Open Call-by-Value

Beniamino Accattoli[1] Giulio Guerrieri[2]

[1]INRIA, LIX, École Polytechnique

[2]I2M, Aix-Marseille Université & Dipartimento di Matematica e Fisica, Università Roma Tre

DICE 2016
Eindhoven, April 2, 2016

Outline

Outline

Plotkin's (strong) call-by-value λ-calculus (1975)

Motivation: Call-by-value (CBV) λ-calculus (Plotkin, 1975) is a version of λ-calculus closer to the real implementation of functional programming languages (e.g. Ocaml) and proof-assistants (e.g. Coq).

Syntax: same as ordinary (i.e. call-by-name) λ-calculus.

$$\begin{array}{lrcl} \text{(values)} & U, V & ::= & x \mid \lambda x\, M \\ \text{(terms)} & L, M, N & ::= & V \mid MN \end{array}$$

Operational semantics: (V is value!)

$$(\lambda x M) V \mapsto_{\beta_v} M\{V/x\}$$

$\to_{\beta_v}$ is the contextual closure of $\mapsto_{\beta_v}$ where contexts (with exactly one hole) are defined by (as usual)

$$\mathtt{C} ::= (\!|\cdot|\!) \mid \lambda x \mathtt{C} \mid \mathtt{C} M \mid M \mathtt{C}$$

Example: $(\lambda x\, y)(z I)$ (where $I = \lambda x\, x$) is a β_v-normal form (but a β-redex).

The perfect world: Closed CBV

The study of functional programming languages (e.g. Ocaml) is modeled by the framework of Plotkin's CBV λ-calculus calles Closed CBV, where

- evaluation is *weak*: it does not reduce under λ's;
- terms are *closed*.

Harmony property for Closed CBV

Closed normal forms are values (and values are normal forms).

The restriction to β_v-reduction instead of β-reduction impacts on the order in which redexes are evaluated, but *evaluation never gets stuck*:

- either every β-redex will eventually become a β_v-redex and be fired
- or evaluation diverges.

The "real" (imperfect) world: Strong and Open CBV

Strong CBV: full Plotkin's CBV λ-calculus

- reduction under λ's is allowed and terms can be open

Open CBV: intermediate setting between Closed CBV and Strong CBV

- evaluation is *weak*: it does not reduce under λ's;
- terms may be *open*.

Motivations to leave the perfect world:

- implementation model of Coq, see Grégoire & Leroy (2002)
- denotational semantics for CBV, see Paolini & Ronchi D. R. (1999, 2004)
- monad and CPS translations, see Moggi (1989), Sabry & Felleisen (1993)
- bisimulations and partial evaluation, see Lassen (2005), Jones *et al.* (1993)
- CBV and linear logic proof-nets, see Accattoli (2015)
- CBV and cost models, see Accattoli & Sacerdoti Coen (2015)
- ...

Here we study Open CBV.

Plotkin's (naïve) Open CBV (1975)

Naïve Open CBV: framework of Plotkin's CBV where β_v-reduction is weak and terms are possibly open.

terms	$t, u, s, r, ::= v \mid tu$
values	$v, w ::= x \mid \lambda x\, t$
evaluation contexts	$\mathtt{C} ::= (\!\vert \cdot \vert\!) \mid \mathtt{C}t \mid t\mathtt{C}$

Root-step	Contextual closure
$(\lambda x\, t)\, \lambda y u \mapsto_{\beta_\lambda} t\{\lambda y u / x\}$	$\mathtt{C}(\!\vert t \vert\!) \to_{\beta_\lambda} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\beta_\lambda} u$
$(\lambda x\, t) y \mapsto_{\beta_Z} t\{y/x\}$	$\mathtt{C}(\!\vert t \vert\!) \to_{\beta_Z} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\beta_Z} u$
$\mapsto_{\beta_v} := \mapsto_{\beta_\lambda} \cup \mapsto_{\beta_Z}$	$\mathtt{C}(\!\vert t \vert\!) \to_{\beta_v} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\beta_v} u$

Naïve Open CBV and its issues: stuck β-redexes

Issue: harmony does no longer hold ($\exists$ open β-normal forms that are not values)
$\rightsquigarrow$ there are stuck β-redexes, i.e. open β-redexes that never will be fired by the β_v-reduction because their argument is normal but not a value.
$\rightsquigarrow$ stuck β-redexes provide *prematures* β_v*-normal forms*, e.g.

$$t := (\lambda y\, \delta)(zz)\delta \qquad u := \delta((\lambda y\, \delta)(zz)) \qquad \text{where } \delta := \lambda x\, xx \tag{1}$$

while one would expect t and u to behave like the prototypical divergent term $\delta\delta$:

- for semantical reasons (empty semantics)
- for operational reasons (potential valuability)
- for logical reasons (translation into linear logic proof-nets)
- for syntactic reasons (hidden β_v-redex)

This issue impacts on termination and notions of observational equivalence.

$\rightsquigarrow$ Many proposals to extend β_v-reduction with some other rules or constructors (as explicit substitutions, i.e. `let...in` expressions): Moggi (1989), Sabry & Felleisen (1993), Sabry & Wadler (1997), Maraist *et al.* (1999), Curien & Herbelin (2000), Dychoff & Lengrand (2007), Herbelin & Zimmermann (2009), *etc.*

Naïve Open CBV and its issues: stuck β-redexes

Issue: harmony does no longer hold ($\exists$ open β-normal forms that are not values)
$\rightsquigarrow$ there are stuck β-redexes, i.e. open β-redexes that never will be fired by the β_v-reduction because their argument is normal but not a value.
$\rightsquigarrow$ stuck β-redexes provide *prematures* β_v*-normal forms*, e.g.

$$t := (\lambda y\, \delta)(zz)\delta \qquad u := \delta((\lambda y\, \delta)(zz)) \qquad \text{where } \delta := \lambda x\, xx \tag{1}$$

while one would expect t and u to behave like the prototypical divergent term $\delta\delta$:

- for semantical reasons (empty semantics)
- for operational reasons (potential valuability)
- for logical reasons (translation into linear logic proof-nets)
- for syntactic reasons (hidden β_v-redex)

This issue impacts on termination and notions of observational equivalence.

$\rightsquigarrow$ Many proposals to extend β_v-reduction with some other rules or constructors (as explicit substitutions, i.e. `let...in` expressions): Moggi (1989), Sabry & Felleisen (1993), Sabry & Wadler (1997), Maraist *et al.* (1999), Curien & Herbelin (2000), Dychoff & Lengrand (2007), Herbelin & Zimmermann (2009), *etc.*

Our contributions

1. Equivalence of the proposals: we show that the proposed generalizations of Naïve Open CBV are equivalent, *i.e.* they have exactly the same sets of normalizing and diverging λ-terms $\rightsquigarrow$ there is *just one notion of Open CBV*, independently of its specific syntactic incarnation.
   We also compare the equational theories of the different proposals, and indicate the finest one for Open CBV.
2. Cost models and an abstract machine: the termination results are complemented with quantitative analyses about the number of steps. Moreover we provide insights into the size-explosion problem for Closed/Open/Strong CBV, that lead to a new abstract machine for Open CBV, simpler than others in the literature.

Our contributions

1. Equivalence of the proposals: we show that the proposed generalizations of Naïve Open CBV are equivalent, *i.e.* they have exactly the same sets of normalizing and diverging λ-terms $\rightsquigarrow$ there is *just one notion of Open CBV*, independently of its specific syntactic incarnation.
   We also compare the equational theories of the different proposals, and indicate the finest one for Open CBV.
2. Cost models and an abstract machine: the termination results are complemented with quantitative analyses about the number of steps. Moreover we provide insights into the size-explosion problem for Closed/Open/Strong CBV, that lead to a new abstract machine for Open CBV, simpler than others in the literature.

Outline

The incarnations for Open CBV that we have studied

We focus on three proposals for Open CBV (other solutions, *e.g.* Moggi's or Herbelin & Zimmerman's, are already known to be equivalent to these ones).

1. The Fireball Calculus $\lambda_{\mathtt{fire}}$, that extends values to fireballs by adding so-called inert terms in order to restore harmony. It was introduced by Paolini & Ronchi Della Rocca (1999, 2004), then rediscovered independently first by Leroy & Grégoire (2002) to improve the implementation of Coq, and then by Accattoli & Sacerdoti Coen (2015) to study cost models;
2. The Value Substitution Calculus $\lambda_{\mathtt{vsub}}$, coming from the linear logic interpretation of CBV and using explicit substitutions (ES) and contextual rewriting rules to circumvent stuck β-redexes. It was introduced by Accattoli & Paolini (2012) and it is isomorphic to proof-nets for CBV λ-calculus;
3. The Shuffling Calculus $\lambda_{\mathtt{sh}}$, that has rules to shuffle constructors, similar to Regnier's σ-rules for CBN (1992, 1994), as an alternative to explicit substitutions. It was introduced by Carraro & Guerrieri (2014) to study the adequacy of Open/Strong CBV with respect to denotational semantics coming from linear logic.

Remark: for $\lambda_{\mathtt{fire}}$, $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{sh}}$, normalization and strong normalization coincide.

The incarnations for Open CBV that we have studied

We focus on three proposals for Open CBV (other solutions, *e.g.* Moggi's or Herbelin & Zimmerman's, are already known to be equivalent to these ones).

1. The Fireball Calculus $\lambda_{\texttt{fire}}$, that extends values to fireballs by adding so-called inert terms in order to restore harmony. It was introduced by Paolini & Ronchi Della Rocca (1999, 2004), then rediscovered independently first by Leroy & Grégoire (2002) to improve the implementation of Coq, and then by Accattoli & Sacerdoti Coen (2015) to study cost models;
2. The Value Substitution Calculus $\lambda_{\texttt{vsub}}$, coming from the linear logic interpretation of CBV and using explicit substitutions (ES) and contextual rewriting rules to circumvent stuck β-redexes. It was introduced by Accattoli & Paolini (2012) and it is isomorphic to proof-nets for CBV λ-calculus;
3. The Shuffling Calculus $\lambda_{\texttt{sh}}$, that has rules to shuffle constructors, similar to Regnier's σ-rules for CBN (1992, 1994), as an alternative to explicit substitutions. It was introduced by Carraro & Guerrieri (2014) to study the adequacy of Open/Strong CBV with respect to denotational semantics coming from linear logic.

Remark: for $\lambda_{\texttt{fire}}$, $\lambda_{\texttt{vsub}}$ and $\lambda_{\texttt{sh}}$, normalization and strong normalization coincide.

The incarnations for Open CBV that we have studied

We focus on three proposals for Open CBV (other solutions, *e.g.* Moggi's or Herbelin & Zimmerman's, are already known to be equivalent to these ones).

1. The Fireball Calculus $\lambda_{\texttt{fire}}$, that extends values to fireballs by adding so-called inert terms in order to restore harmony. It was introduced by Paolini & Ronchi Della Rocca (1999, 2004), then rediscovered independently first by Leroy & Grégoire (2002) to improve the implementation of Coq, and then by Accattoli & Sacerdoti Coen (2015) to study cost models;
2. The Value Substitution Calculus $\lambda_{\texttt{vsub}}$, coming from the linear logic interpretation of CBV and using explicit substitutions (ES) and contextual rewriting rules to circumvent stuck β-redexes. It was introduced by Accattoli & Paolini (2012) and it is isomorphic to proof-nets for CBV λ-calculus;
3. The Shuffling Calculus $\lambda_{\texttt{sh}}$, that has rules to shuffle constructors, similar to Regnier's σ-rules for CBN (1992, 1994), as an alternative to explicit substitutions. It was introduced by Carraro & Guerrieri (2014) to study the adequacy of Open/Strong CBV with respect to denotational semantics coming from linear logic.

Remark: for $\lambda_{\texttt{fire}}$, $\lambda_{\texttt{vsub}}$ and $\lambda_{\texttt{sh}}$, normalization and strong normalization coincide.

The incarnations for Open CBV that we have studied

We focus on three proposals for Open CBV (other solutions, *e.g.* Moggi's or Herbelin & Zimmerman's, are already known to be equivalent to these ones).

1. The Fireball Calculus $\lambda_{\texttt{fire}}$, that extends values to fireballs by adding so-called inert terms in order to restore harmony. It was introduced by Paolini & Ronchi Della Rocca (1999, 2004), then rediscovered independently first by Leroy & Grégoire (2002) to improve the implementation of Coq, and then by Accattoli & Sacerdoti Coen (2015) to study cost models;
2. The Value Substitution Calculus $\lambda_{\texttt{vsub}}$, coming from the linear logic interpretation of CBV and using explicit substitutions (ES) and contextual rewriting rules to circumvent stuck β-redexes. It was introduced by Accattoli & Paolini (2012) and it is isomorphic to proof-nets for CBV λ-calculus;
3. The Shuffling Calculus $\lambda_{\texttt{sh}}$, that has rules to shuffle constructors, similar to Regnier's σ-rules for CBN (1992, 1994), as an alternative to explicit substitutions. It was introduced by Carraro & Guerrieri (2014) to study the adequacy of Open/Strong CBV with respect to denotational semantics coming from linear logic.

Remark: for $\lambda_{\texttt{fire}}$, $\lambda_{\texttt{vsub}}$ and $\lambda_{\texttt{sh}}$, normalization and strong normalization coincide.

Open CBV 1: Fireball Calculus $\lambda_{\texttt{fire}}$

Idea: it extends the notion of value (actually, of variable).

terms (as in Plotkin)	$t, u, s, r, ::= v \mid tu$
values (as in Plotkin)	$v, w ::= x \mid \lambda x\, t$
fireballs	$f ::= i \mid \lambda x\, t$
inert terms	$i ::= x\, f_1 \dots f_n \quad n \geq 0$
evaluation contexts (as in Plotkin)	$\mathtt{C} ::= (\!\vert \cdot \vert\!) \mid \mathtt{C}t \mid t\mathtt{C}$

Root-step	Contextual closure
$(\lambda x\, t)\, \lambda y u \mapsto_{\beta_\lambda} t\{\lambda y u/x\}$	$\mathtt{C}(\!\vert t \vert\!) \to_{\beta_\lambda} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\beta_\lambda} u$
$(\lambda x\, t) i \mapsto_{\beta_i} t\{i/x\}$	$\mathtt{C}(\!\vert t \vert\!) \to_{\beta_i} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\beta_i} u$
$\mapsto_{\beta_f} := \mapsto_{\beta_\lambda} \cup \mapsto_{\beta_i}$	$\mathtt{C}(\!\vert t \vert\!) \to_{\beta_f} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\beta_f} u$

Example:

$$t := (\lambda y\, \delta)(zz)\delta \to_{\beta_i} \delta\delta \to_{\beta_\lambda} \delta\delta \to_{\beta_\lambda} \dots$$
$$u := \delta((\lambda y\, \delta)(zz)) \to_{\beta_i} \delta\delta \to_{\beta_\lambda} \delta\delta \to_{\beta_\lambda} \dots$$

Properties of the Fireball Calculus $\lambda_{\texttt{fire}}$

Using standard rewriting techniques, we have proved:

Proposition (Open harmony)

t is β_f-normal iff t is a fireball.

Proposition

1. $\rightarrow_{\beta_i}$ is strongly normalizing and strongly confluent.
2. $\rightarrow_{\beta_i}$ and $\rightarrow_{\beta_\lambda}$ commute.
3. $\rightarrow_{\beta_f}$ is strongly confluent and all normalizing β_f-reduction sequences d (if any) from t have the same length $|d|_{\beta_f}$, the same number $|d|_{\beta_i}$ of β_i-steps and the same number $|d|_{\beta_\lambda}$ of β_λ-steps.

Open CBV 2: Value Substitution Calculus $\lambda_{\mathtt{vsub}}$

Idea: It extends the syntax of terms with ES, adding rules acting at a distance.

terms	$t, u, s, r, ::= v \mid tu \mid t[u/x]$
values	$v, w ::= x \mid \lambda x\, t$
evaluation contexts	$\mathtt{C} ::= (\!\vert \cdot \vert\!) \mid \mathtt{C}t \mid t\mathtt{C} \mid \mathtt{C}[t/x] \mid t[\mathtt{C}/x]$
substitution contexts	$\mathtt{S} ::= (\!\vert \cdot \vert\!)[t_1/x_1] \dots [t_n/x_n] \quad n \geq 0$

Root-step	Contextual closure
$\mathtt{S}(\!\vert \lambda x\, t \vert\!)\, u \mapsto_{\mathsf{m}} \mathtt{S}(\!\vert t[u/x] \vert\!)$	$\mathtt{C}(\!\vert t \vert\!) \to_{\mathsf{m}} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\mathsf{m}} u$
$t[\mathtt{S}(\!\vert \lambda y\, u \vert\!)/x] \mapsto_{\mathsf{e}_\lambda} \mathtt{S}(\!\vert t\{\lambda y\, u/x\} \vert\!)$	$\mathtt{C}(\!\vert t \vert\!) \to_{\mathsf{e}_\lambda} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\mathsf{e}_\lambda} u$
$t[\mathtt{S}(\!\vert y \vert\!)/x] \mapsto_{\mathsf{e}_Z} \mathtt{S}(\!\vert t\{y/x\} \vert\!)$	$\mathtt{C}(\!\vert t \vert\!) \to_{\beta_i} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\mathsf{e}_Z} u$
$\mapsto_{\mathsf{e}} := \mapsto_{\mathsf{e}_\lambda} \cup \mapsto_{\mathsf{e}_Z}$	$\mathtt{C}(\!\vert t \vert\!) \to_{\mathsf{e}} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\mathsf{e}} u$
$\mapsto_{\mathtt{vsub}} := \mapsto_{\mathsf{e}} \cup \mapsto_{\mathsf{m}}$	$\mathtt{C}(\!\vert t \vert\!) \to_{\mathtt{vsub}} \mathtt{C}(\!\vert u \vert\!)$ iff $t \mapsto_{\mathtt{vsub}} u$

Example:

$$t := (\lambda y\, \delta)(zz)\delta \to_{\mathsf{m}} \delta[zz/y]\delta \to_{\mathsf{m}} (xx)[\delta/x][zz/y] \to_{\mathsf{e}_\lambda} (\delta\delta)[zz/y] \to_{\mathsf{m}} \dots$$

$$u := \delta((\lambda y\, \delta)(zz)) \to_{\mathsf{m}} \delta(\delta[zz/y]) \to_{\mathsf{m}} (xx)[\delta[zz/y]/x] \to_{\mathsf{e}_\lambda} (\delta\delta)[zz/y] \to_{\mathsf{m}} \dots$$

Properties of the Value Substitution Calculus $\lambda_{\mathtt{vsub}}$

Using standard rewriting techniques, we have proved:

Proposition

1. $\to_{\mathsf{m}}$ and $\to_{\mathsf{e}}$ are strongly normalizing.
2. $\to_{\mathsf{m}}$ and $\to_{\mathsf{e}}$ are strongly confluent.
3. $\to_{\mathsf{m}}$ and $\to_{\mathsf{e}}$ commute.
4. $\to_{\mathtt{vsub}}$ is strongly confluent and all normalizing $\mathtt{vsub}$-reduction sequences d (if any) from t have the same length $|d|_{\mathtt{vsub}}$, the same number $|d|_{\mathsf{m}}$ of m-steps and the same number $|d|_{\mathsf{e}}$ of e-steps.
5. Any $\mathtt{vsub}$-reduction sequence d from t is such that $|d|_{\mathsf{e}} \leq |d|_{\mathsf{m}}$.

Open CBV 3: Shuffling Calculus $\lambda_{\mathtt{sh}}$

Idea: It adds new CBV reduction rules.

terms (as in Plotkin) $\quad t, u, s, r, ::= v \mid tu$

values (as in Plotkin) $\quad v, w ::= x \mid \lambda x\, t$

evaluation contexts $\quad \mathtt{C} ::= (\!|\cdot|\!) \mid \mathtt{C}t \mid t\mathtt{C} \mid (\lambda x\, \mathtt{C})t$

Root-step

$$(\lambda x\, t)v \mapsto_{\beta_v} t\{v/x\}$$
$$(\lambda x\, t)sr \mapsto_{\sigma_{\mathtt{1}}} (\lambda x\, tr)s \quad x \notin \mathsf{fv}(r)$$
$$v((\lambda x\, r)s) \mapsto_{\sigma_{\mathtt{3}}} (\lambda x\, vr)s \quad x \notin \mathsf{fv}(v)$$
$$\mapsto_{\sigma} := \mapsto_{\sigma_{\mathtt{1}}} \cup \mapsto_{\sigma_{\mathtt{3}}}$$
$$\mapsto_{\mathtt{sh}} := \mapsto_{\beta_v} \cup \mapsto_{\sigma}$$

Contextual closure

$$\mathtt{C}(\!|t|\!) \to_{\beta_v^{\flat}} \mathtt{C}(\!|u|\!) \text{ iff } t \mapsto_{\beta_v} u$$
$$\mathtt{C}(\!|t|\!) \to_{\sigma_{\mathtt{1}}^{\flat}} \mathtt{C}(\!|u|\!) \text{ iff } t \mapsto_{\sigma_{\mathtt{1}}} u$$
$$\mathtt{C}(\!|t|\!) \to_{\sigma_{\mathtt{3}}^{\flat}} \mathtt{C}(\!|u|\!) \text{ iff } t \mapsto_{\sigma_{\mathtt{3}}} u$$
$$\mathtt{C}(\!|t|\!) \to_{\sigma^{\flat}} \mathtt{C}(\!|u|\!) \text{ iff } t \mapsto_{\sigma} u$$
$$\mathtt{C}(\!|t|\!) \to_{\mathtt{sh}} \mathtt{C}(\!|u|\!) \text{ iff } t \mapsto_{\mathtt{sh}} u$$

Example:

$$t := (\lambda y\, \delta)(zz)\delta \to_{\sigma_{\mathtt{1}}} (\lambda y\, \delta\delta)(zz) \to_{\beta_v} (\lambda y\, \delta\delta)(zz) \to_{\beta_v} \dots$$
$$u := \delta((\lambda y\, \delta)(zz)) \to_{\sigma_{\mathtt{3}}} (\lambda y\, \delta\delta)(zz) \to_{\beta_v} (\lambda y\, \delta\delta)(zz) \to_{\beta_v} \dots$$

Properties of the Shuffling Calculus $\lambda_{\mathtt{sh}}$

Using standard rewriting techniques, we have proved:

Proposition

1. $\to_{\sigma^\flat}$ is strongly normalizing and (not strongly) confluent.
2. $\to_{\mathtt{sh}}$ is (not strongly) confluent.

In contrast to $\lambda_{\mathtt{fire}}$ and $\lambda_{\mathtt{vsub}}$, $\lambda_{\mathtt{sh}}$ is not strongly confluent and not all normalizing $\mathtt{sh}$-reduction sequences (if any) from a given term have the same length.
Example: Consider all normalizing $\mathtt{sh}$-reduction sequences from $(\lambda y\, z)(\delta(zz))$

Fireball Calculus vs. Value Substitution Calculus

The *unfolding* of a term t with ES is the term $t\downarrow$ obtained from t by turning ES into meta-level substitutions; it is defined by:

$$x\downarrow := x \qquad (tu)\downarrow := t\downarrow\, u\downarrow \qquad (\lambda x t)\downarrow := \lambda x\, t\downarrow \qquad (t[u/x])\downarrow := t\downarrow\{u\downarrow/x\}$$

Theorem (Quantitative simulation of $\lambda_{\texttt{fire}}$ into $\lambda_{\texttt{vsub}}$)

For any β_f-reduction sequence $d\colon t \to^*_{\beta_f} u$ there are a term r with ES and a $\texttt{vsub}$-reduction sequence $e\colon t \to^*_{\texttt{vsub}} r$ such that:

1. (qualitative relationship) $r\downarrow = u$;
2. (quantitative relationship) $|d|_{\beta_f} = |e|_{\mathsf{m}}$ and $|d|_{\beta_\lambda} = |e|_{\mathsf{e}_\lambda} = |e|_{\mathsf{e}}$;
3. (normal forms) if u is β_f-normal then there are a $\texttt{vsub}$-normal term s with ES and a $\texttt{vsub}$-reduction sequence $e'\colon r \to_{\mathsf{e}_Z} s$ such that $|e'|_{\mathsf{e}_Z} \leq |e|_{\mathsf{m}} - |e|_{\mathsf{e}_\lambda}$.

Corollary (Linear termination equivalence of $\lambda_{\texttt{vsub}}$ and $\lambda_{\texttt{fire}}$)

There exists a normalizing β_f-reduction sequence d from a term t iff there exists a normalizing $\texttt{vsub}$-reduction sequence e from t. Moreover, $|d|_{\beta_f} \leq |e|_{\texttt{vsub}} \leq 2|d|_{\beta_f}$, i.e. they are related linearly.

Shuffling Calculus vs. Value Substitution Calculus

Theorem (Quantitative simulation of $\lambda_{\mathtt{sh}}$ into $\lambda_{\mathtt{vsub}}$)

For any $\mathtt{sh}$-reduction sequence $d\colon t \to^*_{\mathtt{sh}} u$ there are a term s with ES and a $\mathtt{vsub}$-reduction sequence $e\colon t \to^*_{\mathtt{vsub}} s$ such that:

1. (qualitative relationship) $s \equiv \mathrm{m}(u)$, the m-normal form of u;
2. (quantitative relationship) $|d|_{\beta^\flat_v} = |e|_{\mathtt{e}}$;
3. (normal forms) if u is $\mathtt{sh}$-normal then s and $\mathrm{m}(u)$ are $\mathtt{vsub}$-normal.

Corollary (Termination equivalence of $\lambda_{\mathtt{vsub}}$ and $\lambda_{\mathtt{sh}}$)

There exists a normalizing $\mathtt{sh}$-reduction sequence d from a term t iff there exists a normalizing $\mathtt{vsub}$-reduction sequence e from t. Moreover, $|d|_{\beta^\flat_v} = |e|_{\mathtt{e}}$.

Corollary (Number of $\beta^\flat_v$-steps in invariant in $\lambda_{\mathtt{sh}}$)

All normalizing $\mathtt{sh}$-reduction sequences (if any) from term t have the same number of $\beta^\flat_v$-steps.

What about equational theories?

Given a calculus with a reduction $\to_{\mathsf{r}}$, its equational theory $\simeq_{\mathsf{r}}$ is the reflexive-transitive and symmetric closure of $\to_{\mathsf{r}}$.

Theorem

The Fireball Calculus and the Shuffling Calculus have the same equational theory, modulo some commutation rules: $t \simeq^{\mathtt{ext}}_{\mathtt{sh}} s$ iff $t \simeq^{\equiv}_{\mathtt{vsub}} s$.

On the contrary, the Fireball Calculus equates too much! Let:

$$I := \lambda x\, x \qquad t := (\lambda y\, I)(xx) \qquad s := (\lambda y\, I)(xz)$$

then $t \simeq_{\beta_f} s$ since $t \to_{\beta_f} I \,{}_{\beta_f}\!\leftarrow s$.
But consider the context $\mathtt{C} := (\lambda z(\lambda x(\!|\,|\!))\delta)I$: then, $\mathtt{C}(\!|t|\!) \not\simeq_{\beta_f} \mathtt{C}(\!|s|\!)$ since

$$\mathtt{C}(\!|t|\!) \to^*_{\beta_f} (\lambda y\, I)(\delta\delta) \to^*_{\beta_f} (\lambda y\, I)(\delta\delta) \to^*_{\beta_f} \dots \qquad \mathtt{C}(\!|u|\!) \to^*_{\beta_f} I$$

What about equational theories?

Given a calculus with a reduction $\to_{\mathsf{r}}$, its equational theory $\simeq_{\mathsf{r}}$ is the reflexive-transitive and symmetric closure of $\to_{\mathsf{r}}$.

Theorem

The Fireball Calculus and the Shuffling Calculus have the same equational theory, modulo some commutation rules: $t \simeq^{\mathtt{ext}}_{\mathtt{sh}} s$ iff $t \simeq^{\equiv}_{\mathtt{vsub}} s$.

On the contrary, the Fireball Calculus equates too much! Let:

$$I := \lambda x\, x \qquad t := (\lambda y\, I)(xx) \qquad s := (\lambda y\, I)(xz)$$

then $t \simeq_{\beta_f} s$ since $t \to_{\beta_f} I \,{}_{\beta_f}\!\leftarrow s$.
But consider the context $\mathtt{C} := (\lambda z(\lambda x(\!|\,|\!)\delta)I$: then, $\mathtt{C}(\!|t|\!) \not\simeq_{\beta_f} \mathtt{C}(\!|s|\!)$ since

$$\mathtt{C}(\!|t|\!) \to^*_{\beta_f} (\lambda y\, I)(\delta\delta) \to^*_{\beta_f} (\lambda y\, I)(\delta\delta) \to^*_{\beta_f} \dots \qquad \mathtt{C}(\!|u|\!) \to^*_{\beta_f} I$$

Outline

On Cost Models and All That

Aim: time complexity analyses of the λ-calculus. Is there a way to measure the computational complexity of a λ-term?

Natural Cost Model: the number of β-steps to reach the normal form.

Problem: is it a reasonable cost model?

Weak invariance thesis (Slot and van Emde Boas, 1984)

Reasonable (or *invariant*) computational models can simulate each other within a polynomial overhead in time, *e.g.* Random Access Machines and Turing Machines.

The difficult part (because of *size-explosion*):
to show an implementation of λ-calculus on RAM with a polynomial overhead
(wrt the # of β-steps and the size of the initial term).

The Size-Explosion problem

At first sight, the number of β-steps does not seem to be a reasonable cost model.

Size-Exploding Families:

a family of terms $(t_n)_{n\in\mathbb{N}}$ of linear size
evaluating in a linear number of steps
to a result (*i.e.* normal form) of exponential size.

The size of the result is exponential in the cost model: is it reasonable?

Problem independent of the evaluation strategy
(Closed/Open/Strong λ-calculus in CBN/CBV/CBNeed, as well as optimal).

Solution:
Switch to compact representations of results (sharing) and micro-step evaluation.

Different evaluation strategies $\rightsquigarrow$ Different families and different solutions.

Size-Explosion and Evaluation Strategies (*e.g.* for $\lambda_{\texttt{fire}}$)

Example affecting all evaluation strategies (Closed/Open/Strong):

$$t_1 := \lambda x\, \lambda y\, yxx \qquad r_0 := \lambda x\, x$$
$$t_{n+1} := \lambda x\, t_n(\lambda y\, yxx) \qquad r_{n+1} := \lambda y\, y r_n r_n$$

Abstraction Size-Explosion (note $\to_{\beta_\lambda}$):
$t_n r_0 \to^n_{\beta_\lambda} r_n$ where $|t_n| = O(n)$, $|r_n| = O(2^n)$ and r_n is β_f-normal, $\forall n > 0$.

Example affecting Open and Strong strategies (not the perfect Closed world):

$$u_1 := \lambda x_1\, x_1 x_1 \qquad u_{n+1} := \lambda x_{n+1}\, u_n(x_{n+1} x_{n+1})$$

Inert size-explosion for $\lambda_{\texttt{fire}}$ (note $\to_{\beta_i}$):
$u_n(xx) \to^n_{\beta_i} (xx)^{2^n}$ where $|u_n| = O(n)$ and $|(xx)^{2^n}| = O(2^n)$ is β_f-normal, $\forall n > 0$.

Finally, Strong strategies have additional issues with abstraction size-explosion.

The Literature

Theorem for Closed CBN & CBV λ-calculus

The # of steps is a reasonable cost model.
Overhead: *linear* in the number of steps.

(Blelloch & Greiner '95, Sands, Gustavsson & Moran '02, Dal Lago & Martini '08).

Theorem (Accattoli & Dal Lago, 2014) for Strong CBN λ-calculus

The # number of leftmost-outermost steps is a reasonable cost model.
Overhead: *quadratic* in the number of steps, *linear* in the size of the initial term.

Theorem (Accattoli & Sacerdoti Coen, 2015) for the Fireball Calculus

The # of steps in $\lambda_{\mathtt{fire}}$ is a reasonable cost model.
Overhead: *linear* in the number of steps, *linear* in the size of the initial term.

Corollary (Accattoli & G., 2016) for the Value Substitution Calculus

The # of steps in $\lambda_{\mathtt{vsub}}$ is a reasonable cost model.
Overhead: *bilinear*, as for $\lambda_{\mathtt{fire}}$.

GLAMOUR Machine

Accattoli & Sacerdoti Coen's theorem (2015) rely on an abstract machine:

$\lambda_{\texttt{fire}}$ $\overset{\text{bilinear overhead}}{\rightsquigarrow}$ Glamour (CBV abstract machine) $\overset{\text{bilinear overhead}}{\rightsquigarrow}$ RAM

Features of the Glamour machine:

1. Never substitutes inert terms,
   variables included, *e.g.* $x[xx/x]$ and $x[z/x]$ are normal;
2. Substitutes abstractions on-demand,
   i.e. abstractions are substituted only when they create a β-redex.

The Glamour machine is complex, with labeled environments.

Easy GLAMOUR

Features of the Glamour machine:

1. Never substitutes inert terms,
   variables included, *e.g.* $x[xx/x]$ and $x[z/x]$ are normal;
2. Substitutes abstractions on-demand,
   i.e. abstractions are substituted only when they create a new β-redex.

We introduce Easy Glamour: CBV abstract machine implementing only feature 1.
(thus it substitutes abstractions freely, *i.e.* always)

It does not need labels, it is reasonable, simpler, and slightly slower.

Theorem (Accattoli & G., 2016) for Fireball Calculus

The Easy Glamour is a reasonable implementation of the Fireball calculus.
Overhead: linear in the number of steps, quadratic in the size of the initial term.

$$\lambda_{\texttt{fire}} \overset{\text{linear/quadratic overhead}}{\rightsquigarrow} \text{Easy Glamour} \overset{\text{bilinear overhead}}{\rightsquigarrow} \text{RAM}$$

Easy GLAMOUR

Features of the Glamour machine:

1. Never substitutes inert terms,
   variables included, *e.g.* $x[xx/x]$ and $x[z/x]$ are normal;
2. Substitutes abstractions on-demand,
   i.e. abstractions are substituted only when they create a new β-redex.

We introduce Easy Glamour: CBV abstract machine implementing only feature 1.
(thus it substitutes abstractions freely, *i.e.* always)

It does not need labels, it is reasonable, simpler, and slightly slower.

Theorem (Accattoli & G., 2016) for Fireball Calculus

The Easy Glamour is a reasonable implementation of the Fireball calculus.
Overhead: linear in the number of steps, quadratic in the size of the initial term.

$$\lambda_{\texttt{fire}} \overset{\text{linear/quadratic overhead}}{\rightsquigarrow} \text{Easy Glamour} \overset{\text{bilinear overhead}}{\rightsquigarrow} \text{RAM}$$

Modularity of Glamour abstract machines

Moral: To have a reasonable implementation of the Fireball calculus, it is enough to never substitute compound inert terms.

We show *two modular degrees of freedom* for reasonable implementations:

1. *To Substitute Variables or Not*:
   the overhead is quadratic or linear in the number of steps;
2. *To Substitute Abstractions Freely or On-Demand*:
   the overhead is quadratic or linear in the size of the initial term;.

Let $t \to^{k}_{\beta_f} u$ be an evaluation sequence in the Fireball Calculus.
Glamour machines have the following overhead:

	never substituting variables	substituting variables
substituting abstractions on-demand	$O(k \cdot \|t\|)$	$O(k^2 \cdot \|t\|)$
substituting abstractions freely	$O(k \cdot \|t\|^2)$	$O(k^2 \cdot \|t\|^2)$

Open vs Strong CBV

Strong strategies have additional issues with abstraction size-explosion.

Qualitatively: Iterated Open CBV = Strong CBV.

Insight:
Easy Glamour is reasonable for Open CBV,
Iterated Easy Glamour is not reasonable for Strong CBV.

Quantitatively:

$$\text{Iterated reasonable implementation for Open CBV} \neq \text{reasonable implemementation for Strong CBV.}$$

Moral: substituting abstractions on-demand is necessary for Strong settings.

Consequence:
Reasonable implem. of Strong CBV have only one degree of freedom (variables).

Outline

The Cost of Inert Steps

The number of fireball steps is an invariant cost model.

Roughly, a fireball step costs 1.

Fireball steps = abstraction steps + inert steps.

Thus, a inert step costs 1 wrt the current reasonable cost model.

Are Inert Steps Relevant?

(Easy) Glamour: inert steps implemented in constant time, without substitution.

Suspicion: maybe they are dominated by the number of abstraction steps.

Natural Question: can inert steps be seen as administrative work?

Roughly: do inert steps cost 1 or rather cost 0?

Answer:
inert step seem to cost 1, *i.e.* they are relevant for complexity analyses.

Inert Steps Seem to Be Relevant

We show a family of terms s_n such that:

- when applied to a inert term i, for instance $i := yy$
- Linear Abstraction Prefix: $s_n i$ evaluates in a *linear* number of β_λ-steps,
- Exponential Inert Coda: followed by an *exponential* number of β_i-steps.

$\rightsquigarrow$ It seems the number of β_λ-steps is not an invariant cost model for Open CBV.

Our result: $s_n i \to_{\beta_v}^{n} u_n \to_{\beta_i}^{2^n-1} r_n$.

Where

$$t_0 := x \qquad u_0 := t_0 i \qquad r_0 := u_0 \qquad s_0 := x$$
$$t_{n+1} := \lambda z\, y u_n u_n \qquad u_{n+1} := t_{n+1} i \qquad r_{n+1} := y r_n r_n \qquad s_n := (\lambda x \lambda z\, y(xi)(xi)) s_n$$

Note that r_n is a inert term, hence β_f-normal, whereas u_n is β_v-normal.