Applicative May- and Should-Simulation in the Call-by-Value Lambda Calculus with AMB

Manfred Schmidt-Schauß, David Sabel

Goethe-University, Frankfurt, Germany

RTA/TLCA '14, Vienna, Austria

Motivation

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

- Semantics of **higher-order programming languages**
- **Nondeterminism** and **concurrency**
- Correctness of **program transformations** (e.g. compiler optimizations)
- **Contextual equivalence** as program semantics
- Requires proof **techniques** and **tools**

Contextual Equivalence for Nondeterminism

Contextual Equivalence, informally:

Programs are equal iff

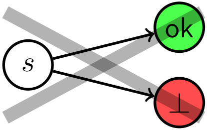
they have the same termination behavior in all program contexts

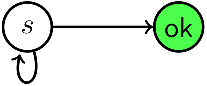
Nondeterminism requires:

- observe whether a program **may** terminate
- **and** observe whether a program **should** (or must) terminate.

Must- and Should termination:

- **must**: terminate (successfully) in any case
- **should**: No possibility to run into an error, weak divergences allowed





Programs s and t are **applicative bisimilar** if

s and t “behave” identically using the following test:

- s terminates with value $v_s \iff t$ terminates with program v_t
- applying v_s and v_t to argument r:
  $(v_s\ r)$ and $(v_t\ r)$ are again applicative bisimilar

Advantages:

- reasoning about contexts is not necessary
- similarity of expressions can be proved by **coinduction**
- a sound similarity is a valuable proof tool

Previous Work and Goals

State of the art:

- several sound applicative similarities for deterministic and nondeterministic calculi exist
  (e.g. Abramsky '90; Howe '89; Ong '93; Lassen & Pitcher '00; Biernacki & Lenglet '12)
- there are some unsound cases:
  - Impure lambda calculi with storage
    (Mason & Talcott '91; Koutavas, Levy & Sumii '10)
  - Nondeterministic languages with recursive bindings
    (Schmidt-Schauß, S., Machkasova '11)
- **none** covers the combination of may- and should-convergence

Our goal

Find a sound applicative similarity for **Should**-Convergence

To keep things simple:

we consider a basic language with nondetermism

McCarthy's amb-Operator

Operational semantics of $(\texttt{amb}\ s\ t)$:

- evaluate s and t **concurrently**
- take **the first result** which becomes available

Equational semantics:

- $\texttt{amb}\ s\ \bot = s = \texttt{amb}\ \bot\ s$ (**bottom-avoidance**)
- $\texttt{amb}\ s\ t = s$ or t if $s \neq \bot \neq t$ (**nondeterminism**)

Expressiveness:

- amb can encode a lot of other nondeterministic operators
- **erratic choice**: $choice\ s\ t = (\texttt{amb}\ (\lambda\_.s)\ (\lambda\_.t))\ id$
- **demonic choice**: $dchoice\ s\ t = (\texttt{amb}\ (\lambda x, y.x)\ (\lambda x, y.y))\ s\ t$
- parallel or, parallel convergence tester, bottom-avoiding list-merge, ...

`amb` is Challenging

- The semantics of `amb` is **studied since several decades**

  (e.g. McCarthy '63, Broy '86, Panangaden '88, Moran '98, Lassen & Moran '99, Lassen '06, Levy '07, S. & Schmidt-Schauß '08)

- Open question whether a **sound applicative similarity** for may- and must-convergence exists (Lassen '06)

- **Negative answer** for a **typed** calculus with may- and **must**-convergence (Levy '07)

Expressions:

$$s, t \in Expr ::= x \mid \lambda x.s \mid (s\ t) \mid (\texttt{amb}\ s\ t)$$

Evaluation contexts:

$$E \in \mathbb{E} ::= [\cdot] \mid (E\ s) \mid ((\lambda x.s)\ E) \mid (\texttt{amb}\ E\ s) \mid (\texttt{amb}\ s\ E)$$

Call-by-value reduction:

$$\begin{array}{ll}
\text{(cbvbeta)} & E[((\lambda x.s)\ (\lambda y.t))] \xrightarrow{LCA} E[s[(\lambda y.t)/x]] \\
\text{(ambl)} & E[(\texttt{amb}\ (\lambda x.s)\ t)] \xrightarrow{LCA} E[\lambda x.s] \\
\text{(ambr)} & E[(\texttt{amb}\ t\ (\lambda x.s))] \xrightarrow{LCA} E[\lambda x.s]
\end{array}$$

Contextual Equivalence in LCA

May-convergence: $s\downarrow$ iff $\exists \lambda x.s' : s \xrightarrow{LCA,*} \lambda x.s'$
(we also write $s \downarrow \lambda x.s'$ in this case)

Should-convergence: $s \Downarrow$ iff $\forall t : s \xrightarrow{LCA,*} t \implies t\downarrow$

Must-Divergence: $s \Uparrow$ iff $\neg(s\downarrow)$

May-Divergence: $s\uparrow$ iff $\neg(s \Downarrow)$ ($= \exists s' : s \xrightarrow{LCA,*} s' \wedge s' \Uparrow$)

Contextual Preorder & Equivalence

For $\xi \in \{\downarrow, \Downarrow, \uparrow, \Uparrow\}$:

- $s \leq_\xi t$ iff for all $C, C[s]$ and $C[t]$ are closed: $C[s]\xi \implies C[t]\xi$
- $s \sim_\xi t$ iff $s \leq_\xi t$ and $t \leq_\xi s$

Contextual preorder: $s \leq_{LCA} t$ iff $s \leq_\downarrow t \wedge s \leq_\Downarrow t$

Contextual equivalence $s \sim_{LCA} t$ iff $s \sim_\downarrow t \wedge s \sim_\Downarrow t$

η^o = *open value-extension* of η:
$s\ \eta^o\ t$ iff $\sigma(s)\ \eta\ \sigma(t)$ for all closing value substitutions σ

$Expr^c$ = all closed expressions

May-Similarity $\preccurlyeq_\downarrow$:

Greatest fixpoint of $F_\downarrow : (Expr^c \times Expr^c) \to (Expr^c \times Expr^c)$ where

$$s\ F_\downarrow(\eta)\ t \text{ if } s{\downarrow}\,\lambda x.s' \implies \left(\exists \lambda x.t' \text{ with } t{\downarrow}\,\lambda x.t' \text{ and } s'\ \eta^o\ t'\right)$$

Lemma

$s \preccurlyeq_\downarrow t$ iff $s{\downarrow}\,\lambda x.s' \implies \left(\exists \lambda x.t' \text{ with } t{\downarrow}\,\lambda x.t' \text{ and } s' \preccurlyeq_\downarrow^o t'\right)$

η^o $=$ *open value-extension* of η:
$s\ \eta^o\ t$ iff $\sigma(s)\ \eta\ \sigma(t)$ for all closing value substitutions σ

$Expr^c$ $=$ all closed expressions

May-Similarity $\preccurlyeq_{\downarrow}$:

Greatest fixpoint of $F_{\downarrow} : (Expr^c \times Expr^c) \to (Expr^c \times Expr^c)$ where

$$s\ F_{\downarrow}(\eta)\ t \ \text{ if } \ s{\downarrow}\,\lambda x.s' \implies \left(\exists \lambda x.t' \text{ with } t{\downarrow}\,\lambda x.t' \text{ and } s'\ \eta^o\ t'\right)$$

Lemma

$s\ \preccurlyeq_{\downarrow} t$ iff $s{\downarrow}\,\lambda x.s' \implies \left(\exists \lambda x.t' \text{ with } t{\downarrow}\,\lambda x.t' \text{ and } s' \preccurlyeq_{\downarrow}^o t'\right)$

Theorem

$\preccurlyeq_{\downarrow}^o \ \subset\ \leq_{\downarrow}$ *and* $\preccurlyeq_{\downarrow}^o$ *is a precongruence.*

Proof: Soundness and precongruence: by Howe's method.
Incompleteness: by counterexample (Lassen'98; Mann'05)

Should-Similarity $\preccurlyeq_{\uparrow}$:

Greatest fixpoint of $F_{\uparrow} : (Expr^c \times Expr^c) \to (Expr^c \times Expr^c)$ where

$s\ F_{\uparrow}(\eta)\ t$ if

- $s\uparrow \implies t\uparrow$
- $t \preccurlyeq_{\downarrow} s$
- $s{\downarrow}\ \lambda x.s' \implies \big(\exists \lambda x.t'$ with $t{\downarrow}\ \lambda x.t'$ and $s'\ \eta^o\ t'\big)$.

Theorem

$\preccurlyeq_{\uparrow}^{o}\ \subset\ \leq_{\uparrow}\ =\ \geq_{\Downarrow}$ *and* $\preccurlyeq_{\uparrow}^{o}$ *is a precongruence.*

Proof: Soundness and precongruence: Howe's method (next slide)
Incompleteness: by counterexample (in the paper)

Goal:

- show that $\preccurlyeq_{\uparrow}^{o}$ is a precongruence
- implies that $\preccurlyeq_{\uparrow}^{o} \;\subseteq\; \leq_{\uparrow}$ (since $s \preccurlyeq_{\uparrow} t$ implies $s\uparrow \implies t\uparrow$)

Problems:

- $\preccurlyeq_{\uparrow}$ is obviously **reflexive and transitive**,
  but there is **no direct proof** of compatibility with contexts

Howe's Method:

- build **candidate** $\preccurlyeq_{H}$ which **is compatible with contexts**
- show that $\preccurlyeq_{H} \;=\; \preccurlyeq_{\uparrow}^{o}$
- implies $\preccurlyeq_{H}$ and $\preccurlyeq_{\uparrow}^{o}$ are precongruences

Precongruence Proof (2)

Candidate Relation $\preccurlyeq_H$

1. If $x \preccurlyeq_{\uparrow}^{o} s$ then $x \ \preccurlyeq_H \ s$.
2. If $\tau(s'_1, \ldots, s'_n) \preccurlyeq_{\uparrow}^{o} s$ with $s_i \ \preccurlyeq_H \ s'_i$, then $\tau(s_1, \ldots, s_n) \preccurlyeq_H s$. (with $\tau = \lambda, @, \texttt{amb}$)

Theorem

$\preccurlyeq_{\uparrow} \ = \ \preccurlyeq_H^c$

Proof sketch:

- $s \preccurlyeq_{\uparrow} t \implies s \preccurlyeq_H^c t$: Induction on the term structure of s
- $s \preccurlyeq_H^c t \implies s \preccurlyeq_{\uparrow} t$: Show that $\preccurlyeq_H^c$ is $F_{\uparrow}$-dense i.e. $\preccurlyeq_H^c \subseteq F_{\uparrow}(\preccurlyeq_H^c)$. Requires to show for $s \preccurlyeq_H^c t$:
  - $s \uparrow \implies t \uparrow$
  - $t \preccurlyeq_{\downarrow} s$
  - $s \downarrow \lambda x.s' \implies \exists \lambda x.t' : t \downarrow \lambda x.t'$ and $s' \preccurlyeq_H t'$

Proof uses $\preccurlyeq_H \subset \preccurlyeq_{\downarrow} \cap \succcurlyeq_{\downarrow}$ and that $\preccurlyeq_{\downarrow}$ is a precongruence.

Main Theorem

For $\alpha \in \{\downarrow, \uparrow\}$:

- **Mutual Similarity** $\approx_\alpha := \preccurlyeq_\alpha \cap \succcurlyeq_\alpha$
- **Bisimilarity** $\simeq_\alpha$: Greatest fixp. of G_α with $G_\alpha(\eta) = F_\alpha(\eta) \cap F_\alpha(\eta^{-1})$

Main Theorem

The similarities $\preccurlyeq_\downarrow^o$ and $\preccurlyeq_\uparrow^o$ are precongruences, the mutual similarities $\approx_\downarrow^o$, $\approx_\uparrow^o$, and the bisimilarity $\simeq_\uparrow^o$ are congruences.
Moreover, the following soundness results hold:

1. $\preccurlyeq_\downarrow^o \ \subset \ \leq_\downarrow$ and $\approx_\downarrow^o \ \subset \ \sim_\downarrow$.
2. $\preccurlyeq_\uparrow^o \ \subset \ \geq_{LCA}$ and $\approx_\uparrow^o \ \subset \ \sim_{LCA}$.
3. $\simeq_\uparrow^o \ \subseteq \ \approx_\uparrow^o \ \subset \ \sim_{LCA}$.

Note: $s \preccurlyeq_\uparrow^o t \implies s \approx_\downarrow t$

$$
\begin{array}{rcl}
(\lambda x.s)\ (\lambda x.t) & \sim_{LCA} & s[\lambda x.t/x] \\
(\texttt{amb}\ \Omega\ s) & \sim_{LCA} & s \\
(\texttt{amb}\ s\ s) & \sim_{LCA} & s \\
(\texttt{amb}\ s\ t) & \sim_{LCA} & (\texttt{amb}\ t\ s) \\
\texttt{amb}\ s_1\ (\texttt{amb}\ s_2\ s_3) & \sim_{LCA} & \texttt{amb}\ (\texttt{amb}\ s_1\ s_2)\ s_3 \\
\underbrace{Y\ \lambda f.\lambda x.\texttt{amb}\ x\ (f\ x)}_{\text{roughly: } f\ x\ =\ \texttt{amb}\ x\ (f\ x)} & \sim_{LCA} & \lambda x.x
\end{array}
$$

Other Definitions of Should-Similarity

- In the paper: other definitions of Should-Similarity
- some are shown to be **unsound**
- for some other definitions their soundness is **open**
- For instance:

  Convex Should-Similarity $\preccurlyeq_{\uparrow X} = \mathrm{gfp}(F_{\uparrow X})$:
  $s\ F_{\uparrow X}(\eta)\ t$ if
  - $s\uparrow \implies t\uparrow$
  - $t \preccurlyeq_{\downarrow} s$
  - $t\Downarrow \implies \big(s\downarrow \lambda x.s' \implies \big(\exists \lambda x.t'$ with $t\downarrow \lambda x.t'$ and $s'\ \eta^o\ t'\big)\big)$.

Proposition

Convex should similarity is **unsound** in LCA.

Expressions:

$$s, t \in \mathit{Expr} ::= x \mid \lambda x.s \mid (s\ t) \mid (\texttt{choice}\ s\ t)$$

Evaluation contexts:

$$E \in \mathbb{E} ::= [\cdot] \mid (E\ s) \mid ((\lambda x.s)\ E)$$

Call-by-value reduction:

$$\begin{array}{lll}
\text{(cbvbeta)} & E[((\lambda x.s)\ (\lambda y.t))] & \xrightarrow{LCC} E[s[(\lambda y.t)/x]] \\
\text{(choicel)} & E[(\texttt{choice}\ s\ t)] & \xrightarrow{LCC} E[s] \\
\text{(choicer)} & E[(\texttt{choice}\ s\ t)] & \xrightarrow{LCC} E[t]
\end{array}$$

May-Similarity in LCC, $\preccurlyeq_{\downarrow}$: $s\ F_{\downarrow}(\eta)\ t$ if:

- $s{\downarrow}\ \lambda x.s' \implies \big(\exists \lambda x.t'$ with $t{\downarrow}\ \lambda x.t'$ and $s'\ \eta^o\ t'\big)$.

Convex Should-Similarity in LCC, $\preccurlyeq_{\uparrow X}$: $s\ F_{\uparrow X}(\eta)\ t$ if:

- $s{\uparrow} \implies t{\uparrow}$
- $t \preccurlyeq_{\downarrow} s$
- $t \Downarrow \implies \big(s{\downarrow}\ \lambda x.s' \implies \big(\exists \lambda x.t'$ with $t{\downarrow}\ \lambda x.t'$ and $s'\ \eta^o\ t'\big)\big)$

Mutual Convex Should-Similarity: $\approx_{\uparrow X} := \preccurlyeq_{\uparrow X} \cap \succcurlyeq_{\uparrow X}$

Theorem

$\preccurlyeq^o_{\uparrow X} \quad \subset \quad \geq_{LCC}$ *and* $\approx^o_{\uparrow X} \quad \subset \quad \sim_{LCC}$.

Proof: Soundness by Howe's method
Incompleteness by counterexample.

Conclusion

- **sound applicative similarities**, and bisimilarities for contextual equivalence with may- and should-convergence
- for call-by-value calculi with `amb` and `choice`
- proof by (adaption of) Howe's method

Further work

- Sound applicative similarity for nondeterministic **call-by-need** calculi with should-convergence
  (may extend results on may-similarity from Mann '05 and Mann & Schmidt-Schauß' 10)
- Sound applicative similarity for **concurrency**, e.g. process calculus CHF (S.& Schmidt-Schauß '11; '12) modeling Concurrent Haskell

Backup slides

Proposition

$\approx_{\downarrow}^{o} \neq \sim_{\downarrow}$

- $Y = \lambda f.(\lambda x.f\ \lambda z.(x\ x\ z))\ (\lambda x.f\ \lambda z.(x\ x\ z))$
- $Top = Y\ \lambda x, y.x$
- $F = \lambda f.\lambda z.choice\ (\lambda x.\Omega)\ ((\lambda x_1, x_2.x_1)\ (f\ z))$
- $Y\ F\ Id$ reduces to $\lambda x_1, \ldots, x_n.\Omega$ for any $n \geq 1$.
- $Y\ F\ Id \sim_{\downarrow} Top$.
- $Top \not\preccurlyeq_{\downarrow} Y\ F\ Id$ since the definition of $\preccurlyeq_{\downarrow}$ requires to choose and fix n before recursively testing.

Proposition

$\preccurlyeq^o_\uparrow \;\neq\; \leq_\uparrow$

- $A = \texttt{choice}\ \Omega\ (\lambda x.A)$,
- $B_0 = \mathit{Top}$, $B_{i+1} = \lambda x.\texttt{choice}\ \Omega\ B_i$; and
- $B = \texttt{choice}\ \Omega\ (\texttt{choice}\ B_0\ (\texttt{choice}\ B_1\ \ldots))$.
- $A \sim_{LCA} B$.
- $A \not\preccurlyeq_\uparrow B$ since
  - $A \preccurlyeq_\uparrow B \implies A \preccurlyeq_\uparrow B_i$ and $A \preccurlyeq_\uparrow B_i \implies A \preccurlyeq_\uparrow B_{i-1}$
  - Thus $A \preccurlyeq_\uparrow B_0$ is required, but $A \not\preccurlyeq_\uparrow \mathit{Top}$ since $A \uparrow$ while $\mathit{Top} \Downarrow$.

Counter Example: Unsoundness of Convex Should-Similarity in LCA

Convex Should-Similarity $\preccurlyeq_{\uparrow X} = \mathrm{gfp}(F_{\uparrow X})$:

$s\ F_{\uparrow X}(\eta)\ t$ if

- $s\uparrow \implies t\uparrow$
- $t \preccurlyeq_{\downarrow} s$
- $t\Downarrow \implies \big(s\downarrow \lambda x.s' \implies \big(\exists \lambda x.t'$ with $t\downarrow \lambda x.t'$ and $s'\ \eta^o\ t'\big)\big)$.

Proposition

Convex should similarity is **unsound** in LCA.

$$b_1 := \lambda x_1.\Omega \qquad b_2 := \lambda x_1, x_2.\Omega \qquad b_3 := \lambda x_1, x_2, x_3.\Omega$$

$$s_1 := \mathtt{amb}\ b_1\ b_3 \qquad s_2 := \mathtt{amb}\ b_1\ (\mathtt{amb}\ b_2\ b_3)$$

- $s_2 \preccurlyeq_{\uparrow X} s_1$: $S \subseteq F_{\uparrow X}(S)$ with $S := \{(s_1, s_2), (b_1, b_1), (b_3, b_3), (b_2, b_1), (b_1, \Omega)\}$
- $s_2 \not\preccurlyeq_{\uparrow} s_1$: $C[s_2]\uparrow$ but $C[s_1]\Downarrow$ with $C := (\mathtt{amb}\ ([\cdot]\ id)\ id)\ id$