frac{\vdash \Gamma, A \qquad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}$$

i.e. exhiber une commutation $TA \otimes TB \Rightarrow T(A \otimes B)$ réversible.

Quid du $\mathcal{Y}$?

On voudrait faire la même chose avec la règle :

$$\frac{\vdash \Gamma, A \qquad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}$$

i.e. exhiber une commutation $TA \otimes TB \Rightarrow T(A \otimes B)$ réversible.

Échec

Mais les solutions habituelles ne marchent pas ! Il faut **orienter** le tenseur.

Selon l'orientation, la dépendance forcée fournit alors :

- $\exists x : A.\, B\, x$ (gauche–droite)
- $\exists y : B.\, A\, y$ (droite–gauche)

Dans les deux cas, c'est une forme dégénérée du `bind`.

Que faire ?

Cette solution n'est pas satisfaisante. Un petit détour par la logique linéaire nous permet d'y voir plus clair.

Proposition

$$(!A \multimap I) \multimap O \cong O^{\perp} \multimap !A \otimes I^{\perp}$$

La monade de continuation linéaire et d'état sont isomorphes...

Et ça, on sait faire :

$$\frac{O_i{}^{\perp} \multimap\, !A_i \otimes I_i{}^{\perp}}{O_1{}^{\perp} \otimes O_2{}^{\perp} \multimap\, !(A_1 \otimes A_2) \otimes (I_1{}^{\perp} \otimes I_2{}^{\perp})}$$

C'est un calcul en parallèle des effets.

Problème, en redualisant, on trouve :

$$(!(A_1 \otimes A_2) \multimap (I_1 \parr I_2)) \multimap O_1 \parr O_2$$

Comment encoder $\parr$?

« S'il n'y a pas de solution, il n'y a pas de problème »

Proverbe Shadok

Regardons le comportement du $\parr$ sur deux télescopes linéaires.

$$(P \multimap M) \parr (Q \multimap N) \cong P \otimes Q \multimap M \parr N$$

La définition du $\parr$ dépend d'elle-même. Comment s'en sortir ?

« S'il n'y a pas de solution, il n'y a pas de problème »

Proverbe Shadok

Regardons le comportement du $\parr$ sur deux télescopes linéaires.

$$(P \multimap M) \parr (Q \multimap N) \cong P \otimes Q \multimap M \parr N$$

La définition du $\parr$ dépend d'elle-même. Comment s'en sortir ?

Solution

L'équation ci-dessus est la définition du $\parr$, dont c'est le **plus grand** point fixe.

Généralisation de la monade délimitée dépendante

Pour pouvoir définir $M \parr N$, il faut alors généraliser la notion de télescope en les prenant comme **co-inductifs** (i.e. des listes dépendantes à l'infini).

Les habitants d'un télescope sont adaptés de même, et sont de la forme :

$$\forall x_1 : P_1. \ldots \forall x_n : P_n. \ldots$$

Les habitants d'un télescope représentent les **négatifs** de la logique linéaire.

Si I et O sont deux négatifs, la monade devient :

$$I \triangleright O := I \Rightarrow O$$

Connecteurs infinitaires

On peut définir des connecteurs sur nos télescopes infinis :

- Les connecteurs $\&$ et $\top$ sont inchangés :

$$(\forall x : A.\, M\, x) \,\&\, (\forall y : B.\, N\, y) :=$$
$$\forall s : (A \oplus B).\, \mathtt{case}\ s\ \mathtt{with}\ [\mathtt{inl}\, x \Rightarrow M\, x \mid \mathtt{inr}\, y \Rightarrow N\, y]$$

$$\top := \forall p : 0.\, \mathtt{case}\ s\ \mathtt{with}\ []$$

- Comme défini précédemment :

$$(\forall x : A.\, M\, x) \parr (\forall y : B.\, N\, y) :=$$
$$\forall p : (A \otimes B).\, \mathtt{case}\ p\ \mathtt{with}\ (x, y) \Rightarrow (M\, x) \parr (N\, y)$$

- Enfin, pour $\bot$:

$$\bot := \forall p : 1.\, \mathtt{case}\ p\ \mathtt{with}\ () \Rightarrow \bot$$

Écrire des $\parr$ en thèse

On peut, dans ce formalisme, écrire la commutation réversible :

$$TA \otimes TB \Rightarrow T(A \otimes B)$$

Elle devient :

$$(I_1 \triangleright O_1) \otimes (I_2 \triangleright O_2) \Rightarrow (I_1 \parr I_2) \triangleright (O_1 \parr O_2)$$

Cela se prouve par coinduction.

Remarques

- La différence essentielle entre $\bot$ et 0 est éclairante :
  - $\rightsquigarrow$ une preuve de 0 est une contradiction ;
  - $\rightsquigarrow$ une preuve de $\bot$ n'existe pas, car $\bot$ est un type infini alors qu'une preuve est un objet fini (parapreuves ?)
- Les négatifs n'existent pas *a priori* : les preuves se contentent de les prendre en argument et de les chaîner.
- L'égalité n'a pas le même statut :
  - $\rightsquigarrow$ Sur les positifs, c'est une élimination par cas ;
  - $\rightsquigarrow$ Sur les négatifs, elle se définit par coinduction.

Retrouver la logique linéaire

- Il nous manque les exponentielles (et les shifts)
- Pour l'instant, ça n'est pas encore très clair sur leur nécessité
- Ce qui a l'air de marchotter :

$$\uparrow(v : A) := R \triangleright Rv$$

$$?A := (\forall x : A.\, Ix) \triangleright O$$

$$!A = \downarrow A = A \quad (???)$$

- Le bang vu comme délimiteur (conversion implicite) ?
- Mettre des quantifications au bon endroit ?

La logique... linéaire ?

- La logique linéaire est une CPS en style direct
  - $\rightsquigarrow$ née des espaces de cohérence
- La linéarité est une contrainte implicite sur le type de retour
  - $\rightsquigarrow$ ... en particulier, rien à voir avec la duplication dans l'absolu
  - $\rightsquigarrow$ ... et la logique linéaire n'internalise pas : $\vdash ?P$ implique $\vdash P$
- Les exponentielles représentent une contrainte sur le contexte
  - $\rightsquigarrow$ Elles n'existent pas hors d'un séquent
  - $\rightsquigarrow$ Elles séquentialisent les dépendances
- Un mismatch fondamental entre $\vdash$ et $\multimap$
  - $\rightsquigarrow$ mal refleté par les modèles catégoriques

La logique ~~linéaire~~

Affirmation gratuite

La logique linéaire est le fragment propositionnel d'une logique « contextuelle » munie :

- de l'élimination dépendante
- d'objets co-inductifs

où la linéarité apparaît comme un artefact des dépendances.

Merci de votre attention