Semantics of Orc

William Cook and Jayadev Misra

Department of Computer Science
University of Texas at Austin

Email: {cook,misra}@cs.utexas.edu
web: http://www.cs.utexas.edu/users/psp

Outline

- Define an asynchronous semantics, using labeled transitions.

  An expression transits to another expression, causing an event.

  Labels are events.

- Refine asynchronous semantics to a synchronous semantics.

Formal Syntax

f, g, h	$\in$	$Expr$	::=	$M(P)$	Site call
				$\| \; E(P)$	Expression call
				$\| \; f \; >x> \; g$	Sequential composition
				$\| \; f \mid g$	Symmetric composition
				$\| \; f$ where $x :\in g$	Asymmetric composition
p	$\in$	$Actual$	::=	$x \; \| \; c \; \| \; M$	
q	$\in$	$Formal$	::=	$x \; \| \; M$	

Enhanced Syntax

Add $?u$ and $let(c)$ as two possible expressions.

f, g, h	$\in$	$Expr$	::=	$M(P)$	Site call
				$\parallel E(P)$	Call to definition
				$\parallel f \ {>}x{>} \ g$	Sequential composition
				$\parallel f \mid g$	Symmetric composition
				$\parallel f \ \text{where} \ x{:}{\in} \ g$	Asymmetric composition
				$\parallel \ ?u$	Waiting for response
				$\parallel let(c)$	Ready to Publish

Events

There are 4 kinds of events.

$l \in Event$	::=	$M\langle c, u\rangle$	Site call with handle u
		$\parallel u?c$	Response
		$\parallel \dagger c$	publish
		$\parallel \tau$	silent transition

Response is outside the control of Orc.

Rules for Site Call

$$\frac{u \text{ fresh}}{M(c) \overset{M\langle c,u\rangle}{\hookrightarrow} ?u} \qquad \text{(SITECALL)}$$

$$?u \overset{u?c}{\hookrightarrow} let(c) \qquad \text{(SITERET)}$$

$$let(c) \overset{\dagger c}{\hookrightarrow} 0 \qquad \text{(LET)}$$

Symmetric Composition

$$\frac{f \overset{l}{\hookrightarrow} f'}{f \mid g \overset{l}{\hookrightarrow} f' \mid g} \quad \text{(SYM1)}$$

$$\frac{g \overset{l}{\hookrightarrow} g'}{f \mid g \overset{l}{\hookrightarrow} f \mid g'} \quad \text{(SYM2)}$$

Sequencing

$$\frac{f \overset{l}{\hookrightarrow} f' \qquad l \neq \dagger c}{f \;{>}x{>}\; g \overset{l}{\hookrightarrow} f' \;{>}x{>}\; g} \qquad \text{(SEQ1N)}$$

$$\frac{f \overset{\dagger c}{\hookrightarrow} f'}{f \;{>}x{>}\; g \overset{\tau}{\hookrightarrow} (f' \;{>}x{>}\; g) \mid [c/x]g} \qquad \text{(SEQ1V)}$$

Asymmetric Composition

$$\frac{f \overset{l}{\hookrightarrow} f' \qquad l \neq \dagger c}{g \text{ where } x{:}\in f \overset{l}{\hookrightarrow} g \text{ where } x{:}\in f'} \qquad \text{(Asym1N)}$$

$$\frac{f \overset{\dagger c}{\hookrightarrow} f'}{g \text{ where } x{:}\in f \overset{\tau}{\hookrightarrow} [c/x]g} \qquad \text{(Asym1V)}$$

$$\frac{g \overset{l}{\hookrightarrow} g'}{g \text{ where } x{:}\in f \overset{l}{\hookrightarrow} g' \text{ where } x{:}\in f} \qquad \text{(Asym2)}$$

Expression Call

$$\frac{[[\, E(q) \;\; \underline{\Delta} \;\; f \,]] \in D}{E(p) \overset{\tau}{\hookrightarrow} [p/q]f} \qquad \text{(Def)}$$

Rules

$$\frac{u \text{ fresh}}{M(c) \overset{M\langle c,u\rangle}{\hookrightarrow} ?u}$$

$$?u \overset{u?c}{\hookrightarrow} let(c)$$

$$let(c) \overset{\dagger c}{\hookrightarrow} 0$$

$$\frac{f \overset{l}{\hookrightarrow} f'}{f \mid g \overset{l}{\hookrightarrow} f' \mid g}$$

$$\frac{g \overset{l}{\hookrightarrow} g'}{f \mid g \overset{l}{\hookrightarrow} f \mid g'}$$

$$\frac{[[\, E(q) \;\; \underline{\Delta} \;\; f \,]] \in D}{E(p) \overset{\tau}{\hookrightarrow} [p/q]f}$$

$$\frac{f \overset{l}{\hookrightarrow} f' \qquad l \neq \dagger c}{f \;>x>\; g \overset{l}{\hookrightarrow} f' \;>x>\; g}$$

$$\frac{f \overset{\dagger c}{\hookrightarrow} f'}{f \;>x>\; g \overset{\tau}{\hookrightarrow} (f' \;>x>\; g) \mid [c/x]g}$$

$$\frac{f \overset{l}{\hookrightarrow} f' \qquad l \neq \dagger c}{g \text{ where } x{:}{\in}\, f \overset{l}{\hookrightarrow} g \text{ where } x{:}{\in}\, f'}$$

$$\frac{f \overset{\dagger c}{\hookrightarrow} f'}{g \text{ where } x{:}{\in}\, f \overset{\tau}{\hookrightarrow} [c/x]g}$$

$$\frac{g \overset{l}{\hookrightarrow} g'}{g \text{ where } x{:}{\in}\, f \overset{l}{\hookrightarrow} g' \text{ where } x{:}{\in}\, f}$$

Pending Event

At any moment, there is a set of pending events.

Processing a pending event may

- transform the expression,
- change the set of pending events.

Example of Event Processing

In $M(x)$ where $x :\in N \mid R$, both $N\langle u\rangle$ and $R\langle v\rangle$ are pending.

$$M(x) \text{ where } x :\in N \mid R \overset{N\langle u\rangle}{\hookrightarrow} M(x) \text{ where } x :\in ?u \mid R$$

$R\langle v\rangle$ is still pending. So are $u?x$, for all x.

After

$$M(x) \text{ where } x :\in ?u \mid R \overset{u?c}{\hookrightarrow} M(c)$$

$R\langle v\rangle$ and $u?x$ are no longer pending.

Rules for Event Processing

- (Fairness) If there is an internal pending event, some pending event is processed eventually.
- (Asynchrony) Order and timing of event processings are arbitrary.

Notes on Event Processing

- Fairness is minimal progress. It does not say that:

  an event which remains pending is eventually processed.

- Only internal events are under client's control.

- If there are only external pending events, no event may be processed.

Examples

- $let(x)$ where $x{:}{\in}\ let(0)\ |\ Rtimer(1) \gg R$

  If it publishes 0, R's response, if any, is never fully processed.

- (Fairness) $Metronome \gg (let(0)\ |\ let(1))$

  May publish 0 forever.

- (Asynchrony) $let(x)$ where $x{:}{\in}\ let(0)\ |\ Rtimer(1) \gg let(1)$

  may publish 0 or 1.

Synchronous Semantics

Specify time (and order) of event processing.

Internal event (action):

$l \in Action$	::=	$M\langle c, u\rangle$	Site call with handle u
		$\parallel \dagger c$	publish; $let(c)$ is internal
		$\parallel \tau$	silent transition

External Event (response): $u?c$

Rule 1: Process a response only if there is no action.

Order among internal events is arbitrary.

Order among external responses is arbitrary.

Examples

$$let(x) \text{ where } x :\in let(0) \mid Rtimer(1) \gg let(1) \quad (1)$$

$$let(x) \text{ where } x :\in let(0) \mid let(2) \mid Rtimer(1) \gg let(1) \quad (2)$$

$$let(x) \text{ where } x :\in if(\textsf{true}) \gg let(0) \mid Rtimer(1) \gg let(1) \quad (3)$$

$$let(x) \text{ where } x :\in Rtimer(1) \gg let(0) \mid Rtimer(2) \gg let(1) \quad (4)$$

(1) publishes 0.

(2) publishes 0 or 2.

(3) publishes 0 or 1.

(4) publishes 0 or 1.

Handle Time

Rule 2: Process events as soon as possible.

Assume event processing is instantaneous.

$$let(x) \text{ where } x :\in Rtimer(1) \gg let(0) \mid Rtimer(2) \gg let(1) \quad (4)$$

publishes 0.

Immediate/deferred Sites

Designate certain sites as immediate, rest as deferred.

An immediate site has to respond instantaneously.

Immediate Sites: let, if, add, or, $\geq \cdots$

Deferred Sites: $Rtimer$, $CNN \cdots$

Rule 3: A response from an immediate site is an internal event.

$$let(x) \text{ where } x :\in if(\mathsf{true}) \gg let(0) \mid Rtimer(1) \gg let(1) \qquad (3)$$

publishes 0.

Positive/Negative Response

An immediate site responds immediately with

- positive response: a result value, written as $u?c$, or
- negative response: that it will be silent, written as $u??$

$$?u \overset{u??}{\hookrightarrow} 0 \qquad \text{(SiteRet)}$$

Summary of Rules for Synchronous Semantics

- Process a response only if there is no action.
- Process events as soon as possible.
- A response from an immediate site is an internal event.

Formal definition of Synchronous Semantics

- Define quiescent expression, $QExpr$, one that has no internal event.
- Define non-quiescent expression, $NExpr$, its complement.

$$
\begin{array}{lclcl}
\hookrightarrow & : & Expr \times Event \times Expr & & \{\text{Defined earlier}\} \\
\overset{r}{\hookrightarrow}_{\mathsf{R}} & : & QExpr \times Response \times Expr & = & \{(q, r, e) \mid q \overset{r}{\hookrightarrow} e\} \\
\overset{a}{\hookrightarrow}_{\mathsf{A}} & : & NExpr \times Action \times Expr & = & \{(\hat{q}, a, e) \mid \hat{q} \overset{a}{\hookrightarrow} e\} \\
\overset{l}{\hookrightarrow}_{\mathsf{S}} & : & Expr \times Event \times Expr & = & \overset{l}{\hookrightarrow}_{\mathsf{R}} \cup \overset{l}{\hookrightarrow}_{\mathsf{A}}
\end{array}
$$

Round-based Execution

- A round consists of processing internal events. This includes calls to and responses from immediate sites.
- A round ends when no more internal events can be processed.
- First round starts at the beginning of evaluation.
- Subsequent rounds start by processing a response from a deferred site.

Laws of Kleene Algebra

(Zero and $\mid$)	$f \mid 0 = f$
(Commutativity of $\mid$)	$f \mid g = g \mid f$
(Associativity of $\mid$)	$(f \mid g) \mid h = f \mid (g \mid h)$
(Idempotence of $\mid$)	$f \mid f = f$
(Associativity of $\gg$)	$(f \gg g) \gg h = f \gg (g \gg h)$
(Left zero of $\gg$)	$0 \gg f = 0$
(Right zero of $\gg$)	$f \gg 0 = 0$
(Left unit of $\gg$)	$Signal \gg f = f$
(Right unit of $\gg$)	$f \ {>}x{>} \ let(x) = f$
(Left Distributivity of $\gg$ over $\mid$)	$f \gg (g \mid h) = (f \gg g) \mid (f \gg h)$
(Right Distributivity of $\gg$ over $\mid$)	$(f \mid g) \gg h = (f \gg h \mid g \gg h)$

Laws which do not hold

(Idempotence of $\mid$)	$f \mid f = f$
(Right zero of $\gg$)	$f \gg 0 = 0$
(Left Distributivity of $\gg$ over $\mid$)	$f \gg (g \mid h) = (f \gg g) \mid (f \gg h)$

Additional Laws

(Distributivity over $\gg$) if g is x-free

$$(f \gg g \text{ where } x{:\in}\ h) = (f \text{ where } x{:\in}\ h) \gg g$$

(Distributivity over $|$) if g is x-free

$$(f \mid g \text{ where } x{:\in}\ h) = (f \text{ where } x{:\in}\ h) \mid g$$

(Distributivity over where) if g is y-free

$$((f \text{ where } x{:\in}\ g) \text{ where } y{:\in}\ h)$$
$$= \quad ((f \text{ where } y{:\in}\ h) \text{ where } x{:\in}\ g)$$

(Elimination of where) if f is x-free, for site M

$$(f \text{ where } x{:\in}\ M) = f \mid M \gg 0$$

Silent Expression

g is silent if it never publishes: $g = g \gg 0$.

$f \backslash x$: In f replace site calls which have x as a parameter by 0.

Law: If g is silent, then $(f \text{ where } x{:}{\in}\ g) = (f \backslash x \mid g)$

Exercise:: Explore identities about silent expressions.

Proofs

- Direct proofs from the asynchronous semantics of:
  - Zero and $|$
  - Commutativity of $|$
  - Left zero of $\gg$
- Others: Bisimulations using safe functions and parallel composition contexts (see Sangiorgi and Walker).

Proofs employ asynchronous semantics. No proof yet using the synchronous semantics.

References

Operational Semantics + Bisimulation; see (with William Cook)
http://www.cs.utexas.edu/users/wcook/projects/orc/papers/OrcCookMisra05.pdf

A Denotational Semantics; see (with Tony Hoare and Galen Menzel)
http://www.cs.utexas.edu/users/psp/Semantics.Orc.pdf

A tutorial on the model with longer examples; see (with William Cook)
http://www.cs.utexas.edu/users/wcook/papers/OrcJSSM05/OrcJSSM.pdf