Dynamics

The Phases of PLs

Static Phase: Well-formedness of programs; e.g., parsing and type checking.

Dynamic Phase: Execution of programs.

Transition Systems

A *transitions systems* is specified by the following four forms of judgment:

1. s state, asserting that s is a *state* of the transition system.
2. s final, where s, state, asserting that s is a final state.
3. s inital, where s state, asserting that s is an initial state.
4. $s_1 \mapsto s_2$, where s_1 state and s_2 state, asserting that state s_1 may transition to the state s_2 .

Transition Systems

1. No transition can transition out of a final state.

2. A state where from which no transition is possible is called a *stuck state*.

3. Final states are stuck by definitions, but others may exist.

4. A transition system is *deterministic* iff for every state s, there exists exactly one transition $s \mapsto s'$ for any state s'. Otherwise, it is *non-deterministic*.

Transition Sequence

A sequence of states $s_0, \ldots, s_n$ such that s_0 initial, and $s_i \mapsto s_{i+1}$ for $0 \leq i < n$.

- *maximal* iff there is no s such that $s_n \mapsto s$.
- co*mplete* iff it is maximal and, s_n final
- $s \downarrow$ asserts that there is a complete sequence starting with s.

Transition Judgment

$$\frac{}{s \longmapsto^{*} s}\ \text{Refl}$$

$$\frac{s \longmapsto s' \qquad s' \longmapsto^{*} s''}{s \longmapsto^{*} s''}\ \text{Step}$$

Transition Judgment: Rule Induction

Suppose we want to show $P(s, s')$ whenever $s \mapsto s'$ holds. Then show:

1. $P(s, s)$
2. if $s \mapsto s'$ and $P(s', s'')$, then $P(s, s'')$ (head expansion)

Transition Judgment: n-times Iterated

$$\frac{}{s \mapsto^{0} s}\ \textsf{Refl} \qquad\qquad \frac{s \mapsto s' \quad s' \mapsto^{n} s''}{s \mapsto^{n+1} s''}\ \textsf{Step}$$

Transition Judgment: n-times Iterated

For all states s_1 and s_2, $s_1 \mapsto^* s_2$ iff $s_1 \mapsto^k s_2$ for some $k \geq 0$.

$$\frac{}{s \mapsto^0 s}\ \textsf{Refl} \qquad\qquad \frac{s \mapsto s' \quad s' \mapsto^n s''}{s \mapsto^{n+1} s''}\ \textsf{Step}$$

Structural Dynamics

Described by a transition system on abstract syntax:

1. States are closed expressions.
2. All states are initial.
3. Final states are closed values.

Structural Dynamics

$$\frac{}{\texttt{num}[n]\ \mathsf{val}}\ \text{numVal}$$

$$\frac{}{\texttt{str}[s]\ \mathsf{val}}\ \text{strVal}$$

$$\frac{n_1 + n_2 = n\ \mathsf{nat}}{\texttt{plus}(\texttt{num}[n_1];\texttt{num}[n_2]) \mapsto \texttt{num}[n]}\ \text{plusVal}$$

$$\frac{e_1 \mapsto e_1'}{\texttt{plus}(e_1;e_2) \mapsto \texttt{plus}(e_1';e_2)}\ \text{plus1}$$

$$\frac{e_1\ \mathsf{val} \quad e_2 \mapsto e_2'}{\texttt{plus}(e_1;e_2) \mapsto \texttt{plus}(e_1;e_2')}\ \text{plus2}$$

Structural Dynamics

$$\frac{s_1 \hat{\ } s_2 = s \ \mathsf{str}}{\mathtt{cat}(\mathtt{str}[s_1];\mathtt{str}[s_2]) \mapsto \mathtt{str}[s]} \ \text{catVal}$$

$$\frac{e_1 \mapsto e_1'}{\mathtt{cat}(e_1;e_2) \mapsto \mathtt{cat}(e_1';e_2)} \ \text{cat1}$$

$$\frac{e_1 \ \mathsf{val} \quad e_2 \mapsto e_2'}{\mathtt{cat}(e_1;e_2) \mapsto \mathtt{cat}(e_1;e_2')} \ \text{cat2}$$

Structural Dynamics

$$\frac{[e_1 \ \mathsf{val}]}{\mathtt{let}(e_1; x.e_2) \mapsto [e_1/x]e_2}\ \text{letVal}$$

$$\left[\frac{e_1 \mapsto e_1'}{\mathtt{let}(e_1; x.e_2) \mapsto \mathtt{let}(e_1'; x.e_2)}\right]\text{let}$$

Structural Dynamics: Example

$$\mathsf{let}(\mathsf{str}[\text{"C"}], s_1 \,.\, \mathsf{let}(\mathsf{str}[\text{"S"}], s_2 \,.\, \mathsf{len}(\mathsf{cat}(s_1, s_2)) \mapsto \mathsf{num}[2]$$

Structural Dynamics: Example

$$\mathsf{let}(\mathsf{str}[\text{"C"}], s_1 \,.\, \mathsf{let}(\mathsf{str}[\text{"S"}], s_2 \,.\, \mathsf{len}(\mathsf{cat}(s_1, s_2))) \mapsto^{\mathsf{LetVal}} \mathsf{let}(\mathsf{str}[\text{"S"}], s_2 \,.\, \mathsf{len}(\mathsf{cat}(\mathsf{str}[\text{"C"}], s_2))$$

$$\mapsto^{\mathsf{LetVal}} \mathsf{len}(\mathsf{cat}(\mathsf{str}[\text{"C"}], \mathsf{str}[\text{"S"}])$$

$$\mapsto^{\mathsf{Len1,CatVal}} \mathsf{len}(\mathsf{str}[\text{"CS"}])$$

$$\mapsto^{\mathsf{LenVal}} \mathsf{num}[2]$$

Structural Dynamics

Lemma. If $e_1 \mapsto e_2$ and $e_1 \mapsto e_3$, then $e_2 \equiv_\alpha e_3$