Big Step Semantics

$e \mapsto e'$ "e steps to e'" (one step)
$e \mapsto^* e'$ "e steps to e'" (many steps)
$e \mapsto^* v$ and v val "e evaluates to v"

$e \Downarrow v$ "e evaluates to v"

$e ::= x \mid () \mid \lambda x.e \mid e\ e \mid (e,e) \mid \text{fst } e \mid \text{snd } e \mid \text{inl } e \mid \text{inr } e \mid \text{case } e \text{ of } \{x.e; y.e\} \mid \text{fix } x = e$
$\tau ::= \text{unit} \mid \tau \to \tau \mid \tau \times \tau \mid \tau + \tau$
$v ::= () \mid \lambda x.e \mid (v,v) \mid \text{inl } v \mid \text{inr } v$

Combines 3 step rules for application

$$\frac{}{v \Downarrow v}\ (E\text{-}1) \qquad \frac{e_1 \Downarrow \lambda x.e \quad e_2 \Downarrow v \quad [v/x]e \Downarrow v'}{e_1\ e_2 \Downarrow v'}\ (E\text{-}2)$$

$$\frac{e_1 \Downarrow v_1 \quad e_2 \Downarrow v_2}{(e_1, e_2) \Downarrow (v_1, v_2)}\ (E\text{-}3) \qquad \frac{e \Downarrow (v_1, v_2)}{\text{fst } e \Downarrow v_1}\ (E\text{-}4) \qquad \frac{e \Downarrow (v_1, v_2)}{\text{snd } e \Downarrow v_2}\ (E\text{-}5)$$

$$\frac{e \Downarrow v}{\text{inl } e \Downarrow \text{inl } v}\ (E\text{-}6) \qquad \frac{e \Downarrow v}{\text{inr } e \Downarrow \text{inr } v}\ (E\text{-}7)$$

$$\frac{e_1 \Downarrow \text{inl } v \quad [v/x]e_2 \Downarrow v'}{\text{case } e_1 \text{ of } \{x.e_2; y.e_3\} \Downarrow v'}\ (E\text{-}8) \qquad \frac{e_1 \Downarrow \text{inr } v \quad [v/x]e_3 \Downarrow v'}{\text{case } e_1 \text{ of } \{x.e_2; y.e_3\} \Downarrow v'}\ (E\text{-}9)$$

Way less rules!

$$\frac{[\text{fix } x = e / x]e \Downarrow v}{\text{fix } x = e \Downarrow v}\ (E\text{-}10)$$

Preservation: If $\Gamma \vdash e : \tau$ and $e \Downarrow v$, then $\Gamma \vdash v : \tau$.

Pf

(E-1) ✓

(E-2) Then $e = e_1\ e_2$ and $e_1 \Downarrow \lambda x.e$ and $e_2 \Downarrow v'$ and $[v/x]e \Downarrow v$.

By inversion, $\tau = \tau_2$ and $\Gamma \vdash e_1 : \tau_1 \to \tau_2$ and $\Gamma \vdash e_2 : \tau_1$.

By induction, $\Gamma \vdash \lambda x.e : \tau_1 \to \tau_2$ and $\Gamma \vdash v' : \tau_1$.

By inversion, $\Gamma, x:\tau_1 \vdash e : \tau_2$.

By substitution, $\Gamma \vdash [v/x]e : \tau_2$. By induction, $\Gamma \vdash v : \tau_2$.

(E-3) Then $e = (e_1, e_2)$ and $e_1 \Downarrow v_1$ and $e_2 \Downarrow v_2$ and $v = (v_1, v_2)$.

By inversion, $\tau = \tau_1 \times \tau_2$ and $\Gamma \vdash e_1 : \tau_1$ and $\Gamma \vdash e_2 : \tau_2$.

By induction, $\Gamma \vdash v_1 : \tau_1$ and $\Gamma \vdash v_2 : \tau_2$.

By typing rules, $\Gamma \vdash v : \tau_1 \times \tau_2$.

(E-4) Then $e = \text{fst}\ e_0$ and $e_0 \Downarrow (v, v_2)$.

By inversion, $\Gamma \vdash e_0 : \tau \times \tau_2$. By induction, $\Gamma \vdash (v, v_2) : \tau \times \tau_2$.

By inversion, $\Gamma \vdash v : \tau$.

(E-6) Then $e = \text{inl}\ e_0$ and $e_0 \Downarrow v'$ and $v = \text{inl}\ v'$.

By inversion, $\tau = \tau_1 + \tau_2$ and $\Gamma \vdash e_0 : \tau_1$.

By induction, $\Gamma \vdash v' : \tau_1$. By typing rules, $\Gamma \vdash v : \tau_1 + \tau_2$.

(E-8) Then $e = \text{case}\ e_1\ \text{of}\ \{x.e_2 ; y.e_3\}$ and $e_1 \Downarrow \text{inl}\ v'$ and $[v'/x]e_2 \Downarrow v$.

By inversion, $\Gamma \vdash e_1 : \tau_1 + \tau_2$ and $\Gamma, x:\tau_1 \vdash e_2 : \tau$.

By induction, $\Gamma \vdash \text{inl}\ v' : \tau_1 + \tau_2$. By inversion, $\Gamma \vdash v' : \tau_1$.

By subst, $\Gamma \vdash [v'/x]e_2 : \tau$. By induction, $\Gamma \vdash v : \tau$.

(E-10) Then $e = \text{fix}\ x = e_0$ and $[\text{fix}\ x = e_0 / x]e_0 \Downarrow v$.

By inv, $\Gamma, x:\tau \vdash e : \tau$. By subst and ind, $\Gamma \vdash v : \tau$.

Progress: ...
One option: If $\Gamma \vdash e : \tau$, then $\exists v$ s.t. $e \Downarrow v$
True in STLC w/o fix but not in most real languages.

$\nexists v.$ fix $x = x \Downarrow v.$

Big-step can't talk about non-terminating expressions
No real way to talk about progress ($\Rightarrow$ type safety).

But: - Don't need to worry about evaluation order
- Don't need all the search rules

Thm. $e \mapsto^* v \iff e \Downarrow v$
$\Leftarrow$ Fairly straightforward with a couple annoying lemmas
$\Rightarrow$ Suffices to show
Lemma. If $e \mapsto e'$ and $e' \Downarrow v$ then $e \Downarrow v$.

Cost Semantics

How long does a program take to run?

$e \mapsto^k v$ $\quad \underbrace{e \mapsto e_1 \mapsto e_2 \mapsto \cdots \mapsto e_{n-1} \mapsto v}_{n \text{ steps}}$

$e \Downarrow v$ not so clear

$e \Downarrow_n v$ "e evaluates to v in 'time' n"

$$\frac{}{v \Downarrow_0 v} \qquad \frac{e_1 \Downarrow_{n_1} \lambda x\, e \quad e_2 \Downarrow_{n_2} v \quad [v/x]e \Downarrow_{n_3} v'}{e_1\, e_2 \Downarrow_{n_1+n_2+n_3+1} v'}$$

$$\frac{e_1 \Downarrow_{n_1} v_1 \quad e_2 \Downarrow_{n_2} v_2}{(e_1, e_2) \Downarrow_{n_1+n_2} (v_1, v_2)} \qquad \frac{e \Downarrow_n (v_1, v_2)}{\text{fst } e \Downarrow_{n+1} v_1} \quad \cdots$$

Thm: $e \mapsto^n v$ iff $e \Downarrow_n v$

Why is this useful?

Static cost analysis $\quad \bullet \vdash e : \tau ; n$ - has type τ, takes $\leq n$ steps to run

If $\bullet \vdash e : \tau ; n$ and $e \mapsto^m e'$ then $m \leq n$. - Hard to show

If $\bullet \vdash e : \tau ; n$ and $e \Downarrow^m v$ then $m \leq n$ - easier

$$\underbrace{\bullet \vdash e : \tau ; n \Rightarrow e \Downarrow^{\leq n} v}_{\text{We did the static analysis right}} \underbrace{\Rightarrow e \mapsto^{\leq n} v}_{\text{We did the cost semantics right}}$$