Generic approaches to effectful realizability models

Liron Cohen Ariel Grunfeld Dominik Kirst Étienne Miquey Ross Tate





Journées CHoCoLa

Lyon, 15/10/25

What are *realizability models*?

Realizability

1. Provides **models** for theories $\not\vdash$ *(such as HA2 / HOL / ZF / ...)*

Tarski

$A \mapsto |A| \in \mathbb{B}$ ← Boolean algebra

(intuition: level of truthness)

Realizability

$A \mapsto \{t : t \Vdash A\}$

(intuition: programs whose computational behavior is guided by A)

2. a for analyzing programs computational behavior

Realizability

1. Provides **models** for theories ↬ *(such as HA2 / HOL / ZF / ...)*

Tarski

$A \mapsto |A| \in \mathbb{B}$ ← Boolean algebra

(intuition: level of truthness)

Realizability

$A \mapsto \{t : t \Vdash A\}$

(intuition: programs whose computational behavior is guided by A)

2. a **tool** for analyzing programs computational behavior

Realizability, a 3-steps recipe

1. formulas (*a.k.a. types*)
   - ↬ *simple types,* 2^{nd} *– order logic , ZF, ...*
2. a computational system (*a.k.a. your favorite calculus*)
   - ↬ *some* λ *– calculus , a combinatory algebra, PCF, etc.*
3. formulas interpretation

Adequacy

If $\mathfrak{p} : (\Gamma \vdash A)$ and $\sigma \Vdash \Gamma$ then $\sigma(\mathfrak{p}^*) \in |A|$.

Realizability, a 3-steps recipe

next slide

1. formulas (*a.k.a. types*)
   ↬ *simple types,* $2^{nd}-$ *order logic* , *ZF, ...*

2. a computational system (*a.k.a. your favorite calculus*)
   ↬ *some* $\lambda-$ *calculus* , *a combinatory algebra, PCF, etc.*

3. formulas interpretation (*a.k.a. truth values*)
   ↬ $|A| = \{t \in \Lambda : t \Vdash A\}$

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash \Gamma$ then $\sigma(t) \in |A|$.

A simple realizability interpretation

Types & terms: (excerpt)

$$
\begin{array}{lrcl}
1^{st}\text{-order exp.} & e & ::= & x \mid 0 \mid S(e) \mid f(e_1, \ldots, e_n) \\
\textit{Formulas} & A, B & ::= & \mathrm{Nat}(e) \mid X(e_1, \ldots, e_n) \mid A \to B \mid \ldots \\
 & & & \mid \forall x.A \mid \exists x.A \mid \forall X.A \mid \exists X.A \\
\textit{Terms} & t, u & ::= & x \mid 0 \mid \mathbf{succ} \mid \mathbf{rec} \mid \lambda x.t \mid t\, u \mid \ldots
\end{array}
$$

where $f : \mathbb{N}^n \to \mathbb{N}$ is any arithmetical function.

Typing rules: (excerpt)

$$\frac{}{\Gamma \vdash 0 : \mathrm{Nat}(0)} \qquad \frac{}{\Gamma \vdash \mathbf{rec} : \forall Z.Z(0) \to (\forall^{\mathbb{N}} y.(Z(y) \to Z(S(y)))) \to \forall^{\mathbb{N}} x.Z(x)}$$

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x\,.\,t : A \to B} \qquad \frac{\Gamma \vdash t : A \to B \qquad \Gamma \vdash u : A}{\Gamma \vdash t\, u : B} \qquad \frac{\Gamma \vdash t : A[x := n]}{\Gamma \vdash t : \exists x.A}$$

$$\frac{\Gamma \vdash t : \forall x.A}{\Gamma \vdash t : A[x := n]} \qquad \frac{\Gamma \vdash t : A \quad x \notin FV(\Gamma)}{\Gamma \vdash t : \forall x.A} \qquad \frac{\Gamma \vdash t : A[X(x_1, \ldots, x_n) := B]}{\Gamma \vdash t : \exists X.A}$$

A simple realizability interpretation

Types & terms: (excerpt)

$$
\begin{array}{lrl}
1^{st}\text{-order exp.} & e & ::= x \mid 0 \mid S(e) \mid f(e_1, \ldots, e_n) \\
\textit{Formulas} & A, B & ::= \mathrm{Nat}(e) \mid X(e_1, \ldots, e_n) \mid A \to B \mid \ldots \\
 & & \quad \mid \forall x.A \mid \exists x.A \mid \forall X.A \mid \exists X.A \\
\textit{Terms} & t, u & ::= x \mid 0 \mid \mathbf{succ} \mid \mathbf{rec} \mid \lambda x.t \mid t\, u \mid \ldots
\end{array}
$$

where $f : \mathbb{N}^n \to \mathbb{N}$ *is any arithmetical function.*

Typing rules: (excerpt)

$$\ldots$$

Reductions:

$$\overline{(\lambda x.t)u \triangleright_\beta t[u/x]} \qquad \overline{\mathbf{rec}\ u_0\, u_1\, (\mathbf{succ}\, t) \triangleright_\beta u_1\, t\, (\mathbf{rec}\ u_0\, u_1\, t)} \qquad \ldots$$

A simple realizability interpretation

Realizability interpretation:

$$
\begin{array}{rcl}
|\mathrm{Nat}(e)|_\rho & \triangleq & \{t \in \Lambda : t \triangleright^* \mathbf{succ}^n 0, \text{ where } n = [\![e]\!]_\rho\} \\
|X(e_1, \ldots, e_n)|_\rho & \triangleq & \rho(X)([\![e_1]\!]_\rho, \ldots, [\![e_n]\!]_\rho) \\
|\ A \ \to \ B\ |_\rho & \triangleq & \{t \in \Lambda : \ \forall u \in |A|_\rho \ .(\ t\,u \in |B|_\rho\ )\} \\
|\ \forall x\ .A|_\rho & \triangleq & \bigcap_{n \in \mathbb{N}} |A|_{\rho, x \leftarrow n} \\
|\ \exists x\ .A|_\rho & \triangleq & \bigcup_{n \in \mathbb{N}} |A|_{\rho, x \leftarrow n} \\
|\forall X.A|_\rho & \triangleq & \bigcap_{F : \mathbb{N}^k \to \mathbf{SAT}} |A|_{\rho, X \leftarrow F} \\
|\exists X.A|_\rho & \triangleq & \bigcup_{F : \mathbb{N}^k \to \mathbf{SAT}} |A|_{\rho, X \leftarrow F}
\end{array}
$$

Adequacy

If $\Gamma \vdash t : A$ and $\sigma \Vdash \Gamma$ then $\sigma(t) \in |A|$.

Key ideas:

- realizers

A simple realizability interpretation

Realizability interpretation:

$$
\begin{array}{rcl}
|\mathrm{Nat}(e)|_\rho & \triangleq & \{t \in \Lambda : t \triangleright^* \mathbf{succ}^n 0, \text{ where } n = [\![e]\!]_\rho\} \\
|X(e_1, \ldots, e_n)|_\rho & \triangleq & \rho(X)([\![e_1]\!]_\rho, \ldots, [\![e_n]\!]_\rho) \\
|\ A \to B\ |_\rho & \triangleq & \{t \in \Lambda : \forall u \in |A|_\rho\ .(\ t\,u \in |B|_\rho\ )\} \\
|\ \forall x\ .A|_\rho & \triangleq & \bigcap_{n \in \mathbf{N}} |A|_{\rho, x \leftarrow n} \\
|\ \exists x\ .A|_\rho & \triangleq & \bigcup_{n \in \mathbf{N}} |A|_{\rho, x \leftarrow n} \\
|\forall X.A|_\rho & \triangleq & \bigcap_{F : \mathbf{N}^k \to \mathbf{SAT}} |A|_{\rho, X \leftarrow F} \\
|\exists X.A|_\rho & \triangleq & \bigcup_{F : \mathbf{N}^k \to \mathbf{SAT}} |A|_{\rho, X \leftarrow F}
\end{array}
$$

Key ideas:

- realizers *compute*
- realizers *defend the validity* of their formula
- truth values are *saturated*: $t \triangleright^* t' \ \wedge\ t' \in |A| \ \Longrightarrow\ t \in |A|$

A simple realizability interpretation

Realizability interpretation:

$$
\begin{array}{rcl}
|\mathrm{Nat}(e)|_\rho & \triangleq & \{t \in \Lambda : t \triangleright^* \mathbf{succ}^n 0, \text{ where } n = [\![e]\!]_\rho\} \\
|X(e_1, \ldots, e_n)|_\rho & \triangleq & \rho(X)([\![e_1]\!]_\rho, \ldots, [\![e_n]\!]_\rho) \\
|\ A \to B\ |_\rho & \triangleq & \{t \in \Lambda : \forall u \in |A|_\rho\ .(\ t\,u \in |B|_\rho\ )\} \\
|\ \forall x\ .A|_\rho & \triangleq & \bigcap_{n \in \mathbb{N}} |A|_{\rho, x \leftarrow n} \\
|\ \exists x\ .A|_\rho & \triangleq & \bigcup_{n \in \mathbb{N}} |A|_{\rho, x \leftarrow n} \\
|\forall X.A|_\rho & \triangleq & \bigcap_{F : \mathbb{N}^k \to \mathbf{SAT}} |A|_{\rho, X \leftarrow F} \\
|\exists X.A|_\rho & \triangleq & \bigcup_{F : \mathbb{N}^k \to \mathbf{SAT}} |A|_{\rho, X \leftarrow F}
\end{array}
$$

Key ideas:

- realizers *compute*
- realizers *defend the validity* of their formula
- truth values are *saturated*: $t \triangleright^* t' \ \wedge\ t' \in |A| \ \Longrightarrow\ t \in |A|$

A simple realizability interpretation

Realizability interpretation:

$$
\begin{array}{rcl}
|\mathrm{Nat}(e)|_\rho & \triangleq & \{t \in \Lambda : t \triangleright^* \mathbf{succ}^n 0, \text{ where } n = [\![e]\!]_\rho\} \\
|X(e_1, \ldots, e_n)|_\rho & \triangleq & \rho(X)([\![e_1]\!]_\rho, \ldots, [\![e_n]\!]_\rho) \\
|\ A \ \to \ B\ |_\rho & \triangleq & \{t \in \Lambda : \ \forall u \in |A|_\rho \ .(\ t\,u \in |B|_\rho\ )\} \\
|\ \forall x\ .A|_\rho & \triangleq & \bigcap_{n \in \mathbf{N}} |A|_{\rho, x \leftarrow n} \\
|\ \exists x\ .A|_\rho & \triangleq & \bigcup_{n \in \mathbf{N}} |A|_{\rho, x \leftarrow n} \\
|\forall X.A|_\rho & \triangleq & \bigcap_{F : \mathbf{N}^k \to \ \mathbf{SAT}} |A|_{\rho, X \leftarrow F} \\
|\exists X.A|_\rho & \triangleq & \bigcup_{F : \mathbf{N}^k \to \ \mathbf{SAT}} |A|_{\rho, X \leftarrow F}
\end{array}
$$

Key ideas:

- realizers *compute*
- realizers *defend the validity* of their formula
- truth values are *saturated*: $t \triangleright^* t' \ \wedge \ t' \in |A| \ \Longrightarrow \ t \in |A|$

A simple realizability interpretation

Realizability interpretation:

$$
\begin{array}{rcl}
|\mathrm{Nat}(e)|_\rho & \triangleq & \{t \in \Lambda : t \triangleright^* \mathbf{succ}^n 0, \text{ where } n = [\![e]\!]_\rho\} \\
|X(e_1, \ldots, e_n)|_\rho & \triangleq & \rho(X)([\![e_1]\!]_\rho, \ldots, [\![e_n]\!]_\rho) \\
|\ A \ \to \ B\ |_\rho & \triangleq & \{t \in \Lambda : \ \forall u \in |A|_\rho \ .(\ t\,u \in |B|_\rho\ )\} \\
|\ \forall x\ .A|_\rho & \triangleq & \bigcap_{n \in \mathbb{N}} |A|_{\rho, x \leftarrow n} \\
|\ \exists x\ .A|_\rho & \triangleq & \bigcup_{n \in \mathbb{N}} |A|_{\rho, x \leftarrow n} \\
|\forall X.A|_\rho & \triangleq & \bigcap_{F : \mathbb{N}^k \to \mathbf{SAT}} |A|_{\rho, X \leftarrow F} \\
|\exists X.A|_\rho & \triangleq & \bigcup_{F : \mathbb{N}^k \to \mathbf{SAT}} |A|_{\rho, X \leftarrow F}
\end{array}
$$

Key ideas:

- realizers *compute*
- realizers *defend the validity* of their formula
- truth values are *saturated*: $t \triangleright^* t' \ \wedge \ t' \in |A| \ \implies \ t \in |A|$

Formal definition

?

Formal definition?

iki/Realizability

Realizability

4 languages

Article Talk Read Edit View history

From Wikipedia, the free encyclopedia

In mathematical logic, **realizability** is a collection of methods in proof theory used to study constructive proofs and extract additional information from them.[1] Formulas from a formal theory are "realized" by objects, known as "realizers", in a way that knowledge of the realizer gives knowledge about the truth of the formula. There are many variations of realizability; exactly which class of formulas is studied and which objects are realizers differ from one variation to another.

Realizability can be seen as a formalization of the BHK interpretation of intuitionistic logic; in realizability the notion of "proof" (which is left undefined in the BHK interpretation) is replaced with a formal notion of "realizer". Most variants of realizability begin with a theorem that any statement that is provable in the formal system being studied is realizable. The realizer, however, usually gives more information about the formula than a formal proof would directly provide.

Beyond giving insight into intuitionistic provability, realizability can be applied to prove the disjunction and existence properties for intuitionistic theories and to extract programs from proofs, as in proof mining. It is also related to topos theory via the realizability topos.

Example: Kleene's 1945-realizability [edit]

Kleene's original version of realizability uses natural numbers as realizers for formulas in Heyting arithmetic. A few pieces of notation are required: first, an ordered pair (n,m) is treated as a single number using a fixed primitive recursive pairing function; second, for each natural number n, φ_n is the computable function with index n. The following clauses are used to define a relation "n realizes A"

Formal definition?

wiki/Boolean_algebra_(structure)

Definition [edit]

A **Boolean algebra** is a six-tuple consisting of a set A, equipped with two binary operations $\wedge$ (called "meet" or "and"), $\vee$ (called "join" or "or"), a unary operation $\neg$ (called "complement" or "not") and two elements 0 and 1 in A (called "bottom" and "top", or "least" and "greatest" element, also denoted by the symbols $\bot$ and $\top$, respectively), such that for all elements a, b and c of A, the following axioms hold:[2]

$a \vee (b \vee c) = (a \vee b) \vee c$	$a \wedge (b \wedge c) = (a \wedge b) \wedge c$	associativity
$a \vee b = b \vee a$	$a \wedge b = b \wedge a$	commutativity
$a \vee (a \wedge b) = a$	$a \wedge (a \vee b) = a$	absorption
$a \vee 0 = a$	$a \wedge 1 = a$	identity
$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$	$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$	distributivity
$a \vee \neg a = 1$	$a \wedge \neg a = 0$	complements

Note, however, that the absorption law and even the associativity law can be excluded from the set of axioms as they can be derived from the other axioms (see Proven properties).

A Boolean algebra with only one element is called a **trivial Boolean algebra** or a **degenerate Boolean algebra**. (In older works, some authors required 0 and 1 to be *distinct* elements in order to exclude this case.)[*citation needed*]

It follows from the last three pairs of axioms above (identity, distributivity and complements), or from the absorption axiom, that

$a = b \wedge a$ if and only if $a \vee b = b$.

The relation $\leq$ defined by $a \leq b$ if these equivalent conditions hold, is a partial order with least element 0 and greatest element 1. The meet $a \wedge b$ and the join $a \vee b$ of two elements coincide with their infimum and supremum, respectively, with respect to $\leq$.

The first four pairs of axioms constitute a definition of a bounded lattice.

It follows from the first five pairs of axioms that any complement is unique.

The set of axioms is self-dual in the sense that if one exchanges $\vee$ with $\wedge$ and 0 with 1 in an axiom, the result is again an axiom.

Formal definition?

tlab.org/nlab/show/realizability+topos

Home Page All Pages Latest Revisior

realizability topos

Contents

Context

Topos Theory

Constructivism, Realizability, Computability

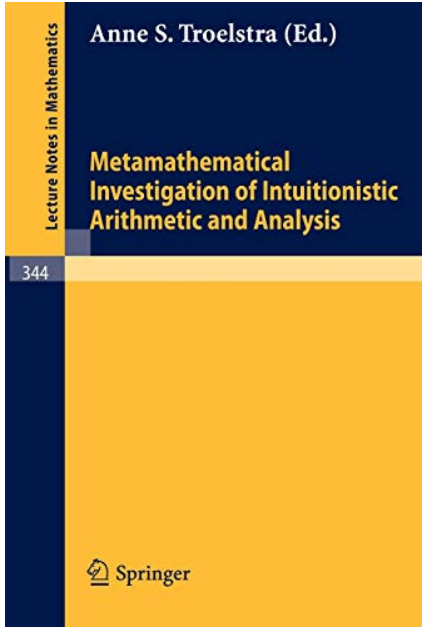
1. Idea

A realizability topos is a topos which embodies the realizability interpretation of intuitionistic number theory (due to Kleene) as part of its internal logic. Realizability toposes form an important class of elementary toposes that are not Grothendieck toposes, and don't even have a geometric morphism to Set.

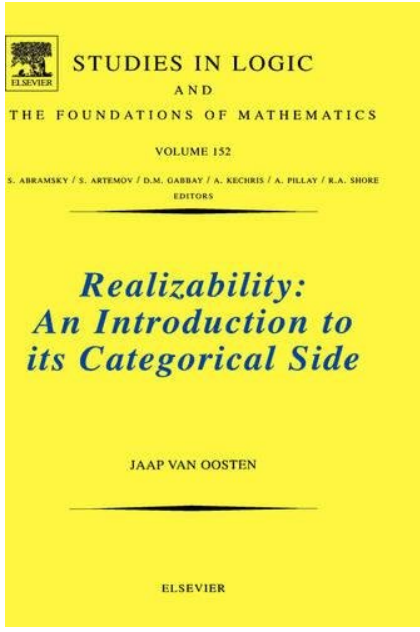
The input datum for forming a realizability topos is a partial combinatory algebra, or PCA.

- When the PCA is Kleene's first algebra $\mathcal{K}_1$, the resulting topos is called the effective topos $\mathrm{RT}(\mathcal{K}_1)$.

Formal definition?



Formal definition?



Topoi, Triposes and PCAs

Pitts' PhD (80s)

PCA → Tripos realizability → Topos model

A realizability model

$\mathcal{M} \vDash A \quad \Leftrightarrow \quad \exists t.t \Vdash A$

Goal #1

Introduce an intermediate structure that connects the logical and computational aspects

Topoi, Triposes and PCAs

Pitts' PhD (80s)

PCA → Tripos (realizability) → Topos (model)

- A functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathbf{HpA}$ with:
  - quantifiers
  - computability w. substitutions
  - generic predicate

Goal #1

Introduce an intermediate structure that connects the logical and computational aspects

Topoi, Triposes and PCAs

Pitts' PhD (80s)

PCA → Tripos (realizability) → Topos (model)

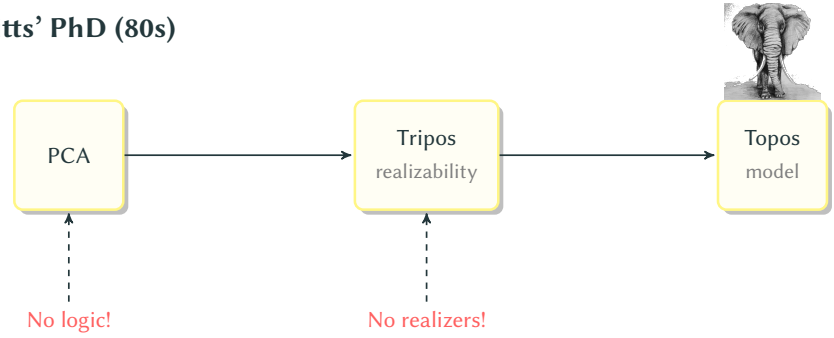
Codes: a set C
Application: a partial binary operator $\cdot : C \times C \rightharpoonup C$ with the functional completeness property.

Goal #1

Introduce an intermediate structure that connects the logical and computational aspects

Topoi, Triposes and PCAs

Pitts' PhD (80s)

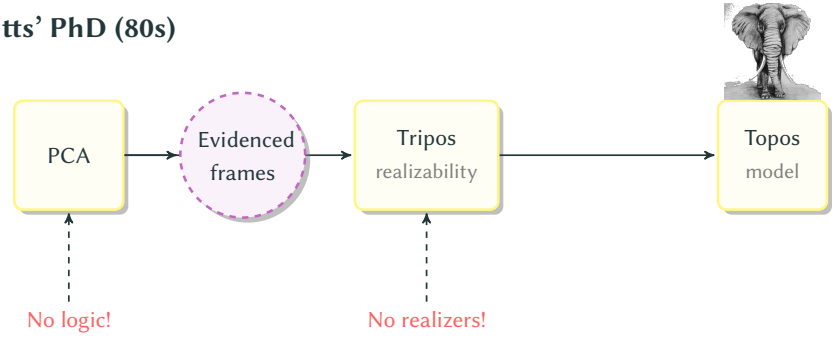


Goal #1

Introduce an intermediate structure that connects the logical and computational aspects

This Talk: Goal #1

Pitts' PhD (80s)



Goal #1

Introduce an intermediate structure that connects the logical and computational aspects

Computational Choices Matter

With side-effects come new reasoning principles:

- exceptions ∼ Markov's principle
- control operators ∼ classical logic
- `quote` instruction ∼ dependent choice
- memoization ∼ dependent choice
- monotonic memory ∼ Cohen's forcing
- monotonic memory ∼ nonstandard analysis
- ...

But PCAs can only support non-termination!

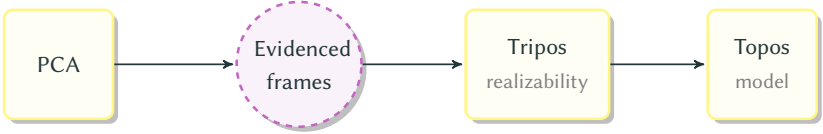
Computational Choices Matter

With side-effects come new reasoning principles:

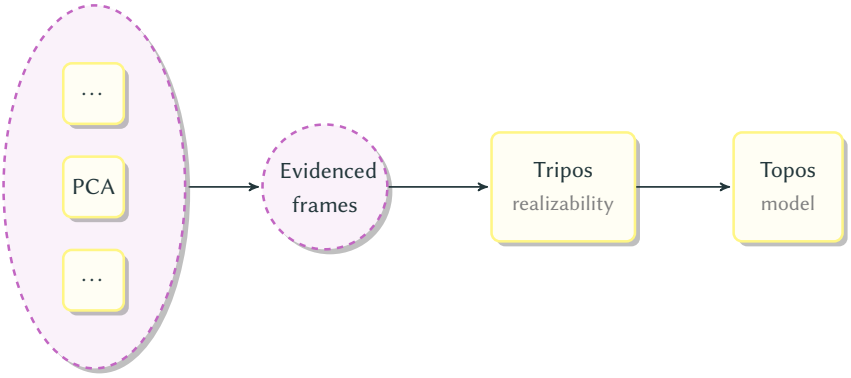
- exceptions ~ Markov's principle
- control operators ~ classical logic
- `quote` instruction ~ dependent choice
- memoization ~ dependent choice
- monotonic memory ~ Cohen's forcing
- monotonic memory ~ nonstandard analysis
- ...

But PCAs can only support non-termination!

This Talk: Goal #2



This Talk: Goal #2



Goal #2

Smooth integration of useful computational effects

This Talk: Goal #3

Question

What about the syntactic approach to realizability?

Realizability, the historical recipe

Partial Combinatory Algebras

Definition - PAS

A **Partial Applicative Structure (PAS)** $(\mathbb{A}, \cdot)$ is given by:

- a set $\mathbb{A}$ of "codes",
- a partial application $\cdot : \mathbb{A} \times \mathbb{A} \rightharpoonup \mathbb{A}$.

Notations:

- $c_f \cdot c_a \downarrow c_r$ means that c_r is the (successful) result of applying c_f to c_a,
- $c_f \cdot c_a \downarrow$ means that there exists some $c_r \in \mathbb{A}$ such that $c_f \cdot c_a \downarrow c_r$.

Definition - PCA

A **Partial Combinatory Algebra** is a PAS that is "functionally complete".

↦ *For every expression e with n free variables, there is a code $\langle \lambda_n.e \rangle \in \mathbb{A}$ such that:*

$$\lambda_{n+1}.e \cdot c_a \downarrow \lambda_n.e\{c_a\} \qquad \text{and} \qquad \lambda_0.e \cdot c_a \downarrow c_r \iff e\{c_a\} \downarrow c_r$$

Partial Combinatory Algebras

Definition - PAS

A **Partial Applicative Structure (PAS)** $(\mathbb{A}, \cdot)$ is given by:

- a set $\mathbb{A}$ of "codes",
- a partial application $\cdot : \mathbb{A} \times \mathbb{A} \rightharpoonup \mathbb{A}$.

Notations:

- $c_f \cdot c_a \downarrow c_r$ means that c_r is the (successful) result of applying c_f to c_a,
- $c_f \cdot c_a \downarrow$ means that there exists some $c_r \in \mathbb{A}$ such that $c_f \cdot c_a \downarrow c_r$.

Definition - PCA

A **Partial Combinatory Algebra** is a PAS that is "functionally complete".

↬ *For every expression e with n free variables, there is a code $\langle \lambda_n.e \rangle \in \mathbb{A}$ such that:*

$$\lambda_{n+1}.e \cdot c_a \downarrow \lambda_n.e\{c_a\} \qquad \text{and} \qquad \lambda_0.e \cdot c_a \downarrow c_r \iff e\{c_a\} \downarrow c_r$$

Partial Combinatory Algebras: examples

Examples of PCAs:

- Kleene's first model $\mathcal{K}_1$

$$\mathbb{N} \text{ with partial recursive application } a, b \mapsto \varphi_a(b),$$

- Kleene's first model $\mathcal{K}_2$

$$\mathbb{N}^{\mathbb{N}} \text{ with application } F : \mathbb{N}^{\mathbb{N}} \times \mathbb{N}^{\mathbb{N}} \to \mathbb{N}^{\mathbb{N}},$$

- Scott graph model: $\mathcal{P}(\mathbb{N})$ as a graph:

$$\text{graph} : F \mapsto \{\langle n, m\rangle \mid m \in F(e_n)\} \qquad \in (\mathcal{P}(\mathbb{N}) \to \mathcal{P}(\mathbb{N})) \to \mathcal{P}(\mathbb{N})$$
$$\text{fun} : A \mapsto B \mapsto \{\langle m \mid \exists n. e_n \subseteq B \wedge \langle n, m\rangle \in A\} \qquad \in \mathcal{P}(\mathbb{N}) \to \mathcal{P}(\mathbb{N}) \to \mathcal{P}(\mathbb{N})$$

- Domains: $\boxed{D \simeq D \to D}$
- Any model $\mathcal{M}$ of Peano Arithmetic:

$$(ab = c) \triangleq \mathcal{M} \models \exists y.\, T(a, b, y) \wedge U(y) = c$$

Historical factory: $\mathcal{P}(\mathbb{A})$-valued predicate

Predicates

A $\mathcal{P}(\mathbb{A})$-valued predicate on a set X is a function $\varphi : X \to \mathcal{P}(\mathbb{A})$.

$a \in \varphi(x)$: a is a realizer of $\varphi(x)$. We can define:

- $(\varphi \Rightarrow \psi)(x) = \{a \in \mathbb{A} \mid \forall b \in \varphi(x),\ ab\downarrow \text{ and } ab \in \psi(x)\}$,
- $(\varphi \wedge \psi)(x) = \{\langle a, b\rangle \in \mathbb{A} \mid a \in \varphi(x),\ b \in \psi(x)\}$,
- $(\varphi \vee \psi)(x) = \{\iota_1(a) \in \mathbb{A} \mid a \in \varphi(x)\} \cup \{\iota_2(b) \in \mathbb{A} \mid b \in \psi(x)\}$.

Entailment

We pose $\varphi \preccurlyeq \psi \triangleq \exists a \in \mathbb{A},\ \forall x \in X,\ \forall b \in \varphi(x),\ ab\downarrow \text{ and } ab \in \psi(x)$.

Proposition

$(\mathcal{P}(\mathbb{A}), \preccurlyeq)$ is a pre-Heyting algebra.

Historical factory: $\mathcal{P}(\mathbb{A})$-valued predicate

Predicates

A $\mathcal{P}(\mathbb{A})$-valued predicate on a set X is a function $\varphi : X \to \mathcal{P}(\mathbb{A})$.

$a \in \varphi(x)$: a is a realizer of $\varphi(x)$. We can define:

- $(\varphi \Rightarrow \psi)(x) = \{a \in \mathbb{A} \mid \forall b \in \varphi(x),\ ab\downarrow \text{ and } ab \in \psi(x)\}$,
- $(\varphi \wedge \psi)(x) = \{\langle a, b\rangle \in \mathbb{A} \mid a \in \varphi(x),\ b \in \psi(x)\}$,
- $(\varphi \vee \psi)(x) = \{\iota_1(a) \in \mathbb{A} \mid a \in \varphi(x)\} \cup \{\iota_2(b) \in \mathbb{A} \mid b \in \psi(x)\}$.

Entailment

We pose $\varphi \preccurlyeq \psi \triangleq \exists a \in \mathbb{A},\ \forall x \in X,\ \forall b \in \varphi(x),\ ab\downarrow \text{ and } ab \in \psi(x)$.

Proposition

$(\mathcal{P}(\mathbb{A}), \preccurlyeq)$ is a pre-Heyting algebra.

Historical factory: $\mathcal{P}(\mathbb{A})$-valued predicate

Predicates

A $\mathcal{P}(\mathbb{A})$-valued predicate on a set X is a function $\varphi : X \to \mathcal{P}(\mathbb{A})$.

$a \in \varphi(x)$: a is a realizer of $\varphi(x)$. We can define:

- $(\varphi \Rightarrow \psi)(x) = \{a \in \mathbb{A} \mid \forall b \in \varphi(x),\ ab \downarrow \text{ and } ab \in \psi(x)\}$,
- $(\varphi \wedge \psi)(x) = \{\langle a, b\rangle \in \mathbb{A} \mid a \in \varphi(x),\ b \in \psi(x)\}$,
- $(\varphi \vee \psi)(x) = \{\iota_1(a) \in \mathbb{A} \mid a \in \varphi(x)\} \cup \{\iota_2(b) \in \mathbb{A} \mid b \in \psi(x)\}$.

Entailment

We pose $\varphi \preccurlyeq \psi \triangleq \exists a \in \mathbb{A},\ \forall x \in X,\ \forall b \in \varphi(x),\ ab \downarrow \text{ and } ab \in \psi(x)$.

Proposition

$(\mathcal{P}(\mathbb{A}), \preccurlyeq)$ is a pre-Heyting algebra.

Historical factory: $\mathcal{P}(\mathbb{A})$-valued predicate

Predicates

A $\mathcal{P}(\mathbb{A})$-valued predicate on a set X is a function $\varphi : X \to \mathcal{P}(\mathbb{A})$.

$a \in \varphi(x)$: a is a realizer of $\varphi(x)$. We can define:

- $(\varphi \Rightarrow \psi)(x) = \{a \in \mathbb{A} \mid \forall b \in \varphi(x),\ ab\downarrow \text{ and } ab \in \psi(x)\}$,
- $(\varphi \wedge \psi)(x) = \{\langle a, b\rangle \in \mathbb{A} \mid a \in \varphi(x),\ b \in \psi(x)\}$,
- $(\varphi \vee \psi)(x) = \{\iota_1(a) \in \mathbb{A} \mid a \in \varphi(x)\} \cup \{\iota_2(b) \in \mathbb{A} \mid b \in \psi(x)\}$.

Entailment

We pose $\varphi \preccurlyeq \psi \triangleq \exists a \in \mathbb{A},\ \forall x \in X,\ \forall b \in \varphi(x),\ ab\downarrow \text{ and } ab \in \psi(x)$.

Proposition

$(\mathcal{P}(\mathbb{A}), \preccurlyeq)$ is a pre-Heyting algebra.

Models and triposes

Realizability model

$$\mathcal{M} \vDash A \quad \Leftrightarrow \quad \exists t.t \Vdash A$$

Categorically speaking

a *topos*

Triposes

PCA / λ-calculus / ...

Tripos realizability

Topos model

Models and triposes

Realizability model

$$\mathcal{M} \vDash A \quad \Leftrightarrow \quad \exists t.t \Vdash A$$

Categorically speaking

a *topos*

Triposes

Models and triposes

A *tripos* is a functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathbf{HpA}$ s.t.:

↦ *Intuition* $\mathcal{T}(\Gamma)$: predicates $\varphi(\vec{x})$ $\quad s^*(\underbrace{\varphi}_{\in \mathcal{T}(\Gamma')}) \triangleq \varphi(\underbrace{s}_{s:\Gamma\to\Gamma'}(\vec{y}))$

Quantifiers. Any s^* has left/right adjoints $\coprod_s / \prod_s : \mathcal{T}(\Gamma) \to \mathcal{T}(\Gamma')$:

$$\varphi \le s^*(\psi) \quad \Leftrightarrow \quad \coprod_s(\varphi) \le \psi \qquad\qquad s^*(\varphi) \le \psi \quad \Leftrightarrow \quad \varphi \le \prod_s(\psi)$$

Compatibility w. substitutions.

$$\text{If} \quad \begin{array}{ccc} \Gamma & \xrightarrow{r} & \Gamma' \\ {\scriptstyle v}\downarrow & \lrcorner & \downarrow{\scriptstyle u} \\ \Gamma'' & \xrightarrow[s]{} & \Gamma''' \end{array} \quad \text{then} \quad \prod_v \circ r^* = s^* \circ \prod_u \text{ and } s^* \circ \coprod_u = \coprod_v \circ r^*.$$

Generic predicate There exists $\Omega \in \mathbf{Set}$ and $\mathrm{holds} \in \mathcal{T}(\Omega)$, s.t.:
for all $\phi \in \mathcal{T}(\Gamma)$, we have $\chi_\phi : \Gamma \to \Omega$ satisfying

$$\chi_\phi^*(\mathrm{holds}) = \phi \qquad\qquad \chi_{s^*(\mathrm{holds})} = s$$

Models and triposes

A *tripos* is a functor $\mathcal{T} : \mathbf{Set}^{op} \to$ **HpA** s.t.:

↬ *Intuition* $\mathcal{T}(\Gamma)$: predicates $\varphi(\vec{x})$ $\qquad s^*(\underbrace{\varphi}_{\in \mathcal{T}(\Gamma')}) \triangleq \varphi(\underbrace{s}_{s:\Gamma\to\Gamma'}(\vec{y}))$

Quantifiers. Any s^* has left/right adjoints $\coprod_s / \prod_s : \mathcal{T}(\Gamma) \to \mathcal{T}(\Gamma')$:

$$\varphi \leq s^*(\psi) \quad \Leftrightarrow \quad \coprod_s(\varphi) \leq \psi \qquad\qquad s^*(\varphi) \leq \psi \quad \Leftrightarrow \quad \varphi \leq \prod_s(\psi)$$

Compatibility w. substitutions.

$$\text{If} \quad \begin{array}{ccc} \Gamma & \xrightarrow{r} & \Gamma' \\ {\scriptstyle v}\downarrow & \lrcorner & \downarrow{\scriptstyle u} \\ \Gamma'' & \xrightarrow[s]{} & \Gamma''' \end{array} \quad \text{then} \quad \prod_v \circ r^* = s^* \circ \prod_u \text{ and } s^* \circ \coprod_u = \coprod_v \circ r^*.$$

Generic predicate There exists $\Omega \in \mathbf{Set}$ and $\mathrm{holds} \in \mathcal{T}(\Omega)$, s.t.:
for all $\phi \in \mathcal{T}(\Gamma)$, we have $\chi_\phi : \Gamma \to \Omega$ satisfying

$$\chi_\phi^*(\mathrm{holds}) = \phi \qquad\qquad \chi_{s^*(\mathrm{holds})} = s$$

Models and triposes

A *tripos* is a functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathbf{HpA}$ s.t.:

↬ *Intuition* $\quad \mathcal{T}(\Gamma)$: predicates $\varphi(\vec{x}) \qquad\qquad s^*(\varphi) \triangleq \varphi(s(\vec{y}))$

Quantifiers. Any s^* has left/right adjoints $\coprod_s / \prod_s : \mathcal{T}(\Gamma) \to \mathcal{T}(\Gamma')$:

$$\varphi \leq s^*(\psi) \quad \Leftrightarrow \quad \coprod_s(\varphi) \leq \psi \qquad\qquad s^*(\varphi) \leq \psi \quad \Leftrightarrow \quad \varphi \leq \prod_s(\psi)$$

Compatibility w. substitutions.

$$\text{If} \quad \begin{array}{ccc} \Gamma & \xrightarrow{r} & \Gamma' \\ {\scriptstyle v}\downarrow & \lrcorner & \downarrow{\scriptstyle u} \\ \Gamma'' & \xrightarrow[s]{} & \Gamma''' \end{array} \quad \text{then } \prod_v \circ r^* = s^* \circ \prod_u \text{ and } s^* \circ \coprod_u = \coprod_v \circ r^*.$$

Generic predicate There exists $\Omega \in \mathbf{Set}$ and $\mathrm{holds} \in \mathcal{T}(\Omega)$, s.t.:
for all $\phi \in \mathcal{T}(\Gamma)$, we have $\chi_\phi : \Gamma \to \Omega$ satisfying

$$\chi_\phi^*(\mathrm{holds}) = \phi \qquad\qquad \chi_{s^*(\mathrm{holds})} = s$$

Models and triposes

A *tripos* is a functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathbf{HpA}$ s.t.:

↬ *Intuition* $\mathcal{T}(\Gamma)$: predicates $\varphi(\vec{x})$ $\qquad$ $s^*(\varphi) \triangleq \varphi(s(\vec{y}))$

Quantifiers. Any s^* has left/right adjoints $\coprod_s / \prod_s : \mathcal{T}(\Gamma) \to \mathcal{T}(\Gamma')$:

$$\varphi \leq s^*(\psi) \quad \Leftrightarrow \quad \coprod_s(\varphi) \leq \psi \qquad\qquad s^*(\varphi) \leq \psi \quad \Leftrightarrow \quad \varphi \leq \prod_s(\psi)$$

↬ *Intuition* $\exists x^X.- : \mathcal{T}(\Gamma \times X) \to \mathcal{T}(\Gamma)$ = left adjoint to $\pi_{\Gamma,X} : \Gamma \times X \to \Gamma$;

$$\forall \vec{y}, x.\, [\varphi(\vec{y}, x) \Rightarrow \psi(\vec{y})] \quad \Leftrightarrow \quad \forall \vec{y}.\, [\exists x.\varphi(\vec{y}, x) \Rightarrow \psi(\vec{y})]$$

Compatibility w. substitutions.

If $\begin{array}{ccc} \Gamma & \xrightarrow{r} & \Gamma' \\ {\scriptstyle v}\downarrow & \lrcorner & \downarrow{\scriptstyle u} \\ \Gamma'' & \xrightarrow[s]{} & \Gamma''' \end{array}$ then $\prod_v \circ r^* = s^* \circ \prod_u$ and $s^* \circ \coprod_u = \coprod_v \circ r^*$.

Generic predicate There exists $\Omega \in \mathbf{Set}$ and $\text{holds} \in \mathcal{T}(\Omega)$, s.t.:

Models and triposes

A *tripos* is a functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathbf{HpA}$ s.t.:

Quantifiers. Any s^* has left/right adjoints $\coprod_s / \prod_s : \mathcal{T}(\Gamma) \to \mathcal{T}(\Gamma')$:

$$\varphi \leq s^*(\psi) \quad \Leftrightarrow \quad \coprod_s(\varphi) \leq \psi \qquad\qquad s^*(\varphi) \leq \psi \quad \Leftrightarrow \quad \varphi \leq \prod_s(\psi)$$

Compatibility w. substitutions.

If $\begin{array}{ccc} \Gamma & \xrightarrow{r} & \Gamma' \\ {\scriptstyle v}\downarrow & \lrcorner & \downarrow{\scriptstyle u} \\ \Gamma'' & \xrightarrow[s]{} & \Gamma''' \end{array}$ then $\prod_v \circ r^* = s^* \circ \prod_u$ and $s^* \circ \coprod_u = \coprod_v \circ r^*$.

↬ *Intuition* $\qquad (\exists x.\varphi(y,x))[y := s(y')] = \exists x.\varphi(s(y'),x))$

Generic predicate There exists $\Omega \in \mathbf{Set}$ and $\mathrm{holds} \in \mathcal{T}(\Omega)$, s.t.:
for all $\phi \in \mathcal{T}(\Gamma)$, we have $\chi_\phi : \Gamma \to \Omega$ satisfying

$$\chi_\phi^*(\mathrm{holds}) = \phi \qquad\qquad \chi_{s^*(\mathrm{holds})} = s$$

Models and triposes

A *tripos* is a functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathbf{HpA}$ s.t.:

Quantifiers. Any s^* has left/right adjoints $\coprod_s / \prod_s : \mathcal{T}(\Gamma) \to \mathcal{T}(\Gamma')$:

$$\varphi \leq s^*(\psi) \quad \Leftrightarrow \quad \coprod_s(\varphi) \leq \psi \qquad\qquad s^*(\varphi) \leq \psi \quad \Leftrightarrow \quad \varphi \leq \prod_s(\psi)$$

Compatibility w. substitutions.

If $\begin{array}{ccc} \Gamma & \xrightarrow{r} & \Gamma' \\ {\scriptstyle v}\downarrow & \lrcorner & \downarrow{\scriptstyle u} \\ \Gamma'' & \xrightarrow[s]{} & \Gamma''' \end{array}$ then $\prod_v \circ r^* = s^* \circ \prod_u$ and $s^* \circ \coprod_u = \coprod_v \circ r^*$.

Generic predicate There exists $\Omega \in \mathbf{Set}$ and $\mathrm{holds} \in \mathcal{T}(\Omega)$, s.t.:
for all $\phi \in \mathcal{T}(\Gamma)$, we have $\chi_\phi : \Gamma \to \Omega$ satisfying

$$\chi_\phi^*(\mathrm{holds}) = \phi \qquad\qquad \chi_{s^*(\mathrm{holds})} = s$$

↬ *Intuition* Ω = set of propositions $\qquad \mathrm{holds}(\chi_\varphi(\vec{x})) \equiv \varphi(\vec{x})$

Models and triposes

A *tripos* is a functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathbf{HpA}$ s.t.:

Quantifiers. Any s^* has left/right adjoints $\coprod_s / \prod_s : \mathcal{T}(\Gamma) \to \mathcal{T}(\Gamma')$:

Compatibility w. substitutions. ...

Generic predicate. There exists $\Omega \in \mathbf{Set}$ and holds $\in \mathcal{T}(\Omega)$, s.t. ...

Standard example (with $\mathcal{H} \in \mathbf{HpA}$)

$$\mathcal{T}(I) = \mathcal{H}^I \qquad \mathcal{T}(s : I \to J) = \lambda(h : \mathcal{H}^J).\, h \circ s$$

Quantifiers. arbitrary meets/joins

Compatibility. yes.

Generic predicate. $\Omega \triangleq \mathcal{H}$ and holds $\triangleq \lambda x.x$

Models and triposes

A *tripos* is a functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathbf{HpA}$ s.t.:

Quantifiers. Any s^* has left/right adjoints $\coprod_s / \prod_s : \mathcal{T}(\Gamma) \to \mathcal{T}(\Gamma')$:

Compatibility w. substitutions. ...

Generic predicate. There exists $\Omega \in \mathbf{Set}$ and holds $\in \mathcal{T}(\Omega)$, s.t. ...

So far so good but:

- It is a heavy structure
- What about terms and the realizability interpretation?

Recently

Models and triposes

A *tripos* is a functor $\mathcal{T} : \mathbf{Set}^{op} \to \mathbf{HpA}$ s.t.:

Quantifiers. Any s^* has left/right adjoints $\coprod_s / \prod_s : \mathcal{T}(\Gamma) \to \mathcal{T}(\Gamma')$:

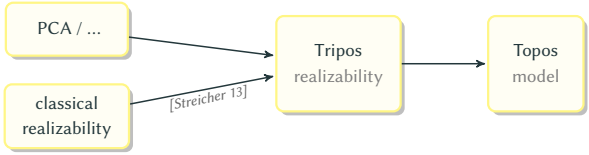
Compatibility w. substitutions. ...

Generic predicate. There exists $\Omega \in \mathbf{Set}$ and holds $\in \mathcal{T}(\Omega)$, s.t. ...

So far so good but:

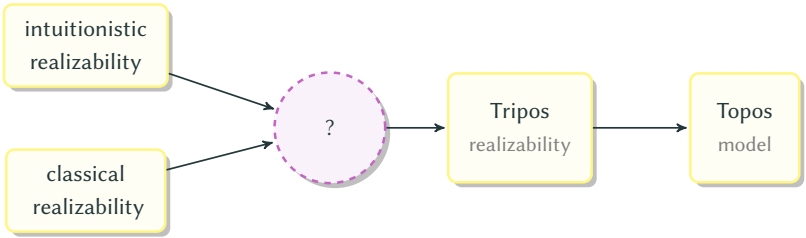
- It is a heavy structure
- What about terms and the realizability interpretation?

Recently



Effectful realizability, from an algebraic perspective

The quest for a better world



Algebrization of realizability models

Key observations : truth values live in $\mathcal{P}(\mathbb{A})$ which

1. is a complete lattice,
2. can be endowed with Kleene arrow:

$$\phi \Rightarrow \psi = \{a \in \mathbb{A} \mid \forall b \in \phi . ab \downarrow \wedge \ ab \in \psi\}$$

↬ *for classical realizability, this applies to* $\mathcal{P}(\Pi)$.

Algebrization of realizability models

Key observations : truth values live in $\mathcal{P}(\mathbb{A})$ which

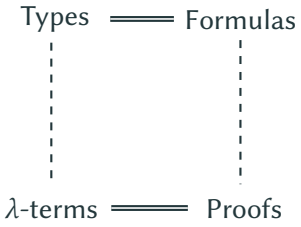
1. is a complete lattice,
2. can be endowed with Kleene arrow:

$$\phi \Rightarrow \psi = \{a \in \mathbb{A} \mid \forall b \in \phi . ab \downarrow \wedge \ ab \in \psi\}$$

↬ *for classical realizability, this applies to* $\mathcal{P}(\Pi)$.

Implicative structures

Implicative algebras: a new (...)
Alexandre Miquel [2018]

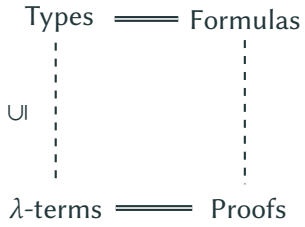


In particular, $a \preccurlyeq b$ reads:

- a is a *subtype* of b
- a is a *realizer* of b
- the realizer a is *more defined* than b

Implicative structures

Implicative algebras: a new (...)
Alexandre Miquel [2018]

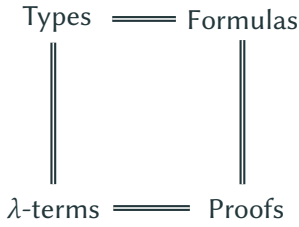


In particular, $a \preccurlyeq b$ reads:

- a is a *subtype* of b
- a is a *realizer* of b
- the realizer a is *more defined* than b

Implicative structures

Implicative algebras: a new (...)
Alexandre Miquel [2018]



In particular, $a \preccurlyeq b$ reads:

- a is a *subtype* of b
- a is a *realizer* of b
- the realizer a is *more defined* than b

Implicative algebras

An *implicative algebra* is a complete meet-semilattice $(\mathcal{A}, \preccurlyeq, \to)$ s.t.:

- if $a_0 \preccurlyeq a$ and $b \preccurlyeq b_0$ then $(a \to b) \preccurlyeq (a_0 \to b_0)$ *(Variance)*
- $\bigwedge_{b \in B}(a \to b) = a \to \bigwedge_{b \in B} b$ *(Distributivity)*

together with a **separator** $\mathcal{S} \in \mathcal{A}$ satisfying axioms.

Main results:

- System F's terms and types can be interpreted in an adequate way:

$$\text{If } \vdash t : A \quad \text{then} \quad t^{\mathcal{A}} \preccurlyeq A^{\mathcal{A}}.$$

- Any implicative algebra induces a tripos :

$$\mathcal{T} : \begin{cases} \mathbf{Set}^{op} & \to & \mathbf{HA} \\ I & \mapsto & \mathcal{A}^I/\mathcal{S}[I] \end{cases}$$

Great, but...

- implicative algebras : a (too?) minimal structure

realizers $\in \mathcal{A} \ni$ formulas

- tripos: does not account for realizers

propositional logic $\leftrightarrow$ Heyting prealgebra
quantifiers $\leftrightarrow$ adjoints to substitutions
higher-order $\leftrightarrow$ generic predicate

- lack of a smooth integration for

states / non-determinism / dependent types / ...

In short

Two general structures, *none account for realizers!*

Great, but...

- implicative algebras : a (too?) minimal structure

realizers $\in \mathcal{A} \ni$ formulas

- tripos: does not account for realizers

propositional logic	$\leftrightarrow$	Heyting prealgebra
quantifiers	$\leftrightarrow$	adjoints to substitutions
higher-order	$\leftrightarrow$	generic predicate

- lack of a smooth integration for

states / non-determinism / dependent types / ...

In short

Two general structures, *none account for realizers!*

Great, but...

- implicative algebras : a (too?) minimal structure

realizers $\in \mathcal{A} \ni$ formulas

- tripos: does not account for realizers

propositional logic	$\leftrightarrow$	Heyting prealgebra
quantifiers	$\leftrightarrow$	adjoints to substitutions
higher-order	$\leftrightarrow$	generic predicate

- lack of a smooth integration for

states / non-determinism / dependent types / ...

In short

Two general structures, *none account for realizers!*

Great, but...

- implicative algebras : a (too?) minimal structure

realizers $\in \mathcal{A} \ni$ formulas

- tripos: does not account for realizers

propositional logic	$\leftrightarrow$	Heyting prealgebra
quantifiers	$\leftrightarrow$	adjoints to substitutions
higher-order	$\leftrightarrow$	generic predicate

- lack of a smooth integration for

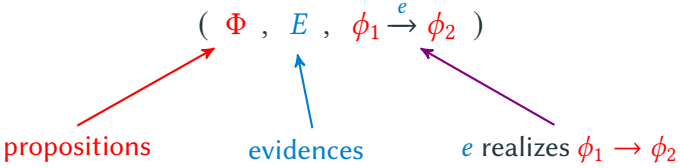
states / non-determinism / dependent types / ...

In short

Two general structures, *none account for realizers!*

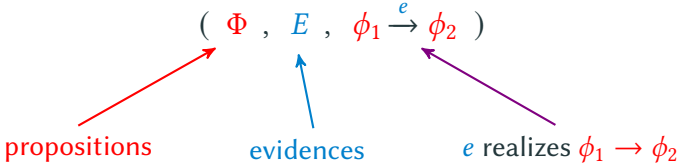
Evidenced Frames

Evidenced Frame: A Unifying Framework for Realizability Models



Intuitively: a "specification" of the minimal structure

Evidenced Frame: A Unifying Framework for Realizability Models



Intuitively: a "specification" of the minimal structure

Evidenced Frame $(\Phi, E, \cdot \dot{\rightarrow} \cdot)$

Reflexivity. $e_{\mathrm{id}} \in E$ s.t.:

- $\phi \xrightarrow{e_{\mathrm{id}}} \phi$

Transitivity. $; \in E \times E \to E$ s.t.:

- $\phi_1 \xrightarrow{e} \phi_2 \wedge \phi_2 \xrightarrow{e'} \phi_3 \implies \phi_1 \xrightarrow{e\,;e'} \phi_3$

Top. $\top \in \Phi$ and $e_\top \in E$ s.t.:

- $\phi \xrightarrow{e_\top} \top$

Conjunction. $\wedge \in \Phi \times \Phi \to \Phi$, $\langle \cdot, \cdot \rangle \in E \times E \to E$, and $e_{\mathrm{fst}}, e_{\mathrm{snd}} \in E$ s.t.:

- $\phi_1 \wedge \phi_2 \xrightarrow{e_{\mathrm{fst}}} \phi_1$
- $\phi_1 \wedge \phi_2 \xrightarrow{e_{\mathrm{snd}}} \phi_2$
- $\phi \xrightarrow{e_1} \phi_1 \wedge \phi \xrightarrow{e_2} \phi_2 \implies \phi \xrightarrow{\langle e_1, e_2 \rangle} \phi_1 \wedge \phi_2$

Universal implication. $\supset \in \Phi \times \mathcal{P}(\Phi) \to \Phi$, $\lambda \in E \to E$, and $e_{\mathrm{eval}} \in E$:

- $(\forall \phi \in \vec{\phi}.\ \phi_1 \wedge \phi_2 \xrightarrow{e} \phi) \implies \phi_1 \xrightarrow{\lambda e} \phi_2 \supset \vec{\phi}$
- $\forall \phi \in \vec{\phi}.\ [(\phi_1 \supset \vec{\phi}) \wedge \phi_1 \xrightarrow{e_{\mathrm{eval}}} \phi]$

Evidenced Frame $(\Phi, E, \cdot \dot{\rightarrow} \cdot)$

Reflexivity. $e_{\mathsf{id}} \in E$ s.t.:

- $\phi \xrightarrow{e_{\mathsf{id}}} \phi$

Transitivity. $; \in E \times E \to E$ s.t.:

- $\phi_1 \xrightarrow{e} \phi_2 \wedge \phi_2 \xrightarrow{e'} \phi_3 \implies \phi_1 \xrightarrow{e\,;e'} \phi_3$

Top. $\top \in \Phi$ and $e_\top \in E$ s.t.:

- $\phi \xrightarrow{e_\top} \top$

Conjunction. $\wedge \in \Phi \times \Phi \to \Phi$, $\langle\!|\cdot,\cdot|\!\rangle \in E \times E \to E$, and $e_{\mathsf{fst}}, e_{\mathsf{snd}} \in E$ s.t.:

- $\phi_1 \wedge \phi_2 \xrightarrow{e_{\mathsf{fst}}} \phi_1$
- $\phi_1 \wedge \phi_2 \xrightarrow{e_{\mathsf{snd}}} \phi_2$
- $\phi \xrightarrow{e_1} \phi_1 \wedge \phi \xrightarrow{e_2} \phi_2 \implies \phi \xrightarrow{\langle\!|e_1,e_2|\!\rangle} \phi_1 \wedge \phi_2$

Universal implication. $\supset \in \Phi \times \mathcal{P}(\Phi) \to \Phi$, $\lambda \in E \to E$, and $e_{\mathsf{eval}} \in E$:

- $(\forall \phi \in \vec{\phi}.\ \phi_1 \wedge \phi_2 \xrightarrow{e} \phi) \implies \phi_1 \xrightarrow{\lambda e} \phi_2 \supset \vec{\phi}$
- $\forall \phi \in \vec{\phi}.[(\phi_1 \supset \vec{\phi}) \wedge \phi_1 \xrightarrow{e_{\mathsf{eval}}} \phi]$

Goal #1: An intermediate structure

PCA to Evidenced Frame

- A set of "codes"
- *partial* application $c_1 \cdot c_2$
- functional completeness

PCA to Evidenced Frame

- $\mathbb{A}$ set of "codes"
- *partial* application $c_1 \cdot c_2$
- functional completeness

The triple $(\mathcal{P}(\mathbb{A}), \mathbb{A}, \cdot \to \cdot)$, where:

- a proposition in $\Phi = \mathcal{P}(\mathbb{A})$ is defined by its set of realizers
- an evidence in $E = \mathbb{A}$ is a code
- $\phi_1 \xrightarrow{e} \phi_2$ if for all $e_1 \in \phi_1$:
  - $e \cdot e_1$ terminates
  - $e \cdot e_1 \downarrow e_2 \implies e_2 \in \phi_2$

↬ *connectives and their evidences are defined as usual in realizability models*

PCA to Evidenced Frame

- $\mathbb{A}$ set of "codes"
- *partial* application $c_1 \cdot c_2$
- functional completeness

The triple $(\mathcal{P}(\mathbb{A}), \mathbb{A}, \cdot \dot{\rightarrow} \cdot)$, where:

- a proposition in $\Phi = \mathcal{P}(\mathbb{A})$ is defined by its set of realizers
- an evidence in $E = \mathbb{A}$ is a code
- $\phi_1 \overset{e}{\longrightarrow} \phi_2$ if for all $e_1 \in \phi_1$:
  - $e \cdot e_1$ terminates
  - $e \cdot e_1 \downarrow e_2 \implies e_2 \in \phi_2$

↬ *connectives and their evidences are defined as usual in realizability models*

Evidenced Frame to Tripos

UFam construction

Given $\mathcal{EF} = (\Phi, E, \cdot \dot{\to} \cdot)$, the structure $\mathrm{UFam}(\mathcal{EF})$ is defined by:

Predicates. $\Gamma \in \mathbf{Set}$ is mapped to $\Phi^\Gamma \in \mathbf{HpA}$

- $\phi \preccurlyeq \phi' \triangleq \exists e. \forall \gamma.\ \phi(\gamma) \overset{e}{\to} \phi'(\gamma)$ ↬ *uniform entailment*
- Heyting prealgebra: pointwise via Φ

Substitution. $s : \Gamma \to \Gamma'$ is mapped to $\mathcal{T}(s) = \lambda h.\, h \circ s$ ↬ *as usual*

Quantifiers. $\prod_u \in \Phi^I \to \Phi^J \triangleq \lambda \phi. \lambda j.\ \prod_{i \in u^{-1}(j)} \phi(i)$ ↬ *as usual*

$\coprod_u \in \Phi^I \to \Phi^J \triangleq \lambda \phi. \lambda j.\ \coprod_{i \in u^{-1}(j)} \phi(i).$

Generic predicate. $\Omega \triangleq \Phi$, holds $\triangleq \mathrm{id}_\Omega$, and $\chi_\phi \triangleq \phi$. ↬ *as usual*

From Implicative Algebras to Evidenced frames

Any implicative algebra $(\mathcal{A}, \preccurlyeq, \to, \mathcal{S})$ induces an evidenced frame

$$\mathsf{UEF}(\mathcal{A}) \triangleq (\underbrace{\mathcal{A}}_{\text{prop.}}, \underbrace{\mathcal{S}}_{\text{evidences}}, \cdot \dot{\to} \cdot) \qquad \text{where} \quad a \overset{e}{\to} b \triangleq e \preccurlyeq a \to b$$

↪ ***Proof.*** *Connectives and quantifiers from the internal logic of $\mathcal{A}$ / evidences via the expected λ-terms.*

Remark

- blurs the distinction btw. evidences & propositions
- $\mathsf{UEF}(\mathcal{A})$ is consistent if and only if $\mathcal{A}$ is.
- the implicative tripos $\mathcal{T}^{\mathcal{A}}$ and $\mathsf{UFam}(\mathsf{UEF}(\mathcal{A}))$ are equivalent.

From Implicative Algebras to Evidenced frames

Any implicative algebra $(\mathcal{A}, \preccurlyeq, \to, \mathcal{S})$ induces an evidenced frame

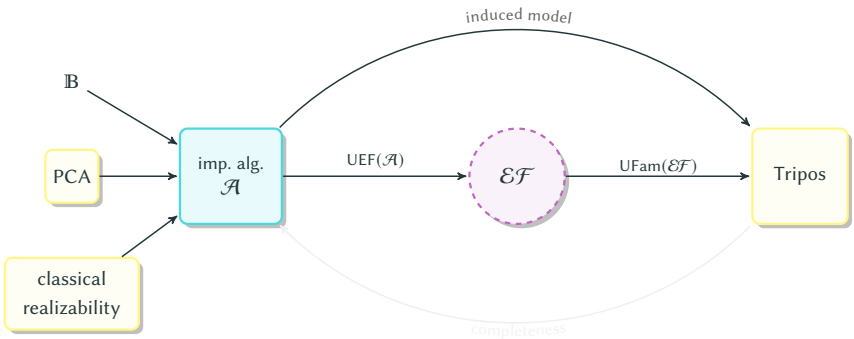
$$\mathrm{UEF}(\mathcal{A}) \triangleq (\underbrace{\mathcal{A}}_{\text{prop.}}, \underbrace{\mathcal{S}}_{\text{evidences}}, \cdot \dot{\to} \cdot) \qquad \text{where} \quad a \overset{e}{\to} b \triangleq e \preccurlyeq a \to b$$

↣ ***Proof.*** *Connectives and quantifiers from the internal logic of $\mathcal{A}$ / evidences via the expected λ-terms.*

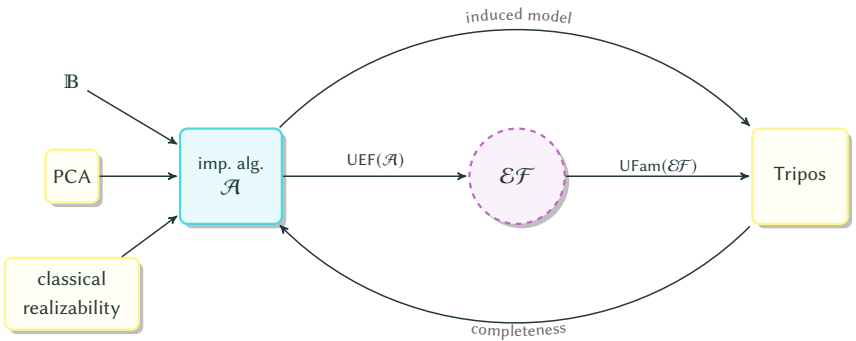
Remark

- blurs the distinction btw. evidences & propositions
- $\mathrm{UEF}(\mathcal{A})$ is consistent if and only if $\mathcal{A}$ is.
- the implicative tripos $\mathcal{T}^{\mathcal{A}}$ and $\mathrm{UFam}(\mathrm{UEF}(\mathcal{A}))$ are equivalent.

From Implicative Algebras to Evidenced frames

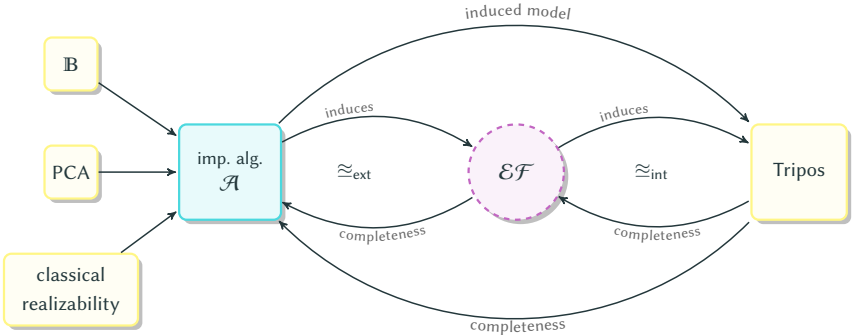


From Implicative Algebras to Evidenced frames



Final picture

Evidenced Frames : a Unifying... (...) *Cohen, M. and Tate* [2021]



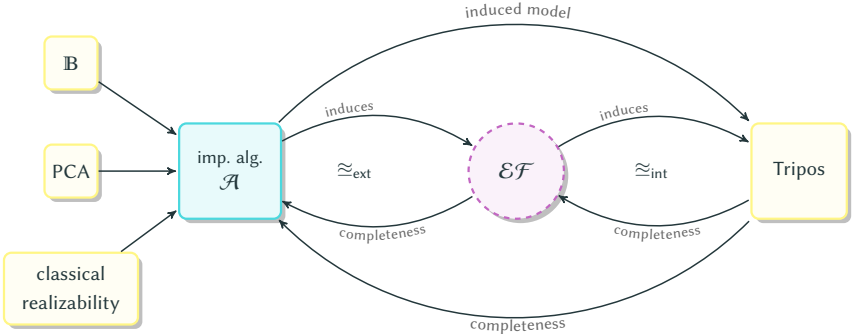
Slogan

Tripos = evidenced frame that has forgotten its evidence.

ROCQ formalization available : https://www.i2m.univ-amu.fr/perso/etienne.miquey/

Final picture

Evidenced Frames : a Unifying... (...) *Cohen, M. and Tate* [2021]

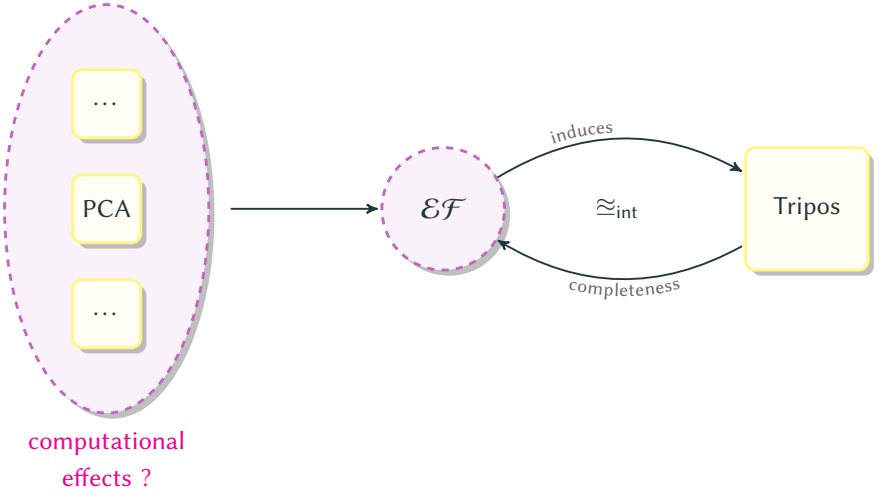


Slogan

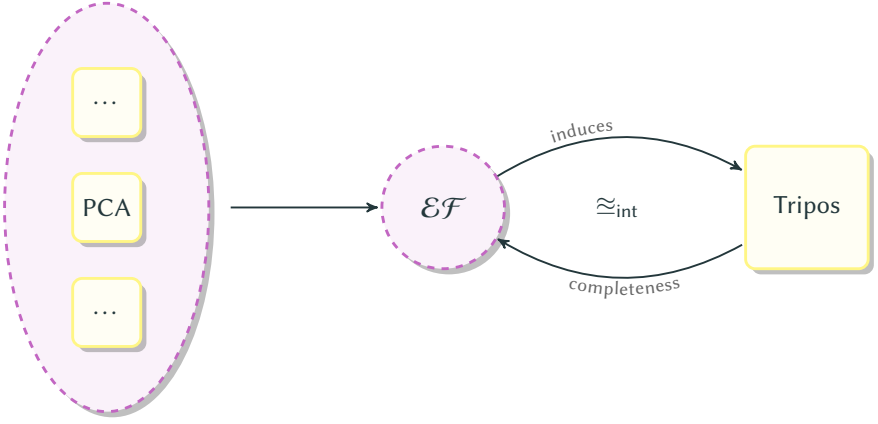
Tripos = evidenced frame that has forgotten its evidence.

ROCQ formalization available : https://www.i2m.univ-amu.fr/perso/etienne.miquey/

This Talk: Goal #2



This Talk: Goal #2



Goal #2

Smooth integration of useful computational effects

Monadic Combinatory Algebras

Reminder : PCA

Definition

A **Partial Applicative Structure** $(\mathbb{A}, \cdot)$ is given by:

a set $\mathbb{A}$ of "codes" + a partial "application" $\cdot : \mathbb{A} \times \mathbb{A} \to \mathbb{A}$

Notations:

- $c_f \cdot c_a \downarrow c_r$: c_r is the (successful) result of the application $c_f \cdot c_a$
- $c_f \cdot c_a \downarrow$: $\exists c_r \in \mathbb{A}, c_f \cdot c_a \downarrow c_r$.

Definition

A **Partial Combinatory Algebra** is a PAS that is "functionally complete".

↬ *for every expression e with n free variables, there is a code* $\langle \lambda^n.e \rangle \in \mathbb{A}$ *s.t.:*

$$\langle \lambda^{n+1}.e \rangle \cdot c_a \downarrow \langle \lambda^n.e\{c_a\} \rangle \qquad \langle \lambda^0.e \rangle \cdot c_a \downarrow c_r \iff e\{c_a\} \downarrow c_r$$

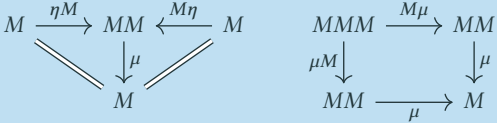
Monads

Definition

A **monad** M over a category $\mathbb{C}$ is a functor $M : \mathbb{C} \to \mathbb{C}$, with:

- $\eta : \mathbf{1} \Rightarrow M$ *(unit)*
- $\mu : MM \Rightarrow M$ *(multiplication)*

satisfying



Computationally :

- $f : A \to MB$: effectful program taking a value in A and returning a computation in B
- $[a] : MA$ stands for $\eta\,(a)$ *('return')*
- let $x \Leftarrow m$ in $n : MB$ with $\lambda x.n : A \to MB$ and $m : MA$ *('bind')*

Monadic Applicative Structure

In this work: M is a **Set** monad

Definition

A **Monadic Applicative Structure** (MAS) over M is a set of "codes" $\mathbb{A}$ with a (Kleisli) application $: (-) \cdot (-) : \mathbb{A} \times \mathbb{A} \to M\mathbb{A}$.

↬ *PAS where partiality is replaced by any monad.*

Intuitively:

- $(-) \cdot (-)$ takes two codes and returns a *'computation'* of a code.
- codes = encoding of programs or inputs of programs.
- $c_f \cdot c_a$: computation executed by running c_f on c_a

Expressions

$$e ::= i \in \mathbb{N} \mid c \in \mathbb{A} \mid e \bullet e \qquad E_n(\mathbb{A}) ::= \{e \mid \text{all } is \text{ in } e \text{ are} < n\}$$

Monadic Applicative Structure

In this work: M is a **Set** monad

Definition

A **Monadic Applicative Structure** (MAS) over M is a set of "codes" $\mathbb{A}$ with a (Kleisli) application $: (-) \cdot (-) : \mathbb{A} \times \mathbb{A} \to M\mathbb{A}$.

↬ *PAS where partiality is replaced by any monad.*

Intuitively:

- $(-) \cdot (-)$ takes two codes and returns a *'computation'* of a code.
- codes = encoding of programs or inputs of programs.
- $c_f \cdot c_a$: computation executed by running c_f on c_a

Expressions

$$e ::= i \in \mathbb{N} \mid c \in \mathbb{A} \mid e \bullet e \qquad E_n(\mathbb{A}) ::= \{e \mid \text{all } is \text{ in } e \text{ are} < n\}$$

Monadic Applicative Structure

In this work: M is a **Set** monad

Definition

A **Monadic Applicative Structure** (MAS) over M is a set of "codes" $\mathbb{A}$ with a (Kleisli) application $: (-) \cdot (-) : \mathbb{A} \times \mathbb{A} \to M\mathbb{A}$.

↬ *PAS where partiality is replaced by any monad.*

Intuitively:

- $(-) \cdot (-)$ takes two codes and returns a *'computation'* of a code.
- codes = encoding of programs or inputs of programs.
- $c_f \cdot c_a$: computation executed by running c_f on c_a

Expressions

$$e ::= i \in \mathbb{N} \mid c \in \mathbb{A} \mid e \bullet e \qquad E_n(\mathbb{A}) ::= \{e \mid \text{all } i\text{s in } e \text{ are} < n\}$$

Evaluation

We follow the PCA historical presentation :

CbV Evaluation

Evaluation v is defined by induction on $E_0(\mathbb{A})$ as follows:

$$v(c) := [c] \qquad v(e_f \bullet e_a) := \text{let } c_f \Leftarrow v(e_f) \text{ in let } c_a \Leftarrow v(e_a) \text{ in } c_f \cdot c_a$$

↬ *evaluation is reflected by an equality in* $\mathbb{A}$

↬ *having an algebraic counterpart of evaluation would require switching to an ordered setting*

Monadic Combinatory Algebra

MCA

A **Monadic Combinatory Algebra** is a MAS $\mathbb{A}$ s.t. for any expression $e \in E_{n+1}(\mathbb{A})$ there is a code $\langle \lambda^n.e \rangle \in \mathbb{A}$ satisfying the following laws:

$$\begin{aligned} \left\langle \lambda^{n+1}.e \right\rangle \cdot c &= \eta\left(\langle \lambda^n.e\{c\} \rangle\right) && (\forall e \in E_{n+2}(\mathbb{A})) \\ \left\langle \lambda^0.e \right\rangle \cdot c &= \nu\left(e\{c\}\right) && (\forall e \in E_1(\mathbb{A})) \end{aligned}$$

↬ *this can also be expressed in categorical terms, see the paper*

Example -

The **sub-singleton monad**:

- $MA = \{S \subseteq A \mid \forall x_1, x_2 \in S. x_1 = x_2\}$
- $\eta_A(x) = \{x\}$
- $\mu_A(m) = \bigcup_{X \in m} X$

Riddle : what is the encoded effect ?

PCA is a special case of an MCA.

Example - PCA

The **sub-singleton monad**:

- $MA = \{S \subseteq A \mid \forall x_1, x_2 \in S. x_1 = x_2\}$
- $\eta_A(x) = \{x\}$
- $\mu_A(m) = \bigcup_{X \in m} X$

Riddle : what is the encoded effect ?

PCA is a special case of an MCA.

Example -

• $MA = (A \to R) \to R$ • $\eta_A(x) = \lambda k.k\,x$ • $\mu_A(m) = \lambda k.m\,(\lambda g.g\,(k))$

Riddle : what is the encoded effect ?

Such an MCA allows for e_{cc} and e_{K_u} which save/restore continuations :

$$(e_{cc} \cdot c_a)(u) = (c_a \cdot e_{K_u})(u) \qquad (e_{K_u} \cdot c_a)(u') = u(c_a)$$

By defunctionalization, we obtain an abstract stack machine with states:

1. *eval* state
2. *apply* state
3. *final* state

Operational semantics:

$$\begin{array}{lll} e_f \cdot e_a \rhd \pi & \hookrightarrow e_f \rhd t(e_a) \cdot \pi & \\ c \rhd \pi & \hookrightarrow c \lhd \pi & \\ c \lhd \emptyset & \hookrightarrow c & \end{array} \qquad \begin{array}{lll} c_f \lhd t(e_a) \cdot \pi & \hookrightarrow e_a \rhd v(c_f) \cdot \pi & \\ c_a \lhd v(\langle \lambda^0.e \rangle) \cdot \pi & \hookrightarrow e\{c_a\} \rhd \pi & \\ c_a \lhd v(\langle \lambda^{n+1}.e \rangle) \cdot \pi & \hookrightarrow \langle \lambda^n.e\{c_a\} \rangle \lhd \pi & \end{array}$$

Example - CPSCA

• $MA = (A \to R) \to R$ • $\eta_A\,(x) = \lambda k.k\,x$ • $\mu_A\,(m) = \lambda k.m\,(\lambda g.g\,(k))$

Such an MCA allows for e_{cc} and e_{K_u} which save/restore continuations :

$$(e_{\mathrm{cc}} \cdot c_a)\,(u) = \left(c_a \cdot e_{\mathrm{K}_u}\right)(u) \qquad \left(e_{\mathrm{K}_u} \cdot c_a\right)(u') = u\,(c_a)$$

By defunctionalization, we obtain an abstract stack machine with states:

1. *eval* state
2. *apply* state
3. *final* state

Operational semantics:

$$\begin{array}{rcl} e_f \cdot e_a \rhd \pi & \hookrightarrow & e_f \rhd t\,(e_a) \cdot \pi \\ c \rhd \pi & \hookrightarrow & c \lhd \pi \\ c \lhd \emptyset & \hookrightarrow & c \end{array} \qquad \begin{array}{rcl} c_f \lhd t\,(e_a) \cdot \pi & \hookrightarrow & e_a \rhd v\,(c_f) \cdot \pi \\ c_a \lhd v\,(\langle \lambda^0.e \rangle) \cdot \pi & \hookrightarrow & e\{c_a\} \rhd \pi \\ c_a \lhd v\,(\langle \lambda^{n+1}.e \rangle) \cdot \pi & \hookrightarrow & \langle \lambda^n.e\{c_a\} \rangle \lhd \pi \end{array}$$

Example - CPSCA

$\bullet\ MA = (A \to R) \to R \qquad \bullet\ \eta_A\,(x) = \lambda k.k\,x \qquad \bullet\ \mu_A\,(m) = \lambda k.m\,(\lambda g.g\,(k))$

Such an MCA allows for e_{cc} and e_{K_u} which save/restore continuations :

$$(e_{\mathrm{cc}} \cdot c_a)\,(u) = \left(c_a \cdot e_{\mathrm{K}_u}\right)(u) \qquad \left(e_{\mathrm{K}_u} \cdot c_a\right)(u') = u\,(c_a)$$

By defunctionalization, we obtain an abstract stack machine with states:

❶ $e \rhd \pi$ *eval* state ❷ $c \lhd \pi$ *apply* state ❸ c *final* state

Operational semantics:

$$\begin{array}{rcl} e_f \cdot e_a \rhd \pi & \hookrightarrow & e_f \rhd \boldsymbol{t}\,(e_a) \cdot \pi \\ c \rhd \pi & \hookrightarrow & c \lhd \pi \\ c \lhd \emptyset & \hookrightarrow & c \end{array} \qquad \begin{array}{rcl} c_f \lhd \boldsymbol{t}\,(e_a) \cdot \pi & \hookrightarrow & e_a \rhd \boldsymbol{v}\left(c_f\right) \cdot \pi \\ c_a \lhd \boldsymbol{v}\left(\left\langle \lambda^0.e \right\rangle\right) \cdot \pi & \hookrightarrow & e\{c_a\} \rhd \pi \\ c_a \lhd \boldsymbol{v}\left(\left\langle \lambda^{n+1}.e \right\rangle\right) \cdot \pi & \hookrightarrow & \langle \lambda^n.e\{c_a\} \rangle \lhd \pi \end{array}$$

More Examples

Comb. Alg.	**Monad** MA	**return** $\eta_A\,(x)$	**bind** $\mu_A\,(m)$
Partial	$\{X \subseteq A \mid \forall x, y \in X.x = y\}$	$\{x\}$	$\bigcup_{X \in m} X$
Relational	$\mathcal{P}(A)$	$\{x\}$	$\bigcup_{X \in m} X$
Stateful	$\Sigma \to \mathcal{P}(\Sigma \times A)$	$\lambda\sigma.\{(\sigma, x)\}$	$\lambda\sigma. \bigcup_{(\sigma', f) \in m(\sigma)} f\,(\sigma)$
CPS	$(A \to R) \to R$	$\lambda k.k\,(x)$	$\lambda k.m\,(\lambda g.g\,(k))$
Parameterized	$\mathbf{P} \to \{X \subseteq A \mid \forall x, y \in X.x = y\}$	$\lambda p.\,\{x\}$	$\lambda p. \bigcup \{g\,(p) \mid g \in m\,(p)\}$

Realizability models

The good old recipe :

- $\mathcal{P}(\mathbb{A})$-valued predicates

$$\phi : X \to \mathcal{P}(\mathbb{A}).$$

- Kleene arrow

$$(\phi \Rightarrow \psi)(x) = \{a \in \mathbb{A} \mid \forall b \in \phi(x).ab \downarrow \wedge\ ab \in \psi(x)\}.$$

Problem

We need to lift $\psi \in \mathcal{P}(\mathbb{A})$ to an adequate set *of computations* $\overline{\psi} \in \mathcal{P}(M\mathbb{A})$

Realizability models

The good old recipe :

- $\mathcal{P}(\mathbb{A})$-valued predicates

$$\phi : X \to \mathcal{P}(\mathbb{A}).$$

- Kleene arrow

$$(\phi \Rightarrow \psi)(x) = \{a \in \mathbb{A} \mid \forall b \in \phi(x).ab \downarrow \wedge \ ab \in \psi(x)\}.$$

Problem

We need to lift $\psi \in \mathcal{P}(\mathbb{A})$ to an adequate set *of computations* $\overline{\psi} \in \mathcal{P}(M\mathbb{A})$

Realizability models

The good old recipe :

- $\mathcal{P}(\mathbb{A})$-valued predicates

$$\phi : X \to \mathcal{P}(\mathbb{A}).$$

- Kleene arrow ✗

$$(\phi \Rightarrow \psi)(x) = \{a \in \mathbb{A} \mid \forall b \in \phi(x).ab \downarrow \wedge \underbrace{ab}_{\in M\mathbb{A}} \in \underbrace{\psi(x)}_{\subseteq \mathbb{A}}\}.$$

Problem

We need to lift $\psi \in \mathcal{P}(\mathbb{A})$ to an adequate set *of computations* $\overline{\psi} \in \mathcal{P}(M\mathbb{A})$

M-modality

An M-modality over $(\Omega, \preccurlyeq)$ a complete **HpA** is a natural transformation:

$$\Diamond_X : MX \to (X \to \Omega) \to \Omega$$

s.t., writing $\langle\!\langle m \rangle\!\rangle \phi$ (*"after m, ϕ holds"*), we have:

1. After / return :

$$\phi(a) \preccurlyeq \langle\!\langle x \leftarrow [a] \rangle\!\rangle \phi(x)$$

2. After / Bind

$$\langle\!\langle x \leftarrow m \rangle\!\rangle \langle\!\langle y \leftarrow f(x) \rangle\!\rangle \phi(y) \preccurlyeq \langle\!\langle y \leftarrow \text{let } x \Leftarrow m \text{ in } f(x) \rangle\!\rangle \phi(y)$$

3. Internal Monotonicity.

$$\prod_c (\phi_1(c) \sqsupset \phi_2(c)) \preccurlyeq \langle\!\langle x \leftarrow m \rangle\!\rangle \phi_1(x) \sqsupset \langle\!\langle x \leftarrow m \rangle\!\rangle \phi_2(x)$$

M-modality

An M-modality over $(\Omega, \preccurlyeq)$ a complete **HpA** is a natural transformation:

$$\Diamond_X : \underbrace{MX}_{\textit{computation}} \to \underbrace{(X \to \Omega)}_{\textit{predicate}} \to \underbrace{\Omega}_{\textit{truth value}}$$

s.t., writing $\Diamond(\phi)(m)$ (*"after m, ϕ holds"*), we have:

1. After / return :

$$\phi(a) \preccurlyeq \Diamond(\phi)([a])$$

2. After / Bind

$$\Diamond(\phi)(m) \preccurlyeq \Diamond(\Diamond(\phi) \circ f)(\text{let } x \Leftarrow m \text{ in } f(x))$$

3. Internal Monotonicity.

$$\bigsqcap_c (\phi_1(c) \sqsupset \phi_2(c)) \preccurlyeq \Diamond(\phi_1)(m) \sqsupset \Diamond(\phi_2)(m)$$

M-modality

An M-modality over $(\Omega, \preccurlyeq)$ a complete **HpA** is a natural transformation:

$$\Diamond_X : \underbrace{MX}_{\textit{computation}} \to \underbrace{(X \to \Omega)}_{\textit{predicate}} \to \underbrace{\Omega}_{\textit{truth value}}$$

s.t., writing $\langle\!\!\langle x \leftarrow m \rangle\!\!\rangle \phi(x) \triangleq \Diamond(m)(\phi)$ (*"after m, ϕ holds"*), we have:

1. After / return :

$$\phi(a) \preccurlyeq \langle\!\!\langle x \leftarrow [a] \rangle\!\!\rangle \phi(x)$$

2. After / Bind

$$\langle\!\!\langle x \leftarrow m \rangle\!\!\rangle \langle\!\!\langle y \leftarrow f(x) \rangle\!\!\rangle \phi(y) \preccurlyeq \langle\!\!\langle y \leftarrow \text{let } x \Leftarrow m \text{ in } f(x) \rangle\!\!\rangle \phi(y)$$

3. Internal Monotonicity.

$$\prod_c (\phi_1(c) \sqsupset \phi_2(c)) \preccurlyeq \langle\!\!\langle x \leftarrow m \rangle\!\!\rangle \phi_1(x) \sqsupset \langle\!\!\langle x \leftarrow m \rangle\!\!\rangle \phi_2(x)$$

M-modality

An M-modality over $(\Omega, \preccurlyeq)$ a complete **HpA** is a natural transformation:

$$\Diamond_X : \underbrace{MX}_{\textit{computation}} \to \underbrace{(X \to \Omega)}_{\textit{predicate}} \to \underbrace{\Omega}_{\textit{truth value}}$$

s.t., writing $\langle\!\!\langle x \leftarrow m \rangle\!\!\rangle \phi(x) \triangleq \Diamond(m)(\phi)$ (*"after m, ϕ holds"*), we have:

1. After / return :

$$\phi(a) \preccurlyeq \langle\!\!\langle x \leftarrow [a] \rangle\!\!\rangle \phi(x)$$

2. After / Bind

$$\langle\!\!\langle x \leftarrow m \rangle\!\!\rangle \langle\!\!\langle y \leftarrow f(x) \rangle\!\!\rangle \phi(y) \preccurlyeq \langle\!\!\langle y \leftarrow \text{let } x \Leftarrow m \text{ in } f(x) \rangle\!\!\rangle \phi(y)$$

3. Internal Monotonicity.

$$\bigsqcap_c (\phi_1(c) \sqsupset \phi_2(c)) \preccurlyeq \langle\!\!\langle x \leftarrow m \rangle\!\!\rangle \phi_1(x) \sqsupset \langle\!\!\langle x \leftarrow m \rangle\!\!\rangle \phi_2(x)$$

All good ?

Consider the **PCA** example, that is:

$$MA = \{S \subseteq A \mid \forall x_1, x_2 \in S. x_1 = x_2\}$$

What about the following M-modality?

$$\left\langle\!\!\left\langle x \leftarrow m \right\rangle\!\!\right\rangle \phi\,(x) \triangleq \bigsqcap_{x \in m} \phi\,(x)$$

Consider :

- $c_{\circlearrowleft} \triangleq \langle \lambda x.\delta\delta \rangle$, that yields no value when applied,
- the predicates $\top = x \mapsto \mathbf{1}$ and $\bot = x \mapsto \mathbf{0}$

Then we get:

$$\top \xrightarrow{c_{\circlearrowleft}} \bot \Leftrightarrow \forall c \in \mathbf{A}.\mathbf{1} \preccurlyeq \left\langle\!\!\left\langle x \leftarrow c_{\circlearrowleft} \cdot c \right\rangle\!\!\right\rangle \mathbf{0}$$

$$\Leftrightarrow \forall c \in \mathbf{A}.\mathbf{1} \preccurlyeq \bigsqcap_{x \in \mathrm{v}(c_{\circlearrowleft} \cdot c)} .\mathbf{0} \Leftrightarrow \mathbf{1} \preccurlyeq \mathbf{1}$$

✗

All good ?

Consider the **PCA** example, that is:

$$MA = \{S \subseteq A \mid \forall x_1, x_2 \in S. x_1 = x_2\}$$

What about the following M-modality?

$$\langle\!\langle x \leftarrow m \rangle\!\rangle \phi(x) \triangleq \bigsqcap_{x \in m} \phi(x)$$

Consider :

- $c_{\circlearrowleft} \triangleq \langle \lambda x.\delta\delta \rangle$, that yields no value when applied,
- the predicates $\top = x \mapsto \mathbf{1}$ and $\bot = x \mapsto \mathbf{0}$

Then we get:

$$\top \xrightarrow{c_{\circlearrowleft}} \bot \Leftrightarrow \forall c \in \mathbf{A}.1 \preccurlyeq \langle\!\langle x \leftarrow c_{\circlearrowleft} \cdot c \rangle\!\rangle 0$$

$$\Leftrightarrow \forall c \in \mathbf{A}.1 \preccurlyeq \bigsqcap_{x \in \forall (c_{\circlearrowleft} \cdot c)} .0 \Leftrightarrow 1 \preccurlyeq 1$$

✗

All good ?

Consider the **PCA** example, that is:

$$MA = \{S \subseteq A \mid \forall x_1, x_2 \in S. x_1 = x_2\}$$

What about the following M-modality?

$$\langle\!\!\langle\, x \leftarrow m \,\rangle\!\!\rangle\, \phi\,(x) \triangleq \bigsqcap_{x \in m} \phi\,(x)$$

Consider :

- $c_{\circlearrowleft} \triangleq \langle \lambda x.\delta\delta \rangle$, that yields no value when applied,
- the predicates $\top = x \mapsto \mathbf{1}$ and $\bot = x \mapsto \mathbf{0}$

Then we get:

$$\top \xrightarrow{c_{\circlearrowleft}} \bot \Leftrightarrow \forall c \in \mathbb{A}.\mathbf{1} \preccurlyeq \langle\!\!\langle\, x \leftarrow c_{\circlearrowleft} \cdot c \,\rangle\!\!\rangle\, \mathbf{0}$$

$$\Leftrightarrow \forall c \in \mathbb{A}.\mathbf{1} \preccurlyeq \bigsqcap_{x \in \nu(c_{\circlearrowleft} \cdot c)} .\mathbf{0} \Leftrightarrow \mathbf{1} \preccurlyeq \mathbf{1}$$

✗

Separator

A **separator** for $\Diamond$ is:

- a combinatory complete subset $\mathcal{S}$ of $\mathbb{A}$,
- such that, for every $c_f, c_a \in \mathcal{S}$, "*progress*" holds:

$$\left\langle\!\!\left\langle r \leftarrow c_f \cdot c_a \right\rangle\!\!\right\rangle \mathbf{0} \preccurlyeq \mathbf{0}$$

↣ *when progress holds for all codes,* $\mathbb{A}$ *is a separator*

Packing all the ingredients we define

Monadic Core

A **monadic core** is a tuple $(\mathbb{A}, \Omega, \Diamond, \mathcal{S})$, where:

- $\mathbb{A}$ is an MCA over a monad M,
- $\Diamond$ is an M-modality over $\mathbb{A}$
- $(\Omega, \preccurlyeq)$ is a complete **HpA**,
- $\mathcal{S}$ is a separator over them.

Example - PCA

Monadic core

- sub-singleton monad

$$MA = \{S \subseteq A \mid \forall x_1, x_2 \in S. x_1 = x_2\}$$

- modality :

??

↬ *if* $\phi_1 \xrightarrow{e} \phi_2$ *and* $\phi_1(c)$*, then* $\langle\!\langle x \leftarrow e \cdot c \rangle\!\rangle \phi_2$ *should express that* $e \cdot c$ *returns a valid code for* ϕ_2

- separator:

the whole set A

Example - PCA

Monadic core

- sub-singleton monad

$$MA = \{S \subseteq A \mid \forall x_1, x_2 \in S. x_1 = x_2\}$$

- modality :

??

↬ *if* $\phi_1 \xrightarrow{e} \phi_2$ *and* $\phi_1(c)$*, then* $\langle\!\langle\, x \leftarrow e \cdot c \,\rangle\!\rangle\, \phi_2$ *should express that* $e \cdot c$ *returns a valid code for* ϕ_2

- separator:

the whole set A

Example - PCA

Monadic core

- sub-singleton monad

$$MA = \{S \subseteq A \mid \forall x_1, x_2 \in S. x_1 = x_2\}$$

- modality :

$$\langle\!\!\langle\, x \leftarrow m \,\rangle\!\!\rangle\, \phi(x) \triangleq \bigsqcup_{x \in m} \phi(x)$$

$\not\hookrightarrow$ *if* $\phi_1 \xrightarrow{e} \phi_2$ *and* $\phi_1(c)$*, then* $\langle\!\!\langle\, x \leftarrow e \cdot c \,\rangle\!\!\rangle\, \phi_2$ *should express that* $e \cdot c$ *returns a valid code for* ϕ_2

- separator:

the whole set A

Example - PCA

Monadic core

- sub-singleton monad

$$MA = \{S \subseteq A \mid \forall x_1, x_2 \in S. x_1 = x_2\}$$

- modality :

$$\langle\!\!\langle\, x \leftarrow m \,\rangle\!\!\rangle\, \phi(x) \triangleq \bigsqcup_{x \in m} \phi(x)$$

$\rightarrowtail$ *if* $\phi_1 \xrightarrow{e} \phi_2$ *and* $\phi_1(c)$*, then* $\langle\!\!\langle\, x \leftarrow e \cdot c \,\rangle\!\!\rangle\, \phi_2$ *should express that* $e \cdot c$ *returns a valid code for* ϕ_2

- separator:

the whole set $\mathbb{A}$

Example - Classical realizability (1/2)

Quick reminder from Krivine realizability:

- fix a $\perp\!\!\!\perp$ ($\simeq$ "valid" computations) $\rightsquigarrow$ here, $\perp\!\!\!\perp \subseteq R$.
- orthogonality relation between $A \longrightarrow R$ and A

$$k \perp a \triangleq k(a) \in \perp\!\!\!\perp \qquad\qquad t \perp k \triangleq t(k) \in \perp\!\!\!\perp$$

  $\rightsquigarrow$ realizers of ϕ = computations in $\phi^{\perp\perp} = \{t \mid \forall k. k \perp \phi \Longrightarrow t \perp k\}$

- if $\perp\!\!\!\perp$ is non-empty, there exists $t \Vdash \bot$
  $\rightsquigarrow$ restrictions to *proof-like* realizers (no captured continuation).

Example - Classical realizability (2/2)

- continuation monad

$$MA = (A \to R) \to R.$$

- modality ($m \in \phi^{\perp\perp}$), with fixed $\perp\!\!\!\perp \subseteq R$:

$$\langle\!\!\langle x \leftarrow m \rangle\!\!\rangle \phi\,(x) \triangleq \bigsqcap_{k \in A \to R} \left(\left(\bigsqcap_{a \in A} (\phi(a) \sqsupset k \perp a) \right) \sqsupset m \perp k \right)$$

- separator:

proof-like: codes obtained by combinatorial completeness extended with e_{cc}

The induced Evidenced frame

Theorem (Evidenced Frame over Monadic Core)

Let $\mathcal{MC}_M = (\mathbb{A}, \Omega, \Diamond, \mathcal{S})$ be a monadic core. The triple $\left(\Omega^{\mathbb{A}}, \mathcal{S}, \cdot \dot{\rightarrow} \cdot\right)$, where

$$\phi_1 \overset{e}{\rightarrow} \phi_2 \triangleq \forall c \in \mathbb{A}. \phi_1(c) \preccurlyeq \left\langle\!\!\left\langle r \leftarrow e \cdot c \right\rangle\!\!\right\rangle \phi_2(r)$$

forms an evidenced frame.

Preuve : As usual! see the paper ;-)

Theorem

Let $\mathcal{MC}_M = (\mathbb{A}, \Omega, \Diamond, \mathcal{S})$ be a monadic core. Then the induced evidenced frame has an evidence $e \in \mathbb{A}$ such that $\top \overset{e}{\rightarrow} \bot$ iff $\mathbf{1} \preccurlyeq \mathbf{0}$.

formalization available : https://github.com/dominik-kirst/mca/

The induced Evidenced frame

Theorem (Evidenced Frame over Monadic Core)

Let $\mathcal{MC}_M = (\mathbb{A}, \Omega, \Diamond, \mathcal{S})$ be a monadic core. The triple $\left(\Omega^{\mathbb{A}}, \mathcal{S}, \cdot \dot{\rightarrow} \cdot\right)$, where

$$\phi_1 \xrightarrow{e} \phi_2 \triangleq \forall c \in \mathbb{A}. \phi_1(c) \preccurlyeq \left\langle\!\!\left\langle r \leftarrow e \cdot c \right\rangle\!\!\right\rangle \phi_2(r)$$

forms an evidenced frame.

Preuve : As usual! see the paper ;-)

Theorem

Let $\mathcal{MC}_M = (\mathbb{A}, \Omega, \Diamond, \mathcal{S})$ be a monadic core. Then the induced evidenced frame has an evidence $e \in \mathbb{A}$ such that $\top \xrightarrow{e} \bot$ iff $\mathbf{1} \preccurlyeq \mathbf{0}$.

ROCQ formalization available : https://github.com/dominik-kirst/mca/

Effectful Higher-Order Syntactic Realizability

Bonus : Syntactic Realizability

Different tradition : Gödel's *Dialectica*, Kreisel *modified realizability*, etc...

In a nutshell: from a derivation of $\Gamma \vdash A$, produce

- a type A^*
- a term t s.t. $\Gamma* \vdash t : A^*$
- which is a realizer of A

Goal #3: mimick the results obtains with MCA, and develop a unified framework for syntactic realizability models.

We aim at a framework that:

- models expressive systems like HOL or type theory
- supports effectful computational interpretations
- enables the perspective of syntactic proof transformations
- can be formalized in proof assistants

Bonus : Syntactic Realizability

Different tradition : Gödel's *Dialectica*, Kreisel *modified realizability*, etc...

In a nutshell: from a derivation of $\Gamma \vdash A$, produce

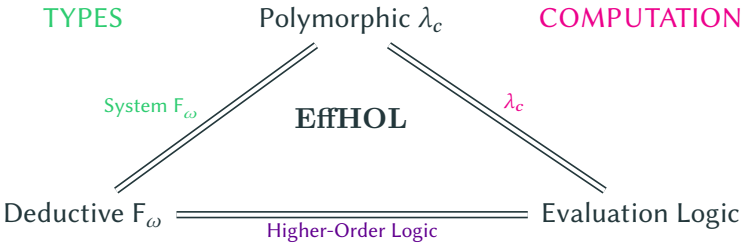
- a type A^*
- a term t s.t. $\Gamma* \vdash t : A^*$
- which is a realizer of A

Goal #3: mimick the results obtains with MCA, and develop a unified framework for syntactic realizability models.

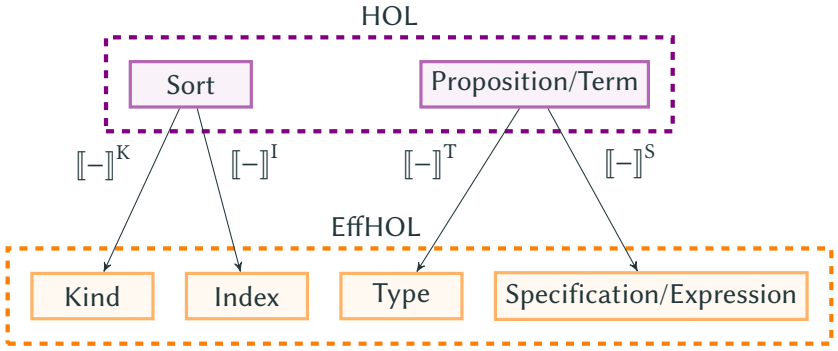
We aim at a framework that:

- models expressive systems like HOL or type theory
- supports effectful computational interpretations
- enables the perspective of syntactic proof transformations
- can be formalized in proof assistants

Contribution: Effectful Higher-Order Logic (EffHOL)



Realizability Translation



Theorem (Soundness)

If $\Gamma \vdash_{\mathsf{HOL}} \varphi$ then there is $p : M([\![\varphi]\!]^{\mathrm{T}})$ such that
$[\![\Gamma]\!]^{\mathrm{K}} \mid [\![\Gamma]\!]^{\mathrm{I}} \vdash_{\mathsf{EffHOL}} \langle x \leftarrow p \rangle\ [\![\varphi]\!]^{\mathrm{S}}\,(x)$.

Conclusion & bonus

Conclusion

Goal #1 : introduce an algebraic structure for realizability models that connects the logical and computational aspects.

- Any models induces a tripos (hence a topos)
- Implicative/arrow algebras : minimalistic axiomatization for *some* models

Evidenced Frame are super cool

An algebraic structure accounting for realizability models.

Conclusion

Goal #1 : introduce an algebraic structure for realizability models that connects the logical and computational aspects.

- Any models induces a tripos (hence a topos)
- Implicative/arrow algebras : minimalistic axiomatization for *some* models

Evidenced Frame are super cool

A | *flexible* / *uniform* / *complete* algebraic structure accounting for realizability models.

My favorite feature of EFs: robust interpretations

The effect of effects on constructivism
Cohen, Faro & Tate [2019]

A short story about Countable Choice

- Thm 1. - the realizability model induced by a PCA models CC.

$$\mathcal{Q}\!\!\rightarrow UFam(\mathcal{EF}^{\mathscr{C}}) \vDash CC$$

- Thm 2. - adding non-determinism makes it negate CC.

$$\mathcal{Q}\!\!\rightarrow UFam(\mathcal{EF}_{D}^{\mathscr{C}_{flip}}) \nvDash CC$$

- Thm 3. - adding states and using memoization restores CC.

$$\mathcal{Q}\!\!\rightarrow UFam(\mathcal{EF}_{D}^{\mathscr{C}_{flip,lookup}}) \vDash CC$$

My favorite feature of EFs: robust interpretations

The effect of effects on constructivism
Cohen, Faro & Tate [2019]

A short story about Countable Choice

- Thm 1. - the realizability model induced by a PCA models CC.

$$\looparrowright \mathit{UFam}(\mathcal{EF}^{\mathscr{C}}) \vDash \mathit{CC}$$

- Thm 2. - adding non-determinism makes it negate CC.

$$\looparrowright \mathit{UFam}(\mathcal{EF}_{D}^{\mathscr{C}_{\mathit{flip}}}) \nvDash \mathit{CC}$$

- Thm 3. - adding states and using memoization restores CC.

$$\looparrowright \mathit{UFam}(\mathcal{EF}_{D}^{\mathscr{C}_{\mathit{flip,lookup}}}) \vDash \mathit{CC}$$

$$\mathrm{UFam}(\mathcal{EF}_{D}^{\mathscr{C}_{\mathrm{flip,lookup}}}) \quad = \quad \textit{specification} \quad \neq \quad \textit{implementation}$$

Conclusion

Goal #2 : Smooth and uniform integration of useful computational effects.

- Monadic combinatory algebras: natural combinatorial view on monadic computations
- M-Modality to relate predicate on values and computations

MCA are really nice

Smoothly extend all the constructions/results for PCA.

Conclusion

Goal #3 : Investigate a syntactic counterpart of MCA.

- EffHOL : a Higher-Order Logic to reason on effectful programs
- realizability as translations from HOL to EffHOL

Realizability translation

If $\Gamma \vdash_{\mathsf{HOL}} \varphi$ then there is $p : M(\llbracket \varphi \rrbracket^{\mathrm{T}})$ such that:

$$\llbracket \Gamma \rrbracket^{\mathrm{K}} \mid \llbracket \Gamma \rrbracket^{\mathrm{I}} \vdash_{\mathsf{EffHOL}} \langle x \leftarrow p \rangle \ \llbracket \varphi \rrbracket^{\mathrm{S}}(x).$$

Future Directions

- Lots of questions on Evidenced Frames (study morphisms, relation to the induced topos, ...)
- Use EF to define *robust* realizability interpretations
- Extension of EffHOL to non-monadic effects ?
- Increased usability and scope of formalization
- Understand better the relation between typed/untyped realizability
- ...

Thank you!

Future Directions

- Lots of questions on Evidenced Frames (study morphisms, relation to the induced topos, ...)
- Use EF to define *robust* realizability interpretations
- Extension of EffHOL to non-monadic effects ?
- Increased usability and scope of formalization
- Understand better the relation between typed/untyped realizability
- ...

Thank you!

Bibliography


L. Cohen, É. Miquey, and R. Tate.
Evidenced frames: A unifying framework broadening realizability models.
In *Proceeding of LICS 2021.*


L. Cohen, A. Grunfeld, D. Kirst, and É. Miquey.
From Partial to Monadic: Combinatory Algebra with Effects.
In *Proceedings of FSCD 2025.*


L. Cohen, A. Grunfeld, D. Kirst, and É. Miquey.
Syntactic Effectful Realizability in Higher-Order Logic.
In *Proceedings of LICS 2025*, June.

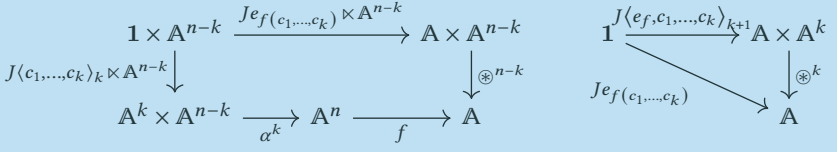
MCA, categorically (1/2)

Definition

For a Freyd category $\langle \mathbb{C}, \mathbb{K}, J \rangle$, an *applicative object* is an object $\mathbb{A} \in \mathbf{Obj}(\mathbb{C})$ equipped with an application morphism $\circledast \in \mathbb{K}(\mathbb{A} \times \mathbb{A}, \mathbb{A})$.

Definition

Given $\mathbb{A}$ applicative in $\langle \mathbb{C}, \mathbb{K}, J \rangle$, a morphism $f \in \mathbb{K}(\mathbb{A}^n, \mathbb{A})$ is $\mathbb{A}$-**computable** when, for all $k < n$, and all $c_1, \ldots, c_k \in \mathbb{C}(\mathbf{1}, \mathbb{A})$, there is a code $e_{f(c_1,\ldots,c_k)} \in \mathbb{C}(\mathbf{1}, \mathbb{A})$ s.t.:



MCA, categorically (2/2)

An $\mathbb{A}$-**monomial** is a morphism in $\mathbb{K}(\mathbb{A}^n, \mathbb{A})$ defined using only projections, morphisms in $\mathbb{C}(\mathbf{1}, \mathbb{A})$, and application $\circledast$, used in an applicative order.

Definition

An applicative object $\mathbb{A}$ is called a *combinatory object* when all positive $\mathbb{A}$-monomials are $\mathbb{A}$-computable.

Theorem

$\mathbb{A}$ is an MCA over M iff if it is a combinatory object in $\mathbf{Set}_M$.