Classical realizability and side-effects

Étienne Miquey[1,2]

[1] Équipe πr^2 (INRIA), IRIF, Université Paris-Diderot
[2] IMERL, Fac. de Ingeniería, Universidad de la República

17 Novembre 2017

Part I

-

What is this thesis about?

A tricky question

Every Ph.D. student has been asked a thousand times:

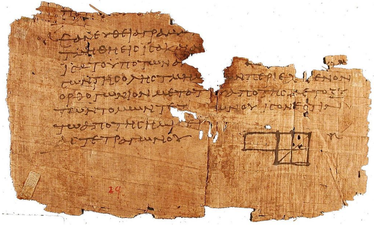
> *"What is the title of your thesis?"*

In my case, the next questions:

- classical?
- realizability?
- side-effects?
- *What does it have to do with logic/mathematics/computer science?*

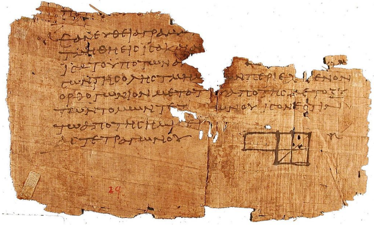
Proofs

A (very) old one:



Proofs

A (very) old one:

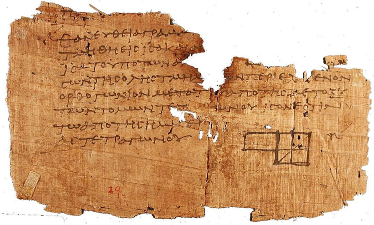


An easy one:

Plato is a cat.
All cats like fish.
Therefore, Plato likes fish .

Proofs

A (very) old one:



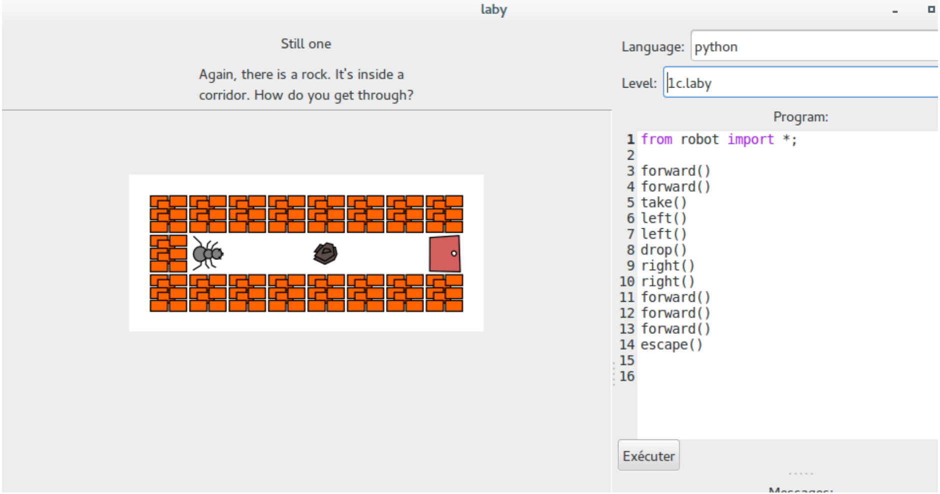
An easy one:

Plato is a cat.
All cats like fish.
Therefore, Plato likes fish .

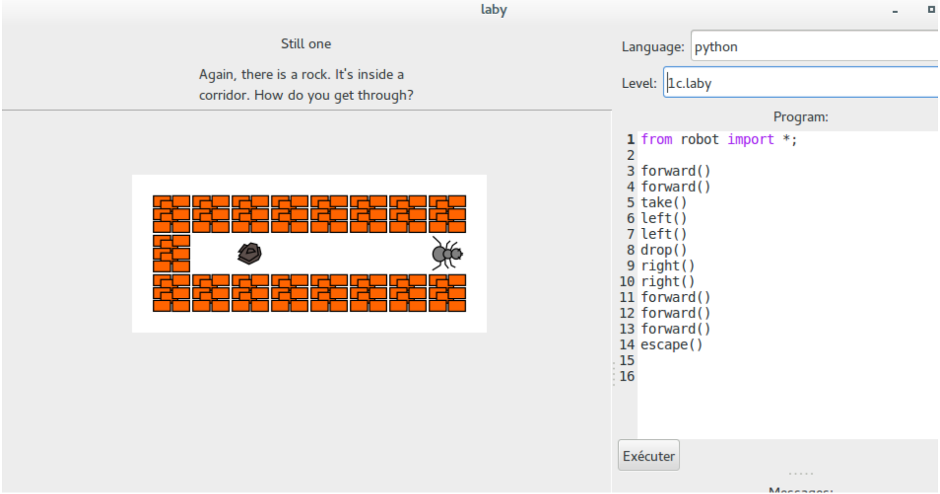
Intuitively:

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

Programs

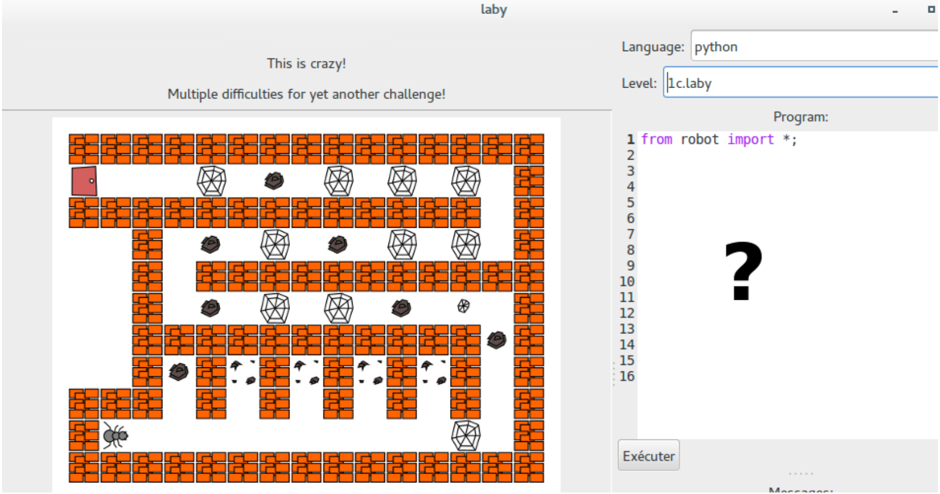

laby
Still one
Again, there is a rock. It's inside a corridor. How do you get through?
Language: python
Level: 1c.laby
Program:
from robot import *;
forward()
forward()
take()
left()
left()
drop()
right()
right()
forward()
forward()
forward()
escape()
Exécuter

Programs

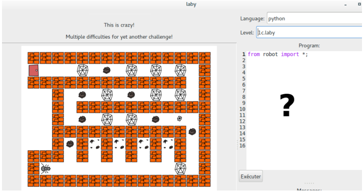

laby
Still one
Again, there is a rock. It's inside a corridor. How do you get through?
Language: python
Level: 1c.laby
Program:
from robot import *;

forward()
forward()
take()
left()
left()
drop()
right()
right()
forward()
forward()
forward()
escape()
Exécuter

Programs


laby
This is crazy!
Multiple difficulties for yet another challenge!
Language: python
Level: 1c.laby
Program:
from robot import *;
?
Exécuter

Programs



Think of it as a **recipe** (algorithm) to draw a computation forward.

Intuitively:

from a set of **inputs**

apply **instructions**

to obtain the **output**

So ?

Proof:

from a set of **hypotheses**

apply **deduction rules**

to obtain a **theorem**

Program:

from a set of **inputs**

apply **instructions**

to obtain the **output**

Curry-Howard

(On well-chosen subsets of mathematics and programs)

That's the same thing!

Proof trees

Sequent:

$$\boxed{\Gamma \vdash A}$$

Deduction rules:

$$\frac{A \in \Gamma}{\Gamma \vdash A}\ (\mathrm{Ax}) \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ (\Rightarrow_I) \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\ (\Rightarrow_E)$$

Example:

Proof trees

Sequent:

$$\boxed{\Gamma \vdash A}$$

Deduction rules:

$$\frac{A \in \Gamma}{\Gamma \vdash A}\ (\mathrm{Ax}) \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ (\Rightarrow_I) \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\ (\Rightarrow_E)$$

Example:

Plato is a cat.

If Plato is cat, Plato likes fish.

Therefore, $\underbrace{\textit{Plato likes fish}}_{\text{Conclusion}}$.

$$\dfrac{\dfrac{(A \Rightarrow B) \in \Gamma}{\Gamma \vdash A \Rightarrow B}\ (\mathrm{Ax}) \qquad \dfrac{B \in \Gamma}{\Gamma \vdash B}\ (\mathrm{Ax})}{\Gamma \vdash B}\ (\Rightarrow_E)$$

Proof trees

Sequent:

$$\boxed{\Gamma \vdash A}$$

Deduction rules:

$$\frac{A \in \Gamma}{\Gamma \vdash A}\,(\mathrm{Ax}) \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\,(\Rightarrow_I) \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\,(\Rightarrow_E)$$

Example:

Hyp. $\left\{\begin{array}{l}\textit{Plato is a cat.}\\ \textit{If Plato is cat, Plato likes fish.}\end{array}\right.$

Therefore, $\underbrace{\textit{Plato likes fish}}_{\text{Conclusion}}$.

$$\cfrac{\cfrac{(A \Rightarrow B) \in \Gamma}{\Gamma \vdash A \Rightarrow B}\,(\mathrm{Ax}) \qquad \cfrac{B \in \Gamma}{\Gamma \vdash B}\,(\mathrm{Ax})}{\Gamma \vdash B}\,(\Rightarrow_E)$$

Proofs-as-programs

The Curry-Howard correspondence

Mathematics	**Computer Science**
Proofs	Programs
Propositions	Types
Deduction rules	Typing rules
$\dfrac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}\ (\Rightarrow_E)$	$\dfrac{\Gamma \vdash t : A \to B \quad \Gamma \vdash u : A}{\Gamma \vdash t\,u : B}\ (\to_E)$

Benefits:

Program your proofs!	*Prove your programs!*

Proofs-as-programs

Limitations

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

↬ *We want more !*

Extending Curry-Howard

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

1990: Griffin discovered that `call/cc` can be typed by Peirce's law
(well-known fact: Peirce's law $\Rightarrow A \vee \neg A$)

Classical Curry-Howard:

λ-calculus + `call/cc`

Other examples:

- quote instruction $\sim$ dependent choice
- monotonic memory $\sim$ Cohen's forcing
- ...

The motto

With side-effects come new reasoning principles.

Extending Curry-Howard

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

1990: Griffin discovered that `call/cc` can be typed by Peirce's law
(well-known fact: Peirce's law $\Rightarrow A \vee \neg A$)

Classical Curry-Howard:

λ-calculus + `call/cc`

Other examples:

- quote instruction $\sim$ dependent choice
- monotonic memory $\sim$ Cohen's forcing
- ...

The motto

With side-effects come new reasoning principles.

Extending Curry-Howard

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

1990: Griffin discovered that `call/cc` can be typed by Peirce's law
(well-known fact: Peirce's law $\Rightarrow A \vee \neg A$)

Classical Curry-Howard:

λ-calculus + `call/cc`

Other examples:

- quote instruction $\sim$ dependent choice
- monotonic memory $\sim$ Cohen's forcing
- ...

The motto

With side-effects come new reasoning principles.

Teaser #1

The motto

With side-effects come new reasoning principles.

In Part II, we will use several **computational features**:

- dependent types
- streams
- lazy evaluation
- shared memory

to get a **proof** for the axioms of **dependent and countable choice** that is compatible with **classical logic**.

Theory vs Model

What is the status of axioms (*e.g.* $A \vee \neg A$)?

$\looparrowright$ neither true nor false in the ambient theory

(here, *true* means *provable*)

There is another point of view:

- **Theory**: *provability* in an axiomatic representation (syntax)
- **Model**: *validity* in a particular structure (semantic)

Example:

$A \wedge B$		
$A \backslash B$	✓	✗
✓	✓	✗
✗	✗	✗

$A \vee B$		
$A \backslash B$	✓	✗
✓	✓	✓
✗	✓	✗

A	$\neg A$	$A \vee \neg A$
✓	✗	✓
✗	✓	✓

Theory vs Model

What is the status of axioms (*e.g.* $A \vee \neg A$)?

$\looparrowright$ neither true nor false in the ambient theory
(here, *true* means *provable*)

There is another point of view:

- **Theory**: *provability* in an axiomatic representation (syntax)
- **Model**: *validity* in a particular structure (semantic)

Example:

$A \wedge B$		
A \ B	✓	✗
✓	✓	✗
✗	✗	✗

$A \vee B$		
A \ B	✓	✗
✓	✓	✓
✗	✓	✗

A	$\neg A$	$A \vee \neg A$
✓	✗	✓
✗	✓	✓

Theory vs Model

What is the status of axioms (*e.g.* $A \vee \neg A$)?

$\looparrowright$ neither true nor false in the ambient theory

(here, *true* means *provable*)

There is another point of view:

- **Theory**: *provability* in an axiomatic representation (syntax)
- **Model**: *validity* in a particular structure (semantic)

Example:

$A \wedge B$		
$A \backslash B$	✓	✗
✓	✓	✗
✗	✗	✗

$A \vee B$		
$A \backslash B$	✓	✗
✓	✓	✓
✗	✓	✗

A	$\neg A$	$A \vee \neg A$
✓	✗	✓
✗	✓	✓

Theory vs Model

What is the status of axioms (*e.g.* $A \vee \neg A$)?

↬ neither true nor false in the ambient theory

(here, *true* means *provable*)

There is another point of view:

- **Theory**: *provability* in an axiomatic representation (syntax)
- **Model**: *validity* in a particular structure (semantic)

Example:

$A \wedge B$		
A \ B	✓	✗
✓	✓	✗
✗	✗	✗

$A \vee B$		
A \ B	✓	✗
✓	✓	✓
✗	✓	✗

A	$\neg A$	$A \vee \neg A$
✓	✗	✓
✗	✓	✓

Valid formula

Teaser #2

Classical realizability:

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

(intuition: programs that share a common computational behavior given by A)

Great news

Classical realizability semantics gives surprisingly new models!

In Part III, we will study the **algebraic structure** of these models.

Part II

-

A constructive proof of dependent choice compatible with classical logic

The axiom of choice

Chapter 5

Axiom of Choice:

$$AC \quad : \quad \forall x^A . \exists y^B . P(x,y) \to \exists f^{A \to B} . \forall x^A . P(x, f(x))$$

The axiom of choice

Chapter 5

Axiom of Choice:

$$\begin{array}{rcl} AC & : & \forall x^A.\exists y^B.P(x,y) \to \exists f^{A\to B}\forall x^A.P(x,f(x)) \\ & := & \lambda H.(\lambda x.\mathtt{wit}\ (H\,x), \lambda x.\mathtt{prf}\ (H\,x)) \end{array}$$

Computational content through **dependent types:**

$$\frac{\Gamma, x : T \vdash t : A}{\Gamma \vdash \lambda x.t : \forall x^T.A}\ (\forall_I) \qquad \frac{\Gamma \vdash p : A[t/x] \quad \Gamma \vdash t : T}{\Gamma \vdash (t,p) : \exists x^T.A}\ (\exists_I)$$

$$\frac{\Gamma \vdash p : \exists x^T.A(x)}{\Gamma \vdash \mathtt{wit}\ p : T}\ (\mathtt{wit}) \qquad \frac{\Gamma \vdash p : \exists x^T.A(x)}{\Gamma \vdash \mathtt{prf}\ p : A(\mathtt{wit}\ p)}\ (\mathtt{prf})$$

Incompatibility with classical logic

Bad news

dependent sum + classical logic = ☠

One can define:

$$H_0 := \texttt{call/cc}_\alpha(1, \texttt{throw}_\alpha(0,p)) : \exists x.P(x)$$

and reach a contradiction:

$$(\texttt{wit}\, H_0, \texttt{prf}\, H_0) \rightarrow \underbrace{(1, \overbrace{p}^{P(0)})}_{\cancel{\exists x.P(x)}}$$

We need to:

↬ **share** ↬ **restrict** dependent types

Incompatibility with classical logic

Bad news

dependent sum + classical logic = ☠

One can define:

$$H_0 := \mathtt{call/cc}_\alpha(1, \mathtt{throw}_\alpha(0,p)) : \exists x.P(x)$$

and reach a contradiction:

$$(\mathtt{wit}\, H_0, \mathtt{prf}\, H_0) \rightarrow \underbrace{(1, \overbrace{p}^{P(0)})}_{\cancel{\exists x.P(x)}}$$

We need to:

↬ **share**

↬ **restrict** dependent types

Toward a solution ?

- Restriction to countable choice:

$$AC_{\mathbb{N}} : \forall x^{\mathbb{N}}.\exists y^{B}.P(x,y) \rightarrow \exists f^{\mathbb{N}\rightarrow B}.\forall x^{\mathbb{N}}.P(x,f(x))$$

- Proof:

$$\begin{aligned} AC := \lambda H.(\lambda n.&\texttt{if } n = 0 \texttt{ then } \texttt{wit}(H\ 0) \texttt{ else} \\ &\texttt{if } n = 1 \texttt{ then } \texttt{wit}(H\ 1) \texttt{ else} \ldots, \\ \lambda n.&\texttt{if } n = 0 \texttt{ then } \texttt{prf}(H\ 0) \texttt{ else} \\ &\texttt{if } n = 1 \texttt{ then } \texttt{prf}(H\ 1) \texttt{ else} \ldots) \end{aligned}$$

Toward a solution ?

- Restriction to countable choice:

$$AC_{\mathbb{N}} : \forall x^{\mathbb{N}}.\exists y^{B}.P(x,y) \to \exists f^{\mathbb{N}\to B}.\forall x^{\mathbb{N}}.P(x,f(x))$$

- Proof:

$$\begin{aligned} AC_{\mathbb{N}} := \lambda H.&\mathtt{let}\ H_0 = H\ 0\ \mathtt{in} \\ &\mathtt{let}\ H_1 = H\ 1\ \mathtt{in} \\ &\ldots \\ &(\lambda n.\mathtt{if}\ n = 0\ \mathtt{then}\ \mathtt{wit}\ H_0\ \mathtt{else} \\ &\quad\ \mathtt{if}\ n = 1\ \mathtt{then}\ \mathtt{wit}\ H_1\ \mathtt{else}\ \ldots, \\ &\ \lambda n.\mathtt{if}\ n = 0\ \mathtt{then}\ \mathtt{prf}\ H_0\ \mathtt{else} \\ &\quad\ \mathtt{if}\ n = 1\ \mathtt{then}\ \mathtt{prf}\ H_1\ \mathtt{else}\ \ldots) \end{aligned}$$

Toward a solution ?

- Restriction to countable choice:

$$AC_{\mathbb{N}} : \forall x^{\mathbb{N}}.\exists y^{B}.P(x,y) \to \exists f^{\mathbb{N}\to B}.\forall x^{\mathbb{N}}.P(x,f(x))$$

- Proof:

$$\begin{aligned} AC_{\mathbb{N}} := \lambda H.\mathtt{let}\ H_\infty = (H\ 0, H\ 1, \ldots, H\ n, \ldots)\ \mathtt{in} \\ (\lambda n.\,\mathtt{wit}\,(\mathtt{nth}\ n\ H_\infty), \lambda n.\,\mathtt{prf}\,(\mathtt{nth}\ n\ H_\infty)) \end{aligned}$$

Toward a solution ?

- Restriction to countable choice:

$$AC_{\mathbb{N}} : \forall x^{\mathbb{N}}.\exists y^{B}.P(x,y) \rightarrow \exists f^{\mathbb{N}\rightarrow B}.\forall x^{\mathbb{N}}.P(x,f(x))$$

- Proof:

$$\begin{aligned} AC_{\mathbb{N}} := \lambda H.\,\texttt{let}\ H_\infty = \texttt{cofix}^{0}_{bn}(H\ n, b(S(n)))\ \texttt{in} \\ (\lambda n.\,\texttt{wit}\ (\texttt{nth}\ n\ H_\infty), \lambda n.\,\texttt{prf}\ (\texttt{nth}\ n\ H_\infty)) \end{aligned}$$

dPA^ω (Herbelin's recipe)

A proof system:

- **classical**:

$$p, q ::= \dots \mid \texttt{catch}_\alpha\ p \mid \texttt{throw}_\alpha\ p$$

- with stratified **dependent types** :
  - terms: $t, u ::= \dots \mid \texttt{wit}\ p$
  - formulas: $A, B ::= \dots \mid \forall x^T.A \mid \exists x^T.A \mid \Pi a : A.B \mid t = u$
  - proofs: $p, q ::= \dots \mid \lambda x.p \mid (t, p) \mid \lambda a.p$

- a **syntactical restriction** of dependencies to NEF proofs
- call-by-value and **sharing**:

$$p, q ::= \dots \mid \texttt{let}\ a = q\ \texttt{in}\ p$$

- with inductive and **coinductive** constructions:

$$p, q ::= \dots \mid \texttt{ind}_{bn}^t[p \mid p] \mid \texttt{cofix}_{bn}^t\ p$$

- **lazy evaluation** for the `cofix`

dPA^ω (Herbelin's recipe)

A proof system:

- **classical**:

$$p, q ::= \ldots \mid \texttt{catch}_\alpha\ p \mid \texttt{throw}_\alpha\ p$$

- with stratified **dependent types** :
  - terms: $t, u ::= \ldots \mid \texttt{wit}\ p$
  - formulas: $A, B ::= \ldots \mid \forall x^T.A \mid \exists x^T.A \mid \Pi a : A.B \mid t = u$
  - proofs: $p, q ::= \ldots \mid \lambda x.p \mid (t, p) \mid \lambda a.p$
- a **syntactical restriction** of dependencies to NEF proofs
- call-by-value and **sharing**:

$$p, q ::= \ldots \mid \texttt{let}\ a = q\ \texttt{in}\ p$$

- with inductive and **coinductive** constructions:

$$p, q ::= \ldots \mid \texttt{ind}_{bn}^t [p \mid p] \mid \texttt{cofix}_{bn}^t\ p$$

- **lazy evaluation** for the cofix

State of the art

Subject reduction

If $\Gamma \vdash p : A$ and $p \to q$, then $\Gamma \vdash q : A$.

Normalization

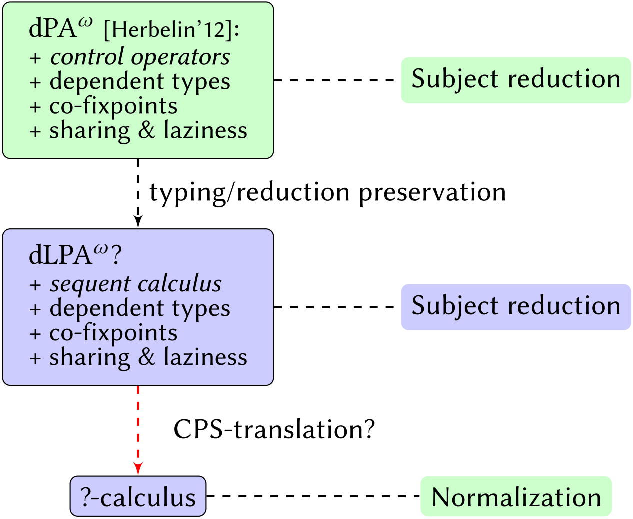
If $\Gamma \vdash p : A$ then p is normalizable.

requires

Consistency

$$\nvdash_{dPA^\omega} \bot$$

Roadmap


dPA$^\omega$ [Herbelin'12]:
+ *control operators*
+ dependent types
+ co-fixpoints
+ sharing & laziness
Subject reduction
typing/reduction preservation
dLPA$^\omega$?
+ *sequent calculus*
+ dependent types
+ co-fixpoints
+ sharing & laziness
Subject reduction
CPS-translation?
?-calculus
Normalization

Roadmap

dPA$^\omega$ [Herbelin'12]:
+ *control operators*
+ dependent types
+ co-fixpoints
+ sharing & laziness

Subject reduction

typing/reduction preservation

dLPA$^\omega$?
+ *sequent calculus*
+ dependent types
+ co-fixpoints
+ sharing & laziness

$\overline{\lambda}_{[lv\tau\star]}$

Subject reduction

CPS-translation?

?-calculus

Normalization

Roadmap

dPA$^\omega$ [Herbelin'12]:
+ *control operators*
+ dependent types
+ co-fixpoints
+ sharing & laziness

Subject reduction

typing/reduction preservation

dLPA$^\omega$?
+ *sequent calculus*
+ dependent types
+ co-fixpoints
+ sharing & laziness

dL

Subject reduction

CPS-translation?

?-calculus

Normalization

Classical call-by-need

Chapter 6

The $\overline{\lambda}_{[lv\tau\star]}$-calculus (Ariola *et al.*):

- a sequent calculus with explicit stores
- Danvy's method of semantics artifact:
  1. derive a small-step reduction system
  2. derive context-free small-step reduction rules
  3. derive an (untyped) CPS

Questions:

↬ Does it normalize?

↬ Can the CPS be typed?

↬ Can we define a realizability interpretation?

Danvy's semantics artifacts

Syntax:

(Proofs)	p	$::=$	$V \mid \mu\alpha.c$
(Weak values)	V	$::=$	$v \mid a$
(Strong values)	v	$::=$	$\lambda a.p$
(Contexts)	e	$::=$	$E \mid \tilde{\mu}a.c$
(Catchable contexts)	E	$::=$	$\alpha \mid F \mid \tilde{\mu}[a].\langle a \parallel F\rangle\tau$
(Forcing contexts)	F	$::=$	$p \cdot E$
(Commands)	c	$::=$	$\langle p \parallel e\rangle$
(Closures)	l	$::=$	$c\tau$
(Store)	τ	$::=$	$\epsilon \mid \tau[a := p]$

Reduction rules:

(Lazy storage)	$\langle p \parallel \tilde{\mu}a.c\rangle\tau$	$\rightarrow$	$c\tau[a := p]$
	$\langle \mu\alpha.c \parallel E\rangle\tau$	$\rightarrow$	$(c[E/\alpha])\tau$
(Lookup)	$\langle a \parallel F\rangle\tau[a := \boldsymbol{p}]\tau'$	$\rightarrow$	$\langle \boldsymbol{p} \parallel \tilde{\mu}[a].\langle a \parallel F\rangle\tau'\rangle\tau$
(Forced eval.)	$\langle \boldsymbol{V} \parallel \tilde{\mu}[a].\langle a \parallel F\rangle\tau'\rangle\tau$	$\rightarrow$	$\langle \boldsymbol{V} \parallel F\rangle\tau[a := \boldsymbol{V}]\tau'$
	$\langle \lambda a.p \parallel q \cdot E\rangle\tau$	$\rightarrow$	$\langle q \parallel \tilde{\mu}a.\langle p \parallel E\rangle\rangle\tau$

Danvy's semantics artifacts

Small steps:

$$
\begin{array}{llcl}
e & \langle p \,\|\, \tilde\mu a.c \rangle_e \tau & \to & c_e \tau[a := p] \\
 & \langle p \,\|\, E \rangle_e \tau & \to & \langle p \,\|\, E \rangle_p \tau \\
p & \langle \mu\alpha.c \,\|\, E \rangle_p \tau & \to & (c[E/\alpha])\tau \\
 & \langle V \,\|\, E \rangle_p \tau & \to & \langle V \,\|\, E \rangle_E \tau \\
E & \langle V \,\|\, \tilde\mu[a].\langle a \,\|\, F \rangle \tau' \rangle_E \tau & \to & \langle V \,\|\, F \rangle_V \tau[a := V]\tau' \\
 & \langle V \,\|\, F \rangle_E \tau & \to & \langle V \,\|\, F \rangle_V \tau \\
V & \langle a \,\|\, F \rangle_V \tau[a := p]\tau' & \to & \langle p \,\|\, \tilde\mu[a].\langle a \,\|\, F \rangle \tau' \rangle_p \tau \\
 & \langle \lambda a.p \,\|\, F \rangle_V \tau & \to & \langle \lambda a.p \,\|\, F \rangle_F \tau \\
F & \langle \lambda a.p \,\|\, q \cdot E \rangle_F \tau & \to & \langle q \,\|\, \tilde\mu a.\langle p \,\|\, E \rangle \rangle_e \tau
\end{array}
$$

Danvy's semantics artifacts

CPS :

$$
\begin{array}{lrcl}
 & [\![\langle p \parallel e\rangle\tau]\!] & := & [\![e]\!]_e\, [\![\tau]\!]_\tau\, [\![p]\!]_p \\
e & [\![\tilde\mu a.c]\!]_e & := & \lambda\tau p.[\![c]\!]\ \tau[a := p] \\
 & [\![E]\!]_e & := & \lambda\tau p.p\,\tau\,[\![E]\!]_E \\
p & [\![\mu\alpha.c]\!]_p & := & \lambda\tau E.([\![c]\!]_c\,\tau)[E/\alpha] \\
 & [\![V]\!]_p & := & \lambda\tau E.E\,\tau\,[\![V]\!]_V \\
E & [\![\tilde\mu[a].\langle a \parallel F\rangle\tau']\!]_E & := & \lambda\tau V.V\,\tau[a := V]\tau'\,[\![F]\!]_F \\
 & [\![F]\!]_E & := & \lambda\tau V.V\,\tau\,[\![F]\!]_F \\
V & [\![a]\!]_V & := & \lambda\tau F.\tau(a)\,\tau\,(\lambda\tau V.V\,\tau[a := V]\tau'\,[\![F]\!]_F) \\
 & [\![\lambda a.p]\!]_V & := & \lambda\tau F.F\,\tau\,(\lambda q\tau E.[\![p]\!]_p\,\tau[a := q]\ E) \\
F & [\![q\cdot E]\!]_F & := & \lambda\tau v.v\,[\![q]\!]_p\,\tau\,[\![E]\!]_E
\end{array}
$$

Realizability interpretation

- **Store extension:** $\tau \lhd \tau'$
- **Term-in-store** $(t|\tau)$: closed store τ s.t. $FV(t) \subseteq \mathrm{dom}(\tau)$.
  ($\looparrowright$ generalizes closed terms)
- **Pole :** set of closures $\perp\!\!\!\perp$ which is:
  - *saturated*:

$$c'\tau' \in \perp\!\!\!\perp \quad \text{and} \quad c\tau \to c'\tau' \quad \text{implies} \quad c\tau \in \perp\!\!\!\perp$$

  - *closed by store extension*:

$$c\tau \in \perp\!\!\!\perp \quad \text{and} \quad \tau \lhd \tau' \quad \text{implies} \quad c\tau' \in \perp\!\!\!\perp$$

- **Compatible stores:** $\tau \diamond \tau'$
- **Orthogonality** $(t|\tau)\perp\!\!\!\perp(e|\tau')$: $\tau \diamond \tau'$ and $\langle t \| e\rangle\overline{\tau\tau'} \in \perp\!\!\!\perp$.
- **Realizers**: definitions derived from the small-step rules!

Realizability interpretation

- **Store extension:** $\tau \lhd \tau'$
- **Term-in-store** $(t|\tau)$: closed store τ s.t. $FV(t) \subseteq \mathrm{dom}(\tau)$.
  (↬ generalizes closed terms)
- **Pole :** set of closures $⫫$ which is:
  - *saturated*:

$$c'\tau' \in ⫫ \quad \text{and} \quad c\tau \to c'\tau' \quad \text{implies} \quad c\tau \in ⫫$$

  - *closed by store extension*:

$$c\tau \in ⫫ \quad \text{and} \quad \tau \lhd \tau' \quad \text{implies} \quad c\tau' \in ⫫$$

- **Compatible stores:** $\tau \diamond \tau'$
- **Orthogonality** $(t|\tau)⫫(e|\tau')$: $\tau \diamond \tau'$ and $\langle t \| e\rangle\overline{\tau\tau'} \in ⫫$.
- **Realizers**: definitions derived from the small-step rules!

Realizability interpretation

Adequacy

For all $\perp\!\!\!\perp$, if $\tau \Vdash \Gamma$ and $\Gamma \vdash_c c$, then $c\tau \in \perp\!\!\!\perp$.

Normalization

If $\vdash_l c\tau$ then $c\tau$ normalizes.

Proof: The set $\perp\!\!\!\perp_{\Downarrow} = \{c\tau \in C_0 : \ c\tau \text{ normalizes}\}$ *is a pole.*

(+ Bonus: typed CPS translation unveiling Kripke's forcing.)

A classical sequent calculus with dependent types

Chapter 7

Can this work?

$$\dfrac{\dfrac{\begin{array}{c}\Pi_p\\ \vdots\\ \Gamma, a : A \vdash p : B[a] \mid \Delta\end{array}}{\Gamma \vdash \lambda a.p : \Pi a : A.B \mid \Delta}\,(\to_r) \qquad \dfrac{\begin{array}{c}\Pi_q\\ \vdots\\ \Gamma \vdash q : A \mid \Delta\end{array} \quad \begin{array}{c}\Pi_e\\ \vdots\\ \Gamma \mid e : B[q] \vdash \Delta\end{array} \quad q \in V}{\Gamma \mid q \cdot e : \Pi a : A.B \vdash \Delta}\,(\to_l)}{\langle \lambda a.p \,\|\, q \cdot e \rangle : (\Gamma \vdash \Delta)}\,(\textsc{Cut})$$

$$\longrightarrow$$

A classical sequent calculus with dependent types

Chapter 7

Can this work?

$$\dfrac{\dfrac{\begin{array}{c}\Pi_p\\ \vdots\\ \Gamma, a : A \vdash p : B[a] \mid \Delta\end{array}}{\Gamma \vdash \lambda a.p : \Pi a : A.B \mid \Delta}(\to_r) \qquad \dfrac{\begin{array}{c}\Pi_q\\ \vdots\\ \Gamma \vdash q : A \mid \Delta\end{array} \quad \begin{array}{c}\Pi_e\\ \vdots\\ \Gamma \mid e : B[q] \vdash \Delta\end{array} \quad q \in V}{\Gamma \mid q \cdot e : \Pi a : A.B \vdash \Delta}(\to_l)}{\langle \lambda a.p \,\|\, q \cdot e \rangle : (\Gamma \vdash \Delta)}(\textsc{Cut})$$

$$\longrightarrow$$

$$\dfrac{\begin{array}{c}\Pi_q\\ \vdots\\ \Gamma \vdash q : A \mid \Delta\end{array} \qquad \dfrac{\dfrac{\Gamma, a : A \vdash p : \cancel{B[a]} \mid \Delta \quad \Gamma, a : A \mid e : \cancel{B[q]} \vdash \Delta}{\langle p \,\|\, e \rangle : (\Gamma, a : A \vdash \Delta)}\ \textit{Mismatch}}{\Gamma \mid \tilde{\mu} a.\langle p \,\|\, e \rangle : A \vdash \Delta}(\tilde{\mu})}{\langle q \,\|\, \tilde{\mu} a.\langle p \,\|\, e \rangle \rangle : (\Gamma \vdash \Delta)}(\textsc{Cut})$$

A classical sequent calculus with dependent types

Chapter 7

Can this work? $\checkmark$

$$\dfrac{\dfrac{\begin{array}{c}\Pi_p\\ \vdots\\ \Gamma, a : A \vdash p : B[a] \mid \Delta\end{array}}{\Gamma \vdash \lambda a.p : \Pi a : A.B \mid \Delta}\,(\to_r) \qquad \dfrac{\begin{array}{c}\Pi_q\\ \vdots\\ \Gamma \vdash q : A \mid \Delta\end{array} \quad \begin{array}{c}\Pi_e\\ \vdots\\ \Gamma \mid e : B[q] \vdash \Delta\end{array} \quad q \in V}{\Gamma \mid q \cdot e : \Pi a : A.B \vdash \Delta}\,(\to_l)}{\langle \lambda a.p \,\|\, q \cdot e\rangle : (\Gamma \vdash \Delta)}\,(\textsc{Cut})$$

$$\longrightarrow$$

$$\dfrac{\begin{array}{c}\Pi_q\\ \vdots\\ \Gamma \vdash q : A \mid \Delta\end{array} \qquad \dfrac{\dfrac{\Gamma, a : A \vdash p : B[a] \mid \Delta \quad \Gamma, a : A \mid e : B[q] \vdash \Delta; \{\cdot|p\}\{a|q\}}{\langle p \,\|\, e\rangle : \Gamma, a : A \vdash \Delta; \{a|q\}}\,(\textsc{Cut})}{\Gamma \mid \tilde{\mu}a.\langle p \,\|\, e\rangle : A \vdash \Delta; \{.|q\}}\,(\tilde{\mu})}{\langle q \,\|\, \tilde{\mu}a.\langle p \,\|\, e\rangle\rangle : (\Gamma \vdash \Delta); \{\cdot|\cdot\}}\,(\textsc{Cut})$$

dL

A type system with:

- a **list of dependencies**:

$$\frac{\Gamma \vdash p : A \mid \Delta; \sigma \qquad \Gamma \mid e : A' \vdash \Delta; \sigma\{\cdot|p\} \qquad A' \in A_\sigma}{\langle p \| e \rangle : (\Gamma \vdash \Delta; \sigma)}\ (\textsc{Cut})$$

- a **value restriction**

Is it enough?

- subject reduction
- normalization
- consistency as a logic
- suitable for CPS translation

dL

A type system with:

- a **list of dependencies**:

$$\frac{\Gamma \vdash p : A \mid \Delta; \sigma \qquad \Gamma \mid e : A' \vdash \Delta; \sigma\{\cdot|p\} \qquad A' \in A_\sigma}{\langle p \,\|\, e \rangle : (\Gamma \vdash \Delta; \sigma)} \ (\textsc{Cut})$$

- a **value restriction**

Is it enough?

- subject reduction ✓
- normalization ✓
- consistency as a logic ✓
- suitable for CPS translation ✗

dL

A type system with:

- a **list of dependencies**:

$$\frac{\Gamma \vdash p : A \mid \Delta; \sigma \qquad \Gamma \mid e : A' \vdash \Delta; \sigma\{\cdot|p\} \qquad A' \in A_\sigma}{\langle p \| e \rangle : (\Gamma \vdash \Delta; \sigma)} \text{(Cut)}$$

- a **value restriction**

Is it enough?

- subject reduction ✓
- normalization ✓
- consistency as a logic ✓
- suitable for CPS translation ✗

$$[\![q]\!]\,[\![\tilde{\mu}a.\langle p \| e \rangle]\!] = \underbrace{[\![q]\!]}_{\neg\neg A}\,(\lambda a.\,\underbrace{[\![p]\!]}_{\neg\neg B(a)}\,\underbrace{[\![e]\!]}_{\neg B(q)})$$

Toward a CPS translation (1/2)

This is quite normal:

- we observed a desynchronization
- we compensated only within the type system

↪ *we need to do this already in the calculus!*

Who's guilty ?

$$[\![\langle q \| \tilde\mu a.\langle p \| e\rangle\rangle]\!] = [\![q]\!]\,(\lambda a.[\![p]\!][\![e]\!])$$

Motto: $[\![p]\!]$ *shouldn't be applied to* $[\![e]\!]$ *before* $[\![q]\!]$ *has reduced*

$$([\![q]\!]\,(\lambda a.[\![p]\!]))[\![e]\!]$$

So, we're looking for:

Toward a CPS translation (1/2)

This is quite normal:

- we observed a desynchronization
- we compensated only within the type system

↪ *we need to do this already in the calculus!*

Who's guilty ?

$$[\![\langle q \,\|\, \tilde{\mu}a.\langle p \,\|\, e\rangle\rangle]\!] = [\![q]\!]\,(\lambda a.[\![p]\!][\![e]\!])$$

Motto: $[\![p]\!]$ *shouldn't be applied to* $[\![e]\!]$ *before* $[\![q]\!]$ *has reduced*

$$([\![q]\!]\,(\lambda a.[\![p]\!]))[\![e]\!]$$

So, we're looking for:

Toward a CPS translation (1/2)

This is quite normal:

- we observed a desynchronization
- we compensated only within the type system

↬ *we need to do this already in the calculus!*

Who's guilty ?

$$[\![\langle q \,\|\, \tilde{\mu}a.\langle p \,\|\, e\rangle\rangle]\!] \;=\; [\![q]\!]\;(\lambda a.[\![p]\!][\![e]\!])$$

Motto: $[\![p]\!]$ *shouldn't be applied to* $[\![e]\!]$ *before* $[\![q]\!]$ *has reduced*

$$([\![q]\!]\;(\lambda a.[\![p]\!]))[\![e]\!]$$

So, we're looking for:

Toward a CPS translation (1/2)

This is quite normal:

- we observed a desynchronization
- we compensated only within the type system

↪ *we need to do this already in the calculus!*

Who's guilty ?

$$[\![\langle q \parallel \tilde{\mu}a.\langle p \parallel e\rangle\rangle]\!] = [\![q]\!]\,(\lambda a.[\![p]\!]\,[\![e]\!])$$

Motto: $[\![p]\!]$ *shouldn't be applied to* $[\![e]\!]$ *before* $[\![q]\!]$ *has reduced*

$$([\![q]\!]\,(\lambda a.[\![p]\!]))\,[\![e]\!]$$

So, we're looking for:

$$\langle \lambda a.p \parallel q \cdot e\rangle \to \langle \mu\hat{\mathrm{tp}}.\langle q \parallel \tilde{\mu}a.\langle p \parallel \hat{\mathrm{tp}}\rangle\rangle \parallel e\rangle$$

Toward a CPS translation (2/2)

$$[\![\langle \lambda a.p \,\|\, q \cdot e \rangle]\!] \xrightarrow{?} ([\![q]\!]\,(\lambda a.[\![p]\!]))[\![e]\!]$$

Questions:

1. Is any q compatible with such a reduction ?
2. Is this typable ?

Toward a CPS translation (2/2)

$$[\![\langle \lambda a.p \,\|\, q \cdot e\rangle]\!] \xrightarrow{?} ([\![q]\!]\,(\lambda a.[\![p]\!]))[\![e]\!]$$

Questions:

1. Is any q compatible with such a reduction ?

- If q eventually gives a value V: $\rightsquigarrow [\![p[V/a]]\!][\![e]\!]$ ✓
- If $[\![q]\!] \to \lambda\_.t$ and drops its continuation: $\rightsquigarrow t[\![e]\!]$ ✗

Toward a CPS translation (2/2)

$$[\![\langle \lambda a.p \,\|\, q \cdot e\rangle]\!] \xrightarrow{?} ([\![q]\!]\ (\lambda a.[\![p]\!]))[\![e]\!]$$

Questions:

1. Is any q compatible with such a reduction ? $\rightsquigarrow q \in \mathrm{NEF}$

- If q eventually gives a value V: $\rightsquigarrow [\![p[V/a]]\!][\![e]\!]$ ✓
- If $[\![q]\!] \to \lambda\_.t$ and drops its continuation: $\rightsquigarrow t[\![e]\!]$ ✗

Negative-elimination free (Herbelin'12)

Values + one continuation variable + no application

Toward a CPS translation (2/2)

$$[\![\langle \lambda a.p \,\|\, q \cdot e\rangle]\!] \stackrel{?}{\longrightarrow} ([\![q]\!]\ (\lambda a.[\![p]\!]))[\![e]\!]$$

Questions:

1. Is any q compatible with such a reduction ? $\rightsquigarrow q \in \mathrm{NEF}$
2. Is this typable ?

Naive attempt:

$$(\underbrace{[\![q]\!]}_{(A\to\bot)\to\bot}\quad (\underbrace{\lambda a.[\![p]\!]}_{\Pi_{(a:A)}\neg\neg B(a)}))\quad \underbrace{[\![e]\!]}_{\neg B[q]}$$

Toward a CPS translation (2/2)

$$[\![\langle \lambda a.p \,\|\, q \cdot e\rangle]\!] \xrightarrow{?} ([\![q]\!]\ (\lambda a.[\![p]\!]))[\![e]\!]$$

Questions:

1. Is any q compatible with such a reduction ? $\rightsquigarrow q \in \mathrm{NEF}$
2. Is this typable ?

Better:

$$\underbrace{(\underbrace{[\![q]\!]}_{\forall R.(\Pi_{(a:A)}R(a))\to R(q)}\quad(\underbrace{\lambda a.[\![p]\!]}_{\Pi_{(a:A)}\neg\neg B(a)}))}_{\neg\neg B(q)}\quad\underbrace{[\![e]\!]}_{\neg B[q]}$$

(*Remark: not possible without* $q \in$ *NEF*)

$\mathrm{dL}_{\hat{\mathrm{tp}}}$

An extension of dL with:

- **delimited continuations**
- dependent types restricted to the **NEF fragment**

Regular mode	Dependent mode
$\dfrac{\Gamma \vdash p : A \mid \Delta \quad \Gamma \mid e : A \vdash \Delta}{\langle p \| e \rangle : \Gamma \vdash \Delta}$	$\dfrac{\Gamma \vdash p : A \mid \Delta \quad \Gamma \mid e : A \vdash_d \Delta, \hat{\mathrm{tp}} : B; \sigma\{\cdot \mid p\}}{\langle p \| e \rangle : \Gamma \vdash_d \Delta, \hat{\mathrm{tp}} : B; \sigma}$

- delimited scope of dependencies:

$$\frac{c : (\Gamma \vdash_d \Delta, \hat{\mathrm{tp}} : A; \{\cdot \mid \cdot\})}{\Gamma \vdash \mu \hat{\mathrm{tp}}.c : A \mid \Delta}\ \hat{\mathrm{tp}}_I \qquad \frac{B \in A_\sigma}{\Gamma \mid \hat{\mathrm{tp}} : A \vdash_d \Delta, \hat{\mathrm{tp}} : B; \sigma\{\cdot \mid p\}}\ \hat{\mathrm{tp}}_E$$

- Mission accomplished?
  - subject reduction
  - normalization
  - consistency as a logic
  - CPS translation

- *(Bonus)* embedding into Lepigre's calculus ✓
  - ↪ realizability interpretation

$\mathrm{dL}_{\hat{\mathrm{tp}}}$

An extension of dL with:

- **delimited continuations**
- dependent types restricted to the **NEF fragment**

Regular mode	*Dependent mode*
$\dfrac{\Gamma \vdash p : A \mid \Delta \quad \Gamma \mid e : A \vdash \Delta}{\langle p \,\|\, e\rangle : \Gamma \vdash \Delta}$	$\dfrac{\Gamma \vdash p : A \mid \Delta \quad \Gamma \mid e : A \vdash_d \Delta, \hat{\mathrm{tp}} : B; \sigma\{\cdot \mid p\}}{\langle p \,\|\, e\rangle : \Gamma \vdash_d \Delta, \hat{\mathrm{tp}} : B; \sigma}$

- delimited scope of dependencies:

$$\frac{c : (\Gamma \vdash_d \Delta, \hat{\mathrm{tp}} : A; \{\cdot \mid \cdot\})}{\Gamma \vdash \mu\hat{\mathrm{tp}}.c : A \mid \Delta}\ \hat{\mathrm{tp}}_I \qquad \frac{B \in A_\sigma}{\Gamma \mid \hat{\mathrm{tp}} : A \vdash_d \Delta, \hat{\mathrm{tp}} : B; \sigma\{\cdot \mid p\}}\ \hat{\mathrm{tp}}_E$$

- Mission accomplished ✓
  - subject reduction ✓
  - normalization ✓
  - consistency as a logic ✓
  - CPS translation ✓
  - *(Bonus)* embedding into Lepigre's calculus ✓
    - ↬ realizability interpretation

dLPA$^\omega$

Chapter 8

A **classical sequent calculus** with:

- stratified **dependent types** :
  - terms: $t, u ::= \dots \mid \mathtt{wit}\, p$
  - formulas: $A, B ::= \dots \mid \forall x^T.A \mid \exists x^T.A \mid \Pi a : A.B \mid t = u$
  - proofs: $p, q ::= \dots \mid \lambda x.p \mid (t, p) \mid \lambda a.p$
- a restriction to the NEF **fragment**
- **arithmetical** terms:

$$t, u ::= \dots \mid 0 \mid S(t) \mid \mathtt{rec}^t_{xy}[t_0 \mid t_S] \mid \lambda x.t \mid t\ u$$

- **stores**:

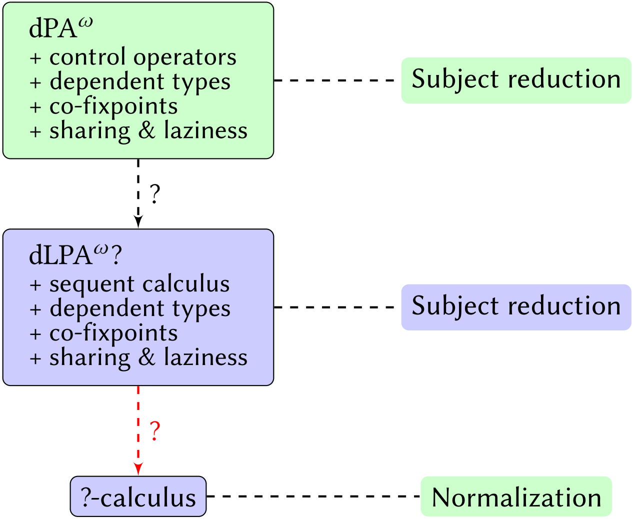
$$\tau ::= \varepsilon \mid \tau[a := p_\tau] \mid \tau[\alpha := e]$$

- inductive and **coinductive** constructions:

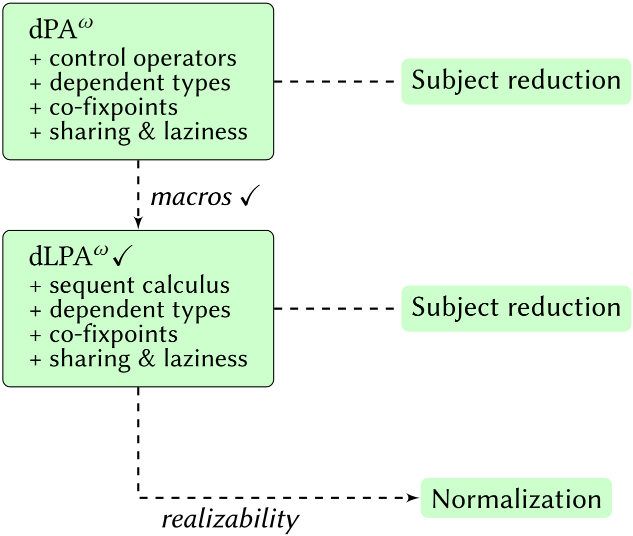
$$p, q ::= \dots \mid \mathtt{ind}^t_{bn}[p \mid p] \mid \mathtt{cofix}^t_{bn}\ p$$

- a **call-by-value** reduction and **lazy evaluation** of $\mathtt{cofix}$

End of the road



End of the road



Realizability interpretation

Same methodology:

1. small-step reductions
2. derive the realizability interpretation

Resembles $\overline{\lambda}_{[lv\tau\star]}$-interpretation, plus:

- dependent types from Lepigre's calculus
- co-inductive formulas

Realizability interpretation

Same methodology:

1. small-step reductions
2. derive the realizability interpretation

Resembles $\overline{\lambda}_{[lv\tau\star]}$-interpretation, plus:

- dependent types from Lepigre's calculus:

$$\Pi a : A.B \quad \triangleq \quad \forall a.(a \in A \to B)$$

- co-inductive formulas

Realizability interpretation

Same methodology:

1. small-step reductions
2. derive the realizability interpretation

Resembles $\overline{\lambda}_{[lv\tau\star]}$-interpretation, plus:

- dependent types from Lepigre's calculus
- co-inductive formulas: *by finite approximations*

$$\|\nu^t_{Xx} A\|_f \triangleq \bigcup_{n\in\mathbb{N}} \|F^n_{A,t}\|_f$$

Realizability interpretation

Same methodology:

1. small-step reductions
2. derive the realizability interpretation

Resembles $\overline{\lambda}_{[lv\tau\star]}$-interpretation, plus:

- dependent types from Lepigre's calculus
- co-inductive formulas: *by finite approximations*

Consequences of adequacy:

Normalization

If $\Gamma \vdash_\sigma c$, then c is normalizable.

Consistency

$$\nvdash_{\mathrm{dLPA}^\omega} p : \bot$$

Conclusion and further work

Contributions of this part:

- classical call-by-need:
  - realizability interpretation
  - typed continuation-and-store-passing style translation
- dependent classical sequent calculus:
  - list of dependencies
  - use of delimited continuations for soundness
  - dependently-typed continuation-passing style translation
- dLPA^ω:
  - soundness and normalization,
  - realizability interpretation of co-fixpoints

Conclusion and further work

Contributions of this part:

- classical call-by-need
- dependent classical sequent calculus
- dLPA^{ω}

Further work:

1. Can $\mathrm{dL}_{\mathrm{tp}}$ be related to:
   - Pédrot-Tabareau's Baclofen Type Theory ?
   - Vákár's categorical presentation ?
   - Bowman *et. al.* CPS for *CC* ?
2. Relation to Krivine's realizability semantics of DC:
   - Compatible with quote?
   - Approximation of the limits required for bar recursion?
3. Algebraic counterpart of side-effects in realizability structures?

Conclusion and further work

Contributions of this part:

- classical call-by-need
- dependent classical sequent calculus
- dLPA^{ω}

Further work:

1. Can $\mathrm{dL}_{\mathrm{tp}}$ be related to:
   - Pédrot-Tabareau's Baclofen Type Theory ?
   - Vákár's categorical presentation ?
   - Bowman *et. al.* CPS for *CC* ?
2. Relation to Krivine's realizability semantics of DC:
   - Compatible with quote?
   - Approximation of the limits required for bar recursion?
3. Algebraic counterpart of side-effects in realizability structures?

Conclusion and further work

Contributions of this part:

- classical call-by-need
- dependent classical sequent calculus
- dLPA^ω

Further work:

1. Can dL_{tp} be related to:
   - Pédrot-Tabareau's Baclofen Type Theory ?
   - Vákár's categorical presentation ?
   - Bowman *et. al.* CPS for *CC* ?
2. Relation to Krivine's realizability semantics of DC:
   - Compatible with quote?
   - Approximation of the limits required for bar recursion?
3. Algebraic counterpart of side-effects in realizability structures?

Part III

-

Algebraic models of classical realizability

Algebraization of classical realizability

Chapter 9

Realizers:

$$t \Vdash A \quad \text{defined as} \quad t \in \|A\|^{\perp\!\!\!\perp}$$

Key elements:

- the pole $\perp\!\!\!\perp$
- the lattice $(\mathcal{P}(\Pi), \supseteq)$

Observations:

- this induces a semantic subtyping:

$$A \leq_{\perp\!\!\!\perp} B \triangleq \|B\| \subseteq \|A\|$$

- connectives/quantifiers:

$$\forall = \curlywedge \qquad \wedge = \times \qquad \textbf{Realizability}$$

This is to be compared with:

$$\forall = \curlywedge = \wedge \qquad \textbf{Forcing}$$

Algebraization of classical realizability

Chapter 9

Realizers:

$$t \Vdash A \quad \text{defined as} \quad t \in \|A\|^{\perp\!\!\!\perp}$$

Key elements:

- the pole $\perp\!\!\!\perp$
- the lattice $(\mathcal{P}(\Pi), \supseteq)$

Observations:

- this induces a semantic subtyping:

$$A \leq_{\perp\!\!\!\perp} B \triangleq \|B\| \subseteq \|A\|$$

- connectives/quantifiers:

$$\forall = \curlywedge \qquad \wedge = \times$$

Realizability

This is to be compared with:

$$\forall = \curlywedge = \wedge$$

Forcing

Algebraization of classical realizability

Chapter 9

Realizers:

$$t \Vdash A \quad \text{defined as} \quad t \in \|A\|^{\perp\!\!\!\perp}$$

Key elements:

- the pole $\perp\!\!\!\perp$
- the lattice $(\mathcal{P}(\Pi), \supseteq)$

Observations:

- this induces a semantic subtyping:

$$A \leq_{\perp\!\!\!\perp} B \triangleq \|B\| \subseteq \|A\|$$

- connectives/quantifiers:

$$\forall = \curlywedge \qquad \wedge = \times$$

Realizability

This is to be compared with:

$$\forall = \curlywedge = \wedge$$

Forcing

Implicative structures

Chapter 10

$$\begin{tikzcd} \text{Types} \arrow[r, equal] \arrow[d, dashed, no head] & \text{Formulas} \arrow[d, dashed, no head] \\ \lambda\text{-terms} \arrow[r, equal] & \text{Proofs} \end{tikzcd}$$

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

Chapter 10

$$\begin{array}{ccc} \text{Types} & = & \text{Formulas} \\ \cup\!\mid & & \vdots \\ \lambda\text{-terms} & = & \text{Proofs} \end{array}$$

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

Chapter 10

$$\begin{array}{ccc} \text{Types} & = & \text{Formulas} \\ \| & & \| \\ \lambda\text{-terms} & = & \text{Proofs} \end{array}$$

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

Chapter 10

Definition:

Complete meet-semilattice $(\mathcal{A}, \preccurlyeq, \rightarrow)$ s.t.:

- if $a_0 \preccurlyeq a$ and $b \preccurlyeq b_0$ then $(a \rightarrow b) \preccurlyeq (a_0 \rightarrow b_0)$
- $\bigcurlywedge_{b\in B}(a \rightarrow b) = a \rightarrow \bigcurlywedge_{b\in B} b$

Examples:

- complete Heyting/Boolean algebras
- classical realizability:
  - $\mathcal{A} \triangleq \mathcal{P}(\Pi)$;
  - $a \preccurlyeq b \triangleq a \supseteq b$
  - $a \rightarrow b \triangleq a^{\perp\!\!\!\perp} \cdot b = \{t \cdot \pi : t \in a^{\perp\!\!\!\perp}, \pi \in b\}$

Interpretating of λ-terms

- λ-terms:

$$a@b \triangleq \curlywedge\{c \in \mathcal{A} : a \preccurlyeq (b \to c)\} \qquad \lambda f \triangleq \curlywedge_{a \in \mathcal{A}}(a \to f(a))$$

- `call/cc`:

$$\mathbf{cc} \triangleq \curlywedge_{a,b \in \mathcal{A}}(((a \to b) \to a) \to a)$$

Adjunction: $a \preccurlyeq b \to c \quad \Leftrightarrow \quad a@b \preccurlyeq c$

Adequacy: If $\vdash t : A$ then $t^{\mathcal{A}} \preccurlyeq A^{\mathcal{A}}$

In particular:

$$\begin{aligned} \mathbf{K}^{\mathcal{A}} &= \curlywedge_{a,b \in \mathcal{A}}(a \to b \to a) \\ \mathbf{S}^{\mathcal{A}} &= \curlywedge_{a,b,c \in \mathcal{A}}((a \to b \to c) \to (a \to b) \to a \to c) \end{aligned}$$

Interpretating of λ-terms

- λ-terms:

$$a@b \triangleq \curlywedge\{c \in \mathcal{A} : a \preccurlyeq (b \to c)\} \qquad \lambda f \triangleq \curlywedge_{a\in\mathcal{A}}(a \to f(a))$$

- call/cc:

$$\boldsymbol{cc} \triangleq \curlywedge_{a,b\in\mathcal{A}}(((a \to b) \to a) \to a)$$

Adjunction: $a \preccurlyeq b \to c \quad \Leftrightarrow \quad a@b \preccurlyeq c$

Adequacy: If $\vdash t : A$ then $t^{\mathcal{A}} \preccurlyeq A^{\mathcal{A}}$

In particular:

$$\begin{aligned} \mathbf{K}^{\mathcal{A}} &= \curlywedge_{a,b\in\mathcal{A}}(a \to b \to a) \\ \mathbf{S}^{\mathcal{A}} &= \curlywedge_{a,b,c\in\mathcal{A}}((a \to b \to c) \to (a \to b) \to a \to c) \end{aligned}$$

Implicative algebras

Separator $\mathcal{S}$:

1. $\kappa^{\mathcal{A}} \in \mathcal{S}$, and $s^{\mathcal{A}} \in \mathcal{S}$ *(Combinators)*
2. If $a \in \mathcal{S}$ and $a \preccurlyeq b$, then $b \in \mathcal{S}$. *(Upwards closure)*
3. If $(a \to b) \in \mathcal{S}$ and $a \in \mathcal{S}$, then $b \in \mathcal{S}$. *(Modus ponens)*

Implicative algebras:

$$(\mathcal{A}, \preccurlyeq, \to) \quad + \quad \text{separator } \mathcal{S}$$

Entailment:

$$a \vdash_{\mathcal{S}} b \quad \triangleq \quad a \to b \in \mathcal{S}.$$

Adjunction

$$a \vdash_{\mathcal{S}} b \to c \qquad \text{if and only if} \qquad a \times b \vdash_{\mathcal{S}} c$$

Implicative algebras

Separator $\mathcal{S}$:

1. $\kappa^{\mathcal{A}} \in \mathcal{S}$, and $s^{\mathcal{A}} \in \mathcal{S}$ *(Combinators)*
2. If $a \in \mathcal{S}$ and $a \preccurlyeq b$, then $b \in \mathcal{S}$. *(Upwards closure)*
3. If $(a \to b) \in \mathcal{S}$ and $a \in \mathcal{S}$, then $b \in \mathcal{S}$. *(Modus ponens)*

Implicative algebras:

$$(\mathcal{A}, \preccurlyeq, \to) \quad + \quad \text{separator } \mathcal{S}$$

Entailment:

$$a \vdash_{\mathcal{S}} b \quad \triangleq \quad a \to b \in \mathcal{S}.$$

Adjunction

$$a \vdash_{\mathcal{S}} b \to c \qquad \text{if and only if} \qquad a \times b \vdash_{\mathcal{S}} c$$

Implicative tripos

Adjunction

$$a \vdash_{\mathcal{S}} b \to c \qquad \text{if and only if} \qquad a \times b \vdash_{\mathcal{S}} c$$

($\looparrowright$ $(\mathcal{A}/\mathcal{S}, \vdash_{\mathcal{S}}, \times, +, \to)$ *is a Heyting algebra*)

Tripos:

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

Collapse criteria

The following are equivalent:

1. $\mathcal{T}$ is isomorphic to a forcing tripos
2. $\mathcal{S} \subseteq \mathcal{A}$ is a principal filter of $\mathcal{A}$.
3. $\mathcal{S} \subseteq \mathcal{A}$ is finitely generated and $\pitchfork \in \mathcal{S}$.

Implicative tripos

Adjunction

$$a \vdash_{\mathcal{S}} b \to c \qquad \text{if and only if} \qquad a \times b \vdash_{\mathcal{S}} c$$

(☞ $(\mathcal{A}/\mathcal{S}, \vdash_{\mathcal{S}}, \times, +, \to)$ *is a Heyting algebra*)

Tripos:

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

Collapse criteria

The following are equivalent:

1. $\mathcal{T}$ is isomorphic to a forcing tripos
2. $\mathcal{S} \subseteq \mathcal{A}$ is a principal filter of $\mathcal{A}$.
3. $\mathcal{S} \subseteq \mathcal{A}$ is finitely generated and $⋔ \in \mathcal{S}$.

Decomposing the arrow

Chapter 11

Logic:

$$A \to B \quad \triangleq \quad \neg A \vee B$$

Different axiomatic:

$$\begin{array}{lll} S1 & : & (A \vee A) \to A \\ S2 & : & A \to (A \vee B) \end{array} \qquad \begin{array}{lll} S3 & : & (A \vee B) \to (B \vee A) \\ S4 & : & (A \to B) \to ((C \vee A) \to (C \vee B)) \end{array}$$

λ-calculus:

$$\lambda x.t \triangleq \tilde{\mu}([x],\beta).\langle t \parallel \beta\rangle : \neg A \parr B$$

- $L^{\parr}$ fragment of Munch-Maccagnoni's system L
- embedding of the *call-by-name* λ-calculus

Disjunctive structures

Complete meet-semilattice $(\mathcal{A}, \preccurlyeq, ⅋, \neg)$:

1. $\neg$ is anti-monotonic
2. $⅋$ is monotonic
3. $\curlywedge_{b\in B}(a ⅋ b) = a ⅋ (\curlywedge_{b\in B} b)$ and $\curlywedge_{b\in B}(b ⅋ a) = (\curlywedge_{b\in B} b) ⅋ a$
4. $\neg \curlywedge_{a\in A} a = \curlyvee_{a\in A} \neg a$

Examples:

- complete Boolean algebras
- classical realizability in $L^{⅋}$:
  - $\mathcal{A} \triangleq \mathcal{P}(\Pi)$
  - $a \preccurlyeq b \triangleq a \supseteq b$
  - $a ⅋ b \triangleq (a, b)$
  - $\neg a \triangleq [a^{\perp\!\!\!\perp}]$

Induced implication

$(\mathcal{A}, \preccurlyeq, \xrightarrow{⅋})$ with $a \xrightarrow{⅋} b \triangleq \neg a ⅋ b$ is an implicative structure

Disjunctive structures

Complete meet-semilattice $(\mathcal{A}, \preccurlyeq, ⅋, \neg)$:

1. $\neg$ is anti-monotonic
2. $⅋$ is monotonic
3. $\bigcurlywedge_{b\in B}(a ⅋ b) = a ⅋ (\bigcurlywedge_{b\in B} b)$ and $\bigcurlywedge_{b\in B}(b ⅋ a) = (\bigcurlywedge_{b\in B} b) ⅋ a$
4. $\neg \bigcurlywedge_{a\in A} a = \bigcurlyvee_{a\in A} \neg a$

Examples:

- complete Boolean algebras
- classical realizability in $L^{⅋}$:
  - $\mathcal{A} \triangleq \mathcal{P}(\Pi)$
  - $a \preccurlyeq b \triangleq a \supseteq b$
  - $a ⅋ b \triangleq (a, b)$
  - $\neg a \triangleq [a^{\perp\!\!\perp}]$

Induced implication

$(\mathcal{A}, \preccurlyeq, \xrightarrow{⅋})$ with $a \xrightarrow{⅋} b \triangleq \neg a ⅋ b$ is an implicative structure

Disjunctive structures

Complete meet-semilattice $(\mathcal{A}, \preccurlyeq, \parr, \neg)$:

1. $\neg$ is anti-monotonic
2. $\parr$ is monotonic
3. $\curlywedge_{b\in B}(a \parr b) = a \parr (\curlywedge_{b\in B} b)$ and $\curlywedge_{b\in B}(b \parr a) = (\curlywedge_{b\in B} b) \parr a$
4. $\neg \curlywedge_{a\in A} a = \curlyvee_{a\in A} \neg a$

Examples:

- complete Boolean algebras
- classical realizability in $L^{\parr}$:
  - $\mathcal{A} \triangleq \mathcal{P}(\Pi)$
  - $a \preccurlyeq b \triangleq a \supseteq b$
  - $a \parr b \triangleq (a, b)$
  - $\neg a \triangleq [a^{\perp\!\!\!\perp}]$

Induced implication

$(\mathcal{A}, \preccurlyeq, \xrightarrow{\parr})$ with $a \xrightarrow{\parr} b \triangleq \neg a \parr b$ is an implicative structure

Interpreting $L^{\parr}$

Contexts:

- $(a,b) \triangleq a \parr b$
- $[a] \triangleq \neg a$
- $\mu^+.c \triangleq \bigcurlyvee_{a\in\mathcal{A}}\{a : c(a) \in \perp\!\!\!\perp\}$

Terms:

- $\mu^-.c \triangleq \bigcurlywedge_{a\in\mathcal{A}}\{a : c(a) \in \perp\!\!\!\perp\}$
- $\mu^{()}.c \triangleq \bigcurlywedge_{a,b\in\mathcal{A}}\{a \parr b : c(a,b) \in \perp\!\!\!\perp\}$
- $\mu^{[]}.c \triangleq \bigcurlywedge_{a\in\mathcal{A}}\{\neg a : c(a) \in \perp\!\!\!\perp\}$

Adequacy

1. for any term t, if $\Gamma \vdash t : A \mid \Delta$, then $(t[\sigma])^{\mathcal{A}} \preccurlyeq A[\sigma]^{\mathcal{A}}$;
2. for any context e, if $\Gamma \mid e : A \vdash \Delta$, then $(e[\sigma])^{\mathcal{A}} \succcurlyeq A[\sigma]^{\mathcal{A}}$;

Besides:

$$\begin{tikzcd} \lambda\text{-calculus} \arrow[r, "{[\![\,]\!]}"] \arrow[d, "{[\,]^{\mathcal{A}^{\to}}}"'] & L^{\parr} \arrow[d, "{[\,]^{\mathcal{A}^{\parr}}}"] \\ (\mathcal{A}^{\to}, \preccurlyeq, \to) \arrow[r, "\iota"] & (\mathcal{A}^{\parr}, \preccurlyeq, \parr, \neg) \end{tikzcd}$$

Interpreting $L^{\parr}$

Contexts:

- $(a,b) \triangleq a \parr b$
- $[a] \triangleq \neg a$
- $\mu^{+}.c \triangleq \bigcurlyvee_{a\in\mathcal{A}}\{a : c(a) \in \perp\!\!\!\perp\}$

Terms:

- $\mu^{-}.c \triangleq \bigcurlywedge_{a\in\mathcal{A}}\{a : c(a) \in \perp\!\!\!\perp\}$
- $\mu^{()}.c \triangleq \bigcurlywedge_{a,b\in\mathcal{A}}\{a \parr b : c(a,b) \in \perp\!\!\!\perp\}$
- $\mu^{[]}.c \triangleq \bigcurlywedge_{a\in\mathcal{A}}\{\neg a : c(a) \in \perp\!\!\!\perp\}$

Adequacy

1. for any term t, if $\Gamma \vdash t : A \mid \Delta$, then $(t[\sigma])^{\mathcal{A}} \preccurlyeq A[\sigma]^{\mathcal{A}}$;
2. for any context e, if $\Gamma \mid e : A \vdash \Delta$, then $(e[\sigma])^{\mathcal{A}} \succcurlyeq A[\sigma]^{\mathcal{A}}$;

Besides:

$$\begin{tikzcd}
\lambda\text{-calculus} \arrow[r, "{[\![\ ]\!]}"] \arrow[d, "{[\ ]^{\mathcal{A}^{\rightarrow}}}"'] & L^{\parr} \arrow[d, "{[\ ]^{\mathcal{A}^{\parr}}}"] \\
(\mathcal{A}^{\rightarrow}, \preccurlyeq, \rightarrow) \arrow[r, "\iota"] & (\mathcal{A}^{\parr}, \preccurlyeq, \parr, \neg)
\end{tikzcd}$$

Disjunctive algebras

Bourbaki's axioms:

$$
\begin{array}{lcll}
s_1^{\parr} & \triangleq & \curlywedge_{a\in\mathcal{A}} & [(a \parr a) \to a] \\
s_2^{\parr} & \triangleq & \curlywedge_{a,b\in\mathcal{A}} & [a \to (a \parr b)] \\
s_3^{\parr} & \triangleq & \curlywedge_{a,b\in\mathcal{A}} & [(a \parr b) \to (b \parr a)] \\
s_4^{\parr} & \triangleq & \curlywedge_{a,b,c\in\mathcal{A}} & [(a \to b) \to (c \parr a) \to (c \parr b)] \\
s_5^{\parr} & \triangleq & \curlywedge_{a,b,c\in\mathcal{A}} & [(a \parr (b \parr c)) \to ((a \parr b) \parr c)]
\end{array}
$$

Separator $\mathcal{S}$:

(1) If $a \in \mathcal{S}$ and $a \preccurlyeq b$ then $b \in \mathcal{S}$ (upward closure)

(2) s_1, s_2, s_3, s_4 and s_5 are in $\mathcal{S}$ (combinators)

(3) If $a \xrightarrow{\parr} b \in \mathcal{S}$ and $a \in \mathcal{S}$ then $b \in \mathcal{S}$ (closure under modus ponens)

Internal logic

Recall:

$$a \vdash_{\mathcal{S}} b \quad \triangleq \quad a \xrightarrow{\parr} b \in \mathcal{S}$$

Sum type:

1. $a \parr b \vdash_{\mathcal{S}} a + b$
2. $a + b \vdash_{\mathcal{S}} a \parr b$

Negation:

1. $\neg a \vdash_{\mathcal{S}} a \xrightarrow{\parr} \bot$
2. $a \xrightarrow{\parr} \bot \vdash_{\mathcal{S}} \neg a$

Double-negation elimination:

1. $a \vdash_{\mathcal{S}} \neg\neg a$
2. $\neg\neg a \vdash_{\mathcal{S}} a$

Theorem

$\mathcal{S}$ is an implicative separator

Internal logic

Recall:

$$a \vdash_{\mathcal{S}} b \quad \triangleq \quad a \xrightarrow{\parr} b \in \mathcal{S}$$

Sum type:

1. $a \parr b \vdash_{\mathcal{S}} a + b$
2. $a + b \vdash_{\mathcal{S}} a \parr b$

Negation:

1. $\neg a \vdash_{\mathcal{S}} a \xrightarrow{\parr} \bot$
2. $a \xrightarrow{\parr} \bot \vdash_{\mathcal{S}} \neg a$

Double-negation elimination:

1. $a \vdash_{\mathcal{S}} \neg\neg a$
2. $\neg\neg a \vdash_{\mathcal{S}} a$

Theorem

$\mathcal{S}$ is an implicative separator

Conclusion

Disjunctive structures:

- induced by classical realizability
- allow to adequately embed $L^{⅋}$
- are implicative structures

Disjunctive algebras:

- are intrinsically classical
- are implicative algebras
- do not necessarily collapse to a forcing situation

Conclusion

Implicative algebras are more general.

Conclusion

Disjunctive structures:

- induced by classical realizability
- allow to adequately embed $L^{\parr}$
- are implicative structures

Disjunctive algebras:

- are intrinsically classical
- are implicative algebras
- do not necessarily collapse to a forcing situation

Conclusion

Implicative algebras are more general.

Conclusion

Disjunctive structures:

- induced by classical realizability
- allow to adequately embed $L^{\mathfrak{P}}$
- are implicative structures

Disjunctive algebras:

- are intrinsically classical
- are implicative algebras
- do not necessarily collapse to a forcing situation

Conclusion

Implicative algebras are more general.

Conjunctive algebras

Chapter 12

Same process:

1. Conjunctive structures $(\mathcal{A}, \preccurlyeq, \otimes, \neg)$
2. Adequate embedding of $\mathrm{L}^{\otimes}$
3. Conjunctive algebras

What we got:

- Duality between conjunctive and disjunctive structures
- Construction of conjunctive algebras from disjunctive algebras.

What we don't:

- Internal logic
- Triposes
- Construction of disjunctive algebras from conjunctive algebras.

Conjunctive algebras

Chapter 12

Same process:

1. Conjunctive structures $(\mathcal{A}, \preccurlyeq, \otimes, \neg)$
2. Adequate embedding of $L^{\otimes}$
3. Conjunctive algebras

What we got:

- Duality between conjunctive and disjunctive structures
- Construction of conjunctive algebras from disjunctive algebras.

What we don't:

- Internal logic
- Triposes
- Construction of disjunctive algebras from conjunctive algebras.

Conjunctive algebras

Chapter 12

Same process:

1. Conjunctive structures $(\mathcal{A}, \preccurlyeq, \otimes, \neg)$
2. Adequate embedding of $L^{\otimes}$
3. Conjunctive algebras

What we got:

- Duality between conjunctive and disjunctive structures
- Construction of conjunctive algebras from disjunctive algebras.

What we don't:

- Internal logic
- Triposes
- Construction of disjunctive algebras from conjunctive algebras.

Final picture

$a \to b = \neg a \parr b$

$a \triangleleft b = b \preccurlyeq a,\ \mathcal{S}^{\otimes} = \neg^{-1}(\mathcal{S}^{\parr})$

Implicative algebras

$\Rightarrow$	$\wedge$	$\forall$	$\mathcal{M} \vDash \_$
$\to$	$\times$	$\curlywedge$	$\_ \in \mathcal{S}$

Disjunctive algebras

$\neg$	$\vee$	$\forall$	$\mathcal{M} \vDash \_$
$\neg$	$\parr$	$\curlywedge$	$\_ \in \mathcal{S}^{\parr}$

Conjunctive algebras

$\neg$	$\wedge$	$\exists$	$\mathcal{M} \vDash \_$
$\neg$	$\otimes$	$\curlyvee$	$\_ \in \mathcal{S}^{\otimes}$

$a \to b = \neg a \curlyvee b$

$\parr = \curlyvee$

$\otimes = \curlywedge$

Boolean algebras

$\neg$	$\wedge$	$\forall$	$\mathcal{M} \vDash \_$
$\neg$	$\curlywedge$	$\curlywedge$	$\_ \in \mathcal{F}$

(Bonus: proven in Coq)

Final picture

$a \to b = \neg a \parr b$ (Disjunctive algebras → Implicative algebras)

$a \blacktriangleleft b = b \preccurlyeq a,\ \mathcal{S}^{\otimes} = \neg^{-1}(\mathcal{S}^{\parr})$ (Disjunctive algebras → Conjunctive algebras)

Implicative algebras

$\Rightarrow$	$\wedge$	$\forall$	$\mathcal{M} \vDash \_$
$\to$	$\times$	$\bigcurlywedge$	$\_ \in \mathcal{S}$

Disjunctive algebras

$\neg$	$\vee$	$\forall$	$\mathcal{M} \vDash \_$
$\neg$	$\parr$	$\bigcurlywedge$	$\_ \in \mathcal{S}^{\parr}$

Conjunctive algebras

$\neg$	$\wedge$	$\exists$	$\mathcal{M} \vDash \_$
$\neg$	$\otimes$	$\bigcurlyvee$	$\_ \in \mathcal{S}^{\otimes}$

$a \to b = \neg a \curlyvee b$ (Boolean algebras → Implicative algebras)

$\parr = \curlyvee$ (Boolean algebras → Disjunctive algebras)

$\otimes = \curlywedge$ (Boolean algebras → Conjunctive algebras)

Boolean algebras

$\neg$	$\wedge$	$\forall$	$\mathcal{M} \vDash \_$
$\neg$	$\curlywedge$	$\bigcurlywedge$	$\_ \in \mathcal{F}$

(Bonus: proven in Coq  *)*

Further works

Several directions to explore:

1. Complete the duality:
    - Conjunctive triposes?
    - From conjunctive algebras to disjunctive algebras?
2. Combination of disjunctive and conjunctive algebras:
    - Would it collapse to a forcing situation?
    - Any chance to get call-by-push-value algebras?
3. Algebraic counterpart of strategy/side-effects:
    - Lazy algebras?
    - Algebraic counterpart of memory?
    - ...

Further works

Several directions to explore:

1. Complete the duality:
   - Conjunctive triposes?
   - From conjunctive algebras to disjunctive algebras?
2. Combination of disjunctive and conjunctive algebras:
   - Would it collapse to a forcing situation?
   - Any chance to get call-by-push-value algebras?
3. Algebraic counterpart of strategy/side-effects:
   - Lazy algebras?
   - Algebraic counterpart of memory?
   - ...

Further works

Several directions to explore:

1. Complete the duality:
    - Conjunctive triposes?
    - From conjunctive algebras to disjunctive algebras?
2. Combination of disjunctive and conjunctive algebras:
    - Would it collapse to a forcing situation?
    - Any chance to get call-by-push-value algebras?
3. Algebraic counterpart of strategy/side-effects:
    - Lazy algebras?
    - Algebraic counterpart of memory?
    - ...

Further works

Several directions to explore:

1. Complete the duality:
    - Conjunctive triposes?
    - From conjunctive algebras to disjunctive algebras?
2. Combination of disjunctive and conjunctive algebras:
    - Would it collapse to a forcing situation?
    - Any chance to get call-by-push-value algebras?
3. Algebraic counterpart of strategy/side-effects:
    - Lazy algebras?
    - Algebraic counterpart of memory?
    - …

Thank you for you attention.

Implicative tripos

Tripos:

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

For the product $\mathcal{A}^I$, two possible separators:

$$\begin{array}{rcll} \mathcal{S}^I & \triangleq & \prod_{i\in I}\mathcal{S} & \text{(product)} \\ \mathcal{S}[I] & \triangleq & \{a\in\mathcal{A}^I : \exists s\in S.\forall i\in I.s\preccurlyeq a_i\} & \text{(uniform)} \end{array}$$

The diagram:

$$\begin{tikzcd} \mathcal{A}^I \arrow[r, two heads, "{[\cdot]_{/\mathcal{S}[I]}}"] \arrow[d, "{[\cdot]_{/\mathcal{S}^I}}"'] & \mathcal{A}^I/\mathcal{S}[I] \arrow[dl, "\iota_I"] \arrow[d, "\rho_I"] & = \mathcal{T}(I) \\ \mathcal{A}^I/\mathcal{S}^I \arrow[r, two heads, "\sim", "\varrho_I"'] & (\mathcal{A}/\mathcal{S})^I & = \mathcal{T}(1)^I \end{tikzcd}$$

Application in $\mathbf{L}^{\otimes}$:

$$t\,u \quad \triangleq \quad \mu\alpha.\langle t \parallel \mu[\beta].\langle (u,[\alpha]) \parallel \beta\rangle\rangle$$

Application in conjunctive structures:

$$t@u \;\; = \;\; \bigwedge\{a : t \preccurlyeq \bigvee\{\neg b : u \otimes \neg a \preccurlyeq b\}$$