Evidenced Frames : A Unifying Framework Broadening Realizability Models
L.Cohen, E.Miquey and R.Tate

Félix Castro

23 juin 2022

Table of Contents

Disclaimer

The first part of this presentation is merely a copy-cut of a previous presentation of Alexandre Miquel while the second part is strongly inspired by a presentation of Étienne Miquey.

Evidenced frames

Question : What is an evidenced frames ? Why should we introduce such a structure ?

Answer :

- "[...]we introduce evidenced frames : a general-purpose framework for building realizability models [...] "
- " evidenced frames form a unifying framework for (realizability) models of higher-order dependent predicate logic."

Wait... It sounds familiar, doesn't it ?

Reminders

Implicative structures

Definition (Implicative structures)

An implicative structure is a triple $(\mathcal{A}, \preccurlyeq, \rightarrow)$ where

1. $(\mathcal{A}, \preccurlyeq)$ is a complete lattice
2. $(\rightarrow) : \mathcal{A}^2 \rightarrow \mathcal{A}$ is a binary operation such that
    1. $a' \preccurlyeq a, b \preccurlyeq b'$ entails $(a \rightarrow b) \preccurlyeq (a' \rightarrow b')$
    2. $\bigcurlywedge_{b \in B}(a \rightarrow b) = a \rightarrow \bigcurlywedge_{b \in B} b$

Example (From a total combinatory algebra)

From a total combinatory algebra $\mathcal{T} = (\mathcal{T}, ., \mathbf{k}, \mathbf{s})$, one can define the implicative structure $(\mathcal{P}(\mathcal{T}), \subseteq, \rightarrow)$ where

$$X \rightarrow Y \quad = \quad \{p \mid \forall x \in X\, p.x \in Y\}$$

Implicative algebras

Definition (Implicative algebras)

An implicative algebra is a quadruple $(\mathcal{A}, \preccurlyeq, \to, \mathcal{S})$ where

1. $(\mathcal{A}, \preccurlyeq, \to)$ is an implicative structure
2. $\mathcal{S} \subseteq \mathcal{A}$ is a separator i.e :
   1. if $a \preccurlyeq b$ and $a \in \mathcal{S}$ then $b \in \mathcal{S}$
   2. $\mathbf{K}^{\mathcal{A}} = \bigcurlywedge_{a,b \in \mathcal{A}} (a \to b \to a) \in \mathcal{S}$ and
      $\mathbf{S}^{\mathcal{A}} = \bigcurlywedge_{a,b,c \in \mathcal{A}} ((a \to b \to c) \to (a \to b) \to a \to c) \in \mathcal{S}$
   3. if $(a \to b) \in \mathcal{S}$ and $a \in \mathcal{S}$, then $b \in \mathcal{S}$

Example (From a total combinatory algebra)

From a total combinatory algebra $\mathcal{T} = (\mathcal{T}, ., \mathbf{k}, \mathbf{s})$, one can define the implicative algebra $(\mathcal{P}(\mathcal{T}), \subseteq, \to, \mathcal{P}(\mathcal{T}) \setminus \{\emptyset\})$.

Set-based tripos

Definition (**Set**-based tripos)

A **Set**-based tripos is a functor $\mathbf{P} : \mathbf{Set}^{\mathbf{op}} \to \mathbf{HA}$ such that

1. for each map $f : X \to Y$ (in **Set**), the associated map $\mathbf{P}(f) : \mathbf{P}(Y) \to \mathbf{P}(X)$ has left and right adjoints $\exists f, \forall f : \mathbf{P}(X) \to \mathbf{P}(Y)$ (in **Pos**)
2. Beck-Chevalley condition : Each pullback square in **Set**(on the lhs) induces the following two commutative diagrams in **Pos**(on the rhs) :

$$\begin{array}{ccc} X & \xrightarrow{f_1} & X_1 \\ {\scriptstyle f_2}\downarrow & \lrcorner & \downarrow{\scriptstyle g_1} \\ X_2 & \xrightarrow[g_2]{} & Y \end{array} \quad \Rightarrow \quad \begin{array}{ccc} \mathbf{P}X & \xrightarrow{\exists f_1} & \mathbf{P}X_1 \\ {\scriptstyle \mathbf{P}f_2}\uparrow & & \uparrow{\scriptstyle \mathbf{P}g_1} \\ \mathbf{P}X_2 & \xrightarrow[\exists g_2]{} & \mathbf{P}Y \end{array} \quad \begin{array}{ccc} \mathbf{P}X & \xrightarrow{\forall f_1} & \mathbf{P}X_1 \\ {\scriptstyle \mathbf{P}f_2}\uparrow & & \uparrow{\scriptstyle \mathbf{P}g_1} \\ \mathbf{P}X_2 & \xrightarrow[\forall g_2]{} & \mathbf{P}Y \end{array}$$

3. The functor $\mathbf{P} : \mathbf{Set}^{\mathbf{op}} \to \mathbf{HA}$ has a generic predicate $tr_\Sigma \in \mathbf{P}\Sigma$ (for some set Σ), i.e. such that for all sets X, the following map is surjective :

$$\begin{array}{ccc} \Sigma^X & \to & \mathbf{P}X \\ \sigma & \mapsto & \mathbf{P}\sigma(tr_\sigma) \end{array}$$

From implicative algebra to tripos

Let $(\mathcal{A}, \preccurlyeq, \to, \mathcal{S})$ be an implicative algebra

- For each set X, we write $\mathbf{P}_{\mathcal{A}}X := \mathcal{A}^X/\mathcal{S}[X]$ the poset reflection of the preordered set $(\mathcal{A}^X, \vdash_{\mathcal{S}[X]})$ where

$$a \vdash_{\mathcal{S}[X]} b \quad \text{iff} \quad \bigwedge_{x \in X}(a_x \to b_x) \in \mathcal{S}$$

- For each map $f : X \to Y$, we write $\mathbf{P}f : \mathbf{P}Y \to \mathbf{P}X$ the unique map that factors the map $\mathcal{A}^f = (a \mapsto a \circ f) : \mathcal{A}^Y \to \mathcal{A}^X$ through the quotients $\mathbf{P}Y := \mathcal{A}^Y/\mathcal{S}[X]$ and $\mathbf{P}X := \mathcal{A}^X/\mathcal{S}[X]$.

Theorem (Implicative tripos)

The functor $\mathbf{P}_{\mathcal{A}} : \mathbf{Set}^{\mathbf{op}} \to \mathbf{HA}$ *is a* $\mathbf{Set}$*-based tripos.*

From tripos to implicative algebra

Let $\mathbf{P}$ be a tripos and $tr_\Sigma \in \mathbf{P}\Sigma$ a generic predicate

1. The whole structure of $\mathbf{P}$ can be derived from Σ equipped with two (non canonical) operations

$$\dot{\to} : \Sigma^2 \to \Sigma \qquad\qquad \dot{\bigwedge} : \mathcal{P}(\Sigma) \to \Sigma$$

and a specific subset $\Phi \subseteq \Sigma$.

2. Using domain-theoretic techniques, Miquel turned this poorly behaved structure $(\Sigma, \dot{\to}, \dot{\bigwedge}, \Phi)$ into an implicative algebra $\mathcal{A} = (\mathcal{A}, \preccurlyeq, \to, \mathcal{S})$ such that $(\mathcal{A}, \to, \bigwedge\!\!\!\!\bigwedge, \mathcal{S})$ contains the information needed to retrieve the tripos
3. In particular, it was shown that

$$\mathbf{P}X \simeq \mathcal{A}^X / \mathcal{S}[X]$$

and that $\mathbf{P}$ is isomorphic to the implicative tripos constructed from $\mathcal{A}$.

All in all

- **Implicative algebras :** a simple algebraic structure that encompasses
  - Complete Heyting Algebras
  - Partial Combinatory Algebras
  - Ordered Combinatory Algebras
  - Abstract Krivine Structure
- Implicative algebras can be used to construct implicative triposes
- Implicative triposes encompass all **Set**-based triposes

Theorem (Completeness/Representation)

Every **Set***-based tripos is isomorphic to an implicative tripos.*

Evidenced frames ? ?

Question : What is an evidenced frames ? Why should we introduce such a structure ?

Answer :You interrupted me ! I was saying...

- "[...]we introduce evidenced frames : a general-purpose framework for building realizability models that support diverse effectful computations."
- "[...]evidenced frames form a unifying framework for (realizability) models of higher-order dependent predicate logic [...]"
- "[...]the existing completeness construction for implicative algebras [...] factors through our simpler construction."

Sorry... Sorry... Let's see it !

The influence of side-effects on realizability interpretation

The influence of control operators on the logic

Definition

Let $(\mathcal{A}, \preccurlyeq, \rightarrow, \mathcal{S})$ an implicative algebra. The separator $\mathcal{S}$ is said to be classical if

$$\text{cc}^{\mathcal{A}} = \bigcurlywedge_{a,b \in \mathcal{A}} (((a \rightarrow b) \rightarrow a) \rightarrow a) \in \mathcal{S}.$$

Example

An implicative algebra is classical if and only if it is obtained from an Abstract Krivine Structure.

Theorem

Every classical tripos is a classical implicative tripos and therefore a Krivine tripos.

TO-GO : If your realizibility model is classical, it can be obtained from an **AKS** (i.e a programming language containing control operators).

The effect of exceptions on the logic

Definition

Markov's Principle Markov's Principle is a weak classical scheme stating that

$$\neg\neg\exists x\, A(x) \to \exists x\, A(x)$$

for every decidable formula $A(x)$.

Herbelin showed that adding exceptions in a programming language can lead to realizability models that satisfy Markov's Principle.

Question : Is Markov's Principle satisfied in all realizability models obtained from a (terminating) programming language that can handle exceptions ?

The Effects of Effects on Countable Choice

Definition

In Higher Order Logic (with a sort $\mathbb{N}$ for natural number), Countable Choice (CC) at sort τ is the formula

$$\forall R : \mathbb{N} \times \tau \to \mathrm{Prop}. (\forall x : \mathbb{N}. \exists y : \tau. R(x, y)) \Rightarrow \exists f : \mathbb{N} \to \tau. \forall x : \mathbb{N}. R(x, f(x))$$

Cohen, Abreu Faro and Tate showed that

1. a realizability model obtained from a (deterministic) PCA satisfies CC
2. adding non determinism to the PCA negates CC
3. adding states restores CC.

Goal Formalizing this result in a general algebraic framework for realizability.

Solution Evidenced Frames !

An intuition (by E.Miquey)

Realizability is a 3-steps recipe :

1. formulas (a.k.a types)
   ex : second order logic, HOL, ZF...
2. a computational system (a.k.a your favorite calculus)
   ex : some λ-calculus, a combinatory algebra...
3. formulas interpretation (a.k.a truth values)
   when a program t realizes a formula A ?

Evidenced frames : the definition

An Evidenced Frame is the data of a triple $(\Phi, E, \cdot \dot{\rightarrow} \cdot)$ where

1. Φ is a set of propositions (a.k.a formulas or types)
2. E is a set of evidences (a.k.a a set of programs)
3. $(\cdot \dot{\rightarrow} \cdot) \subseteq \Phi \times E \times \Phi$ is the realizability relation (when a program t realizes $\phi_1 \rightarrow \phi_2$)

along with some other properties (see next slides). We note

$$\phi_1 \xrightarrow{e} \phi_2 \quad \text{for} \quad (\phi_1, e, \phi_2) \in \cdot \dot{\rightarrow} \cdot.$$

Evidences frames : the definition (2)

The Evendiced Frame $(\Phi, E, \cdot \dot{\rightarrow} \cdot)$ comes with :

- **Reflexivity** e_{id} such that
  $\phi \overset{e_{id}}{\rightarrow} \phi$
- **Transitivity** an operator ; of type $E \times E \rightarrow E$ such that
  $\phi_1 \overset{e_1}{\rightarrow} \phi_2 \wedge \phi_2 \overset{e_2}{\rightarrow} \phi_3 \Rightarrow \phi_1 \overset{e_1 ; e_2}{\rightarrow} \phi_3$
- **Top** $\top$ and $e_\top$ such that
  $\phi \overset{e_\top}{\rightarrow} \top$
- **Conjunction** $\wedge : \Phi \times \Phi \rightarrow \Phi$, $\langle \cdot, \cdot \rangle : E \times E \rightarrow E$ and $e_{fst}, e_{snd} \in E$ s.t :
  $\phi_1 \wedge \phi_2 \overset{e_{fst}}{\rightarrow} \phi_1 \quad \phi \overset{e_1}{\rightarrow} \phi_1 \wedge \phi \overset{e_2}{\rightarrow} \phi_2 \Rightarrow \phi \overset{\langle e_1, e_2 \rangle}{\rightarrow} \phi_1 \wedge \phi_2$
  $\phi_1 \wedge \phi_2 \overset{e_{snd}}{\rightarrow} \phi_2$
- **Universal implication** $\supset : \Phi \times \mathcal{P}(\Phi) \rightarrow \Phi$, $\lambda : E \rightarrow E$ and $e_{eval} \in E$ s.t :
  $(\forall \phi \in \vec{\phi}.\, \phi_1 \wedge \phi_2 \overset{e}{\rightarrow} \phi) \Rightarrow \phi_1 \overset{\lambda e}{\rightarrow} \phi_2 \supset \vec{\phi}$
  $\forall \phi \in \vec{\phi}.\, \phi_1 \supset \vec{\phi} \wedge \phi_1 \overset{e_{eval}}{\rightarrow} \phi$

$(\vec{\phi} \subseteq \Phi)$

Example : from implicative algebra to evidenced frame

Let $\mathcal{A} = (\mathcal{A}, \preccurlyeq, \to, \mathcal{S})$ be an implicative algebra.
The triple $(\mathcal{A}, \mathcal{S}, \cdot \dot{\to} \cdot)$ where

$$\phi_1 \xrightarrow{e} \phi_2 \quad \equiv \quad e \preccurlyeq \phi_1 \to \phi_2$$

defined an evidenced frame $\mathcal{E}_{\mathcal{A}}$. The needed additional structure is obtained as follow :

$$\begin{array}{ll}
e_{id} \equiv (\lambda x.x)^{\mathcal{A}} & e_1; e_2 \equiv (\lambda x.e_2(e_1 x))^{\mathcal{A}} \\
\top \equiv \top^{\mathcal{A}} & e_\top \equiv (\lambda x.x)^{\mathcal{A}} \\
\phi_1 \wedge \phi_2 \equiv \bigcurlywedge_{a \in \mathcal{A}} ((\phi_1 \to \phi_2 \to a) \to a) & \langle e_1, e_2 \rangle \equiv (\lambda xy.y(e_1 x)(e_2 x))^{\mathcal{A}} \\
e_{fst} \equiv (\lambda x.x(\lambda x_1 x_2.x_1))^{\mathcal{A}} & e_{snd} \equiv (\lambda x.x(\lambda x_1 x_2.x_2))^{\mathcal{A}} \\
\phi \supset \vec{\phi} \equiv \phi \to \bigcurlywedge \vec{\phi} & (\vec{\phi} \subseteq \Phi) \\
\lambda e \equiv (\lambda xy.e\langle x, y \rangle)^{\mathcal{A}} & e_{eval} \equiv (\lambda x.(e_{fst} x)(e_{snd} x))^{\mathcal{A}}
\end{array}$$

From evidenced frame to tripos

Let $\mathcal{E} = (\Phi, E, \cdot \dot{\rightarrow} \cdot)$ be an evidenced frame

- For each set X, we write $\mathbf{P}_{\mathcal{E}}X := \Phi^X / \rightarrow [X]$ the poset reflection of the preordered set $(\Phi^X, \vdash_{\rightarrow[X]})$ where

$$a \vdash_{\rightarrow[X]} b \quad \text{iff} \quad \exists e \in E. \forall x \in X. (a_x \overset{e}{\rightarrow} b_x)$$

- For each map $f : X \rightarrow Y$, we write $\mathbf{P}f : \mathbf{P}Y \rightarrow \mathbf{P}X$ the unique map that factors the map $\Phi^f = (a \mapsto a \circ f) : \Phi^Y \rightarrow \Phi^X$ through the quotients $\mathbf{P}Y := \Phi^Y / \rightarrow [Y]$ and $\mathbf{P}X := \Phi^X / \rightarrow [X]$.

Theorem (Tripos from evidenced frame)

The functor $\mathbf{P}_{\mathcal{E}} : \mathbf{Set}^{\mathbf{op}} \rightarrow \mathbf{HA}$ *is a* **Set**-*based tripos.*

Remark

This is a factorization of the implicative algebra to tripos construction : for $\mathcal{A}$ an implicative algebra, $\mathbf{P}_{\mathcal{A}}$ is isomorphic to $\mathbf{P}_{\mathcal{E}_{\mathcal{A}}}$.

Incorporating effects in evidenced frames

An other 3 steps recipe

- **PCA :** definition via functional completeness (and not via **k,s**)
- **Computational system :** Extension of **PCA** in a way to incorporate mutable state, non determinism reduction, failure...
- **Translation :** from computational systems to evidenced frames.

Partial Applicative Structure

Definition (partial applicative structures)

A partial applicative structure (**PAS**) is a set of "codes" $\mathcal{C}$ equipped with a partial binary function **app**.

We use the notation $c_f \cdot c_a \downarrow c_r$ to denote that $(c_f, c_a) \in \mathrm{Dom}(\mathbf{app})$ and $\mathbf{app}(c_f, c_a) = c_r$.

Expressions (or the way of representing open terms)

From a fixed **PAS** we generate the set of expressions (a la *de Brujin*) by the following grammar :

$$e \quad ::= \quad i \in \mathbb{N} \mid c \in \mathcal{C} \mid e \cdot e$$

Substitutions $e[c_a]$ are defined inductively :

$$\begin{array}{rcl} 0[c_a] & = & c_a \\ (i+1)[c_a] & = & i \\ c[c_a] & = & c \\ (e_f \cdot e_a)[c_a] & = & e_f[c_a] \cdot e_a[c_a] \end{array}$$

Reduction $e \downarrow c_r$ is extended to expressions :

$$\frac{}{c \downarrow c} \qquad\qquad \frac{e_f \downarrow c_f \quad e_a \downarrow c_a \quad c_f \cdot c_a \downarrow c_r}{e_f \cdot e_a \downarrow c_r}$$

Functional completeness and **PCA**

The set of expressions with codes of free variables less than n :

$$E_n = \{e \,|\, \forall i \in e.\, i < n\}$$

Definition (Functional completeness)

A **PAS** is functionally complete if for all n, there exists a map $c_{\lambda^n.\_} : E_{n+1} \to \mathcal{C}$ such that

$$c_{\lambda^{n+1}.e} \cdot c_a \downarrow c_{\lambda^n.e[c_a]}$$
$$c_{\lambda^0.e} \cdot c_a \downarrow c_r \Leftrightarrow e[c_a] \downarrow c_r$$

$c_{\lambda^n.e}$ should be think as the result of abstracting the all the free variable in the expression e (that contains only variables $< n$).

Definition (partial combinatory algebra)

A partial combinatory algebra (**PCA**) is a functionally complete **PAS**.

Example

- **PCA** generated by **k,s**

Adding mutable states : computational system

Recall, the set of expressions

$$e \quad ::= \quad i \in \mathbb{N} \mid c \in \mathcal{C} \mid e \cdot e$$

A computational system **C** will be the data of

- a set of codes $\mathcal{C}$ (from which we generate the set of expressions)
- a preorder Σ of states
- maps $c_{\lambda^n.\_} : E_{n+1} \to \mathcal{C}$ for all n
- a reduction relation $e \downarrow^{\sigma}_{\sigma'} c$ (in the state σ, e can reduce to c and go to the state σ')
- a termination relation $e \downarrow^{\sigma}$ (in the state σ, e terminates but potentially does not reduce to any code)

that satisfy the following property

Adding mutable states : computational system (2)

The reduction relations should satisfy

$$\frac{}{c \downarrow^{\sigma}_{\sigma} c} \qquad \frac{e_f \downarrow^{\sigma}_{\sigma'} c_f \quad e_a \downarrow^{\sigma'}_{\sigma''} c_a \quad c_f \cdot c_a \downarrow^{\sigma''}_{\sigma'''} c_r}{e_f \cdot e_a \downarrow^{\sigma}_{\sigma'''} c_r} \qquad \frac{}{c \downarrow^{\sigma}}$$

$$\frac{e_f \downarrow^{\sigma} \quad \forall \sigma', c_f.\, e_f \downarrow^{\sigma}_{\sigma'} c_f \Rightarrow (e_a \downarrow^{\sigma'} \wedge \forall \sigma'', c_a.\, e_a \downarrow^{\sigma'}_{\sigma''} c_a \Rightarrow c_f \cdot c_a \downarrow^{\sigma''})}{e_f \cdot e_a \downarrow^{\sigma}}$$

and a property of preservation

$$c_f \cdot c_a \downarrow^{\sigma}_{\sigma'} c_r \Rightarrow \sigma \leq \sigma'.$$

The maps $c_{\lambda^n.\_}$ should satisfy a generalization of functional completeness :

$$c_{\lambda^{n+1}.e} \cdot c_a \downarrow^{\sigma}_{\sigma'} c_r \Rightarrow \sigma' = \sigma \wedge c_r = c_{\lambda^n.e[c_a]} \qquad \text{(deterministic, } \beta \text{ and stable state)}$$

$$c_{\lambda^{n+1}.e} \cdot c_a \downarrow^{\sigma} \qquad \text{(termination of non fully applied expression)}$$

$$c_{\lambda^0.e} \cdot c_a \downarrow^{\sigma}_{\sigma'} c_r \Rightarrow e[c_a] \downarrow^{\sigma}_{\sigma'} c_r \qquad (\beta)$$

$$e[c_a] \downarrow^{\sigma} \Rightarrow c_{\lambda^0.e} \cdot c_a \downarrow^{\sigma} \qquad (\beta)$$

Examples of instructions

Example

- **Non determinism.** $\mathbf{C}_{\mathrm{flip}}$ is obtained by adding the instruction flip :

$$\frac{}{\mathrm{flip} \cdot c \downarrow^{\sigma}} \qquad \frac{}{\mathrm{flip} \cdot c \downarrow^{\sigma} c_{\lambda^1.0}} \qquad \frac{}{\mathrm{flip} \cdot c \downarrow^{\sigma} c_{\lambda^1.1}}$$

- **Mutable states.** $\mathbf{C}_{\mathrm{lookup}}$ is obtained by taking Σ as finite maps from $\mathbb{N}$ to $\mathcal{C}$ ordered by inclusion and adding instructions lookup_n :

$$\frac{}{\mathrm{lookup}_n \cdot c \downarrow^{\sigma}} \qquad \frac{n \mapsto c' \in \sigma}{\mathrm{lookup}_n \cdot c \downarrow^{\sigma}_{\sigma} c'} \qquad \frac{\nexists c'.\, n \mapsto c' \in \sigma}{\mathrm{lookup}_n \cdot c \downarrow^{\sigma}_{\sigma, n \mapsto c} c}$$

- **Failure.** $\mathbf{C}_{\mathrm{fail}}$ is obtained by adding the instruction fail :

$$\frac{}{\mathrm{fail} \cdot c \downarrow^{\sigma}}$$

From $\mathcal{CS}$ to $\mathcal{EF}$: the set of propositions

Recall Kleene's realizability. From a **PCA** $\mathcal{C}$, we can define an evidenced frame where the set Φ of propositions is $\mathcal{P}(\mathcal{C})$
In a computational system **C**, we need to consider states. Therefore, a proposition ϕ should be a set of pairs of a code and a state, i.e it should be included in $\mathcal{C} \times \Sigma$. We write

$$\phi^\sigma(c) \text{ for } (c, \sigma) \in \phi.$$

We will restrict the set of propositions to futur stable predicate, which means that ϕ will be a proposition if

$$\phi^\sigma(c) \wedge \sigma < \sigma' \Rightarrow \phi^{\sigma'}(c).$$

From $\mathcal{CS}$ to $\mathcal{EF}$: the realizability relation

Recall Kleene's realizability. From a **PCA** $\mathcal{C}$, we can define an evidenced frame where $\cdot \overset{\cdot}{\to} \cdot$ is defined as

$$\phi_1 \overset{c}{\to} \phi_2 \quad \equiv \quad \forall c_a \in \phi_1 .\, \exists c_r .\, c \cdot c_a \downarrow c_r \wedge c_r \in \phi_2$$

In a computational system **C**, we need to take into account that the relation can be non deterministic. There is 2 possible interpretations of non determinism :

- **the demonic one.** (Intuition : every reduction terminates in the pole)

$$c_f \cdot c_a \downarrow\!\downarrow^{\sigma}_{\mathcal{D}} \phi \quad \equiv \quad c_f \cdot c_a \downarrow^{\sigma} \wedge \forall c_r, \sigma' .\, c_f \cdot c_a \downarrow^{\sigma}_{\sigma'} c_r \Rightarrow \phi^{\sigma'}(c_r)$$

- **the angelic one.** (Intuition : at least one reduction terminates in the pole)

$$c_f \cdot c_a \downarrow\!\downarrow^{\sigma}_{\mathcal{A}} \phi \quad \equiv \quad c_f \cdot c_a \downarrow^{\sigma} \wedge \exists c_r, \sigma' .\, c_f \cdot c_a \downarrow^{\sigma}_{\sigma'} c_r \Rightarrow \phi^{\sigma'}(c_r)$$

Therefore, there is (at least) two possible ways to define $\cdot \overset{\cdot}{\to} \cdot$. Let $\mathcal{B}$ be an interpretation of determinism ($\mathcal{D}$ or $\mathcal{A}$), we can define :

$$\phi_1 \overset{e}{\to} \phi_2 \quad \equiv \quad \forall c, \sigma .\, \phi_1^{\sigma}(c) \Rightarrow e \cdot c \downarrow\!\downarrow^{\sigma}_{\mathcal{B}} \phi_2$$

From $\mathcal{CS}$ to $\mathcal{EF}$: the set of evidences

Recall Kleene's realizability. From a **PCA** $\mathcal{C}$, we can define an evidenced frame where the set of evidences if $\mathcal{C}$.
To maintain consistency in presence of failure, we need to restrict the set of evidences (indeed, check that $\top \overset{\text{fail}}{\to} \bot$ with a demonic interpretation of non determinism).
To this hand, we restrict the evidences to a distinguished subset $\mathcal{S}$ that should be

1. **functionally complete.** (i.e stable for all the maps $c_{\lambda^n.\_}$)
2. **closed under reduction.**
3. **Progress.** All code in $\mathcal{S}$ should satisfy *progress*, if it terminates, it should reduce to a code :

$$\forall \sigma, c_f, c_a \in \mathcal{S}.\, c_f \cdot c_a \downarrow^{\sigma} \Rightarrow \exists \sigma', c_r.\, c_f \cdot c_a \downarrow^{\sigma}_{\sigma'} .c_r$$

Two examples are the separator $\mathcal{S}_\top$ containing all terms (if it exists) and the separator $\mathcal{S}_\lambda$ generated by functional completeness.

At last : from $\mathcal{CS}$ to $\mathcal{EF}$

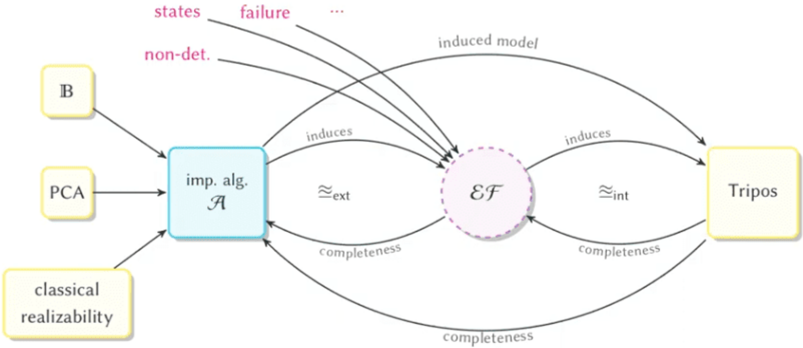
If $\mathfrak{C}$ is the data of a computational system (a separator and an interpretation of non determinism), then $\mathcal{E}_{\mathfrak{C}}$ defined previously is an evidenced frames.
Question : How construct an implicative algebra from such a structure ?

An example of the robustness of this framework

1. if **P** is a tripos obtained from an $\mathcal{EF}$ generated from a **PCA**, it will satisfy countable choice.
2. if **P** is a tripos obtained from a $\mathcal{EF}$ generated from a $\mathcal{CS}$ containing flip (and no other effects) by using demonic non determinism and the separator $\mathcal{S}_\top$, it will satisfy the negation of countable choice.
3. if **P** is a tripos obtained from a $\mathcal{EF}$ generated from a $\mathcal{CS}$ containing $\mathrm{flip}, \mathrm{lookup}_n$ (and no other effects) by using demonic non determinism and the separator $\mathcal{S}_\top$, it will satisfy countable choice.
4. other examples in the paper (notably the link between angelic non determinism and forcing)

Conclusion (and what I didn't talk about)

The final picture : by E.Miquey



Slogan

Tripos = evidenced frame that has forgotten its evidence.