A Correspondence between
Full Normalization by Reduction
and
Full Normalization by Evaluation

Olivier Danvy
Aarhus University
Hommage to Pierre-Louis

- The teacher
- The scientist
Homage to Pierre-Louis
(en anglais dans le texte)

- The teacher
- The scientist
Gosia Biernacka’s PhD defense, 2005

• brunch at our home
Gosia Biernacka’s PhD defense, 2005

- brunch at our home
Gosia Biernacka’s PhD defense, 2005

• brunch at our home

• a math lesson to a 9-year-old
Gosia Biernacka’s PhD defense, 2005

- brunch at our home
- a math lesson to a 9-year-old
- room for growth
Flashback: Val d’Ajol, 1985

• a spring school
Flashback: Val d’Ajol, 1985

- a spring school
- a conversation about the $\lambda$ calculus
Flashback: Val d’Ajol, 1985

- a spring school
- a conversation about the λ calculus
- room for growth
Offshot

CPS Transformation of Beta-Redexes

with Lasse R. Nielsen

IPL 2005
Pierre-Louis, the computer scientist

1. the CAM
2. the $\lambda \rho$ calculus
3. an abstract machine for full normalization
Quick background

Formal semantics has many formats:

- structural operational semantics
- reduction semantics
- abstract/virtual machines
- natural semantics
- denotational semantics

There is beauty in each of them.
Wouldn’t it be nice...

...if there were a structural connection among these formats?
Example: the CAM

```
compile
    ▼
    run
```

Example: the CAM

The compiler is *compositional*!
Example: the CAM

Deforestation...
Example: the CAM

compile

run

Result: an abstract machine over $\lambda$-terms rather than over byte code.
Many other machines are like that

For $\lambda$ terms:

- SECD (Burge)
- VEC (Schmidt)
- Zinc (Leroy)
Why Abstract Machines (AMs) rather than Virtual Machines (VMs)?

Because seen as tail-recursive functional programs, AMs are often in defunctionalized form.
Defunctionalization (Reynolds)

- partition function spaces into summands:
  one for each $\lambda$ abstraction

- has a left inverse
Refunclionalizing an abstract machine

Result: a CPS program.
Refunctionalizing an abstract machine

Result: a CPS program.

The contexts are refunctrionalized into continuations.
Mandatory quote

“CPS is everywhere.”

– Pierre-Louis

Caminha, Portugal, Sep 2000
Mandatory quote

“CPS is everywhere.”

– Pierre-Louis

Caminha, Portugal, Sep 2000

NB. By which Pierre-Louis does not mean

“Cyber-Physical Systems”
Mandatory quote

“CPS is everywhere.”

– Pierre-Louis

Caminha, Portugal, Sep 2000

NB. By which Pierre-Louis does not mean “Cyber-Physical Systems” (I am just saying).
Wand’s combinator-based compilers

• yields similar virtual machines

• but without categorical combinators in mind
  (or at least, not as prominently)
The CPS transformation

- a way to obtain CPS programs
- has a left inverse
From a refunctionalized AM

Reversing the CPS transformation
gives the functional encoding
of a natural semantics.
A representation

Leci n'est pas une pipe.
Ceci n’est pas une lune.
Closure conversion (Landin)

- represent functions by pairing $\lambda$ abstractions with their lexical environment
- has a left inverse
From a natural semantics

The inverse of closure conversion gives a compositional evaluation function.
Definitional interpreters (Reynolds)

- closure conversion
- CPS transformation
- defunctionalization

...a functional correspondence
But what about small steps?

Reynolds’s functional correspondence applies:

From (representations of) SOS

to (representations of) reduction semantics

by CPS transformation and defunctionalization.
A beautiful coincidence

Contexts are defunctionalized continuations:

- **reduction**
  - small-step semantics

- **contexts**

- **evaluation**
  - big-step semantics

Plotkin, 1975
And also

• reduction does not get stuck (Plotkin)

• evaluation does not go wrong (Milner)
One-step reduction visually

```
分解 → 合并
```

Olivier Danvy, Venice – 9 Sep 2013
Reduction-based normalization visually
Refocusing
Reduction-free normalization visually
Result: an abstract machine

And not any abstract machine:

Often, one designed by someone else.
Example

- An abstract framework for environment machines
  Curien, TCS 1991
- A concrete framework for environment machines
  Biernacka & Danvy, TOCL 2007
The first type system for coroutines

Anton & Thiemann, APLAS 2010:

From a calculus of coroutines
to a compositional evaluation function.
The first type system for coroutines

Anton & Thiemann, APLAS 2010:

From a calculus of coroutines
to a compositional evaluation function.

Contribution: retrofit the types of the denotations
to the types of the terms.
The first type system for coroutines

Anton & Thiemann, APLAS 2010:

From a calculus of coroutines to a compositional evaluation function.

Contribution: retrofit the types of the denotations to the types of the terms.

Just like for shift and reset.
Two recent examples

- combinatory graph reduction (TOCL)
- the call-by-need $\lambda$ calculus (PPDP 2013)

See Ian Zerny’s PhD thesis.
Back to Pierre-Louis

pp. 65-66 of The Book,

an abstract machine for full normalization.

• It is mentioned in passing.\[^{a}\]

• It is incredibly minimal and elegant.

\[^{a}\text{En anglais dans le texte.}\]
Back to Pierre-Louis

pp. 65-66 of The Book,

an abstract machine for full normalization.

- It is mentioned en passant.\(^a\)

- It is incredibly minimal and elegant.

\(^a\)En anglais dans le texte itou.
Contribution

- Reynolds’s functional correspondence applies: normalization by evaluation.

- The syntactic correspondence applies too: a calculus of closures with a strategy.
And back to Pierre-Louis

- A cunning sense of computation.
And back to Pierre-Louis

- A cunning sense of computation.
- The ability to see potential in others and to nurture it.
And back to Pierre-Louis

- A cunning sense of computation.
- The ability to see potential in others and to nurture it.

Thank you.
Credits

The two pictures are from images.google.com

The work on full normalization is joint with Kevin Millikin and Johan Munk.