Dialectica peut-elle casser des briques ?

Pierre-Marie Pédrot

PPS/πr^2

22 octobre 2013

Il était une fois...

- Cataclysme du théorème d'incomplétude de Gödel (1931)

Il était une fois...

- Cataclysme du théorème d'incomplétude de Gödel (1931)

> On ne combat pas l'aliénation par une logique aliénée.

Il était une fois...

- Cataclysme du théorème d'incomplétude de Gödel (1931)

> On ne combat pas l'aliénation par une logique aliénée.

- Justifier la cohérence de l'arithmétique classique par d'autres moyens
- ... en se ramenant à une logique constructive
  - Traduction par double-négation (1933)
  - **Dialectica** (années 30, publié en 1958)

Plan

Vue d'ensemble

Dialectica, c'est quoi ?

Vue d'ensemble

Dialectica, c'est quoi ?

- Une traduction de HA dans HA^ω
- Préserve le contenu intuitionniste
- Exactement deux axiomes classiques en plus :

$$\mathsf{MP}\ \frac{\neg(\forall n \in \mathbb{N}.\, \neg P\, n)}{\exists n \in \mathbb{N}.\, P\, n} \qquad \frac{(\forall n \in \mathbb{N}.\, P\, n) \to \exists m \in \mathbb{N}.\, Q\, m}{\exists m \in \mathbb{N}.\, (\forall n \in \mathbb{N}.\, P\, n) \to Q\, m}\ \mathsf{IP}$$

À propos de MP et IP

- Notoirement connus pour être implémentables avec des exceptions
- Sans casser les bonnes propriétés de LJ
  - MP : « si je ne boucle pas, alors je termine »
  - IP : « je n'ai pas besoin d'un argument sans contenu calculatoire pour répondre »

Avertissement

Pour l'exhaustivité, on va présenter Dialectica sous sa forme originelle.

Avertissement

Pour l'exhaustivité, on va présenter Dialectica sous sa forme originelle.

Attention ! Dusty logic inside

- Transformation de formule à formule
- Logique du premier ordre
- Prévalence des connecteurs négatifs

Dialectica, version protohistorique :

$$\vdash A \qquad \mapsto \qquad \vdash A^D \equiv \exists \vec{u}.\, \forall \vec{x}.\, A_D[\vec{u}, \vec{x}]$$

$A \wedge B$	$\exists \vec{u}\, \vec{v}.$	$\forall \vec{x}\, \vec{y}.$	$A_D[\vec{u}, \vec{x}] \wedge B_D[\vec{v}, \vec{y}]$
$A \vee B$	$\exists \vec{u}\, \vec{v}\, b.$	$\forall \vec{x}\, \vec{y}.$	$(b = 0 \wedge A_D[\vec{u}, \vec{x}]) \vee (b = 1 \wedge B_D[\vec{v}, \vec{y}])$
$A \to B$	$\exists \vec{\varphi}\, \vec{\psi}.$	$\forall \vec{u}\, \vec{y}.$	$A_D[\vec{u}, \vec{\psi}(\vec{u}, \vec{y})] \to B_D[\vec{\varphi}(\vec{u}), \vec{y}]$
$\forall n.\, A[n]$	$\exists \vec{\varphi}.$	$\forall \vec{x}\, n.$	$A_D[\vec{\varphi}(n), \vec{x}, n]$
$\exists n.\, A[n]$	$\exists \vec{u}\, n.$	$\forall \vec{x}.$	$A_D[\vec{u}, n, \vec{x}]$

Dialectica, version protohistorique :

$$\vdash A \qquad \mapsto \qquad \vdash A^D \equiv \exists \vec{u}.\, \forall \vec{x}.\, A_D[\vec{u}, \vec{x}]$$

$A \wedge B$	$\exists \vec{u}\, \vec{v}.$	$\forall \vec{x}\, \vec{y}.$	$A_D[\vec{u}, \vec{x}] \wedge B_D[\vec{v}, \vec{y}]$
$A \vee B$	$\exists \vec{u}\, \vec{v}\, b.$	$\forall \vec{x}\, \vec{y}.$	$(b = 0 \wedge A_D[\vec{u}, \vec{x}]) \vee (b = 1 \wedge B_D[\vec{v}, \vec{y}])$
$A \to B$	$\exists \vec{\varphi}\, \vec{\psi}.$	$\forall \vec{u}\, \vec{y}.$	$A_D[\vec{u}, \vec{\psi}(\vec{u}, \vec{y})] \to B_D[\vec{\varphi}(\vec{u}), \vec{y}]$
$\forall n.\, A[n]$	$\exists \vec{\varphi}.$	$\forall \vec{x}\, n.$	$A_D[\vec{\varphi}(n), \vec{x}, n]$
$\exists n.\, A[n]$	$\exists \vec{u}\, n.$	$\forall \vec{x}.$	$A_D[\vec{u}, n, \vec{x}]$

On a bien la correction (et le reste).

Dusty logics II

Quelques remarques à la volée :

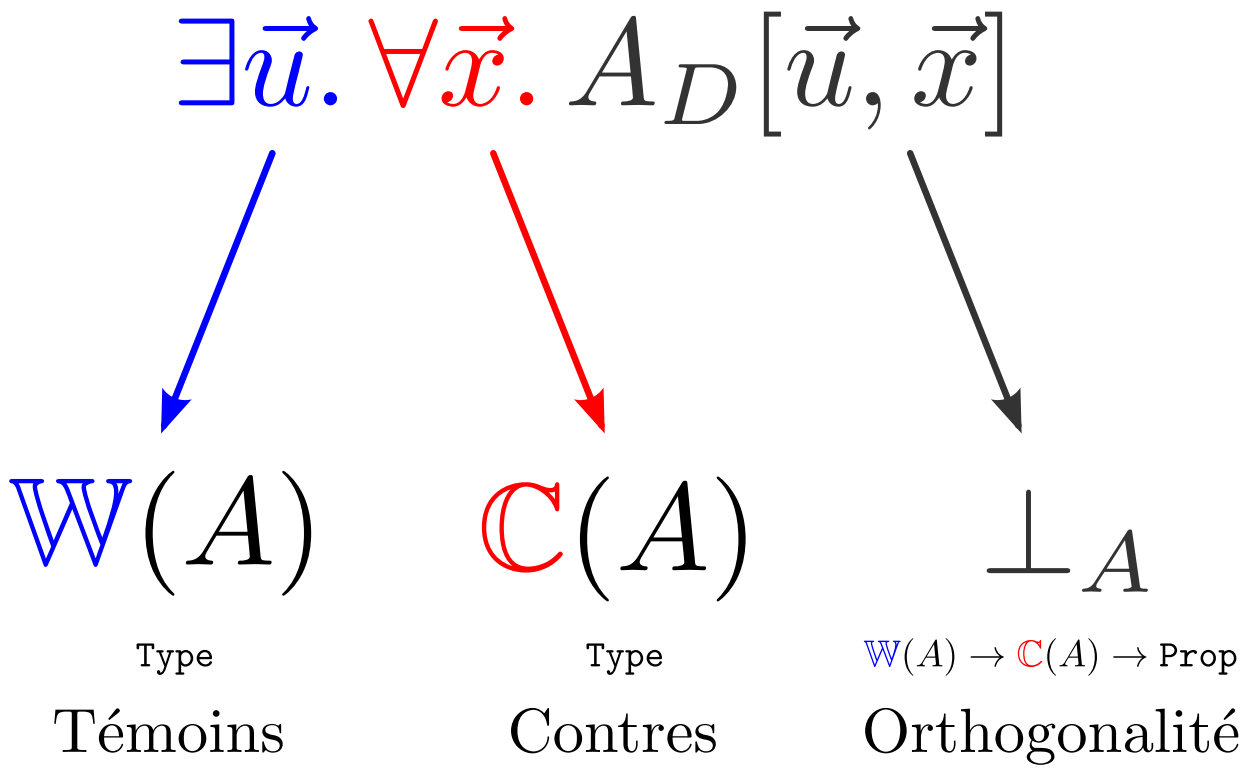
- La traduction enrichit les connecteurs ;
- Le cas du $\vee$ ressemble à s'y méprendre à un encodage sordide d'un type somme ;
- Les traductions de $\to$ et de $\forall$ introduisent le besoin de HA^ω ;
  - $\rightsquigarrow$ En particulier, via Curry-Howard, les fonctions de $A \to B$ sont bien traduites par des fonctions !

Et le contenu calculatoire bordel ?

Oublions les années 50 pour de bon, et sautons au années 90.

- Prendre le contenu **calculatoire** au sérieux
- Dialectica comme un objet **typé**
- Travaux de De Paiva, Hyland, etc.

Gödel's anatomy



Un bref aperçu

Une preuve $\vdash u : A$ est un terme $\vdash u : \mathbb{W}(A)$ tel que :

$$\forall x : \mathbb{C}(A).\, u \perp_A x$$

Un bref aperçu

Une preuve $\vdash u : A$ est un terme $\vdash u : \mathbb{W}(A)$ tel que :

$$\forall x : \mathbb{C}(A).\, u \perp_A x$$

Pour fixer les idées, si on s'en tient aux types :

	$\mathbb{W}$	$\mathbb{C}$
$A \wedge B$	$\exists \vec{u}\, \vec{v}.$	$\forall \vec{x}\, \vec{y}.$
$A \times B$	$\mathbb{W}(A) \times \mathbb{W}(B)$	$\mathbb{C}(A) \times \mathbb{C}(B)$
$A \vee B$	$\exists b\, \vec{u}\, \vec{v}.$	$\forall \vec{x}\, \vec{y}.$
$A + B$	$\mathtt{bool} \times \mathbb{W}(A) \times \mathbb{W}(B)$	$\mathbb{C}(A) \times \mathbb{C}(B)$
$A \to B$	$\exists \vec{\varphi}\, \vec{\psi}.$	$\forall \vec{u}\, \vec{y}.$
$A \to B$	$\begin{cases} \mathbb{W}(A) \to \mathbb{W}(B) \\ \mathbb{C}(B) \to \mathbb{W}(A) \to \mathbb{C}(A) \end{cases}$	$\mathbb{W}(A) \times \mathbb{C}(B)$

Un bref aperçu

Une preuve $\vdash u : A$ est un terme $\vdash u : \mathbb{W}(A)$ tel que :

$$\forall x : \mathbb{C}(A).\, u \perp_A x$$

Pour fixer les idées, si on s'en tient aux types :

	$\mathbb{W}$	$\mathbb{C}$
$A \wedge B$	$\exists \vec{u}\, \vec{v}.$	$\forall \vec{x}\, \vec{y}.$
$A \times B$	$\mathbb{W}(A) \times \mathbb{W}(B)$	$\mathbb{C}(A) \times \mathbb{C}(B)$
$A \vee B$	$\exists b\, \vec{u}\, \vec{v}.$	$\forall \vec{x}\, \vec{y}.$
$A + B$	$\mathtt{bool} \times \mathbb{W}(A) \times \mathbb{W}(B)$	$\mathbb{C}(A) \times \mathbb{C}(B)$
$A \to B$	$\exists \vec{\varphi}\, \vec{\psi}.$	$\forall \vec{u}\, \vec{y}.$
$A \to B$	$\begin{cases} \mathbb{W}(A) \to \mathbb{W}(B) \\ \mathbb{C}(B) \to \mathbb{W}(A) \to \mathbb{C}(A) \end{cases}$	$\mathbb{W}(A) \times \mathbb{C}(B)$

Remarquons encore une fois l'encodage d'un type somme...

Déjà vu

Oh ! Grand-mère ! Comme vous avez un air de famille !

Déjà vu

Oh ! Grand-mère ! Comme vous avez un air de famille !

- Réalisabilité classique : $\mathbb{W}(A)$ preuves $|A|$, $\mathbb{C}(A)$ piles $||A||$
- Modèles par double-orthogonal
- *Double-glueing*
- Candidats de réducibilité
- ...

Pas de précipitation

- On pourrait donner un contenu calculatoire tout de suite

Pas de précipitation

- On pourrait donner un contenu calculatoire tout de suite
- Mais il serait *ad-hoc*, héritant des solécismes de Dialectica
- On va plutôt passer par une version **linéaire** d'abord !
  - Un vrai statut pour l'exponentielle ;
  - Avec des vrais morceaux de type somme dedans...

Dialectica linéaire

Comme on l'a annoncé à la slide précédente, on procède à essentiellement deux transformations :

- L'introduction de la dualité via des types somme ;
- La décomposition de la flèche en appel par nom :

$$A \to B \quad \equiv \quad !A \multimap B$$

Dialectica linéaire

Comme on l'a annoncé à la slide précédente, on procède à essentiellement deux transformations :

- L'introduction de la dualité via des types somme ;
- La décomposition de la flèche en appel par nom :

$$A \to B \quad \equiv \quad !A \multimap B$$

Maintenant, on traduira une formule de LL en une formule de LJ.

Cahier des charges

On veut interpréter les formules définies par :

$$A, B ::= A \otimes B \mid A \parr B \mid A \oplus B \mid A \,\&\, B \mid !A \mid ?A$$

On a juste à définir $\mathbb{W}(A)$, $\mathbb{C}(A)$ et $\perp_A$.

Cahier des charges

On veut interpréter les formules définies par :

$$A, B ::= A \otimes B \mid A \parr B \mid A \oplus B \mid A \,\&\, B \mid !A \mid ?A$$

On a juste à définir $\mathbb{W}(A)$, $\mathbb{C}(A)$ et $\perp_A$.

En s'inspirant des modèles à double orthogonalité linéaire, on veut :

- $\mathbb{W}(A^\perp) \equiv \mathbb{C}(A)$ et inversement ;
- d'où $\perp_A \subseteq \mathbb{W}(A) \times \mathbb{C}(A) \equiv \mathbb{W}(A) \times \mathbb{W}(A^\perp)$

Réglons son cas à la dualité

- $\rightsquigarrow$ Il suffit de définir la traduction sur les positifs
- $\rightsquigarrow$ Le connecteur dual suivra... par dualité

On décrète alors :

$$\frac{u \not\perp_A x}{x \perp_{A^\perp} u}$$

Types somme

	$\mathbb{W}$	$\mathbb{C}$
$A \times B$	$\mathbb{W}(A) \times \mathbb{W}(B)$	$\mathbb{C}(A) \times \mathbb{C}(B)$
$A \,\&\, B$	$\mathbb{W}(A) \times \mathbb{W}(B)$	$\mathbb{C}(A) + \mathbb{C}(B)$
$A + B$	$\texttt{bool} \times \mathbb{W}(A) \times \mathbb{W}(B)$	$\mathbb{C}(A) \times \mathbb{C}(B)$
$A \oplus B$	$\mathbb{W}(A) + \mathbb{W}(B)$	$\mathbb{C}(A) \times \mathbb{C}(B)$

Types somme

	$\mathbb{W}$	$\mathbb{C}$
$A \times B$	$\mathbb{W}(A) \times \mathbb{W}(B)$	$\mathbb{C}(A) \times \mathbb{C}(B)$
$A \,\&\, B$	$\mathbb{W}(A) \times \mathbb{W}(B)$	$\mathbb{C}(A) + \mathbb{C}(B)$
$A + B$	$\mathtt{bool} \times \mathbb{W}(A) \times \mathbb{W}(B)$	$\mathbb{C}(A) \times \mathbb{C}(B)$
$A \oplus B$	$\mathbb{W}(A) + \mathbb{W}(B)$	$\mathbb{C}(A) \times \mathbb{C}(B)$

$$\frac{v \perp_A z_2}{\mathtt{inr}\, v \perp_{A \oplus B} (z_1, z_2)} \qquad \frac{u \perp_A z_1}{\mathtt{inl}\, u \perp_{A \oplus B} (z_1, z_2)}$$

Décomposition linéaire

	$\mathbb{W}$	$\mathbb{C}$
$A \to B$	$\begin{cases} \mathbb{W}(A) \to \mathbb{W}(B) \\ \mathbb{C}(B) \to \mathbb{W}(A) \to \mathbb{C}(A) \end{cases}$	$\mathbb{C}(A) \times \mathbb{C}(B)$
$A \multimap B$	$\begin{cases} \mathbb{W}(A) \to \mathbb{W}(B) \\ \mathbb{C}(B) \to \mathbb{C}(A) \end{cases}$	$\mathbb{W}(A) \times \mathbb{C}(B)$
$!A$	$\mathbb{W}(A)$	$\mathbb{W}(A) \to \mathbb{C}(A)$

Décomposition linéaire

	$\mathbb{W}$	$\mathbb{C}$
$A \to B$	$\begin{cases} \mathbb{W}(A) \to \mathbb{W}(B) \\ \mathbb{C}(B) \to \mathbb{W}(A) \to \mathbb{C}(A) \end{cases}$	$\mathbb{C}(A) \times \mathbb{C}(B)$
$A \multimap B$	$\begin{cases} \mathbb{W}(A) \to \mathbb{W}(B) \\ \mathbb{C}(B) \to \mathbb{C}(A) \end{cases}$	$\mathbb{W}(A) \times \mathbb{C}(B)$
$!A$	$\mathbb{W}(A)$	$\mathbb{W}(A) \to \mathbb{C}(A)$

$$\frac{u \perp_A \psi\, y \quad \to \quad \varphi\, u \perp_B y}{(\varphi, \psi) \perp_{A \multimap B} (u, y)} \qquad \frac{u \perp_A z\, u}{u \perp_{!A} z}$$

Justification avec les mains

- L'interprétation de la flèche force la réversibilité :

$$A \multimap B \cong B^{\perp} \multimap A^{\perp}$$

  $\rightsquigarrow$ Comme les fils des réseaux : parcourus dans les deux sens

Justification avec les mains

- L'interprétation de la flèche force la réversibilité :

$$A \multimap B \cong B^{\perp} \multimap A^{\perp}$$

  - $\rightsquigarrow$ Comme les fils des réseaux : parcourus dans les deux sens
- L'interprétation du bang est un *shift* :
  - $\rightsquigarrow$ L'adversaire est autorisé à attendre que le joueur ait répondu avant de jouer
- La dualité est le changement de joueur

Deux mots sur la linéarité

La notion de **linéarité** n'est pas ici pour faire de la figuration.

1. Sous réserve de définir 1.
2. Photo non contractuelle.

Deux mots sur la linéarité

La notion de **linéarité** n'est pas ici pour faire de la figuration.

En effet, on n'a pas dans Dialectica, en général, de morphismes :

$$\vdash A \multimap 1 \,^{1}$$

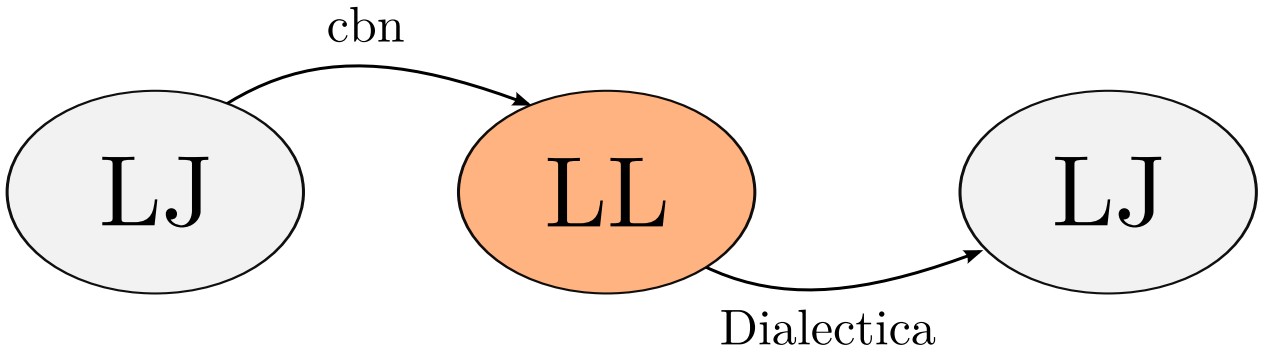
$$\vdash A \multimap A \otimes A$$

On a donc vraiment des contraintes linéaires dans la traduction ![2]

1. Sous réserve de définir 1.
2. Photo non contractuelle.

Interprétation du λ-calcul *call-by-name*

Maintenant, on s'attache à traduire le λ-calcul.



- Via la traduction *call-by-name* dans LL ;
- Puis de LL dans le λ-calcul via Dialectica linéaire.

Bref rappel

On rappelle ici la traduction *call-by-name* du λ-calcul dans LL :

$$[\![A \to B]\!] \equiv ![\![A]\!] \multimap [\![B]\!]$$

$$[\![A \times B]\!] \equiv ![\![A]\!] \otimes ![\![B]\!]$$

$$[\![A + B]\!] \equiv ![\![A]\!] \oplus ![\![B]\!]$$

$$[\![\Gamma \vdash A]\!] \equiv \bigotimes ![\![\Gamma]\!] \vdash [\![A]\!]$$

Pour interpréter le λ-calcul, on a besoin des deux propriétés suivantes :

Terme vide

Pour tout type A, il existe $\vdash \varnothing_A : \mathbb{W}(A)$.

Orthogonalité décidable

La relation $\perp_A$ est décidable. En particulier, il existe un λ-terme

$$@^A : \mathbb{W}(A) \to \mathbb{W}(A) \to \mathbb{C}(A) \to \mathbb{W}(A)$$

qui implémente le comportement suivant :

$$u_1 @_x^A u_2 \cong \texttt{if } u_1 \perp_A x \texttt{ then } u_2 \texttt{ else } u_1$$

As-tu résolu le problème de l'organisation ?

L'utilisation telle quelle de la traduction mène à une bureaucratie non-négligeable. On va plutôt utiliser les isomorphismes suivants :

$$[\![x_1 : \Gamma_1, \dots x_n : \Gamma_n \vdash t : A]\!] \cong \mathbb{W}(\Gamma) \to \begin{cases} \mathbb{W}(A) \\ \mathbb{C}(A) \to \mathbb{C}(\Gamma_1) \\ \vdots \\ \mathbb{C}(A) \to \mathbb{C}(\Gamma_n) \end{cases}$$

As-tu résolu le problème de l'organisation ?

L'utilisation telle quelle de la traduction mène à une bureaucratie non-négligeable. On va plutôt utiliser les isomorphismes suivants :

$$[\![x_1 : \Gamma_1, \dots x_n : \Gamma_n \vdash t : A]\!] \cong \mathbb{W}(\Gamma) \to \begin{cases} \mathbb{W}(A) \\ \mathbb{C}(A) \to \mathbb{C}(\Gamma_1) \\ \vdots \\ \mathbb{C}(A) \to \mathbb{C}(\Gamma_n) \end{cases}$$

Ce qui donne les traductions suivantes :

$$[\![\vec{x} : \Gamma \vdash t : A]\!] \equiv \begin{cases} \vec{x} : \mathbb{W}(\Gamma) \vdash t^\bullet : \mathbb{W}(A) \\ \vec{x} : \mathbb{W}(\Gamma) \vdash t_{x_1} : \mathbb{C}(A) \to \mathbb{C}(\Gamma_1) \\ \vdots \\ \vec{x} : \mathbb{W}(\Gamma) \vdash t_{x_n} : \mathbb{C}(A) \to \mathbb{C}(\Gamma_n) \end{cases}$$

Traduction (enfin)

Pour $(-)^\bullet$:

$$
\begin{array}{lcl}
x^\bullet & \equiv & x \\
(\lambda x.\, t)^\bullet & \equiv & \left\{ \begin{array}{l} \lambda x.\, t^\bullet \\ \lambda \pi x.\, t_x\, \pi \end{array} \right. \\
(t\, u)^\bullet & \equiv & (\mathtt{fst}\ t^\bullet)\, u^\bullet
\end{array}
$$

Traduction II

Pour t_x :

$$
\begin{array}{lcl}
x_x & \equiv & \lambda\pi.\,\pi \\
 & : & \mathbb{C}(A) \to \mathbb{C}(A) \\
y_x & \equiv & \lambda\pi.\,\varnothing \\
 & : & \mathbb{C}(A) \to \mathbb{C}(\Gamma_i) \\
(\lambda y.\,t)_x & \equiv & \lambda(y,\pi).\,t_x\,\pi \\
 & : & \mathbb{W}(A) \times \mathbb{C}(B) \to \mathbb{C}(\Gamma_i) \\
(t\,u)_x & \equiv & \lambda\pi.\,u_x\,((\mathtt{snd}\ t^\bullet)\,\pi\,u^\bullet)\ @_\pi\ t_x\,(u^\bullet,\pi) \\
 & : & \mathbb{C}(B) \to \mathbb{C}(\Gamma_i)
\end{array}
$$

Ça marche !

Correction

Si $\vdash t : A$, alors $\vdash [\![t]\!] : \mathbb{W}(A)$, et de plus, pour tout $\pi : \mathbb{C}(A)$, $t \perp_A \pi$.

Et si j'arrêtais de vous prendre pour des idiots ?

L'utilisation de $\varnothing$ et de @ est encore un encodage de Dialectica.

Et si j'arrêtais de vous prendre pour des idiots ?

L'utilisation de $\varnothing$ et de $@$ est encore un encodage de Dialectica.

- En fait, on veut des **listes** !
- On change juste :

$$\begin{aligned} \mathbb{C}(!A) &\equiv \mathbb{W}(A) \to \mathbb{C}(A) \\ \mathbb{C}(!A) &\equiv \mathbb{W}(A) \to \texttt{list}\ \mathbb{C}(A) \end{aligned}$$

- On ne change (presque) pas l'interprétation des termes :
  - $\varnothing$ devient la liste vide ;
  - $@$ devient la concaténation de liste
  - ...
- Plus vraiment besoin d'orthogonalité...

Et l'interprétation calculatoire dans tout ça ?

La version listifiée de Dialectica nous donne les types suivants :

$$[\![\vec{x} : \Gamma \vdash t : A]\!] \equiv \begin{cases} \vec{x} : \mathbb{W}(\Gamma) \vdash t^\bullet : \mathbb{W}(A) \\ \vec{x} : \mathbb{W}(\Gamma) \vdash t_{x_1} : \mathbb{C}(A) \to \texttt{list}\ \mathbb{C}(\Gamma_1) \\ \vdots \\ \vec{x} : \mathbb{W}(\Gamma) \vdash t_{x_n} : \mathbb{C}(A) \to \texttt{list}\ \mathbb{C}(\Gamma_n) \end{cases}$$

Et l'interprétation calculatoire dans tout ça ?

La version listifiée de Dialectica nous donne les types suivants :

$$[\![\vec{x} : \Gamma \vdash t : A]\!] \equiv \begin{cases} \vec{x} : \mathbb{W}(\Gamma) \vdash t^\bullet : \mathbb{W}(A) \\ \vec{x} : \mathbb{W}(\Gamma) \vdash t_{x_1} : \mathbb{C}(A) \to \texttt{list}\ \mathbb{C}(\Gamma_1) \\ \vdots \\ \vec{x} : \mathbb{W}(\Gamma) \vdash t_{x_n} : \mathbb{C}(A) \to \texttt{list}\ \mathbb{C}(\Gamma_n) \end{cases}$$

- $t^\bullet$ est clairement un relèvement de t ;

Et l'interprétation calculatoire dans tout ça ?

La version listifiée de Dialectica nous donne les types suivants :

$$[\![\vec{x} : \Gamma \vdash t : A]\!] \equiv \begin{cases} \vec{x} : \mathbb{W}(\Gamma) \vdash t^\bullet : \mathbb{W}(A) \\ \vec{x} : \mathbb{W}(\Gamma) \vdash t_{x_1} : \mathbb{C}(A) \to \texttt{list}\ \mathbb{C}(\Gamma_1) \\ \vdots \\ \vec{x} : \mathbb{W}(\Gamma) \vdash t_{x_n} : \mathbb{C}(A) \to \texttt{list}\ \mathbb{C}(\Gamma_n) \end{cases}$$

- $t^\bullet$ est clairement un relèvement de t ;
- Mais *quid* de t_{x_i} ?

Le suspense est à son comble

Un bref interlude de ~~publicité~~ **définitions** pour vous refourguer la KAM.

Le suspense est à son comble

Un bref interlude de ~~publicité~~ **définitions** pour vous refourguer la KAM.

Clôtures	c	$::=$	(t, σ)
Environnements	σ	$::=$	$\emptyset \mid \sigma + (x := c)$
Piles	π	$::=$	$\varepsilon \mid c \cdot \pi$
Processus	p	$::=$	$\langle c \mid \pi \rangle$

Push	$\langle (t\, u, \sigma) \mid \pi \rangle$	$\to$	$\langle (t, \sigma) \mid (u, \sigma) \cdot \pi \rangle$
Pop	$\langle (\lambda x.\, t, \sigma) \mid c \cdot \pi \rangle$	$\to$	$\langle (t, \sigma + (x := c)) \mid \pi \rangle$
Grab	$\langle (x, \sigma + (x := c)) \mid \pi \rangle$	$\to$	$\langle c \mid \pi \rangle$
Garbage	$\langle (x, \sigma + (y := c)) \mid \pi \rangle$	$\to$	$\langle (x, \sigma) \mid \pi \rangle$

La machine de Krivine™

Sous les clôtures, la pile

Soient :

- un terme $\vec{x} : \Gamma \vdash t : A$
- une clôture $\sigma \vdash \Gamma$
- une pile $\vdash \pi : A^{\perp}$ (i.e. $[\![\pi]\!] : \mathbb{C}(A)$)

Sous les clôtures, la pile

Soient :

- un terme $\vec{x} : \Gamma \vdash t : A$
- une clôture $\sigma \vdash \Gamma$
- une pile $\vdash \pi : A^{\perp}$ (i.e. $[\![\pi]\!] : \mathbb{C}(A)$)

Alors $t_{x_i}\,\pi$ est la liste constituée des **piles rencontrées par** x_i dans l'évaluation de $\langle (t, \sigma) \mid \pi \rangle$, i.e.

$$(t_{x_i}\{\vec{x} := \sigma\})\,\pi = [\rho_1; \dots \rho_m]$$

$$\begin{array}{ccc} \langle (t, \sigma) \mid \pi \rangle & \longrightarrow^* & \langle (x_i, \sigma_1) \mid \rho_1 \rangle \\ & \vdots & \vdots \\ & \longrightarrow^* & \langle (x_i, \sigma_m) \mid \rho_m \rangle \end{array}$$

Regardez

$$
\begin{array}{lcl}
x_x & \equiv & \lambda\pi.\,[\pi] \\
 & : & \mathbb{C}(A) \to \mathbb{C}(A) \\
y_x & \equiv & \lambda\pi.\,[\,] \\
 & : & \mathbb{C}(A) \to \mathbb{C}(\Gamma_i) \\
(\lambda y.\,t)_x & \equiv & \lambda(y,\pi).\,t_x\,\pi \\
 & : & \mathbb{W}(A) \times \mathbb{C}(B) \to \mathbb{C}(\Gamma_i) \\
(t\,u)_x & \equiv & \lambda\pi.\,(((\mathtt{snd}\ t^\bullet)\,\pi\,u^\bullet) \gg\!= u_x) \mathbin{@} t_x\,(u^\bullet,\pi) \\
 & : & \mathbb{C}(B) \to \mathbb{C}(\Gamma_i)
\end{array}
$$

Regardez

$$
\begin{array}{lcl}
x_x & \equiv & \lambda\pi.\,[\pi] \\
 & : & \mathbb{C}(A) \to \mathbb{C}(A) \\
y_x & \equiv & \lambda\pi.\,[\,] \\
 & : & \mathbb{C}(A) \to \mathbb{C}(\Gamma_i) \\
(\lambda y.\,t)_x & \equiv & \lambda(y,\pi).\,t_x\,\pi \\
 & : & \mathbb{W}(A) \times \mathbb{C}(B) \to \mathbb{C}(\Gamma_i) \\
(t\,u)_x & \equiv & \lambda\pi.\,(((\mathtt{snd}\ t^\bullet)\,\pi\,u^\bullet) \gg\!= u_x) \mathbin{@} t_x\,(u^\bullet,\pi) \\
 & : & \mathbb{C}(B) \to \mathbb{C}(\Gamma_i)
\end{array}
$$

(On peut généraliser aux types algébriques sans problème.)

Dialectica Reloaded

- La version standard de Dialectica ne retourne qu'une seule pile
  - $\rightsquigarrow$ la première qui ne soit pas bidon
  - $\rightsquigarrow$ testée dynamiquement

Dialectica Reloaded

- La version standard de Dialectica ne retourne qu'une seule pile
  - $\rightsquigarrow$ la première qui ne soit pas bidon
  - $\rightsquigarrow$ testée dynamiquement
- La version protohistorique de Dialectica est presque CBN
  - $\rightsquigarrow$ elle est *call-by-ad-hoc-because-I-am-protohistoric*

Dialectica Reloaded

- La version standard de Dialectica ne retourne qu'une seule pile
  - $\rightsquigarrow$ la première qui ne soit pas bidon
  - $\rightsquigarrow$ testée dynamiquement
- La version protohistorique de Dialectica est presque CBN
  - $\rightsquigarrow$ elle est *call-by-ad-hoc-because-I-am-protohistoric*
- Il s'agit d'une forme de contrôle délimité
  - $\rightsquigarrow$ accès aux piles de première classe via $(-)_x$

Dialectica Reloaded

- La version standard de Dialectica ne retourne qu'une seule pile
  - $\rightsquigarrow$ la première qui ne soit pas bidon
  - $\rightsquigarrow$ testée dynamiquement
- La version protohistorique de Dialectica est presque CBN
  - $\rightsquigarrow$ elle est *call-by-ad-hoc-because-I-am-protohistoric*
- Il s'agit d'une forme de contrôle délimité
  - $\rightsquigarrow$ accès aux piles de première classe via $(-)_x$
- On peut faire la même transformation dans d'autres conventions d'appel
  - $\rightsquigarrow$ En utilisant d'autres traductions de LL
  - $\rightsquigarrow$ Version CBV basée sur la traduction barbante
  - $\rightsquigarrow$ CBN, CBV classiques...
  - $\rightsquigarrow$ Autres ?

J'ai menti (je ne le ferai plus, juré)

En fait, ce n'est pas tout à fait ça. Il y a un problème.

J'ai menti (je ne le ferai plus, juré)

En fait, ce n'est pas tout à fait ça. Il y a un problème.

- Les piles produites sont bien exactement celles rencontrées...

J'ai menti (je ne le ferai plus, juré)

En fait, ce n'est pas tout à fait ça. Il y a un problème.

- Les piles produites sont bien exactement celles rencontrées...
- Elles ont bien la bonne multiplicité...

J'ai menti (je ne le ferai plus, juré)

En fait, ce n'est pas tout à fait ça. Il y a un problème.

- Les piles produites sont bien exactement celles rencontrées...
- Elles ont bien la bonne multiplicité...
- Mais elles sont dans le **désordre** !
- Listes vs. multisets finis...

J'ai menti (je ne le ferai plus, juré)

En fait, ce n'est pas tout à fait ça. Il y a un problème.

- Les piles produites sont bien exactement celles rencontrées...
- Elles ont bien la bonne multiplicité...
- Mais elles sont dans le **désordre** !
- Listes vs. multisets finis...

La faute à la traduction de l'application (plus généralement à la duplication).

$$(t\,u)_x \equiv \lambda\pi.\,(((\mathtt{snd}\ t^\bullet)\,\pi\,u^\bullet) \gg\!= u_x) \mathbin{@} t_x\,(u^\bullet, \pi)$$

Un problème profond

- On a une séquentialité imposée par la KAM
- On veut la refléter dans la traduction

Un problème profond

- On a une séquentialité imposée par la KAM
- On veut la refléter dans la traduction
- Mais on ne peut pas !
- La traduction du $\mathcal{Y}$ est trop symétrique
  - $\rightsquigarrow$ On veut de l'*interleaving*
  - $\rightsquigarrow$ Dialectica n'est pas faite pour ça...
  - $\rightsquigarrow$ Version polarisée ? Tensorielle ? Autre chose ?
  - $\rightsquigarrow$ Demandez-moi pour plus de détails

J'ai menti (encore !)

On n'a pas tout à fait atteint Dialectica proto-historique.

- Pour MP et IP, on a besoin de $\varnothing$ du côté preuve
  - $\rightsquigarrow$ pas seulement comme pile !
  - $\rightsquigarrow$ $\varnothing$ s'y comporte comme une exception

J'ai menti (encore !)

On n'a pas tout à fait atteint Dialectica proto-historique.

- Pour MP et IP, on a besoin de $\varnothing$ du côté preuve
  - $\rightsquigarrow$ pas seulement comme pile !
  - $\rightsquigarrow$ $\varnothing$ s'y comporte comme une exception

- En l'état, on a une version faible de MP

$$MP_w : \neg(\forall x : A. \neg P[x]) \to \texttt{list}\ (\exists x : A. P[x])$$

- Et d'IP point !

J'ai menti (encore !)

On n'a pas tout à fait atteint Dialectica proto-historique.

- Pour MP et IP, on a besoin de $\varnothing$ du côté preuve
  - $\rightsquigarrow$ pas seulement comme pile !
  - $\rightsquigarrow$ $\varnothing$ s'y comporte comme une exception

- En l'état, on a une version faible de MP

$$MP_w : \neg(\forall x : A. \neg P[x]) \to \texttt{list}\ (\exists x : A. P[x])$$

- Et d'IP point !

... à priori pas très grave, et plus ou moins réparable.

Cours camarade, le vieux monde est derrière toi !

- Faisons table rase de la logique du premier ordre...
- On se laisse guider par le contenu calculatoire de Dialectica
- On applique cette transformation aux types dépendants
  - $\rightsquigarrow$ subsume la logique du premier ordre ;
  - $\rightsquigarrow$ donne un contenu calculatoire au $\forall$;
  - $\rightsquigarrow$ vers CC^{ω} et au-delà !

Les grandes lignes

- On garde la structure CBN du λ-calcul
  - $\rightsquigarrow$ elle se relève toute seule au types dépendants
  - $\rightsquigarrow$ aucun effort à faire !

Les grandes lignes

- On garde la structure CBN du λ-calcul
  - $\rightsquigarrow$ elle se relève toute seule au types dépendants
  - $\rightsquigarrow$ aucun effort à faire !
- On donne un **contenu calculatoire vide** aux types
  - $\rightsquigarrow$ c'est décevant ;
  - $\rightsquigarrow$ mais ça semble marcher...
  - $\rightsquigarrow$ et je ne vois pas comment faire sinon !

Traduction des types

Idée : si A est un type,

$$A^\bullet \equiv (\mathbb{W}(A), \mathbb{C}(A)) : \mathtt{Type} \times \mathtt{Type}$$

Traduction des types

Idée : si A est un type,

$$A^\bullet \equiv (\mathbb{W}(A), \mathbb{C}(A)) : \mathtt{Type} \times \mathtt{Type}$$

On obtient :

$$\begin{array}{lcl}
\mathtt{Type}^\bullet & \equiv & (\mathtt{Type} \times \mathtt{Type}, 1) \\
\mathtt{Type}_x & \equiv & \lambda\pi.\,[\,] \\
(\Pi y : A.\, B)^\bullet & \equiv & \left(\begin{array}{c} (\Pi y : \mathbb{W}(A).\, \mathbb{W}(B)) \\ \times \\ (\Pi y : \mathbb{W}(A).\, \mathbb{C}(B) \to \mathtt{list}\ \mathbb{C}(A)) \end{array} , \Sigma y : \mathbb{W}(A).\, \mathbb{C}(B) \right) \\
(\Pi y : A.\, B)_x & \equiv & \lambda\pi.\,[\,]
\end{array}$$

Correction

La traduction est (presque sûrement) correcte :

- Elle est correcte si la logique de destination est extensionnelle ;
- Modulo suffisament d'η-expansions, cela semble marcher...

Le problème provient essentiellement du fait qu'on veut :

$$t \gg\!= (\lambda\pi.\,[\,]) =_{\beta\eta} [\,]$$

pour vérifier la règle de conversion...

Inductifs *please*

Le tableau est plus sombre pour encoder l'élimination dépendante...

Inductifs *please*

Le tableau est plus sombre pour encoder l'élimination dépendante...

- Les premières investigations concluent à la nécessité de la polarisation
- Mais comme Dialectica est déjà cassée à cause de la séquentialité, *who cares* ?

Conclusion

- En fait, Dialectica, c'est assez bête.
  - $\rightsquigarrow$ une fois qu'on a retiré les artéfacts d'encodage

Conclusion

- En fait, Dialectica, c'est assez bête.
  - $\rightsquigarrow$ une fois qu'on a retiré les artéfacts d'encodage
- C'est une approximation de deux effets de bord :
  - $\rightsquigarrow$ Un fragment de contrôle délimité (avec $(-)_x$)
  - $\rightsquigarrow$ Un encodage primitif d'exceptions (avec $\varnothing$)

Conclusion

- En fait, Dialectica, c'est assez bête.
  - $\rightsquigarrow$ une fois qu'on a retiré les artéfacts d'encodage
- C'est une approximation de deux effets de bord :
  - $\rightsquigarrow$ Un fragment de contrôle délimité (avec $(-)_x$)
  - $\rightsquigarrow$ Un encodage primitif d'exceptions (avec $\varnothing$)
- Mais Dialectica est cassée :
  - $\rightsquigarrow$ elle ne respecte pas la séquentialité
  - $\rightsquigarrow$ comment la réparer ?
  - $\rightsquigarrow$ vaut-il mieux considérer la KAM comme plus primitive ?

Conclusion

- En fait, Dialectica, c'est assez bête.
  - $\rightsquigarrow$ une fois qu'on a retiré les artéfacts d'encodage
- C'est une approximation de deux effets de bord :
  - $\rightsquigarrow$ Un fragment de contrôle délimité (avec $(-)_x$)
  - $\rightsquigarrow$ Un encodage primitif d'exceptions (avec $\varnothing$)
- Mais Dialectica est cassée :
  - $\rightsquigarrow$ elle ne respecte pas la séquentialité
  - $\rightsquigarrow$ comment la réparer ?
  - $\rightsquigarrow$ vaut-il mieux considérer la KAM comme plus primitive ?
- Le fragment de contrôle délimité se relève facilement à CC^ω
  - $\rightsquigarrow$ faut-il l'implémenter ?
  - $\rightsquigarrow$ comment ajouter l'élimination dépendante ?

Des questions ?