

# The $\lambda$ -calculus, from minimal to classical logic

Webpage of the course

**Davide Barbarossa**

[db2437@bath.ac.uk](mailto:db2437@bath.ac.uk)

Dept of Computer Science



**Giulio Guerrieri**

[g.guerrieri@sussex.ac.uk](mailto:g.guerrieri@sussex.ac.uk)

Dept of Computer Science



*ESSLLI Summer School, Bochum (Germany)*

*28/07/2025 – 01/08/2025*

The  $\lambda$ -calculus,  
from minimal to classical logic

Lecture 5:  
Krivine's approach to classical logic

Read the notes: they are full of details, proofs, explanations, exercises, bibliography!

Davide Barbarossa

[db2437@bath.ac.uk](mailto:db2437@bath.ac.uk)

Dept of Computer Science



*ESSLLI Summer School, Bochum (Germany)*

*01/08/2025*

## Previously...

- You have seen **minimal logic**
- You have seen that it corresponds to the **simply-typed  $\lambda$ -calculus**
- In the sense that **formulas = types** *and* **cut-elimination =  $\beta$ -reduction**
- **Computational understanding of logic:** proof  
→ program



## Previously...

Curry-Howard formalises the computational understanding (BHK) of logic in the *strongest* sense:

a proof  $x : A \vdash M : B$  is a (typed) program that computes the function

## Previously...

Curry-Howard formalises the computational understanding (BHK) of logic in the *strongest* sense:

a proof  $x : A \vdash M : B$  is a (typed) program that computes the function

$$N \in \text{Proofs}(A) \quad \mapsto \quad \text{nf}_\beta(M\{x := N\}) \in \text{Proofs}(B)$$

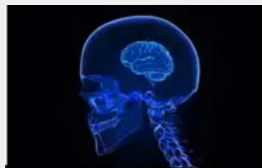
## Previously...

Curry-Howard formalises the computational understanding (BHK) of logic in the *strongest* sense:

a proof  $x : A \vdash M : B$  is a (typed) program that computes the function

$$N \in \text{Proofs}(A) \quad \mapsto \quad \text{nf}_\beta(M\{x := N\}) \in \text{Proofs}(B)$$

Yesterday: *minimal* logic



## Previously...

Curry-Howard formalises the computational understanding (BHK) of logic in the *strongest* sense:

a proof  $x : A \vdash M : B$  is a (typed) program that computes the function

$$M \in \text{Proofs}(A) \quad \mapsto \quad \text{nf}_\beta(M\{x := N\}) \in \text{Proofs}(B)$$

...it actually works for richer and richer *intuitionistic* logics (e.g. system F/2nd-order  $\lambda$ -calculus, MLTT's etc)

## Previously...

Curry-Howard formalises the computational understanding (BHK) of logic in the *strongest* sense:

a proof  $x : A \vdash M : B$  is a (typed) program that computes the function

$$M \in \text{Proofs}(A) \quad \mapsto \quad \text{nf}_\beta(M\{x := N\}) \in \text{Proofs}(B)$$

...it actually works for richer and richer *intuitionistic* logics (e.g. system F/2nd-order  $\lambda$ -calculus, MLTT's etc)





## Previously...

Curry-Howard formalises the computational understanding (BHK) of logic in the *strongest* sense:

a proof  $x : A \vdash M : B$  is a (typed) program that computes the function

$$M \in \text{Proofs}(A) \quad \mapsto \quad \text{nf}_\beta(M\{x := N\}) \in \text{Proofs}(B)$$

...it actually works for richer and richer *intuitionistic* logics (e.g. system F/2nd-order  $\lambda$ -calculus, MLTT's etc)

What about *classical* logic?

## Previously...

Curry-Howard formalises the computational understanding (BHK) of logic in the *strongest* sense:

a proof  $x : A \vdash M : B$  is a (typed) program that computes the function

$$M \in \text{Proofs}(A) \quad \mapsto \quad \text{nf}_\beta(M\{x := N\}) \in \text{Proofs}(B)$$

...it actually works for richer and richer *intuitionistic* logics (e.g. system F/2nd-order  $\lambda$ -calculus, MLTT's etc)

## What about *classical* logic?

Taking the above *literally* fails (“Joyal’s lemma” in category theory, “Lafont’s pairs” in sequent calculus,...) ... which we are not going to see

## Previously...

Curry-Howard formalises the computational understanding (BHK) of logic in the *strongest* sense:

a proof  $x : A \vdash M : B$  is a (typed) program that computes the function

$$M \in \text{Proofs}(A) \quad \mapsto \quad \text{nf}_\beta(M\{x := N\}) \in \text{Proofs}(B)$$

...it actually works for richer and richer *intuitionistic* logics (e.g. system F/2nd-order  $\lambda$ -calculus, MLTT's etc)

## What about *classical* logic?

Classical logic still has a computational content indeed!

$$\begin{array}{ccc} \text{type} & \rightsquigarrow & \text{realiser} \\ \text{purely functional} & \rightsquigarrow & \text{impure functional} \end{array}$$

## Example

Classical realisability,  $\neg\neg+$ Dialectica,  $\lambda\mu$ -calculus,  $\bar{\lambda}\mu\tilde{\mu}$ -calculus,...

## Previously...

Curry-Howard formalises the computational understanding (BHK) of logic in the *strongest* sense:

a proof  $x : A \vdash M : B$  is a (typed) program that computes the function

$$N \in \text{Proofs}(A) \quad \mapsto \quad \text{nf}_\beta(M\{x := N\}) \in \text{Proofs}(B)$$

...it actually works for richer and richer *intuitionistic* logics (e.g. system F/2nd-order  $\lambda$ -calculus, MLTT's etc)



What about *classical* logic?

Classical logic still has a computational content indeed!

type	$\rightsquigarrow$	realiser
purely functional	$\rightsquigarrow$	impure functional

## Example

Classical realisability,  $\neg\neg +$  Dialectica,  $\lambda\mu$ -calculus,  $\bar{\lambda}\mu\tilde{\mu}$ -calculus,...

- 1 2nd order classical logic
- 2 Operational semantics of  $\lambda$ -calculus + `callcc`
- 3 Realisability and its adequacy to provability
- 4 Summary, Exercises, Bibliography

- 1 2nd order classical logic
- 2 Operational semantics of  $\lambda$ -calculus + callcc
- 3 Realisability and its adequacy to provability
- 4 Summary, Exercises, Bibliography

## 2nd order classical logic

Formulas:

$A ::= X \mid A \rightarrow A$

Proofs:

$$\frac{\frac{\underline{A}, \quad B \vdash \quad B}{\underline{A} \vdash \quad B \rightarrow C} \quad \underline{A} \vdash \quad B}{\underline{A} \vdash \quad C}$$

$$\frac{\underline{A}, \quad B \vdash \quad C}{\underline{A} \vdash \quad B \rightarrow C}$$

## 2nd order classical logic

Formulas:

$$A ::= X \mid A \rightarrow A$$

Proofs:

$$\overline{\underline{x} : \underline{A}, y : B \vdash y : B}$$

$$\frac{\underline{x} : \underline{A} \vdash M : B \rightarrow C \quad \underline{x} : \underline{A} \vdash N : B}{\underline{x} : \underline{A} \vdash MN : C}$$

$$\frac{\underline{x} : \underline{A}, y : B \vdash M : C}{\underline{x} : \underline{A} \vdash \lambda y. M : B \rightarrow C}$$

Proof-like terms:

$$M ::= x \mid MN \mid \lambda x. M$$



## 2nd order classical logic

Arithmetic expressions:

$$e ::= n \mid a \mid f(e, \dots, e) \quad (\text{for } n \in \mathbb{N}, a \in \mathcal{V}_1, f \in \mathcal{S}_k)$$

Formulas:

$$A ::= X(e, \dots, e) \mid A \rightarrow A \mid \forall c. A \mid \forall^k X. A \quad (\text{for } X \in \mathcal{V}_2^k)$$

Proofs:

$$\overline{\underline{x} : \underline{A}, y : B \vdash y : B}$$

$$\frac{\underline{x} : \underline{A} \vdash M : B \rightarrow C \quad \underline{x} : \underline{A} \vdash N : B}{\underline{x} : \underline{A} \vdash MN : C}$$

$$\frac{\underline{x} : \underline{A}, y : B \vdash M : C}{\underline{x} : \underline{A} \vdash \lambda y. M : B \rightarrow C}$$

Proof-like terms:

$$M ::= x \mid MN \mid \lambda x. M$$

## 2nd order classical logic

Arithmetic expressions:

$$e ::= n \mid a \mid f(e, \dots, e) \quad (\text{for } n \in \mathbb{N}, a \in \mathcal{V}_1, f \in \mathcal{S}_k)$$

Formulas:

$$A ::= X(e, \dots, e) \mid A \rightarrow A \mid \forall c. A \mid \forall^k X. A \quad (\text{for } X \in \mathcal{V}_2^k)$$

Proofs:

$$\overline{\underline{x} : \underline{A}, y : B \vdash y : B}$$

$$\frac{\underline{x} : \underline{A} \vdash M : B \rightarrow C \quad \underline{x} : \underline{A} \vdash N : B}{\underline{x} : \underline{A} \vdash MN : C}$$

$$\frac{\underline{x} : \underline{A}, y : B \vdash M : C}{\underline{x} : \underline{A} \vdash \lambda y. M : B \rightarrow C}$$

$$\frac{\underline{A} \vdash B}{\underline{A} \vdash \forall c. B}$$

$$\frac{\underline{A} \vdash B}{\underline{A} \vdash \forall^k Y. B}$$

$$\frac{\underline{A} \vdash \forall c. B}{\underline{A} \vdash B\{c := e\}}$$

$$\frac{\underline{A} \vdash \forall^k Y. B}{\underline{A} \vdash B\{Y := \langle C, c_1, \dots, c_k \rangle\}}$$

Proof-like terms:

$$M ::= x \mid MN \mid \lambda x. M$$

## 2nd order classical logic

Arithmetic expressions:

$$e ::= n \mid a \mid f(e, \dots, e) \quad (\text{for } n \in \mathbb{N}, a \in \mathcal{V}_1, f \in \mathcal{S}_k)$$

Formulas:

$$A ::= X(e, \dots, e) \mid A \rightarrow A \mid \forall c. A \mid \forall^k X. A \quad (\text{for } X \in \mathcal{V}_2^k)$$

Proofs:

$$\overline{\underline{x} : \underline{A}, y : B \vdash y : B}$$

$$\frac{\underline{x} : \underline{A} \vdash M : B \rightarrow C \quad \underline{x} : \underline{A} \vdash N : B}{\underline{x} : \underline{A} \vdash MN : C}$$

$$\frac{\underline{x} : \underline{A}, y : B \vdash M : C}{\underline{x} : \underline{A} \vdash \lambda y. M : B \rightarrow C}$$

$$\frac{\underline{x} : \underline{A} \vdash M : B}{\underline{x} : \underline{A} \vdash M : \forall c. B}$$

$$\frac{\underline{x} : \underline{A} \vdash M : B}{\underline{x} : \underline{A} \vdash M : \forall^k Y. B}$$

$$\frac{\underline{x} : \underline{A} \vdash M : \forall c. B}{\underline{x} : \underline{A} \vdash M : B\{c := e\}}$$

$$\frac{\underline{x} : \underline{A} \vdash M : \forall^k Y. B}{\underline{x} : \underline{A} \vdash M : B\{Y := \langle C, c_1, \dots, c_k \rangle\}}$$

Proof-like terms:

$$M ::= x \mid MN \mid \lambda x. M$$

## 2nd order classical logic

Arithmetic expressions:

$$e ::= n \mid a \mid f(e, \dots, e) \quad (\text{for } n \in \mathbb{N}, a \in \mathcal{V}_1, f \in \mathcal{S}_k)$$

Formulas:

$$A ::= X(e, \dots, e) \mid A \rightarrow A \mid \forall c. A \mid \forall^k X. A \quad (\text{for } X \in \mathcal{V}_2^k)$$

Proofs:

$$\begin{array}{c} \frac{}{\underline{x} : \underline{A}, y : B \vdash y : B} \qquad \frac{}{\underline{A} \vdash ((B \rightarrow C) \rightarrow B) \rightarrow B} \\[10pt] \frac{\underline{x} : \underline{A} \vdash M : B \rightarrow C \quad \underline{x} : \underline{A} \vdash N : B}{\underline{x} : \underline{A} \vdash MN : C} \qquad \frac{\underline{x} : \underline{A}, y : B \vdash M : C}{\underline{x} : \underline{A} \vdash \lambda y. M : B \rightarrow C} \\[10pt] \frac{\underline{x} : \underline{A} \vdash M : B}{\underline{x} : \underline{A} \vdash M : \forall c. B} \qquad \frac{\underline{x} : \underline{A} \vdash M : B}{\underline{x} : \underline{A} \vdash M : \forall^k Y. B} \\[10pt] \frac{\underline{x} : \underline{A} \vdash M : \forall c. B}{\underline{x} : \underline{A} \vdash M : B\{c := e\}} \qquad \frac{\underline{x} : \underline{A} \vdash M : \forall^k Y. B}{\underline{x} : \underline{A} \vdash M : B\{Y := \langle C, c_1, \dots, c_k \rangle\}} \end{array}$$

Proof-like terms:

$$M ::= x \mid MN \mid \lambda x. M$$

## 2nd order classical logic

Arithmetic expressions:

$$e ::= n \mid a \mid f(e, \dots, e) \quad (\text{for } n \in \mathbb{N}, a \in \mathcal{V}_1, f \in \mathcal{S}_k)$$

Formulas:

$$A ::= X(e, \dots, e) \mid A \rightarrow A \mid \forall c. A \mid \forall^k X. A \quad (\text{for } X \in \mathcal{V}_2^k)$$

Proofs:

$$\frac{}{\underline{x} : \underline{A}, y : B \vdash y : B}$$

$$\frac{}{\underline{x} : \underline{A} \vdash \text{callcc} : ((B \rightarrow C) \rightarrow B) \rightarrow B}$$

$$\frac{\underline{x} : \underline{A} \vdash M : B \rightarrow C \quad \underline{x} : \underline{A} \vdash N : B}{\underline{x} : \underline{A} \vdash MN : C}$$

$$\frac{\underline{x} : \underline{A}, y : B \vdash M : C}{\underline{x} : \underline{A} \vdash \lambda y. M : B \rightarrow C}$$

$$\frac{\underline{x} : \underline{A} \vdash M : B}{\underline{x} : \underline{A} \vdash M : \forall c. B}$$

$$\frac{\underline{x} : \underline{A} \vdash M : B}{\underline{x} : \underline{A} \vdash M : \forall^k Y. B}$$

$$\frac{\underline{x} : \underline{A} \vdash M : \forall c. B}{\underline{x} : \underline{A} \vdash M : B\{c := e\}}$$

$$\frac{\underline{x} : \underline{A} \vdash M : \forall^k Y. B}{\underline{x} : \underline{A} \vdash M : B\{Y := \langle C, c_1, \dots, c_k \rangle\}}$$

Proof-like terms:

$$M ::= x \mid MN \mid \lambda x. M \mid \text{callcc}$$

# Operational semantics of $\lambda$ -calculus + callcc

- 1 2nd order classical logic
- 2 Operational semantics of  $\lambda$ -calculus + callcc
- 3 Realisability and its adequacy to provability
- 4 Summary, Exercises, Bibliography

## Operational semantics of $\lambda$ -calculus + callcc

<i>OPS via:</i>	<i>programs run...</i>	<i>E.g.</i>	<i>Logic</i>
$\beta$ -reduction	by themselves	$M \twoheadrightarrow_{\beta} N$	intuitionistic
Abstract Machine	by interaction with execution stacks	$M \star \pi \rightarrow N \star \rho$	classical

# Operational semantics of $\lambda$ -calculus + `callcc`

<i>OPS via:</i>	<i>programs run...</i>	<i>E.g.</i>	<i>Logic</i>
$\beta$ -reduction	by themselves	$M \rightarrow_{\beta} N$	intuitionistic
Abstract Machine	by interaction with execution stacks	$M \star \pi \rightarrow N \star \rho$	classical

*Proof-Like terms*    $P ::= x \mid \lambda x. P \mid P P \mid \text{callcc}$

*Terms*    $M ::= x \mid \lambda x. M \mid M M \mid \text{callcc} \mid k_{\pi}$

*Stacks*    $\pi ::= [] \mid M :: \pi$



# Operational semantics of $\lambda$ -calculus + callcc

<i>OPS via:</i>	<i>programs run...</i>	<i>E.g.</i>	<i>Logic</i>
$\beta$ -reduction	by themselves	$M \rightarrow_{\beta} N$	intuitionistic
Abstract Machine	by interaction with execution stacks	$M \star \pi \rightarrow N \star \rho$	classical

*Proof-Like terms*  $P ::= x \mid \lambda x. P \mid PP \mid \text{callcc}$

*Terms*  $M ::= x \mid \lambda x. M \mid MM \mid \text{callcc} \mid k_{\pi}$

*Stacks*  $\pi ::= [] \mid M :: \pi$

Operational Semantics via a (simplified) KAM:

*Weak Head  $\beta$ -reduction*  $\left\{ \begin{array}{ll} \text{(push)} & MN \star \pi \rightarrow M \star N :: \pi \\ \text{(pop)} & \lambda x. M \star N :: \pi \rightarrow M\{x := N\} \star \pi \end{array} \right.$

*Continuation passing style*  $\left\{ \begin{array}{ll} \text{(save)} & \text{callcc} \star M :: \pi \rightarrow M \star k_{\pi} :: \pi \\ \text{(restore)} & k_{\pi} \star M :: \rho \rightarrow M \star \pi \end{array} \right.$

Arena

Player		Opponent
--------	--	----------

$\mathbf{k}_\pi$	$\star$	$M :: \rho$
------------------	---------	-------------

$\downarrow$

$M$	$\star$	$\pi$
-----	---------	-------

# Operational semantics of $\lambda$ -calculus + callcc

Arena			Weapons	
Player	Opponent		Witnesses	Counterwitnesses
$k_\pi$	$\star$	$M :: \rho$		
	$\downarrow$		$M \in \mathcal{W}(A)$	$\pi \in \mathcal{C}(A)$
$M$	$\star$	$\pi$		

# Operational semantics of $\lambda$ -calculus + callcc

Arena			Weapons	
Player	Opponent		Witnesses	Counterwitnesses
$k_\pi$	$\star$	$M :: \rho$		
	$\downarrow$		$M \in \mathcal{W}(A)$	$\pi \in \mathcal{C}(A)$
$M$	$\star$	$\pi$		
	$\cap$			
	$\perp\!\!\!\perp$			

Arena			Weapons	
Player	Opponent		Witnesses	Counterwitnesses
$k_\pi$	$\star$	$M :: \rho$		
	$\downarrow$		$M \in \mathcal{W}(A)$	
			$k_\pi \in \mathcal{W}(A \rightarrow \perp)$	$\pi \in \mathcal{C}(A)$
$M$	$\star$	$\pi$		

$\Rightarrow \quad k_\pi \in \mathcal{W}(\neg A) \quad \text{for all } \pi \in \mathcal{C}(A)$

Arena

Player		Opponent
--------	--	----------

<code>callcc</code>	$\star$	$M :: \pi$
---------------------	---------	------------

$\downarrow$

$M$	$\star$	$k_\pi :: \pi$
-----	---------	----------------

# Operational semantics of $\lambda$ -calculus + `callcc`

Arena			Weapons	
Player		Opponent	Witnesses	Counterwitnesses
<code>callcc</code>	$\star$	$M :: \pi$		
	$\downarrow$			
$M$	$\star$	$k_\pi :: \pi$	$M \in \mathcal{W}(\neg\neg A)$	$\pi \in \mathcal{C}(A)$

# Operational semantics of $\lambda$ -calculus + callcc

Arena			Weapons	
Player	Opponent		Witnesses	Counterwitnesses
callcc	$\star$	$M :: \pi$		
	$\downarrow$		$M \in \mathcal{W}(\neg A \rightarrow \perp)$	$\pi \in \mathcal{C}(A)$
$M$	$\star$	$k_\pi :: \pi$		
	$\cap$			
	$\perp\!\!\!\perp$			



# Operational semantics of $\lambda$ -calculus + callcc

Arena			Weapons	
Player	Opponent		Witnesses	Counterwitnesses
callcc	★	$M :: \pi$		
	$\downarrow$			
M	★	$k_\pi :: \pi$	$M \in \mathcal{W}(\neg\neg A)$ $\text{callcc} \in \mathcal{W}(\neg\neg A \rightarrow A)$	$\pi \in \mathcal{C}(A)$

$\Rightarrow \quad \text{callcc} \Vdash \neg\neg A \rightarrow A$

# Operational semantics of $\lambda$ -calculus + `callcc`

Arena			Weapons	
Player	Opponent		Witnesses	Counterwitnesses
<code>callcc</code>	$\star$	$M :: \pi$		
	$\downarrow$			
$M$	$\star$	$k_\pi :: \pi$	$M \in \mathcal{W}(\neg\neg A)$ $\text{callcc} \in \mathcal{W}(\neg\neg A \rightarrow A)$	$\pi \in \mathcal{C}(A)$

$$\implies \text{callcc} \Vdash \neg\neg A \rightarrow A$$

Similar argument for

$$\lambda x. \text{callcc } x \Vdash \neg\neg A \rightarrow A \quad \text{and} \quad \lambda x. \text{callcc}(\lambda y. y \ x) \Vdash \neg\neg A \rightarrow A$$

# Realisability and its adequacy to provability

- 1 2nd order classical logic
- 2 Operational semantics of  $\lambda$ -calculus + callcc
- 3 Realisability and its adequacy to provability
- 4 Summary, Exercises, Bibliography

## Definition (Realisability semantics of formulas)

$$C^\perp := \{t \in \mathbb{W} \mid t \perp \pi \text{ for all } \pi \in C\}$$

$$\mathcal{W}(\_) := \mathcal{C}(\_)^\perp$$

	$\mathbb{W}$	$\mathbb{C}$	$\perp \subseteq \mathbb{W} \times \mathbb{C}$	$*$
<i>Tarski</i>	$\{\square\}$	$\{\dagger\}$	$\emptyset$	$\Rightarrow$
<i>Krivine</i>	$\Lambda$	$\Lambda^*$	<i>pole</i>	<i>cons</i>

$$\mathcal{C}(X(e_1, \dots, e_k)) = \llbracket X \rrbracket (\llbracket e_1 \rrbracket, \dots, \llbracket e_k \rrbracket)$$

$$\mathcal{C}(A \rightarrow B) = \mathcal{W}(A) * \mathcal{C}(B)$$

$$\mathcal{C}(\forall c. A) = \bigcup_{m \in \mathbb{N}} \mathcal{C}(A\{c := m\})$$

$$\mathcal{C}(\forall^m Y. A) = \bigcup_{Q: \mathbb{N}^m \rightarrow \mathcal{P}(\mathbb{C})} \mathcal{C}(A\{Y := Q\})$$

Realisability relation:  $\mathbf{M} \Vdash A$  whenever  $\mathbf{M}$  is a closed proof like term in  $\mathcal{W}(A)$

Not only  $\lambda x. \text{callcc}(\lambda y. y x) \Vdash \neg\neg A \rightarrow A$  but even:

$$\vdash \lambda x. \text{callcc}(\lambda y. y x) : \neg\neg A \rightarrow A$$

In fact, typing  $\Rightarrow$  realising. The converse fails: that's precisely what we want!

### Adequacy Theorem

Let

$$\mathbf{x}_1 : A_1, \dots, \mathbf{x}_m : A_m \vdash M : B$$

and fix an interpretation of the (free) 1st and 2nd order variables of  $A_1, \dots, A_m, B$ .

For all closed terms  $N_1, \dots, N_m$ , we have:

$$N_1 \in \mathcal{W}(A_1), \dots, N_m \in \mathcal{W}(A_m) \implies M\{\vec{x} := \vec{N}\} \in \mathcal{W}(B).$$

In fact, typing  $\Rightarrow$  realising. The converse fails: that's precisely what we want!

### Adequacy Theorem

Let

$$\mathbf{x}_1 : A_1, \dots, \mathbf{x}_m : A_m \vdash \mathbf{M} : B$$

and fix an interpretation of the (free) 1st and 2nd order variables of  $A_1, \dots, A_m, B$ .

For all closed terms  $N_1, \dots, N_m$ , we have:

$$N_1 \in \mathcal{W}(A_1), \dots, N_m \in \mathcal{W}(A_m) \implies \mathbf{M}\{\vec{x} := \vec{N}\} \in \mathcal{W}(B).$$

### Corollary

$$\vdash \mathbf{M} : A \implies \mathbf{M} \Vdash A$$

## Realisability and its adequacy to provability

In fact, typing  $\Rightarrow$  realising. The converse fails: that's precisely what we want!

### Adequacy Theorem

Let

$$\mathbf{x}_1 : A_1, \dots, \mathbf{x}_m : A_m \vdash \mathbf{M} : B$$

and fix an interpretation of the (free) 1st and 2nd order variables of  $A_1, \dots, A_m, B$ .

For all closed terms  $N_1, \dots, N_m$ , we have:

$$N_1 \in \mathcal{W}(A_1), \dots, N_m \in \mathcal{W}(A_m) \implies \mathbf{M}\{\vec{\mathbf{x}} := \vec{N}\} \in \mathcal{W}(B).$$

In other words, a proof  $\mathbf{x} : A \vdash \mathbf{M} : B$  defines a proof-term  $\lambda \mathbf{x}. \mathbf{M}$  that computes (for each interpretation of variables and notion of winning process), the function

$$N \in \mathcal{W}(A) \mapsto \mathbf{M}\{\mathbf{x} := N\} \in \mathcal{W}(B)$$

Realisability formalises BHK by extending the literal Curry-Howard one!

## Realisability and its adequacy to provability

In fact, typing  $\Rightarrow$  realising. The converse fails: that's precisely what we want!

### Adequacy Theorem

Let

$$\mathbf{x}_1 : A_1, \dots, \mathbf{x}_m : A_m \vdash \mathbf{M} : B$$

and fix an interpretation of the (free) 1st and 2nd order variables of  $A_1, \dots, A_m, B$ .

For all closed terms  $\mathbf{N}_1, \dots, \mathbf{N}_m$ , we have:

$$\mathbf{N}_1 \in \mathcal{W}(A_1), \dots, \mathbf{N}_m \in \mathcal{W}(A_m) \implies \mathbf{M}\{\vec{\mathbf{x}} := \vec{\mathbf{N}}\} \in \mathcal{W}(B).$$

### Corollary

Let  $\mathcal{T}$  be a theory of PA2 (or ZF/+CH/+C+...). If all axioms  $A$  of  $\mathcal{T}$  are realised by programs  $\mathbf{N}_A \Vdash A$ , then:  $\underline{\mathbf{x}} : \underline{A} \vdash \mathbf{M} : \underline{B} \implies \mathbf{M}\{\underline{\mathbf{x}} := \underline{\mathbf{N}}\} \Vdash \underline{B}$ .

E.g.,  $A$  = countable/dependent axiom of choice, ultrafilter axiom on  $\mathbb{N}$ , Continuum Hypothesis,...



### Moral of the story 1:

(The computational content of) Classical logic is about the interaction with some notion of environment

### Moral of the story 1:

(The computational content of) Classical logic is about the interaction with some notion of environment

### Proof.

True for all (that I know) computational approaches of classical logic: classical realisability, Game Semantics, Linear Logic,  $\lambda\mu$ -calculus,  $\neg\neg$ +Dialectica  $\square$

### Moral of the story 1:

(The computational content of) Classical logic is about the interaction with some notion of environment

### Moral of the story 2:

Now that we disclosed the “hidden” computational content of classical logic, we want to realise axioms, not only theorems!

### Moral of the story 1:

(The computational content of) Classical logic is about the interaction with some notion of environment

### Moral of the story 2:

Now that we disclosed the “hidden” computational content of classical logic, we want to realise axioms, not only theorems!

### Example

The theory of the formulas which admit a realiser is deductively closed. Under mild assumption on the pole  $\perp$ , it is also non-contradictory.

Study its models.

In the case for ZFC, this is stronger than forcing!

# Realisability and its adequacy to provability

## Moral of the story 1:

(The computational content of) Classical logic is about the interaction with some notion of environment

## Moral of the story 2:

Now that we disclosed the “hidden” computational content of classical logic, we want to realise axioms, not only theorems!

## Example

The theory of the formulas which admit a realiser is deductively closed. Under mild assumption on the pole  $\perp$ , it is also non-contradictory.

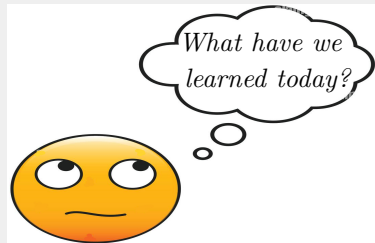
Study its models.

In the case for ZFC, this is stronger than forcing!



- 1 2nd order classical logic
- 2 Operational semantics of  $\lambda$ -calculus + callcc
- 3 Realisability and its adequacy to provability
- 4 Summary, Exercises, Bibliography

- We have seen proof-terms for **classical 2nd order logic**
- We have given them an operational semantics in terms of a **KAM** which manipulates **continuations**
- We have defined the **realisability semantics** by refining Tarski
- We have seen that realisability is **adequate** wrt provability



## Summary, Exercises, Bibliography

- Look at our **notes** on the [webpage of the course](#), there are plenty of **details**, **proofs** and **exercises**.



- Look at our [notes](#) on the [webpage of the course](#), there are plenty of **details**, **proofs** and **exercises**. Here's one

### Exercise

The 2nd order encoding of  $A \vee B$  is:

$A \vee B := \forall^0 X. (A \rightarrow X) \rightarrow (B \rightarrow X) \rightarrow X$ . With that, show (by hand) that:

$$\text{callcc}(\lambda y v h. h(\lambda x. y(\lambda z w. z x))) \Vdash A \vee \neg A.$$

This is actually the proof-term of a derivation of the excluded middle from *Consequentia Mirabilis* (itself an instance of Peirce's law). Do *not* use adequacy though.

## Summary, Exercises, Bibliography

- Look at our **notes** on the [webpage of the course](#), there are plenty of **details**, **proofs** and **exercises**. Here's one

### Exercise

The 2nd order encoding of  $A \vee B$  is:

$A \vee B := \forall^0 X. (A \rightarrow X) \rightarrow (B \rightarrow X) \rightarrow X$ . With that, show (by hand) that:

$$\text{callcc}(\lambda y v h. h(\lambda x. y(\lambda z w. z x))) \Vdash A \vee \neg A.$$

This is actually the proof-term of a derivation of the excluded middle from *Consequentia Mirabilis* (itself an instance of Peirce's law). Do *not* use adequacy though.

- The exercises have **solutions** (but try to do them by yourself before looking at them!).

## Summary, Exercises, Bibliography

- Look at our **notes** on the [webpage of the course](#), there are plenty of **details**, **proofs** and **exercises**. Here's one

### Exercise

The 2nd order encoding of  $A \vee B$  is:

$A \vee B := \forall^0 X. (A \rightarrow X) \rightarrow (B \rightarrow X) \rightarrow X$ . With that, show (by hand) that:

$$\text{callcc}(\lambda y v h. h(\lambda x. y(\lambda z w. z x))) \Vdash A \vee \neg A.$$

This is actually the proof-term of a derivation of the excluded middle from *Consequentia Mirabilis* (itself an instance of Peirce's law). Do *not* use adequacy though.

- The exercises have **solutions** (but try to do them by yourself before looking at them!).

### One million dollars exercise

Find a program  $M$  such that  $M \Vdash$  full Axiom of Choice

[*Hint (?)*: Krivine proved that one exists.]

- Where the idea of callcc with Peirce law was introduced:  
**A formulae-as-type notion of control**, Timothy G. Griffin, 1990,  
<https://dl.acm.org/doi/10.1145/96709.96714>
- A standard introduction to the topic:  
**Realizability in classical logic**, Jean-Luis Krivine, 2004,  
<https://www.irif.fr/~krivine/articles/Luminy04.pdf>
- A very nice and clear PhD manuscript on the topic:  
**On Forcing and Classical Realizability**, Lionel Rieg, 2014,  
<https://www-verimag.imag.fr/~riegl/assets/thesis-color.pdf>
- To go further (one cool example among many possible):  
**A program for the full Axiom of Choice**, Jean-Luis Krivine, 2021,  
[https://www.irif.fr/~krivine/articles/A\\_program\\_for\\_full\\_AC.pdf](https://www.irif.fr/~krivine/articles/A_program_for_full_AC.pdf)