Orc in ACL2

Chad Wellington
9/05/2007

Orc, in general

- Assumptions
  - Distributed
  - “Workflow”
  - “Formal” Specification

- Composition
  - Sites
  - Combinators

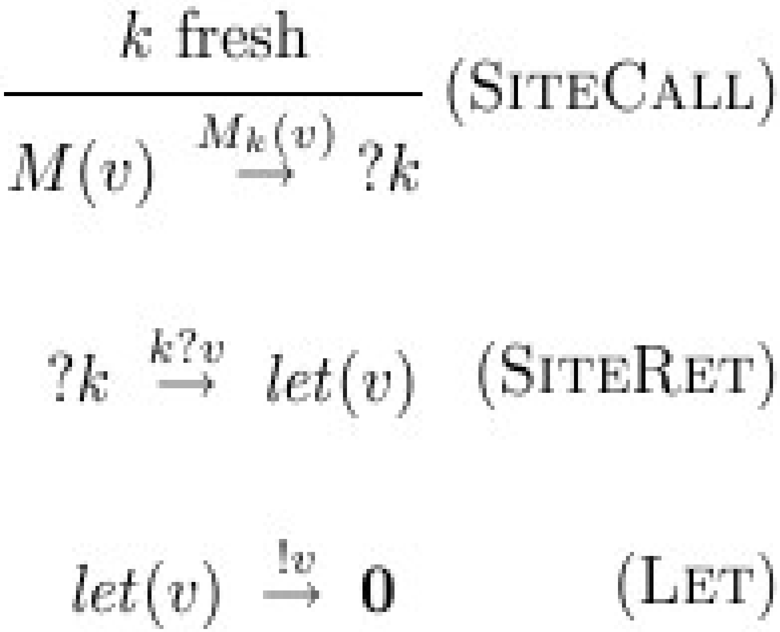
Intuition, briefly

- Sites a form of Remote Procedure Call
  - publishes 0 or 1 times
- Connectives
  - bar : “f | g”
    - simultaneous, non-communicating parallel
  - pipe : “f >x> g”
    - sequential creation of g's on f's publications
    - analogy to dataflow, Unix pipe
  - where : “f where x<-g” or more recently “f<x<g”
    - parallel execution with binding and termination
    - fork-join parallelism

Operational Semantics

- Formalize program meaning
  - analogy to eval (?)
  - as transitions, not final result
- State transition definitions
  - conditional clauses (like Horn clauses)
  - can form chain (or tree)
  - here, labeled

Site definition



$$\frac{k \text{ fresh}}{M(v) \xrightarrow{M_k(v)} ?k} \quad (\textsc{SiteCall})$$

$$?k \xrightarrow{k?v} let(v) \quad (\textsc{SiteRet})$$

$$let(v) \xrightarrow{!v} \mathbf{0} \quad (\textsc{Let})$$

- actuals
  - x :: unbound vbl
  - v :: bound vbl / value
- Sites
  - M(x) :: blocked
  - M(v) :: unblocked
  - k :: unique handle
  - 0 :: silent
  - let(x) :: publication

Life cycle of a site

M(x) ->* M(v) -> ?k -> let(v) -> 0

Publication rules

$$\frac{f \xrightarrow{!v} f'}{f >x> g \xrightarrow{\tau} (f' >x> g) \,|\, [v/x].g} \qquad (\textsc{Seq1V})$$

$$\frac{g \xrightarrow{!v} g'}{f \ \mathbf{where}\ x :\in g \xrightarrow{\tau} [v/x].f} \qquad (\textsc{Asym1V})$$

- “tau” is an internal event
- [v/x].g = all free occurrences of x in g replaced by v (relaxed to all occurrences since we can rename variables)

Transition passing rules

$$\frac{f \xrightarrow{a} f'}{f \mid g \xrightarrow{a} f' \mid g} \quad (\textsc{Sym1})$$

$$\frac{f \xrightarrow{a} f'}{f \text{ where } x :\in g \xrightarrow{a} f' \text{ where } x :\in g} \quad (\textsc{Asym1N})$$

$$\frac{g \xrightarrow{a} g'}{f \mid g \xrightarrow{a} f \mid g'} \quad (\textsc{Sym2})$$

$$\frac{g \xrightarrow{a} g' \qquad a \neq !v}{f \text{ where } x :\in g \xrightarrow{a} f \text{ where } x :\in g'} \quad (\textsc{Asym2})$$

$$\frac{f \xrightarrow{a} f' \qquad a \neq !v}{f > x > g \xrightarrow{a} f' > x > g} \quad (\textsc{Seq1N})$$

- Upper half pass any transition
- Lower half exempt publications

&c.

- structure forces a sequence, not a tree
  - identifiable sequence of steps = “execution”
  - eliminate tau's = “trace”
    - note origin of tau's
- I make convenient (necessary?) assumptions
  - round-based execution
  - currently model semantic steps of tau transitions
    - others execute in separate segment of the round, need different treatment
- Code on web, documentation available but shaky