# Stack Machines

$$\cfrac{\cfrac{\cfrac{\mathrm{fst}\,(7,8) \mapsto 7}{(\lambda x.x)(\mathrm{fst}\,(7,8)) \mapsto (\lambda x.x)\,7}}{(6, (\lambda x.x)(\mathrm{fst}\,(7,8))) \mapsto (6, (\lambda x.x)\,7)}}{(5, (6, (\lambda x.x)(\mathrm{fst}\,(7,8)))) \mapsto (5, (6, (\lambda x.x)\,7))}$$

"search" for the exp that can step... and then find your way back...
Sounds kinda like a call stack...


"Abstract machine" - low-level model of comp.
No search rules - all step rules are axioms
Stack machine - model search as stack

Remember: $\underbrace{s}_{\text{state}} \mapsto s'$ Now a state has a stack

2 types of states:

$k \triangleright e$ "evaluate e"

$k \triangleleft v$ "return v to last stack frame"

$e ::= x \mid () \mid \lambda x.e \mid e\ e \mid (e,e) \mid \mathsf{fst}\ e \mid \mathsf{snd}\ e$

$\tau ::= \mathsf{unit} \mid \tau \to \tau \mid \tau \times \tau$

Frames $f ::= -\ e \mid e\ - \mid (-, e) \mid (e, -) \mid \mathsf{fst}\ - \mid \mathsf{snd}\ -$

Stack $k ::= \varepsilon \mid k ; f$

empty    stack ext. w/ f

State $s ::= k \rhd e \mid k \lhd v$

$(5, (6, (\lambda x.x)\ (\mathsf{fst}\ (7,8))))$

$(5, -); (6, -); (\lambda x.x)\ -; \mathsf{fst}\ - \rhd (7,8)$

or $(5, -); (6, -) \rhd (\lambda x.x)\ (\mathsf{fst}\ (7,8))$

or ...

$$\frac{}{k \rhd () \mapsto k \lhd ()}$$ No eval. to do on values

$$\frac{}{k \rhd \lambda x.e \mapsto k \lhd \lambda x.e}$$

$$\frac{}{k \rhd e_1\ e_2 \mapsto k; -\ e_2 \rhd e_1} \qquad \frac{}{k; -\ e_2 \lhd \lambda x.e_1 \mapsto k; \lambda x.e_1\ - \rhd e_2}$$

$$\frac{}{k; \lambda x.e_1\ - \lhd v \mapsto k \rhd [v/x]e_1}$$

$$\frac{}{k \rhd (e_1, e_2) \mapsto k; (-, e_2) \rhd e_1} \qquad \frac{}{k; (-, e_2) \lhd v_1 \mapsto k; (v_1, -) \rhd e_2}$$

$$\frac{}{k \rhd (v_1, -) \lhd v_2 \mapsto k \lhd (v_1, v_2)}$$

$$\frac{}{k \rhd \mathsf{fst}\ e \mapsto k; \mathsf{fst}\ - \rhd e} \qquad \frac{}{k; \mathsf{fst}\ - \lhd (v_1, v_2) \mapsto k \lhd v_1}$$

$\varepsilon \rhd (5, (6, (\lambda x.x)\ (\mathsf{fst}\ (7,8)))) \mapsto \varepsilon; (-, (6, \cdots)) \rhd 5 \mapsto \varepsilon; (-, (6, \cdots)) \lhd 5$

$\mapsto^* \ldots (5, -); (6, -); (\lambda x.x)\ -; \mathsf{fst}\ - \lhd (7,8)$

$\mapsto (5, -); (6, -); (\lambda x.x)\ - \lhd 7$

$\mapsto (5, -); (6, -) \lhd 7$

$\mapsto^* (5, (6, 7))$

Progress, Preservation? Need types for stacks

$f : \tau \rightsquigarrow \tau'$ f "takes $\tau$ to $\tau'$"
$k \triangleleft : \tau$ k "accepts a $\tau$"

$$\frac{}{\epsilon \triangleleft : \tau} \qquad \frac{k \triangleleft : \tau' \quad f : \tau \rightsquigarrow \tau'}{k; f \triangleleft : \tau} \qquad \frac{k \triangleleft : \tau \quad \cdot \vdash e : \tau}{k \triangleright e \text{ ok}} \qquad \frac{k \triangleleft : \tau \quad \cdot \vdash e : \tau \quad e \text{ val}}{k \triangleleft e \text{ ok}}$$

$$\frac{\cdot \vdash e_1 : \tau \to \tau' \quad e_1 \text{ val}}{e_1 \,-\, : \tau \rightsquigarrow \tau'} \qquad \frac{\cdot \vdash e_2 : \tau}{-\, e_2 : (\tau \to \tau') \rightsquigarrow \tau'}$$

$$\frac{\cdot \vdash e_2 : \tau_2}{(-, e_2) : \tau_1 \rightsquigarrow \tau_1 \times \tau_2} \qquad \frac{\cdot \vdash e_1 : \tau_1 \quad e_1 \text{ val}}{(e_1, -) : \tau_2 \rightsquigarrow (\tau_1 \times \tau_2)}$$

$$\frac{}{\text{fst} \,-\, : \tau_1 \times \tau_2 \rightsquigarrow \tau_1} \qquad \frac{}{\text{snd} \,-\, : \tau_1 \times \tau_2 \rightsquigarrow \tau_2}$$

Progress: If $s$ ok then $s = \epsilon \triangleleft v$ and $v$ val
or $s \mapsto s'$

Preservation: If $s$ ok and $s \mapsto s'$ then $s'$ ok

Prog:

1. $k \triangleright e$ ok. Then $k \triangleleft : \tau$ and $\cdot \vdash e : \tau$.
   Cont. by induction on structure of $e$
   $e = (), \lambda x.e, (v_1, v_2) \Rightarrow k \triangleright e \mapsto k \triangleleft e$
   $e = e_1\ e_2 \Rightarrow k \triangleright e \mapsto k; -\, e_2 \triangleright e_1 \ldots$
2. $k \triangleleft e$ ok. Then $k \triangleleft : \tau$ and $\cdot \vdash e : \tau$ and $e$ val
   Cont. by induction on $k \triangleleft : \tau$.
   a. $k = \epsilon$. Then done.
   b. $k = k_0; f$ and $k_0 \triangleleft : \tau'$ and $f : \tau \rightsquigarrow \tau'$.
      Cont. by induction on $f : \tau \rightsquigarrow \tau'$

• $f = e_1 -$ and $e_1 : \tau_1 \to \tau_2$ and $\tau = \tau_2$ and $e_1$ val.
By CF, $e_1 = \lambda x. e_0$. $k_0; f \triangleleft e \mapsto k_0 \triangleright [e/x] e_0$

• $f = - e_2$ and $e_2 : \tau_1$ and $\tau = \tau_1 \to \tau'$
By CF, $e = \lambda x. e_0$. $k_0; f \triangleleft e \mapsto k_0; \lambda x. e_0 - \triangleright e_2$

...

Preservation: By induction on the derivation of $s \mapsto s'$.

---

$k \triangleright e_1\ e_2 \mapsto k; -\ e_2 \triangleright e_1$
By inversion, $\bullet \vdash e_1\ e_2 : \tau$.
By inversion, $\bullet \vdash e_1 : \tau_1 \to \tau$ and $\bullet \vdash e_2 : \tau_1$
$-\ e_2 : (\tau_1 \to \tau) \to \tau$, so $k; -\ e_2 \triangleleft : \tau_1 \to \tau$ and $k; -\ e_2 \triangleright e_1$ ok

---

$k; \lambda x. e - \triangleleft v \mapsto k \triangleright [v/x] e$
By inversion, $\lambda x. e - : \tau \to \tau'$ and $k \triangleleft : \tau'$.
By inversion, $\bullet \vdash \lambda x. e : \tau \to \tau'$
By inversion, $x : \tau \vdash e : \tau'$
By subst., $\bullet \vdash [v/x] e : \tau'$
$k \triangleright [v/x] e$ ok