Processos e Concorrência 2015/16
Bloco de acetatos 4

Alexandre Madeira
(based on Luís S. Barbosa 2014/15 course slides)
HASLab INESC TEC, DI UMINHO

March, 2016

Actions & processes

Action

- elementary unit of behaviour that can **execute itself atomically in time** (no duration), after which it terminates successfully
- is a **latency for interaction**

$$\alpha \quad ::= \quad \tau \;\mid\; a \;\mid\; \alpha \mid \alpha$$

- $a \mid b \mid \cdots \mid z$ represent a collection of actions that occur at the same time instant
- τ is the empty action, which contains no actions and as such cannot be observed
- $\langle N, \mid, \tau \rangle$ forms a **monoid**

Actions & processes

Process

is a description of how the interaction capacities of a system evolve, i.e., its **behaviour**

for example,

$$E =^{df} a.b + a.E$$

- **analogy**: regular expressions vs finite automata

The framework

Process

... abstract representation of a system's behaviour

Algebra

... a mathematical structure satisfying a particular set of axioms

Process Algebra

... a framework for the specification and manipulation of process terms as induced by a collection of operator symbols, encompassing an operational and an axiomatic theory

The framework

Transition systems operational representation of system's behaviour through labelled graphs

Behavioural equivalences to distinguished states in transition systems

Process terms algebraic representation of transition systems (for the purpose of mathematical reasoning)

Structural operational semantics inductive proof rules to provide each process term with its intended transition system

Equational theory Axiomatic theory of processes, expressed in an equational logic on process terms, that is sound and complete wrt bisimilarity.

Instantiating the framework

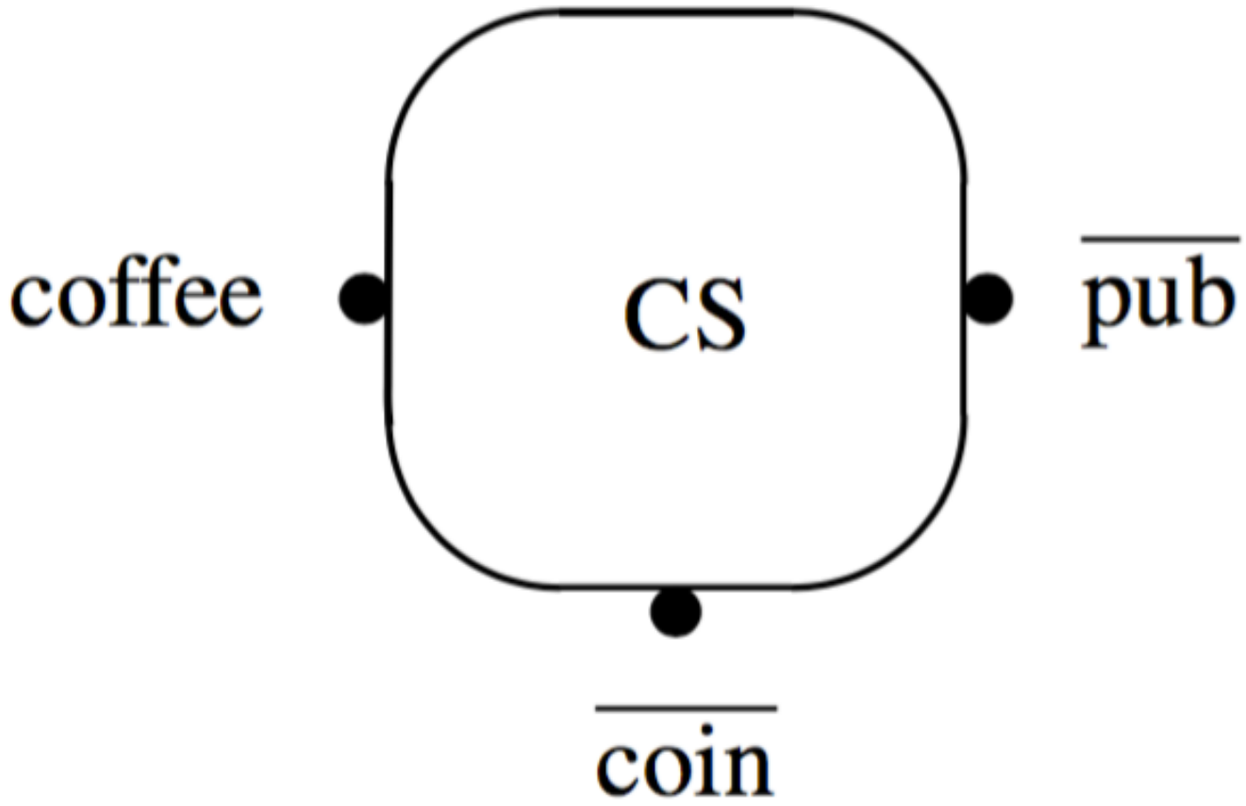
CCS: a prototypical process algebra

- **Calculus of Communicating Systems** [Milner, 1980]

First informal intuitions on CCS

Lets take some formal intuitions on this process algebra

Actions – the process interface



- Given an action (channel/port) a, we use $\overline{a}$ (the complementary actions) for the output of a

First informal intuitions on CCS

Inaction, prefixing and recursive definitions

- 0 – deadlock behaviour process
- If P is a process and a an action, $a.P$ is a process
  - $a.0$
  - $a.a.0$
  - $b.a.0$
- We can introduce **names for processes**:
  - $P = a.b.0$
  - $Clock = tick.Clock$
  - $CM = coin.\overline{coffee}.CM$

First informal intuitions on CCS

Choice operator

- If P and Q are processes then so is $P + Q$
  - a.0 + a.b.0
- A coffee and tea machine:

$$CTM = (\overline{coffee}.CTM + \overline{tea}.CTM)$$

First informal intuitions on CCS

parallel composition

Given two CCS expressions P and Q, the process $P|Q$ describes a system in which P and Q

- may proceed independently
- may communicate via complementary actions $(a/\overline{a})$

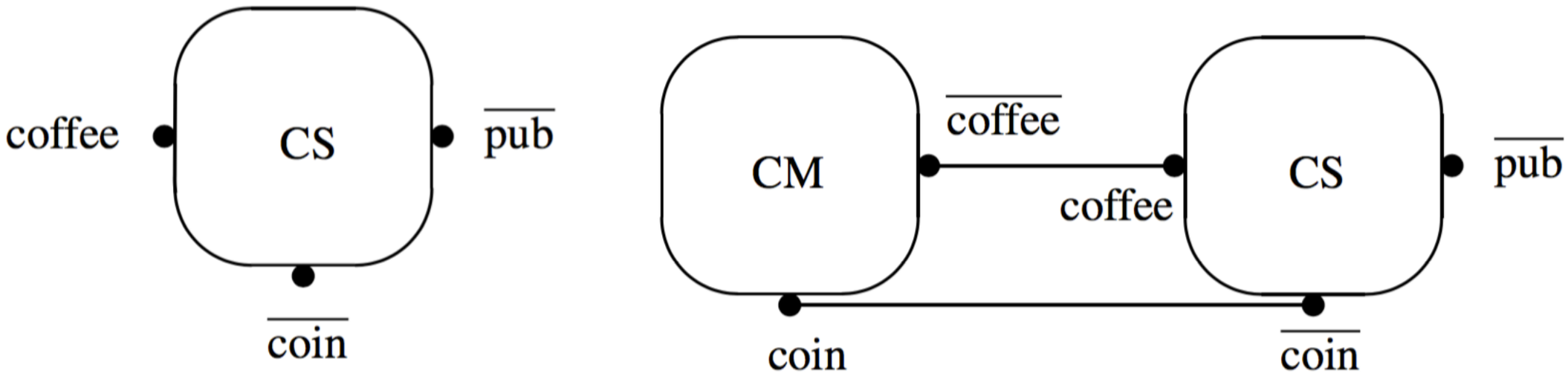
First informal intuitions on CCS

parallel composition

" computer scientists are machines to turn coffees in publications":

$$CS = \overline{pub}.\overline{coin}.coffee.CS \quad CM = coin.\overline{coffee}.CM$$

$$CM \mid CS$$

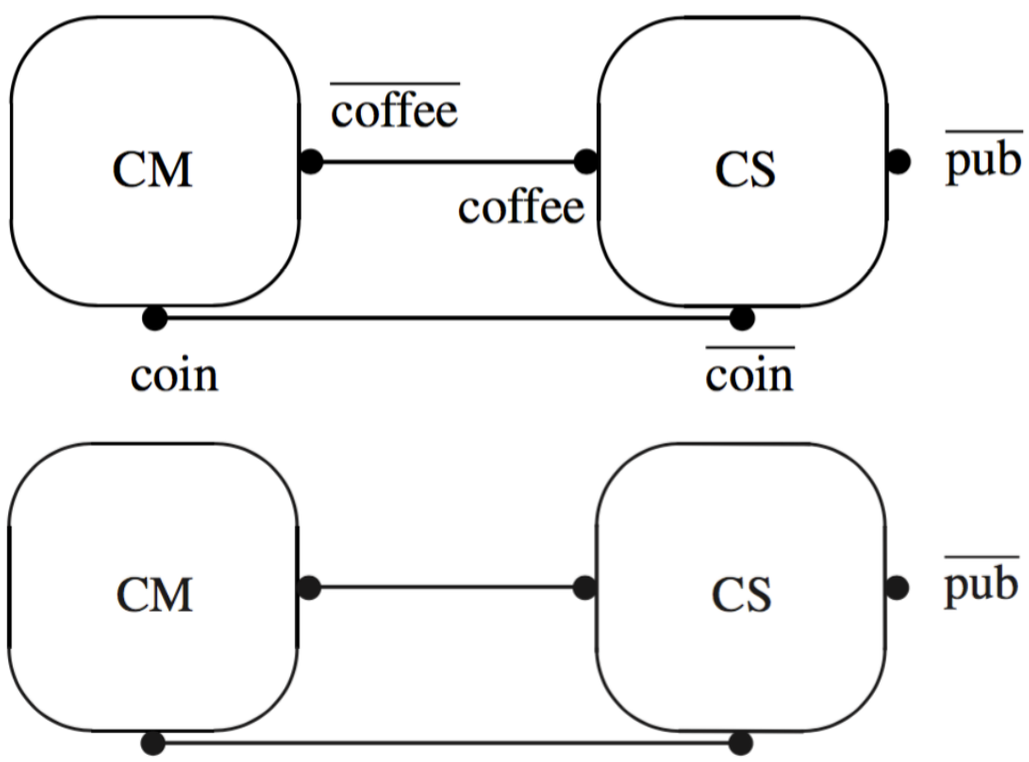


First informal intuitions on CCS

Restriction

- If P is a process and L is a set of actions names, $P \setminus L$ is also a process
- the scope of the port names in L is restricted to P; these port names can only be used for communication within P.

$$(CM|CS) \setminus coin \setminus coffee$$



First informal intuitions on CCS

Relabelling

- If P is a process and f from labels to labels (satisfying some additional properties that will made precise in the sequel), then $P[f]$ is also a process

More vending machines?

$$CM = coin.\overline{coffee}.CM$$

$$CHM = coin.\overline{choc}.CHM$$

$$CRM = coin.\overline{crisps}.CRM$$

$\ldots$

$$VM = coin.\overline{item}.VM[coffee/item]$$

First informal intuitions on CCS

Behaviour of a process

"In CSS, processes change state by performing transitions and these transitions are labelled by the actions that caused them"

- e.g. $a.b.0$ induces the behaviour $\bullet \xrightarrow{a} \bullet \xrightarrow{b} \bullet$

$$a.b.0 \xrightarrow{a} b.0 \xrightarrow{b} 0$$

$$CS = \overline{pub}.\overline{coin}.coffee.CS$$

First informal intuitions on CCS

Again:

Given two CCS expressions P and Q, the process $P|Q$ describes a system in which P and Q

- may proceed independently
- may communicate via complementary actions $(a/\overline{a})$

$$\overline{coin}.coffee.CS \xrightarrow{\overline{coin}} coffee.CS \quad coin.\overline{coffee} \xrightarrow{coin} \overline{coffee}.CM$$

$$\overline{coin}.coffee.CS|coin.\overline{coffee} \xrightarrow{\tau} coffee.CS|\overline{coffee}.CM$$

First informal intuitions on CCS

Parallel composition

Let construct the LTS for

$$(CM|CS) \setminus coin \setminus coffee$$

$\equiv$

$$(coin.\overline{coffee}.CM|\overline{pub}.\overline{coin}.coffee.CS) \setminus coin \setminus coffee$$

Observe that it is "behavioural equivalent" to the process

$$spec = \overline{pub}.spec$$

More examples

Buffers

1-position buffer: $A(in, out) =^{df} in.\overline{out}.0$

... non terminating: $B(in, out) =^{df} in.\overline{out}.B$

... with two output ports: $C(in, o_1, o_2) =^{df} in.(\overline{o_1}.C + \overline{o_2}.C)$

... non deterministic: $D(in, o_1, o_2) =^{df} in.\overline{o_1}.D + in.\overline{o_2}.D$

... with parameters: $B(in, out) =^{df} in(x).\overline{out}\langle x\rangle.B$

More Examples

n-position buffers

1-position buffer: $S =^{df} (B\langle in, m\rangle | B\langle m, out\rangle) \setminus m$

n-position buffer:

$$Bn =^{df} (B\langle in, m_1\rangle \mid B\langle m_1, m_2\rangle \mid \cdots \mid B\langle m_{n-1}, out\rangle) \setminus \{m_i | i < n\}$$

CCS Syntax

Standard syntax

The set $\mathbb{P}$ of processes is the set of all terms generated by the following BNF:

$$E ::= A(x_1, ..., x_n) \mid a.E \mid \sum_{i \in I} E_i \mid E_0 | E_1 \mid E[f] \mid E \setminus K$$

where

- A is a process name
- $a \in Act$ where $Act = \mathcal{A} \cup \overline{\mathcal{A}} \cup \{\tau\}$
- $K \subseteq L$ where $L = \mathcal{A} \cup \overline{\mathcal{A}}$ and
- $f : Act \to Act$ is a **relabelling function** satisfying the following constraints:
  - $f(\tau) = \tau$
  - $f(\overline{a}) = \overline{f(a)}$, $a \in A$

CCS Syntax

Standard syntax

The set $\mathbb{P}$ of processes is the set of all terms generated by the following BNF:

$$E ::= A(x_1, ..., x_n) \mid a.E \mid \sum_{i \in I} E_i \mid E_0 | E_1 \mid E[f] \mid E \setminus K$$

Abbreviatures

$$E_0 + E_1 =^{abv} \sum_{i \in \{0,1\}} E_i$$

$$0 =^{abv} \sum_{i \in \emptyset} E_i$$

CCS Syntax

Process declaration

$$A(\vec{x}) \ =^{df} \ E_A$$

with $\mathrm{fn}(E_A) \subseteq \vec{x}$ (where $\mathrm{fn}(P)$ is the set of **free** variables of P).

- used as, *e.g.*, $\boxed{A(a,b,c) \ =^{df} \ a.b.0 + A\langle d,e,f\rangle}$

Process declaration: fixed point expression

$$\underline{fix}(X = E_X)$$

Exercise

Which of the following expressions are syntactically correct CCS expressions?

1. $a.b.A + B$
2. $(a.0 + \overline{a}.A) \setminus \{a, b\}$
3. $(a.0|\overline{a}.A) \setminus \{a, \tau\}$
4. $a.B + [a/b]$
5. $(a.B + b.B)[a/b, b/a]$
6. $(a.B + b.B)[a/\tau, b/a]$
7. $(a.b.A + \overline{a}.0)|B$
8. $(a.b.A + \overline{a}.0).B$
9. $(a.b.A + \overline{a}.0) + B$
10. $(0|0) + 0$

Sort

A sort of a process P is an **interface** for P

- minimal sort: $\mathcal{L}(P) = \bigcap\{K \subseteq L \mid K \text{ is a sort of } P\}$
- syntactic sort, *i.e.*, the set of free variables:

$$\begin{aligned}
\mathrm{fn}(a.P) &= \{a\} \cup \mathrm{fn}(P) \\
\mathrm{fn}(\tau.P) &= \mathrm{fn}(P) \\
\mathrm{fn}(\sum_{i \in I} P_i) &= \bigcup_{i \in I} \mathrm{fn}(P_i) \\
\mathrm{fn}(P|Q) &= \mathrm{fn}(P) \cup \mathrm{fn}(Q) \\
\mathrm{fn}(P \setminus K) &= \mathrm{fn}(P) - (K \cup \overline{K})
\end{aligned}$$

and, for each $P(\vec{x}) =^{df} E$, $\mathrm{fn}(E) \subseteq \mathrm{fn}(P(\vec{x})) = \vec{x}$.

Sort

The minimal sort $\mathcal{L}(P)$ corresponds to the set of actions which effectively label valid transitions of P. All the other sorts are upper bounds of $\mathcal{L}(P)$ and represent interaction **possibilities**.

Warning

- $(a.b.c.0) \setminus \{a\}$ has no transitions, so its sort is $\emptyset$
- however: $\mathrm{fn}(a.b.c.0 \setminus \{a\}) = \{b, c\}$

Semantics

Two-level semantics

- **behavioural** given by transition rules which express how system's components interact
- **arquitectural**, expresses a notion of **similar assembly configurations** and is expressed through a structural congruence relation;

Semantics

$$\frac{}{a.p \xrightarrow{a} p}\ (act)$$

$$\frac{p \xrightarrow{a} p'}{p + q \xrightarrow{a} p'}\ (sum - l) \qquad \frac{q \xrightarrow{a} q'}{p + q \xrightarrow{a} q'}\ (sum - r)$$

$$\frac{p \xrightarrow{a} p'}{p \mid q \xrightarrow{a} p' \mid q}\ (par - l) \qquad \frac{q \xrightarrow{a} q'}{p \mid q \xrightarrow{a} p \mid q'}\ (par - r)$$

$$\frac{p \xrightarrow{a} p' \quad q \xrightarrow{\overline{a}} q'}{p \mid q \xrightarrow{\tau} p' \mid q'}\ (react) \qquad \frac{p \xrightarrow{a} p'}{p \setminus \{k\} \xrightarrow{a} p' \setminus \{k\}}\ (res)\ \ (\text{if } a \notin \{k, \overline{k}\})$$

$$\frac{p \xrightarrow{a} p'}{p[f] \xrightarrow{f(a)} p'[f]}\ (rel)\ \ (f \text{ relabelling function})$$

$$\frac{p \xrightarrow{a} p'}{k \xrightarrow{a} p'}\ (con)\ \ k =^{df} p$$

Exercise

Assume that $A =^{df} b.a.B$. By using the SOS rules for CCS prove the existence of the following transitions:

1. $(A \mid \overline{b}.0) \setminus \{b\} \xrightarrow{\tau} (a.B \mid 0) \setminus \{b\}$
2. $(A \mid \overline{b}.a.B) + (\overline{b}.A)[a/b] \xrightarrow{b} (A \mid a.B)$
3. $(A \mid \overline{b}.a.B) + (\overline{b}.A)[a/b] \xrightarrow{\overline{a}} A[a/b]$
4. $(a.E + b.0 \mid \overline{a}.F) \xrightarrow{a} (E \mid \overline{a}.F)$
5. $\big((a.E + b.0 \mid \overline{a}.F)\big) \setminus \{a\} \xrightarrow{b} (0 \mid \overline{a}.F) \setminus \{a\}$

Exercise

Let $A(a) =^{df} a.A$ and $B(b) =^{df} \overline{b}.B$. Compute the first derivatives of the following processes:

1. $A + B$
2. $A + B\langle a \rangle$
3. $A \mid B$
4. $A + B\langle a \rangle$
5. $(A \mid B)[a/b]$
6. $(A \mid B\langle a \rangle) \setminus \{a\}$

Exercise

Let $A(a, b, c, d) =^{df} \overline{a}.b.A + \overline{c}.d.A$. Draw the transition graphs of the following processes

1. A
2. $A \setminus \{a\}$
3. $T =^{df} a.(b.0 \mid T)$

Exercise

Consider the following specification of a control system for a crossing between a road and a railway. Events *car* and *train* modelled, respectively, a car or a train approaching the cross. Actions *up* e *dw* stand for the opening and closing of the protection bar to prevent cars to cross. Similarly, *green* and *red* model the semaphore for trains. Finally, events $\overline{ccross}$ and $\overline{tcross}$ come from sensors which register the actual cross of a car or a train, respectively.

$$Road =^{df} car.up.\overline{ccross}.\overline{dw}.Road$$

$$Rail =^{df} train.green.\overline{tcross}.\overline{red}.Rail$$

$$Signal =^{df} \overline{green}.red.Signal + \overline{up}.dw.Signal$$

$$C =^{df} (Road \mid Rail \mid Signal) \setminus \{green, red, up, dw\}$$

1. Explain the behaviour of this process and sketch its synchronisation diagram.
2. Compute the transition graph corresponding to process C

Exercise

Consider the following description of a two-position *buffer* with acknowledgements. Note the process is built from copies of a 1-position *buffer* also with acknowledgements: it acknowledges in *overliner* the reception of a message and waits in t the confirmation that a message sent was arrived to its destination.

$$Bs =^{df} (B(in, mo, mi, r) \mid B(mo, out, t, mi)) \setminus \{mo, mi\}$$

$$B(in, out, t, r) =^{df} in.\overline{out}.t.\overline{r}.B$$

1. Draw the graph of B.
2. Draw the synchronisation graph of Bs.
3. Check whether the behaviour of Bs is the intended one (drawing, for this purpose, the corresponding transition graph)
4. Find a solution to the problem detected (if any) and draw the corresponding transition graph. try with
   $B(in, out, t, r) =^{df} in.\overline{r}.\overline{out}.t.B$