What can sequent calculus do for functional programs ?

in memory of
Peter Landin (1930-2009) and Gerhard Gentzen (1909-1945))

Pierre-Louis Curien (CNRS and University Paris 7)

visiting the Computer Lab at Cambridge on a

Leverhulme grant

The alphabet of the talk

We start from the basic Curry-Howard correspondence :

simply typed λ-terms
=
proofs of (minimal) intuitionistic logic in natural deduction

and show how to extend it to cover

- abstract machines, through sequent calculus : $\overline{\lambda}$ (Herbelin 1995)
- classical logic : $\mathcal{C}$ (Griffin 1990) and μ (Parigot 1992)
- call-by-value : $\tilde{\mu}$ (Curien-Herbelin 2000)
- polarisation and focalisation (Munch 2009, Curien-Munch 2009) : “types should reflect the evaluation order”

All these ingredients will be incorporated progressively in a syntax based on sequent calculus (whose letter is L).

Natural deduction and sequent calculus

Both invented by Gerhard Gentzen, who introduced natural deduction first (NJ, NK), and then moved to sequent calculus (LJ ,LK) which he found easier to work with (for consistency proofs).

In natural deduction, the rules for logical connectives are (right)

introduction and elimination rules

In sequent calculus, there are no elimination rules :

right introduction and left introduction rules

i.e. right elimination traded against left introduction

Implication in Natural deduction

$$\frac{}{\Gamma, A \vdash A}$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}$$

$$\frac{\Gamma \vdash A \to B \quad \Gamma \vdash A}{\Gamma \vdash B}$$

The basic picture of the Curry-Howard isomorphism

$$\frac{}{\Gamma, A \vdash x : A}$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B}$$

$$\frac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}$$

Read this as

– either adding the red to the blue : λ-calculus as a language of proof terms

– or adding the blue to the red : provide a typing system for (untyped) λ-calculus based programs

From abstract machines to sequent calculus

The goal of the next few slides is to convey the idea that sequent calculus could have arisen from the goal of providing a typing system for the states of an abstract machine for the

“mechanical evaluation of expressions”

(to quote the title of Peter Landin’s pioneering paper)

Krivine abstract machine

Here is a simple device for executing a (closed) λ-term in call-by-name :

$$\boxed{\begin{array}{lcl} \langle M N \mid E \rangle & \longrightarrow & \langle M \mid N \cdot E \rangle \\ \langle \lambda x.M \mid N \cdot E \rangle & \longrightarrow & \langle M[N/x] \mid E \rangle \end{array}}$$

A state of the machine is thus a pair $\langle M \mid E \rangle$ where

– M is "where the computation is currently active"

– E is the stack of *all* things that are waiting to be done in the future, or the continuation, or the evaluation context

In λ-calculus litterature, contexts are more traditionally presented as terms with a hole : if you are more used to that, read

– $\langle M \mid E \rangle$ as $E[M]$ (fill the hole with M)

– $M \cdot E$ as $E[[]M]$ (fill the hole of E with the context $[]M$)

Typing Krivine abstract machine (1/2)

We have three categories of terms :

Expressions	$M ::= x \parallel \lambda x.M \parallel M\,M$	(value producing)
Contexts	$E ::= [\,] \parallel M \cdot E$	(value expecting)
Commands	$c ::= \langle M \mid E \rangle$	(system)

and we set three kinds of typing judgements :

$$\Gamma \vdash M : A \qquad \Gamma \,|\, E : A \vdash R \qquad c : (\Gamma \vdash R)$$

where R is a (fixed) type of *final results*. We give typing rules for contexts and commands in the next slide.

Typing Krivine abstract machine (2/2)

$$\frac{}{\Gamma \mid [\,] : R \vdash R} \qquad \frac{\Gamma \vdash M : A \quad \Gamma \mid E : B \vdash R}{\Gamma \mid M \cdot E : A \to B \vdash R}$$

$$\frac{\Gamma \vdash M : A \quad \Gamma \mid E : A \vdash R}{\langle M \mid E \rangle : (\Gamma \vdash R)}$$

Stripping up the term information, the second and third rules are the following rules of *sequent calculus* :

left introduction	cut
$\frac{\Gamma \vdash A \quad \Gamma \mid B \vdash R}{\Gamma \mid A \to B \vdash R}$	$\frac{\Gamma \vdash A \quad \Gamma \mid A \vdash R}{\Gamma \vdash R}$

$\overline{\boldsymbol{\lambda}}$

So we have reached one of our milestones.

Herbelin's PhD thesis (1995) provides Curry-Howard for intuitionistic as well as (in a moment) classical sequent calculus.

He coined the letter $\overline{\lambda}$, because

$$\langle (\ldots (M N_1) \ldots N_n) \mid [\,] \rangle \longrightarrow^* \langle M \mid (N_1 \cdot (\ldots N_n \cdot [\,]) \ldots) \rangle$$

whence the bar (as a witness of this order-reversal of arguments).

But we left out the earlier milestone (Griffin's typing of Felleisen's control operator C). There we go !

Felleisen's control operator $\mathcal{C}$ (1/2)

Control operators are features that allow programmers to manipulate continuations.

Here is the behaviour of Felleisen's $\mathcal{C}$ constructor (best expressed at the level of an abstract machine) :

$$\boxed{\begin{array}{l} \langle \mathcal{C}(M) \mid E \rangle \longrightarrow \langle M \mid E^* \cdot [\,] \rangle \\ \langle E^* \mid N \cdot [\,] \rangle \longrightarrow \langle N \mid E \rangle \end{array}}$$

The first rule explains how the continuation E gets *captured*, and the second rule how it gets *restored*.

Felleisen's control operator $\mathcal{C}$ (2/2)

Griffin observed (1990) that the typing constraints inferred by reduction are met exactly when M and $\mathcal{C}(M)$ have type $(A \to R) \to R$ and A, resp. :

$$\langle \mathcal{C}(M) \overbrace{\mid}^{A} E\rangle \quad \langle M \overbrace{\mid}^{(A\to R)\to R} E^* \cdot [\,]\rangle \quad \langle E^* \underbrace{\mid}_{A\to R} N \cdot [\,]\rangle \quad \langle N \underbrace{\mid}_{A} E\rangle$$

But this is the rule that one adds to intutionistic natural deduction to make it classical, if we interpret R as $\perp$ (false) :

Negation elimination / Negation introduction

$$\frac{\Gamma \vdash M : (A \to R) \to R}{\Gamma \vdash \mathcal{C}(M) : A} \qquad \frac{\Gamma \mid E : A \vdash R}{\Gamma \vdash E^* : A \to R}$$

Hence, we get no less than Curry-Howard for classical logic ! But how does this sound in sequent calculus style ?

$\boldsymbol{\mu}$ (1/2)

Sequent calculus deals with the difference classical / intuitionistic in a different way. In classical sequent calculus, sequents have several formulas on the right and $\Gamma \vdash \Delta$ reads : if all formulas in Γ holds then at least one formula of Δ holds. Then it is natural to associate continuation variables with the formulas in Δ : a term will depend on its input variables, and on its output continuations.

With this in mind, let us come back to the operational rule for $\mathcal{C}(M)$: we can read it as "$\mathcal{C}(M)$ is a map $E \mapsto \langle M \mid E^* \cdot [\,]\rangle$" and write it with a new binder (that comes from 1992) :

$$\boxed{\mathcal{C}(M) = \mu\beta.\langle M \mid \beta^* \cdot [\,]\rangle}$$

where [ ] is now a continuation variable (for the "top-level", of type R).

(One can also define $E^* = \lambda x.\mu\alpha.\langle x \mid E\rangle$, with α, x fresh.)

μ (2/2)

The typing judgements are now :

$$\Gamma \vdash M : A \mid \Delta \qquad \Gamma \mid E : A \vdash \Delta \qquad c : (\Gamma \vdash \Delta)$$

The two relevant new typing rules are :

Right activation

$$\frac{}{\Gamma \mid \alpha : A \vdash \alpha : A, \Delta} \qquad \frac{c : (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \vdash \mu\alpha.c : A \mid \Delta}$$

plus a reduction rule :

$$\langle \mu\alpha.c \mid E \rangle \longrightarrow c[E/\alpha]$$

Note that in this setting, there is no more need to “reify” a context E into an expression E^*, it can be directly substituted for a continuation variable.

$\overline{\boldsymbol{\lambda}} + \boldsymbol{\mu}$

Similarly, we can read off a definition of MN from the operational rule for MN :

$$MN = \mu\beta.\langle M \mid N.\beta\rangle$$

and hence remove it, and arrive at a system in sequent calculus style **only** (no more elimination rule).

This yields Herbelin's $\overline{\lambda}\mu$-calculus :

Expressions	$M ::= x \parallel \lambda x.M \parallel \mu\alpha.c$
Contexts	$E ::= \alpha \parallel M \cdot E$
Commands	$c ::= \langle M \mid E\rangle$

which combines the first two milestones : “sequent calculus”, “classical".

Call by value

Let us step back to the λ-calculus. The following describes a call-by-value version of Krivine machine (cf. Landin's SECD machine, or Felleisen's CEK machine, or the Categorical Abstract Machine) :

$$\boxed{\begin{array}{lcl} \langle MN \mid E\rangle & \longrightarrow & \langle N \mid M \odot E\rangle \\ \langle V \mid M \odot E\rangle & \longrightarrow & \langle M \mid V \cdot E\rangle \end{array}}$$

(the operational rule for $\lambda x.M$ is unchanged)

Here, V is either a variable or an abstraction.

$\tilde{\mu}$

Again, we can read $M \odot E$ as “a map $V \mapsto \langle M \mid V \cdot E\rangle$”, or, introducing a new binder $\tilde{\mu}$ (binding now ordinary variables) :

$$\boxed{M \odot E = \tilde{\mu}x.\langle M \mid x \cdot E\rangle}$$

The typing rule for this operator is :

Left activation

$$\frac{c : (\Gamma, x : A \vdash \Delta)}{\Gamma \mid \tilde{\mu}x.c : A \vdash \Delta}$$

and the operational rule is

$$\langle V \mid \tilde{\mu}x.c\rangle \longrightarrow c[V/x] \qquad (V \text{ value})$$

We arrive at the following language of Curien-Herbelin 2000 (next slide) :

$\overline{\lambda}\mu\tilde{\mu}$ (by value)

Expressions	$M ::= V^{\Diamond} \parallel \mu\alpha.c$	$\Gamma \vdash M : A \,\vert\, \Delta$
Values	$V ::= x \parallel \lambda x.M$	$\Gamma \vdash V : A \,;\, \Delta$
Contexts	$e ::= \alpha \parallel V \cdot e \parallel \tilde{\mu}x.c$	$\Gamma \,\vert\, E : A \vdash \Delta$
Commands	$c ::= \langle M \mid E \rangle$	$c : (\Gamma \vdash \Delta)$

with a new judgement for values (more on this later) and an explicit coercion from values to expressions. The syntax for contexts (for which we use now e rather than E) is both extended ($\tilde{\mu}x.c$) and restricted ($V \cdot E$ instead of $M \cdot E$). The reduction rules are as follows :

$$\boxed{\begin{array}{ll} \langle (\lambda x.M)^{\Diamond} \mid V \cdot e \rangle \longrightarrow \langle M[V/x] \mid e \rangle & \text{(Plotkin 1975 !)} \\ \langle \mu\alpha.c \mid e \rangle \longrightarrow c[e/\alpha] & \\ \langle V^{\Diamond} \mid \tilde{\mu}x.c \rangle \longrightarrow c[V/x] & \end{array}}$$

The $\mu - \tilde{\mu}$ critical pair

Consider

$$\langle \mu\alpha.c_1 \mid \tilde{\mu}x.c_2 \rangle$$

For modelling CBV, we have given priority to μ (typically, when $\mu\alpha.c = N_1 N_2$ and $\tilde{\mu}x.c_2 = M \odot E$).

Symmetrically, one can read call-by-name as "giving priority to $\tilde{\mu}$".

In Curien-Herbelin 2000, we left also room for an unrestricted (and highly non-confluent) calculus where

$$c_1[\tilde{\mu}x.c_2/\alpha] \longleftarrow \langle \mu\alpha.c_1 \mid \tilde{\mu}x.c_2 \rangle \longrightarrow c_2[\mu\alpha.c_1/x]$$

Here we instead further investigate the logical meaning of call-by-value and call-by-name in terms of polarities and focalisation.

The dynamics of sequent calculus : cut elimination

In sequent calculus, computation consists in eliminating cuts, by replacing cuts progressively by cuts on smaller formulas. For example :

$$\langle (\lambda x.M)^{\Diamond} \overbrace{\mid}^{A\to B} V \cdot e\rangle \longrightarrow \langle M[V \underbrace{/}_{A} x] \underbrace{\mid}_{B} e\rangle$$

Now, the above critical pair is "devastating", as when both x, α are fresh (weakening on both sides), we get computational inconsistency (everything is equal to everything) :

$$c_1 \longleftarrow \langle \mu\alpha.c_1 \mid \tilde{\mu}x.c_2\rangle \longrightarrow c_2$$

We need less symmetry, less proofs, and less freedom in cut-elimination ! We work now with conjunction, disjunction, and negation.

Irreversibility versus reversibility

We adopt an irreversible style for the right introduction of disjunction, while we use a reversible style for the left introduction of conjunction :

$$\frac{\Gamma, A_1, A_2 \vdash \Delta}{\Gamma, A_1 \wedge A_2 \vdash \Delta} \qquad \frac{\Gamma \vdash A_1, \Delta}{\Gamma \vdash A_1 \vee A_2, \Delta} \qquad \frac{\Gamma \vdash A_2, \Delta}{\Gamma \vdash A_1 \vee A_2, \Delta}$$

Implicitly, this describes two disjunctions (since the behaviour of the conjunction on the other side is different) and two conjunctions. We shall make this soon explicit.

Focalised proof search

The (weak) focalisation discipline is a proof search strategy that alternates between right and left phases, as follows :

- Left phase : Decompose formulas on the left, in any order. Every decomposition of a negation on the left feeds the right part of the sequent. At any moment, one can change the phase from left to right.
- Right phase : Choose a formula A on the right, and hereditarily decompose it in all branches of the proof search. This focusing in any branch can only end with an axiom (which ends the proof search in that branch), or with a decomposition of a negation, which prompts a phase change back to the left. Etc. . .

Polarisation

– We define a syntax of positive (or strict), and negative (or lazy) formulas :

$$P ::= X \mid P \otimes P \mid P \oplus P \mid {\downarrow}N$$
$$N ::= \overline{X} \mid N \& N \mid N \invamp N \mid {\uparrow}P$$

– We rename $\boxed{A \wedge B, A \vee B, \neg A}$ as $\boxed{P \otimes Q, P \oplus Q, {\downarrow}\overline{P}}$

– We bring all formulas to the right of $\vdash$ (the left formulas becoming negative)

– The typing judgements are now

$$\vdash v : P \mid \Delta \qquad \vdash V : P \,;\, \Delta \qquad \vdash E : N \mid \Delta \qquad c : (\vdash \Delta)$$

We have thus split the (controlled) version of classical negation into

– classical negation proper, which is not a connective, but is defined by the following De Morgan equations :

$$\overline{X} = \overline{X} \qquad \overline{P_1 \otimes P_2} = \overline{P_1} \invamp \overline{P_2} \qquad \overline{P_1 \oplus P_2} = \overline{P_1} \& \overline{P_2} \qquad \overline{{\downarrow}N} = {\uparrow}\overline{N}$$

– and a polarity change operator $\downarrow$

$\mu\tilde{\mu}$ proof terms for focalised classical proofs (1/2)

Values correspond to focusing phases (“ ;” is Girard’s stoup) :

$$\frac{}{\vdash x : P \,;\, x : \overline{P}, \Delta} \qquad \frac{\vdash e : N \mid \Delta}{\vdash e^{\bullet} : {\downarrow}N \,;\, \Delta}$$

$$\frac{\vdash V_1 : P_1 \,;\, \Delta \qquad \vdash V_2 : P_2 \,;\, \Delta}{\vdash (V_1, V_2) : P_1 \otimes P_2 \,;\, \Delta}$$

$$\frac{\vdash V_1 : P_1 \,;\, \Delta}{\vdash \mathit{inl}(V_1) : P_1 \oplus P_2 \,;\, \Delta} \qquad \frac{\vdash V_2 : P_2 \,;\, \Delta}{\vdash \mathit{inr}(V_2) : P_1 \oplus P_2 \,;\, \Delta}$$

Expressions :

$$\frac{\vdash V : P \,;\, \Delta}{\vdash V^{\Diamond} : P \mid \Delta} \qquad \frac{C : (\vdash \alpha : P \,,\, \Delta)}{\vdash \mu\alpha.C : P \mid \Delta}$$

$\mu\tilde{\mu}$ proof terms for focalised classical proofs (2/2)

Commands are typed using counter-patterns ...

$$\frac{\vdash v : P \mid \Delta \qquad \vdash e : \overline{P} \mid \Delta}{\langle v \mid e \rangle : (\vdash \Delta)}$$

$$\frac{C : (\vdash \alpha : P\ ,\ \Delta)}{C : (\vdash \alpha^{\bullet} : {\uparrow}P\ ,\ \Delta)} \qquad \frac{C : (\vdash q_1 : N_1\ ,\ q_2 : N_2\ ,\ \Delta)}{C : (\vdash (q_1, q_2) : N_1 \parr N_2\ ,\ \Delta)}$$

$$\frac{C_1 : (\vdash q_1 : N_1\ ,\ \Delta) \qquad C_2 : (\vdash q_2 : N_2\ ,\ \Delta)}{[C_1\ {}^{q_1,q_2}\ C_2] : (\vdash [q_1, q_2] : N_1 \& N_2\ ,\ \Delta)}$$

which are abstracted as pattern-matching binders in contexts :

$$\frac{}{\vdash \alpha : \overline{P} \mid \alpha : P\ ,\ \Delta} \qquad \frac{C : (\vdash q : N, \Delta)}{\vdash \tilde{\mu}q.C : N \mid \Delta}$$

Counterpatterns (1/2)

We have introduced counter-patterns in order to follow closely the proof-search process. Summing up, we have the following syntax :

Simple commands	$c ::= \langle v \mid e \rangle$
Commands	$C ::= c \parallel [C \; {}^{q,q} \; C]$
Expressions	$v ::= V^{\Diamond} \parallel \mu\alpha.C$
Values	$V ::= x \parallel (V, V) \parallel inl(V) \parallel inr(V) \parallel e^{\bullet}$
Contexts	$e ::= \alpha \parallel \tilde{\mu}q.C$
Counterpatterns	$q ::= x \parallel \alpha^{\bullet} \parallel (q, q) \parallel [q, q]$

Counterpatterns (2/2)

And the dynamics is as follows :

$$\boxed{\begin{array}{l} \langle V^{\Diamond} \mid \tilde{\mu} q.C \rangle \longrightarrow C[V/q] \\ \\ C[(V_1, V_2)/(q_1, q_2), \sigma] \longrightarrow C[V_1/q_1, V_2/q_2, \sigma] \\ C[e^{\bullet}/\alpha^{\bullet}, \sigma] \longrightarrow C[e/\alpha, \sigma] \\ [C_1 \;{}^{q_1,q_2}\; C_2][inl(V_1)/[q_1, q_2], \sigma] \longrightarrow C_1[V_1/q_1, \sigma] \\ [C_1 \;{}^{q_1,q_2}\; C_2][inr(V_2)/[q_1, q_2], \sigma] \longrightarrow C_2[V_2/q_2, \sigma] \end{array}}$$

where $\sigma = \ldots, V/q, \ldots, e/\alpha, \ldots$ is a simultaneous substitution, with associated domain $dom(\sigma) = \{\ldots, q \ldots, \alpha, \ldots\}$

CBV on the left (V/x) and CBN on the right (e/α)

This system is confluent and (strongly) normalising : focalisation saves us from the otherwise wild non-determinism of classical logic.

Embedding CBV and CBN λ-calculi

CBV translation : $[\![X]\!]^+_{\mathtt{v}} = X \quad [\![A \to B]\!]^+_{\mathtt{v}} = {\downarrow}(\overline{[\![A]\!]^+_{\mathtt{v}}} \parr {\uparrow}[\![B]\!]^+_{\mathtt{v}})$

$$[\![x]\!]^+_{\mathtt{v}} = x^\diamond$$
$$[\![\lambda x.M]\!]^+_{\mathtt{v}} = ((\tilde{\mu}(x,\alpha^\bullet).\langle [\![M]\!]^+_{\mathtt{v}} \mid \alpha \rangle)^\bullet)^\diamond$$
$$[\![MN]\!]^+_{\mathtt{v}} = \mu\alpha.\langle [\![N]\!]^+_{\mathtt{v}} \mid \tilde{\mu}x.\langle [\![M]\!]^+_{\mathtt{v}} \mid \tilde{\mu}\alpha^\bullet.\langle (x,\alpha^\bullet)^\diamond \mid \alpha \rangle\rangle\rangle$$

CBN translation : $[\![X]\!]^+_{\mathtt{n}} = \overline{X} \quad [\![A \to B]\!]^+_{\mathtt{n}} = {\uparrow}\overline{[\![A]\!]^+_{\mathtt{n}}} \parr [\![B]\!]^+_{\mathtt{n}}$

$[\![x]\!]^+_{\mathtt{n}} = \overline{x}$ (for some continuation variable associated with x)
$$[\![\lambda x.M]\!]^+_{\mathtt{n}} = \tilde{\mu}(\overline{x}^\bullet, y).\langle y^\diamond \mid [\![M]\!]^+_{\mathtt{n}} \rangle$$
$$[\![MN]\!]^+_{\mathtt{n}} = \tilde{\mu}y.\langle ([\![N]\!]^+_{\mathtt{n}})^\bullet, y)^\diamond \mid [\![M]\!]^+_{\mathtt{n}} \rangle$$

The order of evaluation is reflected at the level of types.

Strong focalisation

The focalised syntax that we have given remains a bit “half-cooked” : it does some quotient on the proofs (typically the order of decomposition of two ⅋’s is not recorded), but the order of decomposition of &’s is recorded (via a tree of $[C \;{}^{q,q}\; C]$).

Our syntax with counterpatterns pays off only when we force maximal negative phases : a negative formula must be decomposed completely, reaching a $\uparrow$ (prompting a change of phase) or an atom. Positive phases are also “more maximal” since the positive axiom is now atomic.

Strongly focalising proofs are complete : any provable classical sequent is provable through this discipline.

Patterns

We are ready for our last step (Curien-Munch 2009). We can reformulate values as :

Atomic values	$\mathcal{V} ::= x \parallel e^\bullet$
Values	$V ::= p \, \langle \mathcal{V}_i / i \mid i \in p \rangle$
Patterns	$p ::= x \parallel \alpha^\bullet \parallel (p,p) \parallel inl(p) \parallel inr(p)$

where $\mathcal{V}_i = y$ when $i = x$, and $\mathcal{V}_i = e^\bullet$ when $i = \alpha^\bullet$ and

$$\frac{}{x \in x} \quad \frac{}{\alpha^\bullet \in \alpha^\bullet} \quad \frac{i \in p_1}{i \in (p_1, p_2)} \quad \frac{i \in p_2}{i \in (p_1, p_2)} \quad \frac{i \in p_1}{i \in inl(p_1)} \quad \frac{i \in p_2}{i \in inr(p_2)}$$

Synthetic connectives

We may then package blocks of positive (resp. negative) connectives together, completely forgetting the order of the decomposition of the negatives (remember that they are reversible, so their order does not matter), and replace

$$\tilde{\mu} q.C \qquad \text{with} \qquad \tilde{\mu} q.\{p \mapsto c_p \mid q \perp p\}$$

The orthogonality relation between patterns and counterpatterns is defined as follows :

$$\frac{}{x \perp x} \qquad \frac{}{\alpha^\bullet \perp \alpha^\bullet}$$

$$\frac{q_1 \perp p_1 \quad q_2 \perp p_2}{(q_1, q_2) \perp (p_1, p_2)} \qquad \frac{q_1 \perp p_1}{[q_1, q_2] \perp inl(p_1)} \qquad \frac{q_2 \perp p_2}{[q_1, q_2] \perp inr(p_2)}$$

The S-calculus

$$\boxed{\begin{array}{ll} c ::= \langle V^{\Diamond} \mid e \rangle & \\ \mathcal{V} ::= x \parallel e^{\bullet} & \\ V ::= p\,\langle \mathcal{V}_i/i \mid i \in p \rangle & p ::= x \parallel \alpha^{\bullet} \parallel (p,p) \parallel \mathit{inl}(p) \parallel \mathit{inr}(p) \\ e ::= \alpha \parallel \tilde{\mu} q.\{p \mapsto c_p \mid q \perp p\} & q ::= x \parallel \alpha^{\bullet} \parallel (q,q) \parallel [q,q] \end{array}}$$

cf. Zeilberger 2008: $\{p \mapsto c \mid q \perp p\}$ as function from patterns to commands.

$$\boxed{\begin{array}{ll} (\tilde{\mu}^{+}) & \langle (p\,\langle \ldots, y/x, \ldots, e^{\bullet}/\alpha^{\bullet} \ldots \rangle)^{\Diamond} \mid \tilde{\mu} q.\{p \mapsto c_p \mid q \perp p\} \rangle \\ & \downarrow \\ & c_p\,\{\ldots, y/x, \ldots, e/\alpha, \ldots \rangle\} \end{array}}$$

(select field p, and then substitute)

As the final step in this journey, we have obtained a term-calculus for (strongly focalised) classical proofs that "has the simplicity of" the λ-calculus.

Conclusion

Sequent calculus, controlled via the μ and $\tilde{\mu}$ binders, provides us with a robust infrastructure to enforce evaluation orders, both in programming, and for proof-theoretical investigations.

The polarisation of logic allows us to reflect strictness and lazyness at the level of types (Zeilberger 2008 and Munch 2009, independently).

We hope to apply this infrastructure to the verification of a "less abstract" abstract machine like Leroy's ZINC machine.