Open Call-By-Push-Value

M2 Internship

Xavier MONTILLET

under the supervision of

Guillaume MUNCH-MACCAGNONI

22nd June 2017

Outline

Goal

A system that:

1. Expresses computations of higher-order functions and sums
2. Is compatible with the presence of effects
3. Subsumes Call-By-Value and Call-By-Name
4. Has a nice rewriting theory

Outline:

- Background
  - λ-calculus, evaluation strategies (CBN, CBV)
  - Abstract machines
  - Call-By-Push-Value (CBPV): 1, 2 and 3
  - Intuitionistic sequent calculus with evaluation order $\left(\mathbf{LJ}^{\eta}_{p}\right)$: 4
- Contributions
  - Simulation of CBPV by $\mathbf{LJ}^{\eta}_{p}$
  - Weak reduction in $\mathbf{LJ}^{\eta}_{p}$

λ-calculus

- Represents computations with higher-order functions
  - Variable: x
  - Abstraction: $\lambda x.t$
  - Application: $f\,a$
- Head reduction: $(\lambda x.t)u \rhd_\beta t[u/x]$
- Weak reduction:
  - Why:
    - Head reduction gets stuck when it should not: $((\lambda x.x)u_1)\ u_2$
    - Strong reduction breaks termination
  - How: Reduce everywhere except under abstractions
  - Not well-behaved: normal forms ($\approx$ results) are not unique

λ-calculus
Call-By-Name

- Call-By-Name (CBN):
  - No reduction under abstractions and of the argument of an application
  - $\approx$ Lazy (Haskell)
- Krivine machine: Moves focus explicitly
  - Term represented by the subterm which is focused and a context containing the rest: $\langle t \parallel k \rangle \approx k\left[\boxed{t}\right]$
  - Transitions:
    - On an application, move the focus to the function:

$$k\left[\boxed{f\,a}\right] \blacktriangleright k\left[\boxed{f}\,a\right]$$

    - On an abstraction, apply it to the argument given by the context:

$$k\left[\boxed{\lambda x.t}\,u\right] \blacktriangleright k\left[\boxed{t[u/x]}\right]$$

λ-calculus
Call-By-Value

- Values:
  - Variable: x
  - Abstraction: $\lambda x.t$
- Call-By-Value (CBV):
  - Restricted head reduction: Only reduce when the argument is a value: $(\lambda x.t)v \triangleright_\beta t[v/x]$
  - No reduction under abstractions
  - $\approx$ Eager (OCaml)
- Two deterministic subreductions:
  - Left-to-right (reduce function first)
  - Right-to-left (reduce argument first)
- Abstract machine: Moves focus explicitely

λ-calculus

Open Call-By-Value

- Inert terms are normal but not values: Variable applied to some arguments: $i ::= x t_1 \ldots t_n$
- Stuck redexes: Function applied to inert term: $(\lambda x.t)i$
- Unexpected behaviours: A function that ignores its argument and returns a diverging term, applied to an inert term: $(\lambda x.\Omega)i$
  - Is a normal form
  - Diverges no matter what closed terms are substituted for variables
- Solutions are known
  - Some involve commutation rules such as:

$$\pi_1 \left(\begin{array}{l} \text{pm } t \text{ as} \\ \left| \begin{array}{l} \iota_1\,(x) \mapsto u_1 \\ \iota_2\,(x) \mapsto u_2 \end{array} \right. \end{array} \right) \rightsquigarrow \left(\begin{array}{l} \text{pm } t \text{ as} \\ \left| \begin{array}{l} \iota_1\,(x) \mapsto \pi_1 u_1 \\ \iota_2\,(x) \mapsto \pi_1 u_2 \end{array} \right. \end{array} \right)$$

λ-calculus
Effects

- η-conversion for functions: $f \approx_\eta \lambda x.fx$ whenever f is of functional type
- In the presence of effects:
  - Satisfied in CBN
  - Not satisfied in CBV: η-conversion allows to replace an arbitrary function by a value

λ-calculus
Effects

- η-conversion for booleans:
  $t[u/x] \approx_\eta$ if u then $t[\text{true}/x]$ else $t[\text{false}/x]$ whenever u is of boolean type
  - "A boolean can be evaluated at any given time"
- In the presence of effects:
  - Satisfied neither in CBV nor in CBN
- Restricted to values: $t[v/x] \approx_\eta$ if v then $t[\text{true}/x]$ else $t[\text{false}/x]$
  - Satisfied in CBV
  - Not satisfied in CBN: We recover the unrestricted η-conversion using the restricted version with $v = x$ and applying the substitution $[u/x]$

Call-By-Push-Value

- Why:
  - Subsumes both Call-By-Value and Call-By-Name
  - All types behave well, even in the presence of effects
- How:
  - Terms are split into values W and computations M:
    - *"A value is, a computation does"* (Levy)
    - Values $\approx$ terms built from variables and constructors of pattern-matchable types (for example $(y, ()))$
    - Computations $\approx$ everything else (all destructors, and constructors for functions...)
    - thunk $M \approx \lambda().M$ and force $W \approx W()$
  - Reductions defined on configurations: $\langle M \parallel K \rangle$

$\mathbf{LJ}_p^\eta$

- Why:
  - Nice rewriting theory (two constructions per type, HORS, can define weak and strong reductions)
  - Commutation rules are "for free"
- How:
  - Intuitionistic sequent calculus with evaluation order
  - Reduction defined on polarised commands $\langle t \parallel e \rangle^\varepsilon$
  - Subterms also represented by commands $\rightsquigarrow$ Can define strong and weak reduction
  - Polarity determines the strategy locally: $\varepsilon = -$ means CBN and $\varepsilon = +$ means CBV
  - A term constructor $\mu\star.c$:
    - Moves focus: $\langle \mu\star.c \parallel S \rangle \rhd_\mu c[S/\star]$, i.e. $S\left[\boxed{\mu\star.c}\right] \rhd S[c]$
    - Allows to define a term by its behaviour in a given context: If we want $\langle f a \parallel S \rangle \rhd \langle f \parallel S[\Box a] \rangle$, we can define application by $f a := \mu\star.\langle f \parallel \star[\Box a] \rangle$

Simulation of CBPV by $\mathbf{LJ}^{\eta}_{p}$

Translation and simulation

- Translation (macro expressible):
  - $\boxed{\cdot}^{\mathsf{W}}_{+} : \mathbb{W} \to \mathbb{V}_{+}$, $\boxed{\cdot}^{\mathsf{M}}_{-} : \mathbb{M} \to \mathbb{V}_{-}$ and $\boxed{\cdot}^{\mathsf{K}}_{-} : \mathbb{K} \to \mathbb{S}$
  - $\boxed{\langle M \parallel K \rangle}^{\mathsf{E}} := \left\langle \boxed{M}^{\mathsf{M}}_{-} \,\middle\|\, \boxed{K}^{\mathsf{K}}_{-} \right\rangle^{-}$
- In CBPV, $K\left[\boxed{\mathrm{pm}\,()\,\mathrm{as}\,().M}\right] \rhd K[M]$ while in $\mathbf{LJ}^{\eta}_{p}$,

$$k\left[\boxed{\mathrm{pm}\,()\,\mathrm{as}\,().t}\right] \rhd k\left[\mathrm{pm}\,\boxed{()}\,\mathrm{as}\,().t\right] \rhd k[t]$$

$$\rightsquigarrow \boxed{E} := \begin{cases} \boxed{E} & \text{if } \boxed{E} \not\rhd_{\mu} \\ c' & \text{if } \boxed{E} \rhd_{\mu} c' \text{ and whenever } E \rhd E', c' \neq \boxed{E'} \end{cases}$$

Proposition (Simulation)

If $E_1 \rhd E_2$ then $\boxed{E_1} \rhd^{+} \boxed{E_2}$.

Proposition

E is a normal form if and only if $\boxed{E}$ is a $\rhd$-normal form.

Simulation of CBPV by $\mathbf{LJ}_p^\eta$

Translation and simulation

Proposition (Simulation)

If $E_1 \triangleright E_2$ *then* $\lceil E_1 \rceil \triangleright^+ \lceil E_2 \rceil$.

Proposition

E *is a normal form if and only if* $\lceil E \rceil$ *is a* $\triangleright$*-normal form.*

Corollary

If $\lceil E_1 \rceil \triangleright c_2$*, then there exist some* E_3 *so that* $E_1 \triangleright E_3$ *and* $c_2 \triangleright^* \lceil E_3 \rceil$.

- $E_1 \triangleright^+ E_2$ *if and only if* $\lceil E_1 \rceil \triangleright^+ \lceil E_2 \rceil$.
- E_2 *is a normal form of* E_1 *if and only if* $\lceil E_2 \rceil$ *is a* $\triangleright$*-normal form of* $\lceil E_1 \rceil$.
- E *is normalising if and only if* $\lceil E \rceil$ *is* $\triangleright$*-normalising.*

Simulation of CBPV by $\mathbf{LJ}_p^\eta$

Type preservation

Proposition

If $\Gamma \vdash M : \underline{B}$ *and* $\Gamma \mid \underline{B} \vdash K : \underline{C}$, *then* $\langle M \| K \rangle : \left(\boxed{\Gamma}_+ \vdash \star : \boxed{\underline{C}}_- \right)$.

Simulation of CBPV by $\mathbf{LJ}_p^\eta$

Type preservation

Definition

A configuration E (resp. command c) is potentially reducible if there is a substitution $\sigma : \mathbb{x} \to \mathbb{W}$ (resp. $\sigma : \mathbb{x} \to \mathbb{V}$) and a stack K (resp. S) so that $E\left[\sigma, K/\mathrm{nil}\right]$ (resp. $c\left[\sigma, S/\star\right]$) is not normal.

Remark

Potentially reducible normal forms correspond to “good” normal form.

Example

$\langle \mathrm{pm}\,()\,\mathrm{as}\,(x_1, x_2).M \parallel K\rangle$ is not potentially reducible but $\langle \mathrm{force}\, x \parallel K\,[\Box W]\rangle$ is because

$\langle \mathrm{force\ thunk}\,(\lambda y.t) \parallel K\,[\Box W]\rangle \rhd \langle \lambda y.t \parallel K\,[\Box W]\rangle \rhd \left\langle t\left[W/y\right] \parallel K\right\rangle$.

Proposition

If E is a potentially reducible normal form then so is $\lceil E \rceil$.

Weak reduction in $\mathbf{LJ}_p^\eta$

- Translation from (non-deterministic) CBV to $\mathbf{LJ}_p^\eta$:
  - Left-to-right: “Evaluate f and bind x_f to the result, evaluate a bind x_a to the result, reduce $x_f x_a$”
  - Right-to-left: Evaluate a before f
- Weak (non-deterministic) CBV reduction is not sent on head reduction by the reduction:
  - In the left to right translation, $\boxed{a} \triangleright \boxed{a'}$ does not imply $\boxed{fa} \triangleright \boxed{f'a}$
  - In the right to left translation, $\boxed{f} \triangleright \boxed{f'}$ does not imply $\boxed{fa} \triangleright \boxed{fa'}$
- On open terms, weak left-to-right CBV reduction is not sent on head reduction.

Weak reduction in $\mathbf{LJ}^{\eta}_{p}$

Warning

In this section, we consider $\mathbf{LJ}^{\eta}_{p}$ without sums.

Definition

The weak reduction $\rightarrow$ on $\mathbf{LJ}^{\eta}_{p}$ is defined as the closure of the head reduction $\rhd$ by all constructors except negative covariable binders ($\mu\star^{-}.c$, $\mu(x \cdot \star).c$, $\mu{<}\star.c_1\,;\star.c_2{>}$ and $\mu\{\star\}.c$).

Lemma

$V \not\rightarrow$, i.e. V is a normal form with respect to $\rightarrow$.

Lemma

If $S \rightarrow S'$, then $c\left[S/\star\right] \rightarrow c\left[S'/\star\right]$.

Weak reduction in $\mathbf{LJ}_p^\eta$

Proposition

The weak reduction $\rightarrow$ is uniformly confluent[1].

Corollary

The weak reduction $\rightarrow$ is confluent.

Proposition

For any closed command c, c is strongly $\rightarrow$-normalising if and only if it is strongly $\triangleright$-normalising.

[1] If $c_l \leftarrow c \rightarrow c_r$ with $c_l \neq c_r$, then there is some c_{lr} so that $c_l \rightarrow c_{lr} \leftarrow c_r$

Conclusion

- Translation from CBPV to $\mathbf{LJ}_p^\eta$
  - Macro-expressible
  - Is a simulation
  - Preserves (“good”) normal forms
  - Preserves typing judgements
- Weak reduction in $\mathbf{LJ}_p^\eta$:
  - CBV and CBN are macro-expressible in $\mathbf{LJ}_p^\eta$
  - Works for non-deterministic CBV
  - Works on open terms