Environmental Bisimulations for Higher-Order Languages

Davide Sangiorgi
Naoki Kobayashi
Eijiro Sumii

Main Result

A bisimulation proof technique for various higher-order languages

- Pure λ-calculi (call-by-name/call-by-value)
- Cbv λ-calculus with higher-order store
- Higher-order π-calculus

  - Sound & complete (i.e., characterizes contextual equivalence) in each language

Talk Outline

- Background
  - Contextual equivalence
  - Bisimulation
  - Problems of bisimulation for higher-order languages
- Environmental bisimulation
- Up-to techniques
- Related work

Contextual Equivalence [Morris 73]

Two programs M, N are contextually equivalent

$$M \equiv N$$

if they "behave the same" under any context

E.g., in pure lambda-calculi, $M \equiv N$ if

$$\forall C.\ \ C[M]\Downarrow \text{ iff } C[N]\Downarrow$$

- Direct proof is hard because of "$\forall$C"

  $\Rightarrow$ Proof technique is desired

Bisimulation

Two programs M, N are bisimilar

M ~ N

if they can simulate

each other's input/output behavior

- Soundness: Bisimilar programs are contextually equivalent
- Completeness: Vice versa

  ⇒ Gives a proof technique for contextual equivalence

Problem: Bisimulation for Higher-Order Languages (1/2)

M ~ N if:

1. If M outputs M_1 and becomes M',
   then N outputs N_1 and becomes N'
   with M' ~ N'

What condition is needed for M_1 and N_1?

- "M_1 ~ N_1" is too strong, because M_1 and M' (N_1 and N') may share a "secret"
  ⇒ Incomplete in impure languages

Problem: Bisimulation for Higher-Order Languages (2/2)

$M \sim N$ if:

2. If M becomes M' for input M_1, then N becomes N' for input N_1 with $M' \sim N'$

What condition is needed for M_1 and N_1?

- "$M_1 \sim N_1$" is ill-formed, because it appears in a negative position

  $\Rightarrow$ Bisimilarity ($\sim$) may not exist

Talk Outline

- Background
- Environmental bisimulation
  - Key idea
  - General definition
  - Specific definitions
- Up-to techniques
- Related work

Environmental Bisimulation

Key idea:

Use relation-indexed relation $\sim_R$ to represent the "changing world"

- R is called an environment
- Accounts for the generativity of
  - Locations (in λ-calculus with store),
  - Channels (in higher-order π-calculus), etc.

- Complete also in impure languages
- Monotone (union-closed) and well-defined

General Definifion (1/3)

X is an environmental simulation
if M X_R N implies:

1. If M $\rightarrow$ M', then N $\Rightarrow$ N' and M' X_R N'
2. If M outputs M_1 and becomes M', then N outputs N_1 and becomes N' with M' $X_{R \cup \{(M1, N1)\}}$ N'

General Definifion (2/3)

X is an environmental simulation
if M X_R N implies:

3. For all M_1 R^* N_1,
   if M becomes M' for input M_1,
   then N becomes N' for input N_1
   with M' X_R N'
   - R^* is the <u>context closure</u> of R
     $\{ (C[M_1,\ldots,M_n], C[N_1,\ldots,N_n]) \mid \forall i.\ M_i\ R\ N_i \}$
   - Represents "synthesis of knowledge" by the context

General Definition (3/3)

- X is an environmental bisimulation if both X and X^{-1} are environmental simulations
  - X^{-1} is defined by $(X^{-1})_R = (X_R)^{-1}$
- $\sim$ is the largest environmental bismulation

Instance 1: Env. Bisim. for Higher-Order π-Calculus (Simplified)

X is an environmental simulation if $P \; X_R \; Q$ implies:

1. If $P \rightarrow P'$, then $Q \Rightarrow Q'$ and $P' \; X_R \; Q'$
2. If $P = c!M.P'$, then $Q \Rightarrow c!N.Q'$ and $P' \; X_{R \cup \{(M, N)\}} \; Q'$
3. If $P = c?x.P'$, then $Q \Rightarrow c?x.Q'$ and $P'\{P_1/x\} \; X_R \; Q'\{Q_1/x\}$ for all $P_1 \; R^* \; Q_1$
4. $P \mid P_1 \; X_R \; Q \mid Q_1$ for all $P_1 \; R \; Q_1$

Instance 2: Env. Bisim. for Pure Call-by-Name λ-Calculus

X is an environmental simulation if $M \; X_R \; N$ implies:

1. If $M \to M'$, then $N \Rightarrow N'$ and $M' \; X_R \; N'$
2. If $M = \lambda x.M'$, then $N \Rightarrow \lambda x.N'$ and $\lambda x.M' \; X_{R \cup \{(\lambda x.M', \lambda x.N')\}} \; \lambda x.N'$
   - Moreover, $M'\{M_1/x\} \; X_R \; N'\{N_1/x\}$ for all $M_1 \; R^* \; N_1$

Simple Example (for Pedagogy)

$M = \lambda x.(\lambda y.y)x$ and $N = \lambda x.x$

- Consider $X_0 = \{ (R, M, N) \}$ where $R = \{(M, N)\}$
- For any M_1 R^* N_1,

  $M\ M_1 \rightarrow (\lambda y.y)M_1 \rightarrow M_1$

  $N\ N_1 \rightarrow N_1$
- Extend X_0 to $X =$

  $\{ (R^*, (\lambda y.y)M_1, N_1), (R^*, M_1, N_1) \mid M_1\ R^*\ N_1 \}$
- X is an environmental bisimulation

Talk Outline

- Background
- Environmental bisimulation
- Up-to techniques
  - Big-step environmental bisimulation up to reduction and context
- Related work

Big-Step Env. Bisim. up to Reduction and Context

X is a big-step environmental simulation up to reduction and context if $M \; X_R \; N$ impilies:

- If $M \Rightarrow \lambda x.M'$, then $N \Rightarrow \lambda x.N'$ and for all $M_1 \; R^* \; N_1$,

  $M'\{M_1/x\} \Rightarrow (X_{R \cup \{(\lambda x.M', \lambda x.N')\}})^* \Leftarrow N'\{N_1/x\}$

  - Recall R^* is the context closure of R

The Example Revisited

$M = \lambda x.(\lambda y.y)x$ and $N = \lambda x.x$

- Take $X = \{ (R, M, N) \}$ where $R = \{(M, N)\}$
- For any $M_1 \; R^* \; N_1$,

  $M \; M_1 \Rightarrow M_1$

  $R \; R^* \qquad R^* = (X_R)^*$

  $N \; N_1 \Rightarrow N_1$
- X is a big-step environmental bisimulation up to reduction and context
  - The proof is now as easy as it should be!

In the paper

- Environmental bisimulations for
  - Pure cbv λ-calculus
  - Cbv λ-calculus with higher-order store
- Up-to techniques
  - Up-to environment / bisimilarity / reduction / expansion / contexts / full contexts
  - Combinations of the above
- Soundness and completeness proofs
- More examples

Talk Outline

- Background
- Environmental bisimulation
- Up-to techniques
  - Big-step environmental bisimulation up to reduction and context
- Related work

Applicative Bisimulation [Abramsky 90]

$\lambda x.M \sim \lambda x.N$ if $(\lambda x.M)M_1 \sim (\lambda x.N)M_1$
for any closed term M_1

- Soundness proof is hard [Howe 96]
- Unsound in languages with information hiding
  - Abstract types ($\exists\alpha$), generative names (νx), etc.

Reason:
The lhs and the rhs are "different worlds"

Normal Form Bisimulation [Sangiorgi 94, Lassen et al.]

$\lambda x.M \sim \lambda x.N$ if $(\lambda x.M)y \sim (\lambda x.N)y$ for a fresh variable y

- Easy to use: one argument suffices
- Complete only in languages with control (μ), and state (:=) [Lassen et al.]

Logical Bisimulation [Sangorgi-Kobayashi-Sumii 07]

$\lambda x.M \sim \lambda x.N$ if

$(\lambda x.M)C[M_1,\ldots,M_n] \sim (\lambda x.N)C[N_1,\ldots,N_n]$

<u>for all C with $M_1,\ldots,M_n \sim N_1,\ldots,N_n$</u>

- Sound (and complete in pure λ-calculi)
- Not monotone, but works for pure λ-calculi

Previous "Environmental" Bisimulations

- For first-order languages
  - Polymorphic π-calculus [Pierce-Sangiorgi 00]
  - Spi calculus [Abadi-Gordon 98]
- For higher-order languages (with a few "built-in" up-to techniques)
  - λ-calculi with perfect encryption / existential types [Sumii-Pierce 04, 05]
  - Imperative λ-calculus / object calculi [Koutavas-Wand 06, 06, 07]

Conclusion

- Sound and complete bisimulations for
  - Pure λ-calculi (call-by-name/call-by-value)
  - Cbv λ-calculus with higher-order store
  - Higher-order π-calculus
- Up-to techniques for the bisimulations

Future work:

- "More formal" general framework
- More formal comparison with other proof techniques