모듈별 프로그램 따로 분석

이준협

2023년 8월 4일

ROPAS Show & Tell

Problem Statement

- Given two program segments e_1 and e_2,
- Want to derive a sound approximation of what $e_1 \propto e_2$ evaluates to, by *separately* analyzing e_1 and e_2.

Overview

Abstract Syntax

Concrete Semantics

Concrete Linking

Abstract Syntax

Abstract Syntax

$$
\begin{array}{lrcll}
\text{Identifiers} & x, M & \in & \mathrm{Var} & \\
\text{Expression} & e & \rightarrow & x & \text{value identifier} \\
 & & \mid & \lambda x.e & \text{function} \\
 & & \mid & e\ e & \text{application} \\
 & & \mid & e \propto e & \text{linked expression} \\
 & & \mid & \varepsilon & \text{empty module} \\
 & & \mid & M & \text{module identifier} \\
 & & \mid & \mathtt{let}\ x\ e\ e & \text{binding expression} \\
 & & \mid & \mathtt{let}\ M\ e\ e & \text{binding module}
\end{array}
$$

Example

```
(* A.ml *)
let true x y = x

(* main.ml *)
open A
let id x = x
;;
id true
```

```
(* In our language *)
(let A =
  let true = \x.\y.x in ε
in ε ) ⋊
  (A ⋊
    ((let id = \x.x
    in ε ) ⋊
      (id true)))
```

Concrete Semantics

Semantic Domains

$$
\begin{array}{rrcl}
\text{Time} & t & \in & \mathbb{T} \\
\text{Context} & C & \in & \mathsf{Ctx}(\mathbb{T}) \\
\text{Value(Expr)} & v & \in & \mathsf{Val}(\mathbb{T}) \triangleq \mathsf{Expr} \times \mathsf{Ctx}(\mathbb{T}) \\
\text{Value(Expr/Mod)} & V & \in & \mathsf{Val}(\mathbb{T}) \uplus \mathsf{Ctx}(\mathbb{T}) \\
\text{Memory} & m & \in & \mathsf{Mem}(\mathbb{T}) \triangleq \mathbb{T} \xrightarrow{\mathsf{fin}} \mathsf{Val}(\mathbb{T}) \\
\text{State} & s & \in & \mathsf{State}(\mathbb{T}) \triangleq \mathsf{Ctx}(\mathbb{T}) \times \mathsf{Mem}(\mathbb{T}) \times \mathbb{T} \\
\text{Result} & r & \in & \mathsf{Result}(\mathbb{T}) \triangleq (\mathsf{Val}(\mathbb{T}) \uplus \mathsf{Ctx}(\mathbb{T})) \times \mathsf{Mem}(\mathbb{T}) \times \mathbb{T} \\
\text{Tick} & \mathsf{tick} & \in & \mathsf{Tick}(\mathbb{T}) \triangleq (\mathsf{State}(\mathbb{T}) \times \mathsf{Var} \times \mathsf{Val}(\mathbb{T})) \to \mathbb{T} \\
\text{Context} & C & \to & [] \\
 & & \mid & (x, t) :: C \\
 & & \mid & (M, C) :: C \\
\text{Value(Expr)} & v & \to & \langle \lambda x.e, C \rangle
\end{array}
$$

addr and ctx

- addr(C, x) looks up when x was bound in C.
- ctx(C, M) looks up what is bound to M in C

$$\mathsf{addr}(C,x) \triangleq \begin{cases} \bot & C = [] \\ t & C = (x,t) :: C' \\ \mathsf{addr}(C',x) & C = (x',t) :: C' \wedge x' \neq x \\ \mathsf{addr}(C'',x) & C = (M,C') :: C'' \end{cases}$$

$$\mathsf{ctx}(C,M) \triangleq \begin{cases} \bot & C = [] \\ C' & C = (M,C') :: C'' \\ \mathsf{ctx}(C'',M) & C = (M',C') :: C'' \wedge M' \neq M \\ \mathsf{ctx}(C',M) & C = (x,t) :: C' \end{cases}$$

Operational Semantics : $\rightsquigarrow_{\text{tick}}$

$$\boxed{(e, C, m, t) \rightsquigarrow_{\text{tick}} (V, m', t') \text{ or } (e', C', m', t')}$$

$$[\textsc{ExprVar}]\ \frac{t_x = \mathsf{addr}(C, x) \qquad v = m(t_x)}{(x, C, m, t) \rightsquigarrow (v, m, t)}$$

$$[\textsc{AppL}]\ \frac{}{(e_1\ e_2, C, m, t) \rightsquigarrow (e_1, C, m, t)} \qquad [\textsc{AppR}]\ \frac{(e_1, C, m, t) \rightsquigarrow (\langle \lambda x.e_\lambda, C_\lambda\rangle, m_\lambda, t_\lambda)}{(e_1\ e_2, C, m, t) \rightsquigarrow (e_2, C, m_\lambda, t_\lambda)}$$

$$[\textsc{AppBody}]\ \frac{\begin{array}{c}(e_1, C, m, t) \rightsquigarrow (\langle \lambda x.e_\lambda, C_\lambda\rangle, m_\lambda, t_\lambda) \\ (e_2, C, m_\lambda, t_\lambda) \rightsquigarrow (v, m_a, t_a)\end{array}}{(e_1\ e_2, C, m, t) \rightsquigarrow (e_\lambda, (x, t_a) :: C_\lambda, m_a[t_a \mapsto v], \mathsf{tick}((C, m_a, t_a), x, v))}$$

$\mathbb{T}$ and tick

Q: So, what $\mathbb{T}$ and tick should we use?

A: Anything! (If it satisfies two requirements)

$\mathbb{T}$ and tick

Q: So, what $\mathbb{T}$ and tick should we use?

A: Anything! (If it satisfies two requirements)

1. tick produces a strictly larger timestamp with respect to some total order on $\mathbb{T}$.

$$\forall t \in \mathbb{T} : t < \mathsf{tick}((\_, \_, t), \_, \_)$$

2. All $(V, m, t) \in \mathsf{Result}(\mathbb{T})$ satisfies: $V < t \wedge m < t$

$$C < t \triangleq \begin{cases} \mathsf{True} & C = [] \\ t' < t \wedge C' < t & C = (x, t') :: C' \\ C' < t \wedge C'' < t & C = (M, C') :: C'' \end{cases} \qquad V < t \triangleq \begin{cases} C < t & V = \langle \_, C \rangle \\ C < t & V = C \end{cases}$$

$$m < t \triangleq \forall t' \in \mathsf{dom}(m) : t' < t \wedge m(t') < t$$

Then all ticks will produce *fresh* timestamps.

Irrelevence of tick

Q: How can we consider semantics instantiated with different $\mathbb{T}$ and tick to be equivalent?

A: If they have the same structure!

Irrelevence of tick

Q: How can we consider semantics instantiated with different $\mathbb{T}$ and tick to be equivalent?

A: If they have the same structure!

- Let $r = (V, m, t) \in \text{Result}(\mathbb{T})$, $r' = (V', m', t') \in \text{Result}(\mathbb{T}')$.
- If we can *translate* the timestamps in r into r' and vice versa, r and r' are *isomorphic* ($r \cong r'$).

Irrelevence of tick

Q: How can we consider semantics instantiated with different $\mathbb{T}$ and tick to be equivalent?

A: If they have the same structure!

- Let $r = (V, m, t) \in \text{Result}(\mathbb{T})$, $r' = (V', m', t') \in \text{Result}(\mathbb{T}')$.
- If we can *translate* the timestamps in r into r' and vice versa, r and r' are *isomorphic* ($r \cong r'$).
- "Can translate r to r'":
  $\exists f \in \mathbb{T} \to \mathbb{T}' : f(V) = V' \land f \circ m = m' \circ f \land f(t) = t'$.

Irrelevence of tick

If we start from isomorphic states, the execution stays isomorphic!

Irrelevence of tick

If we start from isomorphic states, the execution stays isomorphic!

Notation: $\ell \in \mathsf{L} \triangleq \mathsf{Expr} \times \mathsf{State}(\mathbb{T})$, $\rho \in \mathsf{R} \triangleq \mathsf{L} \cup \mathsf{Result}(\mathbb{T})$

Theorem (Irrelevence of tick)

Let $s \in State(\mathbb{T})$ and $s' \in State(\mathbb{T}')$. If $s \cong s'$, then:

$$\forall \mathsf{tick}, \mathsf{tick}', e, \rho \;:\; (e, s) \rightsquigarrow_{\mathsf{tick}} \rho \Rightarrow \exists \rho' \;:\; (e, s') \rightsquigarrow_{\mathsf{tick}'} \rho' \wedge \rho \cong \rho'$$

Collecting Semantics

$$[\![e]\!]S \subseteq (\mathsf{R} \times \mathsf{Tick}(\mathbb{T})) \cup (\mathsf{L} \times \mathsf{Tick}(\mathbb{T}) \times \mathsf{R})$$

- Collecting semantics of e under $S \subseteq \mathsf{State}(\mathbb{T}) \times \mathsf{Tick}(\mathbb{T})$.
- Collect *all* configurations in the proof tree starting from $\{((e, s), \mathsf{tick}) | (s, \mathsf{tick}) \in S\} \subseteq \mathsf{R} \times \mathsf{Tick}(\mathbb{T})$.

Collecting Semantics

Definition (Transfer Function)

$$\mathsf{Step}(A) \triangleq \left\{ \ell \rightsquigarrow_{\mathsf{tick}} \rho, (\rho, \mathsf{tick}) \middle| \frac{A'}{\ell \rightsquigarrow_{\mathsf{tick}} \rho} \land A' \subseteq A \land (\ell, \mathsf{tick}) \in A \right\}$$

Definition (Collecting Semantics)

$$[\![e]\!]S \triangleq \mathsf{lfp}(\lambda X.\mathsf{Step}(X) \cup \{((e,s),\mathsf{tick}) | (s,\mathsf{tick}) \in S\})$$

Concrete Linking

Our Goal

What does $e_1 \infty e_2$ evaluate to under S? : $|[\![e_1 \infty e_2]\!]S|$

- Normally, first evaluate e_1 under S : $|[\![e_1]\!]S|$
- Then evaluate e_2 under the exported states : $|[\![e_2]\!]|[\![e_1]\!]S||$

$$|[\![e_1 \infty e_2]\!]S| = |[\![e_2]\!]|[\![e_1]\!]S||$$

- Instead, collect e_2 *in advance* under an *assumed* S_2 : $[\![e_2]\!]S_2$
- Later, check if the assumption is guaranteed : $|[\![e_1]\!]S| \cong S_1 \rhd S_2$
- Fill in the blanks with S_1, then finish : $S_1 \infty [\![e_2]\!]S_2$

$$|[\![e_1 \infty e_2]\!]S| \cong |S_1 \infty [\![e_2]\!]S_2|$$

Definitions to Come

1. What is $|[\![e]\!]S|$ and $|S_1 \propto [\![e]\!]S_2|$(final results)?

Definitions to Come

1. What is $|[\![e]\!]S|$ and $|S_1 \propto [\![e]\!]S_2|$(final results)?

$$|[\![e]\!]S| \triangleq \{r|(e, \_) \rightsquigarrow r \in [\![e]\!]S\} \quad |S_1 \propto [\![e]\!]S_2| \triangleq \{r|(e, \_) \rightsquigarrow r \in S_1 \propto [\![e]\!]S_2\}$$

Definitions to Come

1. What is $|\llbracket e \rrbracket S|$ and $|S_1 \propto \llbracket e \rrbracket S_2|$(final results)?

$$|\llbracket e \rrbracket S| \triangleq \{r | (e, \_) \rightsquigarrow r \in \llbracket e \rrbracket S\} \quad |S_1 \propto \llbracket e \rrbracket S_2| \triangleq \{r | (e, \_) \rightsquigarrow r \in S_1 \propto \llbracket e \rrbracket S_2\}$$

2. What is $\rhd$(injection)?
3. What is $\propto$(semantic linking)?

Definition of $\triangleright$: Desired Properties

Elementwise injection: $(s_+, \mathsf{tick}_+) = (s_1, \mathsf{tick}_1) \triangleright (s_2, \mathsf{tick}_2)$

1. If $s_2 = ([], \varnothing, 0)$, $s_+ \cong s_1$
2. If $(e, s_2) \rightsquigarrow^*_{\mathsf{tick}_2} (e', s_2')$, then $(s_+', \mathsf{tick}_+') = (s_1, \mathsf{tick}_1) \triangleright (s_2', \mathsf{tick}_2)$ must satisfy $\mathsf{tick}_+ = \mathsf{tick}_+'$ and $(e, s_+) \rightsquigarrow^*_{\mathsf{tick}_+} (e', s_+')$.

Definition of $\triangleright$, Step 1: Fill in the Blanks

$$C_2\langle C_1\rangle \triangleq \begin{cases} C_1 & C_2 = [] \\ (x,t) :: C'\langle C_1\rangle & C_2 = (x,t) :: C' \\ (M, C'\langle C_1\rangle) :: C''\langle C_1\rangle & C_2 = (M, C') :: C'' \end{cases}$$

$$C_2\langle C_1\rangle^{-1} \triangleq \begin{cases} [] & C_2 = C_1 \vee C_2 = [] \\ (x,t) :: C'\langle C_1\rangle^{-1} & C_2 = (x,t) :: C' \\ (M, C'\langle C_1\rangle^{-1}) :: C''\langle C_1\rangle^{-1} & C_2 = (M, C') :: C'' \end{cases}$$

Definition of $\triangleright$, Step 1: Fill in the Blanks

$$C_2\langle C_1\rangle \triangleq \begin{cases} C_1 & C_2 = [] \\ (x,t) :: C'\langle C_1\rangle & C_2 = (x,t) :: C' \\ (M, C'\langle C_1\rangle) :: C''\langle C_1\rangle & C_2 = (M, C') :: C'' \end{cases}$$

$$C_2\langle C_1\rangle^{-1} \triangleq \begin{cases} [] & C_2 = C_1 \vee C_2 = [] \\ (x,t) :: C'\langle C_1\rangle^{-1} & C_2 = (x,t) :: C' \\ (M, C'\langle C_1\rangle^{-1}) :: C''\langle C_1\rangle^{-1} & C_2 = (M, C') :: C'' \end{cases}$$

$$V_2\langle C_1\rangle \triangleq \begin{cases} C_2\langle C_1\rangle & V_2 = C_2 \\ \langle \lambda x.e, C_2\langle C_1\rangle\rangle & V_2 = \langle \lambda x.e, C_2\rangle \end{cases}$$

$$m_2\langle C_1\rangle \triangleq \bigcup_{t\in \mathsf{dom}(m_2)} \{t \mapsto m_2(t)\langle C_1\rangle\}$$

$$r_2\langle s_1\rangle \triangleq (V_2\langle C_1\rangle, m_1 \cup m_2\langle C_1\rangle, t_2)$$

Definition of $\triangleright$, Step 2: tick_+

Note $C_2\langle C_1\rangle \in \mathsf{Ctx}(\mathbb{T}_1 + \mathbb{T}_2)$. What should tick_+ be?

$$\mathsf{tick}_+((C, m, t), x, v) \triangleq \begin{cases} \mathsf{tick}_1((C.1, m.1, t), x, v.1) & t \in \mathbb{T}_1 \\ \mathsf{tick}_2((C\langle C_1\rangle^{-1}.2, m\langle C_1\rangle^{-1}.2, t), x, v\langle C_1\rangle^{-1}.2) & t \in \mathbb{T}_2 \end{cases}$$

where

$$C.i \triangleq \begin{cases} [] & C = [] \\ (x, t) :: C'.i & C = (x, t) :: C' \land t \in \mathbb{T}_i \\ C'.i & C = (x, t) :: C' \land t \notin \mathbb{T}_i \\ (M, C'.i) :: C''.i & C = (M, C') :: C'' \end{cases}$$

$$V.i \triangleq \begin{cases} C.i & V = C \\ \langle \lambda x.e, C.i\rangle & V = \langle \lambda x.e, C\rangle \end{cases}$$

$$m.i \triangleq \bigcup_{t \in \mathsf{dom}(m) \cap \mathbb{T}_i} \{t \mapsto m(t).i\}$$

Definition of $\triangleright$

Definition (Definition of $\triangleright$)

$$(s_1, \mathsf{tick}_1) \triangleright (r_2, \mathsf{tick}_2) \triangleq (r_2\langle s_1\rangle, \mathsf{tick}_+)$$

$$S_1 \triangleright S_2 \triangleq \{(s_1, \mathsf{tick}_1) \triangleright (s_2, \mathsf{tick}_2) | (s_1, \mathsf{tick}_1) \in S_1 \land (s_2, \mathsf{tick}_2) \in S_2\}$$

To inject into a cache,

$$(s_1, \mathsf{tick}_1) \triangleright (\rho_2, \mathsf{tick}_2) \triangleq \begin{cases} (r_+, \mathsf{tick}_+) & \rho_2 = r_2 \land (r_+, \mathsf{tick}_+) = (s_1, \mathsf{tick}_1) \triangleright (r_2, \mathsf{tick}_2) \\ ((e, s_+), \mathsf{tick}_+) & \rho_2 = \ell_2 = (e, s_2) \land (s_+, \mathsf{tick}_+) = (s_1, \mathsf{tick}_1) \triangleright (s_2, \mathsf{tick}_2) \end{cases}$$

$$(s_1, \mathsf{tick}_1) \triangleright (\ell_2 \rightsquigarrow_{\mathsf{tick}_2} \rho_2) \triangleq \ell_+ \rightsquigarrow_{\mathsf{tick}_+} \rho_+$$

$$\text{where } (\ell_+, \mathsf{tick}_+) = (s_1, \mathsf{tick}_1) \triangleright (\ell_2, \mathsf{tick}_2) \land (\rho_+, \mathsf{tick}_+) = (s_1, \mathsf{tick}_1) \triangleright (\rho_2, \mathsf{tick}_2)$$

Definition of ∞

Definition (Semantic Linking)

Let $S_1 \subseteq \mathsf{State}(\mathbb{T}_1) \times \mathsf{Tick}(\mathbb{T}_1)$ and $A_2 \subseteq (\mathsf{L}_2 \times \mathsf{Tick}(\mathbb{T}_2) \times \mathsf{R}_2) \cup (\mathsf{R}_2 \times \mathsf{Tick}(\mathbb{T}_2))$. Then:

$$S_1 \infty A_2 \triangleq \mathsf{lfp}(\lambda X.\mathsf{Step}(X) \cup (S_1 \rhd A_2))$$

Advance

Lemma (Advance)

Let $S_1 \subseteq State(\mathbb{T}_1) \times Tick(\mathbb{T}_1)$ *and* $S_2 \subseteq State(\mathbb{T}_2) \times Tick(\mathbb{T}_2)$.
Then:

$$[\![e]\!](S_1 \rhd S_2) = S_1 \infty [\![e]\!]S_2$$

Thus

$$|[\![e_1 \infty e_2]\!]S| = |[\![e_2]\!]|[\![e_1]\!]S|| \cong |[\![e_2]\!](S_1 \rhd S_2)| = |S_1 \infty [\![e_2]\!]S_2|$$

when $\cong$ is due to the separability assumption and irrelevence of tick

A Trivial Example

- When $S_2 = \text{empty} \triangleq \{([], \varnothing, 0)\}$.
- Any S is trivially separable as $S \cong S \rhd \text{empty}$, so $|[\![e_1]\!]S| \cong |[\![e_1]\!]S| \rhd \text{empty}$
- Since our language is CBV λ-calculus, can't do much better

Corollary (A Simple Case)

$$|[\![e_1 \propto e_2]\!]S| \cong ||[\![e_1]\!]S| \propto [\![e_2]\!]\text{empty}|$$

A Sketch of Analysis

- Abstract $\mathbb{T}$ by a finite $\mathbb{T}^{\#}$
- Use a sound $\mathsf{tick}^{\#}$ satisfying $\mathsf{tick}^{\#} \circ \alpha = \alpha \circ \mathsf{tick}$
- Define sound $\triangleright^{\#}$ and $\infty^{\#}$