

Pour s'asseoir sur les fondations

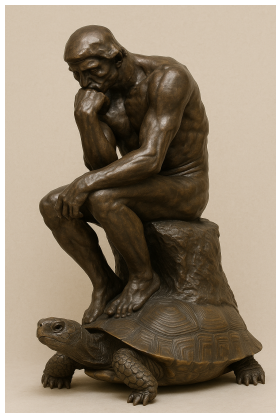


FIG. 1. JUSQU'OU UN INTELLECTUEL ASSIS PEUT-IL ALLER ?

Pierre-Marie Pédrot
INRIA

Collège de France, 02/06/25

« FORMALISATION DES MATHÉMATIQUES ET TYPES DÉPENDANTS »

Tout un programme

« FORMALISATION DES MATHÉMATIQUES ET TYPES DÉPENDANTS »

Une tension intrinsèque immédiate

« FORMALISATION DES MATHÉMATIQUES ET TYPES DÉPENDANTS »

Une tension intrinsèque immédiate

↪ À ma gauche, le monde mathématique

« FORMALISATION DES MATHÉMATIQUES ET TYPES DÉPENDANTS »

Une tension intrinsèque immédiate

↪ À ma gauche, le monde mathématique

↪ À ma droite, le monde informatique

« FORMALISATION DES MATHÉMATIQUES ET TYPES DÉPENDANTS »

Une tension intrinsèque immédiate

↪ À ma gauche, le monde mathématique

↪ À ma droite, le monde informatique

L'interprétation du ET est heureusement laissée au lecteur !

- Intersection ? $A \cap B$
- Juxtaposition ? $A \& B$
- Superposition ? $A \otimes B$

Tout un programme

« FORMALISATION DES MATHÉMATIQUES ET TYPES DÉPENDANTS »

Une tension intrinsèque immédiate

↪ À ma gauche, le monde mathématique

↪ À ma droite, le monde informatique

L'interprétation du ET est heureusement laissée au lecteur !

- Intersection ? $A \cap B$
- Juxtaposition ? $A \& B$
- Superposition ? $A \otimes B$

Comment en est-on arrivé là ?

L'équivalence preuve-programme, alias Curry-Howard

L'équivalence preuve-programme, alias Curry-Howard

Le mathématicien

Théorème. Pour tout $n \in \mathbb{N}$, il existe $p \in \mathbb{N}$ t.q. $n = 2p$ ou $n = 2p + 1$.

Preuve. Par induction sur n .

- Si $n = 0$, trivial car $p = 2 \times 0$.
- Sinon, soit $n = m + 1$. Par induction, soit p t.q. $m = 2p$ ou $m = 2p + 1$.
 - Dans le cas 1, $n = 2p + 1$.
 - Sinon, $n = 2(p + 1)$.

Le programmeur

```
val div2 : int -> int * bool
(* [div2 n] renvoie [n / 2] et
la parité de [n]. *)
```

```
let rec div2 n = match n with
| 0 -> (0, true)
| m + 1 ->
    let (p, even) = div2 m in
    if even then (p, false)
    else (p + 1, true)
```

L'équivalence preuve-programme, alias Curry-Howard

Le mathématicien

Théorème. Pour tout $n \in \mathbb{N}$, il existe $p \in \mathbb{N}$ t.q. $n = 2p$ ou $n = 2p + 1$.

Preuve. Par induction sur n .

- Si $n = 0$, trivial car $p = 2 \times 0$.
- Sinon, soit $n = m + 1$. Par induction, soit p t.q. $m = 2p$ ou $m = 2p + 1$.
 - Dans le cas 1, $n = 2p + 1$.
 - Sinon, $n = 2(p + 1)$.

Le programmeur

```
val div2 : int -> int * bool
(* [div2 n] renvoie [n / 2] et
la parité de [n]. *)
```

```
let rec div2 n = match n with
| 0 -> (0, true)
| m + 1 ->
    let (p, even) = div2 m in
    if even then (p, false)
    else (p + 1, true)
```

C'est la même chose !

Constructions dans un monde qui bouge

Cette équivalence est à la base de la théorie des types

Constructions dans un monde qui bouge

Cette équivalence est à la base de la théorie des types

Un **système logique** très puissant :

- On peut y écrire toutes les mathématiques conventionnelles
- Notion de calcul de première classe et inductifs de l'enfer

Cette équivalence est à la base de la théorie des types

Un **système logique** très puissant :

- On peut y écrire toutes les mathématiques conventionnelles
- Notion de calcul de première classe et inductifs de l'enfer

Un **langage de programmation** très expressif :

- Des types extrêmement fins pour éliminer les *bugs*
- Pas de séparation nette entre typage et *runtime*

Constructions dans un monde qui bouge

Cette équivalence est à la base de la théorie des types

Un **système logique** très puissant :

- On peut y écrire toutes les mathématiques conventionnelles
- Notion de calcul de première classe et inductifs de l'enfer

Un **langage de programmation** très expressif :

- Des types extrêmement fins pour éliminer les *bugs*
- Pas de séparation nette entre typage et *runtime*

MLTT / CIC : l'incarnation de l'équivalence preuve-programme

Constructions dans un monde qui bouge

Cette équivalence est à la base de la théorie des types

Un **système logique** très puissant :

- On peut y écrire toutes les mathématiques conventionnelles
- Notion de calcul de première classe et inductifs de l'enfer

Un **langage de programmation** très expressif :

- Des types extrêmement fins pour éliminer les *bugs*
- Pas de séparation nette entre typage et *runtime*

MLTT / CIC : l'incarnation de l'équivalence preuve-programme

Des systèmes implémentés dans un certain nombre d'assistants à la preuve



D'où parles-tu, camarade ?

Qui suis-je ?

D'où parles-tu, camarade ?

Qui suis-je ?

UN INFORMATICIEN !

D'où parles-tu, camarade ?

Qui suis-je ?

UN INFORMATICIEN !

- Thèse soutenue en 2015
- Mon sujet d'étude est la théorie des types
- Un des principaux développeurs de l'assistant à la preuve Rocq

D'où parles-tu, camarade ?

Qui suis-je ?

UN INFORMATICIEN !

- Thèse soutenue en 2015
- Mon sujet d'étude est la théorie des types
- Un des principaux développeurs de l'assistant à la preuve Rocq

ATTENTION

Cet exsopé n'est pas un exsopé sur les isomorphismes

- Volontairement très haut niveau (pas de λ -termes !)
- Beaucoup d'enfonçage de portes ouvertes
- Public cible : les mathématiciens et les profanes

L'origine précise de la théorie des types est débattue

L'origine précise de la théorie des types est débattue

Selon la personne à qui on demande :

- Une structuralisation des ensembles (192X)
- Une origine philosophique constructiviste (197X)
- Un raffinement du λ -calcul (197X)
- L'algèbre universelle pour de vrai

Un peu d'histoire

L'origine précise de la théorie des types est débattue

Selon la personne à qui on demande :

- Une structuralisation des ensembles (192X)
- Une origine philosophique constructiviste (197X)
- Un raffinement du λ -calcul (197X)
- L'algèbre universelle pour de vrai

Moins de débat sur les premiers utilisateurs

Un peu d'histoire

L'origine précise de la théorie des types est débattue

Selon la personne à qui on demande :

- Une structuralisation des ensembles (192X)
- Une origine philosophique constructiviste (197X)
- Un raffinement du λ -calcul (197X)
- L'algèbre universelle pour de vrai

Moins de débat sur les premiers utilisateurs

LES INFORMATIENS !

Un peu d'histoire

L'origine précise de la théorie des types est débattue

Selon la personne à qui on demande :

- Une structuralisation des ensembles (192X)
- Une origine philosophique constructiviste (197X)
- Un raffinement du λ -calcul (197X)
- L'algèbre universelle pour de vrai

Moins de débat sur les premiers utilisateurs

LES INFORMATIENS !

Automath (1968) et surtout dans sa version moderne **Coq** (1984).

L'engouement des mathématiciens pour les assistants à la preuve est récent

- Moins de 10 ans
- Encore très niche
- Surreprésentation de certains domaines

L'engouement des mathématiciens pour les assistants à la preuve est récent

- Moins de 10 ans
- Encore très niche
- Surreprésentation de certains domaines

Un certain choc des cultures

- Pas les mêmes priorités
- Pas le même *background*
- Pas le même point de vue

Un assistant à la preuve force à se poser des questions

- Quelle structure de donnée utiliser ?
- Comment passer d'une représentation à une autre ?

Un assistant à la preuve force à se poser des questions

- Quelle structure de donnée utiliser ?
- Comment passer d'une représentation à une autre ?

... et très rapidement, des questions sur le langage

- Existe-t-il des types satisfaisant telle propriété ?
- Y a-t-il une représentation directe d'un encodage ?
- Puis-je transformer des isomorphismes en égalités ?

Le pourquoi du comment

Un assistant à la preuve force à se poser des questions

- Quelle structure de donnée utiliser ?
- Comment passer d'une représentation à une autre ?

... et très rapidement, des questions sur le langage

- Existe-t-il des types satisfaisant telle propriété ?
- Y a-t-il une représentation directe d'un encodage ?
- Puis-je transformer des isomorphismes en égalités ?

Ce sont des questions sur les **fondations**



Triste topique

En général, les mathématiciens ont une aversion pour la logique

↪ surtout les questions fondationnelles

↪ très particulièrement en France

Triste topique

En général, les mathématiciens ont une aversion pour la logique

↪ surtout les questions fondationnelles

↪ très particulièrement en France

Arguments usuels par ordre d'acceptabilité sociale :

- Ça ne m'intéresse pas

Triste topique

En général, les mathématiciens ont une aversion pour la logique

↪ surtout les questions fondationnelles

↪ très particulièrement en France

Arguments usuels par ordre d'acceptabilité sociale :

- Ça ne m'intéresse pas
- Ça ne sert à rien

Triste topique

En général, les mathématiciens ont une aversion pour la logique

↪ surtout les questions fondationnelles

↪ très particulièrement en France

Arguments usuels par ordre d'acceptabilité sociale :

- Ça ne m'intéresse pas
- Ça ne sert à rien
- « La logique n'est plus stérile, elle engendre des paradoxes. » (Poincaré)



Triste topique

En général, les mathématiciens ont une aversion pour la logique

↪ surtout les questions fondationnelles

↪ très particulièrement en France

Arguments usuels par ordre d'acceptabilité sociale :

- Ça ne m'intéresse pas
- Ça ne sert à rien
- « La logique n'est plus stérile, elle engendre des paradoxes. » (Poincaré)



Il semble y avoir une cause historico-sociologique

Hilbert, Bourbaki and the scorning of logic (A. Mathias, 2013)

QUAND ON FORMALISE, LES FONDATIONS C'EST IMPORTANT...

QUAND ON FORMALISE, LES FONDATIONS C'EST IMPORTANT...

- 10 raisons méconnues, la 9^{ème} va vous surprendre !

QUAND ON FORMALISE, LES FONDATIONS C'EST IMPORTANT...

- ~~10 raisons méconnues, la 9^{ème} va vous surprendre !~~

QUAND ON FORMALISE, LES FONDATIONS C'EST IMPORTANT...

- ~~10 raisons méconnues, la 9^{ème} va vous surprendre !~~
- ... pour pouvoir s'en passer.

QUAND ON FORMALISE, LES FONDATIONS C'EST IMPORTANT...

- ~~10 raisons méconnues, la 9^{ème} va vous surprendre !~~
- ... pour pouvoir s'en passer.

Dans le reste de l'exposé, je vais tenter de vous en convaincre

- à travers la correspondance de Curry-Howard
- en surjouant le côté turbo-informaticien
- j'assume mes biais et autres monomanies

Pourquoi tant de n ?

Radotage

La théorie des types est un langage de programmation

Pourquoi tant de n ?

Radotage

La théorie des types est un langage de programmation

Observation

Il existe **beaucoup** de langages de programmation.

Pourquoi tant de n ?

Radotage

La théorie des types est un langage de programmation

Observation

Il existe **beaucoup** de langages de programmation.

- ... Mais alors, beaucoup.
- Vraiment beaucoup.
- J'insiste.

Pourquoi tant de n ?

Radotage

La théorie des types est un langage de programmation

Observation

Il existe **beaucoup** de langages de programmation.

- ... Mais alors, beaucoup.
- Vraiment beaucoup.
- J'insiste.

Une recherche rapide suggère plus de 8000 sur des critères incertains.

↪ Tout étudiant en informatique implémente son propre langage un jour.

Tout est permis, mais rien n'est possible

Essentiellement tous équivalents (Turing-complétude)

Tout est permis, mais rien n'est possible

Essentiellement tous équivalents (Turing-complétude)

Pourquoi y a-t-il autant de langages de programmation ?

Essentiellement tous équivalents (Turing-complétude)

Pourquoi y a-t-il autant de langages de programmation ?

“— Beware of the Turing tarpit in which everything is possible but nothing of interest is easy.”
(Perlis)

Tout est permis, mais rien n'est possible

Essentiellement tous équivalents (Turing-complétude)

Pourquoi y a-t-il autant de langages de programmation ?

“— Beware of the Turing tarpit in which everything is possible but nothing of interest is easy.”
(Perlis)

La notion d'*intéressant* n'est pas absolue !

- Peu de points en commun entre les développeurs
- Pléthore de langages pour chaque besoin
- Il n'existe pas de *meilleur* langage (mais des langages *meilleurs* ?...)

Quel est le paysage du côté de la logique ?

Quel est le paysage du côté de la logique ?

Pas de logique universelle (incomplétude de Gödel)

Quel est le paysage du côté de la logique ?

Pas de logique universelle (incomplétude de Gödel)

- les fondations sont fondamentalement non-équivalentes
- mais en pratique cette question n'est pas pertinente
- expressivité logique $\neq$ expressivité linguistique



Quel est le paysage du côté de la logique ?

Pas de logique universelle (incomplétude de Gödel)

- les fondations sont fondamentalement non-équivalentes
- mais en pratique cette question n'est pas pertinente
- expressivité logique $\neq$ expressivité linguistique



Assistants à la preuve matures basés sur la théorie des types : environ $3^\dagger$

Quel est le paysage du côté de la logique ?

Pas de logique universelle (incomplétude de Gödel)

- les fondations sont fondamentalement non-équivalentes
- mais en pratique cette question n'est pas pertinente
- expressivité logique $\neq$ expressivité linguistique



Assistants à la preuve matures basés sur la théorie des types : environ $3^\dagger$

Il y a ici un paradoxe majeur !

De pire en pire

En fait, c'est bien pire que ça.

De pire en pire

En fait, c'est bien pire que ça.

« La » mathématique n'existe pas

- Une unité de surface qui se craquelle très vite
- Des objets manipulés très différents
- Des techniques de preuves extrêmement variables

De pire en pire

En fait, c'est bien pire que ça.

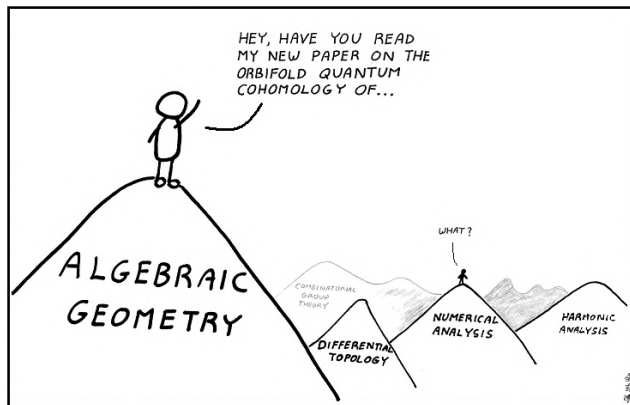
« La » mathématique n'existe pas

- Une unité de surface qui se craquelle très vite
- Des objets manipulés très différents
- Des techniques de preuves extrêmement variables

Même les langages de surface sont incomparables

- algèbre vs. analyse
- symboles vs. représentation pictoriale
- « modèle intuitif »

C'est pas moi qui le dit



The Landscape of Modern Mathematics

Abstruse Goose (CC-BY-NC 3.0)

C'est une feature, pas un bug

- Une sorte de Sapir-Whorf à l'envers
- Le langage **doit** évoluer pour mieux manipuler les concepts
- *Premier Contact* vs. platonisme ?

C'est une feature, pas un bug

- Une sorte de Sapir-Whorf à l'envers
- Le langage **doit** évoluer pour mieux manipuler les concepts
- *Premier Contact* vs. platonisme ?

Cette diversité doit être portée à la formalisation des mathématiques

- Une condition *sine qua non* pour passer à l'échelle
- Langages de haut niveau $\Rightarrow$ faisabilité
- Nous en sommes encore aux cartes perforées

C'est une feature, pas un bug

- Une sorte de Sapir-Whorf à l'envers
- Le langage **doit** évoluer pour mieux manipuler les concepts
- *Premier Contact* vs. platonisme ?

Cette diversité doit être portée à la formalisation des mathématiques

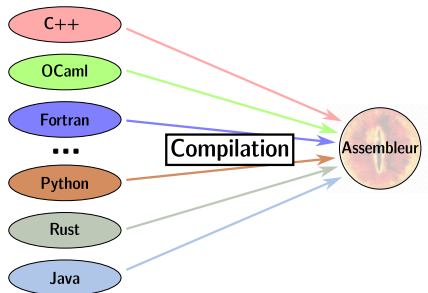
- Une condition *sine qua non* pour passer à l'échelle
- Langages de haut niveau $\Rightarrow$ faisabilité
- Nous en sommes encore aux cartes perforées

Comment éviter les schismes ?

Pluralité dans le monisme

En informatique

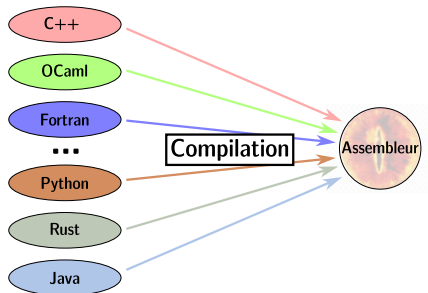
Le multilinguisme est résolu par la **compilation**



Pluralité dans le monisme

En informatique

Le multilinguisme est résolu par la **compilation**

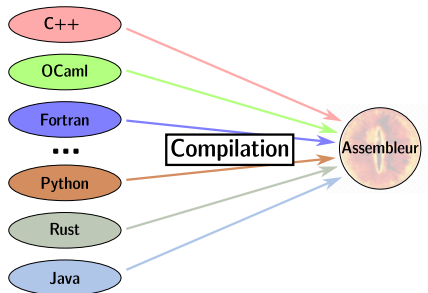


(* Photo non-contractuelle. Peut contenir des traces de fruits à coque et de bytecode VM.)

Pluralité dans le monisme

En informatique

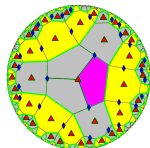
Le multilinguisme est résolu par la **compilation**



(* Photo non-contractuelle. Peut contenir des traces de fruits à coque et de bytecode VM.)

L'assembleur de notre machine sont nos fondations à nous

Le temps est venu de compiler...



...
à bec frisé

COHRQUI
Cohomologie
des orbivariétés
quantiques

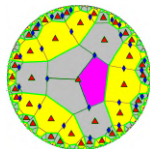
COMPILATION

Théorie
des types

...
à poils bleus

```
040: 00 7e 01 17 84 00 00  c# nope 0x0(\rax,\rax,1)
040: 00 00 00
040: 4c 89 f0                mov     %r14,%rax
040: 4d 8d 76 f8             lea     -0x0(\r14),%r14
040: 4c 89 68 f8             mov     %r13,-0x0(\rax)
040: 4c 89 f8                mov     %r15,%rax
040: 4d 8d 77 04             lea     0x4(\r15),%r15
040: 4d 8d 10                movslq  (%rax),%rdx
040: 4d 8d 70 ff ff ff ff    mov     -0x0(%rip,%rax
040: 4c 8d 2c 00             mov     (%rax,%rdx,8),%r13
040: 4c 89 f0                mov     %r14,%rax
040: 4d 8d 16                mov     (%r14),%rdx
040: 4d 8d 27 01             add     $0x1,%r15
040: 4d 8d 76 e8             lea     -0x10(\r14),%r14
040: 4d 8d 58 f8             mov     %rdx,-0x0(\rax)
040: 4d 8d 5d 6d ff ff ff    mov     -0x0(%rip,%rax
040: 4d 89 5d e8             mov     %rdx,-0x10(%rax)
040: 4d 8d 54 1b 01         lea     0x1(%rdx,%rax,1),%rdx
040: 4c 89 78 f8             mov     %r15,%rax
040: 4c 89 e8                mov     %r13,%rax
040: 4d 8d 10                mov     %rdx,%rax
040: 4d 8d 7d 00             mov     0x0(\r13),%r15
040: 4d c7 85 6d ff ff ff    cmovq  %rdx,-0x0(%rip)
040: 00 00 00 00
```


Le temps est venu de compiler...



...
à bec frisé

COHRQUI
Cohomologie
des orbivariétés
quantiques

COMPILATION

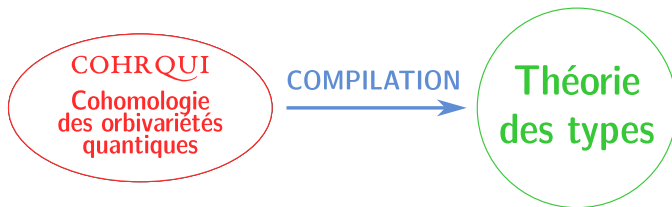
Théorie
des types

...
à poils bleus

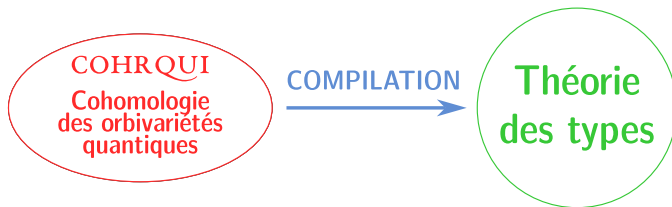
```
000: 00 2c ff ff 04 00 00  c3 nopw 0x0(%rax,%rax,1)
000: 00 00 00
000: 4c 80 f0
003: 4d 80 78 f3  ldd  -0x0(%rax),%r14
007: 4c 80 68 f3  mov  %r13,-0x0(%rax)
00a: 4c 80 f0  mov  %r15,%rax
00e: 4d 80 77 04  ldd  0x0(%r13),%r15
012: 4d 81 00  movlq (%rax),%rdx
015: 4d 80 25 73 ff ff ff  mov  -0x0(%rip,%rax,%r13)
01c: 4c 80 2c 00  mov  (%rax,%rdx,8),%r13
020: 4c 80 f0  mov  %r14,%rax
023: 4d 80 36  mov  (%r14),%rdx
026: 4d 83 07 f0  mov  0x0,%r13
02a: 4d 80 76 00  ldd  -0x0(%r14),%r14
02e: 4d 80 50 f3  mov  %rdx,-0x0(%rax)
032: 4d 80 50 60 ff ff ff  mov  -0x0(%rip,%rax,%r13)
039: 4d 80 50 00  mov  %rdx,-0x0(%rax)
03d: 4d 80 50 20 01  ldd  0x1(%rdx,%rax,4),%rdx
072: 4c 80 70 f3  mov  %r15,-0x0(%rax)
076: 4c 80 00  mov  %r13,%rax
07a: 4d 80 00  mov  %rdx,(%rax)
07d: 4d 80 74 00  mov  0x0(%r13),%r15
07e: 4d c7 8c 00 ff ff ff  movq  0x0,-0x0(%rip)
202:
```

... pour s'asseoir sur les fondations !

Plus facile à dire qu'à faire



Plus facile à dire qu'à faire



- La source : c'est (en partie) le boulot des matheux
- La compilation : de vrais morceaux de mathématiques
- La théorie cible : un sujet à part entière

Plus facile à dire qu'à faire



- La source : c'est (en partie) le boulot des matheux
- La compilation : de vrais morceaux de mathématiques
- La théorie cible : un sujet à part entière

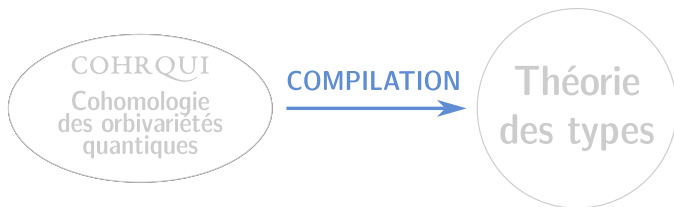
Appel à contribution : du boulot pour les décennies à venir

DANS LA SUITE DE CET EXPOSÉ

Un échantillon absolument objectif des défis

- ① Compilation : les calculs, c'est douloureux
- ② Assembleur : CIC citron ou CIC orange ?
- ③ Pas de côté : et les assistants à la preuve dans tout ça

1. Les calculs, c'est douloureux

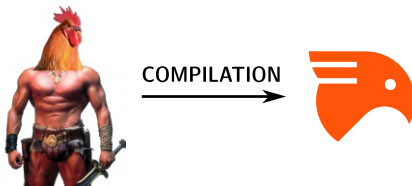


Rencontre du troisième type

Cas idéal : le langage source est **aussi** une théorie des types

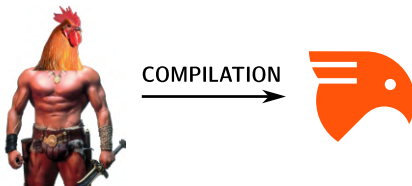
Rencontre du troisième type

Cas idéal : le langage source est **aussi** une théorie des types



Rencontre du troisième type

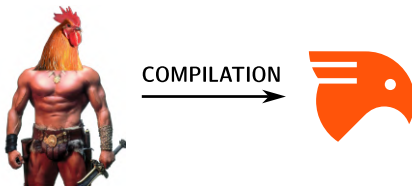
Cas idéal : le langage source est **aussi** une théorie des types



- étend la puissance expressive
- permet de réutiliser les bibliothèques de la cible
- réduit l'impédance entre les théories

Rencontre du troisième type

Cas idéal : le langage source est **aussi** une théorie des types



- étend la puissance expressive
- permet de réutiliser les bibliothèques de la cible
- réduit l'impédance entre les théories

Mot-clef : approche **synthétique**

- Homotopie synthétique $\sim$ MLTT + univalence
- Calculabilité synthétique $\sim$ MLTT + CT
- ...

La logique mène à tout à condition d'en sortir

Bonne nouvelle 1 : techniques archi-connues dans les topoi

(Pré)faisceaux alias LA LOGIQUE IKEA

- usine à gaz catégorique $\Rightarrow$ compilation à la pogne
- universel pour les logiques « positives »
- ma logique en kit



La logique mène à tout à condition d'en sortir

Bonne nouvelle 1 : techniques archi-connues dans les topoi

(Pré)faisceaux alias LA LOGIQUE IKEA

- usine à gaz catégorique $\Rightarrow$ compilation à la pogne
- universel pour les logiques « positives »
- ma logique en kit



Bonne nouvelle 2 : Théorie des types $\sim$ topoi

La logique mène à tout à condition d'en sortir

Bonne nouvelle 1 : techniques archi-connues dans les topoi

(Pré)faisceaux alias LA LOGIQUE IKEA

- usine à gaz catégorique $\Rightarrow$ compilation à la pogne
- universel pour les logiques « positives »
- ma logique en kit



Bonne nouvelle 2 : Théorie des types $\sim$ topoi

Conclusion ?

On peut implémenter les (pré)faisceaux en TT par compilation (?)

Et le désir s'accroît quand les faisceaux reculent

Conclusion ?

On peut implémenter les (pré)faisceaux en TT par compilation (?)

Et le désir s'accroît quand les faisceaux reculent

~~Conclusion 2~~

~~On peut implémenter L₁ (pas) faisceaux - TT par compilation (?)~~

Et le désir s'accroît quand les faisceaux reculent

~~Conclusion 2~~

~~On peut implémenter la (première) faisceau TT par compilation (?)~~

Radotage

La théorie des types est un langage de programmation

Et le désir s'accroît quand les faisceaux reculent

Conclusion 2

On peut implémenter le λ -calcul (les faisceaux) en TT par compilation (?)

Radotage

La théorie des types est un langage de programmation

Deux notions d'égalité : **définitionnelle** et **propositionnelle**

$\vdash M \equiv N : A$	vs	$\vdash P : M =_A N$
jugement		type
calcul		logique

Coincident dans les topoi, pas en théorie des types !

Et le désir s'accroît quand les faisceaux reculent

Conclusion 2

On peut implémenter les préfaisceaux TT par compilation (?)

Radotage

La théorie des types est un langage de programmation

Deux notions d'égalité : **définitionnelle** et **propositionnelle**

$\vdash M \equiv N : A$	vs	$\vdash P : M =_A N$
jugement		type
calcul		logique

Coincident dans les topoi, pas en théorie des types !

La traduction de préfaisceaux repose sur cette identification !

Solution 1 : toposifier la cible

- régression technique
- perte de propriétés critiques pour l'implémentation

Solution 1 : toposifier la cible

- régression technique
- perte de propriétés critiques pour l'implémentation

Solution 2 : changer complètement la traduction

préfaisceaux ensemblistes $\Rightarrow$ types préfascistes

- inspirés par la théorie de la programmation
- préservent la conversion
- vraiment pas des préfaisceaux
- (remarque sur les faisceaux *intensionnellement* laissée vide)

Solution 1 : toposifier la cible

- régression technique
- perte de propriétés critiques pour l'implémentation

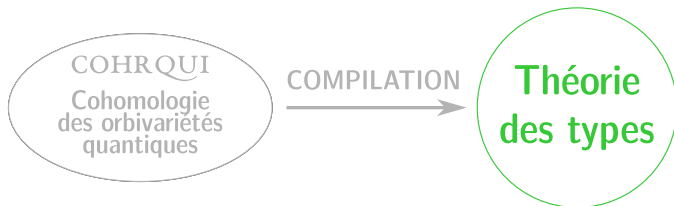
Solution 2 : changer complètement la traduction

préfaisceaux ensemblistes $\Rightarrow$ types préfascistes

- inspirés par la théorie de la programmation
- préservent la conversion
- vraiment pas des préfaisceaux
- (remarque sur les faisceaux *intensionnellement* laissée vide)

Il faut revisiter le folklore en profondeur !

2. CIC citron ou CIC orange ?



Pour s'asseoir sur les fondations, elles doivent être confortables

- Une théorie des types la plus expressive possible
- ... tout en maintenant de bonnes propriétés (cohérence, canonicité, décidabilité...)

Pour s'asseoir sur les fondations, elles doivent être confortables

- Une théorie des types la plus expressive possible
- ... tout en maintenant de bonnes propriétés (cohérence, canonicité, décidabilité...)

Deux points clairement en tension !

Pour s'asseoir sur les fondations, elles doivent être confortables

- Une théorie des types la plus expressive possible
- ... tout en maintenant de bonnes propriétés (cohérence, canonicité, décidabilité...)

Deux points clairement en tension !

- L'exploration de cet espace est le travail des informaticiens
- ... mais il doit être informé des besoins des sources
- Les fondations doivent être évaluées à l'aune de leur utilisabilité

Pour s'asseoir sur les fondations, elles doivent être confortables

- Une théorie des types la plus expressive possible
- ... tout en maintenant de bonnes propriétés (cohérence, canonicité, décidabilité...)

Deux points clairement en tension !

- L'exploration de cet espace est le travail des informaticiens
- ... mais il doit être informé des besoins des sources
- Les fondations doivent être évaluées à l'aune de leur utilisabilité

Au hasard : **Univalence** ou **UIP** ?

Et pourquoi pas les deux ?

Et pourquoi pas les deux ?

- Surtout ne pas répondre à la question
- Une extension conceptuellement triviale qui passe à l'échelle
- Le multivers à portée de clavier

Et pourquoi pas les deux ?

- Surtout ne pas répondre à la question
- Une extension conceptuellement triviale qui passe à l'échelle
- Le multivers à portée de clavier

via **le polymorphisme de sortes**

- Une addition récente à Rocq
- Compatibilité arrière par construction
- Gain d'expressivité linguistique, conservatif logiquement

Tu erreras désormais dans un monde inconnu

Rocq a déjà une notion d'univers parallèles : les sortes

Tu erreras désormais dans un monde inconnu

Rocq a déjà une notion d'univers parallèles : les sortes

- Prop / Type / SProp
- Une distinction à l'origine introduite pour l'extraction
- Chaque univers capture une logique différente

Tu erreras désormais dans un monde inconnu

Rocq a déjà une notion d'univers parallèles : les sortes

- Prop / Type / SProp
- Une distinction à l'origine introduite pour l'extraction
- Chaque univers capture une logique différente

On peut rajouter de nouvelles sortes

- Par exemple des types fibrants, des types stricts...
- Rocq est agnostique à priori sur le contenu de ces sortes

Tu erreras désormais dans un monde inconnu

Rocq a déjà une notion d'univers parallèles : les sortes

- Prop / Type / SProp
- Une distinction à l'origine introduite pour l'extraction
- Chaque univers capture une logique différente

On peut rajouter de nouvelles sortes

- Par exemple des types fibrants, des types stricts...
- Rocq est agnostique à priori sur le contenu de ces sortes

Nihil novis

cf. la théorie des types à deux niveaux

La notion de variable

La notion de variable

- On peut maintenant quantifier sur les sortes
- Permet de partager le code
- Une théorie polymorphe est valide dans tous les univers

La notion de variable

- On peut maintenant quantifier sur les sortes
- Permet de partager le code
- Une théorie polymorphe est valide dans tous les univers

La question difficile : l'interaction entre sortes

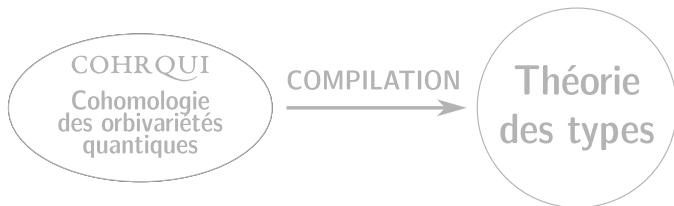
La notion de variable

- On peut maintenant quantifier sur les sortes
- Permet de partager le code
- Une théorie polymorphe est valide dans tous les univers

La question difficile : l'interaction entre sortes

Ça bouge encore dans les fondations !

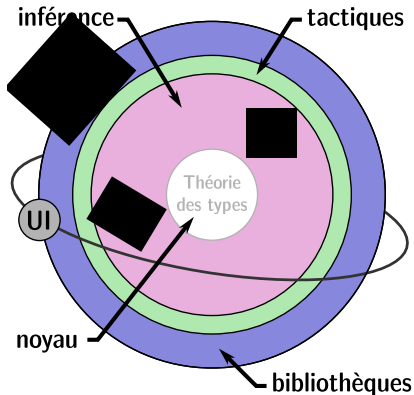
Et les assistants à la preuve dans tout ça ?



J'ai menti

Une vision très biaisée !

Une vision très biaisée !



HERE BE DRAGONS

Un assistant à la preuve, c'est bien plus que ça

Encore une fois, les informaticiens ont déjà frappé

Encore une fois, les informaticiens ont déjà frappé

The IRIS framework

- Un assistant à la preuve *au-dessus* de Rocq
- Bas-niveau : logique de séparation via des préfaisceaux
- Haut-niveau : un *mod* de preuve pour votre jeu vidéo favori

Les couches hautes sont entièrement redéfinies

```
=====
"Hinv" : inv N (lock_inv y l R)
-----□
"H0" :  $\forall b : \text{bool}, (\text{if } b \text{ then locked } y * R \text{ else True}) \multimap \Phi \# b$ 
"Hl" :  $\triangleright l \mapsto \# \text{true}$ 
"HR" :  $\triangleright \text{True}$ 
-----*
WP CmpXchg #l #false #true @ T  $\wedge \uparrow N \{ \{ v, l = \{ T \wedge \uparrow N \} \Rightarrow \triangleright \text{lock\_inv } y \text{ l R} * \text{WP Snd } v \{ \{ v, \Phi v \} \} \}$ 

goal 2 (ID 1116) is:
"Hinv" : inv N (lock_inv y l R)
-----□
"H0" :  $\forall b : \text{bool}, (\text{if } b \text{ then locked } y * R \text{ else True}) \multimap \Phi \# b$ 
"Hl" :  $\triangleright l \mapsto \# \text{false}$ 
"HR" :  $\triangleright (\text{token } y * R)$ 
-----*
WP CmpXchg #l #false #true @ T  $\wedge \uparrow N \{ \{ v, l = \{ T \wedge \uparrow N \} \Rightarrow \triangleright \text{lock\_inv } y \text{ l R} * \text{WP Snd } v \{ \{ v, \Phi v \} \} \}$ 
```

- Des notations de partout
- Un langage de tactiques ad-hoc
- Même le mode de preuve est un encodage !

Songe d'une nuit de MLTT

Je rêve d'une myriade de sur-assistants à la preuve



Songe d'une nuit de MLTT

Je rêve d'une myriade de sur-assistants à la preuve



À votre tour de faire fleurir les *mods*

Conclusion

- Le multilinguisme c'est la norme, en maths comme en info
- Il doit devenir systématique en théorie des types
- Un travail au long cours
- ... à l'interface entre mathématiques et informatique

Merci de votre attention.