

# On the relation between call-by-value and call-by-name

Dylan McDermott   Alan Mycroft

# Goal

Suppose we have two semantics for a single language

- ▶ e.g. call-by-value and call-by-name

How does replacing one with the other affect the behaviour of programs?

## Goal

- ▶ Call-by-value:  $(\lambda x. e) e' \rightsquigarrow_v^* (\lambda x. e) v \rightsquigarrow_v e[x \mapsto v] \rightsquigarrow_v^* \dots$
- ▶ Call-by-name:  $(\lambda x. e) e' \rightsquigarrow_n e[x \mapsto e'] \rightsquigarrow_n^* \dots$

## Goal

- ▶ Call-by-value:  $(\lambda x. e) e' \rightsquigarrow_v^* (\lambda x. e) v \rightsquigarrow_v e[x \mapsto v] \rightsquigarrow_v^* \dots$
- ▶ Call-by-name:  $(\lambda x. e) e' \rightsquigarrow_n e[x \mapsto e'] \rightsquigarrow_n^* \dots$

If we replace call-by-value with call-by-name, then:

- ▶ No side-effects: nothing changes
- ▶ Only recursion: behaviour changes, but if CBV terminates with result  $v$ , CBN terminates with  $v$

## Goal

- ▶ Call-by-value:  $(\lambda x. e) e' \rightsquigarrow_v^* (\lambda x. e) v \rightsquigarrow_v e[x \mapsto v] \rightsquigarrow_v^* \dots$
- ▶ Call-by-name:  $(\lambda x. e) e' \rightsquigarrow_n e[x \mapsto e'] \rightsquigarrow_n^* \dots$

If we replace call-by-value with call-by-name, then:

- ▶ No side-effects: nothing changes
- ▶ Only recursion: behaviour changes, but if CBV terminates with result  $v$ , CBN terminates with  $v$
- ▶ Only nondeterminism: behaviour also different, but if CBV can terminate with result  $v$ , then CBN can also terminate with result  $v$
- ▶ Mutable state: behaviour changes, we can't say much about how

Questions:

- ▶ How can we prove these?
- ▶ What properties of the side-effects do we need to prove something?

# How to relate different semantics of the same language

1. Define another language that captures both semantics via two sound and adequate translations  $\llbracket - \rrbracket^v, \llbracket - \rrbracket^n$

$$\text{(CBV)} \quad \llbracket e \rrbracket^v \longleftarrow e \longrightarrow \llbracket e \rrbracket^n \quad \text{(CBN)}$$

5. For programs (closed, ground expressions)  $e$

$$\llbracket e \rrbracket^v \leq \llbracket e \rrbracket^n$$

# How to relate different semantics of the same language

1. Define another language that captures both semantics via two sound and adequate translations  $\llbracket - \rrbracket^v, \llbracket - \rrbracket^n$

$$(CBV) \quad \llbracket e \rrbracket^v \longleftarrow e \longrightarrow \llbracket e \rrbracket^n \quad (CBN)$$

ON THE RELATION BETWEEN DIRECT AND CONTINUATION SEMANTICS<sup>†</sup>

John C. Reynolds

Systems and Information Science

Syracuse University

5. For programs (closed, ground expressions)  $e$

$$\llbracket e \rrbracket^v \leq \llbracket e \rrbracket^n$$

# How to relate different semantics of the same language

1. Define another language that captures both semantics via two sound and adequate translations  $\llbracket - \rrbracket^v, \llbracket - \rrbracket^n$

$$(CBV) \quad \llbracket e \rrbracket^v \longleftarrow e \longrightarrow \llbracket e \rrbracket^n \quad (CBN)$$

2. Define maps between the two translations

$$\text{CBV translation of } \tau \xrightleftharpoons[\Psi_\tau]{\Phi_\tau} \text{CBN translation of } \tau$$

3. Show that  $\Phi, \Psi$  satisfy nice properties
4. Relate the two translations of (possibly open) expressions  $e$

$$\llbracket e \rrbracket^v \leq_{\text{ctx}} \Psi_\tau(\llbracket e \rrbracket^n[\Phi_\Gamma])$$

5. For programs (closed, ground expressions)  $e$

$$\llbracket e \rrbracket^v \leq \llbracket e \rrbracket^n$$



# How to relate different semantics of the same language

To relate CBV and CBN:

1. **Call-by-push-value** [Levy '99] captures CBV and CBN
2. We can define maps  $\Phi_\tau, \Psi_\tau$  using the syntax of CBPV
3.  $\Phi$  and  $\Psi$ :
  - ▶ behave nicely wrt the CBV and CBN translations, e.g.

$$\Phi_{\tau_1 \times \tau_2}(\langle e \rangle^v) = (\Phi_{\tau_1}(\langle \mathbf{fst} \, e \rangle^v), \Phi_{\tau_2}(\langle \mathbf{snd} \, e \rangle^v))$$

- ▶ form Galois connections  $\Phi_\tau \dashv \Psi_\tau$  (wrt  $\leq_{\text{ctx}}$ ) when side-effects are **thunkable**
4. (3) implies  $\langle e \rangle^v \leq_{\text{ctx}} \Psi_\tau(\langle e \rangle^n [\Phi_\Gamma])$
  5. (4) is  $\langle e \rangle^v \leq \langle e \rangle^n$  when  $e$  is a program

## Example

For recursion and nondeterminism, define

$$M_1 \leqslant M_2 \quad \Leftrightarrow \quad \forall V. M_1 \Downarrow \mathbf{return} V \Rightarrow M_2 \Downarrow \mathbf{return} V$$

( $\Downarrow$  is evaluation in CBPV)

so  $M_1 \leqslant_{\text{ctx}} M_2$  means

$$\forall V. C[M_1] \Downarrow \mathbf{return} V \Rightarrow C[M_2] \Downarrow \mathbf{return} V$$

for closed, ground contexts  $C$

Both side-effects are thunkable, so  $\Phi$  and  $\Phi$  form Galois connections, so

$$(\llbracket e \rrbracket^v)^{\text{v}} \leqslant_{\text{ctx}} \Psi_{\tau}((\llbracket e \rrbracket^{\text{n}})^{\text{n}}[\Phi_{\Gamma}])$$

## Example

For programs  $e$ , we have

$$\langle e \rangle^v \leq \langle e \rangle^n$$

so

$$\begin{aligned} e \rightsquigarrow_v^* v &\Leftrightarrow \langle e \rangle^v \Downarrow \mathbf{return} \langle v \rangle && \text{(soundness)} \\ &\Rightarrow \langle e \rangle^n \Downarrow \mathbf{return} \langle v \rangle && (\langle e \rangle^v \leq \langle e \rangle^n) \\ &\Leftrightarrow e \rightsquigarrow_n^* v && \text{(adequacy)} \end{aligned}$$

# Call-by-push-value [Levy '99]

Split syntax into **values** and **computations**

- ▶ Values don't reduce, computations do

# Call-by-push-value [Levy '99]

Split syntax into **values** and **computations**

- ▶ Values don't reduce, computations do

Evaluation order is **explicit**

- ▶ Sequencing of computations:

$$\frac{\Gamma \vdash V : A}{\Gamma \vdash \mathbf{return} V : \mathbf{F}A} \qquad \frac{\Gamma \vdash M_1 : \mathbf{F}A \quad \Gamma, x : A \vdash M_2 : \underline{C}}{\Gamma \vdash M_1 \mathbf{to} x. M_2 : \underline{C}}$$

- ▶ Thunks:

$$\frac{\Gamma \vdash M : \underline{C}}{\Gamma \vdash \mathbf{thunk} M : \mathbf{U}\underline{C}} \qquad \frac{\Gamma \vdash V : \mathbf{U}\underline{C}}{\Gamma \vdash \mathbf{force} V : \underline{C}}$$

# Call-by-value and call-by-name

Source language types:

$$\tau ::= 1 \mid 2 \mid \tau \rightarrow \tau'$$

CBV and CBN translations into CBPV:

$\tau \mapsto \text{value type } \llbracket \tau \rrbracket^v$	$\tau \mapsto \text{computation type } \llbracket \tau \rrbracket^n$
$1 \mapsto 1$	$1 \mapsto \mathbf{F} 1$
$2 \mapsto 2$	$2 \mapsto \mathbf{F} 2$
$(\tau \rightarrow \tau') \mapsto \mathbf{U}(\llbracket \tau \rrbracket^v \rightarrow \mathbf{F} \llbracket \tau' \rrbracket^v)$	$(\tau \rightarrow \tau') \mapsto ((\mathbf{U} \llbracket \tau \rrbracket^n) \rightarrow \llbracket \tau' \rrbracket^n)$
$\Gamma, x : \tau \mapsto \llbracket \Gamma \rrbracket^v, x : \llbracket \tau \rrbracket^v$	$\Gamma, x : \tau \mapsto \llbracket \Gamma \rrbracket^n, x : \mathbf{U} \llbracket \tau \rrbracket^n$
$\Gamma \vdash e : \tau \mapsto \llbracket \Gamma \rrbracket^v \vdash \llbracket e \rrbracket^v : \mathbf{F} \llbracket \tau \rrbracket^v$	$\Gamma \vdash e : \tau \mapsto \llbracket \Gamma \rrbracket^n \vdash \llbracket e \rrbracket^n : \llbracket \tau \rrbracket^n$

## Call-by-value and call-by-name

Define maps between CBV and CBN:

$$\Gamma \vdash M : \mathbf{F} \langle \tau \rangle^v \quad \mapsto \quad \Gamma \vdash \Phi_\tau M : \langle \tau \rangle^n \quad (\text{CBV to CBN})$$

$$\Gamma \vdash N : \langle \tau \rangle^n \quad \mapsto \quad \Gamma \vdash \Psi_\tau N : \mathbf{F} \langle \tau \rangle^v \quad (\text{CBN to CBV})$$

## Call-by-value and call-by-name

Define maps between CBV and CBN:

$$\Gamma \vdash M : \mathbf{F} \langle \tau \rangle^v \quad \mapsto \quad \Gamma \vdash \Phi_\tau M : \langle \tau \rangle^n \quad (\text{CBV to CBN})$$

$$\Gamma \vdash N : \langle \tau \rangle^n \quad \mapsto \quad \Gamma \vdash \Psi_\tau N : \mathbf{F} \langle \tau \rangle^v \quad (\text{CBN to CBV})$$

Example: for  $\tau = 1 \rightarrow 1$ , we have

$$\langle 1 \rightarrow 1 \rangle^v = \mathbf{U} (1 \rightarrow \mathbf{F} 1)$$

$$\langle 1 \rightarrow 1 \rangle^n = \mathbf{U} (\mathbf{F} 1) \rightarrow \mathbf{F} 1$$

$$M \quad \xrightarrow{\Phi_{1 \rightarrow 1}} \quad M \text{ to } f. \lambda x. \text{force } x \text{ to } z. z \text{ ' force } f$$

$$N \quad \xrightarrow{\Psi_{1 \rightarrow 1}} \quad \text{return (thunk } (\lambda x. (\text{thunk return } x) \text{ ' } N))$$



## Galois connection between CBV and CBN?

Since  $\Phi$  and  $\Psi$  behave nicely wrt translations, e.g.

$$\Phi_{\tau_1 \times \tau_2}(\llbracket e \rrbracket^v) = (\Phi_{\tau_1}(\llbracket \mathbf{fst} \, e \rrbracket^v), \Phi_{\tau_2}(\llbracket \mathbf{snd} \, e \rrbracket^v))$$

if  $(\Phi_\tau, \Psi_\tau)$  is a Galois connection (adjunction) for each  $\tau$ , i.e.

$$M \leq_{\text{ctx}} \Psi_\tau(\Phi_\tau M) \quad \Phi_\tau(\Psi_\tau N) \leq_{\text{ctx}} N$$

then

$$(\llbracket e \rrbracket^v) \leq_{\text{ctx}} \Psi_\tau((\llbracket e \rrbracket^n[\Phi_\Gamma]))$$

# Galois connection between CBV and CBN?

These do not always hold!

$$M \leqslant_{\text{ctx}} \Psi_{\tau}(\Phi_{\tau}M) \quad \Phi_{\tau}(\Psi_{\tau}N) \leqslant_{\text{ctx}} N$$

- ▶ **Don't** hold for: exceptions, mutable state

$$\text{raise} \not\leqslant_{\text{ctx}} \text{return} (\dots) = \Psi_{1 \rightarrow 1}(\Phi_{1 \rightarrow 1} \text{raise}) \\ (\diamond \vdash \text{raise} : F(U(1 \rightarrow F1)))$$

- ▶ **Do** hold for: no side-effects, recursion, nondeterminism

This is where the side-effects matter

# Galois connection between CBV and CBN?

## Definition (Thunkable [Führmann '99])

A computation  $\Gamma \vdash M : \mathbf{F} A$  is (lax) *thunkable* if

$$M \text{ to } x. \text{return} (\text{thunk} (\text{return } x)) \leq_{\text{ctx}} \text{return} (\text{thunk } M)$$

- ▶ Essentially: we're allowed to suspend the computation  $M$
- ▶ Implies  $M$  commutes with other computations, is (lax) discardable, (lax) copyable

# Galois connection between CBV and CBN?

## Definition (Thunkable [Führmann '99])

A computation  $\Gamma \vdash M : \mathbf{F} A$  is (lax) *thunkable* if

$$M \text{ to } x. \text{return}(\text{thunk}(\text{return } x)) \leqslant_{\text{ctx}} \text{return}(\text{thunk } M)$$

- ▶ Essentially: we're allowed to suspend the computation  $M$
- ▶ Implies  $M$  commutes with other computations, is (lax) discardable, (lax) copyable

## Lemma

*If every computation is thunkable, then  $(\Phi_\tau, \Psi_\tau)$  is a Galois connection.*

# How to relate call-by-value to call-by-name

If every computation is thunkable then

$$\langle e \rangle^v \leq_{\text{ctx}} \Psi_\tau(\langle e \rangle^n[\Phi_\Gamma])$$

for each  $e$ . (And the converse holds for computations of ground type.)

And if  $e$  is a program then

$$\langle e \rangle^v \leq \langle e \rangle^n$$

# Overview

How to relate two different semantics:

1. Translate from source language to intermediate language
2. Define maps between two translations
3. Relate terms:

$$\langle e \rangle^v \leq_{\text{ctx}} \Psi_\tau(\langle e \rangle^n[\Phi_\Gamma])$$

- ▶ Works for call-by-value and call-by-name
- ▶ Also works for other things like comparing direct and continuation-style semantics [Reynolds '74], strict and lazy products, etc.