Effects, Substitution and Induction

An Explosive Ménage à Trois

Pierre-Marie Pédrot, Nicolas Tabareau

INRIA

TYPES 2019

13th June 2019

It's Time to CIC Ass

CIC, the Calculus of Inductive Constructions.

It's Time to CIC Ass

CIC, the Calculus of Inductive Constructions.

CIC, a very fancy **intuitionistic** **logical system**.

- Not just higher-order logic, not just first-order logic
- First class notion of computation and crazy inductive types

It's Time to CIC Ass

CIC, the Calculus of Inductive Constructions.

CIC, a very fancy **intuitionistic** **logical system**.

- Not just higher-order logic, not just first-order logic
- First class notion of computation and crazy inductive types

CIC, a very powerful **functional** **programming language**.

- Finest types to describe your programs
- No clear phase separation between runtime and compile time

It's Time to CIC Ass

CIC, the Calculus of Inductive Constructions.

CIC, a very fancy **intuitionistic** **logical system**.

- Not just higher-order logic, not just first-order logic
- First class notion of computation and crazy inductive types

CIC, a very powerful **functional** **programming language**.

- Finest types to describe your programs
- No clear phase separation between runtime and compile time

The Pinnacle of the Curry-Howard correspondence

CIC, the Calculus of Inductive Constructions.

CIC, a very fancy **intuitionistic** **logical system**.

- Not just higher-order logic, not just first-order logic
- First class notion of computation and crazy inductive types

CIC, a very powerful **functional** **programming language**.

- Finest types to describe your programs
- No clear phase separation between runtime and compile time

The Pinnacle of the Curry-Howard correspondence

The Most Important Issue of Them All

Yet CIC suffers from a **fundamental** flaw.

The Most Important Issue of Them All

Yet CIC suffers from a **fundamental** flaw.

You want to show the wonders of Coq to a fellow programmer

You fire your favourite IDE

… and you're asked the DREADFUL question.

The Most Important Issue of Them All

Yet CIC suffers from a **fundamental** flaw.

You want to show the wonders of Coq to a fellow programmer

You fire your favourite IDE

... and you're asked the DREADFUL question.

COULD YOU WRITE A HELLO WORLD?

The Most Important Issue of Them All, Bis

Intuitionistic Logic $\Leftrightarrow$ **Functional** Programming

The Most Important Issue of Them All, Bis

Intuitionistic Logic $\Leftrightarrow$ **Functional** Programming

Thus, the same problem for mathematically inclined users.

The Most Important Issue of Them All, Bis

Intuitionistic Logic $\Leftrightarrow$ **Functional** Programming

Thus, the same problem for mathematically inclined users.

HOW DO I REASON CLASSICALLY?

The Most Important Issue of Them All, Bis

Intuitionistic Logic ⇔ **Functional** Programming

Thus, the same problem for mathematically inclined users.

HOW DO I REASON CLASSICALLY?

Thesis

We want a type theory with effects!

Thesis

We want a type theory with effects!

To program more!

Non-termination

Exceptions

State...

To prove more!

Classical logic

Univalence

Choice...

Something is Rotten in the State of Type Theory

Classical logic does not play well with type theory.

Barthe and Uustalu: CPS cannot interpret dependent elimination

Herbelin's paradox: CIC + `callcc` is unsound!

Something is Rotten in the State of Type Theory

Classical logic does not play well with type theory.

Barthe and Uustalu: CPS cannot interpret dependent elimination

Herbelin's paradox: CIC + `callcc` is unsound!

We have been working on effectful type theories

We are not the only ones, but our specialty:

Something is Rotten in the State of Type Theory

Classical logic does not play well with type theory.

Barthe and Uustalu: CPS cannot interpret dependent elimination

Herbelin's paradox: CIC + `callcc` is unsound!

We have been working on effectful type theories

We are not the only ones, but our specialty:

We justify them through program translations into CIC itself.

Forcing, reader monad, exceptions, free algebraic...

Something is Rotten in the State of Type Theory

Classical logic does not play well with type theory.

Barthe and Uustalu: CPS cannot interpret dependent elimination

Herbelin's paradox: CIC + `callcc` is unsound!

We have been working on effectful type theories

We are not the only ones, but our specialty:

We justify them through program translations into CIC itself.

Forcing, reader monad, exceptions, free algebraic...

Effectful theories are always half-broken

dependent elimination has to be restricted (BTT)
or consistency forsaken, or worse

The Important Bit

This is not a coincidence!

Herbelin / Barthe-Uustalu results are instances of a generic phenomenon!

The Important Bit

This is not a coincidence!

Herbelin / Barthe-Uustalu results are instances of a generic phenomenon!

Also, this is kind of folklore.

... but I don't recall reading it formally anywhere.

Effects, Effects Everywhere!

Definition

A type theory has *observable effects* if there is a closed term $t : \mathbb{B}$ that is **not observationally equivalent to a value**, i.e. there is a context $C[\cdot]$ s.t.

$$C[\texttt{true}] \equiv \texttt{true} \quad \text{and} \quad C[\texttt{false}] \equiv \texttt{true} \quad \text{but} \quad C[t] \equiv \texttt{false}$$

This happens for many kind of effects, including continuations.

Effects, Effects Everywhere!

Definition

A type theory has *observable effects* if there is a closed term $t : \mathbb{B}$ that is **not observationally equivalent to a value**, i.e. there is a context $C[\cdot]$ s.t.

$$C[\texttt{true}] \equiv \texttt{true} \quad \text{and} \quad C[\texttt{false}] \equiv \texttt{true} \quad \text{but} \quad C[t] \equiv \texttt{false}$$

This happens for many kind of effects, including continuations.

Such terms are typically called **non-standard** booleans.

e.g. a function $\texttt{is\_empty} : \Pi A.\, A \rightarrow \mathbb{B}$ deciding inhabitation of a type.

A Tension Build-up

Definition

A type theory enjoys *substitution* if the following rule is derivable.

$$\frac{\Gamma, x : X \vdash \bullet : A \qquad \Gamma \vdash t : X}{\Gamma \vdash \bullet : A\{x := t\}}$$

A Tension Build-up

Definition

A type theory enjoys *substitution* if the following rule is derivable.

$$\frac{\Gamma, x : X \vdash \bullet : A \qquad \Gamma \vdash t : X}{\Gamma \vdash \bullet : A\{x := t\}}$$

Substitution is usually taken for granted

... hint: this is a bias

Almost There

Definition

A type theory enjoys *dependent elimination* on booleans if we have:

$$\frac{\Gamma, b : \mathbb{B} \vdash P : \square \qquad \Gamma \vdash \bullet : P\{b := \texttt{true}\} \qquad \Gamma \vdash \bullet : P\{b := \texttt{false}\}}{\Gamma, b : \mathbb{B} \vdash \bullet : P}$$

Almost There

Definition

A type theory enjoys *dependent elimination* on booleans if we have:

$$\frac{\Gamma, b : \mathbb{B} \vdash P : \square \qquad \Gamma \vdash \bullet : P\{b := \texttt{true}\} \qquad \Gamma \vdash \bullet : P\{b := \texttt{false}\}}{\Gamma, b : \mathbb{B} \vdash \bullet : P}$$

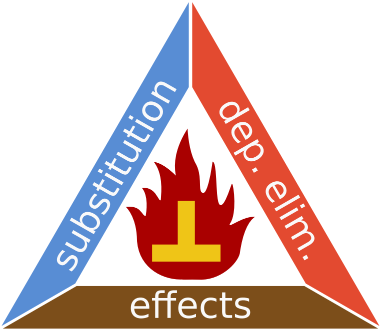
The landmark of dependent type theory, used to encode induction!

Absence of dependent elimination smells of trivial theories.

« Ciel, mon mari ! »

Sounds like desirable features, right?

Sounds like desirable features, right?



Theorem (Explosive Ménage à Trois a.k.a. Fire Triangle)

Effects + substitution + dep. elimination $\vdash$ *logically inconsistent.*

There Is No Free Lunch

The proof is actually straightforward.

Proof.

If C distinguishes boolean values from an effectful term M, prove by dependent elimination $\Pi(b : \mathbb{B}).\ C[b] = \texttt{false}$, apply to M and derive $\texttt{true} = \texttt{false}$. $\square$

There Is No Free Lunch

The proof is actually straightforward.

Proof.

If C distinguishes boolean values from an effectful term M, prove by dependent elimination $\Pi(b : \mathbb{B}).\ C[b] = \texttt{false}$, apply to M and derive $\texttt{true} = \texttt{false}$. □

We essentially retrofitted the definition of effects to make it work.

There Is No Free Lunch

The proof is actually straightforward.

Proof.

If C distinguishes boolean values from an effectful term M, prove by dependent elimination $\Pi(b : \mathbb{B}).\ C[b] = \texttt{false}$, apply to M and derive $\texttt{true} = \texttt{false}$. $\square$

We essentially retrofitted the definition of effects to make it work.

‘But most effects are also observables effects!

So it’s not cheating either.

There Is No Free Lunch

The proof is actually straightforward.

Proof.

If C distinguishes boolean values from an effectful term M, prove by dependent elimination $\Pi(b : \mathbb{B}).\ C[b] = \texttt{false}$, apply to M and derive $\texttt{true} = \texttt{false}$. $\square$

We essentially retrofitted the definition of effects to make it work.

‘But most effects are also observables effects!

So it’s not cheating either.

And now for a high-level overview of the problem and solutions

It is not a Bug, it is a Feature™

Dependency entails one major difference with usual type systems.

It is not a Bug, it is a Feature™

Dependency entails one major difference with usual type systems.

Meet conversion:

$$\frac{A \equiv_\beta B \qquad \Gamma \vdash M : B}{\Gamma \vdash M : A}$$

It is not a Bug, it is a Feature™

Dependency entails one major difference with usual type systems.

Meet conversion:

$$\frac{A \equiv_\beta B \qquad \Gamma \vdash M : B}{\Gamma \vdash M : A}$$

Bad news 1

Typing rules embed the dynamics of programs!

It is not a Bug, it is a Feature™

Dependency entails one major difference with usual type systems.

Meet conversion:

$$\frac{A \equiv_\beta B \qquad \Gamma \vdash M : B}{\Gamma \vdash M : A}$$

Bad news 1

Typing rules embed the dynamics of programs!

Combine that with this other observation and we're in trouble.

Bad news 2

Effects make reduction strategies relevant.

Call-by-name vs. Call-by-value

Call-by-name vs. Call-by-value

Call-by-name: **functions** well-behaved vs. **inductives** ill-behaved

Call-by-value: **inductives** well-behaved vs. **functions** ill-behaved

Call-by-name vs. Call-by-value

Call-by-name: **functions** well-behaved vs. **inductives** ill-behaved
Call-by-value: **inductives** well-behaved vs. **functions** ill-behaved

In **call-by-name** + effects:

$$(\lambda x.\, M)\; N \equiv M\{x := N\} \quad \rightsquigarrow \quad \textbf{arbitrary substitution}$$
$$(\lambda b : \texttt{bool}.\, M)\; \mathbf{fail} \quad \rightsquigarrow \quad \textbf{non-standard booleans}$$

Substitution is a feature of call-by-name

Call-by-name vs. Call-by-value

Call-by-name: **functions** well-behaved vs. **inductives** ill-behaved
Call-by-value: **inductives** well-behaved vs. **functions** ill-behaved

In **call-by-name** + effects:

$$(\lambda x.\, M)\ N \equiv M\{x := N\} \quad \rightsquigarrow \quad \textbf{arbitrary substitution}$$
$$(\lambda b : \texttt{bool}.\, M)\ \textbf{fail} \quad \rightsquigarrow \quad \textbf{non-standard booleans}$$

Substitution is a feature of call-by-name

In **call-by-value** + effects:

$$(\lambda x.\, M)\ V \equiv M\{x := V\} \quad \rightsquigarrow \quad \textbf{substitute only values}$$
$$(\lambda b : \mathbb{B}.\, M)\ N \equiv (\lambda b : \mathbb{B}.\, M)\ V \quad \rightsquigarrow \quad \textbf{boolean values are booleans}$$

Dependent elimination is a feature of call-by-value

Impossible is not French

Three knobs $\Rightarrow$ **Four** solutions

Impossible is not French

Three knobs ⇒ **Four** solutions

- **Down with effects**: CBN and CBV reconcile

This is good ol' CIC, Keep Calm and Carry on.

Impossible is not French

Three knobs $\Rightarrow$ **Four** solutions

- **Down with effects**: CBN and CBV reconcile

This is good ol' CIC, Keep Calm and Carry on.

- **Go CBN** and restrict dependent elimination: Baclofen Type Theory

$$\texttt{if}\ M\ \texttt{then}\ N_1\ \texttt{else}\ N_2 : \texttt{if}\ M\ \texttt{then}\ P_1\ \texttt{else}\ P_2$$

Impossible is not French

Three knobs $\Rightarrow$ **Four** solutions

- **Down with effects**: CBN and CBV reconcile

This is good ol' CIC, Keep Calm and Carry on.

- **Go CBN** and restrict dependent elimination: Baclofen Type Theory

$\texttt{if } M \texttt{ then } N_1 \texttt{ else } N_2 : \texttt{if } M \texttt{ then } P_1 \texttt{ else } P_2$

- **CBV rules**, respect values, and dump substitution: one weird trick

The least conservative approach

Impossible is not French

Three knobs $\Rightarrow$ **Four** solutions

- **Down with effects**: CBN and CBV reconcile

This is good ol' CIC, KEEP CALM AND CARRY ON.

- **Go CBN** and restrict dependent elimination: Baclofen Type Theory

$\texttt{if}\ M\ \texttt{then}\ N_1\ \texttt{else}\ N_2 : \texttt{if}\ M\ \texttt{then}\ P_1\ \texttt{else}\ P_2$

- **CBV rules**, respect values, and dump substitution: one weird trick

The least conservative approach

- Who cares about consistency? **I want all!**

A paradigm shift: from type theory to dependent languages, e.g. ExTT

A Generic Workaround

We have a proposal for a generalization of CBPV to factor both.

A Generic Workaround

We have a proposal for a generalization of CBPV to factor both.

$$\partial\mathsf{CBPV}$$

(We had to pick a fancy name.)

A Generic Workaround

We have a proposal for a generalization of CBPV to factor both.

∂CBPV

(We had to pick a fancy name.)

The main novelties: two for the price of one

- Not one, but **two** parallel hierarchies of universes: $\square_v$ vs. $\square_c$!
- Not one, but **two** let-bindings!

$$\frac{\Gamma \vdash t : F\ A \qquad \Gamma \vdash X : \square_c \qquad \Gamma, x : A \vdash u : X}{\Gamma \vdash \texttt{let}\ x := t\ \texttt{in}\ u : X}$$

$$\frac{\Gamma \vdash t : F\ A \qquad \Gamma, x : A \vdash X : \square_c \qquad \Gamma, x : A \vdash u : X}{\Gamma \vdash \texttt{dlet}\ x := t\ \texttt{in}\ u : \texttt{let}\ x := t\ \texttt{in}\ X}$$

- Justified by all of our syntactic models so far (and we have quite a few)

Many More

This was a very high-level talk

Many things I did not discuss here!

- A good notion of purity: thunkability vs. linearity
- Complex ∂CBPV encodings
- Presheaves as observationally pure terms of an impure CBV theory

`http://pédrot.fr/articles/dcbpv.pdf`

Conclusion

- Effects and dependent types: choose your side.
  $\rightsquigarrow$ Purity, CBN, CBV, Michael Bay?
- Even inconsistent theories have practical interest.

Thanks for your attention.