Type soundness proofs with big-step operational semantics of OOL

Davide Ancona

DIBRIS, Università di Genova

Shonan Meeting on Semantics and Verification of Object-Oriented Languages
September 21-25, 2015

Motivations

- Proof of type soundness with big-step operational semantics
- Previous proposals use extra rules [LeroyGrall09,ErnstOstermann06,KusmierekBono10]
- Proof techniques for proving type soundness with the standard rules

Outline

- background on inference system
- type soundness with small-step and big-step operational semantics
- coinductive big-step semantics
- approximating semantics
- future directions

Inference systems

- a formalism for recursively defining predicates (or relations)
- strongly related to logic programming
- a simple example
  - definition of predicate **leq**
  - domain: terms over constant 0 and unary operator *s*

$$\text{(rule 1)}\frac{}{\mathbf{leq}(0, N)} \qquad \text{(rule 2)}\frac{\mathbf{leq}(N_1, N_2)}{\mathbf{leq}(s(N_1), s(N_2))}$$

- standard inductive semantics:
  the equation

$$S = \{\mathbf{leq}(0, t) \mid t \text{ term}\} \cup \{\mathbf{leq}(s(t_1), s(t_2)) \mid \mathbf{leq}(t_1, t_2) \in S\}$$

  has only one solution

$$S^{in} = \{\mathbf{leq}(s^n(0), s^m(0)) \mid n \leq m\}$$

Inference systems and coinductive semantics

$$\text{(rule 1)}\,\frac{}{\mathbf{leq}(0, N)} \qquad \text{(rule 2)}\,\frac{\mathbf{leq}(N_1, N_2)}{\mathbf{leq}(s(N_1), s(N_2))}$$

- let us extend the domain with t s.t. $t = s(t)$ (that is, $t = s^\omega$)
- equation

$$S = \{\mathbf{leq}(0, t) \mid t \text{ term}\} \cup \{\mathbf{leq}(s(t_1), s(t_2)) \mid \mathbf{leq}(t_1, t_2) \in S\}$$

  has two different solutions
  - least solution: $S^{in} = \{\mathbf{leq}(s^n(0), s^m(0)) \mid n \leq m\} \cup \{\mathbf{leq}(s^n(0), s^\omega)\}$
  - greatest solution: $S^{co} = S^{in} \cup \{\mathbf{leq}(s^\omega, s^\omega)\}$

Fixed-point semantics

- *HU*: Herbrand Universe of terms
- Herbrand model: a subset of $HB = \{\mathbf{leq}(t_1, t_2) \mid t_1, t_2 \in HU\}$
- from the recursive definition of **leq** the following one step inference function $\mathcal{F}$ can be derived

$$\mathcal{F} : \mathcal{P}(HB) \rightarrow \mathcal{P}(HB)$$

$$\mathcal{F}(S) = \{\mathbf{leq}(0, t) \mid t \in HU\} \cup \{\mathbf{leq}(s(t_1), s(t_2)) \mid \mathbf{leq}(t_1, t_2) \in S\}$$

- Tarski-Knaster theorem
  $\mathcal{F}$ monotone over a complete lattice $\Rightarrow$ there exist $lfp(\mathcal{F})$, and $gfp(\mathcal{F})$

Inductive and coinductive fixed-point semantics

- inductive semantics
  - $HU =$ terms over 0 and s
  - model: $lfp(\mathcal{F}) = \{\mathbf{leq}(s^n(0), s^m(0)) \mid n \leq m\}$
- coinductive semantics
  - $HU =$ finite and infinite terms over 0 and s (non well-founded trees)
  - model: $gfp(\mathcal{F}) = lfp(\mathcal{F}) \cup \{\mathbf{leq}(s^n(0), s^\omega)\} \cup \{\mathbf{leq}(s^\omega, s^\omega)\}$

Proof tree semantics

- $\textbf{leq}(t_1, t_2)$ is derivable iff there exists a proof tree (a.k.a. derivation) s.t.
  - all nodes are labeled over HB
  - the root is labeled with $\textbf{leq}(t_1, t_2)$
  - all nodes with their children constitute a valid instantiation of a rule

example:

$$\text{(rule 2)}\dfrac{\text{(rule 1)}\dfrac{}{\textbf{leq}(0, s(0))}}{\textbf{leq}(s(0), s(s(0)))}$$

- inductive semantics
  - $HU =$ terms over 0 and s
  - proof trees can only be finite
- coinductive semantics
  - $HU =$ finite and infinite terms over 0 and s (non well-founded trees)
  - proof trees are allowed to be infinite (non well-founded) as well

Equivalence of the two semantics

- fixed-point and proof tree semantics are equivalent
- the result holds for both inductive and coinductive semantics
- published proofs can be found in
  - inductive semantics: *P. Aczel, An introduction to inductive definitions, Handbook of Mathematical Logic, Vol. 90 of Studies in Logics and the Foundations of Mathematics, 1977*
  - coinductive semantics: *X. Leroy and H. Grall. Coinductive big-step operational semantics. Information and Computation, 207:284304, 2009*
- example:
  $\mathbf{leq}(s^\omega, s^\omega) \notin \mathit{lfp}(\mathcal{F})$, $\mathbf{leq}(s^\omega, s^\omega) \in \mathit{gfp}(\mathcal{F})$
  the proof tree for $\mathbf{leq}(s^\omega, s^\omega)$ is infinite

$$\text{(rule 2)}\,\dfrac{\text{(rule 2)}\,\dfrac{\vdots}{\mathbf{leq}(s^\omega, s^\omega)}}{\mathbf{leq}(s^\omega, s^\omega)}$$

A Featherweight Java-like language

$$
\begin{array}{rcl}
p & ::= & \overline{cd}^n\ e \\
cd & ::= & \textbf{class}\ c_1\ \textbf{extends}\ c_2\ \{\ \overline{fd}^n\ \overline{md}^k\ \}\quad (c_1 \neq \texttt{Object}) \\
fd & ::= & \tau\ f; \\
md & ::= & \tau_0\ m(\overline{\tau\ x}^n)\ \{e\}\quad x_i \neq \textbf{this}\ \forall i = 1..n \\
\tau & ::= & c \mid \texttt{bool} \\
e & ::= & \textbf{new}\ c(\overline{e}^n) \mid x \mid e.f \mid e_0.m(\overline{e}^n) \mid \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2 \\
 & & \mid \textbf{false} \mid \textbf{true}
\end{array}
$$

assumptions: $n, k \geq 0$, inheritance is acyclic, names of declared classes in a program, methods and fields in a class, and parameters in a method are distinct.

Small-step operational semantics (call by value)

values

$$v \quad ::= \quad \textbf{new}\ c(\overline{v}^n) \mid \textbf{false} \mid \textbf{true}$$

contexts

$$\mathcal{C}[\,] \quad ::= \quad \Box \mid \textbf{new}\ c(\overline{v}^n, \Box, \overline{e}^k) \mid \Box.f \mid \Box.m(\overline{e}^n) \mid v.m(\overline{v}^n, \Box, \overline{e}^k) \mid \textbf{if}\ (\Box)\ e_1\ \textbf{else}\ e_2$$

rewriting rules (with usual auxiliary functions *fields* and *meth*)

$$\text{(fld)}\ \frac{\mathit{fields}(c) = \overline{\tau}^n\ \overline{f}^n, \quad 1 \leq i \leq n}{\textbf{new}\ c(\overline{v}^n).f_i \rightarrow v_i}$$

$$\text{(inv)}\ \frac{\mathit{meth}(c, m) = \overline{\tau}^n\ \overline{x}^n.e{:}\tau}{\textbf{new}\ c(\overline{v}^k).m(\overline{v'}^n) \rightarrow e[\textbf{this} \mapsto \textbf{new}\ c(\overline{v}^k), \overline{x}^n \mapsto \overline{v'}^n]}$$

$$\text{(ift)}\ \frac{}{\textbf{if}\ (\textbf{true})\ e_1\ \textbf{else}\ e_2 \rightarrow e_1} \qquad \text{(iff)}\ \frac{}{\textbf{if}\ (\textbf{false})\ e_1\ \textbf{else}\ e_2 \rightarrow e_2} \qquad \text{(ctx)}\ \frac{e \rightarrow e'}{\mathcal{C}[e] \rightarrow \mathcal{C}[e']}$$

Big-step operational semantics (call by value)

- Values (for the moment let us consider well-founded values)

$$\mathbb{v}, \mathbb{u} \quad ::= \quad obj(c, [\bar{f}^n \mapsto \bar{\mathbb{v}}^n]) \mid false \mid true$$

- Main judgment

$$\Pi \vdash e \Rightarrow \mathbb{v}, \text{ where } \Pi = \bar{x}^n \mapsto \bar{\mathbb{v}}^n$$

Rules

$$(\text{VAR})\ \frac{\Pi(x) = \mathbb{v}}{\Pi \vdash x \Rightarrow \mathbb{v}} \qquad (\text{FAL})\ \frac{}{\Pi \vdash \textbf{false} \Rightarrow \mathit{false}} \qquad (\text{TRU})\ \frac{}{\Pi \vdash \textbf{true} \Rightarrow \mathit{true}}$$

$$(\text{NEW})\ \frac{\forall\, i = 1..n\ \Pi \vdash e_i \Rightarrow \mathbb{v}_i \quad \mathit{fields}(c) = \overline{\tau}^n\ \overline{f}^n}{\Pi \vdash \textbf{new}\ c(\overline{e}^n) \Rightarrow \mathit{obj}(c, [\overline{f}^n \mapsto \overline{\mathbb{v}}^n])} \qquad (\text{IFT})\ \frac{\Pi \vdash e \Rightarrow \mathit{true} \qquad \Pi \vdash e_1 \Rightarrow \mathbb{v}}{\Pi \vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2 \Rightarrow \mathbb{v}}$$

$$(\text{IFF})\ \frac{\Pi \vdash e \Rightarrow \mathit{false} \qquad \Pi \vdash e_2 \Rightarrow \mathbb{v}}{\Pi \vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2 \Rightarrow \mathbb{v}} \qquad (\text{FLD})\ \frac{\Pi \vdash e \Rightarrow \mathit{obj}(c, [\overline{f}^n \mapsto \overline{\mathbb{v}}^n]) \quad 1 \leq i \leq n}{\Pi \vdash e.f_i \Rightarrow \mathbb{v}_i}$$

$$(\text{INV})\ \frac{\begin{array}{c}\forall\, i = 0..n\ \Pi \vdash e_i \Rightarrow \mathbb{v}_i \quad \textbf{this} \mapsto \mathbb{v}_0, \overline{x}^n \mapsto \overline{\mathbb{v}}^n \vdash e \Rightarrow \mathbb{v} \\ \mathbb{v}_0 = \mathit{obj}(c, [\ldots]) \quad \mathit{meth}(c, m) = \overline{\tau}^n\ \overline{x}^n.e{:}\tau\end{array}}{\Pi \vdash e_0.m(\overline{e}^n) \Rightarrow \mathbb{v}}$$

Type system (well-typed programs and declarations)

$$(pro)\ \frac{\forall\, i = 1..n\ \vdash cd_i{:}\diamond \quad \emptyset \vdash e{:}\tau}{\vdash \overline{cd}^n\ e{:}\diamond}$$

$$(cla)\ \frac{\forall\, i = 1..k\ c \vdash md_i{:}\diamond \quad fields(c)\ \text{defined}}{\vdash \textbf{class}\ c\ \textbf{extends}\ c'\ \{\ \overline{fd}^n\ \overline{md}^k\ \}{:}\diamond}$$

$$(met)\ \frac{\textbf{this}{:}c, \overline{x}^n{:}\overline{\tau}^n \vdash e{:}\tau \quad \tau \leq \tau_0 \quad override(c, m, \overline{\tau}^n, \tau_0)}{c \vdash \tau_0\ m(\overline{\tau}^n\ \overline{x}^n)\ \{e\}{:}\diamond}$$

Type system (well-typed expressions)

$$(var)\ \frac{}{\Gamma \vdash x{:}\tau}\ \Gamma(x) = \tau \qquad (fal)\ \frac{}{\Gamma \vdash \textbf{false}{:}bool} \qquad (tru)\ \frac{}{\Gamma \vdash \textbf{true}{:}bool}$$

$$(new)\ \frac{\forall\, i = 1..n\ \Gamma \vdash e_i{:}\tau_i \quad fields(c) = \overline{\tau'}^n\ \overline{f}^n \quad \forall\, i = 1..n\ \tau_i \leq \tau'_i}{\Gamma \vdash \textbf{new}\ c(\overline{e}^n){:}c}$$

$$(fld)\ \frac{\Gamma \vdash e{:}c \quad fields(c) = \overline{\tau}^n\ \overline{f}^n \quad 1 \leq i \leq n}{\Gamma \vdash e.f_i{:}\tau_i}$$

$$(inv)\ \frac{\forall\, i = 0..n\ \Gamma \vdash e_i{:}\tau_i \quad meth(\tau_0, m) = \overline{\tau'}^n\ \overline{x}^n.e{:}\tau \quad \forall\, i = 1..n\ \tau_i \leq \tau'_i}{\Gamma \vdash e_0.m(\overline{e}^n){:}\tau}$$

$$(if)\ \frac{\Gamma \vdash e{:}bool \quad \Gamma \vdash e_1{:}\tau_1 \quad \Gamma \vdash e_2{:}\tau_2}{\Gamma \vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2{:}\vee(\tau_1, \tau_2)}$$

Small-step and big-step operational semantics

Small-step operational semantics

- rewriting system approach
- suited for
  - type soundness proofs
  - concurrency

Big-step operational semantics

- more abstract approach
- suited for
  - interpreter specification
  - imperative languages

Type soundness

- general claim: if P is well-typed, then P cannot crash
- more in detail: if e is well-typed, then any possible evaluation of e either diverges, or returns a value
- with small-step semantics: if e is well-typed, $e \to e'$, and e' is in normal form, then e' is a value

Type soundness and big-step semantics

Program 1

```
true.m() // main expression
```

Program 2

```
class C extends Object {bool m() {this.m()}}
new C().m() // main expression
```

- program 1 is not well-typed, program 2 is well-typed
- program 1 crashes, program 2 does not terminate
- there is no value $\mathtt{v}$ s.t. $\emptyset \vdash \mathbf{true}.\mathtt{m()} \Rightarrow \mathtt{v}$
  $\emptyset \vdash \mathbf{new}\ \mathtt{C().m()} \Rightarrow \mathtt{v}$

Type soundness and big-step semantics

Program 1

```
true.m() // main expression
```

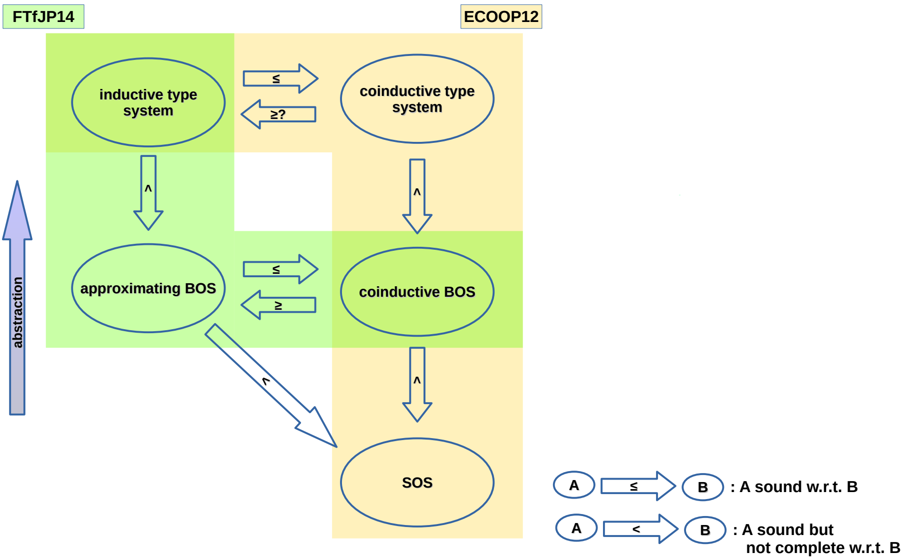
Program 2

```
class C extends Object {bool m() {this.m()}}
new C().m() // main expression
```

- program 1 is not well-typed, program 2 is well-typed
- program 1 crashes, program 2 does not terminate
- there is no value v s.t. $\emptyset \vdash \textbf{true}.\texttt{m()} \Rightarrow \texttt{v}$
 $\emptyset \vdash \textbf{new}\ \texttt{C().m()} \Rightarrow \texttt{v}$

Big-step semantics cannot distinguish programs that crash from programs that do not terminate

Overview of the main results



Coinductive big-step semantics (values)

Definition

$$\mathtt{v}, \mathtt{u} \quad ::= \quad \textit{obj}(c, [\bar{f}^n \mapsto \bar{\mathtt{v}}^n]) \mid \textit{false} \mid \textit{true} \quad (\textbf{non-well-founded})$$

Examples of valid values

$\textit{obj}(\mathtt{C}, [\mathtt{n} \mapsto \textit{obj}(\mathtt{C}, [\mathtt{n} \mapsto \textit{obj}(\mathtt{C}, [\mathtt{n} \mapsto \ldots])])])$
that is, the unique value $\mathtt{v}$ s.t. $\mathtt{v} = \textit{obj}(\mathtt{C}, [\mathtt{n} \mapsto \mathtt{v}])$

$\textit{obj}(\mathtt{L}, [\mathtt{e} \mapsto \textit{obj}(\mathtt{Z}, [\,]), \mathtt{n} \mapsto \textit{obj}(\mathtt{L}, [\mathtt{e} \mapsto \textit{obj}(\mathtt{N}, [\mathtt{p} \mapsto \textit{obj}(\mathtt{Z}, [\,])]), \mathtt{n} \mapsto \ldots])])$

Coinductive big-step semantics (rules)

$$\text{(VAR)}\ \frac{\Pi(x) = \mathbb{v}}{\Pi \Vdash x \Rightarrow \mathbb{v}} \qquad \text{(FAL)}\ \frac{}{\Pi \Vdash \textbf{false} \Rightarrow \mathit{false}} \qquad \text{(TRU)}\ \frac{}{\Pi \Vdash \textbf{true} \Rightarrow \mathit{true}}$$

$$\text{(NEW)}\ \frac{\forall\, i = 1..n\ \Pi \Vdash e_i \Rightarrow \mathbb{v}_i \quad \mathit{fields}(c) = \overline{\tau}^n\ \overline{f}^n}{\Pi \Vdash \textbf{new}\ c(\overline{e}^n) \Rightarrow \mathit{obj}(c, [\overline{f}^n \mapsto \overline{\mathbb{v}}^n])} \qquad \text{(IFT)}\ \frac{\Pi \Vdash e \Rightarrow \mathit{true} \qquad \Pi \Vdash e_1 \Rightarrow \mathbb{v}}{\Pi \Vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2 \Rightarrow \mathbb{v}}$$

$$\text{(IFF)}\ \frac{\Pi \Vdash e \Rightarrow \mathit{false} \qquad \Pi \Vdash e_2 \Rightarrow \mathbb{v}}{\Pi \Vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2 \Rightarrow \mathbb{v}} \qquad \text{(FLD)}\ \frac{\Pi \Vdash e \Rightarrow \mathit{obj}(c, [\overline{f}^n \mapsto \overline{\mathbb{v}}^n]) \quad 1 \leq i \leq n}{\Pi \Vdash e.f_i \Rightarrow \mathbb{v}_i}$$

$$\text{(INV)}\ \frac{\begin{array}{l}\forall\, i = 0..n\ \Pi \Vdash e_i \Rightarrow \mathbb{v}_i \quad \texttt{this} \mapsto \mathbb{v}_0, \overline{x}^n \mapsto \overline{\mathbb{v}}^n \Vdash e \Rightarrow \mathbb{v} \\ \mathbb{v}_0 = \mathit{obj}(c, [\ldots]) \quad \mathit{meth}(c, m) = \overline{\tau}^n\ \overline{x}^n.e{:}\tau\end{array}}{\Pi \Vdash e_0.m(\overline{e}^n) \Rightarrow \mathbb{v}}$$

Inductive and coinductive semantics

- inductive semantics $\subseteq$ coinductive semantics
- the only difference concerns non terminating programs
- if *e* is well-typed and does not terminate, then
  - there exists **no** $\mathbb{v}$ s.t. $\emptyset \vdash e \Rightarrow \mathbb{v}$
  - there exists $\mathbb{v}$ s.t. $\emptyset \Vdash e \Rightarrow \mathbb{v}$
- valid soundness claim: $\Gamma \vdash e{:}\tau \Rightarrow \exists\, \mathbb{v}.\Pi \Vdash e \Rightarrow \mathbb{v}$

Example 1

Program

```
class C extends Object {bool m() {this.m()}}
new C().m() // main expression
```

Example 1

Program

```
class C extends Object {bool m() {this.m()}}
new C().m() // main expression
```

Proof tree

$$\dfrac{\dfrac{}{\emptyset \Vdash \textbf{new}\ \texttt{C()} \Rightarrow \mathbb{u}} \quad \dfrac{\dfrac{}{\Pi \Vdash \texttt{this} \Rightarrow \mathbb{u}} \quad \dfrac{\vdots}{\Pi \Vdash \texttt{this.m()} \Rightarrow \mathbb{v}}}{\Pi \Vdash \texttt{this.m()} \Rightarrow \mathbb{v}}}{\emptyset \Vdash \textbf{new}\ \texttt{C().m()} \Rightarrow \mathbb{v}}$$

- $\emptyset \Vdash e \Rightarrow \mathbb{v}$ for any $\mathbb{v}$
- $\mathbb{u} = \mathit{obj}(\texttt{C}, [\,]), \Pi = \texttt{this} \mapsto \mathbb{u}$

Example 2

Program

```
class M extends Object {L m(){new L(this.m())}}
class L extends Object {L n;}
new M().m() // main expression
```

Example 2

Program

```
class M extends Object {L m(){new L(this.m())}}
class L extends Object {L n;}
new M().m() // main expression
```

Proof tree

$$\dfrac{\dfrac{}{\emptyset \Vdash \textbf{new}\ \texttt{M()} \Rightarrow \textit{obj}(\texttt{M},[\,])} \quad \dfrac{\dfrac{\dfrac{}{\Pi \Vdash \texttt{this} \Rightarrow \textit{obj}(\texttt{M},[\,])} \quad \dfrac{\vdots}{\Pi \Vdash \textbf{new}\ \texttt{L(this.m())} \Rightarrow \mathtt{v}}}{\Pi \Vdash \texttt{this.m()} \Rightarrow \mathtt{v}}}{\Pi \Vdash \textbf{new}\ \texttt{L(this.m())} \Rightarrow \textit{obj}(\texttt{L},[\texttt{n} \mapsto \mathtt{v}])}}{\emptyset \Vdash \textbf{new}\ \texttt{M().m()} \Rightarrow \textit{obj}(\texttt{L},[\texttt{n} \mapsto \mathtt{v}])}$$

- $\mathtt{v} = \textit{obj}(\texttt{L},[\texttt{n} \mapsto \mathtt{v}])$, $\Pi = \texttt{this} \mapsto \textit{obj}(\texttt{M},[\,])$

Example 3

Program

```
class Nat extends Object { }
class Z extends Nat { }
class NZ extends Nat {Nat p;}
class M extends Object {L m(Nat e){new L(e,this.m(new NZ(e)))}}
class L extends Object {Nat e; L n;}
new M().m(new Z()) // main expression
```

Example 3

Proof tree

$$
{}_{\Rightarrow}\nabla_i = \dfrac{\dfrac{}{\Pi_i \Vdash \mathtt{e} \Rightarrow u_i} \qquad \dfrac{\dfrac{}{\Pi_i \Vdash \mathtt{this} \Rightarrow \mathit{obj}(\mathtt{M}, [\,])} \quad \dfrac{\dfrac{}{\Pi_i \Vdash \mathtt{e} \Rightarrow u_i}}{\Pi_i \Vdash \textbf{new}\ \mathtt{NZ(e)} \Rightarrow u_{i+1}} \quad {}_{\Rightarrow}\nabla_{i+1}}{\Pi_i \Vdash \mathtt{this.m(}\textbf{new}\ \mathtt{NZ(e))} \Rightarrow v_{i+1}}}{\Pi_i \Vdash \textbf{new}\ \mathtt{L(e,this.m(}\textbf{new}\ \mathtt{NZ(e)))} \Rightarrow v_i}
$$

$$
\dfrac{\dfrac{}{\emptyset \Vdash \textbf{new}\ \mathtt{M()} \Rightarrow \mathit{obj}(\mathtt{M}, [\,])} \quad \dfrac{}{\emptyset \Vdash \textbf{new}\ \mathtt{Z()} \Rightarrow u_0} \quad {}_{\Rightarrow}\nabla_0}{\emptyset \Vdash \textbf{new}\ \mathtt{M().m(}\textbf{new}\ \mathtt{Z())} \Rightarrow v_0}
$$

- $v_i = \mathit{obj}(\mathtt{L}, [\mathtt{e} \mapsto u_i, \mathtt{n} \mapsto v_{i+1}])$ for all $i \in \mathbb{N}$
  $u_0 = \mathit{obj}(\mathtt{Z}, [\,])$ $\quad u_i = \mathit{obj}(\mathtt{NZ}, [\mathtt{p} \mapsto u_{i-1}])$ for all $i \in \mathbb{N} \setminus \{0\}$
- $\Pi_i = \mathtt{this} \mapsto \mathit{obj}(\mathtt{M}, [\,]), \mathtt{e} \mapsto u_i$ for all $i \in \mathbb{N}$

Example 4

Program

```
class C extends Object {bool m() {this.m()}}
if (new C().m()) true.m() else true.m() // main expression
```

- there exists **no** $\mathtt{v}$ s.t. $\emptyset \Vdash$ **if** (**new** C().m()) **true**.m() **else** **true**.m() $\Rightarrow \mathtt{v}$
- **if** (**new** C().m()) **true**.m() **else** **true**.m() not well-typed

Type soundness

Main claim

$\emptyset \vdash e{:}\tau \Rightarrow \exists\, \mathbb{v}. \emptyset \Vdash e \Rightarrow \mathbb{v}$

Technique

the proof uses a coinductive version of the standard type system

- $\Gamma \vdash e{:}\tau$ (standard system, inductive semantics)
- $\Gamma \Vdash e{:}\tau$ (non standard system, coinductive semantics)

Proof

the claim can be split as follows

1. $\emptyset \vdash e{:}\tau \Rightarrow \emptyset \Vdash e{:}\tau$ (direct proof)
2. $\emptyset \Vdash e{:}\tau \Rightarrow \exists\, \mathbb{v}. \emptyset \Vdash e \Rightarrow \mathbb{v}$

proof of claim 2

- the proof requires to define a complete metric space of proof trees
- based on a concretization relation $\mathcal{R}_\gamma$ (defined coinductively)

Switching from the inductive to the coinductive TS

$$(pro)\ \frac{\forall\, i = 1..n\ \vdash cd_i{:}\diamond \quad \emptyset \vdash e{:}\tau}{\vdash \overline{cd}^n\ e{:}\diamond} \qquad (cla)\ \frac{\forall\, i = 1..k\ c \vdash md_i{:}\diamond \quad \mathit{fields}(c)\ \text{defined}}{\vdash \textbf{class}\ c\ \textbf{extends}\ c'\ \{\ \overline{fd}^n\ \overline{md}^k\ \}{:}\diamond}$$

$$(met)\ \frac{\mathtt{this}{:}c, \overline{x}^n{:}\overline{\tau}^n \vdash e{:}\tau \quad \tau \leq \tau_0 \quad \mathit{override}(c, m, \overline{\tau}^n, \tau_0)}{c \vdash \tau_0\ m(\overline{\tau}^n\ \overline{x}^n)\ \{e\}{:}\diamond}$$

$$(var)\ \frac{\Gamma(x) = \tau}{\Gamma \vdash x{:}\tau} \qquad (fal)\ \frac{}{\Gamma \vdash \textbf{false}{:}\mathit{bool}} \qquad (tru)\ \frac{}{\Gamma \vdash \textbf{true}{:}\mathit{bool}}$$

$$(new)\ \frac{\forall\, i = 1..n\ \Gamma \vdash e_i{:}\tau_i \quad \mathit{fields}(c) = \overline{\tau'}^n\ \overline{f}^n \quad \forall\, i = 1..n\ \tau_i \leq \tau_i'}{\Gamma \vdash \textbf{new}\ c(\overline{e}^n){:}c}$$

$$(fld)\ \frac{\Gamma \vdash e{:}c \quad \mathit{fields}(c) = \overline{\tau}^n\ \overline{f}^n \quad 1 \leq i \leq n}{\Gamma \vdash e.f_i{:}\tau_i} \qquad (if)\ \frac{\Gamma \vdash e{:}\mathit{bool} \quad \Gamma \vdash e_1{:}\tau_1 \quad \Gamma \vdash e_2{:}\tau_2}{\Gamma \vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2{:}\vee(\tau_1, \tau_2)}$$

$$(inv)\ \frac{\forall\, i = 0..n\ \Gamma \vdash e_i{:}\tau_i \quad \mathit{meth}(\tau_0, m) = \overline{\tau'}^n\ \overline{x}^n.e{:}\tau \quad \forall\, i = 1..n\ \tau_i \leq \tau_i'}{\Gamma \vdash e_0.m(\overline{e}^n){:}\tau}$$

Switching from the inductive to the coinductive TS

$$(pro)\ \frac{\forall i = 1..n\ \Gamma \Vdash : cd_i \diamond \quad \emptyset \Vdash e : \tau}{\Gamma \Vdash : \overline{cd}^n\ e \diamond} \qquad (cla)\ \frac{\forall i = 1..k\ \Gamma \Vdash c : md_i \diamond \quad fields(c) \text{ defined}}{\Gamma \Vdash : \textbf{class}\ c\ \textbf{extends}\ c'\ \{\ \overline{fd}^n\ \overline{md}^k\ \} \diamond}$$

$$(met)\ \frac{\Gamma \Vdash \texttt{this}{:}c, \overline{x}^n{:}\overline{\tau}^n : e\tau \quad \tau \leq \tau_0 \quad override(c, m, \overline{\tau}^n, \tau_0)}{\Gamma \Vdash c : \tau_0\ m(\overline{\tau}^n\ \overline{x}^n)\ \{e\} \diamond}$$

$$(var)\ \frac{\Gamma(x) = \tau}{\Gamma \Vdash x : \tau} \qquad (fal)\ \frac{}{\Gamma \Vdash \textbf{false} : bool} \qquad (tru)\ \frac{}{\Gamma \Vdash \textbf{true} : bool}$$

$$(new)\ \frac{\forall i = 1..n\ \Gamma \Vdash e_i : \tau_i \quad fields(c) = \overline{\tau'}^n\ \overline{f}^n \quad \forall i = 1..n\ \tau_i \leq \tau'_i}{\Gamma \Vdash \textbf{new}\ c(\overline{e}^n) : c}$$

$$(fld)\ \frac{\Gamma \Vdash e : c \quad fields(c) = \overline{\tau}^n\ \overline{f}^n \quad 1 \leq i \leq n}{\Gamma \Vdash e.f_i : \tau_i} \qquad (if)\ \frac{\Gamma \Vdash e : bool \quad \Gamma \Vdash e_1 : \tau_1 \quad \Gamma \Vdash e_2 : \tau_2}{\Gamma \Vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2 : \vee(\tau_1, \tau_2)}$$

$$(co\text{-}inv)\ \frac{\begin{array}{c}\forall i = 0..n. \Gamma \Vdash e_i : \tau_i \quad \texttt{this}{:}\tau_0, \overline{x}^n{:}\overline{\tau}^n \Vdash e : \tau' \\ meth(\tau_0, m) = \overline{\tau'}^n\ \overline{x}^n.e : \tau \quad \forall i = 1..n. \tau_i \leq \tau'_i, \tau' \leq \tau\end{array}}{\Gamma \Vdash e_0.m(\overline{e}^n) : \tau}$$

Relation between the coinductive TS and BOS

$$
{}_{\Vdash}\nabla = \dfrac{\dfrac{}{\emptyset \Vdash \textbf{new}\ \texttt{M():M}} \qquad \dfrac{\dfrac{\dfrac{}{\texttt{this:M} \Vdash \texttt{this:M}} \quad \dfrac{\vdots}{\texttt{this:M} \Vdash \textbf{new}\ \texttt{L(this.m()):L}}}{\texttt{this:M} \Vdash \texttt{this.m():L}}}{\texttt{this:M} \Vdash \textbf{new}\ \texttt{L(this.m()):L}}}{\emptyset \Vdash \textbf{new}\ \texttt{M().m():L}}
$$

Relation between the coinductive TS and BOS

$$\nabla_{\Rightarrow} = \cfrac{\emptyset \Vdash \textbf{new}\ \texttt{M()} \Rightarrow \textit{obj}(\texttt{M}, [\,]) \qquad \cfrac{\cfrac{\Pi \Vdash \texttt{this} \Rightarrow \textit{obj}(\texttt{M}, [\,]) \qquad \cfrac{\vdots}{\Pi \Vdash \textbf{new}\ \texttt{L(this.m())} \Rightarrow \mathbb{v}}}{\Pi \Vdash \texttt{this.m()} \Rightarrow \mathbb{v}}}{\Pi \Vdash \textbf{new}\ \texttt{L(this.m())} \Rightarrow \textit{obj}(\texttt{L}, [\texttt{n} \mapsto \mathbb{v}])}}{\emptyset \Vdash \textbf{new}\ \texttt{M().m()} \Rightarrow \textit{obj}(\texttt{L}, [\texttt{n} \mapsto \mathbb{v}])}$$

Relation between the coinductive TS and BOS

$$
{}_{\Rightarrow}\nabla = \dfrac{\dfrac{}{\emptyset \Vdash \textbf{new}\ \texttt{M()} \Rightarrow \textit{obj}(\texttt{M}, [\,])} \qquad \dfrac{\dfrac{\dfrac{}{\Pi \Vdash \texttt{this} \Rightarrow \textit{obj}(\texttt{M}, [\,])} \qquad \dfrac{\vdots}{\Pi \Vdash \textbf{new}\ \texttt{L(this.m())} \Rightarrow \mathtt{v}}}{\Pi \Vdash \texttt{this.m()} \Rightarrow \mathtt{v}}}{\Pi \Vdash \textbf{new}\ \texttt{L(this.m())} \Rightarrow \textit{obj}(\texttt{L}, [\texttt{n} \mapsto \mathtt{v}])}}{\emptyset \Vdash \textbf{new}\ \texttt{M().m()} \Rightarrow \textit{obj}(\texttt{L}, [\texttt{n} \mapsto \mathtt{v}])}
$$

${}_{\Vdash}\nabla \mathcal{R}_{\gamma}\, {}_{\Rightarrow}\nabla$: derivation ${}_{\Rightarrow}\nabla$ is a concretization of derivation ${}_{\Vdash}\nabla$

Relationship with the small-step semantics

Coinductive BOS is sound w.r.t. SOS

If $\emptyset \Vdash e \Rightarrow \mathbb{v}$, $e \xrightarrow{*} e'$, and e' is in normal form, then e' is a value, and $\emptyset \Vdash e' \Rightarrow \mathbb{v}$

Proof: progress + subject reduction

- progress: if $\emptyset \Vdash e \Rightarrow \mathbb{v}$, then either e is a value, or there exists e' s.t. $e \rightarrow e'$
- subject reduction: if $\emptyset \Vdash e \Rightarrow \mathbb{v}$, and $e \rightarrow e'$, then $\emptyset \Vdash e' \Rightarrow \mathbb{v}$

Coinductive BOS is not complete w.r.t. SOS

Counter-example: example 4

```
class C extends Object {bool m() {this.m()}}
if(new C().m()) true.m() else true.m() // main expression
```

there exists **no** $\mathbb{v}$ s.t. $\emptyset \Vdash$ **if** (**new** `C().m()`) **true**`.m()` **else** **true**`.m()` $\Rightarrow \mathbb{v}$

Approximating big-step semantics

Drawbacks of coinductive big-step semantics

Involved proof of soundness based on

- additional equivalent coinductive type system
- complete metric space of big-step semantics proof trees

Approximating big-step semantics

Support for simpler soundness proof

- no additional type system
- fully inductive type soundness proof
- no coinduction, infinite values, and metric spaces!

Approximating semantics

$$\mathbb{v}, \mathbb{u} \quad ::= \quad obj(c, [\overline{f}^n \mapsto \overline{\mathbb{v}}^n]) \mid false \mid true$$
$$\Pi \quad ::= \quad \overline{x_i}^n \mapsto \overline{\mathbb{v}}^n \qquad \overline{x_i}^n \text{ distinct}$$

$$(\text{VAR})\ \frac{\Pi(x) = \mathbb{v}}{\Pi \vdash x \Rightarrow \mathbb{v}} \qquad (\text{FAL})\ \frac{}{\Pi \vdash \textbf{false} \Rightarrow false} \qquad (\text{TRU})\ \frac{}{\Pi \vdash \textbf{true} \Rightarrow true}$$

$$(\text{NEW})\ \frac{\forall\, i = 1..n.\Pi \vdash e_i \Rightarrow \mathbb{v}_i \quad fields(c) = \overline{\tau}^n\ \overline{f}^n}{\Pi \vdash \textbf{new}\ c(\overline{e}^n) \Rightarrow obj(c, [\overline{f}^n \mapsto \overline{\mathbb{v}}^n])} \qquad (\text{IFT})\ \frac{\Pi \vdash e \Rightarrow true \qquad \Pi \vdash e_1 \Rightarrow \mathbb{v}}{\Pi \vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2 \Rightarrow \mathbb{v}}$$

$$(\text{IFF})\ \frac{\Pi \vdash e \Rightarrow false \qquad \Pi \vdash e_2 \Rightarrow \mathbb{v}}{\Pi \vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2 \Rightarrow \mathbb{v}} \qquad (\text{FLD})\ \frac{\Pi \vdash e \Rightarrow obj(c, [\overline{f}^n \mapsto \overline{\mathbb{v}}^n]) \quad 1 \leq i \leq n}{\Pi \vdash e.f_i \Rightarrow \mathbb{v}_i}$$

$$(\text{INV})\ \frac{\begin{array}{c}\forall\, i = 0..n.\Pi \vdash e_i \Rightarrow \mathbb{v}_i \quad \texttt{this} \mapsto \mathbb{v}_0, \overline{x}^n \mapsto \overline{\mathbb{v}}^n \vdash e \Rightarrow \mathbb{v} \\ \mathbb{v}_0 = obj(c, [\ldots]) \quad meth(c, m) = \overline{\tau}^n\ \overline{x}^n.e{:}\tau\end{array}}{\Pi \vdash e_0.m(\overline{e}^n) \Rightarrow \mathbb{v}}$$

Approximating semantics

$$\mathbb{v}, \mathbb{u} \quad ::= \quad obj(c, [\overline{f}^n \mapsto \overline{\mathbb{v}}^n]) \mid false \mid true$$
$$\Pi \quad ::= \quad \overline{x_i}^n \mapsto \overline{\mathbb{v}}^n \qquad \overline{x_i}^n \text{ distinct}$$

$$\text{(VAR)}\ \frac{\Pi(x) = \mathbb{v}}{\Pi \vdash x \Rightarrow \mathbb{v}} \qquad \text{(FAL)}\ \frac{}{\Pi \vdash \textbf{false} \Rightarrow false} \qquad \text{(TRU)}\ \frac{}{\Pi \vdash \textbf{true} \Rightarrow true}$$

$$\text{(NEW)}\ \frac{\forall\, i = 1..n. \Pi \vdash e_i \Rightarrow \mathbb{v}_i \quad fields(c) = \overline{\tau}^n\ \overline{f}^n}{\Pi \vdash \textbf{new}\ c(\overline{e}^n) \Rightarrow obj(c, [\overline{f}^n \mapsto \overline{\mathbb{v}}^n])} \qquad \text{(IFT)}\ \frac{\Pi \vdash e \Rightarrow true \qquad \Pi \vdash e_1 \Rightarrow \mathbb{v}}{\Pi \vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2 \Rightarrow \mathbb{v}}$$

$$\text{(IFF)}\ \frac{\Pi \vdash e \Rightarrow false \qquad \Pi \vdash e_2 \Rightarrow \mathbb{v}}{\Pi \vdash \textbf{if}\ (e)\ e_1\ \textbf{else}\ e_2 \Rightarrow \mathbb{v}} \qquad \text{(FLD)}\ \frac{\Pi \vdash e \Rightarrow obj(c, [\overline{f}^n \mapsto \overline{\mathbb{v}}^n]) \quad 1 \leq i \leq n}{\Pi \vdash e.f_i \Rightarrow \mathbb{v}_i}$$

$$\text{(INV)}\ \frac{\begin{array}{c}\forall\, i = 0..n. \Pi \vdash e_i \Rightarrow \mathbb{v}_i \quad \texttt{this} \mapsto \mathbb{v}_0, \overline{x}^n \mapsto \overline{\mathbb{v}}^n \vdash e \Rightarrow \mathbb{v} \\ \mathbb{v}_0 = obj(c, [\ldots]) \quad meth(c, m) = \overline{\tau}^n\ \overline{x}^n.e{:}\tau\end{array}}{\Pi \vdash e_0.m(\overline{e}^n) \Rightarrow \mathbb{v}} \qquad \text{(APPROX)}\ \frac{}{\Pi \vdash e \Rightarrow \mathbb{v}}$$

Derived judgment $\Pi \vdash^{\approx}_{n} e \Rightarrow \mathbb{v}$
a finite derivation for $\Pi \vdash e \Rightarrow \mathbb{v}$ where (APPROX) is allowed only at depth $\geq n$.

Example 2 revisited

```
class M extends Object {L m(){new L(this.m())}}
class L extends Object {L n;}
```

Coinductive semantics

$$
\dfrac{\dfrac{}{\emptyset \Vdash \textbf{new}\ \texttt{M()} \Rightarrow obj(\texttt{M},[\,])} \qquad \dfrac{\dfrac{\dfrac{}{\Pi \Vdash \texttt{this} \Rightarrow obj(\texttt{M},[\,])} \qquad \dfrac{\vdots}{\Pi \Vdash \textbf{new}\ \texttt{L(this.m())} \Rightarrow \mathtt{v}_0}}{\Pi \Vdash \texttt{this.m()} \Rightarrow \mathtt{v}_0}}{\Pi \Vdash \textbf{new}\ \texttt{L(this.m())} \Rightarrow \mathtt{v}_0}}{\emptyset \Vdash \textbf{new}\ \texttt{M().m()} \Rightarrow \mathtt{v}_0}
$$

where $\Pi = \texttt{this} \mapsto obj(\texttt{M},[\,])$ $\qquad \mathtt{v}_0 = obj(\texttt{L},[\texttt{n} \mapsto \mathtt{v}_0])$

Approximating semantics

$\emptyset \vdash^{\approx}_{0} \textbf{new}\ \texttt{M().m()} \Rightarrow \mathtt{v}_{any}$

$\emptyset \vdash^{\approx}_{1} \textbf{new}\ \texttt{M().m()} \Rightarrow \mathtt{v}_{any}$

Example 2 revisited

```
class M extends Object {L m(){new L(this.m())}}
class L extends Object {L n;}
```

Coinductive semantics

$$\dfrac{\dfrac{}{\emptyset \Vdash \mathbf{new}\ \mathtt{M()} \Rightarrow \mathit{obj}(\mathtt{M}, [\,])} \quad \dfrac{\dfrac{\dfrac{}{\Pi \Vdash \mathtt{this} \Rightarrow \mathit{obj}(\mathtt{M}, [\,])} \quad \dfrac{\vdots}{\Pi \Vdash \mathbf{new}\ \mathtt{L(this.m())} \Rightarrow \mathtt{v}_0}}{\Pi \Vdash \mathtt{this.m()} \Rightarrow \mathtt{v}_0}}{\Pi \Vdash \mathbf{new}\ \mathtt{L(this.m())} \Rightarrow \mathtt{v}_0}}{\emptyset \Vdash \mathbf{new}\ \mathtt{M().m()} \Rightarrow \mathtt{v}_0}$$

where $\Pi = \mathtt{this} \mapsto \mathit{obj}(\mathtt{M}, [\,])$ $\quad \mathtt{v}_0 = \mathit{obj}(\mathtt{L}, [\mathtt{n} \mapsto \mathtt{v}_0])$

Approximating semantics

$\emptyset \vdash_2^{\approx} \mathbf{new}\ \mathtt{M().m()} \Rightarrow \mathit{obj}(\mathtt{L}, [\mathtt{n} \mapsto \mathtt{v}_{any}])$

$\emptyset \vdash_3^{\approx} \mathbf{new}\ \mathtt{M().m()} \Rightarrow \mathit{obj}(\mathtt{L}, [\mathtt{n} \mapsto \mathtt{v}_{any}])$

Example 2 revisited

```
class M extends Object {L m(){new L(this.m())}}
class L extends Object {L n;}
```

Coinductive semantics

$$\dfrac{\dfrac{}{\emptyset \Vdash \textbf{new}\ \texttt{M()} \Rightarrow obj(\texttt{M},[\,])} \qquad \dfrac{\dfrac{\dfrac{}{\Pi \Vdash \texttt{this} \Rightarrow obj(\texttt{M},[\,])} \quad \dfrac{\vdots}{\Pi \Vdash \textbf{new}\ \texttt{L(this.m())} \Rightarrow v_0}}{\Pi \Vdash \texttt{this.m()} \Rightarrow v_0}}{\Pi \Vdash \textbf{new}\ \texttt{L(this.m())} \Rightarrow v_0}}{\emptyset \Vdash \textbf{new}\ \texttt{M().m()} \Rightarrow v_0}$$

where $\Pi = \texttt{this} \mapsto obj(\texttt{M},[\,])$ $\quad v_0 = obj(\texttt{L},[\texttt{n} \mapsto v_0])$

Approximating semantics

$\emptyset \vdash^{\approx}_{4} \textbf{new}\ \texttt{M().m()} \Rightarrow obj(\texttt{L},[\texttt{n} \mapsto obj(\texttt{L},[\texttt{n} \mapsto v_{any}])])$

$\emptyset \vdash^{\approx}_{5} \textbf{new}\ \texttt{M().m()} \Rightarrow obj(\texttt{L},[\texttt{n} \mapsto obj(\texttt{L},[\texttt{n} \mapsto v_{any}])])$

Example 2 revisited

```
class M extends Object {L m(){new L(this.m())}}
class L extends Object {L n;}
```

Coinductive semantics

$$\dfrac{\dfrac{}{\emptyset \Vdash \mathbf{new}\ \mathtt{M()} \Rightarrow obj(\mathtt{M},[\,])} \quad \dfrac{\dfrac{\dfrac{}{\Pi \Vdash \mathtt{this} \Rightarrow obj(\mathtt{M},[\,])} \quad \dfrac{\vdots}{\Pi \Vdash \mathbf{new}\ \mathtt{L(this.m())} \Rightarrow \mathtt{v}_0}}{\Pi \Vdash \mathtt{this.m()} \Rightarrow \mathtt{v}_0}}{\Pi \Vdash \mathbf{new}\ \mathtt{L(this.m())} \Rightarrow \mathtt{v}_0}}{\emptyset \Vdash \mathbf{new}\ \mathtt{M().m()} \Rightarrow \mathtt{v}_0}$$

where $\Pi = \mathtt{this} \mapsto obj(\mathtt{M},[\,])$ $\quad \mathtt{v}_0 = obj(\mathtt{L},[\mathtt{n} \mapsto \mathtt{v}_0])$

Approximating semantics

$$\emptyset \vdash^{\approx}_{n} \mathbf{new}\ \mathtt{M().m()} \Rightarrow obj(\mathtt{L},[\mathtt{n} \mapsto obj(\mathtt{L},[\mathtt{n} \mapsto \ldots \mapsto \mathtt{v}_{any}])])$$
$$\emptyset \vdash^{\approx}_{n+1} \mathbf{new}\ \mathtt{M().m()} \Rightarrow obj(\mathtt{L},[\mathtt{n} \mapsto obj(\mathtt{L},[\mathtt{n} \mapsto \ldots \mapsto \mathtt{v}_{any}])])$$

Relationship with the other semantics

Equivalence of approx. and coind. big-step semantics

- if $\emptyset \Vdash e \Rightarrow \mathbb{v}$, then $\emptyset \vdash_i^{\approx} e \Rightarrow \mathbb{v}$ for all $i \in \mathbb{N}$ (by def.)
- if $\emptyset \vdash_i^{\approx} e \Rightarrow \mathbb{v}_i$ for all $i \in \mathbb{N}$, then there exists $\mathbb{v}$ s.t. $\emptyset \Vdash e \Rightarrow \mathbb{v}$

  the proof exploits the compactness of the metric space of values (standard distance between non-well-founded trees)

Approximating semantics is sound w.r.t. SOS

if $\emptyset \vdash_i^{\approx} e \Rightarrow \mathbb{v}_i$ for all $i \in \mathbb{N}$, $e \xrightarrow{*} e'$, and e' is in normal form, then e' is a value

directly from the soundness of BOS

Type soundness

Claim

if $\emptyset \vdash e{:}\tau$, then for all $k \in \mathbb{N}$ there exists $\mathbb{v}_k$ s.t. $\emptyset \vdash^{\approx}_{k} e \Rightarrow \mathbb{v}_k$

Proof

By induction on k and case analysis on e.

- $k = 0$: immediate, since all types are inhabited
- $k > 0$: sketch for method invocation

$$(inv)\ \frac{\forall\, i = 0..n\ \Gamma \vdash e_i{:}\tau_i \quad meth(\tau_0, m) = \overline{\tau'}^n\ \overline{x}^n.e{:}\tau \quad \forall\, i = 1..n\ \tau_i \leq \tau'_i}{\Gamma \vdash e_0.m(\overline{e}^n){:}\tau}$$

$$(\text{INV})\ \frac{\begin{array}{c}\forall\, i = 0..n.\Pi \vdash^{\approx}_{k-1} e_i \Rightarrow \mathbb{v}_i \quad \mathtt{this} \mapsto \mathbb{v}_0, \overline{x}^n \mapsto \overline{\mathbb{v}}^n \vdash^{\approx}_{k-1} e \Rightarrow \mathbb{v} \\ \mathbb{v}_0 = obj(c, [\ldots]) \quad meth(c, m) = \overline{\tau}^n\ \overline{x}^n.e{:}\tau\end{array}}{\Pi \vdash^{\approx}_{k} e_0.m(\overline{e}^n) \Rightarrow \mathbb{v}}$$

Ongoing and future work

- extension to an imperative oo language
- non determinism
- complete approximating semantics

Questions?

Thank you!