Linearly-used Continuations and Self-duality

Alex Simpson

LFCS, School of Informatics
University of Edinburgh, UK

Joint work with: Jeff Egger (LFCS, Edinburgh)
Rasmus Møgelberg (ITU, Copenhagen)
Computational effects

Computational effects are the non-functional aspects of computation, such as:

- nontermination, nondeterminism, probabilistic choice,
- exceptions, side effects, input/output, continuations,
- resumptions, …

Moggi (1989) introduced the type $TA$ of computations that produce values of type $A$.

A value $e : TA$ is invoked as a computation by executing it. On execution, $e$ may perform effects. On termination (if this occurs!), a value of type $A$ is produced.
(Generalised) computation types

Filinski (1996) and Levy (1999, 2004) consider $TA$ as a (basic) computation type amongst a richer class of (generalised) computation types.

If $B$, $C$ are computation types then so are:

- $A \to B$.
  A function $f : A \to B$ is **invoked** as a computation by applying it to a value $a : A$ and proceeding with the computation of $f(a)$.

- $B \times C$.
  A pair $(b, c) : B \times C$ is **invoked** as a computation either by selecting the computation $b$ and proceeding with it, or by selecting $c$ and proceeding with it.
Effect calculus (EMS, CSL 2009)

We write $A, B, C, \ldots$ for general value types.

Certain (value) types are distinguished as computation types. We write $A, B, C, \ldots$ for computation types.

$$A, B, \ldots ::= \alpha \mid \alpha \mid 1 \mid A \times B \mid A \to B \mid TA$$  \hfill (value types)

$$A, B, \ldots ::= \alpha \mid 1 \mid A \times B \mid A \to B \mid TA$$  \hfill (computation types)

Every computation type has an associated method for invoking its values as computations.
Cbv and cbn translations

The call-by-value (cbv) translation [Moggi] translates a simple type $\sigma$ to a value type $\sigma^{cbv}$.

The call-by-name (cbn) translation (cf. [Filinski, Levy]) translates $\sigma$ to a computation type $\sigma^{cbn}$.

$$
\alpha^{cbv} = \alpha \\
1^{cbv} = 1 \\
(\sigma \times \tau)^{cbv} = \sigma^{cbv} \times \tau^{cbv} \\
(\sigma \rightarrow \tau)^{cbv} = \sigma^{cbv} \rightarrow T\tau^{cbv} \\
\alpha^{cbn} = \alpha \\
1^{cbn} = 1 \\
(\sigma \times \tau)^{cbn} = \sigma^{cbn} \times \tau^{cbn} \\
(\sigma \rightarrow \tau)^{cbn} = \sigma^{cbn} \rightarrow \tau^{cbn} .
$$

Typing judgements $\Gamma \vdash t : \sigma$ get translated to:

$$
\Gamma^{cbv} \vdash t^{cbv} : T\sigma^{cbv} \\
\Gamma^{cbn} \vdash t^{cbn} : \sigma^{cbn}.
$$
Linearity in effect calculus

Basic dichotomy.

— In general, a value is a pervasive static object, it just is. Values can be copied and discarded. There is no natural notion of linear function between general value types.

— However, a value of computation type has a dynamic side: it can be invoked. There is the basis for a natural intuition of linearity: the computation is invoked exactly once. (Actually, we shall impose a slightly stricter requirement. See next slide!)

Our treatment of linearity is thus intimately tied up with the distinction between value types and computation types
Enriching effect calculus with $\searrow$

We add type constructor $A \searrow B$ for linear function space between computation types.

**Intuition:** A linear $f : A \searrow B$ must transform a value $a : A$ to a value $f[a] : B$ in such a way that:

- the invocation of $f[a]$ as a computation begins by invoking $a$,
- and this is the only time that $a$ is ever invoked.

It is helpful to think of $f[-]$ as a (generalised) evaluation context.
Function decomposition

The intuitive isomorphism

\[ A \to B \cong TA \to B \]

is derivable from the relevant rules

\[
\begin{align*}
\frac{\Gamma, x: A | \Delta \vdash t: B}{\Gamma | \Delta \vdash \lambda x: A. t: A \to B} & \quad \frac{\Gamma | \Delta \vdash s: A \to B \quad \Gamma | - \vdash t: A}{\Gamma | \Delta \vdash s(t): B} \\
\frac{\Gamma | - \vdash t: A}{\Gamma | - \vdash \lfloor t \rfloor: TA} & \quad \frac{\Gamma | \Delta \vdash t: TA \quad \Gamma, x: A | - \vdash u: B}{\Gamma | \Delta \vdash \text{let } x \leftarrow t \text{ in } u: B} \\
\frac{\Gamma | z: A \vdash t: B}{\Gamma | - \vdash \lambda z: A. t: A \to B} & \quad \frac{\Gamma | - \vdash s: A \to B \quad \Gamma | \Delta \vdash t: A}{\Gamma | \Delta \vdash s[t]: B}
\end{align*}
\]
Girard decomposition

By changing $TA$ to $!A$ we have that

$$A \to B \cong !A \to \circ B$$

is derivable from the rules:

\[
\begin{align*}
\Gamma, x : A \mid \Delta \vdash t : B & \quad \Gamma \mid \Delta \vdash s : A \to B \quad \Gamma \mid \vdash t : A \\
\Gamma \mid \Delta \vdash \lambda x : A. t : A \to B & \quad \Gamma \mid \Delta \vdash s(t) : B \\
\Gamma \mid \vdash t : A & \quad \Gamma \mid \Delta \vdash t : !A \quad \Gamma, x : A \mid \vdash u : B \\
\Gamma \mid \vdash !t : !A & \quad \Gamma \mid \Delta \vdash \text{let} !x \text{ be } t \text{ in } u : B \\
\Gamma \mid z : A \vdash t : B & \quad \Gamma \mid \vdash s : A \to \circ B \quad \Gamma \mid \Delta \vdash t : A \\
\Gamma \mid \vdash \lambda z : A. t : A \to \circ B & \quad \Gamma \mid \Delta \vdash s[t] : B
\end{align*}
\]
Enriched effect calculus (EMS, CSL 2009)

Extend effect calculus with selection of linear type constructors:

\[
A ::= \ldots | A \leftarrow B | \!A \otimes B | 0 | A \oplus B \\
A ::= \ldots | \!A \otimes B | 0 | A \oplus B .
\]

N.B., \( A \leftarrow B \) is not assumed to be a computation type itself. Thus:

\[
(A \rightarrow B) \rightarrow C \quad \text{value type} \\
(A \leftarrow B) \rightarrow C \quad \text{computation (hence value) type} \\
(A \leftarrow B) \leftarrow C \quad \text{not available} \\
A \leftarrow B \leftarrow C \quad \text{not available}
\]

N.B., \( \!A \otimes B \) is the application of the single primitive type-constructor \( \!(\_ \otimes \_ ) \) to \( A \) and \( B \)
Some isomorphisms

Isomorphisms that hold in the enriched effect calculus, cf. linear logic:

\[
\begin{align*}
A \rightarrow B & \cong !A \rightarrow B \\
(!A \otimes B) \rightarrow C & \cong A \rightarrow (B \rightarrow C) \\
& \cong B \rightarrow (A \rightarrow C) \\
!1 \otimes A & \cong A \\
!A \otimes !B & \cong !(A \times B) \\
!A \otimes 0 & \cong 0 \\
!A \otimes (B \oplus C) & \cong (!A \otimes B) \oplus (!A \otimes C)
\end{align*}
\]

(As value types)

(As computation types)
Semantics

A model of the enriched effect calculus is given by (EMS, CSL 2009):

— categories $\mathcal{V}$ (value types) and $\mathcal{C}$ (computation types)
— $\mathcal{V}$ is cartesian closed (models $1, A \times B, A \to B$)
— $\mathcal{C}$ is $\mathcal{V}$-enriched (models $A \multimap B$)
— $\mathcal{C}$ has $\mathcal{V}$-powers (models $A \to B$) and $\mathcal{V}$-copowers (models $!A \otimes B$)
— $\mathcal{C}$ has finite $\mathcal{V}$-enriched products (models $1, A \times B$) and coproducts (models $0, A \oplus B$)
— a $\mathcal{V}$-enriched adjunction $F \dashv U : \mathcal{C} \to \mathcal{V}$ (models $!A$)

We write the entire model as $F \dashv U : \mathcal{C} \to \mathcal{V}$.
(All structure other than the adjunction is determined by universal properties.)
Syntactic model

\[ \mathcal{V}: \text{Value types for objects} \]

\[ \mathcal{V}(A, B) = \{ t \mid x : A \vdash t : B \} / \text{equations} \]

\[ \mathcal{C}: \text{Computation types for objects} \]

\[ \mathcal{C}(A, B) = \{ t \mid x : A \vdash t : B \} / \text{equations} \]

\[ U(A) = A \quad U([t]_C) = [t]_\mathcal{V} \]
\[ F(A) = !A \quad F([t]_\mathcal{V}) = [\text{let } y \text{ be } x \text{ in } !t[y/x]]_C \]

Theorem (EMS, CSL 2009) The syntactic model is initial (up to coherent natural isomorphism) w.r.t. structure preserving functors.
Models of ILL

A linear/nonlinear model (Benton 1995) is given by symmetric monoidal closed category $\mathcal{C}$ (the linear category), a cartesian closed category $\mathcal{V}$ (the intuitionistic category) and a monoidal adjunction $F \dashv U : \mathcal{C} \to \mathcal{V}$. If $\mathcal{C}$ also has finite products and coproducts then the model is said to have additives.

Proposition (EMS, CSL 2009) If $F \dashv U : \mathcal{C} \to \mathcal{V}$ is a linear/nonlinear model with additives then it is a model of the enriched effect calculus.

N.B. the linear-logic syntax of EEC agrees with the interpretation of ILL in a linear/nonlinear model.

EEC is a fragment of ILL interpretable in a wider class of models.
Adjunction models of CBPV

Adjunction models (Levy 2005) are the natural models of call-by-push-value (CBPV)

Every strong monad on a cartesian-closed category $\mathcal{V}$, gives rise to an adjunction model (possibly in several non-equivalent ways).

Every model of EEC is an adjunction model of CBPV

Theorem (EMS, CSL 2009) Every adjunction model of CBPV fully embeds in a model of the enriched effect calculus.

EEC is a conservative extension of CBPV (hence Moggi’s computational metalanguage) with a gain in expressivity and essentially no loss in range of applicability
Set-based models

Let $\mathcal{C}$ be a locally small category with small products and coproducts.

Let $I$ be a chosen object of $\mathcal{C}$

Define $F \dashv U : \mathcal{C} \to \textbf{Set}$ by:

$$U(A) = \mathcal{C}(I, A)$$

$$F(X) = \coprod_X I$$

Proposition $F \dashv U : \mathcal{C} \to \textbf{Set}$ is a model of the enriched effect calculus.
Dual models

Given an EEC model $\mathcal{M} = (F \dashv U : \mathcal{C} \to \mathcal{V})$ and an object $R \in \mathcal{C}$, define the $R$-dual model:

$$\mathcal{M}^R := (\mathcal{R}^{(-)} \dashv \mathcal{C}^{(-), R} : \mathcal{C}^{\text{op}} \to \mathcal{V})$$

Lawvere calls the induced monad $\mathcal{C}(\mathcal{R}^{(-)}, R)$ on $\mathcal{V}$ the $R$-dual monad of $T = UF$. (Such a “dual” is determined from $T$ by assuming $U$ to be monadic.)

Our construction of a dual model is a duality in the following sense.

**Theorem** For any EEC model $\mathcal{M}$, there is an equivalence of EEC models between $\mathcal{M}$ and $\mathcal{M}^{R^{F1}}$.

In particular, there is an equivalence between $\mathcal{M}$ and $\mathcal{M}^{F1F1}$. 

---

Linearity Workshop, September 2009
Improved formulation: dual pointed models

The dual of a pointed model $\mathcal{M} = (F \dashv U : \mathcal{C} \to \mathcal{V}, R)$ is the pointed model

$$\mathcal{M}^\perp := (R^{(-)} \dashv \mathcal{C}(-, R) : \mathcal{C}^{\text{op}} \to \mathcal{V}, F1)$$

**Theorem**  For any pointed EEC model $\mathcal{M}$, there is an equivalence of pointed models between $\mathcal{M}$ and $\mathcal{M}^\perp\perp$.

(To avoid choosing an arbitrary point, the canonical pointing of a model $F \dashv U : \mathcal{C} \to \mathcal{V}$ is the pointed model $(F \dashv U : \mathcal{C} \to \mathcal{V}, F1)$.)
Dual ILL models

Let $\mathcal{M} = (F \dashv U : \mathcal{C} \to \mathcal{V}, R)$ be a pointed model where $F \dashv U : \mathcal{C} \to \mathcal{V}$ is a linear/nonlinear model with additives.

In general, the dual model $\mathcal{M}^\perp$ is not a linear/nonlinear model.

Unlike models of EEC, models of ILL are not closed under the dual model construction.

Proposition If $F \dashv U : \mathcal{C} \to \mathcal{V}$ is a classical linear/nonlinear model (i.e., $\mathcal{C}$ is $\ast$-autonomous) with additives and $R = \bot$ then $\mathcal{M}^\perp$ is equivalent to $\mathcal{M}$, and hence again a classical linear/nonlinear model.

The proposition describes a situation in which $\mathcal{M}$ is self dual. (N.B., this $\mathcal{M}$ is not canonically pointed.)
A set-based self-dual model

Let $\mathcal{C}$ be the free completion of the terminal (single morphism) category with small products and coproducts (Cf. Joyal’s free bicompletions; Cockett and Santocanale CSL 2009)

$\mathcal{C}$ has a distinguished object $*$ which is a fixed point of the self duality $(-)^*: \mathcal{C} \to \mathcal{C}^{\text{op}}$.

Exhibit $\mathcal{C}$ as an EEC model $\mathcal{M}$ over $\textbf{Set}$ by defining $I = *$ (hence $F1 \cong *$).

Proposition The model $\mathcal{M}$ is canonically self dual.
Syntactic self duality

Let $\mathcal{S}$ be the syntactic model for EEC

By initiality, there is an (essentially) unique map of models:

$$L : \mathcal{S} \rightarrow \mathcal{S}^\bot$$

satisfying:

$$L(\alpha) = \alpha$$

$$L(\overline{\alpha}) = \overline{\alpha}$$

**Theorem**  $L$ is an equivalence of canonically pointed models.

Thus the syntactic model $\mathcal{S}$ is canonically self dual.
Non-canonical self duality

Now let $S_{R}$ be the syntactic model for EEC with a distinguished computation type constant $R$ as point.

By initiality, there is an (essentially) unique map of models:

$$L: S_{R} \to S_{R}^\perp$$

satisfying:

$$L(\alpha) = \alpha$$
$$L(\alpha) = \alpha$$
$$L(R) = !1$$

**Theorem** $L$ is an equivalence of pointed models.

Thus the free model $S$ is also non-canonically self dual.
Syntactic version

Syntactically, \( L: S_R \rightarrow S_R^\perp \) amounts to a translation of EEC into itself.

A value type \( A \) translates to a value type \( A^{\nu R} \).

A computation type \( A \) translates to a computation type \( A^{\nu C R} \).

These translations satisfy \( A^{\nu R} \simeq A^{\nu C R} \rightarrow R \).

A typing judgement \( \Gamma \vdash t : A \) translates to
\[
\Gamma^{\nu R} \vdash t^{\nu R} : A^{\nu R}
\]

A typing judgement \( \Gamma \vdash z : A \vdash u : B \) translates to
\[
\Gamma^{\nu R} \vdash z : B^{\nu C R} \vdash u^{\nu C R} : A^{\nu C R}
\]
\[
\begin{align*}
\alpha^\nu_R &= \alpha \\
\alpha^\nu_R &= \alpha^c_R \to R \\
1^\nu_R &= 1 \\
(A \times B)^\nu_R &= A^\nu_R \times B^\nu_R \\
(A \to B)^\nu_R &= A^\nu_R \to B^\nu_R \\
(!A)^\nu_R &= (!A)^c_R \to R \\
(A \to B)^\nu_R &= B^{c_R} \to A^c_R \\
(!A \otimes B)^\nu_R &= (!A \otimes B)^{c_R} \to R \\
(0)^\nu_R &= (0)^{c_R} \to R \\
(A \oplus B)^\nu_R &= (A \oplus B)^{c_R} \to R \\
R^\nu_R &= R \\
\alpha^{c_R} &= \alpha \\
1^{c_R} &= 0 \\
(A \times B)^{c_R} &= A^{c_R} \oplus B^{c_R} \\
(A \to B)^{c_R} &= !A^\nu_R \otimes B^{c_R} \\
(!A)^{c_R} &= A^\nu_R \to R \\
(A \otimes B)^{c_R} &= A^\nu_R \to B^{c_R} \\
(0)^{c_R} &= 1 \\
(A \oplus B)^{c_R} &= A^{c_R} \times B^{c_R} \\
R^{c_R} &= !1
\end{align*}
\]
The self duality of the syntactic model manifests itself syntactically as the involutivity of the above translation.

**Theorem (Involution property)** We have isomorphisms

\[
A \rightarrow A^\nu R^\nu R \quad \quad A \rightarrow A^{CR CR}
\]

modulo which \( t = t^\nu R^\nu R \) and \( u = u^{CR CR} \).

Our proof of this is semantic, using self duality. (Obtaining a syntactic proof looks like a formidable exercise.)

**Corollary** The translations \((\cdot)^\nu R\) and \((\cdot)^{CR}\) are full and faithful.
Linearly-used continuations

Programs from $X$ to $Y$ with control operators are modelled as continuation transformers:

$$(Y \rightarrow R) \rightarrow (X \rightarrow R) \cong X \rightarrow (Y \rightarrow R) \rightarrow R$$

Cbv for the continuations monad $TY = (Y \rightarrow R) \rightarrow R$.

For many common structured forms of control the continuation transformers are linear [Berdine et al. 2002]:

$$(Y \rightarrow R) \Rightarrow (X \rightarrow R) \cong X \rightarrow (Y \rightarrow R) \Rightarrow R$$

Cbv for the linearly-used continuations monad $TY = (Y \rightarrow R) \Rightarrow R$. 
Linearly-used Continuations and Self-duality


Simple types $\sigma$ translate to ILL types $\sigma_{Rv}$ (cbv) and $\sigma_{Rn}$ (cbn):

\[
\begin{align*}
\alpha_{Rv} &= \alpha \\
1_{Rv} &= 1 \\
(\sigma \times \tau)_{Rv} &= \sigma_{Rv} \times \tau_{Rv} \\
(\sigma \to \tau)_{Rv} &= (\tau_{Rv} \to R) \to (\sigma_{Rv} \to R)
\end{align*}
\]

\[
\begin{align*}
\alpha_{Rn} &= \alpha \to \circ R \\
1_{Rn} &= 1 \\
(\sigma \times \tau)_{Rn} &= \sigma_{Rn} \times \tau_{Rn} \\
(\sigma \to \tau)_{Rn} &= \sigma_{Rn} \to \tau_{Rn}
\end{align*}
\]

Typing judgements $\Gamma \vdash t : \sigma$ translate to:

\[
\begin{align*}
\Gamma_{Rv} \vdash t_{Rv} : (\sigma_{Rv} \to R) \to R \\
\Gamma_{Rn} \vdash t_{Rn} : \sigma_{Rn}
\end{align*}
\]

This translation directly lands in EEC!
Linearly-used CPS translations from self-duality

Theorem (cbv) There is an isomorphism \( \sigma_{RV} \cong \sigma_{cbv}^{VR} \), modulo which we have \( t_{RV} = t_{cbv}^{VR} \).

Corollary cf. [Hasegawa 2002] The cbv linearly-used CPS translation \((\cdot)^{RV}\) gives a full and faithful translation from Moggi’s \(\lambda_v\)-calculus into EEC.

Theorem (cbn) There is an isomorphism \( \sigma_{RN} \cong \sigma_{cbn}^{VR} \), modulo which we have \( t_{RN} = t_{cbn}^{VR} \).

Corollary cf. [Hasegawa 2004] The cbn linearly-used CPS translation \((\cdot)^{RN}\) gives a full and faithful translation from simply-typed \(\lambda_{\beta\eta}\)-calculus into EEC.

Hasegawa has the corresponding results for ILL rather than EEC.
Summary

- The enriched effect calculus extends effect calculi (Moggi’s computational metalanguage, CBPV, the effect calculus) with some of the expressivity of intuitionistic linear logic; and also implements an intuitive notion of linearity.

- Models of EEC strictly generalise models of ILL.

- Unlike models of ILL, models of EEC are closed under an (interesting) dual model construction.

- Standard linearly-used CPS transforms and their properties fall out from the self-duality of the initial (syntactic) model.