Jan Hoffman lecture 2 [2018/06/03]

1

Proof of Lemma 1

$P(\Gamma, e, \tau) =$ If $\Gamma \vdash e : \tau_1$ and $\Gamma \vdash e : \tau_2$ then $\tau_1 = \tau_2$.

- Case (plus): Then $e = plus(e_1, e_2)$ and $\Gamma \vdash e_1 : num$ and $\Gamma \vdash e_2 : num$ and $\tau_1 : num$

  Since $\Gamma \vdash plus(e_1, e_2) : \tau_2$ by inversion $\Gamma \vdash e_1 : num$ $\Gamma \vdash e_2 : num$, and $\tau_2 : num$
  Thus $\tau_1 = \tau_2 = num$ ✓

- Case (var): Then $e = x$ and $\Gamma = \Gamma', x : \tau_1$

  Since $\Gamma \vdash x : \tau_2$ by inversion $\Gamma = \Gamma'', x : \tau_2$
  But then $\Gamma', x : \tau_1 = \Gamma'', x : \tau_2$ and $\tau_1 = \tau_2$ ✓

Other cases are similar.

Lemma 2 (Substitution) If $\Gamma, x : \tau \vdash e' : \tau'$ and $\Gamma \vdash e : \tau$ then $\Gamma \vdash [e/x]e' : \tau'$

EX: $x : num \vdash \underbrace{x \leq 5}_{e'} : bool$, $\cdot \vdash \underbrace{6}_{e} : num$

$[e/x]e' = 6 \leq 5 : bool$

Proof: (by rule induction)

Jan Hoffman

[2018/06/05]

2

Lemma 3 (weakening) If $\Gamma \vdash e : \tau$ and $x \notin \Gamma$
then $\Gamma, x : \tau' \vdash e : \tau$

(The idea is that $x : \tau'$ must be irrelevant to $e : \tau$. Otherwise it would appear in Γ in the judgment $\Gamma \vdash e : \tau$.)

Dynamic Semantics

(What does it mean to run or evaluate a prog?)

Different Kinds

- operational semantics: How to run a prog
- axiomatic semantics: What can you prove about a program?
- denotational semantics: Describe programs as mathematical functions.

Operational Semantics (our focus)

Today: structural (or small step) operational sem.

Structural Dynamics

transition system (low level, very flexible)

4 judgments: s state s initial
s final $s \longmapsto s'$
(s can step to s')

Jan Hoffman [2018/06/03] (3)

Iterated transition: $\longmapsto$ (single step) $\longmapsto^*$ (many steps)

$$\frac{}{s \longmapsto^* s} \qquad \frac{s \longmapsto s' \quad s' \longmapsto^* s''}{s \longmapsto^* s''}$$

- States are expressions (well-typed & closed) (no free variables)
- all states are initial
- values are final

Values: e val

$$\frac{}{\mathrm{num}[n] : \mathrm{val}} \qquad \frac{}{\mathrm{bool}[b] : \mathrm{val}}$$

(Digression)

Def: e is closed & well-typed if $\cdot \vdash e : \tau$ for some type τ.

(For this judgment to be valid in an empty context, e cannot have free variables (ie. e is closed).)

Transitions.

$$\frac{n = n_1 + n_2}{\mathrm{plus}(\mathrm{num}[n_1], \mathrm{num}[n_2]) \longmapsto \mathrm{num}[n]}$$

$$\frac{e_1 \longmapsto e_1'}{\mathrm{plus}(e_1, e_2) \longmapsto \mathrm{plus}(e_1', e_2)}$$

$$\frac{e_2 \longmapsto e_2'}{\mathrm{plus}(\mathrm{num}[n], e_2) \longmapsto \mathrm{plus}(\mathrm{num}[n], e_2')}$$

Transition for Let

(a)
$$\frac{e_1 \longmapsto e_1'}{let(e_1, x.e_2) \longmapsto let(e_1', x.e_2)}$$

(b)
$$\frac{e_1 \ val}{let(e_1, x.e_2) \longmapsto [e_1/x]\, e_2}$$ (call-by-value)

OR, instead of (a) & (b), we could have the rule (c):

(c)
$$\frac{}{let(e_1, x.e_2) \longmapsto [e_1/x]\, e_2}$$ (call-by-name)

Transition for If/then/else

$$\frac{e \longmapsto e'}{if(e, e_1, e_2) \longmapsto if(e', e_1, e_2)}$$

$$\frac{}{if(bool[true], e_1, e_2) \longmapsto e_1} \qquad \frac{}{if(bool(false), e_1, e_2) \longmapsto e_2}$$

(Similar transition rules for leq.)

Example

let $x = 8+2$ in $(x+x)+2$ (call-by-val)

$\longmapsto$ let $x = 10$ in $(x+x)+2$

$\longmapsto (10+10)+2 \longmapsto 20+2 \longmapsto 22.$

let $x = 8+2$ in $(x+x)+2$ (call-by-name)

$\longmapsto ((8+2)+(8+2))+2$

$\longmapsto (10+10)+2 \longmapsto 20+2 \longmapsto 22$

Lemma: There is no expr e such that e val and $e \longmapsto e'$ for some e'.

Lemma: If $e \longmapsto e_1$ and $e \longmapsto e_2$ then $e_1 =_\alpha e_2$.

Type Safety

- You don't get stuck in the dynamics

Theorem

1. (progress) If $\cdot \vdash e : \tau$ then either e val or $\exists e'.\ e \longmapsto e'$.
2. (preservation) If $\cdot \vdash e : \tau$ and $e \longmapsto e'$, then $\cdot \vdash e' : \tau$

Proof: (progress) Rule induction on $e : \tau$.

Rule (Plus): Then $e = plus(e_1, e_2)$ $\tau = num$, $e_1 : num$. $e_2 : num$.

IH: either e_1 val or $\exists e_1'.\ e_1 \longmapsto e_1'$
either e_2 val or $\exists e_2'.\ e_2 \longmapsto e_2'$

- Case (e_1 val, e_2 val) Then by canonical forms lemma (which we didn't cover), $e_1 = num[n_1]$ $e_2 = num[n_2]$ for some n_1, n_2. But then $e \longmapsto num[n_1 + n_2]$.

Jan Hoffman (Lect2) [2018/06/03] 6

- Case (e_1 val, $e_2 \mapsto e_2'$) Then by canonical forms lemma $e_1 = \text{num}[n_1]$ and $e \mapsto \text{plus}(\text{num}[n_1], e_2')$

- Case ($e_1 \mapsto e_1'$) Then $e \mapsto \text{plus}(e_1', e_2)$.

We have proved progress for Plus.
We would also have to do the same for each of the other rules.

Homework

1. Add Mult
2. Do other cases of progress
3. Do preservation proof.