Galois connecting call-by-value and call-by-name

Dylan McDermott Alan Mycroft

Goal

- Call-by-value: $(\lambda x.\, e)\, e' \rightsquigarrow_{\mathrm{v}}^{*} (\lambda x.\, e)\, v \rightsquigarrow_{\mathrm{v}} e[x \mapsto v] \rightsquigarrow_{\mathrm{v}}^{*} \cdots$
- Call-by-name: $(\lambda x.\, e)\, e' \rightsquigarrow_{\mathrm{n}} e[x \mapsto e'] \rightsquigarrow_{\mathrm{n}}^{*} \cdots$

Goal

- Call-by-value: $(\lambda x.\, e)\, e' \rightsquigarrow_{\mathrm{v}}{}^{*} (\lambda x.\, e)\, v \rightsquigarrow_{\mathrm{v}} e[x \mapsto v] \rightsquigarrow_{\mathrm{v}}{}^{*} \cdots$
- Call-by-name: $(\lambda x.\, e)\, e' \rightsquigarrow_{\mathrm{n}} e[x \mapsto e'] \rightsquigarrow_{\mathrm{n}}{}^{*} \cdots$

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

- No side-effects: nothing changes
- Only recursion: behaviour changes

$$\begin{array}{lll} \text{CBV:} & (\lambda x.\, \mathbf{false})\Omega \rightsquigarrow_{\mathrm{v}} (\lambda x.\, \mathbf{false})\Omega \rightsquigarrow_{\mathrm{v}} \cdots \\ \text{CBN:} & (\lambda x.\, \mathbf{false})\Omega \rightsquigarrow_{\mathrm{n}} \mathbf{false} \end{array}$$

but if CBV terminates with result v, CBN terminates with v

Goal

- Call-by-value: $(\lambda x.\, e)\, e' \rightsquigarrow_{\mathrm{v}}{}^{*} (\lambda x.\, e)\, v \rightsquigarrow_{\mathrm{v}} e[x \mapsto v] \rightsquigarrow_{\mathrm{v}}{}^{*} \cdots$
- Call-by-name: $(\lambda x.\, e)\, e' \rightsquigarrow_{\mathrm{n}} e[x \mapsto e'] \rightsquigarrow_{\mathrm{n}}{}^{*} \cdots$

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 $(\!|-|\!)^{\mathrm{v}}, (\!|-|\!)^{\mathrm{n}}$

$$\text{(CBV)} \qquad (\!|e|\!)^{\mathrm{v}} \;\longleftarrow\!\dashv\; e \;\longmapsto\; (\!|e|\!)^{\mathrm{n}} \qquad \text{(CBN)}$$

5. For programs (closed, ground expressions) e

$$(\!|e|\!)^{\mathrm{v}} \preccurlyeq (\!|e|\!)^{\mathrm{n}}$$

How to relate different semantics of the same language

1. Define another language that captures both semantics via two sound and adequate translations $(\!|-|\!)^{\mathrm{v}}, (\!|-|\!)^{\mathrm{n}}$

$$\text{(CBV)} \qquad (\!|e|\!)^{\mathrm{v}} \;\longleftarrow\!\!\dashv\; e \;\longmapsto\; (\!|e|\!)^{\mathrm{n}} \qquad \text{(CBN)}$$

2. Define maps between the two translations

$$\text{CBV translation of } \tau \underset{\Psi_\tau}{\overset{\Phi_\tau}{\rightleftarrows}} \text{CBN translation of } \tau$$

3. Show that Φ, Ψ satisfy nice properties
4. Relate the two translations of (possibly open) expressions e

$$(\!|e|\!)^{\mathrm{v}} \;\preccurlyeq_{\mathrm{ctx}}\; \Psi_\tau((\!|e|\!)^{\mathrm{n}}[\Phi_\Gamma])$$

5. For programs (closed, ground expressions) e

$$(\!|e|\!)^{\mathrm{v}} \;\preccurlyeq\; (\!|e|\!)^{\mathrm{n}}$$

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 Φ_τ, Ψ_τ using the syntax of CBPV
3. When side-effects are (lax) thunkable, these form Galois connections

$$\Phi_\tau \dashv \Psi_\tau$$

   (wrt $\preccurlyeq_{\mathrm{ctx}}$)
4. (3) implies $(\!|e|\!)^{\mathrm{v}} \preccurlyeq_{\mathrm{ctx}} \Psi_\tau((\!|e|\!)^{\mathrm{n}}[\Phi_\Gamma])$
5. (4) is $(\!|e|\!)^{\mathrm{v}} \preccurlyeq (\!|e|\!)^{\mathrm{n}}$ when e is a program

Example

For recursion and nondeterminism, define

$$M_1 \preccurlyeq M_2 \quad \Leftrightarrow \quad \forall V.\, M_1 \Downarrow \mathbf{return}\, V \;\Rightarrow\; M_2 \Downarrow \mathbf{return}\, V \qquad (\Downarrow \text{ is evaluation in CBPV})$$

so $M_1 \preccurlyeq_{\mathrm{ctx}} M_2$ 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 Φ and Ψ form Galois connections, so

$$(\!|e|\!)^{\mathrm{v}} \preccurlyeq_{\mathrm{ctx}} \Psi_\tau((\!|e|\!)^{\mathrm{n}}[\Phi_\Gamma])$$

Example

For programs e, we have

$$(\!|e|\!)^{\mathrm{v}} \preccurlyeq (\!|e|\!)^{\mathrm{n}}$$

so

$$\begin{aligned}
e \rightsquigarrow_{\mathrm{v}}^{*} v \quad &\Leftrightarrow \quad (\!|e|\!)^{\mathrm{v}} \Downarrow \textbf{return}\,(\!|v|\!) && \text{(soundness)} \\
&\Rightarrow \quad (\!|e|\!)^{\mathrm{n}} \Downarrow \textbf{return}\,(\!|v|\!) && ((\!|e|\!)^{\mathrm{v}} \preccurlyeq (\!|e|\!)^{\mathrm{n}}) \\
&\Leftrightarrow \quad e \rightsquigarrow_{\mathrm{n}}^{*} v && \text{(adequacy)}
\end{aligned}$$

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:

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

- Thunks:

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

Call-by-value and call-by-name

Source language types:

$$\tau ::= \mathbf{unit} \mid \mathbf{bool} \mid \tau \to \tau'$$

CBV and CBN translations into CBPV:

$$\begin{array}{rcl} \tau & \mapsto & \text{value type } (\!|\tau|\!)^{\mathrm{v}} \\ \mathbf{unit} & \mapsto & \mathbf{unit} \\ \mathbf{bool} & \mapsto & \mathbf{bool} \\ (\tau \to \tau') & \mapsto & \mathbf{U}((\!|\tau|\!)^{\mathrm{v}} \to \mathbf{F}\,(\!|\tau'|\!)^{\mathrm{v}}) \end{array}
\qquad
\begin{array}{rcl} \tau & \mapsto & \text{computation type } (\!|\tau|\!)^{\mathrm{n}} \\ \mathbf{unit} & \mapsto & \mathbf{F}\ \mathbf{unit} \\ \mathbf{bool} & \mapsto & \mathbf{F}\ \mathbf{bool} \\ (\tau \to \tau') & \mapsto & ((\mathbf{U}\,(\!|\tau|\!)^{\mathrm{n}}) \to (\!|\tau'|\!)^{\mathrm{n}}) \end{array}$$

$$\Gamma, x : \tau \ \mapsto\ (\!|\Gamma|\!)^{\mathrm{v}}, x : (\!|\tau|\!)^{\mathrm{v}} \qquad \Gamma, x : \tau \ \mapsto\ (\!|\Gamma|\!)^{\mathrm{n}}, x : \mathbf{U}\,(\!|\tau|\!)^{\mathrm{n}}$$

$$\Gamma \vdash e : \tau \ \mapsto\ (\!|\Gamma|\!)^{\mathrm{v}} \vdash_{\!c} (\!|e|\!)^{\mathrm{v}} : \mathbf{F}\,(\!|\tau|\!)^{\mathrm{v}} \qquad \Gamma \vdash e : \tau \ \mapsto\ (\!|\Gamma|\!)^{\mathrm{n}} \vdash_{\!c} (\!|e|\!)^{\mathrm{n}} : (\!|\tau|\!)^{\mathrm{n}}$$

Call-by-value and call-by-name

Define maps between CBV and CBN:

$$\Gamma \vdash M : \mathbf{F}\,(\!|\tau|\!)^{\mathrm{v}} \quad \mapsto \quad \Gamma \vdash \Phi_\tau M : \;\; (\!|\tau|\!)^{\mathrm{n}} \qquad \text{(CBV to CBN)}$$

$$\Gamma \vdash N : \;\; (\!|\tau|\!)^{\mathrm{n}} \quad \mapsto \quad \Gamma \vdash \Psi_\tau N : \mathbf{F}\,(\!|\tau|\!)^{\mathrm{v}} \qquad \text{(CBN to CBV)}$$

Call-by-value and call-by-name

Define maps between CBV and CBN:

$$\Gamma \vDash M : \mathbf{F}\,(\!|\tau|\!)^{\mathrm{v}} \quad \mapsto \quad \Gamma \vDash \Phi_\tau M : \ (\!|\tau|\!)^{\mathrm{n}} \qquad \text{(CBV to CBN)}$$

$$\Gamma \vDash N : \ (\!|\tau|\!)^{\mathrm{n}} \quad \mapsto \quad \Gamma \vDash \Psi_\tau N : \mathbf{F}\,(\!|\tau|\!)^{\mathrm{v}} \qquad \text{(CBN to CBV)}$$

Example: for $\tau = \mathbf{unit} \to \mathbf{unit}$, we have

$$(\!|\mathbf{unit} \to \mathbf{unit}|\!)^{\mathrm{v}} = \mathbf{U}\,(\mathbf{unit} \to \mathbf{F}\,\mathbf{unit})$$
$$(\!|\mathbf{unit} \to \mathbf{unit}|\!)^{\mathrm{n}} = \mathbf{U}\,(\mathbf{F}\,\mathbf{unit}) \to \mathbf{F}\,\mathbf{unit}$$

$$M \quad \overset{\Phi_{\mathbf{unit}\to\mathbf{unit}}}{\longmapsto} \quad M \ \mathbf{to}\ f.\,\lambda x.\,\mathbf{force}\,x\ \mathbf{to}\ z.\,z\,\text{`}\,\mathbf{force}\,f$$

$$N \quad \overset{\Psi_{\mathbf{unit}\to\mathbf{unit}}}{\longmapsto} \quad \mathbf{return}\,(\mathbf{thunk}\,(\lambda x.\,(\mathbf{thunk}\,\mathbf{return}\,x)\,\text{`}\,N))$$

Galois connection between CBV and CBN?

Lemma

If (Φ_τ, Ψ_τ) *is a Galois connection (adjunction) for each* τ*, i.e.*

$$M \;\preccurlyeq_{\mathrm{ctx}}\; \Psi_\tau(\Phi_\tau M) \qquad \Phi_\tau(\Psi_\tau N) \;\preccurlyeq_{\mathrm{ctx}}\; N$$

then

$$(\!| e |\!)^{\mathrm{v}} \;\preccurlyeq_{\mathrm{ctx}}\; \Psi_\tau((\!| e |\!)^{\mathrm{n}}[\Phi_\Gamma])$$

Galois connection between CBV and CBN?

These do not always hold!

$$M \preccurlyeq_{\text{ctx}} \Psi_\tau(\Phi_\tau M) \qquad \Phi_\tau(\Psi_\tau N) \preccurlyeq_{\text{ctx}} N$$

- Don't hold for: exceptions, mutable state

$$\textbf{raise} \not\preccurlyeq_{\text{ctx}} \textbf{return}\,(\ldots) = \Psi_{\textbf{unit}\to\textbf{unit}}(\Phi_{\textbf{unit}\to\textbf{unit}}\,\textbf{raise}) \qquad (\diamond \vdash \textbf{raise} : \textbf{F}\,(\textbf{U}\,(\textbf{unit} \to \textbf{F}\,\textbf{unit})))$$

- 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

$$M \;\mathbf{to}\; x.\, \mathbf{return}\,(\mathbf{thunk}\,(\mathbf{return}\,x)) \;\preccurlyeq_{\text{ctx}}\; \mathbf{return}\,(\mathbf{thunk}\,M)$$

- 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

$$M \ \mathbf{to}\ x.\,\mathbf{return}\,(\mathbf{thunk}\,(\mathbf{return}\,x)) \ \preccurlyeq_{\mathrm{ctx}}\ \mathbf{return}\,(\mathbf{thunk}\,M)$$

- 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 (Φ_τ, Ψ_τ) is a Galois connection.

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

If every computation is thunkable then

$$(\!|e|\!)^{\mathrm{v}} \;\; \preccurlyeq_{\mathrm{ctx}} \;\; \Psi_\tau((\!|e|\!)^{\mathrm{n}}[\Phi_\Gamma])$$

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

And if e is a program then

$$(\!|e|\!)^{\mathrm{v}} \preccurlyeq (\!|e|\!)^{\mathrm{n}}$$

Denotational semantics

Given an order-enriched model of CBPV

- cartesian closed **Poset**-category
- coproduct $1 + 1$
- strong **Poset**-monad T

prove that if

- T is lax idempotent ($T\eta_X \sqsubseteq \eta_{TX}$)

then

$$[\![(\!| e |\!)^{\mathrm{v}}]\!] \sqsubseteq \psi_\tau \circ [\![(\!| e |\!)^{\mathrm{n}}]\!] \circ \phi_\Gamma$$

For example:

	$[\![\Gamma]\!]$	T	$[\![M]\!]$	$[\![M]\!] \sqsubseteq [\![N]\!]$
No side-effects	set	Id	function	equality
Recursion	ωcpo	$(-)_\perp$	continuous function	pointwise
Nondeterminism	poset	free join-semilattice	monotone function	pointwise

Denotational semantics

Given an order-enriched model of CBPV

- cartesian closed **Poset**-category
- coproduct $1 + 1$
- strong **Poset**-monad T

prove that if

- T is lax idempotent $(T\eta_X \sqsubseteq \eta_{TX})$

then

$$[\![(\!| e |\!)^{\mathrm{v}}]\!] \ \sqsubseteq \ \psi_\tau \circ [\![(\!| e |\!)^{\mathrm{n}}]\!] \circ \phi_\Gamma$$

If the model is adequate:

$$[\![M_1]\!] \sqsubseteq [\![M_2]\!] \quad \Longrightarrow \quad M_1 \preccurlyeq_{\mathrm{ctx}} M_2$$

then

$$(\!| e |\!)^{\mathrm{v}} \ \preccurlyeq_{\mathrm{ctx}} \ \Psi_\tau((\!| e |\!)^{\mathrm{n}}[\Phi_\Gamma])$$

Overview

How to relate two different semantics:

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

$$(\!|e|\!)^{\mathrm{v}} \;\; \preccurlyeq_{\mathrm{ctx}} \;\; \Psi_\tau((\!|e|\!)^{\mathrm{n}}[\Phi_\Gamma])$$

- 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.