Topics in Concurrency

Lecture 2

Jonathan Hayman

18 January 2016

The Calculus of Communicating Systems

- Introduced by Robin Milner in 1980
- First process calculus developed with its operational semantics
- Supports algebraic reasoning about equivalence
- Simplifies Dijkstra's Guarded Command Language by removing the store (store locations can be encoded as processes)
- Processes communicate by sending values (numbers) on channels.

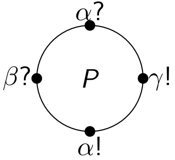
Action and communication

- As before, communication is synchronous and between two processes.
- Input: $\alpha?x$
- Output: $\alpha!a$
- Silent action: τ. Internal to the process.
- We will use λ to range over all the kinds of action.

Processes will structure actions

Interface diagrams

- *Interface diagrams* describe the channels used by processes for input and output.
- The use of a channel by a process is called a *port*.
- Example: process P inputs on α, β and outputs on α, γ.



- Later examples: links between processes to represent the possibility of communication

Syntax of CCS

- Expressions: Arithmetic a and Boolean b
- Processes:

$$
\begin{array}{lcll}
p & ::= & \mathbf{nil} & \text{nil process} \\
 & \mid & (\tau \to p) & \text{silent/internal action} \\
 & \mid & (\alpha!a \to p) & \text{output} \\
 & \mid & (\alpha?x \to p) & \text{input} \\
 & \mid & (b \to p) & \text{Boolean guard} \\
 & \mid & p_0 + p_1 & \text{non-deterministic choice} \\
 & \mid & p_0 \parallel p_1 & \text{parallel composition} \\
 & \mid & p \backslash L & \text{restriction } (L \text{ a set of channel identifiers}) \\
 & \mid & p[f] & \text{relabelling } (f \text{ a function on channel identifiers}) \\
 & \mid & P(a_1, \cdots, a_k) & \text{process identifier}
\end{array}
$$

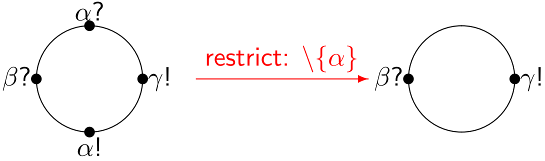
- Process definitions:

$$P(x_1, \cdots, x_k) \stackrel{\text{def}}{=} p$$

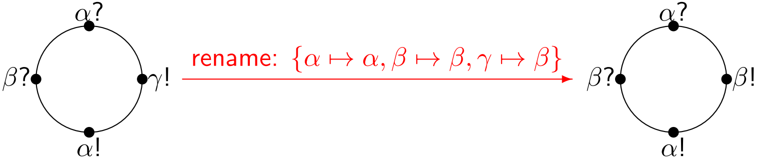
(free variables of $p \subseteq \{x_1, \cdots, x_n\}$)

Restriction and relabelling: interface diagrams

- $p \setminus L$: Disallow external interaction on channels in L



- $p[f]$: Rename external interface to channels by f



Operational semantics of CCS

- **Guarded processes**

$$(\tau \to p) \xrightarrow{\tau} p$$

$$\frac{a \to n}{(\alpha!a \to p) \xrightarrow{\alpha!n} p} \qquad (\alpha?x \to p) \xrightarrow{\alpha?n} p[n/x]$$

$$\frac{b \to \mathrm{true} \qquad p \xrightarrow{\lambda} p'}{(b \to p) \xrightarrow{\lambda} p'}$$

- **Sum**

$$\frac{p_0 \xrightarrow{\lambda} p_0'}{p_0 + p_1 \xrightarrow{\lambda} p_0'} \qquad \frac{p_1 \xrightarrow{\lambda} p_1'}{p_0 + p_1 \xrightarrow{\lambda} p_1'}$$

- **Parallel composition**

$$\frac{p_0 \xrightarrow{\lambda} p_0'}{p_0 \parallel p_1 \xrightarrow{\lambda} p_0' \parallel p_1} \qquad \frac{p_0 \xrightarrow{\alpha?n} p_0' \qquad p_1 \xrightarrow{\alpha!n} p_1'}{p_0 \parallel p_1 \xrightarrow{\tau} p_0' \parallel p_1'}$$

$$\frac{p_1 \xrightarrow{\lambda} p_1'}{p_0 \parallel p_1 \xrightarrow{\lambda} p_0 \parallel p_1'} \qquad \frac{p_0 \xrightarrow{\alpha!n} p_0' \qquad p_1 \xrightarrow{\alpha?n} p_1'}{p_0 \parallel p_1 \xrightarrow{\tau} p_0' \parallel p_1'}$$

- **Restriction**

$$\frac{p \xrightarrow{\lambda} p'}{p \setminus L \xrightarrow{\lambda} p' \setminus L} \quad \text{where if } \lambda \equiv \alpha?n \text{ or } \lambda \equiv \alpha!n \text{ then } \alpha \notin L$$

- **Relabelling**

$$\frac{p \xrightarrow{\lambda} p'}{p[f] \xrightarrow{f(\lambda)} p'[f]}$$

  where f is extended to labels as $f(\tau) = \tau$ and $f(\alpha?n) = f(\alpha)?n$ and $f(\alpha!n) = f(\alpha)!n$

- **Identifiers**

$$\frac{p[a_1/x_1, \cdots, a_n/x_n] \xrightarrow{\lambda} p'}{P(a_1, \cdots, a_n) \xrightarrow{\lambda} p'}$$

- **Nil process** no rules

A derivation from the operational semantics

$$\dfrac{\dfrac{(\alpha!3 \rightarrow \textbf{nil}) \xrightarrow{\alpha!3} \textbf{nil}}{(\alpha!3 \rightarrow \textbf{nil}) + P \xrightarrow{\alpha!3} \textbf{nil}}}{((\alpha!3 \rightarrow \textbf{nil}) + P) \parallel (\tau \rightarrow \textbf{nil}) \xrightarrow{\alpha!3} \textbf{nil} \parallel (\tau \rightarrow \textbf{nil})}$$

A derivation from the operational semantics

$$\dfrac{\dfrac{\dfrac{(\alpha!3 \to \mathbf{nil}) \xrightarrow{\alpha!3} \mathbf{nil}}{(\alpha!3 \to \mathbf{nil}) + P \xrightarrow{\alpha!3} \mathbf{nil}}}{((\alpha!3 \to \mathbf{nil}) + P) \parallel (\tau \to \mathbf{nil}) \xrightarrow{\alpha!3} \mathbf{nil} \parallel (\tau \to \mathbf{nil})} \qquad (\alpha?x \to \mathbf{nil}) \xrightarrow{\alpha?3} \mathbf{nil}}{(((\alpha!3 \to \mathbf{nil}) + P) \parallel (\tau \to \mathbf{nil})) \parallel (\alpha?x \to \mathbf{nil}) \xrightarrow{\tau} (\mathbf{nil} \parallel (\tau \to \mathbf{nil})) \parallel \mathbf{nil}}$$

A derivation from the operational semantics

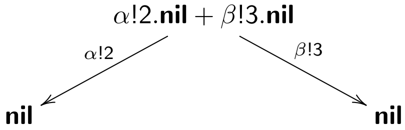
$$\dfrac{\dfrac{\dfrac{\dfrac{(\alpha!3 \to \mathbf{nil}) \xrightarrow{\alpha!3} \mathbf{nil}}{(\alpha!3 \to \mathbf{nil}) + P \xrightarrow{\alpha!3} \mathbf{nil}}}{((\alpha!3 \to \mathbf{nil}) + P) \parallel (\tau \to \mathbf{nil}) \xrightarrow{\alpha!3} \mathbf{nil} \parallel (\tau \to \mathbf{nil})} \qquad (\alpha?x \to \mathbf{nil}) \xrightarrow{\alpha?3} \mathbf{nil}}{(((\alpha!3 \to \mathbf{nil}) + P) \parallel (\tau \to \mathbf{nil})) \parallel (\alpha?x \to \mathbf{nil}) \xrightarrow{\tau} (\mathbf{nil} \parallel (\tau \to \mathbf{nil})) \parallel \mathbf{nil}}}{(((\alpha!3 \to \mathbf{nil} + P) \parallel \tau \to \mathbf{nil}) \parallel \alpha?x \to \mathbf{nil}) \setminus \{\alpha\} \xrightarrow{\tau} ((\mathbf{nil} \parallel \tau \to \mathbf{nil}) \parallel \mathbf{nil}) \setminus \{\alpha\}}$$

Further examples

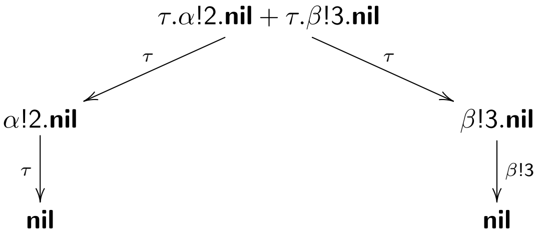
(Write . for $\rightarrow$)

Each step justified by a derivation:

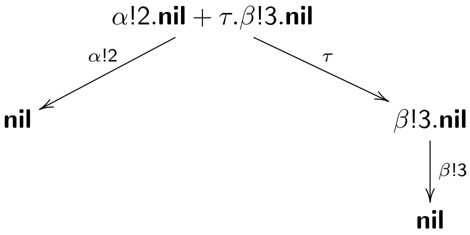
- External choice



- Internal choice



- Mixed choice



- Exercise:

$$\alpha!3.\textbf{nil} \parallel \alpha?x.\beta!x.\textbf{nil}$$

- Exercise:

$$(\alpha!3.\textbf{nil} \parallel \alpha?x.\beta!x.\textbf{nil}) \setminus \{\alpha\}$$

- Exercise:

$$(\alpha?x.\textbf{nil} \parallel \beta!4)[\alpha \mapsto \beta, \beta \mapsto \beta]$$

- Exercise: P where

$$P \stackrel{\text{def}}{=} \alpha?x \to x = 1 \to Q(x) \qquad Q(x) \stackrel{\text{def}}{=} \beta!x \to \gamma!x \to Q(x)$$

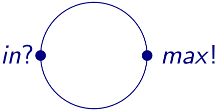
Conditionals

- Encoding of conditionals:

$$\texttt{if } b \texttt{ then } p_0 \texttt{ else } p_1 \equiv (b \to p_0) + (\neg b \to p_1)$$

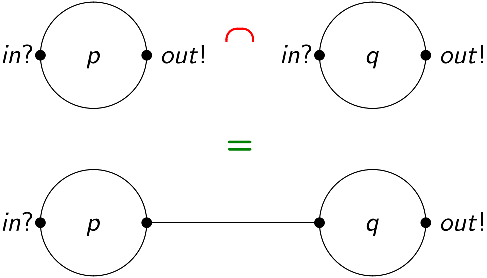
- Example: Maximum of two inputs

$$\begin{array}{l} in?x \to (in?y \to \\ \quad (\;\; x \leq y \to max!y \\ \quad\; + \\ \quad\;\;\; y \leq x \to max!x \quad)) \end{array}$$



Linking processes

Connect p's output port to q's input port:



Definition:

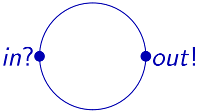
$$p \frown q = (p[c/out] \parallel q[c/in]) \setminus c$$

where c is a fresh channel name

Buffers

- **Definition:**

$B \stackrel{\text{def}}{=} in?x \to (out!x \to B)$



- n-ary buffer

$$\underbrace{B \frown B \frown \ldots \frown B}_{n \text{ times}}$$

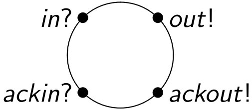
- Exercise: Draw the transition system for $B \frown B$

Remember: $p \frown q = (p[c/out] \parallel q[c/in]) \setminus c$

Buffer with acknowledgements

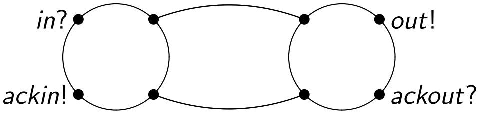
- **Definition:**

$$D \stackrel{\text{def}}{=} in?x \rightarrow out!x \rightarrow ackout? \rightarrow ackin! \rightarrow D$$



- Chaining now establishes two links:

$D \Cap D$

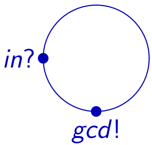


- How would this differ from the following process?

$$D' \stackrel{\text{def}}{=} in?x \rightarrow ackin! \rightarrow out!x \rightarrow ackout? \rightarrow D'$$

Euclid's algorithm in CCS

Interface:



Implementation:

$$
\begin{aligned}
E(x,y) \stackrel{\text{def}}{=} &\quad x = y \rightarrow gcd!x \rightarrow \mathbf{nil} \\
&+ \quad x < y \rightarrow E(x, y-x) \\
&+ \quad y < x \rightarrow E(x-y, x)
\end{aligned}
$$

$$Euclid \stackrel{\text{def}}{=} in?x \rightarrow in?y \rightarrow E(x,y)$$

Euclid's algorithm in CCS (without parameterized processes)

$$
\begin{aligned}
Step \stackrel{\text{def}}{=} \; & in?x \rightarrow \\
& in?y \rightarrow \\
& \quad (x = y \rightarrow gcd!x \rightarrow \mathbf{nil}) \\
& \quad + \\
& \quad (x < y \rightarrow out!x \rightarrow out!(y - x) \rightarrow \mathbf{nil}) \\
& \quad + \\
& \quad (y < x \rightarrow out!(x - y) \rightarrow out!y \rightarrow \mathbf{nil})
\end{aligned}
$$

$$Euclid \stackrel{\text{def}}{=} Step^{\frown} Euclid$$