Introduction

λ_c-calculus 2nd-c

2nd-order logic

Realizability interpretation

Realizing axioms

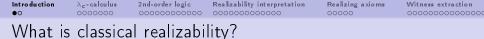
Witness extraction

Computational interpretation of proofs: Classical realizability

Alexandre Miquel



Semantics of proofs and certified mathematics PhD school – April 10th, 2014 – CIRM – Luminy



- Complete reformulation of the principles of Kleene realizability to take into account classical reasoning
 - Based on Griffin's discovery about the connection between classical reasoning and control operators (call/cc)

$$\mathsf{call/cc} : ((A \Rightarrow B) \Rightarrow A) \Rightarrow A \qquad (\mathsf{Peirce's \ law})$$

- Interprets the axiom of dependent choices (DC)
- Initially designed for PA2 (+ DC), but extends to:
 - Higher-order arithmetic ($PA\omega$)
 - Zermelo-Fraenkel set theory (ZF)
 - The calculus of constructions with universes (with classical logic in Prop)
- Deep connections with Cohen forcing

(3rd lecture)

Introduction ⊖●	λ c-calculus 0000000	2nd-order logic 0000000000000	Realizability interpretation	Realizing axioms 00000	Witness extraction
Plan					

- 1 Introduction
- 2 The λ_c -calculus
- Second-order logic
- 4 Realizability interpretation
- **(5)** Realizing the axioms of PA2⁻
- 6 Witness extraction

Introduction	λ _c -calculus ●000000	2nd-order logic 0000000000000	Realizability interpretation	Realizing axioms 00000	Witness extraction
Plan					

1 Introduction

2 The λ_c -calculus

Second-order logic

4 Realizability interpretation

(5) Realizing the axioms of PA2⁻

6 Witness extraction

Introduction	λ _c -calculus 0●00000	2nd-order logic 0000000000000	Realizability interpretation	Realizing axioms 00000	Witness extraction
Terms	, stacks a	nd process	ses		

- Syntax of the language parameterized by
 - A countable set K = {x;...} of instructions, containing at least the instruction x (call/cc)
 - A countable set Π_0 of stack constants (or stack bottoms)

Terms, stacks and processes									
Terms	t, u	::=	x		$\lambda x . t$	tu	κ	k_{π}	$(\kappa \in \mathcal{K})$
Stacks	π,π'	::=	α		$t\cdot\pi$			$(\alpha \in \Pi$	I_0, t closed)
Processes	p,q	::=	t *	π					(t closed)

- A λ -calculus with two kinds of constants:
 - Instructions $\kappa \in \mathcal{K}$, including ∞
 - Continuation constants k_{π} , one for every stack π (generated by ∞)

• Notation: Λ , Π , $\Lambda \star \Pi$ (sets of closed terms / stacks / processes)

Introduction	λ c-calculus 00●0000	2nd-order logic	Realizability interpretation	Realizing axioms 00000	Witness extraction
Proof-lik	ke terms				

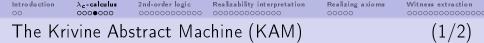
• **Proof-like term** \equiv Term containing no continuation constant

Proof-like terms $t, u ::= x | \lambda x \cdot t | tu | \kappa \quad (\kappa \in \mathcal{K})$

- Idea: All realizers coming from actual proofs are of this form, continuation constants k_{π} are treated as paraproofs
- Notation: PL \equiv set of closed proof-like terms
- Natural numbers encoded as proof-like terms by:

Krivine numerals $\overline{n} \equiv \overline{s}^n \overline{0} \in \mathsf{PL}$ $(n \in \mathsf{IN})$ writing $\overline{0} \equiv \lambda xy \cdot x$ and $\overline{s} \equiv \lambda nxy \cdot y (n \times y)$

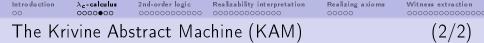
• Note: Krivine numerals \neq Church numerals, but β -equivalent



 We assume that the set Λ ★ Π comes with a preorder p ≻ p' of evaluation satisfying the following rules:

Krivine Abstract Machine (KAM)							
Push	tu \star π	\succ	$t \star u \cdot \pi$				
Grab	$\lambda x.t \star u \cdot \pi$	\succ	$t\{x := u\} \star \pi$				
Save	$cc \star u \cdot \pi$	\succ	$u \star k_{\pi} \cdot \pi$				
Restore	$k_{\pi} \star u \cdot \pi'$	\succ	$u \star \pi$				
(+ reflexivity & tr	ansitivity)						

- Evaluation not defined but axiomatized. The preorder $p \succ p'$ is another parameter of the calculus, just like the sets \mathcal{K} and Π_0
- Extensible machinery: can add extra instructions and rules (We shall see examples later)



 Rules Push and Grab implement weak head β-reduction (call-by-name strategy):

Push	tu $\star \pi$	\succ	$t \star u \cdot \pi$
Grab	λx . t \star u \cdot π	\succ	$t\{x := u\} \star \pi$

• Rules **Save** and **Restore** implement backtracking:

Save	$\mathbf{c} \star \mathbf{u} \cdot \pi$	\succ	$u \star k_{\pi} \cdot \pi$
Restore	$k_{\pi} \star u \cdot \pi'$	\succ	$u \star \pi$

- \bullet Instruction ∞ creates continuation constants $k_{\pi} \colon$

 $\bullet\,$ Continuation constant k_{\pi} restores the saved context π



• The instruction quote

quote
$$\star t \cdot u \cdot \pi \succ u \star \overline{[t]} \cdot \pi$$

where $t \mapsto \lceil t \rceil$ is a fixed bijection from Λ to IN

- Useful to realize the axiom of dependent choices
- The instruction eq

$$eq \star t_1 \cdot t_2 \cdot u \cdot v \cdot \pi \succ \begin{cases} u \star \pi & \text{if } t_1 \equiv t_2 \\ v \star \pi & \text{if } t_1 \not\equiv t_2 \end{cases}$$

1

- Tests syntactic equality $t_1 \equiv t_2$
- Can be implemented using quote
- The instruction ₼ (fork)

$$\pitchfork \star u \cdot v \cdot \pi \succ \begin{cases} u \star \pi \\ v \star \pi \end{cases}$$

- Non deterministic choice operator
- Useful for pedagogy bad for realizability

(collapses to forcing)

[Krivine'03]

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	000000	000000000000	000000000000	00000	000000000000000000000000000000000000000
Examp	es of ext	tra instruc	tions		(2/2)

• The instruction stop:

stop
$$\star \pi \not\succ$$

Stops execution. Final result returned on the stack π

• The instruction print:

print $\star \overline{n} \cdot u \cdot \pi \succ u \star \pi$ (formal specification)

and prints integer n on standard output (informal specification)

- → Displays intermediate results without stopping the machine (poor man's side effect)
- The instruction make_coffee:

 $\mathsf{make_coffee} \star u \cdot \pi \hspace{0.2cm} \succ \hspace{0.2cm} u \star \pi \hspace{0.2cm} + \hspace{0.2cm} \mathsf{makes coffee}$

Introduction 00	λ c-calculus 0000000	2nd-order logic ●೦೦೦೦೦೦೦೦೦೦೦	Realizability interpretation	Realizing axioms 00000	Witness extraction
Plan					

1 Introduction

2 The λ_c -calculus

Second-order logic

4 Realizability interpretation

(5) Realizing the axioms of PA2⁻

6 Witness extraction

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	00000000000000000	000000000000	00000	000000000000000000000000000000000000000
T I I		c / · ·		1 •	

The language of (minimal) second-order logic

- Second-order logic deals with two kinds of objects:
 - 1st-order objects = individuals (i.e. basic objects of the theory)
 - 2nd-order objects = k-ary relations over individuals

First-order terms and formulas

First-order terms	e,e'	::=	$x \mid f(e_1,\ldots,e_k)$
Formulas	А, В		$egin{array}{lll} X(e_1,\ldots,e_k) & & A \Rightarrow B \ orall x A & & orall X A \end{array}$

- Two kinds of variables
 - 1st-order vars: x, y, z, ... (not to be confused with λ -variables!)
 - 2nd-order vars: X, Y, Z, ... of all arities $k \ge 0$

• Two kinds of substitution:

- 1st-order subst.: $e\{x:=e_0\}, A\{x:=e_0\}$ (defined as usual)
- 2nd-order subst.: $A\{X := P_0\}, P\{X := P_0\}$ (postponed)

Introduction	λ c-calculus 0000000	2nd-order logic 00●0000000000	Realizability interpretation	Realizing axioms 00000	Witness extraction		
First-order terms							

• Defined from a first order signature Σ (as usual):

First-order terms $e, e' ::= x | f(e_1, \ldots, e_k)$

• f ranges over k-ary function symbols in Σ

- In what follows we assume that:
 - Each $k\text{-}{\rm ary}$ function symbol f is interpreted in IN by a function $f^{\mathbb{N}} \ : \ {\rm IN}^k \to {\rm IN}$
 - The signature Σ contains a function symbol for every primitive recursive function: 0, s, +, ×, ↑, ...
- Denotation (in IN) of a closed first-order term *e* written [*e*]

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000000000000000000000000000000	000000000000	00000	000000000000000000000000000000000000000
Formul	as				

• Formulas of minimal second-order logic

Formulas
$$A, B ::= X(e_1, \dots, e_k) | A \Rightarrow B$$

 $| \forall x A | \forall X A$

only based on implication and 1st/2nd-order universal quantification

• Other connectives/quantifiers are defined (second-order encodings)

_	_	$\begin{array}{l} \forall Z \ Z \\ A \Rightarrow \bot \end{array}$	(absurdity) (negation)
		$ \begin{array}{l} \forall Z \left((A \Rightarrow B \Rightarrow Z) \Rightarrow Z \right) \\ \forall Z \left((A \Rightarrow Z) \Rightarrow (B \Rightarrow Z) \Rightarrow Z \right) \end{array} $	(conjunction) (disjunction)
		$ \forall Z (\forall x (A(x) \Rightarrow Z) \Rightarrow Z) \forall Z (\forall X (A(X) \Rightarrow Z) \Rightarrow Z) $	(1st-order ∃) (2nd-order ∃)
$e_1 = e_2$	≡	$\forall Z (Z(e_1) \Rightarrow Z(e_2))$	(Leibniz equality)

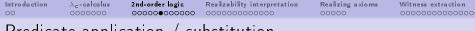
Introduction	λ_{c} - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000000000000000000000000000000	00000000000	00000	000000000000000000000000000000000000000
Predicat	tes				

- 2nd-order variables represent unknown (abstract) relations
- Concrete relations are represented using predicates (syntactic sugar)

Predicates	P,Q	::=	$\hat{x}_1 \cdots \hat{x}_k A$	(of arity <i>k</i>)
------------	-----	-----	--------------------------------	----------------------

• Let
$$P \equiv \hat{x}_1 \cdots \hat{x}_k A$$

- Variables x_1, \ldots, x_n (pairwise \neq) are the arguments of P
- Other free variables of formula A are the parameters of P
- Notation: $FV(P) = FV(A) \setminus \{x_1, \dots, x_k\}$ (free vars = params)
- Predicates are subject to α -conversion $(\hat{x}_i s \text{ treated as binders})$
- 0-ary predicates are formulas



Predicate application / substitution

• Partial/total application of $P \equiv \hat{x}_1 \cdots \hat{x}_k A$ to e_1, \dots, e_ℓ :

$$\begin{split} P(e_1,\ldots,e_\ell) &\equiv \hat{x}_{\ell+1}\cdots\hat{x}_k A\{x_1 := e_1;\ldots;x_\ell := e_\ell\} \qquad (\ell \le k) \\ & \text{where} \quad x_j \notin FV(e_i) \quad \text{for } i \in [1..\ell], j \in [\ell+1..k]) \end{split}$$

Result is a $(k - \ell)$ -ary predicate, and a formula if $k = \ell$

• Every k-ary 2nd-order variable may be viewed as a predicate:

$$X \equiv \hat{x}_1 \cdots \hat{x}_k X(x_1, \ldots, x_k)$$

• Second-order substitution (X, P of same arity) $(X(e_1, \ldots, e_k))\{X := P\} \equiv P(e_1, \ldots, e_k)$ • In a formula: $A\{X := P\}$ • In a predicate: $Q\{X := P\}$

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	0000000000000	000000000000	00000	000000000000000000000000000000000000000
Unary g	oredicate	es as sets			

• Unary predicates represent sets of individuals Syntactic sugar: $\{x : A\} \equiv \hat{x}A, e \in P \equiv P(e)$

Example: The set IN of Dedekind numerals

 $\mathbb{N} \equiv \{x : \forall Z (0 \in Z \Rightarrow \forall y (y \in Z \Rightarrow s(y) \in Z) \Rightarrow x \in Z\}$

• Relativized quantifications:

۰

$$\begin{aligned} (\forall x \in P) A(x) &\equiv \forall x (x \in P \Rightarrow A(x)) \\ (\exists x \in P) A(x) &\equiv \forall Z (\forall x (x \in P \Rightarrow A(x) \Rightarrow Z) \Rightarrow Z) \\ &\Leftrightarrow \exists x (x \in P \land A(x)) \end{aligned}$$

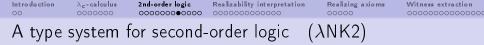
• Inclusion and extensional equality:

$$P = Q \equiv \forall x (x \in P \Rightarrow x \in Q)$$

$$P = Q \equiv \forall x (x \in P \Leftrightarrow x \in Q)$$

Set constructors: $P \cup Q \equiv \{x : x \in P \lor x \in Q\}$ (etc.)

 $P \subset O = \forall \mathbf{y} (\mathbf{y} \in P \rightarrow \mathbf{y} \in O)$



- Use proof-like terms as Curry-style proof terms Represent the computational contents of classical proofs
- Typing judgement:

$$\underline{\mathbf{x_1}:A_1,\ldots,\mathbf{x_n}:A_n} \vdash t:B$$

typing context **F**

Typing rules

$$\overline{\Gamma \vdash \mathbf{x} : A} (x:A) \in \Gamma$$

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

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

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash t : \forall X A} \times \notin FV(\Gamma) \qquad \frac{\Gamma \vdash t : \forall X A}{\Gamma \vdash t : A\{x := P\}}$$

$$\overline{\Gamma \vdash c : ((A \Rightarrow B) \Rightarrow A) \Rightarrow A}$$

	Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction		
	00	0000000	00000000000000	000000000000	00000	000000000000000000000000000000000000000		
Typing examples								

• Intuitionistic principes:

$$\begin{array}{rcl} \text{pair} &\equiv& \lambda xyz . z \, x \, y &:& \forall X \,\forall Y \, (X \Rightarrow Y \Rightarrow X \wedge Y) \\ \text{fst} &\equiv& \lambda z . \, z \, (\lambda xy . x) &:& \forall X \,\forall Y \, (X \wedge Y \Rightarrow X) \\ \text{snd} &\equiv& \lambda z . \, z \, (\lambda xy . y) &:& \forall X \,\forall Y \, (X \wedge Y \Rightarrow Y) \\ \text{refl} &\equiv& \lambda z . \, z &:& \forall x \, (x = x) \\ \text{trans} &\equiv& \lambda xyz . \, y \, (x \, z) &:& \forall x \,\forall y \,\forall z \, (x = y \Rightarrow y = z \Rightarrow x = z) \end{array}$$

• Excluded middle, double negation elimination:

$$\begin{array}{rcl} \mathsf{left} &\equiv& \lambda xuv \cdot u \, x &:& \forall X \,\forall Y \, (X \Rightarrow X \lor Y) \\ \mathsf{right} &\equiv& \lambda yuv \cdot v \, y &:& \forall X \,\forall Y \, (Y \Rightarrow X \lor Y) \\ \mathsf{EM} &\equiv& \alpha \, (\lambda k \, . \, \mathsf{right} \, (\lambda x \, . \, k \, (\mathsf{left} \, x))) &:& \forall X \, (X \lor \neg X) \\ \mathsf{DNE} &\equiv& \lambda z \, . \, \alpha \, (\lambda k \, . \, z \, k) &:& \forall X \, (\neg \neg X \Rightarrow X) \end{array}$$

• De Morgan laws:

$$\begin{array}{rcl} \lambda zy \, . \, z \, (\lambda x \, . \, yx) & : & \exists x \, A(x) \, \Rightarrow \, \neg \forall x \, \neg A(x) \\ \lambda zy \, . \, \mathfrak{c} \, (\lambda k \, . \, z \, (\lambda x \, . \, k \, (y \, x))) & : & \neg \forall x \, \neg A(x) \, \Rightarrow \, \exists x \, A(x) \end{array}$$



System λ NK2 defines provability in classical 2nd-order *logic* (NK2). For classical 2nd-order *arithmetic* (PA2), add the following axioms:

• Defining equations of primitive recursive functions:

$$\begin{aligned} \forall x (x + 0 = x) & \forall x \forall y (x + s(y) = s(x + y)) \\ \forall x (x \times 0 = 0) & \forall x \forall y (x \times s(y) = x \times y + x) \end{aligned}$$
 (etc.)

• Peano 3rd and 4th axioms:

• The induction axiom:

$$\begin{array}{ll} \mathsf{Ind} & \equiv & \forall x \ (x \in \mathsf{IN}) \\ \Leftrightarrow & \forall Z \ [0 \in Z \Rightarrow \forall y \ (y \in Z \Rightarrow s(y) \in Z) \Rightarrow \forall x \ (x \in Z)] \end{array}$$

Introduction	λ_c -calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000	000000000000	00000	000000000000000000000000000000000000000
The i	nduction	axiom			

• Problem: The induction axiom is not realizable!

$$\begin{array}{ll} \mathsf{Ind} & \equiv & \forall x \ (x \in \mathbb{N}) \\ \Leftrightarrow & \forall Z \ [0 \in Z \Rightarrow \forall y \ (y \in Z \Rightarrow s(y) \in Z) \Rightarrow \forall x \ (x \in Z)] \end{array}$$

• Solution: Relativize all 1st-order quantifications to IN:

Non relativized		Relativized	
$\forall x A(x)$	\rightsquigarrow	$(\forall x \in \mathbb{N}) A(x)$ $\forall x (x \in \mathbb{N} \Rightarrow A(x))$	
$\exists x \ A(x) \\ \forall Z \ (\forall x \ (A(x) \Rightarrow Z) \Rightarrow Z)$	$\sim \rightarrow$	$(\exists x \in \mathbb{N}) A(x) \\ \forall Z (\forall x (x \in \mathbb{N} \Rightarrow A(x) \Rightarrow Z) \Rightarrow Z)$	

Theorem

If $PA2 \vdash A$, then $PA2 - Ind \vdash A^{\mathbb{N}}$

 $(A^{\mathbb{N}} \equiv A \text{ relativized to } \mathbb{N})$

Intro duction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	00000000000	000000000000	00000	000000000000000000000000000000000000000
6			<u> </u>		

Computational contents of relativization

Intuition:

$$\begin{array}{rcl} (\forall x \in \mathbb{N}) \ A(x) & \equiv & \forall x \ (x \in \mathbb{N} \Rightarrow A(x)) \\ & \approx & (\Pi x \in \mathsf{nat}) \ A(x) \end{array} \tag{Coq, Agda}$$

• Recall: $x \in \mathbb{N} \equiv \forall Z [Z(0) \Rightarrow \forall y (Z(y) \Rightarrow Z(s(y))) \Rightarrow Z(x)]$

$$\begin{split} \overline{0} &\equiv \lambda zf \cdot fz &: 0 \in \mathbb{N} \\ \overline{s} &\equiv \lambda nzf \cdot f(n z f) &: (\forall x \in \mathbb{N}) s(x) \in \mathbb{N} \\ \overline{n} &\equiv \overline{s}^{n} \overline{0} &: n \in \mathbb{N} \\ \text{plus} &\equiv \lambda nm \cdot m n \overline{s} &: (\forall x, y \in \mathbb{N}) x + y \in \mathbb{N} \\ \text{mult} &\equiv \lambda nm \cdot m \overline{0} (\lambda p \cdot \text{plus } p n) &: (\forall x, y \in \mathbb{N}) x \times y \in \mathbb{N} \\ (\text{etc.}) \end{split}$$

Introduction	λ c-calculus 0000000	2nd-order logic 0000000000000	Realizability interpretation 000000000000	Realizing axioms	Witness extraction
Plan					

- 1 Introduction
- 2 The λ_c -calculus
- Second-order logic
- 4 Realizability interpretation
- **(5)** Realizing the axioms of PA2⁻
- 6 Witness extraction



- Intuitions:
 - term = "proof" / stack = "counter-proof"
 - process = "contradiction" (slogan: never trust a classical realizer!)
- ullet Classical realizability model parameterized by a pole ${oldsymbol \bot}$
 - = set of processes closed under anti-evaluation (or saturated)

 $\text{ If } p\succ p' \quad \text{and} \quad p'\in \mathbb{L}, \ \ \text{then} \quad p\in \mathbb{L} \\$

- Each formula A is interpreted as two sets:
 - A set of stacks ||A|| (falsity value)
 - A set of terms |A| (truth value)
- Falsity value ||A|| defined by induction on A (negative interpretation)
- Truth value |A| defined by orthogonality: $|A| = ||A||^{\perp} = \{t \in \Lambda : \forall \pi \in ||A|| \ t \star \pi \in \perp\}$

Introduction	λ_{c} - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction		
00	0000000	000000000000	00000000000	00000	000000000000000000000000000000000000000		
Architecture of the realizability model							

- The realizability model $\mathscr{M}_{\mathbb{L}}$ is defined from:
 - The full standard model *M* of PA2: the ground model (but we could take any model *M* of PA2 as well)
 - An instance $(\mathcal{K}, \Pi_0, \succ)$ of the λ_c -calculus
 - A saturated set of processes $\bot\!\!\!\bot \subseteq \Lambda \star \Pi$ (the pole)
- Architecture:
 - First-order terms/variables interpreted as natural numbers $n \in \mathbb{N}$
 - Formulas interpreted as falsity values $S \in \mathfrak{P}(\Pi)$
 - k-ary 2nd-order variables (and k-ary predicates) interpreted as falsity functions $F : \mathbb{N}^k \to \mathfrak{P}(\Pi)$.

Formulas with parameters $A, B ::= \cdots | \dot{F}(e_1, \dots, e_k)$

Add a predicate constant \dot{F} for every falsity function $\dot{F}: \mathbb{N}^k \to \mathfrak{P}(\Pi)$

Interpreting closed formulas with parameters

Let A be a closed formula (with parameters)

• Falsity value ||A|| defined by induction on A:

$$\begin{aligned} \|\dot{F}(e_1,\ldots,e_n)\| &= F(\llbracket e_1 \rrbracket,\ldots,\llbracket e_n \rrbracket) \\ \|A \Rightarrow B\| &= |A| \cdot \|B\| = \{t \cdot \pi : t \in |A|, \ \pi \in \|B\|\} \\ \|\forall x \ A\| &= \bigcup_{n \in \mathbb{N}} \|A\{x := n\}\| \\ \|\forall X \ A\| &= \bigcup_{F : \mathbb{N}^k \to \mathfrak{P}(\Pi)} \|A\{X := \dot{F}\}\| \end{aligned}$$

• Truth value |A| defined by orthogonality:

$$|A| = ||A||^{\perp} = \{t \in \Lambda : \forall \pi \in ||A|| \quad t \star \pi \in \bot\}$$

Introduction	λ c-calculus 0000000	2nd-order logic	Realizability interpretation	Realizing axioms 00000	Witness extraction
The real	izability	relation			

Falsity value ||A|| and truth value |A| depend on the pole \bot \rightsquigarrow write them (sometimes) $||A||_{\bot}$ and $|A|_{\bot}$ to recall the dependency

Realizability relations							
$t \Vdash A \equiv$	$t\in A _{\perp}$	(Realizability w.r.t. \bot)					
$t \Vdash A \equiv$	$\forall \bot\!\!\!\bot \ t \in A _{\bot\!\!\!\bot}$	(Universal realizability)					



Fundamental idea: The computational behavior of a term determines the formula(s) it realizes:

Example 1: A closed term *t* is identity-like if:

 $t \star u \cdot \pi \succ u \star \pi$ for all $u \in \Lambda, \pi \in \Pi$

Proposition

If t is identity-like, then $t \Vdash \forall X (X \Rightarrow X)$

Proof: Exercise! (Remark: converse implication holds - exercise!)

- Examples of identity-like terms:
 - $\lambda x . x$, $(\lambda x . x) (\lambda x . x)$, etc.
 - $\lambda x . \alpha \lambda k . x$, $\lambda x . \alpha \lambda k . k x$, $\lambda x . \alpha \lambda k . k x (\delta \delta)$, etc.
 - λx . marshal $x \lambda n$. unmarshal $n (\lambda z . z)$



$$\begin{array}{cccc} \mathbf{cc} & \star & t \cdot \pi & \succ & t & \star & \mathsf{k}_{\pi} \cdot \pi \\ \mathbf{k}_{\pi} & \star & t \cdot \pi' & \succ & t & \star & \pi \end{array}$$

• "Typing"
$$k_{\pi}$$
: $k_{\pi} \star t \cdot \pi' \succ t \star \pi$

Lemma If $\pi \in ||A||$, then $k_{\pi} \Vdash A \Rightarrow B$ (B any) Proof: Exercise • "Typing" α : $\alpha \star t \cdot \pi \succ t \star k_{\pi} \cdot \pi$ Proposition (Peirce's law) $\alpha \parallel \vdash ((A \Rightarrow B) \Rightarrow A) \Rightarrow A$

Proof: Exercise

Introduction	λ c-calculus 0000000	2nd-order logic 0000000000000	Realizability interpretation	Realizing axioms 00000	Witness extraction
Adequa	су				(1/2)

Aim: Prove the theorem of adequacy:

- t: A (in the sense of λ NK2) implies $t \Vdash A$ (in the sense of realizability)
- Closing typing judgments $x_1: A_1, \ldots, x_n: A_n \vdash t: A$
 - We close logical objects (1st-order terms, formulas, predicates) using semantic objects (natural numbers, falsity values, falsity functions)
 - We close proof-terms using realizers

Definition (Valuations)

• A valuation is a function ρ such that:

- $\rho(x) \in \mathbb{N}_{p}$
- $\rho(X) : \mathbb{N}^k \to \mathfrak{P}(\Pi)$

for each 1st-order variable *x* for each 2nd-order variable *X* of arity *k*

2 Closure of A with ρ written $A[\rho]$

(formula with parameters)

Introduction 00	λ_{c} -calculus	2nd-order logic 0000000000000	Realizability interpretation	Realizing axioms 00000	Witness extraction
Adequad	cy				(2/2)

Definition (Adequate judgment, adequate rule)

Given a fixed pole \bot :

• A judgment $x_1 : A_1, \ldots, x_n : A_n \vdash t : A$ is adequate if for every valuation ρ and for all $u_1 \Vdash A_1[\rho], \ldots, u_n \Vdash A_n[\rho]$ we have:

$$t\{x_1 := u_1, \ldots, x_n := u_n\} \Vdash A[\rho]$$

 A typing rule is adequate if it preserves the property of adequacy (from the premises to the conclusion of the rule)

Theorem

- All typing rules of λ NK2 are adequate
- ② All derivable judgments of λ NK2 are adequate

Corollary: If $\vdash t : A$ (A closed formula), then $t \Vdash A$

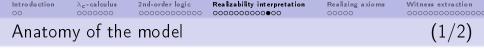
Introduction	λ_{c} - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000	0000000000000	00000	000000000000000000000000000000000000000
Extendir	ng adequ	uacy to su	btyping		

Definition (Adequate subtyp	ing j	judgment)	
Judgment $A \leq B$ adequate	≡	$\ B[ho]\ \subseteq \ A[ho]\ $	(for all valuations $ ho)$

Remark: Implies: $|A[\rho]| \subseteq |B[\rho]|$ (for all ρ), but strictly stronger

Adequate typing/subtyping rules

• Example: $\underbrace{\forall X \forall Y (((X \Rightarrow Y) \Rightarrow X) \Rightarrow X)}_{\text{Peirce's law}} \leq \underbrace{\forall X (\neg \neg X \Rightarrow X)}_{\text{DNE}}$



• Denotation of universal quantification:

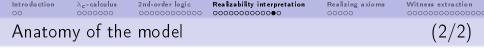
Falsity value:
$$\|\forall x A\| = \bigcup_{n \in \mathbb{N}} \|A\{x := n\}\|$$
 (by definition)

Truth value:
$$|\forall x A| = \bigcap_{n \in \mathbb{N}} |A\{x := n\}|$$
 (by orthogonality)

(and similarly for 2nd-order universal quantification)

• Denotation of implication:

- Falsity value: $||A \Rightarrow B|| = |A| \cdot ||B||$ (by definition)Truth value: $|A \Rightarrow B| \subseteq |A| \rightarrow |B|$ (by orthogonality)writing $|A| \rightarrow |B| = \{t \in \Lambda : \forall u \in |A| \ tu \in |B|\}$ (realizability arrow)
 - $\begin{array}{l} \textcircled{0} \quad \text{Converse inclusion does not hold in general, unless } \bot \text{ closed under Push} \\ \hline{0} \quad \text{In all cases:} \quad \text{If } t \in |A| \to |B|, \quad \text{then } \lambda x \cdot tx \in |A \Rightarrow B| \qquad (\eta\text{-expansion}) \end{array}$



- Falsity value $\|A\|$ and truth value |A| depend on the pole \bot
 - \rightsquigarrow write them $\|A\|_{\mathbb{L}}$ and $|A|_{\mathbb{L}}$ to recall the dependecy
- Degenerate case: $\bot\!\!\!\bot = \varnothing$
 - $\bullet\,$ Truth values can take only two values: $\,\,\, \ensuremath{\varnothing} \,$ and $\Lambda\,$
 - Classical realizability simply mimics the Tarski interpretation:

Degenerated interpretation

$$|A| = \begin{cases} \Lambda & \text{if } \mathscr{M} \models A \\ \varnothing & \text{if } \mathscr{M} \not\models A \end{cases}$$

- Non degenerate cases: $\bot\!\!\!\bot \neq \varnothing$
 - Every truth value |A| is inhabited:

If $t_0 \star \pi_0 \in \bot$, then $k_{\pi_0} t_0 \in |A|$ for all A (paraproof)

 $(\in PL)$

We shall only consider realizers that are proof-like terms

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000	00000000000	00000	000000000000000000000000000000000000000
Provab	ility, univ	/ersal reali	zability and tri	uth	

• From what precedes:

(by a proof-like term)

(in the standard model)

→ Universal realizability: an intermediate notion between provability and truth

Beware!

Intuitionistic proofs of A	\subset	Classical proofs of A
\cap		\cap
Intuitionistic realizers of A	⊈ ⊉	Classical realizers of A

Introduction 00	λ _c -calculus 0000000	2nd-order logic 0000000000000	Realizability interpretation	Realizing axioms ●0000	Witness extraction
Plan					

- 1 Introduction
- 2 The λ_c -calculus
- Second-order logic
- 4 Realizability interpretation
- **(5)** Realizing the axioms of PA2⁻
- 6 Witness extraction



• Defining equations of primitive recursive functions:

$$\begin{aligned} \forall x (x + 0 = x) & \forall x \forall y (x + s(y) = s(x + y)) \\ \forall x (x \times 0 = 0) & \forall x \forall y (x \times s(y) = x \times y + x) \end{aligned}$$
 (etc.)

• Peano 3rd and 4th axioms:

The induction axiom:

Ind
$$\equiv \forall x (x \in \mathbb{N})$$

 $\Leftrightarrow \forall Z [0 \in Z \Rightarrow \forall y (y \in Z \Rightarrow s(y) \in Z) \Rightarrow \forall x (x \in Z)]$

• **Beware!** Since induction is not realizable (in general), we work in $PA2^- = PA2 - Ind$, relativizing all 1st-order \forall/\exists to IN

Introduction 00	λ c-calculus 0000000	2nd-order logic 0000000000000	Realizability interpretation	Realizing axioms 00●00	Witness extraction
Realizing	g equalit	ties			

• Equality between individuals defined by:

$$e_1 = e_2 \equiv \forall Z \left(Z(e_1) \Rightarrow Z(e_2) \right)$$
 (Leibniz equality)

(and a pole \bot)

Denotation of Leibniz equality

Given two closed 1st-order terms e1, e2

$$\|e_1 = e_2\| = \begin{cases} \|\mathbf{1}\| = \{t \cdot \pi : (t \star \pi) \in \mathbb{L}\} & \text{if } \llbracket e_1 \rrbracket = \llbracket e_2 \rrbracket \\ \|\top \Rightarrow \bot\| = \Lambda \cdot \Pi & \text{if } \llbracket e_1 \rrbracket \neq \llbracket e_2 \rrbracket \end{cases}$$

where $\mathbf{1} \equiv orall Z (Z \Rightarrow Z)$ and $\top \equiv \dot{\varnothing}$

- Intuitions:
 - A realizer of a true equality (in the model) behaves as the identity function λz . z
 - A realizer of a false equality (in the model) behaves as a point of backtrack (breakpoint)

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000	000000000000	00000	000000000000000000
Realizin	g Peano	axioms			

Coroll	Corollary 1 (Realizing true equations)						
lf	$\mathscr{M} \models \forall \vec{x} (e_1(\vec{x}) = e_2(\vec{x}))$	(truth in the ground model)					
then	$\lambda z \cdot z \Vdash \forall \vec{x} (e_1(\vec{x}) = e_2(\vec{x}))$	(universal realizability)					

Corollary 2

All the defining equations of the arithmetic function symbols (+, ×, \uparrow , etc.) are realized by $\lambda z . z$

Corollary 3 (Realizing Peano axioms)

$$\begin{array}{rcl} \lambda z \, . \, z & \Vdash & \forall x \, \forall y \, (s(x) = s(y) \Rightarrow x = y) \\ \lambda z \, . \, z \, (\lambda w \, . \, w) & \Vdash & \forall x \, \neg (s(x) = 0) \end{array}$$

Remark: Corollary 3 generalizes to all the Horn formulas that are true in the ground model (using trivial realizers)

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000	000000000000	00000	000000000000000000000000000000000000000
Progra	n extrac	tion			

Extracting a program from a proof in PA2

If $PA2 \vdash A$, then there is $t \in PL$ such that $t \parallel \vdash A^{\mathbb{N}}$ $(A^{\mathbb{N}}$ obtained from A by relativizing all 1st-order quantifications to \mathbb{N})

In practice:

- Only apply the adequacy theorem to the computationally relevant parts of the proof
- For the computationally irrelevant parts (i.e. Horn formulas), use 'default realizers' \rightsquigarrow realizer optimization
- Example 1: $\lambda nmz \cdot z \Vdash (\forall x, y \in \mathbb{N}) x + y = y + x$
- Example 2: Fermat's last theorem¹

 $(\forall x, y, z, n \in \mathbb{N}) (x \ge 1 \Rightarrow y \ge 1 \Rightarrow n \ge 3 \Rightarrow x^n + y^n \neq z^n)$

1 realized by $\lambda xyznuvw . u(v(w I))$

Introduction	λ _c -calculus 0000000	2nd-order logic 000000000000	Realizability interpretation	Realizing axioms 00000	Witness extraction •000000000000000000000000000000000000
Plan					

- 1 Introduction
- 2 The λ_c -calculus
- Second-order logic
- 4 Realizability interpretation
- **(5)** Realizing the axioms of PA2⁻
- 6 Witness extraction

Introduction	λ c-calculus 0000000	2nd-order logic 0000000000000	Realizability interpretation	Realizing axioms 00000	Witness extraction	00
Some	problems	of classica	al realizability			
0	The specif	ication probl	em			
		ormula A, chara computational	cterize its universal re behavior	ealizers		
			Specifying Peirce's	<i>law</i> [Guillermo	-Miquel'12]	
3	Witness e>	traction from	n classical realizers	5	(cf below)	
3	Realizabilit	ty algebras +	Cohen forcing	(3	3rd lecture)	
		Deelizehilit	· · · · · · · · · · · · · · · · · · ·	a wall andar ID	DZ doda oznati	

Realizability algebras: a program to well-order IR [Krivine'11] Forcing as a program transformation [Miquel'11]

Models induced by classical realizability

What are the interesting formulas that are realized in \mathcal{M}_{\perp} that are not already true in the ground model \mathcal{M} ?

Realizability algebras II: new models of ZF + DC [Krivine'12]

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000	000000000000	00000	000000000000000000000000000000000000000
The pro	blem of	witness e	xtraction		

• **Problem:** Extract a witness from a universal realizer (or a proof) $t_0 \Vdash (\exists x \in \mathbb{N}) A(x)$

i.e. some $n \in \mathbb{N}$ such that A(n) is true

• This is not always possible!

 $t_0 \quad \Vdash \quad (\exists x \in \mathbb{N}) ((x = 1 \land C) \lor (x = 0 \land \neg C))$

(C = Continuum hypothesis, Goldbach's conjecture, etc.)

- Two possible compromises:
 - Intuitionistic logic: restrict the shape of the realizer t₀ (by only keeping intuitionistic reasoning principles)
 - Classical logic: restrict the shape of the formula A(x)(typically: Δ_0^0 -formulas)

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000	00000000000	00000	000000000000000000000000000000000000000
Storage					

• The call-by-value implication:

FormulasA, B::= $(e\} \Rightarrow A$ Semantics: $||\{e\} \Rightarrow A|| = \{\overline{n} \cdot \pi : n = [\![e]\!], \pi \in |\![A|]\!\}$ • From the definition: $e \in \mathbb{N} \Rightarrow A \leq \{e\} \Rightarrow A$ so that: $\mathbb{I} \Vdash \forall x \forall Z ((x \in \mathbb{N} \Rightarrow Z) \Rightarrow (\{x\} \Rightarrow Z)))$ (direct implication)Definition (Storage operator)

A storage operator is a closed proof-like term *M* such that:

 $M \Vdash \forall x \forall Z ((\{x\} \Rightarrow Z) \Rightarrow (x \in \mathbb{N} \Rightarrow Z))$ (converse implication)

• They exist, for instance: $M \equiv \lambda fn \cdot nf (\lambda hx \cdot h(\overline{s}x))\overline{0}$

• **Conclusion**: $e \in \mathbb{N} \Rightarrow A$ and $\{e\} \Rightarrow A$ interchangeables

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000	000000000000	00000	000000000000000000000000000000000000000
Storage					

Intuitively, a storage operator

$$M \quad \Vdash \quad \forall x \,\forall Z \, ((\{x\} \Rightarrow Z) \Rightarrow (x \in \mathbb{N} \Rightarrow Z))$$

is a proof-like term that is intended to be applied to

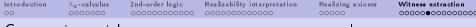
- a function f specified only on totally evaluated numerals (i.e. intuitionistic integers)
- a classical integer $t \Vdash n \in {\sf I\!N}$ (n arbitrary)

and that evaluates (or "smoothes") the classical integer t into a value of the form \overline{n} before passing this value to f

• Alternative point of view:

$$M \quad \Vdash \quad \forall x \left(\{x\} \Rightarrow A(x) \right) \ \Rightarrow \ \left(\forall x \in \mathbb{N} \right) A(x)$$

A property that holds for all values (i.e. intuitionistic integers) also holds for all classical integers



Computing with storage operators: an example

• Given a *k*-ary function *f* , we let:

$$\begin{aligned} \mathsf{Total}(f) &\equiv (\forall x_1 \in \mathsf{IN}) \cdots (\forall x_k \in \mathsf{IN}) (f(x_1, \dots, x_k) \in \mathsf{IN}) \\ \mathsf{Comput}(f) &\equiv \forall x_1 \cdots \forall x_k \, \forall Z \ [\{x_1\} \Rightarrow \dots \Rightarrow \{x_k\} \Rightarrow \\ (\{f(x_1, \dots, x_k)\} \Rightarrow Z) \Rightarrow Z] \end{aligned}$$

Theorem (Specification of the formula Comput(f))

For all $t \in \Lambda$, the following assertions are equivalent:

2 t computes f: for all $(n_1, \ldots, n_k) \in \mathbb{N}^k$, $u \in \Lambda$, $\pi \in \Pi$:

$$t \star \overline{n}_1 \cdots \overline{n}_k \cdot u \cdot \pi \succ u \star \overline{f(n_1, \dots, n_k)} \cdot \pi$$

• Using a storage operator *M*, we can build proof-like terms such that:

$$\begin{array}{lll} \xi_k & \Vdash & \mathsf{Total}(f) & \Rightarrow & \mathsf{Comput}(f) \\ \xi'_k & \Vdash & \mathsf{Comput}(f) & \Rightarrow & \mathsf{Total}(f) \end{array}$$



• A classical realizer $t_0 \parallel \mid (\exists x \in \mathbb{N}) A(x)$ always evaluates to a pair witness/justification

Naive extraction

If $t_0 \Vdash (\exists x \in \mathbb{N}) A(x)$, then there are $n \in \mathbb{N}$ and $u \in \Lambda$ such that

 $t_0 \star M(\lambda xy . \operatorname{stop} x y) \cdot \pi \succ \operatorname{stop} \star \overline{n} \cdot u \cdot \pi$

(w.r.t. a particular pole ⊥L)

- But $n \in \mathbb{N}$ might be a false witness because the justification $u \Vdash A(n)$ is cheating! (i.e. u might contain hidden continuations)
- In the case where t_0 comes from an intuitionistic proof, extracted witness $n \in \mathbb{N}$ is always correct



Extraction in the Σ_1^0 -case (+ display intermediate results)

If
$$t_0 \Vdash (\exists x \in \mathbb{N}) (f(x) = 0)$$
, then

 $t_0 \star M(\lambda xy . \operatorname{print} x y (\operatorname{stop} x)) \cdot \pi \succ \operatorname{stop} \star \overline{n} \cdot \pi$

for some $n \in \mathbb{N}$ such that f(n) = 0

- Storage operator M used to evaluate 1st component
- 2nd component (y) used as a breakpoint (Relies on the particular structure of equality realizers)
- Holds independently from the instruction set
- Supports any representation of numerals (one have to implement the storage operator M accordingly)



Definition (conditional refutation)

 $r_A \in \Lambda$ is a conditional refutation of the predicate A(x) if

For all $n \in \mathbb{N}$ such that $\mathscr{M} \not\models A(n)$: $r_A \overline{n} \Vdash \neg A(n)$

• Such a conditional refutation can be constructed for every predicate A(x) of 1st-order arithmetic

This result is a consequence of the:

Theorem [Krivine-Miquey]

For every formula A of 1st-order arithmetic, there exists a proof-like term t_A such that:

If $\mathscr{M} \models A$ then $t_A \Vdash A$



The Kamikaze extraction method

Let

 \circ r_A a conditional refutation of the predicate A(x)

Then the process

 $t_0 \star M(\lambda xy . \operatorname{print} x(r_A x y)) \cdot \pi$

displays a correct witness after finitely many evaluation steps

• **Remark:** No correctness invariant is ensured as soon as the (first) correct witness has been displayed!

After, everything may happen: crash, infinite loop, displaying incorrect witnesses, etc.



Extend the machine with the following instructions:

• For every integer $n \in \mathbb{N}$, an instruction $\hat{n} \in \mathcal{K}$ with no evaluation rule (i.e. inert constant).

Intuition: $\hat{n} \star \pi \succ$ segmentation fault

• An instruction null with the rules:

$$\operatorname{null} \star \widehat{n} \cdot u \cdot v \cdot \pi \quad \succ \quad \begin{cases} u \star \pi & \text{if } n = 0 \\ v \star \pi & \text{otherwise} \end{cases}$$

• Instructions *f* with the rules:

 $\check{f} \star \widehat{n}_1 \cdots \widehat{n}_k \cdot u \cdot \pi \succ u \star \widehat{m} \cdot \pi$ where $m = f(n_1, \dots, n_k)$

for all the usual arithmetic operations

Introduction	λ_c - calculus	2nd-order logic	Realizability interpretation	Realizing axioms	Witness extraction
00	0000000	000000000000	000000000000	00000	000000000000000000000000000000000000000
Primitiv	(2/2)				

• Call-by-value implication, yet another definition:

Formulas $A, B ::= \cdots | [e] \Rightarrow A$

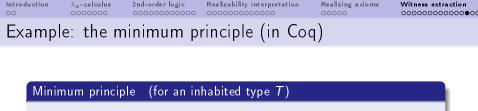
Semantics: $\|[e] \Rightarrow A\| = \{\widehat{n} \cdot \pi : n = [\![e]\!], \pi \in \|A\|\}$

• Redefining the set of natural numbers: $\mathbb{N}' \equiv \{x : \forall Z (([e] \Rightarrow Z) \Rightarrow Z)\}$

box $\equiv \lambda x r.r x$ ||+ $\forall x ([x] \Rightarrow x \in \mathbb{N}')$ box \widehat{n} ||+ $n \in \mathbb{N}'$ $\lambda n. n \lambda x. \check{s} x \text{ box}$ ||+ $(\forall x \in \mathbb{N}') s(x) \in \mathbb{N}'$ $\lambda nm. n \lambda x. m \lambda y. (\check{+}) x y \text{ box}$ ||+ $(\forall x, y \in \mathbb{N}') x + y \in \mathbb{N}'$

 $\operatorname{rec_cbv} \equiv \lambda z_0 z_s \cdot \mathbf{Y} \lambda xr \cdot \operatorname{null} x z_0 (\operatorname{pred} x \lambda y \cdot z_s y (r y))$ $\Vdash \forall Z [Z(0) \Rightarrow \forall y ([y] \Rightarrow Z(y) \Rightarrow Z(s(y))) \Rightarrow \forall x ([x] \Rightarrow Z(x))]$ $\operatorname{rec} \equiv \lambda z_0 z_s n \cdot n \lambda x \cdot \operatorname{rec_cbv} z_0 (\lambda yz \cdot z_s (\operatorname{box} y) z) x$ $\Vdash \forall Z [Z(0) \Rightarrow (\forall y \in \mathbb{N}') (Z(y) \Rightarrow Z(s(y))) \Rightarrow (\forall x \in \mathbb{N}') Z(x)]$

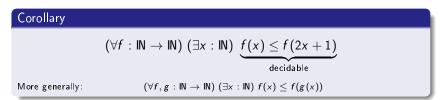
• Conclusion: $\Vdash \forall x (x \in \mathbb{N}' \Leftrightarrow x \in \mathbb{N})$



$$(\forall f: T \to \mathbb{N}) (\exists x: T) \underbrace{(\forall y: T) f(x) \leq f(y)}_{undecidable}$$

Proof: Reductio ad absurdum + course-by-value induction.

• Remark: No intuitionistic proof (oracle)



Proof. Take the point x given by the minimum principle.

	•	0000000000	00000	00000000000000000000000000000000000000					
Krivine's	realizability <i>vs</i> the L	.RS <i>R</i> -tran	slation (1	/2)					
	 Krivine's realizability can seen as the composition of the Lafont- Reus-Streicher (LRS) <i>R</i>-translation with Kleene's realizability CPS \circ Krivine = Kleene \circ LRS [Oliva-Streicher'08] 								
The dic	The dictionary								
	Classical realizability LRS <i>R</i> -translation								
	Pole ⊥	Return f	ormula <i>R</i>						
	Falsity value $\ A\ $	Negative translation A^{\perp}							
	$\ A \Rightarrow B\ = A \cdot \ B\ $	$(A \Rightarrow B)^{\perp} \equiv A^{LRS} \wedge B^{\perp}$							
	Truth value $ A = \ A\ ^{ota}$	$A^{LRS} \equiv$	$A^{\perp} \Rightarrow R$						

Realizability interpretation

Realizing axioms

Witness extraction

000

Introduction

λ - - calculus

2nd-order logic

• Through the CPS translation, Krivine's extraction method in the $\Sigma_1^0\text{-}case$ is exactly Friedman's trick [Miquel'10]



Beware of reductionism!

- It only holds for pure classical reasoning (extra instructions are not taken into account)
- Classical realizers are easier to understand than their CPS-translations (and more efficient...)
- Classical realizability is more than Kleene's realizability composed with the Lafont-Reus-Streicher *R*-translation!

An image:

$$2H_2 + O_2 \longrightarrow 2H_2O$$

but can we deduce the properties of water from the ones of H_2 and O_2 ?