A journey through Krivine realizability

Étienne Miquey

ÉNS de Lyon, LIP

Seminário de Lógica Matemática
16/11/2020

ENS DE LYON

Introduction

The cornerstone

Proof:

from a set of **hypotheses**
apply **deduction rules**
to obtain a **theorem**

Program:

from a set of **inputs**
apply **instructions**
to obtain the **output**

Curry-Howard

(On well-chosen subsets of mathematics and programs)

That's the same thing!

A somewhat obvious observation

Deduction rules

$$\frac{A \in \Gamma}{\Gamma \vdash A}\ (\text{Ax})$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ (\rightarrow_I)$$

$$\frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\ (\rightarrow_E)$$

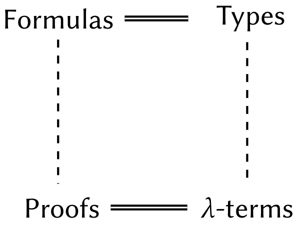
Typing rules

$$\frac{(x : A) \in \Gamma}{\Gamma \vdash x : A}\ (\text{Ax})$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \rightarrow B}\ (\rightarrow_I)$$

$$\frac{\Gamma \vdash t : A \rightarrow B \quad \Gamma \vdash u : A}{\Gamma \vdash t\, u : B}\ (\rightarrow_E)$$

Proofs-as-programs



Proofs-as-programs

The Curry-Howard correspondence

Mathematics	**Computer Science**
Proofs	Programs
Propositions	Types
Deduction rules	Typing rules
$\dfrac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\ (\Rightarrow_E)$	$\dfrac{\Gamma \vdash t : A \to B \quad \Gamma \vdash u : A}{\Gamma \vdash t\,u : B}\ (\to_E)$
A implies B	function $A \to B$
A and B	pair of A and B
$\forall x \in A.B(x)$	dependent product $\Pi x : A.B$

Benefits:

Program your proofs! | *Prove your programs!*

Proofs-as-programs

This is about **provability**.

syntax

Intuitionistic realizability

Brouwer-Heyting-Kolmogoroff interpretation (1932)

evidence of $A \wedge B$: evidence of A and an evidence of B;
evidence of $A \vee B$: either an evidence of A or an evidence of B;
evidence of $A \longrightarrow B$: transforms evidence of A into evidence of B;
evidence of $\perp$: absurdity $\perp$ (contradiction) has no evidence.

Intuitionistic realizability :

"evidence" = *realizer*

Intuitionistic realizability

Brouwer-Heyting-Kolmogoroff interpretation (1932)

evidence of $A \wedge B$: evidence of A and an evidence of B;
evidence of $A \vee B$: either an evidence of A or an evidence of B;
evidence of $A \longrightarrow B$: transforms evidence of A into evidence of B;
evidence of $\perp$: absurdity $\perp$ (contradiction) has no evidence.

Intuitionistic realizability :

"evidence" = *realizer*

Intuitionistic realizability

Kleene realizability (1945)

Realizers = Computable functions

(identified with its Gödel's number)

Rephrased with λ-terms:

$$
\begin{aligned}
t \Vdash A \wedge B &: \ t \longrightarrow (t_1, t_2) \ \text{ s.t. } \ t_1 \Vdash A \ \wedge \ t_2 \Vdash B; \\
t \Vdash A \vee B &: \ (t \longrightarrow \mathtt{inl}(u) \ \wedge \ u \Vdash A) \ \text{ or } \ (t \longrightarrow \mathtt{inr}(u) \ \wedge \ u \Vdash B); \\
t \Vdash A \rightarrow B &: \ \text{for any } u \Vdash A,\ tu \Vdash B; \\
\nVdash \bot &: \ \text{there is no } t \text{ s.t. } t \Vdash \bot
\end{aligned}
$$

Intuitionistic realizability

Kleene realizability (1945)

Realizers = Computable functions

(identified with its Gödel's number)

Rephrased with λ-terms:

$$
\begin{aligned}
t \Vdash A \wedge B &: \; t \longrightarrow (t_1, t_2) \text{ s.t. } t_1 \Vdash A \;\wedge\; t_2 \Vdash B; \\
t \Vdash A \vee B &: \; (t \longrightarrow \texttt{inl}(u) \;\wedge\; u \Vdash A) \text{ or } (t \longrightarrow \texttt{inr}(u) \;\wedge\; u \Vdash B); \\
t \Vdash A \longrightarrow B &: \; \text{for any } u \Vdash A,\ tu \Vdash B; \\
\nVdash \bot &: \; \text{there is no } t \text{ s.t. } t \Vdash \bot
\end{aligned}
$$

Realizability in 1 slide

Main intuition

t realizes *A* : *t* computes adequatly w.r.t. *A*

Soundness

$$HA \vdash A \qquad \Rightarrow \qquad t \Vdash A$$

A realizability interpretation is a **model**.

Consistency

$$HA \nvdash \bot$$

Consequences:

- disjunction/witness property
- normalization of typed terms
  reducibility candidates

Realizability in 1 slide

Main intuition

t realizes A : t computes adequatly w.r.t. A

Soundness

$$HA \vdash A \qquad \Rightarrow \qquad t \Vdash A$$

A realizability interpretation is a **model**.

Consistency

$$HA \nvdash \bot$$

Consequences:

- disjunction/witness property
- normalization of typed terms
  reducibility candidates

Realizability in 1 slide

Main intuition

t realizes A : t computes adequatly w.r.t. A

Adequacy

$$\vdash t : A \qquad \Rightarrow \qquad t \Vdash A$$

A realizability interpretation is a **model**.

Consistency

$$HA \nvdash \bot$$

Consequences:

- disjunction/witness property
- normalization of typed terms
  reducibility candidates

Realizability in 1 slide

Main intuition

t realizes *A* : *t* computes adequatly w.r.t. *A*

Soundness

$$HA \vdash A \qquad \Rightarrow \qquad t \Vdash A$$

A realizability interpretation is a **model**.

Consistency

$$HA \nvdash \bot$$

Consequences:

- disjunction/witness property
- normalization of typed terms
  reducibility candidates

Realizability in 1 slide

Main intuition

t realizes A : t computes adequatly w.r.t. A

Soundness

$$HA \vdash A \qquad \Rightarrow \qquad t \Vdash A$$

A realizability interpretation is a **model**.

Consistency

$$HA \nvdash \bot$$

Consequences:

- disjunction/witness property
- normalization of typed terms
  reducibility candidates

Realizability in 1 slide

Main intuition

t realizes A : t computes adequatly w.r.t. A

Soundness

$$HA \vdash A \qquad \Rightarrow \qquad t \Vdash A$$

A realizability interpretation is a **model**.

Consistency

$$HA \nvdash \bot$$

Consequences:

- disjunction/witness property
- normalization of typed terms
  reducibility candidates

Bad news

Yet a lot of things are missing

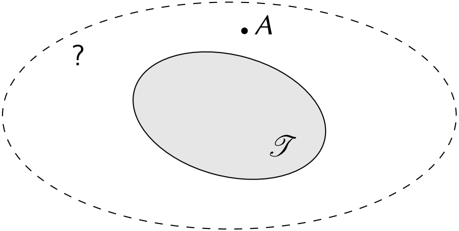
Limitations

Mathematics	Computer Science
$A \vee \neg A$	`try... catch ...`
$\neg\neg A \Rightarrow A$	`x := 42`
All sets can be well-ordered	`random()`
Sets that have the same elements are equal	`stop`
	`goto`

↬ *We want more !*

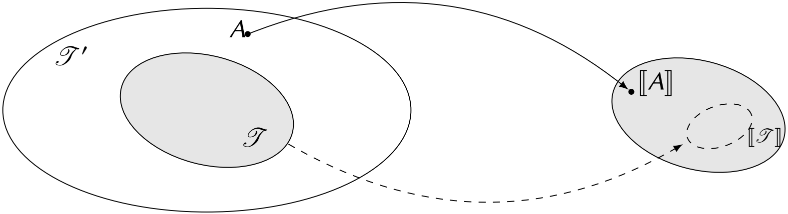
Non-constructive principles	*Side-effects*

Extending Curry-Howard



New axiom	$\sim$	Programing primitive
$\Updownarrow$		$\Updownarrow$
Logical translation	$\sim$	Program translation

Extending Curry-Howard

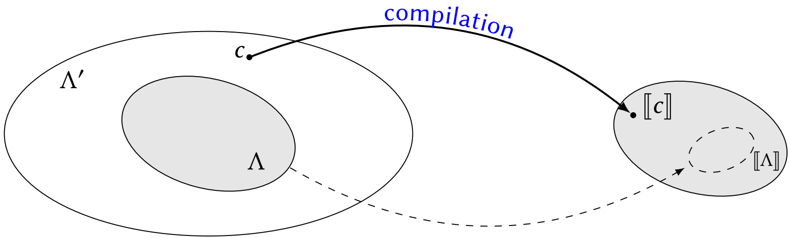


New axiom $\sim$ Programing primitive

$\Updownarrow$ $\Updownarrow$

Logical translation $\sim$ Program translation

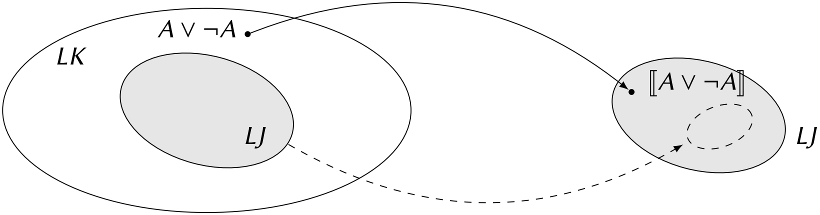
Extending Curry-Howard



New axiom	$\sim$	Programing primitive
$\Updownarrow$		$\Updownarrow$
Logical translation	$\sim$	Program translation

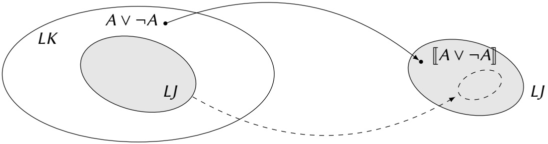
Classical logic

Classical logic = Intuitionistic logic + $A \vee \neg A$



Classical logic

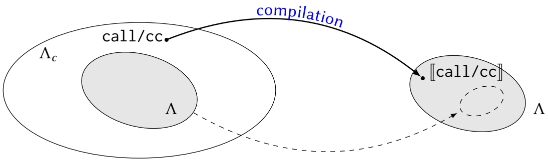
Classical logic = Intuitionistic logic + $A \vee \neg A$



New axiom		Programing primitive
$A \vee \neg A$	$\sim$	call/cc
Who doesn't use it?		Backtracking operator
$\Updownarrow$		$\Updownarrow$
Logical translation		Program translation
$A \mapsto \neg\neg A$	$\sim$	$2 \mapsto \lambda k.k\,2$
Gödel's negative translation		Continuation-passing style translation

Classical logic

Classical logic = Intuitionistic logic + $A \vee \neg A$



New axiom		Programing primitive
$A \vee \neg A$	~	call/cc
Who doesn't use it?		Backtracking operator
⇕		⇕
Logical translation		Program translation
$A \mapsto \neg\neg A$	~	$2 \mapsto \lambda k.k\,2$
Gödel's negative translation		Continuation-passing style translation

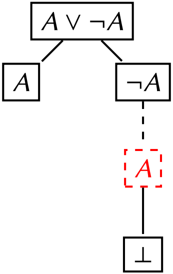
Computational content of classical logic

What is a program for $A \vee (A \to \bot)$?

In the pure λ-calculus:

- $A \vee B$ $\rightsquigarrow$ *choose* one side and give a proof
- $A \to B$ $\rightsquigarrow$ given a proof of A, computes a proof of B

Which side to choose?



Extension: `call/cc` allows us to *backtrack*!

1. Create a backtrack point
2. Play right: $A \to \bot$
3. Given a proof t of A, go back to 1
4. Play left: A
5. Give t

$$\mathtt{em} \triangleq \mathtt{call/cc}\,(\lambda k.\mathtt{inr}(\lambda t.k\,\mathtt{inl}(t)))$$

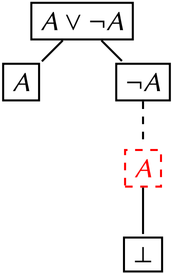
Computational content of classical logic

What is a program for $A \vee (A \to \bot)$?

In the pure λ-calculus:

- $A \vee B \rightsquigarrow$ *choose* one side and give a proof
- $A \to B \rightsquigarrow$ given a proof of A, computes a proof of B

Extension: `call/cc` allows us to *backtrack*!



1. Create a backtrack point
2. Play right: $A \to \bot$
3. Given a proof t of A, go back to 1
4. Play left: A
5. Give t

$\mathtt{em} \triangleq \mathtt{call/cc}\,(\lambda k.\mathtt{inr}(\lambda t.k\,\mathtt{inl}(t)))$

Computational content of classical logic

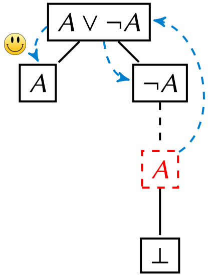
What is a program for $A \vee (A \to \bot)$?

In the pure λ-calculus:

- $A \vee B \rightsquigarrow$ *choose* one side and give a proof
- $A \to B \rightsquigarrow$ given a proof of A, computes a proof of B

Extension: `call/cc` allows us to *backtrack*!

1. Create a backtrack point
2. Play right: $A \to \bot$
3. Given a proof t of A, go back to 1
4. Play left: A
5. Give t



$\mathtt{em} \triangleq \mathtt{call/cc}\,(\lambda k.\mathtt{inr}(\lambda t.k\,\mathtt{inl}(t)))$

Computational content of classical logic

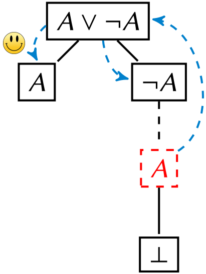
What is a program for $A \vee (A \rightarrow \bot)$?

In the pure λ-calculus:

- $A \vee B \rightsquigarrow$ *choose* one side and give a proof
- $A \rightarrow B \rightsquigarrow$ given a proof of A, computes a proof of B

Extension: `call/cc` allows us to *backtrack*!

1. Create a backtrack point
2. Play right: $A \rightarrow \bot$
3. Given a proof t of A, go back to 1
4. Play left: A
5. Give t



$$\texttt{em} \triangleq \texttt{call/cc}\,(\lambda k.\texttt{inr}(\lambda t.k\,\texttt{inl}(t)))$$

Curien-Herbelin's duality of computation

The duality of computation
Curien, Herbelin [2000]

Griffin (1990): classical logic ≅ control operator

Curien-Herbelin's observation:

Computational duality:

This is where classical computation lie.

Curien-Herbelin's duality of computation

The duality of computation
Curien, Herbelin [2000]

Griffin (1990): classical logic ≅ control operator

Curien-Herbelin's observation:

calculus and $\lambda\mu$-calculus. Our starting point was the observation that the call-by-value discipline manipulates input much in the same way as (the classical extension of) λ-calculus manipulates output. Computing MN in call-by-

Computational duality:

This is where classical computation lie.

Curien-Herbelin's duality of computation

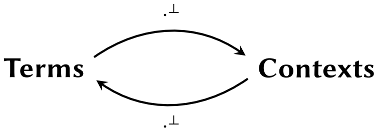
The duality of computation
Curien, Herbelin [2000]

Griffin (1990): classical logic ≅ control operator

Curien-Herbelin's observation:

calculus and $\lambda\mu$-calculus. Our starting point was the observation that the call-by-value discipline manipulates input much in the same way as (the classical extension of) λ-calculus manipulates output. Computing MN in call-by-

Computational duality:



This is where classical computation lie.

Curien-Herbelin's duality of computation

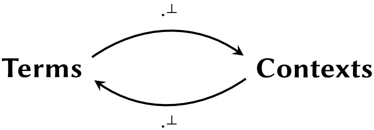
The duality of computation
Curien, Herbelin [2000]

Griffin (1990): classical logic ≅ control operator

Curien-Herbelin's observation:

> calculus and $\lambda\mu$-calculus. Our starting point was the observation that the call-by-value discipline manipulates input much in the same way as (the classical extension of) λ-calculus manipulates output. Computing MN in call-by-

Computational duality:

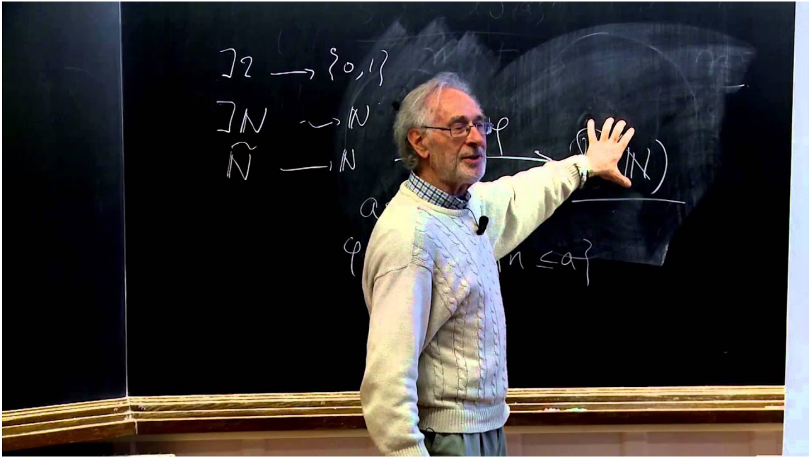


This is where classical computation lie.

Krivine realizability

Unveiling the computational contents of proofs

What belongs to Caesar...



What belongs to Caesar...

©V. Padovani

Krivine realizability, from above

- A **complete reformulation** of intuitionistic realizability.

  Necessary reformulation:

  $$\forall x.(H(x) \vee \neg H(x)) \textit{ not realized}$$

-
  - duality between terms / contexts
  - interaction player / opponent
- *(wait for the next slides)*
  - prove normalization/soundness properties
  - analyze computational behaviours of programs
  - build new models

  (wait for next week)

Krivine realizability, from above

- A **complete reformulation** of intuitionistic realizability.

- Computational **classical logic**:
  - duality between terms / contexts
  - interaction player / opponent

  (wait for the next slides)

- **Powerful tool** to:
  - prove normalization/soundness properties
  - analyze computational behaviours of programs
  - build new models

  (wait for next week)

Krivine realizability, from above

- A **complete reformulation** of intuitionistic realizability.

- Computational **classical logic**:
  - duality between terms / contexts
  - interaction player / opponent

  (wait for the next slides)

- **Powerful tool** to:
  - prove normalization/soundness properties
  - analyze computational behaviours of programs
  - build new models

  (wait for next week)

Krivine realizability, from inside

A 3-steps recipe

1. an operational semantics
2. a logical language
3. formulas interpretation

Krivine realizability, from inside

A 3-steps recipe

1. an operational semantics (*a.k.a. the abstract Krivine machine*)

$$
\begin{array}{lcrcl}
\textsc{Push} & : & \langle tu \parallel \pi \rangle & \succ & \langle t \parallel u \cdot \pi \rangle \\
\textsc{Grab} & : & \langle \lambda x.t \parallel u \cdot \pi \rangle & \succ & \langle t\{x := u\} \parallel \pi \rangle \\
\textsc{Save} & : & \langle \boldsymbol{cc} \parallel t \cdot \pi \rangle & \succ & \langle t \parallel \boldsymbol{k}_\pi \cdot \pi \rangle \\
\textsc{Restore} & : & \langle \boldsymbol{k}_\pi \parallel t \cdot \rho \rangle & \succ & \langle t \parallel \pi \rangle
\end{array}
$$

2. a logical language (*a.k.a. a type system*)

1st-order terms	e	$::=$	$x \mid f(e_1, \ldots, e_k)$
Formulas	A, B	$::=$	$X(e_1, \ldots, e_k) \mid A \Rightarrow B \mid \forall x.A \mid \forall X.A$

3. formulas interpretation

Krivine realizability, from inside

A 3-steps recipe

1. an operational semantics (*a.k.a. the abstract Krivine machine*)

$$
\begin{array}{lrcl}
\textsc{Push} : & \langle tu \parallel \pi\rangle & \succ & \langle t \parallel u \cdot \pi\rangle \\
\textsc{Grab} : & \langle \lambda x.t \parallel u \cdot \pi\rangle & \succ & \langle t\{x := u\} \parallel \pi\rangle \\
\textsc{Save} : & \langle \boldsymbol{cc} \parallel t \cdot \pi\rangle & \succ & \langle t \parallel \boldsymbol{k}_\pi \cdot \pi\rangle \\
\textsc{Restore} : & \langle \boldsymbol{k}_\pi \parallel t \cdot \rho\rangle & \succ & \langle t \parallel \pi\rangle
\end{array}
$$

2. a logical language (*a.k.a. a type system*)
3. formulas interpretation

- falsity value $\|A\|$: stacks, opponent to A
- truth value $|A|$: terms, player of A
- pole $\perp\!\!\!\perp$: processes, referee

λ_c-calculus

Terms, stacks, commands

Terms	t, u	$::=$	$x \mid \lambda x.t \mid tu \mid \mathbf{k}_\pi \mid \kappa$	$(\kappa \in C)$
Stacks	π	$::=$	$\alpha \mid t \cdot \pi$	$(\alpha \in \mathcal{B}, t$ closed$)$
Commands	c, c'	$::=$	$\langle t \parallel \pi \rangle$	$(t$ closed$)$

$\mathcal{B}$: stack constants

C: instructions (s.t. $\mathbf{cc} \in C$), enumerable

KAM

Push	$\langle tu \parallel \pi \rangle$	$\succ$	$\langle t \parallel u \cdot \pi \rangle$
Grab	$\langle \lambda x.t \parallel u \cdot \pi \rangle$	$\succ$	$\langle t\{x := u\} \parallel \pi \rangle$

λ_c-calculus

Terms, stacks, commands

Terms	t, u	$::=$	$x \mid \lambda x.t \mid tu \mid \mathbf{k}_\pi \mid \kappa$	$(\kappa \in \mathcal{C})$
Stacks	π	$::=$	$\alpha \mid t \cdot \pi$	$(\alpha \in \mathcal{B}, t \text{ closed})$
Commands	c, c'	$::=$	$\langle t \parallel \pi \rangle$	$(t \text{ closed})$

$\mathcal{B}$: stack constants *$\mathcal{C}$: instructions (s.t. $\mathbf{cc} \in \mathcal{C}$), enumerable*

KAM

Push	$\langle tu \parallel \pi \rangle$	$\succ$	$\langle t \parallel u \cdot \pi \rangle$
Grab	$\langle \lambda x.t \parallel u \cdot \pi \rangle$	$\succ$	$\langle t\{x := u\} \parallel \pi \rangle$

λ_c-calculus

Terms, stacks, commands

Terms	t, u	$::=$	$x \mid \lambda x.t \mid tu \mid \mathbf{k}_\pi \mid \kappa$	$(\kappa \in \mathcal{C})$
Stacks	π	$::=$	$\alpha \mid t \cdot \pi$	$(\alpha \in \mathcal{B}, t \text{ closed})$
Commands	c, c'	$::=$	$\langle t \parallel \pi \rangle$	$(t \text{ closed})$

$\mathcal{B}$: stack constants *$\mathcal{C}$: instructions (s.t. $\mathbf{cc} \in \mathcal{C}$), enumerable*

KAM

Push	$\langle tu \parallel \pi \rangle$	$\succ$	$\langle t \parallel u \cdot \pi \rangle$
Grab	$\langle \lambda x.t \parallel u \cdot \pi \rangle$	$\succ$	$\langle t\{x := u\} \parallel \pi \rangle$
Save	$\langle \mathbf{cc} \parallel t \cdot \pi \rangle$	$\succ$	$\langle t \parallel \mathbf{k}_\pi \cdot \pi \rangle$
Restore	$\langle \mathbf{k}_\pi \parallel t \cdot \rho \rangle$	$\succ$	$\langle t \parallel \pi \rangle$

Second order arithmetic

Language

Expressions	e	$::=$	$x \mid f(e_1, \ldots, e_k)$
Formulas	A, B	$::=$	$X(e_1, \ldots, e_k) \mid A \Rightarrow B \mid \forall x.A \mid \forall X.A$

Abbreviations :

$$
\begin{aligned}
\bot &\equiv \forall Z.Z \\
\neg A &\equiv A \Rightarrow \bot \\
A \wedge B &\equiv \forall Z((A \Rightarrow B \Rightarrow Z) \Rightarrow Z) \\
A \vee B &\equiv \forall Z((A \Rightarrow Z) \Rightarrow (B \Rightarrow Z) \Rightarrow Z) \\
A \Leftrightarrow B &\equiv (A \Rightarrow B) \wedge (B \Rightarrow A) \\
\exists x A(x) &\equiv \forall Z(\forall x(A(x) \Rightarrow Z) \Rightarrow Z) \\
\exists X A(X) &\equiv \forall Z(\forall X(A(X) \Rightarrow Z) \Rightarrow Z) \\
e_1 = e_2 &\equiv \forall Z(Z(e_1) \Rightarrow Z(e_2))
\end{aligned}
$$

Second order arithmetic

Language

Expressions	e	$::=$	$x \mid f(e_1, \ldots, e_k)$
Formulas	A, B	$::=$	$X(e_1, \ldots, e_k) \mid A \Rightarrow B \mid \forall x.A \mid \forall X.A$

Abbreviations :

$$\begin{aligned}
\bot &\equiv \forall Z.Z \\
\neg A &\equiv A \Rightarrow \bot \\
A \wedge B &\equiv \forall Z((A \Rightarrow B \Rightarrow Z) \Rightarrow Z) \\
A \vee B &\equiv \forall Z((A \Rightarrow Z) \Rightarrow (B \Rightarrow Z) \Rightarrow Z) \\
A \Leftrightarrow B &\equiv (A \Rightarrow B) \wedge (B \Rightarrow A) \\
\exists x A(x) &\equiv \forall Z(\forall x(A(x) \Rightarrow Z) \Rightarrow Z) \\
\exists X A(X) &\equiv \forall Z(\forall X(A(X) \Rightarrow Z) \Rightarrow Z) \\
e_1 = e_2 &\equiv \forall Z(Z(e_1) \Rightarrow Z(e_2))
\end{aligned}$$

Typing rules

$$\frac{}{\Gamma \vdash x : A}\ (x : A) \in \Gamma \qquad \frac{}{\Gamma \vdash t : \top}\ FV(t) \subset \mathrm{dom}(\Gamma)$$

$$\frac{\Gamma \vdash t : A \Rightarrow B \qquad \Gamma \vdash t : A}{\Gamma \vdash tu : B} \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \Rightarrow B}$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash t : \forall x.A}\ x \notin FV(\Gamma) \qquad \frac{\Gamma \vdash t : \forall x.A}{\Gamma \vdash t : A\{x := e\}}$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash t : \forall X.A}\ X \notin FV(\Gamma) \qquad \frac{\Gamma \vdash t : \forall X.A}{\Gamma \vdash t : A\{X(x_1, \ldots, x_k) := B\}}$$

$$\frac{}{\Gamma \vdash \mathbf{cc} : ((A \Rightarrow B) \Rightarrow A) \Rightarrow A}$$

Realizability interpretation (1/2)

Intuition

- falsity value $\|A\|$: stacks, opponent to A
- truth value $|A|$: proofs, player of A
- pole $\perp\!\!\!\perp$: commands, referee

$$\langle t \| \pi \rangle \succ c_0 \succ \cdots \succ c_n \in \perp\!\!\!\perp ?$$

$\rightsquigarrow$ $\perp\!\!\!\perp \subset \Lambda \times \Pi$ closed by anti-reduction

Truth value defined by **orthogonality** :

$$|A| = \|A\|^{\perp\!\!\!\perp} = \{t \in \Lambda : \forall \pi \in \|A\|, \langle t \| \pi \rangle \in \perp\!\!\!\perp\}$$

Realizability interpretation (1/2)

Intuition

- falsity value $\|A\|$: stacks, opponent to A
- truth value $|A|$: proofs, player of A
- pole $\perp\!\!\!\perp$: commands, referee

$$\langle t \| \pi \rangle \succ c_0 \succ \cdots \succ c_n \in \perp\!\!\!\perp ?$$

$\rightsquigarrow \perp\!\!\!\perp \subset \Lambda \times \Pi$ closed by anti-reduction

Truth value defined by **orthogonality** :

$$|A| = \|A\|^{\perp\!\!\!\perp} = \{t \in \Lambda : \forall \pi \in \|A\|, \langle t \| \pi \rangle \in \perp\!\!\!\perp\}$$

Realizability interpretation (1/2)

Intuition

- falsity value $\|A\|$: stacks, opponent to A
- truth value $|A|$: proofs, player of A
- pole $\perp\!\!\!\perp$: commands, referee

$$\langle t \parallel \pi \rangle \succ c_0 \succ \cdots \succ c_n \in \perp\!\!\!\perp ?$$

$\rightsquigarrow \perp\!\!\!\perp \subset \Lambda \times \Pi$ closed by anti-reduction

Truth value defined by **orthogonality** :

$$|A| = \|A\|^{\perp\!\!\!\perp} = \{t \in \Lambda : \forall \pi \in \|A\|, \langle t \parallel \pi \rangle \in \perp\!\!\!\perp\}$$

Realizability interpretation (1/2)

Intuition

- falsity value $\|A\|$: stacks, opponent to A
- truth value $|A|$: proofs, player of A
- pole $\perp\!\!\!\perp$: commands, referee

$$\langle t \,\|\, \pi \rangle \succ c_0 \succ \cdots \succ c_n \in \perp\!\!\!\perp ?$$

$\rightsquigarrow \perp\!\!\!\perp \subset \Lambda \times \Pi$ closed by anti-reduction

Truth value defined by **orthogonality** :

$$|A| = \|A\|^{\perp\!\!\!\perp} = \{t \in \Lambda : \forall \pi \in \|A\|, \langle t \,\|\, \pi \rangle \in \perp\!\!\!\perp\}$$

Realizability interpretation (1/2)

Intuition

- falsity value $\|A\|$: stacks, opponent to A
- truth value $|A|$: proofs, player of A
- pole $\perp\!\!\!\perp$: commands, referee

$$\langle t \parallel \pi \rangle \succ c_0 \succ \cdots \succ c_n \in \perp\!\!\!\perp ?$$

$\rightsquigarrow$ $\perp\!\!\!\perp \subset \Lambda \times \Pi$ closed by anti-reduction

Truth value defined by **orthogonality** :

$$|A| = \|A\|^{\perp\!\!\!\perp} = \{t \in \Lambda : \forall \pi \in \|A\|, \langle t \parallel \pi \rangle \in \perp\!\!\!\perp\}$$

Realizability interpretation (1/2)

Intuition

- falsity value $\|A\|$: stacks, opponent to A
- truth value $|A|$: proofs, player of A
- pole $⫫$: commands, referee

$$\langle t \,\|\, \pi \rangle \succ c_0 \succ \cdots \succ c_n \in ⫫ ?$$

$\leadsto ⫫ \subset \Lambda \times \Pi$ closed by anti-reduction

Truth value defined by **orthogonality** :

$$|A| = \|A\|^{⫫} = \{t \in \Lambda : \forall \pi \in \|A\|, \langle t \,\|\, \pi \rangle \in ⫫\}$$

Realizability interpretation (2/2)

Standard model $\mathbb{N}$ for 1st-order expressions

Definition (Pole)

$\perp\!\!\!\perp \subseteq \Lambda \times \Pi$ s.t. $\forall c, c', (c' \in \perp\!\!\!\perp \wedge c \rightarrow c') \quad \Rightarrow \quad c \in \perp\!\!\!\perp$

Falsity value (opponent):

$$\begin{aligned}
\|\dot{F}(e_1, \ldots, e_k)\| &= F([\![e_1]\!], \ldots, [\![e_k]\!]) \\
\|A \rightarrow B\| &= \{u \cdot \pi : \quad u \in |A| \wedge \pi \in \|B\|\} \\
\|\forall x.A\| &= \bigcup_{n \in \mathbb{N}} \|A[n/x]\| \\
\|\forall X.A\| &= \bigcup_{F:\mathbb{N}^k \rightarrow \mathcal{P}(\Pi)} \|A[\dot{F}/X]\|
\end{aligned}$$

Truth value (player):

$$|A| = \|A\|_E^{\perp\!\!\!\perp} = \{t \in \Lambda : \forall \pi \in \|A\|, \langle t \| \pi \rangle \in \perp\!\!\!\perp\}$$

Realizability interpretation (2/2)

Standard model $\mathbb{N}$ for 1st-order expressions

Definition (Pole)

$\perp\!\!\!\perp \subseteq \Lambda \times \Pi$ s.t. $\forall c, c', (c' \in \perp\!\!\!\perp \wedge c \to c') \Rightarrow c \in \perp\!\!\!\perp$

Falsity value (opponent):

$$\begin{array}{rcl} \|\dot{F}(e_1, \ldots, e_k)\| & = & F(\llbracket e_1 \rrbracket, \ldots, \llbracket e_k \rrbracket) \\ \|A \to B\| & = & \{u \cdot \pi : \quad u \in |A| \wedge \pi \in \|B\|\} \\ \|\forall x.A\| & = & \bigcup_{n \in \mathbb{N}} \|A[n/x]\| \\ \|\forall X.A\| & = & \bigcup_{F : \mathbb{N}^k \to \mathcal{P}(\Pi)} \|A[\dot{F}/X]\| \end{array}$$

Truth value (player):

$$|A| = \|A\|_E^{\perp\!\!\!\perp} = \{t \in \Lambda : \forall \pi \in \|A\|, \langle t \| \pi \rangle \in \perp\!\!\!\perp\}$$

Realizability interpretation (2/2)

Standard model $\mathbb{N}$ for 1st-order expressions

Definition (Pole)

$\perp\!\!\!\perp \subseteq \Lambda \times \Pi$ s.t. $\forall c, c', (c' \in \perp\!\!\!\perp \wedge c \to c') \Rightarrow c \in \perp\!\!\!\perp$

Falsity value (opponent):

$$\begin{aligned}
\|\dot{F}(e_1, \ldots, e_k)\| &= F([\![e_1]\!], \ldots, [\![e_k]\!]) \\
\|A \to B\| &= \{u \cdot \pi : \quad u \in |A| \wedge \pi \in \|B\|\} \\
\|\forall x.A\| &= \bigcup_{n \in \mathbb{N}} \|A[n/x]\| \\
\|\forall X.A\| &= \bigcup_{F : \mathbb{N}^k \to \mathcal{P}(\Pi)} \|A[\dot{F}/X]\|
\end{aligned}$$

Truth value (player):

$$|A| = \|A\|_E{}^{\perp\!\!\!\perp} = \{t \in \Lambda : \forall \pi \in \|A\|, \langle t \,\|\, \pi \rangle \in \perp\!\!\!\perp\}$$

Realizability interpretation (2/2)

Standard model $\mathbb{N}$ for 1st-order expressions

Definition (Pole)

$\perp\!\!\!\perp \subseteq \Lambda \times \Pi$ s.t. $\forall c, c', (c' \in \perp\!\!\!\perp \wedge c \rightarrow c') \quad \Rightarrow \quad c \in \perp\!\!\!\perp$

Falsity value (opponent):

$$\begin{array}{rcl} \|\dot{F}(e_1, \ldots, e_k)\| & = & F([\![e_1]\!], \ldots, [\![e_k]\!]) \\ \|A \rightarrow B\| & = & \{u \cdot \pi : \quad u \in |A| \wedge \pi \in \|B\|\} \\ \|\forall x.A\| & = & \bigcup_{n \in \mathbb{N}} \|A[n/x]\| \\ \|\forall X.A\| & = & \bigcup_{F:\mathbb{N}^k \rightarrow \mathcal{P}(\Pi)} \|A[\dot{F}/X]\| \end{array}$$

Truth value (player):

$$|A| = \|A\|_E^{\perp\!\!\!\perp} = \{t \in \Lambda : \forall \pi \in \|A\|, \langle t \,\|\, \pi \rangle \in \perp\!\!\!\perp\}$$

One lemma to rule them all

Adequacy

If $\vdash t : A$ then $t \in |A|$ for any pole.

Consequences

Normalizing commands

$$\perp\!\!\!\perp_{\Downarrow} \triangleq \{c : c \text{ normalizes}\} \text{ defines a valid pole.}$$

Proof. If $c \to c'$ and c' normalizes, so does c. □

Normalization

Typed terms normalize.

Proof. Using the lemma above. □

Soundness

There is no term t such that $\vdash t : \bot$.

Proof. Otherwise, $t \in |\bot| = \Pi^{\perp\!\!\!\perp}$ for any pole, absurd ($\perp\!\!\!\perp \triangleq \emptyset$). □

Consequences

Normalizing commands

$\perp\!\!\!\perp_{\Downarrow} \triangleq \{c : c \text{ normalizes}\}$ defines a valid pole.

Proof. *If $c \rightarrow c'$ and c' normalizes, so does c.* □

Normalization

Typed terms normalize.

Proof. *Using the lemma above.* □

Soundness

There is no term t such that $\vdash t : \bot$.

Proof. *Otherwise, $t \in |\bot| = \Pi^{\perp\!\!\!\perp}$ for any pole, absurd ($\perp\!\!\!\perp \triangleq \emptyset$).* □

Consequences

Normalizing commands

$⫫_{\Downarrow} \triangleq \{c : c \text{ normalizes}\}$ defines a valid pole.

Proof. If $c \rightarrow c'$ and c' normalizes, so does c. □

Normalization

Typed terms normalize.

Proof. Using the lemma above. □

Soundness

There is no term t such that $\vdash t : \bot$.

Proof. Otherwise, $t \in |\bot| = \Pi^{⫫}$ for any pole, absurd ($⫫ \triangleq \emptyset$). □

Connecting the dots

We saw that:

1.

New axiom		Programing primitive
$A \vee \neg A$	$\sim$	`call/cc`
Who doesn't use it?		Backtracking operator
$\Updownarrow$		$\Updownarrow$
Logical translation		Program translation
$A \mapsto \neg\neg A$	$\sim$	$2 \mapsto \lambda k.k\,2$
Gödel's negative translation		Continuation-passing style translation

2. $\forall x.(H(x) \vee \neg H(x))$ is not realized intuitionistically.

On Krivine's Realizability Interpretation of Classical Second-Order Arithmetic
Oliva, Streicher [2008]

In fact:

Theorem

Krivine $=$ Int. Realizability $\circ$ Negative translation

Connecting the dots

We saw that:

1.

New axiom		Programing primitive
$A \vee \neg A$	$\sim$	call/cc
Who doesn't use it?		Backtracking operator
$\Updownarrow$		$\Updownarrow$
Logical translation		Program translation
$A \mapsto \neg\neg A$	$\sim$	$2 \mapsto \lambda k.k\,2$
Gödel's negative translation		Continuation-passing style translation

2. $\forall x.(H(x) \vee \neg H(x))$ is not realized intuitionistically.

In fact:

Theorem

Krivine = Int. Realizability $\circ$ Negative translation

On Krivine's Realizability Interpretation of Classical Second-Order Arithmetic
Oliva, Streicher [2008]

Connecting the dots

We saw that:

1.

New axiom		Programing primitive
$A \vee \neg A$	$\sim$	call/cc
Who doesn't use it?		Backtracking operator
$\Updownarrow$		$\Updownarrow$
Logical translation		Program translation
$A \mapsto \neg\neg A$	$\sim$	$2 \mapsto \lambda k.k\,2$
Gödel's negative translation		Continuation-passing style translation

2. $\forall x.(H(x) \vee \neg H(x))$ is not realized intuitionistically.

In fact:

Theorem

Krivine = Int. Realizability ∘ Negative translation

On Krivine's Realizability Interpretation of Classical Second-Order Arithmetic
Oliva, Streicher [2008]

Connecting the dots

We saw that:

1.

New axiom		Programing primitive
$A \vee \neg A$	$\sim$	call/cc
Who doesn't use it?		Backtracking operator
$\Updownarrow$		$\Updownarrow$
Logical translation		Program translation
$A \mapsto \neg\neg A$	$\sim$	$2 \mapsto \lambda k.k\,2$
Gödel's negative translation		Continuation-passing style translation

2. $\forall x.(H(x) \vee \neg H(x))$ is not realized intuitionistically.

On Krivine's Realizability Interpretation of Classical Second-Order Arithmetic
Oliva, Streicher [2008]

In fact:

Theorem

Krivine = Int. Realizability $\circ$ Negative translation

User guide

Diving into the details

HowTo: Realizability proofs

Building realizers (1/2)

The easiest proof of all :

$$\lambda x.x \Vdash \forall X.(X \Rightarrow X)$$

Proof :

Let $\bot\!\!\!\bot$ be any pole. By definition, we have

$$\|\forall X(X \Rightarrow X)\| = \bigcup_{\mathbb{X} \in \mathcal{P}(\Pi)} \|\dot{\mathbb{X}} \Rightarrow \dot{\mathbb{X}}\|$$

$$= \bigcup_{\mathbb{X} \in \mathcal{P}(\Pi)} \{u \cdot \pi : u \in |\mathbb{X}| \wedge \pi \in \mathbb{X}\}$$

Besides,

$$\langle \lambda x.x \| u \cdot \pi \rangle \succ \langle u \| \pi \rangle$$

Then by anti-reduction, we get :

$$\langle \lambda x.x \| u \cdot \pi \rangle \in \bot\!\!\!\bot$$

Building realizers (1/2)

The easiest proof of all :

$$\lambda x.x \Vdash \forall X.(X \Rightarrow X)$$

Proof :

Let $⫫$ be any pole. By definition, we have

$$\|\forall X(X \Rightarrow X)\| = \bigcup_{\mathbb{X} \in \mathcal{P}(\Pi)} \|\dot{\mathbb{X}} \Rightarrow \dot{\mathbb{X}}\|$$

$$= \bigcup_{\mathbb{X} \in \mathcal{P}(\Pi)} \{u \cdot \pi : u \in |\dot{\mathbb{X}}| \wedge \pi \in \mathbb{X}\}$$

Besides,

$$\langle \lambda x.x \| u \cdot \pi \rangle \succ \langle u \| \pi \rangle$$

Then by anti-reduction, we get :

$$\langle \lambda x.x \| u \cdot \pi \rangle \in ⫫$$

Building realizers (1/2)

The easiest proof of all :

$$\lambda x.x \Vdash \forall X.(X \Rightarrow X)$$

Proof :

Let $\perp\!\!\!\perp$ be any pole. By definition, we have

$$\begin{aligned}\|\forall X(X \Rightarrow X)\| &= \bigcup_{\mathbb{X}\in\mathcal{P}(\Pi)} \|\dot{\mathbb{X}} \Rightarrow \dot{\mathbb{X}}\| \\ &= \bigcup_{\mathbb{X}\in\mathcal{P}(\Pi)} \{u \cdot \pi : u \in |\dot{\mathbb{X}}| \wedge \pi \in \mathbb{X}\}\end{aligned}$$

Besides,

$$\langle \lambda x.x \,\|\, u \cdot \pi \rangle \succ \langle u \,\|\, \pi \rangle \;.$$

Then by anti-reduction, we get :

$$\langle \lambda x.x \,\|\, u \cdot \pi \rangle \in \perp\!\!\!\perp$$

Building realizers (1/2)

The easiest proof of all :

$$\lambda x.x \Vdash \forall X.(X \Rightarrow X)$$

Proof :

Let $⫫$ be any pole. By definition, we have

$$\begin{aligned}\|\forall X(X \Rightarrow X)\| &= \bigcup_{\mathbb{X} \in \mathcal{P}(\Pi)} \|\dot{\mathbb{X}} \Rightarrow \dot{\mathbb{X}}\| \\ &= \bigcup_{\mathbb{X} \in \mathcal{P}(\Pi)} \{u \cdot \pi : u \in |\dot{\mathbb{X}}| \wedge \pi \in \mathbb{X}\}\end{aligned}$$

Besides,

$$\langle \lambda x.x \| u \cdot \pi \rangle \succ \langle u \| \pi \rangle \in ⫫$$

Then by anti-reduction, we get :

$$\langle \lambda x.x \| u \cdot \pi \rangle \in ⫫$$

Building realizers (1/2)

The easiest proof of all :

$$\lambda x.x \Vdash \forall X.(X \Rightarrow X)$$

Proof :
Let $\perp\!\!\!\perp$ be any pole. By definition, we have

$$\begin{aligned} \|\forall X(X \Rightarrow X)\| &= \bigcup_{\mathbb{X}\in\mathcal{P}(\Pi)} \|\dot{\mathbb{X}} \Rightarrow \dot{\mathbb{X}}\| \\ &= \bigcup_{\mathbb{X}\in\mathcal{P}(\Pi)} \{u\cdot\pi : u \in |\dot{\mathbb{X}}| \wedge \pi \in \mathbb{X}\} \end{aligned}$$

Besides,

$$\langle \lambda x.x \| u\cdot\pi\rangle \succ \langle u \| \pi\rangle \in \perp\!\!\!\perp$$

Then by anti-reduction, we get :

$$\langle \lambda x.x \| u\cdot\pi\rangle \in \perp\!\!\!\perp$$

Building realizers (2/2)

The most classical proof of all

$$\texttt{em} := \lambda \ell r.\mathbf{cc}\,(\lambda k.r\,(\lambda a.k\,(\ell\ a))) \Vdash \forall A.(A \vee \neg A)$$

Proof :

Let $⫫$ be any pole. By definition, we have

$$\forall A.(A \vee \neg A) \equiv \forall A.(\forall X.((A \Rightarrow X) \Rightarrow ((A \Rightarrow \bot) \Rightarrow X) \Rightarrow X)$$

Besides,

$$\langle \texttt{em} \parallel \ell \cdot r \cdot \pi \rangle \succ \langle r \parallel \lambda a.\mathbf{k}_\pi(\ell\ a) \cdot \pi \rangle$$

The question become :

$$\lambda a.\mathbf{k}_\pi(\ell\ a) \Vdash A \Rightarrow \bot ?$$

We conclude by anti-reduction, since for any $a \in |A|$, $\rho \in \|\bot\|$:

$$\langle \lambda a.\mathbf{k}_\pi(\ell\ a) \parallel a \cdot \rho \rangle \succ \langle \ell a \parallel \pi \rangle \succ \langle \ell \parallel \underbrace{a \cdot \pi}_{\in \|A \to X\|} \rangle \in ⫫$$

Building realizers (2/2)

The most classical proof of all

$$\mathtt{em} := \lambda \ell r.\mathbf{cc}\,(\lambda k.r\,(\lambda a.k\,(\ell\, a))) \Vdash \forall A.(A \vee \neg A)$$

Proof :

Let $\perp\!\!\!\perp$ be any pole. By definition, we have

$$\forall A.(A \vee \neg A) \equiv \forall A.(\forall X.((A \Rightarrow X) \Rightarrow ((A \Rightarrow \bot) \Rightarrow X) \Rightarrow X)$$

Besides,

$$\langle \mathtt{em} \,\|\, \ell \cdot r \cdot \pi \rangle \succ \langle r \,\|\, \lambda a.\mathbf{k}_\pi(\ell\, a) \cdot \pi \rangle$$

The question become :

$$\lambda a.\mathbf{k}_\pi(\ell\, a) \Vdash A \Rightarrow \bot ?$$

We conclude by anti-reduction, since for any $a \in |A|, \rho \in \|\bot\|$:

$$\langle \lambda a.\mathbf{k}_\pi(\ell\, a) \,\|\, a \cdot \rho \rangle \succ \langle \ell a \,\|\, \pi \rangle \succ \langle \ell \,\|\, \underbrace{a \cdot \pi}_{\in \|A \to X\|} \rangle \in \perp\!\!\!\perp$$

Building realizers (2/2)

The most classical proof of all

$$\texttt{em} := \lambda \ell r.\mathbf{cc}\,(\lambda k.r\,(\lambda a.k\,(\ell\; a))) \Vdash \forall A.(A \vee \neg A)$$

Proof :
Let $⫫$ be any pole. By definition, we have

$$\forall A.(A \vee \neg A) \equiv \forall A.(\forall X.((A \Rightarrow X) \Rightarrow ((A \Rightarrow \bot) \Rightarrow X) \Rightarrow X)$$

Besides,

$$\langle \texttt{em} \,\|\, \ell \cdot r \cdot \pi \rangle \succ \langle r \,\|\, \lambda a.\mathbf{k}_\pi(\ell\; a) \cdot \pi \rangle$$

The question become :

$$\lambda a.\mathbf{k}_\pi(\ell\; a) \Vdash A \Rightarrow \bot ?$$

We conclude by anti-reduction, since for any $a \in |A|$, $\rho \in \|\bot\|$:

$$\langle \lambda a.\mathbf{k}_\pi(\ell\; a) \,\|\, a \cdot \rho \rangle \succ \langle \ell a \,\|\, \pi \rangle \succ \langle \ell \,\|\, \underbrace{a \cdot \pi}_{\in \|A \to X\|} \rangle \in ⫫$$

Building realizers (2/2)

The most classical proof of all

$$\texttt{em} := \lambda \ell r.\mathbf{cc}\,(\lambda k.r\,(\lambda a.k\,(\ell\ a))) \Vdash \forall A.(A \vee \neg A)$$

Proof :
Let ⫫ be any pole. By definition, we have

$$\forall A.(A \vee \neg A) \equiv \forall A.(\forall X.((A \Rightarrow X) \Rightarrow ((A \Rightarrow \bot) \Rightarrow X) \Rightarrow X)$$

Besides,

$$\langle \texttt{em} \,\|\, \ell \cdot r \cdot \pi \rangle \succ \langle r \,\|\, \lambda a.\mathbf{k}_\pi(\ell\ a) \cdot \pi \rangle$$

The question become :

$$\lambda a.\mathbf{k}_\pi(\ell\ a) \Vdash A \Rightarrow \bot ?$$

We conclude by anti-reduction, since for any $a \in |A|, \rho \in \|\bot\|$:

$$\langle \lambda a.\mathbf{k}_\pi(\ell\ a) \,\|\, a \cdot \rho \rangle \succ \langle \ell a \,\|\, \pi \rangle \succ \langle \ell \,\|\, \underbrace{a \cdot \pi}_{\in \|A \to X\|} \rangle \in ⫫$$

Building realizers (2/2)

The most classical proof of all

$$\mathtt{em} := \lambda \ell r.\mathbf{cc}\,(\lambda k.r\,(\lambda a.k\,(\ell\; a))) \Vdash \forall A.(A \vee \neg A)$$

Proof :

Let $⫫$ be any pole. By definition, we have

$$\forall A.(A \vee \neg A) \equiv \forall A.(\forall X.((A \Rightarrow X) \Rightarrow ((A \Rightarrow \bot) \Rightarrow X) \Rightarrow X)$$

Besides,

$$\langle \mathtt{em} \,\|\, \ell \cdot r \cdot \pi \rangle \succ \langle r \,\|\, \lambda a.\boldsymbol{k}_\pi(\ell\; a) \cdot \pi \rangle$$

The question become :

$$\lambda a.\boldsymbol{k}_\pi(\ell\; a) \Vdash A \Rightarrow \bot ?$$

We conclude by anti-reduction, since for any $a \in |A|, \rho \in \|\bot\|$:

$$\langle \lambda a.\boldsymbol{k}_\pi(\ell\; a) \,\|\, a \cdot \rho \rangle \succ \langle \ell a \,\|\, \pi \rangle \succ \langle \ell \,\|\, \underbrace{a \cdot \pi}_{\in \|A \to X\|} \rangle \in ⫫$$

Adequacy

Adequacy

If $\vdash t : A$ then $t \in |A|$ for any pole.

Proof?

Claim:

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :

Adequacy

Adequacy

If $\vdash t : A$ then $t \in |A|$ for any pole.

Claim:

Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :

Adequacy

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash_\rho \Gamma$ then $\sigma(t) \in |A|_\rho$ for any pole and any ρ.

Claim:

Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :
Induction on typing rules.

Adequacy

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash_\rho \Gamma$ then $\sigma(t) \in |A|_\rho$ for any pole and any ρ.

Claim:

Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :

Case $\dfrac{\Gamma \vdash t : A \to B \quad \Gamma \vdash u : B}{\Gamma \vdash t\, u : B}\ (\to_I)$

IH :
- $t \Vdash_\rho A \to B$
- $u \Vdash_\rho A$

Goal : $tu \Vdash_\rho B$

Let $\pi \in \|B\|_\rho$, we have :

$$\langle tu \,\|\, \pi \rangle \succ \langle t \,\|\, u \cdot \pi \rangle$$

By **IH**: $u \cdot \pi \in \|A \to B\|_\rho$, hence the result by anti-reduction.

Adequacy

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash_\rho \Gamma$ then $\sigma(t) \in |A|_\rho$ for any pole and any ρ.

Claim:

Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :

Case $$\frac{\Gamma \vdash t : A \to B \quad \Gamma \vdash u : B}{\Gamma \vdash t\, u : B}\ (\to_I)$$

IH :
- $t \Vdash_\rho A \to B$
- $u \Vdash_\rho A$

Goal : $tu \Vdash_\rho B$

Let $\pi \in \|B\|_\rho$, we have :

$$\langle tu \,\|\, \pi \rangle \succ \langle t \,\|\, u \cdot \pi \rangle$$

By **IH**: $u \cdot \pi \in \|A \to B\|_\rho$, hence the result by anti-reduction.

Adequacy

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash_\rho \Gamma$ then $\sigma(t) \in |A|_\rho$ for any pole and any ρ.

Claim:

Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :

Case $$\frac{\Gamma \vdash t : A \to B \quad \Gamma \vdash u : B}{\Gamma \vdash t\,u : B}\ (\to_I)$$

IH :
- $t \Vdash_\rho A \to B$
- $u \Vdash_\rho A$

Goal : $tu \Vdash_\rho B$

Let $\pi \in \|B\|_\rho$, we have :

$$\langle tu \,\|\, \pi \rangle \succ \langle t \,\|\, u \cdot \pi \rangle \in \perp\!\!\!\perp$$

By **IH**: $u \cdot \pi \in \|A \to B\|_\rho$, hence the result by anti-reduction.

Adequacy

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash_\rho \Gamma$ then $\sigma(t) \in |A|_\rho$ for any pole and any ρ.

Claim:

Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :

Case $\dfrac{\Gamma \vdash t : A \to B \quad \Gamma \vdash u : B}{\Gamma \vdash t\, u : B}\ (\to_I)$

IH :
- $t \Vdash_\rho A \to B$
- $u \Vdash_\rho A$

Goal : $tu \Vdash_\rho B$

Let $\pi \in \|B\|_\rho$, we have :

$$\langle tu \,\|\, \pi \rangle \succ \langle t \,\|\, u \cdot \pi \rangle \in \perp\!\!\!\perp$$

Push

By **IH**: $u \cdot \pi \in \|A \to B\|_\rho$, hence the result by anti-reduction.

Adequacy

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash_\rho \Gamma$ then $\sigma(t) \in |A|_\rho$ for any pole and any ρ.

Claim:

Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :

Case $\dfrac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B}\ (\to_E)$

IH : $\sigma' \Vdash_\rho \Gamma, x : a \ \Rightarrow\ \sigma'(t) \Vdash_\rho B$

Goal : $\sigma(\lambda x.t) \Vdash_\rho A \to B$

Let $u \cdot \pi \in \|A \to B\|_\rho$, we have :

$$\langle \sigma(\lambda x.t) \parallel u \cdot \pi \rangle \succ \langle \sigma(t)[u/x] \parallel \pi \rangle$$

Since: $\sigma, x \mapsto u \Vdash_\rho \Gamma, x : A$

By **IH**: $\sigma(t)[u/x] \in |B|_\rho$, hence the result by anti-reduction.

Adequacy

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash_\rho \Gamma$ then $\sigma(t) \in |A|_\rho$ for any pole and any ρ.

Claim:

Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :

Case $\dfrac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B}\ (\to_E)$

IH : $\sigma' \Vdash_\rho \Gamma, x : a \ \Rightarrow\ \sigma'(t) \Vdash_\rho B$
Goal : $\sigma(\lambda x.t) \Vdash_\rho A \to B$

Let $u \cdot \pi \in \|A \to B\|_\rho$, we have :

$$\langle \sigma(\lambda x.t) \,\|\, u \cdot \pi \rangle \succ \langle \sigma(t)[u/x] \,\|\, \pi \rangle$$

Since: $\sigma, x \mapsto u \Vdash_\rho \Gamma, x : A$
By **IH**: $\sigma(t)[u/x] \in |B|_\rho$, hence the result by anti-reduction.

Adequacy

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash_\rho \Gamma$ then $\sigma(t) \in |A|_\rho$ for any pole and any ρ.

Claim:

Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :

Case $\dfrac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B}\ (\to_E)$

IH : $\sigma' \Vdash_\rho \Gamma, x : a \Rightarrow \sigma'(t) \Vdash_\rho B$
Goal : $\sigma(\lambda x.t) \Vdash_\rho A \to B$

Let $u \cdot \pi \in \|A \to B\|_\rho$, we have :

$$\langle \sigma(\lambda x.t) \parallel u \cdot \pi \rangle \succ \langle \sigma(t)[u/x] \parallel \pi \rangle \in \perp\!\!\!\perp$$

Since: $\sigma, x \mapsto u \Vdash_\rho \Gamma, x : A$
By **IH**: $\sigma(t)[u/x] \in |B|_\rho$, hence the result by anti-reduction.

Adequacy

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash_\rho \Gamma$ then $\sigma(t) \in |A|_\rho$ for any pole and any ρ.

Claim:

Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

Proof :

Case $\dfrac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \to B} \, (\to_E)$

IH : $\sigma' \Vdash_\rho \Gamma, x : a \;\Rightarrow\; \sigma'(t) \Vdash_\rho B$
Goal : $\sigma(\lambda x.t) \Vdash_\rho A \to B$

Let $u \cdot \pi \in \|A \to B\|_\rho$, we have :

$$\langle \sigma(\lambda x.t) \,\|\, u \cdot \pi \rangle \succ \langle \sigma(t)[u/x] \,\|\, \pi \rangle \in \perp\!\!\!\perp$$

Grab

Since: $\sigma, x \mapsto u \Vdash_\rho \Gamma, x : A$
By **IH**: $\sigma(t)[u/x] \in |B|_\rho$, hence the result by anti-reduction.

Adequacy

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash_\rho \Gamma$ then $\sigma(t) \in |A|_\rho$ for any pole and any ρ.

Claim:

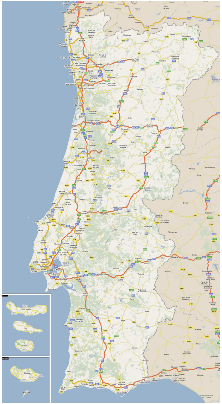
Normalization by realizability
Dagand, Rieg, Scherer [2019]

"The adequacy theorem [...] corresponds to an evaluation procedure."

✓ Yes, indeed!

Typing vs. realizability

This is highly **syntactic**.
(provability)



Typing
$\vdash t : A$

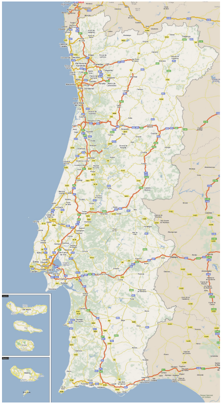
Realizability is about **semantics**.
(validity)

Realizability
$t \Vdash A$

Typing vs. realizability

This is highly **syntactic**.
(provability)

Realizability is about **semantics**.
(validity)



$\subseteq$
adequacy

Typing
$\vdash t : A$

Realizability
$t \Vdash A$

HowTo: Extensions

Modularity at the rescue

An innocent riddle

Say we have an axiom we cannot realize. What can we do?

(Hint: think of $A \vee \neg A$ and `call/cc` *)*

An innocent riddle

Say we have an axiom we cannot realize. What can we do?

(Hint: think of $A \vee \neg A$ and `call/cc` *)*

An innocent riddle

The motto

With side-effects come new reasoning principles.

An innocent riddle

Terms, stacks, commands

Terms	t, u	$::=$	$x \mid \lambda x.t \mid tu \mid \boldsymbol{k}_\pi \mid \kappa$	$(\kappa \in \mathcal{C})$
Stacks	π	$::=$	$\alpha \mid t \cdot \pi$	$(\alpha \in \mathcal{B}, t \text{ closed})$
Commands	c, c'	$::=$	$\langle t \parallel \pi \rangle$	$(t \text{ closed})$

$\mathcal{B}$: stack constants

$\mathcal{C}$: instructions (s.t. $\boldsymbol{cc} \in \mathcal{C}$), enumerable

Realizing dependent choice

quote

Suppose we have a surjective map $n \mapsto t_n$, we set :

(Quote) $$\langle \texttt{quote} \parallel t \cdot u \cdot \pi \rangle \succ \langle u \parallel \overline{n} \cdot \pi \rangle$$

where n is any integer such that $t_n = t$.

Dependent Choice [Krivine'03]

Using quote, one can define ϕ st $t \Vdash DC$

Actually, not the only approach to get DC:

- with a bar recursor (Berger-Oliva, Krivine, Blot)
- via dependent types and memoization (Herbelin, M.)

Remarks

- If $t \Vdash A$ for any pole, then $\vdash t : A$ is a valid axiom from the

Realizing dependent choice

Dependent Choice [Krivine'03]

Using `quote`, one can define ϕ st $t \Vdash DC$

Actually, not the only approach to get DC:

- with a bar recursor (Berger-Oliva, Krivine, Blot)
- via dependent types and memoization (Herbelin, M.)

Remarks

- If $t \Vdash A$ for any pole, then $\vdash t : A$ is a valid axiom from the computational point-of-view
- We can add other instructions : ⫛, eq, `stop`, `print`...
- Adding new instructions may break some properties!

Realizing dependent choice

Dependent Choice [Krivine'03]

Using quote, one can define ϕ st $t \Vdash DC$

Actually, not the only approach to get DC:

- with a bar recursor (Berger-Oliva, Krivine, Blot)
- via dependent types and memoization (Herbelin, M.)

Remarks

- If $t \Vdash A$ for any pole, then $\vdash t : A$ is a valid axiom from the computational point-of-view
- We can add other instructions : $\pitchfork$, eq, stop, print...
- Adding new instructions may break some properties!

HowTo: Get your own interpretation

Danvy's semantic artifacts

The $\lambda\mu\tilde{\mu}$-calculus

The duality of computation
Curien, Herbelin [2000]

Syntax:

Terms	$t ::= x \mid \lambda x.t \mid \mu\alpha.c$
Contexts	$e ::= \alpha \mid t \cdot e \mid \tilde{\mu}x.c$
Commands	$c ::= \langle t \| e \rangle$

Reduction:

$$\begin{aligned}
\langle \lambda x.t \| u \cdot e \rangle &\to \overbrace{\langle u \| \tilde{\mu}x.\langle t \| e \rangle \rangle}^{\texttt{let } x=u \texttt{ in } \langle t \| e \rangle} \\
\langle t \| \tilde{\mu}x.c \rangle &\to c[t/x] \\
\langle \mu\alpha.c \| e \rangle &\to c[e/\alpha]
\end{aligned}$$

The $\lambda\mu\tilde{\mu}$-calculus

The duality of computation
Curien, Herbelin [2000]

Syntax:

TERMS	$t ::= x \mid \lambda x.t \mid \mu\alpha.c$
CONTEXTS	$e ::= \alpha \mid t \cdot e \mid \tilde{\mu}x.c$
COMMANDS	$c ::= \langle t \parallel e \rangle$

Reduction:

$$\begin{array}{rcl} \langle \lambda x.t \parallel u \cdot e \rangle & \to & \langle u \parallel \tilde{\mu}x.\langle t \parallel e \rangle \rangle \\ \langle t \parallel \tilde{\mu}x.c \rangle & \to & c[t/x] \\ \langle \mu\alpha.c \parallel e \rangle & \to & c[e/\alpha] \end{array}$$

Critical pair:

$$\langle \mu\alpha.c \parallel \tilde{\mu}a.c' \rangle$$

$$\swarrow \qquad\qquad \searrow$$

$$c[\tilde{\mu}x.c'/\alpha] \qquad\qquad c'[\mu\alpha.c/x]$$

The $\lambda\mu\tilde{\mu}$-calculus

The duality of computation
Curien, Herbelin [2000]

Syntax:

Terms	$t ::= V \mid \mu\alpha.c$	(Values)	$V ::= x \mid \lambda x.t$
Contexts	$e ::= E \mid \tilde{\mu}x.c$	(Co-values)	$E ::= \alpha \mid t \cdot e$
Commands	$c ::= \langle t \parallel e \rangle$		

Reduction:

$$\langle \lambda x.t \parallel u \cdot e \rangle \rightarrow \langle u \parallel \tilde{\mu}x.\langle t \parallel e \rangle \rangle$$

$$\langle t \parallel \tilde{\mu}x.c \rangle \rightarrow c[t/x] \qquad t \in \mathcal{T}$$

$$\langle \mu\alpha.c \parallel e \rangle \rightarrow c[e/\alpha] \qquad e \in \mathcal{E}$$

Critical pair:

$$\langle \mu\alpha.c \parallel \tilde{\mu}a.c' \rangle$$

CbV $\swarrow$ $c[\tilde{\mu}x.c'/\alpha]$

CbN $\searrow$ $c'[\mu\alpha.c/x]$

The $\lambda\mu\tilde{\mu}$-calculus

The duality of computation
Curien, Herbelin [2000]

Syntax:

TERMS	$t ::= V \mid \mu\alpha.c$	(Values)	$V ::= x \mid \lambda x.t$
CONTEXTS	$e ::= E \mid \tilde{\mu}x.c$	(Co-values)	$E ::= \alpha \mid t \cdot e$
COMMANDS	$c ::= \langle t \,\|\, e \rangle$		

Typing rules:

$$\frac{\Gamma \vdash t : A \mid \Delta \qquad \Gamma \mid e : A \vdash \Delta}{\langle t \,\|\, e \rangle : (\Gamma \vdash \Delta)}$$

$$\frac{(x : A) \in \Gamma}{\Gamma \vdash x : A \mid \Delta} \qquad \frac{\Gamma, x : A \vdash t : B \mid \Delta}{\Gamma \vdash \lambda x.t : A \to B \mid \Delta} \qquad \frac{c : (\Gamma \vdash \Delta, \alpha : A)}{\Gamma \vdash \mu\alpha.c : A \mid \Delta}$$

$$\frac{(\alpha : A) \in \Delta}{\Gamma \mid \alpha : A \vdash \Delta} \qquad \frac{\Gamma \vdash t : A \mid \Delta \qquad \Gamma \mid e : B \vdash \Delta}{\Gamma \mid t \cdot e : A \to B \vdash \Delta} \qquad \frac{c : (\Gamma, x : A \vdash \Delta)}{\Gamma \mid \tilde{\mu}x.c : A \vdash \Delta}$$

The $\lambda\mu\tilde{\mu}$-calculus

The duality of computation
Curien, Herbelin [2000]

Syntax:

TERMS	$t ::= V \mid \mu\alpha.c$	(Values)	$V ::= x \mid \lambda x.t$
CONTEXTS	$e ::= E \mid \tilde{\mu}x.c$	(Co-values)	$E ::= \alpha \mid t \cdot e$
COMMANDS	$c ::= \langle t \parallel e \rangle$		

Typing rules:

$$\frac{\Gamma \vdash \quad A \mid \Delta \qquad \Gamma \mid \quad A \vdash \Delta}{(\Gamma \vdash \Delta)}$$

$$\frac{A \;\in \Gamma}{\Gamma \vdash \quad A \mid \Delta} \qquad \frac{\Gamma, \quad A \vdash \quad B \mid \Delta}{\Gamma \vdash \quad A \to B \mid \Delta} \qquad \frac{\Gamma \vdash \Delta, \quad A}{\Gamma \vdash \quad A \mid \Delta}$$

$$\frac{A \;\in \Delta}{\Gamma \mid \quad A \vdash \Delta} \qquad \frac{\Gamma \vdash \quad A \mid \Delta \qquad \Gamma \mid \quad B \vdash \Delta}{\Gamma \mid \quad A \to B \vdash \Delta} \qquad \frac{\Gamma, \quad A \vdash \Delta}{\Gamma \mid \quad A \vdash \Delta}$$

The $\lambda\mu\tilde{\mu}$-calculus

The duality of computation
Curien, Herbelin [2000]

Syntax:

TERMS	$t ::= V \mid \mu\alpha.c$	(Values)	$V ::= x \mid \lambda x.t$
CONTEXTS	$e ::= E \mid \tilde{\mu}x.c$	(Co-values)	$E ::= \alpha \mid t \cdot e$
COMMANDS	$c ::= \langle t \parallel e \rangle$		

Typing rules:

$$\frac{\Gamma \vdash t : A \mid \Delta \qquad \Gamma \mid e : A \vdash \Delta}{\langle t \parallel e \rangle : (\Gamma \vdash \Delta)}$$

$$\frac{(x : A) \in \Gamma}{\Gamma \vdash x : A \mid \Delta} \qquad \frac{\Gamma, x : A \vdash t : B \mid \Delta}{\Gamma \vdash \lambda x.t : A \to B \mid \Delta} \qquad \frac{c : (\Gamma \vdash \Delta, \alpha : A)}{\Gamma \vdash \mu\alpha.c : A \mid \Delta}$$

$$\frac{(\alpha : A) \in \Delta}{\Gamma \mid \alpha : A \vdash \Delta} \qquad \frac{\Gamma \vdash t : A \mid \Delta \qquad \Gamma \mid e : B \vdash \Delta}{\Gamma \mid t \cdot e : A \to B \vdash \Delta} \qquad \frac{c : (\Gamma, x : A \vdash \Delta)}{\Gamma \mid \tilde{\mu}x.c : A \vdash \Delta}$$

Call-by-value $\lambda\mu\tilde{\mu}$-calculus

Syntax:

Terms	$t ::= V \mid \mu\alpha.c$	Contexts	$e ::= E \mid \tilde{\mu}x.c$
Values	$V ::= x \mid \lambda x.t$	Co-values	$E ::= \alpha \mid t \cdot e$
	Commands	$c ::= \langle t \| e \rangle$	

Reduction rules:

$$\begin{array}{rcl} \langle \mu\alpha.c \| e \rangle & \rightarrow & c[e/\alpha] \\ \langle V \| \tilde{\mu}x.c \rangle & \rightarrow & c[V/x] \\ \langle \lambda x.t \| u \cdot e \rangle & \rightarrow & \langle u \| \tilde{\mu}x.\langle t \| e \rangle \rangle \end{array}$$

Question

How to define an adequate realizability interpretation?

Call-by-value $\lambda\mu\tilde{\mu}$-calculus

Syntax:

Terms	$t ::= V \mid \mu\alpha.c$	Contexts	$e ::= E \mid \tilde{\mu}x.c$
Values	$V ::= x \mid \lambda x.t$	Co-values	$E ::= \alpha \mid t \cdot e$

Commands $\quad c ::= \langle t \| e \rangle$

Reduction rules:

$$\begin{array}{rcl} \langle \mu\alpha.c \| e \rangle & \to & \mathbf{c[e/\alpha]} \\ \langle V \| \tilde{\mu}x.c \rangle & \to & \mathbf{c[V/x]} \\ \langle \lambda x.t \| u \cdot e \rangle & \to & \langle u \| \tilde{\mu}x.\langle t \| e \rangle\rangle \end{array}$$

Intuitions

Substitutions $\rightsquigarrow \begin{cases} \sigma \Vdash \Gamma, x : A & \triangleq \sigma \Vdash \Gamma \wedge \sigma(x) \in \underline{|A| \cap \textbf{\textit{Values}}} \\ \sigma \Vdash \Delta, \alpha : A & \triangleq \sigma \Vdash \Delta \wedge \sigma(\alpha) \in \|A\| \end{cases}$

Call-by-value $\lambda\mu\tilde{\mu}$-calculus

Syntax:

Terms	$t ::= V \mid \mu\alpha.c$	Contexts	$e ::= E \mid \tilde{\mu}x.c$
Values	$V ::= x \mid \lambda x.t$	Co-values	$E ::= \alpha \mid t \cdot e$

Commands $\quad c ::= \langle t \parallel e \rangle$

Reduction rules:

$$\begin{array}{rcl} \langle \mu\alpha.c \parallel e \rangle & \to & c[e/\alpha] \\ \langle V \parallel \tilde{\mu}x.c \rangle & \to & c[V/x] \\ \langle \lambda x.t \parallel u \cdot e \rangle & \to & \langle u \parallel \tilde{\mu}x.\langle t \parallel e \rangle \rangle \end{array}$$

Intuitions

Substitutions $\rightsquigarrow \begin{cases} \sigma \Vdash \Gamma, x : A & \triangleq \sigma \Vdash \Gamma \wedge \sigma(x) \in \underline{|A|}_V \\ \sigma \Vdash \Delta, \alpha : A & \triangleq \sigma \Vdash \Delta \wedge \sigma(\alpha) \in \|A\| \end{cases}$

Call-by-value $\lambda\mu\tilde{\mu}$-calculus

Syntax:

Terms	$t ::= V \mid \mu\alpha.c$	Contexts	$e ::= E \mid \tilde{\mu}x.c$
Values	$V ::= x \mid \lambda x.t$	Co-values	$E ::= \alpha \mid t \cdot e$

Commands $\quad c ::= \langle t \| e \rangle$

Reduction rules:

✓ $\quad \langle \mu\alpha.c \| e \rangle \rightarrow c[e/\alpha]$

✗ $\quad \langle V \| \tilde{\mu}x.c \rangle \rightarrow c[V/x]$

✗ $\quad \langle \lambda x.t \| u \cdot e \rangle \rightarrow \langle u \| \tilde{\mu}x.\langle t \| e \rangle \rangle$

Intuitions

A realizer should “work” given any opponent...

The unity of semantic artifacts

Terms	$t ::= V \mid \mu\alpha.c$	Contexts	$e ::= E \mid \tilde{\mu}x.c$
Values	$V ::= x \mid \lambda x.t$	Co-values	$E ::= \alpha \mid t \cdot e$

Commands $c ::= \langle t \parallel e \rangle$

Small steps

$$
\begin{array}{lrcl}
t & \langle \mu\alpha.c \parallel e\rangle_t & \rightsquigarrow & c_t[e/\alpha] \\
 & \langle V \parallel e\rangle_t & \rightsquigarrow & \langle V \parallel e\rangle_e \\
e & \langle V \parallel \tilde{\mu}x.c\rangle_e & \rightsquigarrow & c_t[V/x] \\
 & \langle V \parallel E\rangle_e & \rightsquigarrow & \langle V \parallel E\rangle_V \\
V & \langle \lambda x.t \parallel E\rangle_V & \rightsquigarrow & \langle \lambda x.t \parallel E\rangle_E \\
E & \langle \lambda x.t \parallel u \cdot e\rangle_E & \rightsquigarrow & \langle u \parallel \tilde{\mu}x.\langle t \parallel e\rangle\rangle_t
\end{array}
$$

Big steps

$$
\begin{array}{lcl}
\langle \mu\alpha.c \parallel e\rangle & \rightarrow & c[E/\alpha] \\
\langle V \parallel \tilde{\mu}x.c\rangle & \rightarrow & c[V/x] \\
\langle \lambda x.t \parallel u \cdot e\rangle & \rightarrow & \langle u \parallel \tilde{\mu}x.\langle t \parallel e\rangle\rangle
\end{array}
$$

The unity of semantic artifacts

TERMS	$t ::= V \mid \mu\alpha.c$	CONTEXTS	$e ::= E \mid \tilde{\mu}x.c$
VALUES	$V ::= x \mid \lambda x.t$	CO-VALUES	$E ::= \alpha \mid t \cdot e$

COMMANDS $c ::= \langle t \| e \rangle$

Small steps

t:

$$\langle \mu\alpha.c \| e\rangle_t \rightsquigarrow c_t[e/\alpha]$$
$$\langle V \| e\rangle_t \rightsquigarrow \langle V \| e\rangle_e$$

e:

$$\langle V \| \tilde{\mu}x.c\rangle_e \rightsquigarrow c_t[V/x]$$
$$\langle V \| E\rangle_e \rightsquigarrow \langle V \| E\rangle_V$$

V:

$$\langle \lambda x.t \| E\rangle_V \rightsquigarrow \langle \lambda x.t \| E\rangle_E$$

E:

$$\langle \lambda x.t \| u \cdot e\rangle_E \rightsquigarrow \langle u \| \tilde{\mu}x.\langle t \| e\rangle\rangle_t$$

Realizability

$$|A|_t \triangleq \|A\|_e{}^{\perp\!\!\!\perp}$$

$$\|A\|_e \triangleq |A|_V{}^{\perp\!\!\!\perp}$$

$$|A \rightarrow B|_V \triangleq \{\lambda x.t : \forall V \in |A|_V, t[V/x] \in |B|_t\}$$

The unity of semantic artifacts

"The adequacy theorem [...] corresponds to an evaluation procedure."

We get "for free":

Adequacy

If $\sigma \Vdash \Gamma$ and $\sigma \Vdash \Delta$, then for any $⫫$:

1. $\Gamma \vdash t : A \mid \Delta \quad \Rightarrow \quad \sigma(t) \in |A|_t$
2. $\Gamma \vdash V : A \mid \Delta \quad \Rightarrow \quad \sigma(V) \in |A|_V$
3. $\Gamma \mid e : A \vdash \Delta \quad \Rightarrow \quad \sigma(e) \in \|A\|_e$
4. $c : (\Gamma \vdash \Delta) \quad \Rightarrow \quad \sigma(c) \in ⫫$

↪ *With the same method, we also can mechanically derive a typed CPS translation.*

The unity of semantic artifacts

> *"The adequacy theorem [...] corresponds to an evaluation procedure."*

We get "for free":

Adequacy

If $\sigma \Vdash \Gamma$ and $\sigma \Vdash \Delta$, then for any $⊥\!\!⊥$:

1. $\Gamma \vdash t : A \mid \Delta \quad \Rightarrow \quad \sigma(t) \in |A|_t$
2. $\Gamma \vdash V : A \mid \Delta \quad \Rightarrow \quad \sigma(V) \in |A|_V$
3. $\Gamma \mid e : A \vdash \Delta \quad \Rightarrow \quad \sigma(e) \in \|A\|_e$
4. $c : (\Gamma \vdash \Delta) \quad \Rightarrow \quad \sigma(c) \in ⊥\!\!⊥$

↬ *With the same method, we also can mechanically derive a typed CPS translation.*

The unity of semantic artifacts

Normalizing commands

$\perp\!\!\!\perp_{\Downarrow} \triangleq \{c : c \text{ normalizes}\}$ defines a valid pole.

Proof. If $c \to c'$ and c' normalizes, so does c. □

Normalization

For any command c, if $c : \Gamma \vdash \Delta$, then c normalizes.

Proof. By adequacy, any typed command c belongs to the pole $\perp\!\!\!\perp_{\Downarrow}$. □

Soundness

There is no term t such that $\vdash t : \perp\,|$.

Proof. Otherwise, $p \in |\perp|_p = \Pi^{\perp\!\!\!\perp}$ for any pole, absurd ($\perp\!\!\!\perp \triangleq \emptyset$). □

Resumindo

We saw:

- **Classical logic**: interaction terms/contexts
- **Krivine realizability**:
  - interaction player/opponent
  - primitive falsity values + **orthogonality**
- Key property: **adequacy** w.r.t. typing

Killer features

- Normalization / soundness as corollaries
- Very modular
- Compatible with *your* favorite calculus (probably)
- More coming soon!

Resumindo

We saw:

- **Classical logic**: interaction terms/contexts
- **Krivine realizability**:
  - interaction player/opponent
  - primitive falsity values + **orthogonality**
- Key property: **adequacy** w.r.t. typing

Killer features

- Normalization / soundness as corollaries
- Very modular
- Compatible with *your* favorite calculus (probably)
- More coming soon!

Take-away message

The motto

With side-effects come new reasoning principles.

In the next episode

Next week, we shall dwell on:

- **specification problem**

"Who are the realizer of A?"

- **witness extraction**

Spoiler: *it works for Σ_1^0-formulas.*

- connexion with **forcing**

Spoiler: *realizability generalizes forcing!*

- the **algebraic structure** of realizability models

The wonderland of implicative algebras

In the next episode

Next week, we shall dwell on:

- **specification problem**

"Who are the realizer of A?"

- **witness extraction**

Spoiler: *it works for Σ_1^0-formulas.*

- connexion with **forcing**

Spoiler: *realizability generalizes forcing!*

- the **algebraic structure** of realizability models

The wonderland of implicative algebras

In the next episode

Next week, we shall dwell on:

- **specification problem**

"Who are the realizer of A?"

- **witness extraction**

Spoiler: *it works for Σ^0_1-formulas.*

- connexion with **forcing**

Spoiler: *realizability generalizes forcing!*

- the **algebraic structure** of realizability models

The wonderland of implicative algebras

In the next episode

Next week, we shall dwell on:

- **specification problem**

"Who are the realizer of A?"

- **witness extraction**

Spoiler: *it works for Σ_1^0-formulas.*

- connexion with **forcing**

Spoiler: *realizability generalizes forcing!*

- the **algebraic structure** of realizability models

The wonderland of implicative algebras

Questions?