Curry-Howard Correspondence

for Classical Logic

Stéphane Graham-Lengrand

CNRS, Laboratoire d'Informatique de l'X

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

Stephane.Lengrand@Polytechnique.edu

Lecture II
Confluence?

Review from the previous lecture

Easy enough to introduce proof-terms to represent classical proofs

symmetry of classical logic = symmetry between programs and *continuations*

use of classical reasoning = *control* = programs can capture their continuations

Curien-Herbelin-Wadler - typing

$$\frac{}{\Gamma, x : A \vdash x : A \; ; \Delta} \qquad \frac{}{\Gamma \; ; \alpha : A \vdash \alpha : A, \Delta}$$

$$\frac{\Gamma, x : A \vdash t : B \; ; \Delta}{\Gamma \vdash \lambda x.t : A \to B \; ; \Delta} \qquad \frac{\Gamma \vdash t : A \; ; \Delta \quad \Gamma \; ; e : B \vdash \Delta}{\Gamma \; ; t :: e : A \to B \vdash \Delta}$$

$$\frac{\Gamma \vdash t_1 : A_1 \; ; \Delta \quad \Gamma \vdash t_2 : A_2 \; ; \Delta}{\Gamma \vdash \langle t_1, t_2 \rangle : A_1 \wedge A_2 \; ; \Delta} \qquad \frac{\Gamma \; ; e : A_i \vdash \Delta}{\Gamma \; ; \mathsf{inj}_i(e) : A_1 \wedge A_2 \vdash \Delta}$$

$$\frac{\Gamma \vdash t : A_i \; ; \Delta}{\Gamma \vdash \mathsf{inj}_i(t) : A_1 \vee A_2 \; ; \Delta} \qquad \frac{\Gamma \; ; e_1 : A_1 \vdash \Delta \quad \Gamma \; ; e_2 : A_2 \vdash \Delta}{\Gamma \; ; \langle e_1, e_2 \rangle : A_1 \vee A_2 \vdash \Delta}$$

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

$$\frac{\Gamma \vdash t : A \; ; \Delta \quad \Gamma \; ; e : A \vdash \Delta}{\langle t \bullet e \rangle : (\Gamma \vdash \Delta)}$$

Contents

I. Reduction

Curien-Herbelin-Wadler - reduction

Reduction

$$
\begin{array}{llcl}
(\to) & \langle \lambda x.t_1 \bullet t_2 :: e \rangle & \to & \langle t_2 \bullet \mu x.\langle t_1 \bullet e \rangle \rangle \\
(\wedge) & \langle \langle t_1, t_2 \rangle \bullet \mathsf{inj}_i(e) \rangle & \to & \langle t_i \bullet e \rangle \\
(\vee) & \langle \mathsf{inj}_i(t) \bullet \langle e_1, e_2 \rangle \rangle & \to & \langle t \bullet e_i \rangle \\
 & \langle \mu\beta.c \bullet e \rangle & \to & \{{}^{e}\!/\!_{\beta}\}c \\
 & \langle t \bullet \mu x.c \rangle & \to & \{{}^{t}\!/\!_{x}\}c
\end{array}
$$

Theorem : Subject Reduction **OK**

Theorem : Progress? **OK**

Cuts remaining in normal forms are of the form $\langle x \bullet e \rangle$ and $\langle t \bullet \alpha \rangle$,

i.e. they represent contraction-left and contraction-right

Theorem : Normalisation? **OK**

(Barbanera-Berardi's symmetric reducibility candidates, see next lecture)

Symmetry of LK = Symmetry of terms vs. continuations. Now in the very syntax.

II. (Non-)confluence

Lafont's example

$$\dfrac{\dfrac{\dfrac{\vdots\,\pi}{\Gamma \vdash \Delta}}{\Gamma \vdash \Delta, A} \qquad \dfrac{\dfrac{\vdots\,\pi'}{\Gamma' \vdash \Delta'}}{\Gamma', A \vdash \Delta'}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$$

Two ways to eliminate the cut: $\dfrac{\begin{matrix}\vdots\,\pi \\ \Gamma \vdash \Delta\end{matrix}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$ or $\dfrac{\begin{matrix}\vdots\,\pi' \\ \Gamma' \vdash \Delta'\end{matrix}}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$

but we could have the mix rule: $\dfrac{\Gamma \vdash \Delta \quad \Gamma' \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}$

Do we want this derivation as a normal proof?

Lafont's example (in an additive world)

$$\dfrac{\dfrac{\begin{array}{c}\vdots\,\pi\\ \Gamma\vdash\Delta\end{array}}{\Gamma\vdash\Delta, A}\qquad\dfrac{\begin{array}{c}\vdots\,\pi'\\ \Gamma\vdash\Delta\end{array}}{\Gamma, A\vdash\Delta}}{\Gamma\vdash\Delta}$$

Two ways to eliminate the cut: $\begin{array}{c}\vdots\,\pi\\ \Gamma\vdash\Delta\end{array}$ or $\begin{array}{c}\vdots\,\pi'\\ \Gamma\vdash\Delta\end{array}$

but we could have: $\dfrac{\Gamma\vdash\Delta\quad\Gamma\vdash\Delta}{\Gamma\vdash\Delta}$

Do we want this derivation as a normal proof?

More problematic example

$$
\dfrac{\dfrac{\begin{array}{c}\vdots\,\pi\\ \Gamma \vdash \Delta, A, A\end{array}}{\Gamma \vdash \Delta, A} \qquad \dfrac{\begin{array}{c}\vdots\,\pi'\\ \Gamma, A, A \vdash \Delta\end{array}}{\Gamma, A \vdash \Delta}}{\Gamma \vdash \Delta}
$$

$$
\begin{array}{ccc}\vdots\,\pi' & & \vdots\,\pi'\\ \bullet & & \bullet\\ & \vdots\,\pi(\simeq) & \\ \Gamma \vdash \Delta & & \end{array}
\qquad \text{or} \qquad
\begin{array}{ccc}\vdots\,\pi & & \vdots\,\pi\\ \bullet & & \bullet\\ & \vdots\,\pi'(\simeq) & \\ \Gamma \vdash \Delta & & \end{array}
$$

e.g.:

$$
\dfrac{(A \to B) \to A \vdash \color{red}{A} \qquad \color{red}{A}\color{black}{, A \to C, A \to D \vdash C \wedge D}}{(A \to B) \to A, A \to C, A \to D \vdash C \wedge D}
$$

In concrete terms

In Curien-Herbelin-Wadler's calculus Curien and Herbelin [2000]; Wadler [2003], both examples appear as:

$$\dfrac{\dfrac{c\colon (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \vdash \mu\alpha.c : A \,;\, \Delta} \qquad \dfrac{c' : (\Gamma, x : A \vdash \Delta)}{\Gamma \,;\, \mu x.c' : A \vdash \Delta}}{\langle \mu\alpha.c \bullet \mu x.c' \rangle : (\Gamma \vdash \Delta)}$$

α (resp. x) could be used 0 (weakening), 1, or several (contraction) times in c (resp. c')

$$\dfrac{c\colon (\Gamma \vdash \alpha : A, \Delta) \qquad \dfrac{c' : (\Gamma, x : A \vdash \Delta)}{\Gamma \,;\, \mu x.c' : A \vdash \Delta}}{\left\{{}^{\mu x.c'}\!/_{\alpha}\right\} c : (\Gamma \vdash \Delta)} \qquad \dfrac{\dfrac{c\colon (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \vdash \mu\alpha.c : A \,;\, \Delta} \qquad c' : (\Gamma, x : A \vdash \Delta)}{\left\{{}^{\mu\alpha.c}\!/_{x}\right\} c' : (\Gamma \vdash \Delta)}$$

(Dotted lines not inference rules, but properties of typing system)

In conclusion

Easy enough to give rewrite system on proof-terms to represent cut-elimination,

system follows the intuitions of continuations and control

Gives non-confluent calculus because cut-elimination is non-confluent in classical logic

because programs and continuations fight for the control of computation

This makes it very hard to give a semantics of classical proofs / typed proof-terms

Today's challenge: **Find a way to construct a denotational semantics**

Back to the main issue

Remember that a CCC with $\neg\neg A \simeq A$ collapses.

3 ways to get away:

1. Break the symmetry between $\wedge$ and $\vee$
2. Break the cartesian product (Dosen-Petric et al.)
3. Break the curryfication (Strassburger et al.)

In this course: Break the symmetry between $\wedge$ and $\vee$

Why? Only one among the three for which computational interpretations of cut-elimination are reasonably well-understood

Breaking the $\wedge\vee$ symmetry by the CBN/CBV approach

$$\dfrac{\dfrac{\vdots\ \pi}{\Gamma \vdash \Delta, A} \qquad \dfrac{\vdots\ \pi'}{\Gamma, A \vdash \Delta}}{\Gamma \vdash \Delta}$$

Give systematic priority to

- the right (push π into π')
- or to the left (push π' into π)

1. Both solutions make the calculus confluent.
2. Suggests to construct 2 denotational semantics $[\![c]\!]_{\mathsf{N}}$ and $[\![c]\!]_{\mathsf{V}}$ with the hope that:

   $[\![c_0]\!]_{\mathsf{N}} = [\![c_1]\!]_{\mathsf{N}}$ iff "$c_0 \longleftrightarrow^* c_1$ with systematic priority to the right"

   $[\![c_0]\!]_{\mathsf{V}} = [\![c_1]\!]_{\mathsf{V}}$ iff "$c_0 \longleftrightarrow^* c_1$ with systematic priority to the left"
3. Relates to the notions of *Call-by-name* and *Call-by-value*
   - Plotkin Plotkin [1975] CBV/CBN
   - Moggi Moggi [1989] monadic λ-calculus

III. From programming languages to rewriting theory

Call-by-name and call-by-value

```
proc MyFavoriteFunction(x){
... x ...
}
...
MyFavoriteFunction(A)
```

Should A be evaluated before entering the function (CBV) or when it is used (CBN)?

. . . depends on the compiler

. . . may depend on datatype (base types may have different behaviour)

In presence of side-effects, knowing which of the two the compiler implements, is vital

Functional programming

How to evaluate a functional program?

Evaluation should produce values. ex: Boolans *true, false*

In functional programming, *functions are values* (e.g. can be given as arguments)

$\Rightarrow$ No need to reduce them.

λ-calculus: a core functional language vs. a theory of functions

…equipped with an operational semantics (close to implementation)

…which can be expressed by an evaluation strategy

that selects a unique β-redex to reduce:

- Never reduce a λ-abstraction, as it is a "value" (this is called weak reduction)
- Always reduce M first in an application $M\ N$. Then:
  - If M is an abstraction: reduce the β-redex first (CBN)

    reduce N first (CBV)
  - Otherwise, reduce N (never happens with closed terms)

Strategies denoted $\longrightarrow_{\mathsf{CBN}}$ and $\longrightarrow_{\mathsf{CBV}}$

Neither is "better" than the other -cf. Haskell (CBN) vs. Caml (CBV)

λ-calculus: a core functional language vs. a theory of functions

...equipped with a denotational semantics (close to mathematical functions)

...where equalities are congruences (e.g. if $M = N$ then $\lambda x.M = \lambda x.N$)

and reductions are congruences (this is called strong reduction)

Formally, in λ-calculus:

values: $\lambda x.M$ and x (denoted $V\ldots$)

not values: $M\,N$

Why? because by evaluating $M\,N$, you may get something completely different

In this view,

"Call-by-name" = general β-reduction $(\lambda x.M)\ N \longrightarrow_{\beta} \{^{N}\!/_{x}\}M$

"Call-by-value" = restriction to arguments being values

$$(\lambda x.M)\ V \longrightarrow_{\beta_v} \{^{V}\!/_{x}\}M$$

λ-calculus: a core functional language vs. a theory of functions

Question:

CBN: Is there a relation between $\longrightarrow_{\text{CBN}}$ and $\longrightarrow_{\beta}$?

CBV: Is there a relation between $\longrightarrow_{\text{CBV}}$ and $\longrightarrow_{\beta_v}$?

Answer:

Clearly, $\longrightarrow_{\text{CBN}} \subseteq \longrightarrow_{\beta}$ and $\longrightarrow_{\text{CBV}} \subseteq \longrightarrow_{\beta_v}$

What about the other way round?

λ-calculus: a core functional language vs. a theory of functions

Bridge between weak and strong reductions = Plotkin's result Plotkin [1975]:

CBN: $\longrightarrow^*_\beta$ is the closure of $\longrightarrow^*_{\text{CBN}}$ under $\dfrac{M_1 \longrightarrow^*_{\text{CBN}} C[M_2] \qquad M_2 \longrightarrow M_3}{M_1 \longrightarrow C[M_3]}$

CBV: $\longrightarrow^*_{\beta_v}$ is the closure of $\longrightarrow^*_{\text{CBV}}$ under $\dfrac{M_1 \longrightarrow^*_{\text{CBV}} C[M_2] \qquad M_2 \longrightarrow M_3}{M_1 \longrightarrow C[M_3]}$

What's the point?

This result allows us to call CBN and CBV

not some operational semantics of some functional programming language

but some rewriting theories in λ-calculus.

IV. The comeback of continuations

Compiling with continuations

CBN/CBV= question of compilation

λ-calculus can be compiled into (a fragment of) itself!

called Continuation Passing Style (CPS)-translation

CBN-translation (Plotkin Plotkin [1975])

$$\begin{aligned}
\underline{x} &:= \lambda k.x\ k \\
\underline{\lambda x.M} &:= \lambda k.(k\ (\lambda x.\underline{M})) \\
\underline{M\ N} &:= \lambda k.\underline{M}\ (\lambda y.y\ \underline{N}\ k)
\end{aligned}$$

CBV-translation (Reynolds Reynolds [1972])

$$\begin{aligned}
\overline{x} &:= \lambda k.k\ x \\
\overline{\lambda x.M} &:= \lambda k.(k\ (\lambda x.\lambda k'.\overline{M}\ k')) \\
\overline{M\ N} &:= \lambda k.\overline{M}\ (\lambda y.\overline{N}\ (\lambda z.y\ z\ k))
\end{aligned}$$

What's the point? Look, arguments are always values!

$\Rightarrow$ CPS-evaluation (i.e. evaluation of the CPS-translated term) is strategy-indifferent

($\longrightarrow_\beta = \longrightarrow_{\beta_v}$ for the translated terms)

The CPS-translations preserve reductions

Theorem (Simulations - soundness)

CBN If $M \longrightarrow_{\beta} N$ then $\underline{M} \longrightarrow^{*}_{\beta} \underline{N}$

CBV If $M \longrightarrow_{\beta_v} N$ then $\overline{M} \longrightarrow^{*}_{\beta} \overline{N}$

Theorem (Simulations - completeness)

CBN If $\underline{M} \longleftrightarrow^{*}_{\beta} \underline{N}$ then $M \longleftrightarrow^{*}_{\beta} N$

CBV Not the case for CBV! (unless extended -Moggi Moggi [1989])

The CPS-translations preserve types

We deal here with simple types $\quad A, B, \ldots ::= \ \alpha \mid A{\to}B$

Assume $\Gamma \vdash M : A$. Do we have:

$\Gamma' \vdash \underline{M} : A'$ (for some Γ', A')? $\quad \Gamma'' \vdash \overline{M} : A''$ (for some Γ'', A'')?

CPS-translations reveal 2 classes of terms in the target: *values* & *continuations* (like k)

The types of values and continuations in the translated terms depend on CBN or CBV:

We choose or we add a particular atomic type R, an abstract type of *responses*, then

CBN		CBV	
$\underline{\alpha}$	$:= \alpha$	$\overline{\alpha}$	$:= \alpha$
$\underline{A \to B}$	$:= ((\underline{A}{\to}R){\to}R){\to}(\underline{B}{\to}R){\to}R$	$\overline{A \to B}$	$:= \overline{A}{\to}(\overline{B}{\to}R){\to}R$

Theorem : Preservation of typing

If $\Gamma \vdash M : A$ then $\quad (\underline{\Gamma}{\to}R){\to}R \vdash \underline{M} : (\underline{A}{\to}R){\to}R$

If $\Gamma \vdash M : A$ then $\quad \overline{\Gamma} \vdash \overline{M} : (\overline{A}{\to}R){\to}R$

Variants

Fischer's translation for CBV Fischer [1972]

$$\begin{aligned} \overline{x} &:= \lambda k.k\ x \\ \overline{\lambda x.M} &:= \lambda k.(k\ (\lambda k'.\lambda x.\overline{M}\ k')) \\ \overline{M\ N} &:= \lambda k.\overline{M}\ (\lambda y.\overline{N}\ (\lambda z.y\ k\ z)) \end{aligned} \qquad \begin{aligned} \overline{\alpha} &:= \alpha \\ \overline{A \to B} &:= (\overline{B}{\to}R) \to \overline{A}{\to}R \end{aligned}$$

Hofmann & Streicher's translation for CBN Hofmann and Streicher [1997].*using product types*

$$\begin{aligned} \underline{x} &:= \lambda k.x\ k \\ \underline{\lambda x.M} &:= \lambda\langle x, k\rangle.\underline{M}\ k \\ \underline{M\ N} &:= \lambda k.\underline{M}\ \langle \underline{N}, k\rangle \end{aligned} \qquad \begin{aligned} \underline{\underline{\alpha}} &:= \alpha{\to}R \\ \underline{\underline{A \to B}} &:= (\underline{\underline{A}}{\to}R) \times \underline{\underline{B}} \end{aligned}$$

Theorem : If $\Gamma \vdash M : A$ then $\underline{\underline{\Gamma}}{\to}R \vdash \underline{M} : \underline{\underline{A}}{\to}R$

CPS-translations and categorical semantics

Remember: simple-typed λ-terms have a semantics in a Cartesian Closed Category

CPS-translations compile the simply-typed λ-calculus into itself

in a semantically meaningful way:

We can now assign to a simply-typed λ-term M, the semantics (in a CCC) of $\underline{M}$ or $\overline{M}$

(semantics now depends on CBN/CBV).

By the simulation theorem, reductions are sound w.r.t. that semantics.

CPS-Fragment $\Rightarrow$ we need less than a CCC:

Exponentials just of the form $R^A \Rightarrow$ *Response Category*.

Sub-cat of the objects of that form: *Continuation category* = CCC + **rich structure**

also called *Control Category* (Selinger Selinger [2001])

Useful for classical logic.

V. Classical logic and CBN/CBV

Translating classical logic into intuitionistic logic

Turning P into P' by adding (enough) double negations, you get

If $\vdash_c P$ then $\vdash_i P'$.

Obviously, $\vdash_c P \leftrightarrow P'$.

$\neg\neg$-translation, Goedel's A-translation,...

$$
\begin{aligned}
\alpha^\bullet &:= \alpha \\
(A \to B)^\bullet &:= (((A^\bullet)\to\bot)\to\bot) \to (((B^\bullet)\to\bot)\to\bot)
\end{aligned}
$$

$$
\begin{aligned}
\alpha^\star &:= \alpha \\
(A \to B)^\star &:= A^\star\to(B^\star\to\bot) \to \bot
\end{aligned}
$$

!!!

Having selected a response type $\bot$, a continuation is a proof of negation

Why such a fuss about intuitionistic vs. classical, then?

If it suffices to add negations in a classical provable formula,

are the two logics really different?

Yes. Adding negations breaks nice properties of intuitionistic logic:

In intuitionistic logic:

If $\vdash A_1 \vee A_2$ then either $\vdash A_1$ or $\vdash A_2$.

If $\vdash \exists x A$ then there is t such that $\vdash \left\{{}^{t}\!/\!_{x}\right\} A$

Getting t from the proof of $\vdash \exists x A$ = **Witness extraction**

Also true in some theories, like arithmetics (Heyting arithmetics):

If $HA \vdash \exists x A$ then there is t such that $HA \vdash \left\{{}^{t}\!/\!_{x}\right\} A$

Cannot say anything when $\vdash \neg\neg(A_1 \vee A_2)$ or $\vdash \neg\neg\exists x A$

What to do with a *classical proof* of $\vdash \exists x A$?

If A is nice enough, **Classical witness extraction.**

Reminder: classical proof-terms Curien and Herbelin [2000]; Wadler [2003]

$$
\begin{array}{lll}
\text{terms} & t & ::= x \mid \mu\beta.c \mid \lambda x.t \mid \langle t_1, t_2 \rangle \mid \mathsf{inj}_i(t) \\
\text{continuations} & e & ::= \alpha \mid \mu x.c \mid t :: e \mid \langle e_1, e_2 \rangle \mid \mathsf{inj}_i(e) \\
\text{commands} & c & ::= \langle t \bullet e \rangle
\end{array}
$$

$$
\begin{array}{lrcl}
(\to) & \langle \lambda x.t_1 \bullet t_2 :: e \rangle & \to & \langle t_2 \bullet \mu x.\langle t_1 \bullet e \rangle \rangle \\
(\wedge) & \langle \langle t_1, t_2 \rangle \bullet \mathsf{inj}_i(e) \rangle & \to & \langle t_i \bullet e \rangle \\
(\vee) & \langle \mathsf{inj}_i(t) \bullet \langle e_1, e_2 \rangle \rangle & \to & \langle t \bullet e_i \rangle \\
 & \langle \mu\beta.c \bullet e \rangle & \to & \{{}^{e}\!/\!_{\beta}\}c \\
 & \langle t \bullet \mu x.c \rangle & \to & \{{}^{t}\!/\!_{x}\}c
\end{array}
$$

CBN and CBV for classical proof-terms Curien and Herbelin [2000]; Wadler [2003]

$$
\begin{array}{lll}
\text{term values} & V & ::= x \mid \lambda x.t \mid \langle V_1, V_2 \rangle \mid \mathsf{inj}_i(V) \\
\text{continuation values} & E & ::= \alpha \mid t :: E \mid \langle E_1, E_2 \rangle \mid \mathsf{inj}_i(E)
\end{array}
$$

$$
\begin{array}{rcl}
\langle \lambda x.t_1 \bullet t_2 :: E \rangle & \to & \langle t_2 \bullet \mu x.\langle t_1 \bullet E \rangle \rangle \\
\langle \langle t_1, t_2 \rangle \bullet \mathsf{inj}_i(E) \rangle & \to & \langle t_i \bullet E \rangle \\
\langle \mathsf{inj}_i(t) \bullet \langle E_1, E_2 \rangle \rangle & \to & \langle t \bullet E_i \rangle \\
\langle \mu\beta.c \bullet E \rangle & \to & \{E/\beta\}c \\
\langle t \bullet \mu x.c \rangle & \to & \{t/x\}c
\end{array}
\qquad
\begin{array}{rcl}
\langle \lambda x.t \bullet V :: e \rangle & \to & \langle V \bullet \mu x.\langle t \bullet e \rangle \rangle \\
\langle \langle V_1, V_2 \rangle \bullet \mathsf{inj}_i(e) \rangle & \to & \langle V_i \bullet e \rangle \\
\langle \mathsf{inj}_i(V) \bullet \langle e_1, e_2 \rangle \rangle & \to & \langle V \bullet e_i \rangle \\
\langle \mu\beta.c \bullet e \rangle & \to & \{e/\beta\}c \\
\langle V \bullet \mu x.c \rangle & \to & \{V/x\}c
\end{array}
$$

CBN CBV

plus some **focussing rules** to ensure progress.

The two reduction relations now denoted $\longrightarrow_{\mathsf{CBN}}$ and $\longrightarrow_{\mathsf{CBV}}$.

Theorem $\longrightarrow_{\mathsf{CBN}}$ and $\longrightarrow_{\mathsf{CBV}}$ are confluent

CPS-translations of classical proof-terms Curien and Herbelin [2000]; Wadler [2003]

It is possible to define CPS-translations of *terms*, *continuations*, and *commands*:

CBN	$\underline{t}$	$\underline{e}$	$\underline{c}$
CBV	$\overline{t}$	$\overline{e}$	$\overline{c}$

Theorem (Preservation of reduction)

CBN If $c_1 \longrightarrow_{\mathsf{CBN}} c_2$ then $\underline{c_1} \longrightarrow^*_\beta \underline{c_2}$

CBV If $c_1 \longrightarrow_{\mathsf{CBV}} c_2$ then $\overline{c_1} \longrightarrow^*_\beta \overline{c_2}$

Theorem (Preservation of typing) If $\begin{array}{c} \Gamma \vdash t : A \,;\, \Delta \\ \Gamma \,;\, e : A \vdash \Delta \\ c : (\Gamma \vdash \Delta) \end{array}$ then

$$\begin{array}{l} \underline{\underline{\Gamma}} \to R, \underline{\underline{\Delta}} \vdash \underline{t} : \underline{\underline{A}} \to R \\ \underline{\underline{\Gamma}} \to R, \underline{\underline{\Delta}} \vdash \underline{e} : (\underline{\underline{A}} \to R) \to R \\ \underline{\underline{\Gamma}} \to R, \underline{\underline{\Delta}} \vdash \underline{c} : R \end{array} \qquad \begin{array}{l} \overline{\Gamma}, \overline{\Delta} \to R \vdash \overline{t} : (\overline{A} \to R) \to R \\ \overline{\Gamma}, \overline{\Delta} \to R \vdash \overline{e} : \overline{A} \to R \\ \overline{\Gamma}, \overline{\Delta} \to R \vdash \overline{c} : R \end{array}$$

Using Hofman-Streicher Hofmann and Streicher [1997] Using Fischer Fischer [1972]

Categorical semantics

Define $[\![c]\!]_{\mathsf{V}} := [\![\overline{c}]\!]$ and $[\![c]\!]_{\mathsf{N}} := [\![\underline{c}]\!]$

where $[\![t]\!]$ is the semantics, in a response category, of a λ-term t in the CPS-fragment

Assume $c : (x_1 : A_1, \ldots, x_n : A_n \vdash \alpha_1 : B_1, \ldots, \alpha_m : B_m)$

Remember that a *control category* is the

sub-category of a response category $\mathcal{C}$

whose objects are in $\{R^A | A \in \mathcal{C}\}$

Write $R^A \invamp R^B$ for $R^{A \times B}$

CBN Write K_A for the object corresponding to $\underline{\underline{A}}$, and C_A for R^{K_A},

$$[\![c]\!]_{\mathsf{N}} : \begin{cases} (C_{A_1} \times \ldots \times C_{A_n} \times K_{B_1} \times \ldots \times K_{B_m}) \to R & \text{in a response category} \\ C_{A_1} \times \ldots \times C_{A_n} \to C_{B_1} \invamp \ldots \invamp C_{B_m} & \text{in a control category} \end{cases}$$

CBV Write V_A for the object corresponding to $\overline{A}$, K_A for R^{V_A} and C_A for R^{K_A},

$$[\![c]\!]_{\mathsf{V}} : \begin{cases} (V_{A_1} \times \ldots \times V_{A_n} \times K_{B_1} \times \ldots \times K_{B_m}) \to R & \text{in a response category} \\ K_{B_1} \times \ldots \times K_{B_m} \to K_{A_1} \invamp \ldots \invamp K_{A_n} & \text{in a control category} \\ K_{A_1} \otimes \ldots \otimes K_{A_n} \to K_{B_1} + \ldots + K_{B_m} & \text{in a co-control category} \\ & \text{(where } \otimes \text{ is the dual of } \invamp \text{)} \end{cases}$$

Semantics for classical proofs: the historical point of view

Here we see that

CBN= Control categories,

CBV= Co-Control categories.

The semantics validate the reductions

If $c \longrightarrow_{\text{CBN}} c'$ then $[\![c]\!]_{\text{N}} = [\![c']\!]_{\text{N}}$

If $c \longrightarrow_{\text{CBV}} c'$ then $[\![c]\!]_{\text{N}} = [\![c']\!]_{\text{V}}$

Today's goal is achieved

...by breaking the symmetry between $\wedge$ and $\vee$:

$\parr$ is not the dual of $\times$!!

(equivalently, $+$ is not the dual of $\otimes$)

Due to Selinger Selinger [2001].

Comes from preliminary works:

- De Groote, Barbanera, Berardi, Ong,...
- Hofmann, Streicher, Reus Hofmann and Streicher [1997]; Streicher and Reus [1998]

  Semantics of continuations.

  Question of Duality CBV/CBN (in $\lambda\mu$) is conjectured.

Perspectives and hot topics

Many variants have been studied

- variants of Parigot's $\lambda\mu$ have different properties with respect to separation. Delimited control (Saurin, Herbelin, etc)
- Lots of open issues on extensionality, observational equivalence and separation, η-conversion, etc...
- Classical calculi and focusing: Zeilberger, Herbelin, Munch, Houtmann.

Questions?

References

P.-L. Curien and H. Herbelin. The duality of computation. In *Proc. of the* 5^{th} *ACM SIGPLAN Int. Conf. on Functional Programming (ICFP'00)*, pages 233–243. ACM Press, 2000.

M. J. Fischer. Lambda calculus schemata. In *Proc. of the ACM Conf. on Proving Assertions about Programs*, pages 104–109. SIGPLAN Notices, Vol. 7, No 1 and SIGACT News, No 14, Jan. 1972.

M. Hofmann and T. Streicher. Continuation models are universal for $\lambda\mu$-calculus. In *Proc. of the 12th Annual IEEE Symp. on Logic in Computer Science*, pages 387–397. IEEE Computer Society Press, July 1997.

E. Moggi. Computational lambda-calculus and monads. In *Proc. of the 4th Annual IEEE Symp. on Logic in Computer Science*, pages 14–23. IEEE Computer Society Press, June 1989.

G. D. Plotkin. Call-by-name, call-by-value and the lambda-calculus. *Theoret. Comput. Sci.*, 1: 125–159, 1975.

J. C. Reynolds. Definitional interpreters for higher-order programming languages. In *Proc. of the ACM annual Conf.* , pages 717–740, 1972.

P. Selinger. Control categories and duality: on the categorical semantics of the $\lambda\mu$-calculus. *Math. Structures in Comput. Sci.*, 11(2):207–260, 2001.

T. Streicher and B. Reus. Classical logic, continuation semantics and abstract machines. *J. Funct. Programming*, 8(6):543–572, 1998. URL `http://journals.cambridge.org/action/displayAbstract?aid=44197`.

P. Wadler. Call-by-value is dual to call-by-name. In *Proc. of the 8th ACM SIGPLAN Int. Conf. on Functional programming (ICFP'03)*, volume 38, pages 189–201. ACM Press, Sept. 2003. ISBN 1581137567.