Communication and Concurrency
Lecture 1

Colin Stirling (cps)

School of Informatics

September 19th 2013

First example

A clock that perpetually ticks

$$\texttt{Cl} \stackrel{\text{def}}{=} \texttt{tick.Cl}$$

- `tick` action name
- `Cl` process name
- $\stackrel{\text{def}}{=}$ ties a process name to a process expression
- `tick.Cl` process expression
- . prefix operator

Behaviour: transitions

Behaviour of processes is captured by transitions

$$E \xrightarrow{a} F$$

Goal-directed rules for deriving transitions

- axiom (.)

$$\mathrm{R}(.) \quad a.E \xrightarrow{a} E$$

Behaviour: transitions

Behaviour of processes is captured by transitions

$$E \xrightarrow{a} F$$

Goal-directed rules for deriving transitions

- axiom (.)

$$\mathrm{R}(.) \quad a.E \xrightarrow{a} E$$

- $\stackrel{\text{def}}{=}$

$$\mathrm{R}(\stackrel{\text{def}}{=}) \quad \frac{P \xrightarrow{a} F}{E \xrightarrow{a} F} \; P \stackrel{\text{def}}{=} E$$

Behaviour: transitions

Behaviour of processes is captured by transitions

$$E \xrightarrow{a} F$$

Goal-directed rules for deriving transitions

- axiom (.)

$$\mathrm{R}(.) \quad a.E \xrightarrow{a} E$$

- $\stackrel{\text{def}}{=}$

$$\mathrm{R}(\stackrel{\text{def}}{=}) \quad \frac{P \xrightarrow{a} F}{E \xrightarrow{a} F} \; P \stackrel{\text{def}}{=} E$$

Example

$$\mathtt{Cl} \xrightarrow{\mathtt{tick}} \mathtt{Cl}$$

Behaviour: transition graphs

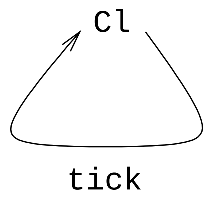


Figure: The transition graph for Cl

Behaviour: transition graphs

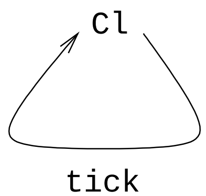


Figure: The transition graph for Cl

Labelled graph

- vertices: process expressions
- labelled edges: transitions
- Each derivable transition of a vertex is depicted
- Abstract from the derivations of transitions

Interlude: exercise

Draw the transition graphs for the following clocks

1. $\mathtt{Cl}_1 \stackrel{\text{def}}{=} \mathtt{tick.tock.Cl}_1$

Interlude: exercise

Draw the transition graphs for the following clocks

1. $\mathtt{Cl}_1 \stackrel{\text{def}}{=} \mathtt{tick.tock.Cl}_1$
2. $\mathtt{Cl}_2 \stackrel{\text{def}}{=} \mathtt{tick.tick.Cl}_2$

Interlude: exercise

Draw the transition graphs for the following clocks

1. $\mathtt{Cl}_1 \stackrel{\text{def}}{=} \mathtt{tick.tock.Cl}_1$
2. $\mathtt{Cl}_2 \stackrel{\text{def}}{=} \mathtt{tick.tick.Cl}_2$
3. $\mathtt{Cl}_3 \stackrel{\text{def}}{=} \mathtt{tick.Cl}$

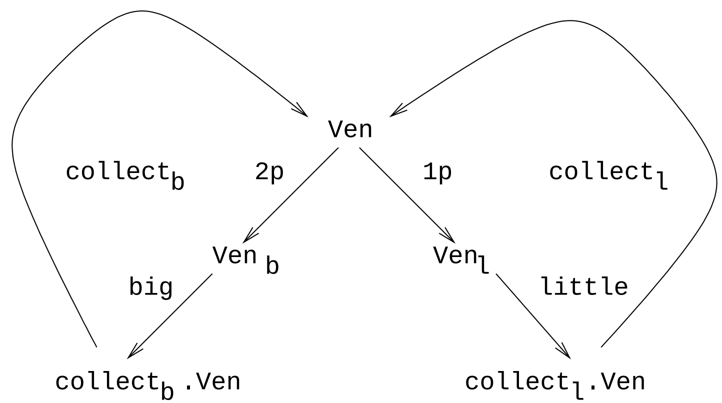
The + operator

$$
\begin{aligned}
\mathtt{Ven} &\stackrel{\text{def}}{=} \mathtt{2p.Ven_b} + \mathtt{1p.Ven_l} \\
\mathtt{Ven_b} &\stackrel{\text{def}}{=} \mathtt{big.collect_b.Ven} \\
\mathtt{Ven_l} &\stackrel{\text{def}}{=} \mathtt{little.collect_l.Ven}
\end{aligned}
$$

Transition Rule

$$
\mathrm{R}(+) \quad \frac{E_1 + E_2 \stackrel{a}{\longrightarrow} F}{E_1 \stackrel{a}{\longrightarrow} F} \qquad \frac{E_1 + E_2 \stackrel{a}{\longrightarrow} F}{E_2 \stackrel{a}{\longrightarrow} F}
$$

Transition Graph

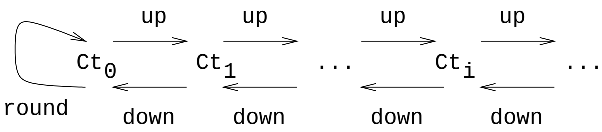


Generalising: indexed definitions

$$\texttt{Ct}_0 \stackrel{\text{def}}{=} \texttt{up}.\texttt{Ct}_1 + \texttt{round}.\texttt{Ct}_0$$

$$\texttt{Ct}_{\texttt{i+1}} \stackrel{\text{def}}{=} \texttt{up}.\texttt{Ct}_{\texttt{i+2}} + \texttt{down}.\texttt{Ct}_{\texttt{i}}$$

$$\frac{\dfrac{\texttt{Ct}_3 \xrightarrow{\texttt{up}} \texttt{Ct}_4}{\texttt{up}.\texttt{Ct}_4 + \texttt{down}.\texttt{Ct}_2 \xrightarrow{\texttt{up}} \texttt{Ct}_4}}{\texttt{up}.\texttt{Ct}_4 \xrightarrow{\texttt{up}} \texttt{Ct}_4}$$



Generalising: indexed sums

$\sum\{E_i \,:\, i \in I\}$ I indexing set
($E_1 + E_2$ abbreviates $\sum\{E_i \,:\, i \in \{1,2\}\}$)

Generalising: indexed sums

$\sum\{E_i \,:\, i \in I\}$ I indexing set
($E_1 + E_2$ abbreviates $\sum\{E_i \,:\, i \in \{1,2\}\}$)

$$\texttt{Reg}'_{\texttt{i}} \stackrel{\text{def}}{=} \texttt{read}_{\texttt{i}}.\texttt{Reg}'_{\texttt{i}} + \sum\{\texttt{write}_j.\texttt{Reg}'_j \,:\, j \in \mathbb{N}\}$$

Generalising: indexed sums

$\sum\{E_i \,:\, i \in I\}$ I indexing set
($E_1 + E_2$ abbreviates $\sum\{E_i \,:\, i \in \{1,2\}\}$)

$$\texttt{Reg}'_{\texttt{i}} \stackrel{\text{def}}{=} \texttt{read}_{\texttt{i}}.\texttt{Reg}'_{\texttt{i}} + \sum\{\texttt{write}_j.\texttt{Reg}'_j \,:\, j \in \mathbb{N}\}$$

Transition Rule for Σ

$$\mathrm{R}(\textstyle\sum) \quad \frac{\sum\{E_i \,:\, i \in I\} \xrightarrow{a} F}{E_j \xrightarrow{a} F} \; j \in I$$

Generalising: indexed sums

$\sum\{E_i : i \in I\}$ I indexing set
($E_1 + E_2$ abbreviates $\sum\{E_i : i \in \{1,2\}\}$)

$$\mathtt{Reg}'_{\mathtt{i}} \stackrel{\mathrm{def}}{=} \mathtt{read}_{\mathtt{i}}.\mathtt{Reg}'_{\mathtt{i}} + \sum\{\mathtt{write}_j.\mathtt{Reg}'_j : j \in \mathbb{N}\}$$

Transition Rule for Σ

$$\mathrm{R}(\textstyle\sum) \; \frac{\sum\{E_i : i \in I\} \xrightarrow{a} F}{E_j \xrightarrow{a} F} \; j \in I$$

Special Case $\sum\{E_i : i \in \emptyset\}$ abbreviated to 0 "nil"

Generalising: parameterized actions

- input of data at port a, $a(x).E$
  $a(x)$ binds free occurrences of x in E. Port a represents $\{a(v) : v \in D\}$ where D is a family of data values

Generalising: parameterized actions

- input of data at port a, $a(x).E$
  $a(x)$ binds free occurrences of x in E. Port a represents $\{a(v) : v \in D\}$ where D is a family of data values
- output of data at port a, $\overline{a}(e).E$
  where e is a data expression.

Generalising: parameterized actions

- input of data at port a, $a(x).E$
  $a(x)$ binds free occurrences of x in E. Port a represents $\{a(v) : v \in D\}$ where D is a family of data values
- output of data at port a, $\overline{a}(e).E$
  where e is a data expression.
- Transition Rules: depend on extra machinery for expression evaluation. Let $\mathrm{Val}(e)$ be data value in D (if there is one) to which e evaluates

Generalising: parameterized actions

- input of data at port a, $a(x).E$
  $a(x)$ binds free occurrences of x in E. Port a represents $\{a(v) : v \in D\}$ where D is a family of data values
- output of data at port a, $\overline{a}(e).E$
  where e is a data expression.
- Transition Rules: depend on extra machinery for expression evaluation. Let Val(e) be data value in D (if there is one) to which e evaluates
- $\mathrm{R(in)} \quad a(x).E \xrightarrow{a(v)} E\{v/x\} \quad \text{if} \quad v \in D$
  where $\{v/x\}$ is substitution

Generalising: parameterized actions

- input of data at port a, $a(x).E$
  $a(x)$ binds free occurrences of x in E. Port a represents $\{a(v) : v \in D\}$ where D is a family of data values
- output of data at port a, $\overline{a}(e).E$
  where e is a data expression.
- Transition Rules: depend on extra machinery for expression evaluation. Let Val(e) be data value in D (if there is one) to which e evaluates
- $\mathrm{R(in)} \quad a(x).E \xrightarrow{a(v)} E\{v/x\} \quad \text{if} \quad v \in D$
  where $\{v/x\}$ is substitution
- $\mathrm{R(out)} \quad \overline{a}(e).E \xrightarrow{\overline{a}(v)} E \quad \text{if } \mathrm{Val}(e) = v$

Examples

$\mathrm{R(in)} \quad a(x).E \xrightarrow{a(v)} E\{v/x\} \quad \text{if} \quad v \in D$

$\mathrm{R(out)} \quad \overline{a}(e).E \xrightarrow{\overline{a}(v)} E \quad \text{if } \mathrm{Val}(e) = v$

Examples

$\mathrm{R(in)} \quad a(x).E \xrightarrow{a(v)} E\{v/x\} \quad \text{if} \quad v \in D$

$\mathrm{R(out)} \quad \overline{a}(e).E \xrightarrow{\overline{a}(v)} E \quad \text{if } \mathrm{Val}(e) = v$

A Copier: $\mathtt{Cop} \stackrel{\mathrm{def}}{=} \mathtt{in}(x).\overline{\mathtt{out}}(x).\mathtt{Cop}$

$$\frac{\mathtt{Cop} \xrightarrow{\mathtt{in}(v)} \overline{\mathtt{out}}(v).\mathtt{Cop}}{\mathtt{in}(x).\overline{\mathtt{out}}(x).\mathtt{Cop} \xrightarrow{\mathtt{in}(v)} \overline{\mathtt{out}}(v).\mathtt{Cop}}$$

Examples

R(in) $a(x).E \xrightarrow{a(v)} E\{v/x\}$ if $v \in D$

R(out) $\overline{a}(e).E \xrightarrow{\overline{a}(v)} E$ if $\mathrm{Val}(e) = v$

A Copier: $\texttt{Cop} \stackrel{\mathrm{def}}{=} \texttt{in}(x).\overline{\texttt{out}}(x).\texttt{Cop}$

$$\frac{\texttt{Cop} \xrightarrow{\texttt{in}(v)} \overline{\texttt{out}}(v).\texttt{Cop}}{\texttt{in}(x).\overline{\texttt{out}}(x).\texttt{Cop} \xrightarrow{\texttt{in}(v)} \overline{\texttt{out}}(v).\texttt{Cop}}$$

A Register: $\texttt{Reg}_{\texttt{i}} \stackrel{\mathrm{def}}{=} \overline{\texttt{read}}(i).\texttt{Reg}_{\texttt{i}} + \texttt{write}(x).\texttt{Reg}_x$

$$\frac{\frac{\texttt{Reg}_5 \xrightarrow{\texttt{write}(3)} \texttt{Reg}_3}{\overline{\texttt{read}}(5).\texttt{Reg}_5 + \texttt{write}(x).\texttt{Reg}_x \xrightarrow{\texttt{write}(3)} \texttt{Reg}_3}}{\texttt{write}(x).\texttt{Reg}_x \xrightarrow{\texttt{write}(3)} \texttt{Reg}_3}$$

Exercise

Assume that the space of values consists of two elements, 0 and 1.
Draw transition graphs for the following three copiers

1. $\texttt{Cop} \stackrel{\mathrm{def}}{=} \texttt{in}(x).\overline{\texttt{out}}(x).\texttt{Cop}$

Exercise

Assume that the space of values consists of two elements, 0 and 1.
Draw transition graphs for the following three copiers

1. $\texttt{Cop} \stackrel{\mathrm{def}}{=} \texttt{in}(x).\overline{\texttt{out}}(x).\texttt{Cop}$
2. $\texttt{Cop}_1 \stackrel{\mathrm{def}}{=} \texttt{in}(x).\texttt{in}(x).\overline{\texttt{out}}(x).\texttt{Cop}_1$

Exercise

Assume that the space of values consists of two elements, 0 and 1.
Draw transition graphs for the following three copiers

1. $\texttt{Cop} \stackrel{\mathrm{def}}{=} \texttt{in}(x).\overline{\texttt{out}}(x).\texttt{Cop}$
2. $\texttt{Cop}_1 \stackrel{\mathrm{def}}{=} \texttt{in}(x).\texttt{in}(x).\overline{\texttt{out}}(x).\texttt{Cop}_1$
3. $\texttt{Cop}_2 \stackrel{\mathrm{def}}{=} \texttt{in}(x).\overline{\texttt{out}}(x).\overline{\texttt{out}}(x).\texttt{Cop}_2$

Summary

- Introduction of process expressions, process combinators

Summary

- Introduction of process expressions, process combinators
- Derivation of transitions between expressions

Summary

- Introduction of process expressions, process combinators
- Derivation of transitions between expressions
- Abstraction of derivations into transition graphs

Summary

- Introduction of process expressions, process combinators
- Derivation of transitions between expressions
- Abstraction of derivations into transition graphs
- Background Reading: Chapter 1 of
  R. Milner, *Communication and Concurrency*, Prentice-Hall