A Calculus of Communicating Systems (CCS)

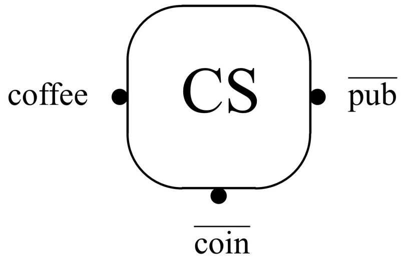
- labelled transition systems
- process algebras
- informal introduction to CCS
- syntax of CCS
- semantics of CCS

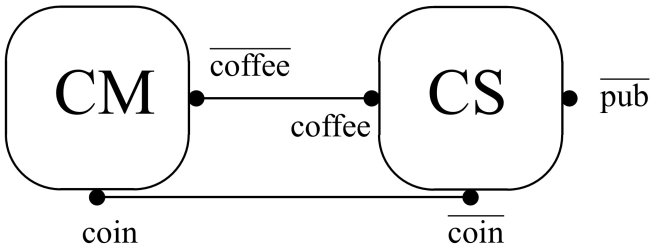
Definition of CCS (channels, actions, process names)

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
  - $\overline{\mathcal{A}} = \{\overline{a} \mid a \in \mathcal{A}\}$
    ($\mathcal{A}$ are called names and $\overline{\mathcal{A}}$ 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).



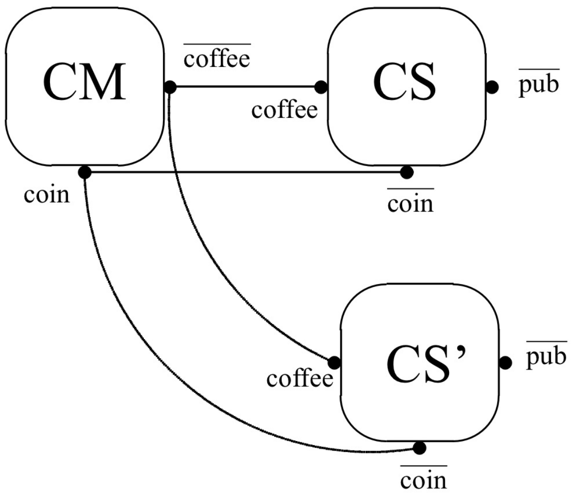


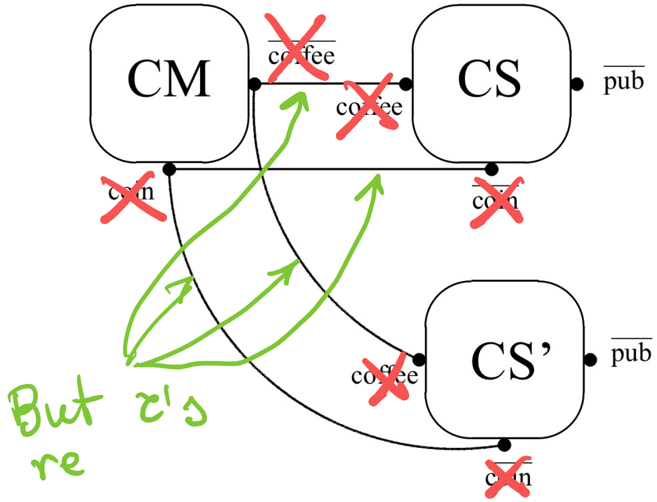
$\mathrm{CM} \mid \mathrm{CS}$

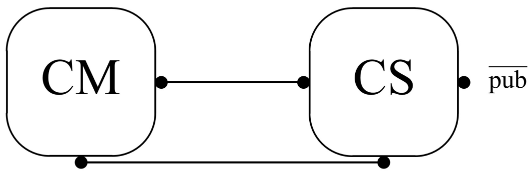
$\mathrm{CS} \stackrel{\mathrm{def}}{=} \overline{\mathrm{pub}}.\overline{\mathrm{coin}}.\mathrm{coffee}.\mathrm{CS}$

$\mathrm{CM} \stackrel{\mathrm{def}}{=} \mathrm{coin}.\overline{\mathrm{coffee}}.\mathrm{CM}$

$\mathrm{CTM} \stackrel{\mathrm{def}}{=} \mathrm{coin}.(\overline{\mathrm{coffee}}.\mathrm{CTM} + \overline{\mathrm{tea}}.\mathrm{CTM})$



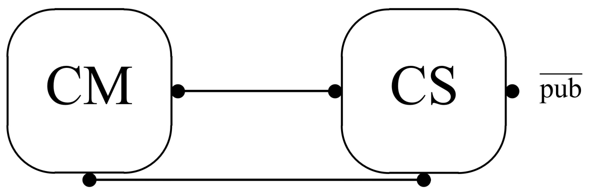




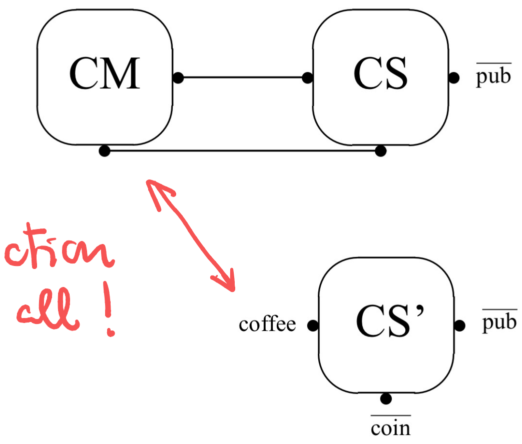
$\mathrm{SmUni} \stackrel{\mathrm{def}}{=} (\mathrm{CM} \mid \mathrm{CS}) \setminus \mathrm{coin} \setminus \mathrm{coffee}$

$UNI_2 = CMS_{1,2} \setminus \{coin, coffee\}$

$CMS_{1,2} = \mathrm{CM} \mid \mathrm{CS} \mid \mathrm{CS}'$



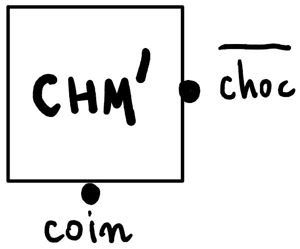
$$\mathrm{SmUni} \stackrel{\mathrm{def}}{=} (\mathrm{CM} \mid \mathrm{CS}) \setminus \mathrm{coin} \setminus \mathrm{coffee}$$



no connection at all !

$\mathrm{SmUni} \mid \mathrm{CS}'$

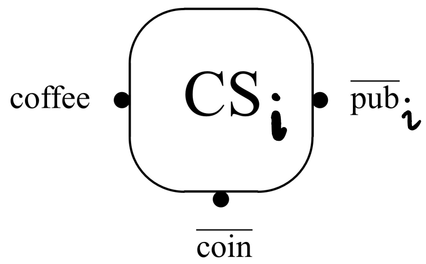
$$\mathrm{CHM} \stackrel{\mathrm{def}}{=} \mathrm{coin}.\overline{\mathrm{choc}}.\mathrm{CHM}$$



$$CS_i = CS[pub/pub_i]$$

$$\mathrm{VM} \stackrel{\mathrm{def}}{=} \mathrm{coin}.\overline{\mathrm{item}}.\mathrm{VM}$$

$$\mathrm{CHM}' \stackrel{\mathrm{def}}{=} \mathrm{VM}[\mathrm{choc}/\mathrm{item}]$$



Process Algebra

Basic Principle

1. Define a few atomic processes (modelling the simplest process behaviour).
2. Define compositionally new operations (building more complex process behaviour from simple ones).

Example

1. atomic instruction: assignment (e.g. x:=2 and x:=x+2)
2. new operators:
   - sequential composition ($P_1; P_2$)
   - parallel composition ($P_1 \mid P_2$)

   Now e.g. (x:=1 | x:=2); x:=x+2; (x:=x-1 | x:=x+5) is a process.

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 (up to isomorphism) described by using the operations above.

CCS Basics (Parallelism and Renaming)

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

Definition of CCS (expressions)

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

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

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?

Labelled Transition System

Definition

A labelled transition system (LTS) is a triple $(Proc, Act, \{\xrightarrow{a} \mid a \in Act\})$ where

- *Proc* is a set of states (or processes),
- *Act* is a set of labels (or actions), and
- for every $a \in Act$, $\xrightarrow{a} \subseteq Proc \times Proc$ is a binary relation on states called the transition relation.

We will use the infix notation $s \xrightarrow{a} s'$ meaning that $(s, s') \in \xrightarrow{a}$.

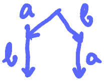
Sometimes we distinguish the initial (or start) state.

Sequencing, Nondeterminism and Parallelism

LTS explicitly focuses on interaction. triggered execution of actiones

LTS can also describe:

- sequencing $(a; b)$
- choice (nondeterminism) $(a + b)$



- limited notion of parallelism (by using interleaving) $(a|b)$

Let $(Proc, Act, \{\xrightarrow{a} \mid a \in Act\})$ be an LTS.

- we extend $\xrightarrow{a}$ to the elements of Act^* $\xrightarrow{\alpha}$
- $\longrightarrow = \bigcup_{a \in Act} \xrightarrow{a}$
- $\longrightarrow^*$ is the reflexive and transitive closure of $\longrightarrow$
- $s \xrightarrow{a}$ and $s \overset{a}{\not\longrightarrow}$
- reachable states

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 $(Proc, Act, \{\xrightarrow{a} \mid a \in Act\})$:

- $Proc = \mathcal{P}$ (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:

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

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

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

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

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

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

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

Deriving Transitions in CCS

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

$$\big((A \mid \overline{a}.Nil) \mid b.Nil\big)[c/a] \xrightarrow{c} \big((A \mid \overline{a}.Nil) \mid b.Nil\big)[c/a].$$

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

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

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