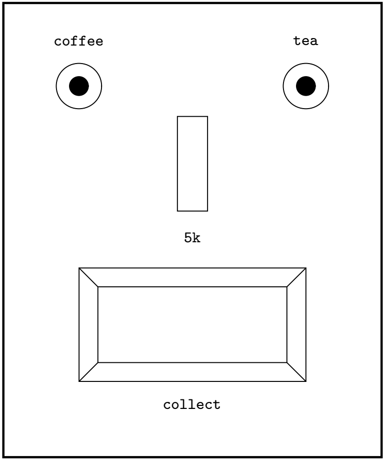
Lecture 2

- informal introduction to CCS
- syntax of CCS
- semantics of CCS

Vending machines

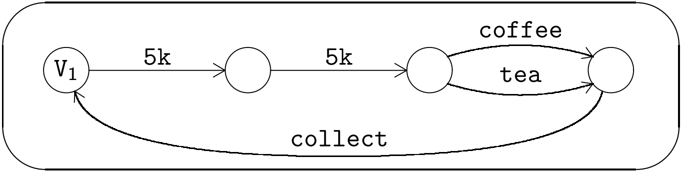
$$\mathtt{V}_1 \stackrel{\text{def}}{=} \mathtt{5k.5k.(\ coffee.collect.}\mathtt{V}_1 \\ \qquad\qquad\qquad + \mathtt{tea.collect.}\mathtt{V}_1\mathtt{)}$$

$$\mathtt{V}_3 \stackrel{\text{def}}{=} \mathtt{5k.5k.coffee.collect.}\mathtt{V}_3 \\ \qquad + \mathtt{5k.5k.tea.collect.}\mathtt{V}_3$$

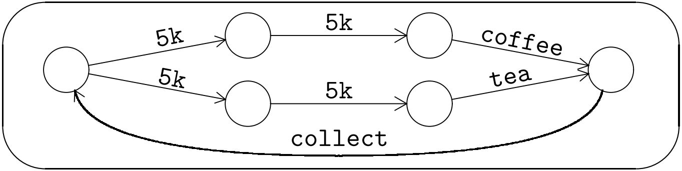


Vending machines - cont.

$\mathtt{V}_1 \stackrel{\mathrm{def}}{=} \mathtt{5k.5k.(\ coffee.collect.V_1 + tea.collect.V_1)}$



$\mathtt{V}_2 \stackrel{\mathrm{def}}{=} \mathtt{5k.5k.coffee.collect.V_2 + 5k.5k.tea.collect.V_2}$



CCS Basics (Sequential Fragment)

- *Nil* (or 0) process (the only atomic process)
- action prefixing ($a.P$)
- names and recursive definitions ($\stackrel{\mathrm{def}}{=}$)
- nondeterministic choice ($+$)

This is Enough to Describe Sequential Processes

Any finite LTS can be described by using the operations above.

CCS Basics (Parallelism and Renaming)

- parallel composition ($|$)
  (synchronous communication between two components = handshake synchronization)
- restriction ($P \smallsetminus L$)
- relabelling ($P[f]$)

Definition of CCS (channels, actions, process names)

Let

- $\mathcal{A}$ be a set of channel names (e.g. *tea*, *coffee* are channel names)
- $\mathcal{L} = \mathcal{A} \cup \overline{\mathcal{A}}$ be a set of labels where
  - $\overline{\mathcal{A}} = \{\overline{a} \mid a \in \mathcal{A}\}$
    (elements of $\mathcal{A}$ are called names,
    elements of $\overline{\mathcal{A}}$ are called co-names)
  - by convention $\overline{\overline{a}} = a$
- $Act = \mathcal{L} \cup \{\tau\}$ is the set of actions where
  - τ is the internal or silent action

  (e.g. τ, *tea*, $\overline{coffee}$ are actions)
- $\mathcal{K}$ is a set of process names (constants) (e.g. CM).

Definition of CCS (expressions)

$$
\begin{array}{llll}
P := & K & | & \text{process constants } (K \in \mathcal{K}) \\
& \alpha.P & | & \text{prefixing } (\alpha \in Act) \\
& \sum_{i \in I} P_i & | & \text{summation } (I \text{ is an arbitrary index set}) \\
& P_1 | P_2 & | & \text{parallel composition} \\
& P \smallsetminus L & | & \text{restriction } (L \subseteq \mathcal{A}) \\
& P[f] & | & \text{relabelling } (f : Act \rightarrow Act) \text{ such that}
\end{array}
$$

- $f(\tau) = \tau$
- $f(\overline{a}) = \overline{f(a)}$

The set of all terms generated by the abstract syntax is called CCS process expressions (and denoted by $\mathcal{P}$).

Notation

$$P_1 + P_2 = \sum_{i \in \{1,2\}} P_i \qquad Nil = 0 = \sum_{i \in \emptyset} P_i$$

Precedence

Precedence

1. restriction and relabelling (tightest binding)
2. action prefixing
3. parallel composition
4. summation

Example: $R + a.P|b.Q \setminus L$ means $R + \big((a.P)|(b.(Q \setminus L))\big)$.

Definition of CCS (defining equations)

CCS program

A collection of defining equations of the form

$$K \stackrel{\mathrm{def}}{=} P$$

where $K \in \mathcal{K}$ is a process constant and $P \in \mathcal{P}$ is a CCS process expression.

- Only one defining equation per process constant.
- Recursion is allowed: e.g. $A \stackrel{\mathrm{def}}{=} \overline{a}.A \mid A$.

Semantics of CCS

Syntax

CCS
(collection of defining equations)

$\longrightarrow$

Semantics

LTS
(labelled transition systems)

HOW?

Structural Operational Semantics for CCS

Structural Operational Semantics (SOS) – G. Plotkin 1981

Small-step operational semantics where the behaviour of a system is inferred using syntax driven rules.

Given a collection of CCS defining equations, we define the following LTS $(Proc, Act, (\xrightarrow{a})_{a \in Act})$:

- $Proc = \mathcal{P}$ (the set of all CCS process expressions)
- $Act = \mathcal{L} \cup \{\tau\}$ (the set of all CCS actions including τ)
- transition relation is given by SOS rules of the form:

$$\text{RULE} \quad \frac{\textit{premises}}{\textit{conclusion}} \quad \textit{conditions}$$

SOS rules for CCS ($\alpha \in Act$, $a \in \mathcal{L}$)

$$\text{ACT} \quad \frac{}{\alpha.P \xrightarrow{\alpha} P} \qquad\qquad \text{SUM}_j \quad \frac{P_j \xrightarrow{\alpha} P'_j}{\sum_{i \in I} P_i \xrightarrow{\alpha} P'_j} \; j \in I$$

$$\text{COM1} \quad \frac{P \xrightarrow{\alpha} P'}{P|Q \xrightarrow{\alpha} P'|Q} \qquad\qquad \text{COM2} \quad \frac{Q \xrightarrow{\alpha} Q'}{P|Q \xrightarrow{\alpha} P|Q'}$$

$$\text{COM3} \quad \frac{P \xrightarrow{a} P' \qquad Q \xrightarrow{\overline{a}} Q'}{P|Q \xrightarrow{\tau} P'|Q'}$$

$$\text{RES} \quad \frac{P \xrightarrow{\alpha} P'}{P \smallsetminus L \xrightarrow{\alpha} P' \smallsetminus L} \; \alpha, \overline{\alpha} \notin L \qquad\qquad \text{REL} \quad \frac{P \xrightarrow{\alpha} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]}$$

$$\text{CON} \quad \frac{P \xrightarrow{\alpha} P'}{K \xrightarrow{\alpha} P'} \; K \stackrel{\text{def}}{=} P$$

Deriving Transitions in CCS

Let $A \stackrel{\mathrm{def}}{=} a.A$. Then

$$\big((A \,|\, \overline{a}.Nil) \,|\, b.Nil\big)[c/a] \stackrel{c}{\longrightarrow} \big((A \,|\, \overline{a}.Nil) \,|\, b.Nil\big)[c/a].$$

$$\text{REL}\ \dfrac{\text{COM1}\ \dfrac{\text{COM1}\ \dfrac{\text{CON}\ \dfrac{\text{ACT}\ \dfrac{}{a.A \stackrel{a}{\longrightarrow} A}}{A \stackrel{a}{\longrightarrow} A}\, A \stackrel{\mathrm{def}}{=} a.A}{A \,|\, \overline{a}.Nil \stackrel{a}{\longrightarrow} A \,|\, \overline{a}.Nil}}{(A \,|\, \overline{a}.Nil) \,|\, b.Nil \stackrel{a}{\longrightarrow} (A \,|\, \overline{a}.Nil) \,|\, b.Nil}}{\big((A \,|\, \overline{a}.Nil) \,|\, b.Nil\big)[c/a] \stackrel{c}{\longrightarrow} \big((A \,|\, \overline{a}.Nil) \,|\, b.Nil\big)[c/a]}$$

LTS of the Process $a.\mathit{Nil} \mid \overline{a}.\mathit{Nil}$

$$\begin{tikzcd}
 & a.\mathit{Nil} \mid \overline{a}.\mathit{Nil} \arrow[dl, "a"'] \arrow[dd, "\tau"] \arrow[dr, "\overline{a}"] & \\
\mathit{Nil} \mid \overline{a}.\mathit{Nil} \arrow[dr, "\overline{a}"] & & a.\mathit{Nil} \mid \mathit{Nil} \arrow[dl, "a"'] \\
 & \mathit{Nil} \mid \mathit{Nil} &
\end{tikzcd}$$