Priority in Process Algebra

Gerald Lüttgen

Department of Computer Science

The University of Sheffield

www.dcs.shef.ac.uk/~luettgen

Joint work with Rance Cleaveland, SUNY at Stony Brook.

Reference: R. Cleaveland, G. Lüttgen, and V. Natarajan. Priority in Process Algebra. In *Handbook of Process Algebra*, ch. 12, pp. 711–765.

What “Priority” Means

Traditional process algebras model:

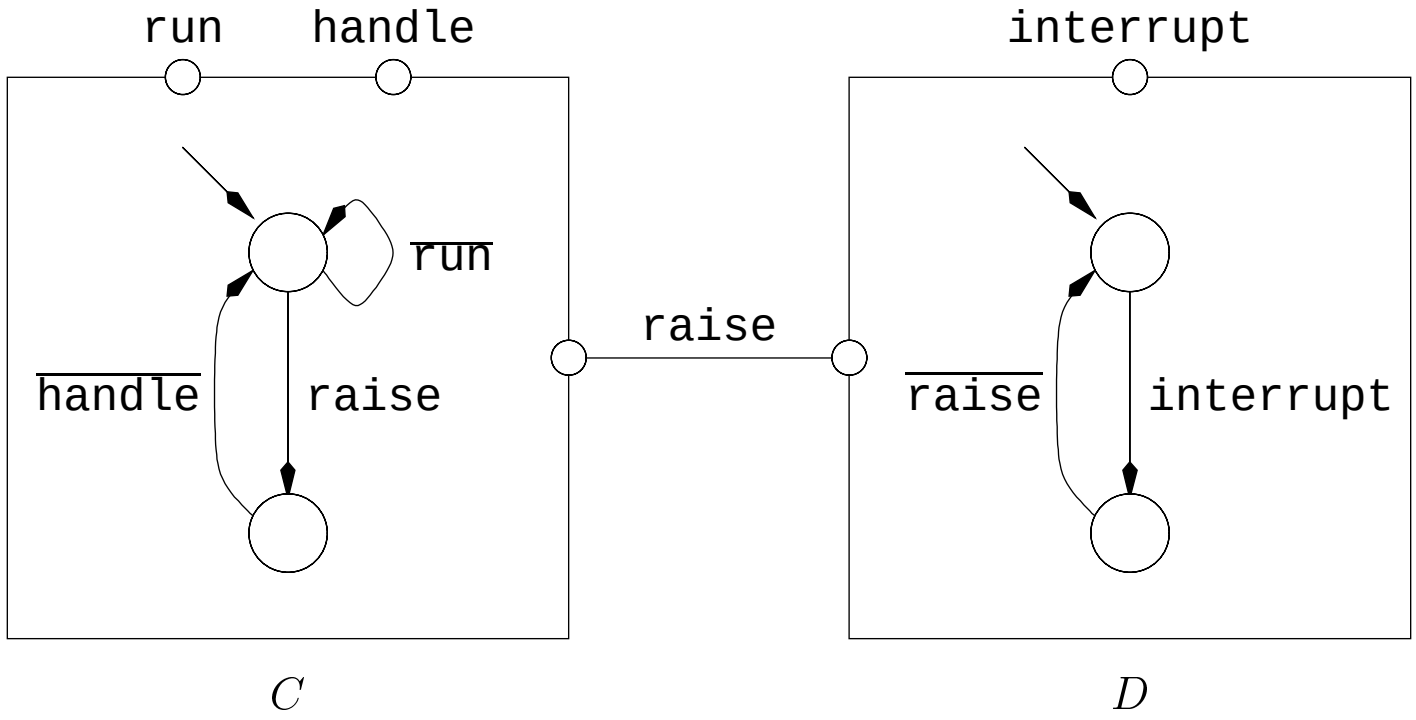
- Interactions between processes, environments.
- Nondeterminism regarding which interactions are performed.

Interactions are usually uninterpreted: no interaction has special significance.

But In many systems, some interactions take precedence over others!

This lecture Semantic issues involved in incorporating precedence / priority among interactions into process algebra (CCS).

Motivating Example #1: Interrupts



C: Computer which runs and handles interrupts.

D: Device driver which generates interrupts.

Example #1 (cont.)

System M_1 may be modeled in CCS as follows:

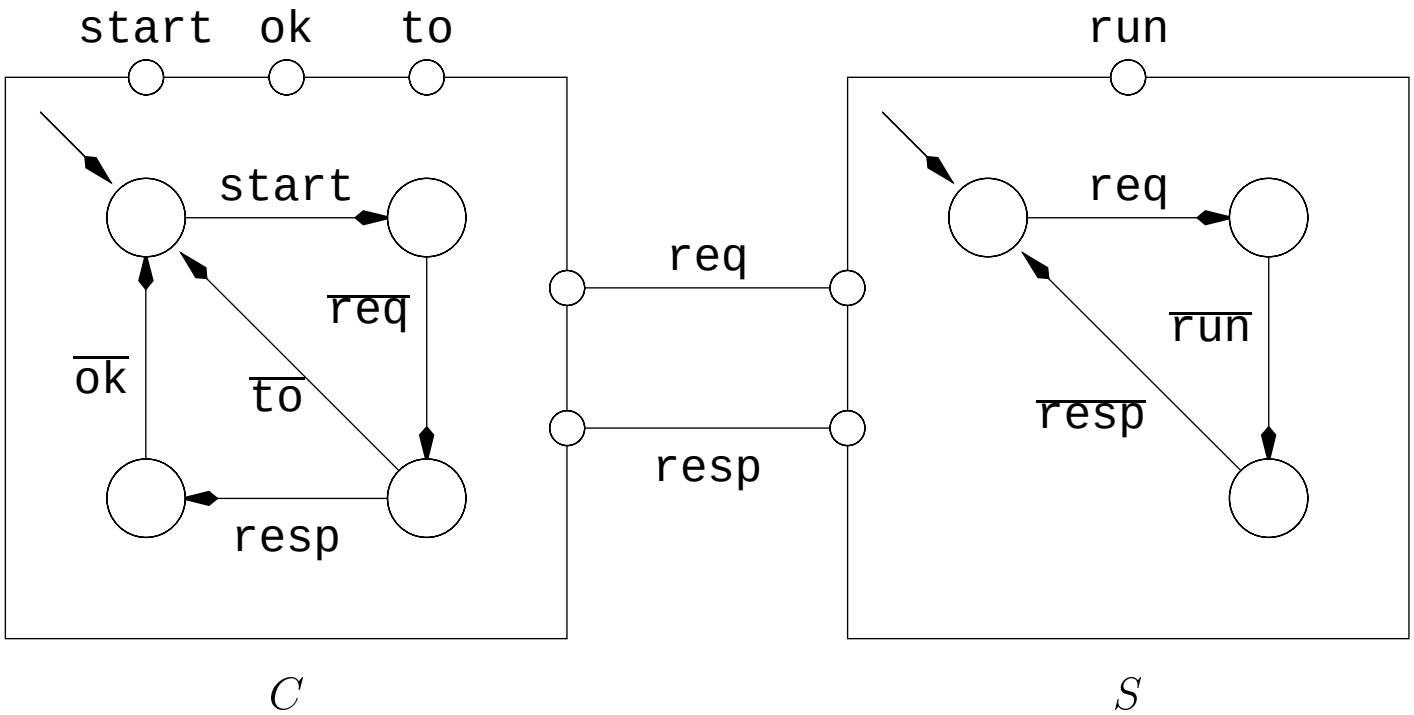
$$
\begin{aligned}
M_1 &\stackrel{\text{def}}{=} (C \mid D) \setminus \{\mathtt{raise}\} \\
C &\stackrel{\text{def}}{=} \overline{\mathtt{run}}.C + \mathtt{raise}.\overline{\mathtt{handle}}.C \\
D &\stackrel{\text{def}}{=} \mathtt{interrupt}.\overline{\mathtt{raise}}.D
\end{aligned}
$$

Possible computation:

$$M_1 \xrightarrow{\overline{\mathtt{run}}} \xrightarrow{\mathtt{interrupt}} \xrightarrow{\overline{\mathtt{run}}} \ldots$$

Interrupt is ignored because raise is not given higher priority than other actions!

Motivating Example #2: Time–outs



C: Client which starts, issues a request to / awaits a response from the server.

S: Server which, in response to a request, runs and then issues a response.

Example #2 (cont.)

System M_2 in CCS:

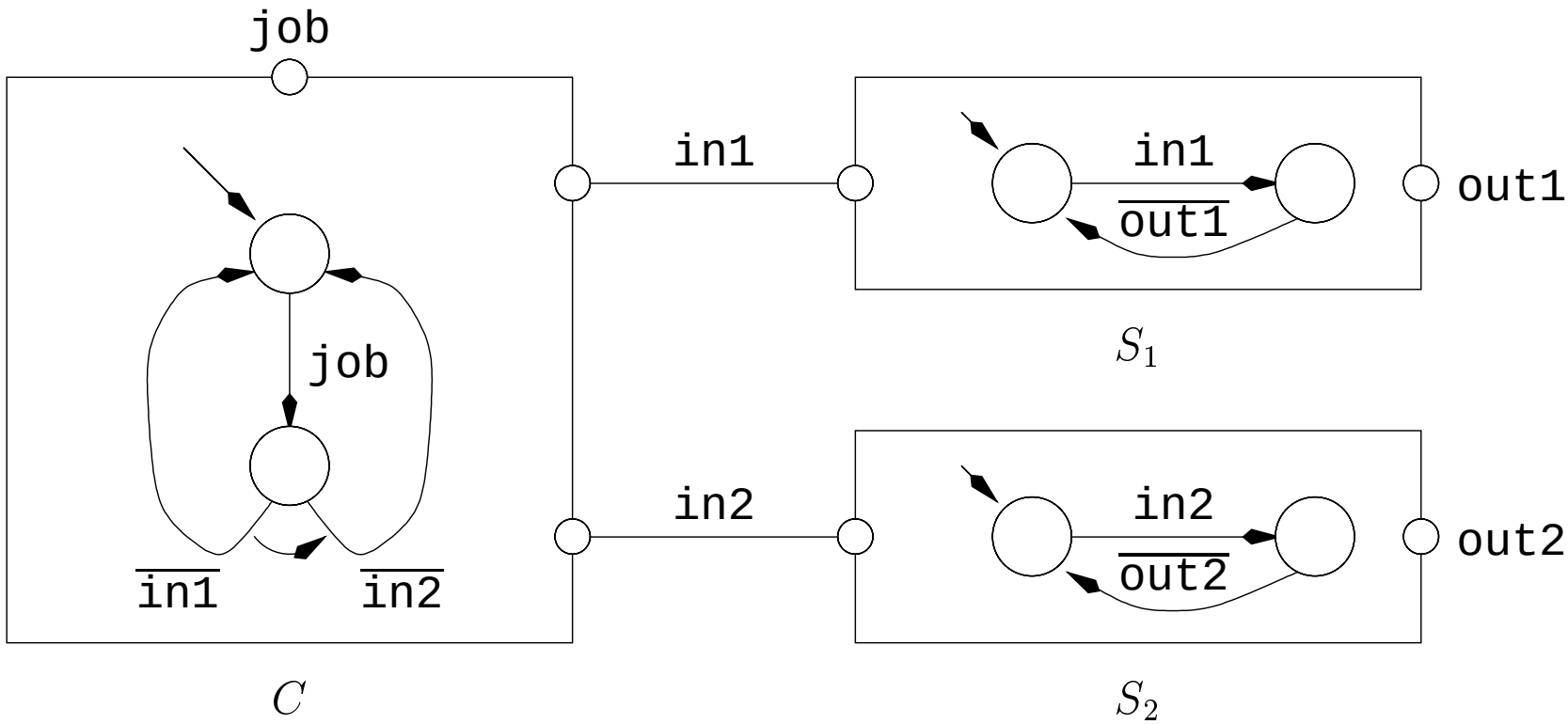
$$
\begin{aligned}
M_2 &\stackrel{\text{def}}{=} (C \mid S) \setminus \{\mathtt{req}, \mathtt{resp}\} \\
C &\stackrel{\text{def}}{=} \mathtt{start}.\overline{\mathtt{req}}.(\overline{\mathtt{to}}.C + \mathtt{resp}.\overline{\mathtt{ok}}.C) \\
S &\stackrel{\text{def}}{=} \mathtt{req}.\overline{\mathtt{run}}.\overline{\mathtt{resp}}.S
\end{aligned}
$$

Potential computation:

$$M_2 \xrightarrow{\mathtt{start}} \xrightarrow{\tau} \xrightarrow{\overline{\mathtt{run}}} \xrightarrow{\overline{\mathtt{to}}} \ldots$$

Time–out happens even though response is ready because priority of time–out is too high!

Motivating Example #3: Ordered Alternatives



C: Controller routing jobs first to server S_1 if it is free, else to S_2.

S_i: Servers, for $i \in \{1, 2\}$.

Example #3 (cont.)

System M_3 in CCS:

$$
\begin{aligned}
M_3 &\stackrel{\text{def}}{=} (C \mid S_1 \mid S_2) \setminus \{\mathtt{in1}, \mathtt{in2}\} \\
C &\stackrel{\text{def}}{=} \mathtt{job}.(\overline{\mathtt{in1}}.C + \overline{\mathtt{in2}}.C) \\
S_i &\stackrel{\text{def}}{=} \mathtt{in}i.\overline{\mathtt{out}i}.S_i
\end{aligned}
$$

Possible computation:

$$M_3 \xrightarrow{\mathtt{job}} \xrightarrow{\tau} \xrightarrow{\overline{\mathtt{out2}}} \ldots$$

Job is sent to S_2 even though S_1 is available, because $\overline{\mathtt{in1}}$ is not given precedence over $\overline{\mathtt{in2}}$.

This Lecture

Goal Enhance theory of CCS by allowing actions to have different priorities.

Outline

- Syntax of $\mathrm{CCS}^{\mathrm{prio}}$: CCS with two–level priority structure on actions
- A structural operational semantics for $\mathrm{CCS}^{\mathrm{prio}}$
- Strong bisimulation and observational congruence for $\mathrm{CCS}^{\mathrm{prio}}$
- An alternative notion of precedence for $\mathrm{CCS}^{\mathrm{prio}}$
- Summary and outlook

Part I

CCS with two–level priority structure on actions

— Syntax and operational semantics —

Syntax of CCS^{prio}: Actions

Fix disjoint sets:

- $(\lambda, \lambda', \lambda_1, \ldots \in)\ \Lambda$ — unprioritized ports
- $(\underline{\lambda}, \underline{\lambda'}, \underline{\lambda_1}, \ldots \in)\ \underline{\Lambda}$ — prioritized ports

and let $\ell \in \Lambda \cup \underline{\Lambda}$. Then an action is either:

- ℓ: input on port ℓ
- $\overline{\ell}$: output on port ℓ
- $\tau, \underline{\tau} \notin \Lambda \cup \underline{\Lambda}$: unprioritized, prioritized internal actions.

Notation

Unprioritized actions: $(a \in) A = \Lambda \cup \overline{\Lambda} \cup \{\tau\}$

Prioritized actions: $(\underline{a} \in) \underline{A} = \underline{\Lambda} \cup \underline{\overline{\Lambda}} \cup \{\underline{\tau}\}$

All actions: $(\alpha \in) \mathcal{A} = A \cup \underline{A}$

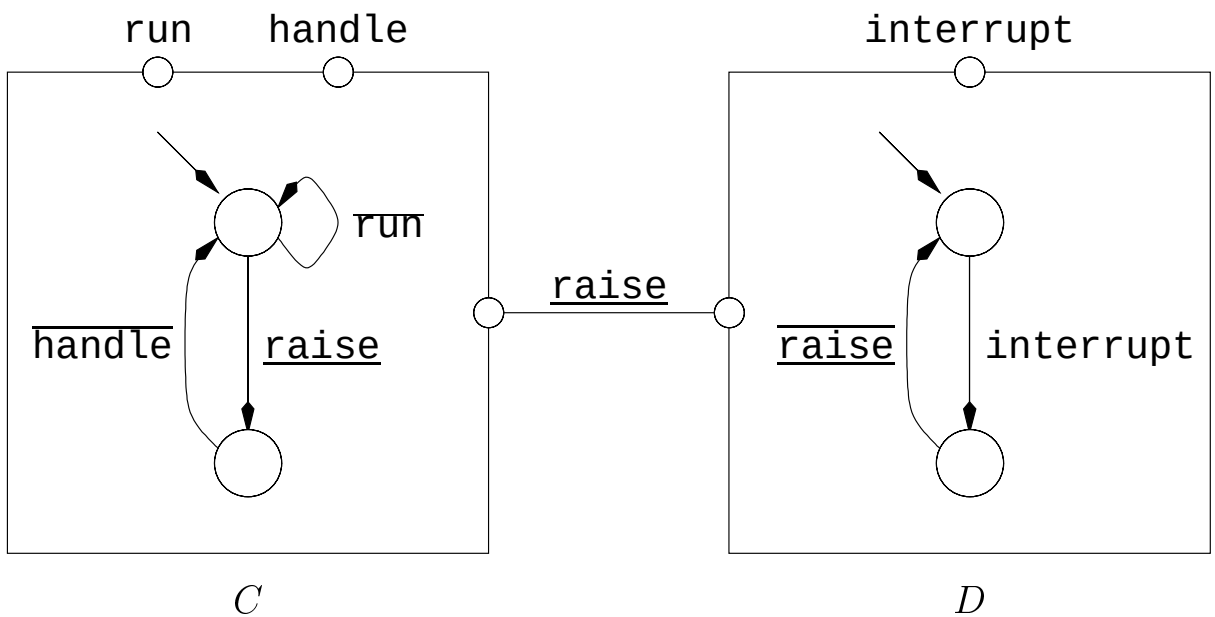
Syntax of CCS^{prio}: Operators

P	$::=$	$\mathbf{0}$	Termination	
	$\mid$	C	Constant	(bounded by $C \stackrel{\text{def}}{=} P$)
	$\mid$	$\alpha.P$	Prefix	($\alpha \in \mathcal{A}$)
	$\mid$	$P+P$	Choice	
	$\mid$	$P \mid P$	Parallel composition	
	$\mid$	$P[f]$	Relabeling	($f \in \mathcal{A} \to \mathcal{A}$ a relabeling)
	$\mid$	$P\backslash L$	Restriction	($L \subseteq \Lambda \cup \underline{\Lambda}$)

Definition $f \in \mathcal{A} \to \mathcal{A}$ is a relabeling if
$f(\tau) = \tau$, $f(\underline{\tau}) = \underline{\tau}$, $f(\overline{\alpha}) = \overline{f(\alpha)}$, $f(a) \in A$, and $f(\underline{a}) \in \underline{A}$.

Notation $(P, P', Q, \ldots \in)\ \mathcal{P}$ denotes the set of processes,
i.e., closed (and guarded) terms.

Recoding Example #1 (Interrupts) in CCS^{prio}

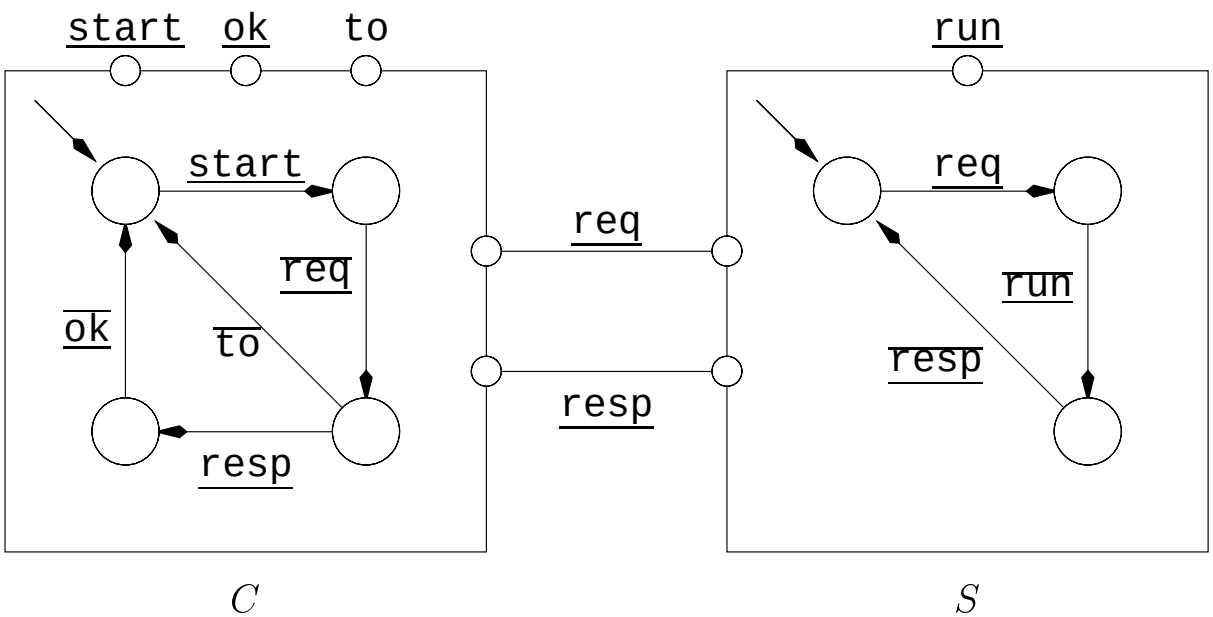


$$M_1 \stackrel{\text{def}}{=} (C \mid D) \setminus \{\underline{\texttt{raise}}\}$$

$$C \stackrel{\text{def}}{=} \overline{\texttt{run}}.C + \underline{\texttt{raise}}.\overline{\texttt{handle}}.C$$

$$D \stackrel{\text{def}}{=} \texttt{interrupt}.\overline{\underline{\texttt{raise}}}.D$$

Recoding Example #2 (Time–outs) in CCS^{prio}

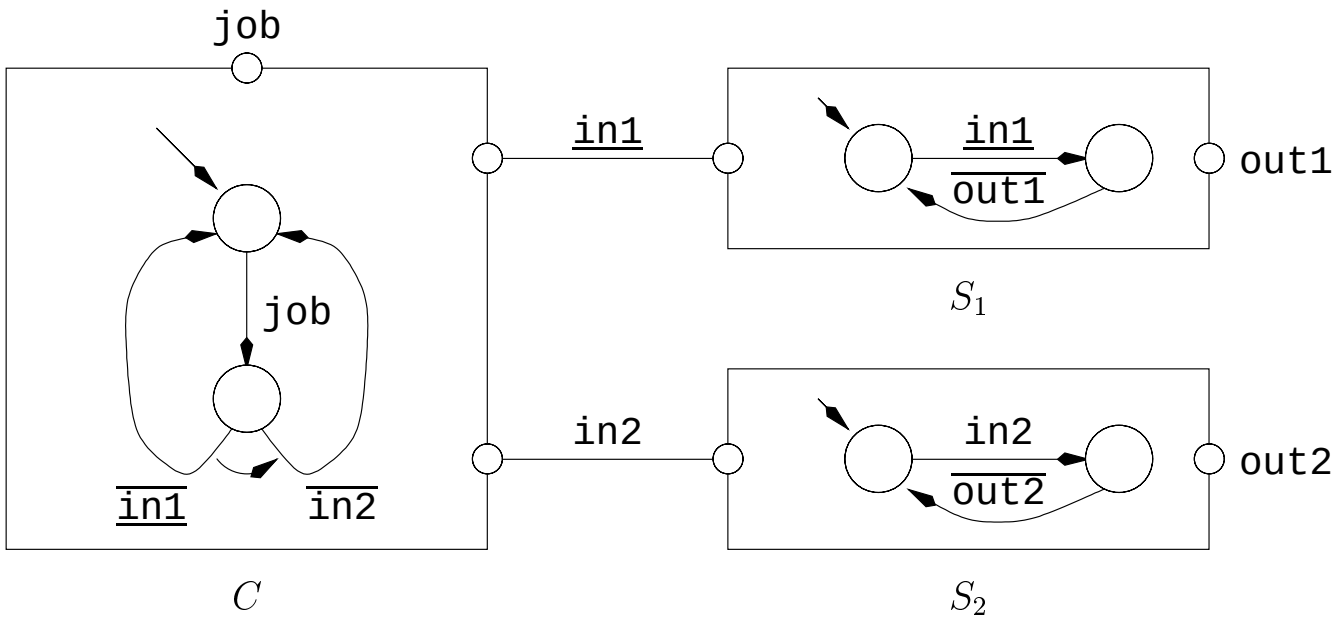


$$M_2 \stackrel{\text{def}}{=} (C \mid S) \setminus \{\underline{\texttt{req}}, \underline{\texttt{resp}}\}$$

$$C \stackrel{\text{def}}{=} \underline{\texttt{start}}.\underline{\overline{\texttt{req}}}.(\overline{\texttt{to}}.C + \underline{\texttt{resp}}.\underline{\overline{\texttt{ok}}}.C)$$

$$S \stackrel{\text{def}}{=} \underline{\texttt{req}}.\underline{\overline{\texttt{run}}}.\underline{\overline{\texttt{resp}}}.S$$

Recoding Example #3 (Ordered Alternatives) in CCS^{prio}



$$M_3 \stackrel{\text{def}}{=} (C \mid S_1 \mid S_2) \setminus \{\underline{\texttt{in1}}, \texttt{in2}\}$$

$$C \stackrel{\text{def}}{=} \texttt{job}.(\overline{\underline{\texttt{in1}}}.C + \overline{\texttt{in2}}.C)$$

$$S_1 \stackrel{\text{def}}{=} \underline{\texttt{in1}}.\overline{\texttt{out1}}.S_1$$

$$S_2 \stackrel{\text{def}}{=} \texttt{in2}.\overline{\texttt{out2}}.S_2$$

Operational Semantics of CCS^{prio}

Standard SOS: We want to define a relation $\longrightarrow \subseteq \mathcal{P} \times \mathcal{A} \times \mathcal{P}$ using rules.

Question When should $P \xrightarrow{\alpha} P'$ hold?

Answer (common) $P \xrightarrow{\alpha} P'$ means " P performs α and evolves to P' ."

Common answer suggests semantics for CCS^{prio} should have following property:

If $P \xrightarrow{\underline{a}} P'$, for some $\underline{a} \in \underline{A}$, then $P \not\xrightarrow{a}$, for any $a \in A$.

But then, e.g., $\overline{\underline{\texttt{in1}}}.C + \overline{\texttt{in2}}.C$ can never perform $\overline{\texttt{in2}}$, and Example #3 will be invalidated!

Solution A more refined understanding of $\longrightarrow$.

A Refined Understanding of $\longrightarrow$

A more precise reading of $P \xrightarrow{\alpha} P'$ in traditional CCS:

$P \xrightarrow{\alpha} P'$ means when α is enabled, P may evolve to P'.

- τ is always enabled.
- If $\alpha \in \Lambda \cup \overline{\Lambda}$, then the environment is responsible for enabling α (by e.g. "pressing buttons"). Thus, $P \xrightarrow{\alpha} P'$ represents a conditional transition.

CCS^{prio} builds on this point of view by only allowing enabled prioritized actions to pre–empt unprioritized transitions:

If $P \xrightarrow{\tau} P'$ then $P \not\xrightarrow{a}$, for any $a \in A$.

The SOS rules for CCS^{prio} are intended to reflect this behavior.

Defining $\longrightarrow$ for CCS^{prio}

Based on previous discussion:

$P \xrightarrow{a} P'$ for some $a \in A$ only if $P \overset{\mathcal{I}}{\not\longrightarrow}$.

One would therefore expect a SOS rule like the following:

$$\frac{P \xrightarrow{a} P', \quad Q \overset{\mathcal{I}}{\not\longrightarrow}}{P + Q \xrightarrow{a} P'} \; a \in A$$

- Rule has a negative premise: $Q \overset{\mathcal{I}}{\not\longrightarrow}$.
- Negative premises are tricky in SOS. [cf., rule formats: Bol & Groote]

Solution (ours) Define prioritized initial action sets, $\underline{\mathcal{I}}(P)$, separately, then use this in SOS.

Defining $\underline{\mathcal{I}}()$

$\underline{\mathcal{I}}(P)$, for $P \in \mathcal{P}$, is the least subset of $\underline{A}$ satisfying the following:

$$
\begin{aligned}
\underline{\mathcal{I}}(\underline{a}.P) &= \{\underline{a}\} \\
\underline{\mathcal{I}}(C) &= \underline{\mathcal{I}}(P) \text{ if } C \stackrel{\text{def}}{=} P \\
\underline{\mathcal{I}}(P+Q) &= \underline{\mathcal{I}}(P) \cup \underline{\mathcal{I}}(Q) \\
\underline{\mathcal{I}}(P \mid Q) &= \underline{\mathcal{I}}(P) \cup \underline{\mathcal{I}}(Q) \cup \{\underline{\tau} \mid \underline{\mathcal{I}}(P) \cap \overline{\underline{\mathcal{I}}(Q)} \neq \emptyset\} \\
\underline{\mathcal{I}}(P[f]) &= \{f(\underline{a}) \mid \underline{a} \in \underline{\mathcal{I}}(P)\} \\
\underline{\mathcal{I}}(P \setminus L) &= \underline{\mathcal{I}}(P) \setminus (L \cup \overline{L})
\end{aligned}
$$

Example $\underline{\mathcal{I}}((a.\mathbf{0} + \underline{b}.\underline{c}.\mathbf{0}) \mid \overline{\underline{b}}.\mathbf{0}) = \{\underline{b}, \overline{\underline{b}}, \underline{\tau}\}$

Rules for $\longrightarrow$

For some operators, the rules for CCS^{prio} mirror those in CCS.

Prefix $$\frac{-}{\alpha.P \xrightarrow{\alpha} P}$$

Rec $$\frac{P \xrightarrow{\alpha} P'}{C \xrightarrow{\alpha} P'} \; C \stackrel{\text{def}}{=} P$$

Rel $$\frac{P \xrightarrow{\alpha} P'}{P[f] \xrightarrow{f(\alpha)} P'[f]}$$

Res $$\frac{P \xrightarrow{\alpha} P'}{P \setminus L \xrightarrow{\alpha} P' \setminus L} \; \alpha \notin L \cup \overline{L}$$

Rules for $\longrightarrow$: Choice

Operator $+$ requires separate rules for prioritized / unprioritized actions ...

$$\underline{\text{Sum1}}\ \frac{P \xrightarrow{\underline{a}} P'}{P + Q \xrightarrow{\underline{a}} P'} \qquad \text{Sum1}\ \frac{P \xrightarrow{a} P'}{P + Q \xrightarrow{a} P'}\ \underline{\tau} \notin \underline{\mathcal{I}}(Q)$$

$$\underline{\text{Sum2}}\ \frac{Q \xrightarrow{\underline{a}} Q'}{P + Q \xrightarrow{\underline{a}} Q'} \qquad \text{Sum2}\ \frac{Q \xrightarrow{a} Q'}{P + Q \xrightarrow{a} Q'}\ \underline{\tau} \notin \underline{\mathcal{I}}(P)$$

Rules for $\longrightarrow$: Parallel Composition

... as does operator $|$.

$$\underline{\text{Com1}}\ \frac{P \xrightarrow{\underline{a}} P'}{P \mid Q \xrightarrow{\underline{a}} P' \mid Q} \qquad \text{Com1}\ \frac{P \xrightarrow{a} P'}{P \mid Q \xrightarrow{a} P' \mid Q}\ \underline{\tau} \notin \underline{\mathcal{I}}(P \mid Q)$$

$$\underline{\text{Com2}}\ \frac{Q \xrightarrow{\underline{a}} Q'}{P \mid Q \xrightarrow{\underline{a}} P \mid Q'} \qquad \text{Com2}\ \frac{Q \xrightarrow{a} Q'}{P \mid Q \xrightarrow{a} P \mid Q'}\ \underline{\tau} \notin \underline{\mathcal{I}}(P \mid Q)$$

$$\underline{\text{Com3}}\ \frac{P \xrightarrow{\underline{a}} P' \quad Q \xrightarrow{\underline{\overline{a}}} Q'}{P \mid Q \xrightarrow{\underline{\tau}} P' \mid Q'} \qquad \text{Com3}\ \frac{P \xrightarrow{a} P' \quad Q \xrightarrow{\overline{a}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}\ \underline{\tau} \notin \underline{\mathcal{I}}(P \mid Q)$$

Metatheorems about CCS^{prio} Semantics

1. The set $\underline{\mathcal{I}}(P)$ indeed contains exactly the prioritized actions in which P can initially engage, i.e.,

$$\underline{a} \in \underline{\mathcal{I}}(P) \text{ if and only if } P \xrightarrow{\underline{a}} .$$

2. The transition relation $\longrightarrow$ encodes the desired notion of pre–emption, i.e.,

$$P \xrightarrow{a}, \text{ for some } a \in A, \text{ implies } P \not\xrightarrow{\tau} .$$

CCS^{prio} Semantics and Example #1

$$M_1 \stackrel{\text{def}}{=} (C \mid D) \setminus \{\underline{\texttt{raise}}\}$$

$$C \stackrel{\text{def}}{=} \overline{\texttt{run}}.C + \underline{\texttt{raise}}.\overline{\texttt{handle}}.C$$

$$D \stackrel{\text{def}}{=} \texttt{interrupt}.\overline{\underline{\texttt{raise}}}.D$$

In the CCS encoding the following bad computation was possible:

$$M_1 \xrightarrow{\overline{\texttt{run}}} \xrightarrow{\texttt{interrupt}} \xrightarrow{\overline{\texttt{run}}} \ldots$$

Can this happen in CCS^{prio}? No!

$$M_1 \xrightarrow{\overline{\texttt{run}}} M_1$$

$$\xrightarrow{\texttt{interrupt}} (C \mid \overline{\underline{\texttt{raise}}}.D) \setminus \{\underline{\texttt{raise}}\}$$

$$\not\xrightarrow{\overline{\texttt{run}}}$$ since $\underline{\tau} \in \underline{\mathcal{I}}((C \mid \overline{\underline{\texttt{raise}}}.D) \setminus \{\underline{\texttt{raise}}\})$!

CCS^{prio} Semantics and Example #2

$$M_2 \stackrel{\text{def}}{=} (C \mid S) \setminus \{\underline{\mathtt{req}}, \underline{\mathtt{resp}}\}$$

$$C \stackrel{\text{def}}{=} \underline{\mathtt{start}}.\underline{\overline{\mathtt{req}}}.(\overline{\mathtt{to}}.C + \underline{\mathtt{resp}}.\overline{\mathtt{ok}}.C)$$

$$S \stackrel{\text{def}}{=} \underline{\mathtt{req}}.\underline{\overline{\mathtt{run}}}.\underline{\overline{\mathtt{resp}}}.S$$

CCS bad computation: $M_2 \xrightarrow{\mathtt{start}} \xrightarrow{\tau} \xrightarrow{\overline{\mathtt{run}}} \xrightarrow{\overline{\mathtt{to}}} \ldots$

Can this happen in CCS^{prio}? No!

$$\begin{aligned}
M_2 &\xrightarrow{\mathtt{start}} (\underline{\overline{\mathtt{req}}}.(\overline{\mathtt{to}}.C + \underline{\mathtt{resp}}.\overline{\mathtt{ok}}.C) \mid S) \setminus \{\underline{\mathtt{req}}, \underline{\mathtt{resp}}\} \\
&\xrightarrow{\tau} ((\overline{\mathtt{to}}.C + \underline{\mathtt{resp}}.\overline{\mathtt{ok}}.C) \mid \underline{\overline{\mathtt{run}}}.\underline{\overline{\mathtt{resp}}}.S) \setminus \{\underline{\mathtt{req}}, \underline{\mathtt{resp}}\} \\
&\xrightarrow{\overline{\mathtt{run}}} ((\overline{\mathtt{to}}.C + \underline{\mathtt{resp}}.\overline{\mathtt{ok}}.C) \mid \underline{\overline{\mathtt{resp}}}.S) \setminus \{\underline{\mathtt{req}}, \underline{\mathtt{resp}}\} \\
&\not\xrightarrow{\overline{\mathtt{to}}} \underline{\tau} \in \underline{\mathcal{I}}(((\overline{\mathtt{to}}.C + \underline{\mathtt{resp}}.\overline{\mathtt{ok}}.C) \mid \underline{\overline{\mathtt{resp}}}.S) \setminus \{\underline{\mathtt{req}}, \underline{\mathtt{resp}}\})
\end{aligned}$$

CCS^{prio} Semantics and Example #3

$$
\begin{aligned}
M_3 &\stackrel{\text{def}}{=} (C \mid S_1 \mid S_2) \setminus \{\underline{\mathtt{in1}}, \mathtt{in2}\} \\
C &\stackrel{\text{def}}{=} \mathtt{job}.(\overline{\underline{\mathtt{in1}}}.C + \overline{\mathtt{in2}}.C) \\
S_1 &\stackrel{\text{def}}{=} \underline{\mathtt{in1}}.\overline{\mathtt{out1}}.S_1 \\
S_2 &\stackrel{\text{def}}{=} \mathtt{in2}.\overline{\mathtt{out2}}.S_2
\end{aligned}
$$

CCS bad computation: $M_3 \xrightarrow{\mathtt{job}} \xrightarrow{\tau} \xrightarrow{\overline{\mathtt{out2}}} \ldots$

Is this possible in CCS^{prio}? No!

$$
\begin{aligned}
M_3 &\xrightarrow{\mathtt{job}} ((\overline{\underline{\mathtt{in1}}}.C + \overline{\mathtt{in2}}.C) \mid S_1 \mid S_2) \setminus \{\underline{\mathtt{in1}}, \mathtt{in2}\} \\
&\xrightarrow{\tau} (C \mid \overline{\mathtt{out1}}.S_1 \mid S_2) \setminus \{\underline{\mathtt{in1}}, \mathtt{in2}\} \\
&\not\xrightarrow{\overline{\mathtt{out2}}}
\end{aligned}
$$

Part II

CCS^{prio}: Bisimulation semantics [Milner, Park]

Bisimulation Equivalences for CCS^{prio}

The definition of $\longrightarrow$ turns CCS^{prio} into a labeled transition system in the usual process–algebraic sense.

Question When should states / terms in this labeled transition system be deemed to behave the same?

Approach (ours) Adapt notions of bisimulation / observational equivalence, with the goal of developing a relation that:

- Takes account of branching structure;
- Abstracts away from internal computation; and
- Is a congruence, i.e.,

  " $P \sim Q$ implies $K[P] \sim K[Q]$, for all CCS^{prio} contexts $K[\cdot]$."

(Strong) Bisimulation Equivalence for CCS^{prio}

First step towards an equivalence: Strong bisimulation which

- Does not abstract with respect to internal computation, but
- Is easier to work with mathematically.

Program

- Define relation.
- Characterize largest congruence if relation is not a congruence already.
- Axiomatize this congruence for finite processes.

Defining Bisimulation Equivalence for CCS^{prio}

Definition A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a (strong) bisimulation if, for all $\langle P, Q \rangle \in \mathcal{R}$ and $\alpha \in \mathcal{A}$:

$P \xrightarrow{\alpha} P'$ implies $Q \xrightarrow{\alpha} Q'$ for some Q' with $\langle P', Q' \rangle \in \mathcal{R}$.

$$\begin{tikzcd}
P \arrow[r, dash] \arrow[d, "\alpha", "\forall"'] & \mathcal{R} \arrow[r, dash] & Q \arrow[d, dashed, "\alpha"', "\exists"] \\
P' \arrow[r, dash, dashed] & \mathcal{R} \arrow[r, dash, dashed] & Q'
\end{tikzcd}$$

Definition (bisimulation equivalence)

$P \simeq Q$ if there exists a strong bisimulation $\mathcal{R}$ such that $\langle P, Q \rangle \in \mathcal{R}$.

Facts about $\simeq$

1. $\simeq$ is the largest bisimulation.
2. $\simeq$ is an equivalence relation.
3. $\simeq$ is a congruence for $\mathrm{CCS}^{\mathrm{prio}}$.

The last fact means that $\simeq$ is its own largest congruence.

Axiomatizing $\simeq$ for Finite Processes

Goal Give set of equational axioms with property that

$$\vdash P = Q \text{ if and only if } P \simeq Q$$

for finite CCS^{prio} processes P and Q (i.e., no recursion).

Many axioms are inherited from CCS, such as

(A1)	$t + u$	$=$	$u + t$
(A2)	$t + (u + v)$	$=$	$(t + u) + v$
(A3)	$t + t$	$=$	t
(A4)	$t + \mathbf{0}$	$=$	t

Axiomatization (cont.)

Other axioms from CCS:

(Res1)	$\mathbf{0} \setminus L$	$=$	$\mathbf{0}$	
(Res2)	$(\alpha.t) \setminus L$	$=$	$\mathbf{0}$	$(\alpha \in L \cup \overline{L})$
(Res3)	$(\alpha.t) \setminus L$	$=$	$\alpha.(t \setminus L)$	$(\alpha \notin L \cup \overline{L})$
(Res4)	$(t + u) \setminus L$	$=$	$(t \setminus L) + (u \setminus L)$	
(Rel1)	$\mathbf{0}[f]$	$=$	$\mathbf{0}$	
(Rel2)	$(\alpha.t)[f]$	$=$	$f(\alpha).(t[f])$	
(Rel3)	$(t + u)[f]$	$=$	$t[f] + u[f]$	

Axiomatization (cont.)

One modified axiom from CCS (expansion law):

(E) Let $t = \sum_i \alpha_i.t_i$ and $u = \sum_j \beta_j.u_j$. Then

$$\begin{aligned} t \mid u \quad = \quad & \textstyle\sum_i \alpha_i.(t_i \mid u) + \\ & \textstyle\sum_j \beta_j.(t \mid u_j) + \\ & \textstyle\sum_{\alpha_i = \overline{\beta_j}} \{\underline{\tau}.(t_i \mid u_j) \mid \alpha_i \in \underline{A}\} + \\ & \textstyle\sum_{\alpha_i = \overline{\beta_j}} \{\tau.(t_i \mid u_j) \mid \alpha_i \in A\} \end{aligned}$$

One new axiom (priority rule; pre–emption):

(P) $\underline{\tau}.t + a.u \quad = \quad \underline{\tau}.t \qquad (a \in A)$

The Axiomatization is Sound and Complete!

For any finite terms P and Q:

Soundness: If $\vdash P = Q$ then $P \simeq Q$.

Completeness: If $P \simeq Q$ then $\vdash P = Q$.

Proofs rely on standard techniques:

- Construction of bisimulations for soundness.
- Definition of normal forms for completeness.

An Observational Congruence for CCS^{prio}

Relation $\simeq$ makes no special accommodation for τ, $\underline{\tau}$; they are treated just like any other action.

To abstract from these internal actions, we will adapt $\simeq$ by introducing a concept of weak transitions.

Program

- Define weak transitions and a weak equivalence relation.
- Characterize the largest congruence if relation is not a congruence already.
- Axiomatize the congruence for finite processes.

In the following, we adapt Milner's definitions for CCS to CCS^{prio}.

Defining (Naive) Weak Transitions

Idea $P \stackrel{\alpha}{\Longrightarrow}_{\times} P'$ should hold whenever

$$P \stackrel{\tau/\underline{\tau}}{\longrightarrow} \cdots \stackrel{\tau/\underline{\tau}}{\longrightarrow} P_1 \stackrel{\alpha}{\longrightarrow} P_2 \stackrel{\tau/\underline{\tau}}{\longrightarrow} \cdots \stackrel{\tau/\underline{\tau}}{\longrightarrow} P' .$$

Definition

1. $\stackrel{\epsilon}{\Longrightarrow}_{\times} = (\stackrel{\tau}{\longrightarrow} \cup \stackrel{\underline{\tau}}{\longrightarrow})^*$
2. $P \stackrel{\alpha}{\Longrightarrow}_{\times} P'$ if $P \stackrel{\epsilon}{\Longrightarrow}_{\times} \stackrel{\alpha}{\longrightarrow} \stackrel{\epsilon}{\Longrightarrow}_{\times} P'$.
3. The visible content $\hat{\alpha}$ of action $\alpha \in \mathcal{A}$ is defined by:

$$\hat{\alpha} = \begin{cases} \epsilon & \text{if } \alpha \in \{\tau, \underline{\tau}\} \\ \alpha & \text{otherwise} \end{cases}$$

Defining (Naive) Observational Equivalence

Definition A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a naive weak bisimulation if, for all $\langle P, Q \rangle \in \mathcal{R}$ and $\alpha \in \mathcal{A}$:

$P \xrightarrow{\alpha} P'$ implies $Q \overset{\hat{\alpha}}{\Longrightarrow}_{\times} Q'$ for some Q' with $\langle P', Q' \rangle \in \mathcal{R}$.

$$
\begin{tikzcd}
P \arrow[r, dash] \arrow[d, "\alpha", "\forall"'] & \mathcal{R} \arrow[r, dash] & Q \arrow[d, dashed, "\hat{\alpha}", "\times"', "\exists"] \\
P' \arrow[r, dashed, dash] & \mathcal{R} \arrow[r, dashed, dash] & Q'
\end{tikzcd}
$$

Definition (naive observational equivalence)

$P \approx_{\times} Q$ if there is a naive weak bisimulation $\mathcal{R}$ with $\langle P, Q \rangle \in \mathcal{R}$.

Facts about $\approx_\times$

1. $\approx_\times$ is the largest naive weak bisimulation.
2. $\approx_\times$ is an equivalence relation.

However, $\approx_\times$ is not a congruence for CCS^{prio}.

- There is the "standard" problem regarding choice, e.g.,

$$\tau.a.\mathbf{0} \approx_\times a.\mathbf{0} \quad \text{but} \quad \tau.a.\mathbf{0} + b.\mathbf{0} \not\approx_\times a.\mathbf{0} + b.\mathbf{0}\,.$$

- There are also problems regarding parallel composition!

Problem #1 Regarding Parallel Composition

$$P_1 \stackrel{\text{def}}{=} \mathbf{0}$$
$$Q_1 \stackrel{\text{def}}{=} \underline{\tau}.Q_1$$
$$R_1 \stackrel{\text{def}}{=} a.\mathbf{0}$$

Here, $P_1 \approx_\times Q_1$, but $P_1 \mid R_1 \not\approx_\times Q_1 \mid R_1$, because

$$P_1 \mid R_1 \xrightarrow{a} P_1 \mid \mathbf{0}, \text{ while } Q_1 \mid R_1 \not\xrightarrow{a}_\times .$$

Source of problem Pre-emption by $\underline{\tau}^\omega$.

Infinite sequences of $\underline{\tau}$–transitions suppress all unprioritized transitions.

Problem #2 Regarding Parallel Composition

$$P_2 \stackrel{\text{def}}{=} a.\mathbf{0} + \tau.(a.\mathbf{0} + \underline{b}.\mathbf{0})$$

$$Q_2 \stackrel{\text{def}}{=} a.\mathbf{0} + \underline{b}.\mathbf{0}$$

$$R_2 \stackrel{\text{def}}{=} \overline{\underline{b}}.\mathbf{0}$$

Here $P_2 \approx_\times Q_2$, but $P_2 \mid R_2 \not\approx_\times Q_2 \mid R_2$, because

$P_2 \mid R_2 \xrightarrow{a} \mathbf{0} \mid R_2$, while $Q_2 \mid R_2 \not\stackrel{a}{\Longrightarrow}_\times$.

Source of problem "Pre–emption potentials"

In some contexts, the $\underline{b}$–transition in Q_2 has the potential of pre–empting the a–transition.

Characterizing the Largest Congruence in $\approx_\times$

$\approx_\times$ is not a congruence for $\mathrm{CCS}^{\mathrm{prio}}$, but it does reflect reasonable intuitions about abstracting from internal behavior.

To obtain a compositional theory, we therefore want to define a congruence that is "as close as possible" to $\approx_\times$.

Approach Two stages:

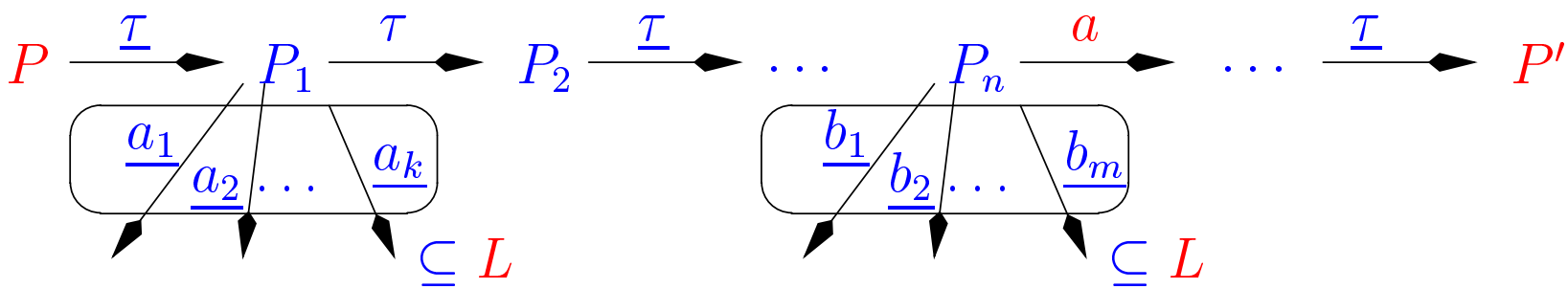
- Fix the problems with parallel composition.
- Fix the problem with choice.

Fixing the “Pre–emption Potentials” Problem with $|$

Idea

- Redefine $\Longrightarrow_{\times}$ to incorporate information about pre–emption potentials.
- Use this new weak transition relation in defining observational equivalence.

New relation $P \overset{a}{\underset{L}{\Longrightarrow}} P'$, for $a \in A$ and $L \subseteq \underline{A} \setminus \{\underline{\tau}\}$:



Note If $P \overset{a}{\underset{L}{\Longrightarrow}} P'$ and the environment does not communicate on any action in L, then the transition sequence from P to P' cannot be pre–empted!

Defining $\overset{a}{\underset{L}{\Longrightarrow}}$ and $\overset{\underline{a}}{\Longrightarrow}$

Let $L \subseteq \underline{A}$. Then

1. $P \overset{a}{\underset{L}{\longrightarrow}} P'$ if $P \overset{a}{\longrightarrow} P'$ and $\underline{\mathcal{I}}(P) \setminus \{\underline{\tau}\} \subseteq L$
2. $\overset{\epsilon}{\underset{L}{\Longrightarrow}} = (\overset{\tau}{\underset{L}{\longrightarrow}} \cup \overset{\underline{\tau}}{\longrightarrow})^*$; $\quad \overset{\underline{\epsilon}}{\Longrightarrow} = (\overset{\underline{\tau}}{\longrightarrow})^*$
3. $P \overset{a}{\underset{L}{\Longrightarrow}} P'$ if $P \overset{\epsilon}{\underset{L}{\Longrightarrow}} \overset{a}{\underset{L}{\longrightarrow}} \overset{\underline{\epsilon}}{\Longrightarrow} P'$
4. $P \overset{\underline{a}}{\Longrightarrow} P'$ if $P \overset{\underline{\epsilon}}{\Longrightarrow} \overset{\underline{a}}{\longrightarrow} \overset{\underline{\epsilon}}{\Longrightarrow} P'$
5. $\hat{\alpha} = \begin{cases} \epsilon & \text{if } \alpha = \tau \\ \underline{\epsilon} & \text{if } \alpha = \underline{\tau} \\ \alpha & \text{otherwise} \end{cases}$

Fixing $\approx_\times$ for Parallel Composition

Definition A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a prioritized weak bisimulation if, for all $\langle P, Q\rangle \in \mathcal{R}$, $\underline{a} \in \underline{A}$, $a \in A$, the following holds:

1. If $\underline{\tau} \notin \underline{\mathcal{I}}(P)$ then $Q \underset{\underline{\mathcal{I}}(P)}{\overset{\epsilon}{\Longrightarrow}} Q'$ for some Q' with $\underline{\tau} \notin \underline{\mathcal{I}}(Q')$, $\langle P, Q'\rangle \in \mathcal{R}$, and $\underline{\mathcal{I}}(Q') \setminus \{\underline{\tau}\} \subseteq \underline{\mathcal{I}}(P)$.
2. $P \xrightarrow{\underline{a}} P'$ implies $Q \overset{\hat{\underline{a}}}{\Longrightarrow} Q'$ for some Q' with $\langle P', Q'\rangle \in \mathcal{R}$.
3. $P \xrightarrow{a} P'$ implies $Q \underset{\underline{\mathcal{I}}(P)}{\overset{\hat{a}}{\Longrightarrow}} Q'$ for some Q' with $\langle P', Q'\rangle \in \mathcal{R}$.

Intuition: Condition 1 fixes problem with τ^ω, while Condition 3 fixes problem related to pre–emption potentials.

Definition (prioritized observational equivalence)

$P \underline{\approx} Q$ if there is a prioritized weak bisimulation $\mathcal{R}$ with $\langle P, Q\rangle \in \mathcal{R}$.

Examples

1. Recall $P_1 \stackrel{\text{def}}{=} \mathbf{0}$ and $Q_1 \stackrel{\text{def}}{=} \underline{\tau}.Q_1$.
   $P_1 \not\underline{\approx} Q_1$ since $\underline{\tau} \notin \underline{\mathcal{I}}(P_1)$ and $\underline{\tau} \in \underline{\mathcal{I}}(Q')$ for every Q' s.t. $Q_1 \stackrel{\epsilon}{\underset{\emptyset}{\Longrightarrow}} Q'$.

2. Recall $P_2 \stackrel{\text{def}}{=} a.\mathbf{0} + \tau.(a.\mathbf{0} + \underline{b}.\mathbf{0})$ and $Q_2 \stackrel{\text{def}}{=} a.\mathbf{0} + \underline{b}.\mathbf{0}$.
   Then $P_2 \not\underline{\approx} Q_2$ since $P_2 \stackrel{a}{\longrightarrow} \mathbf{0}$ but $Q_2 \stackrel{a}{\underset{\emptyset}{\not\Longrightarrow}} (\emptyset = \underline{\mathcal{I}}(P_2))$.

3. Note that $P_2 + \underline{b}.\mathbf{0} \underline{\approx} Q_2$. In particular, $P_2 + \underline{b}.\mathbf{0} \stackrel{a}{\longrightarrow} \mathbf{0}$ can be matched by $Q_2 \stackrel{a}{\underset{\{\underline{b}\}}{\Longrightarrow}} \mathbf{0}$ $(\underline{\mathcal{I}}(P_2 + \underline{b}.\mathbf{0}) = \{\underline{b}\})$.

Facts about $\underline{\approx}$

1. $\underline{\approx}$ is the largest prioritized weak bisimulation.
2. $\underline{\approx}$ is an equivalence relation contained in $\approx_{\times}$.
3. $\underline{\approx}$ is preserved by all CCS^{prio} operators except choice.
4. The largest congruence $\approx^{\mathrm{l}}_{\times}$ of $\approx_{\times}$ wrt. CCS^{prio} is contained within $\underline{\approx}$.

   The proof requires the identification of special CCS^{prio} contexts; its utility is emphasized by the following fact from universal algebra:

   > Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be equivalences in an algebra such that $\mathcal{R}_1^l \subseteq \mathcal{R}_2 \subseteq \mathcal{R}_1$. Then $\mathcal{R}_1^l = \mathcal{R}_2^l$.

Fixing the Problem with Choice

$\underline{\approx}$ is not preserved by choice:

$$\tau.a.\mathbf{0} \underline{\approx} a.\mathbf{0}, \quad \text{and yet} \quad \tau.a.\mathbf{0} + b.\mathbf{0} \not\underline{\approx} a.\mathbf{0} + b.\mathbf{0}\,.$$

Fortunately, the "standard fix" for choice works, i.e., require initial internal computation steps to be matched by "nontrivial" sequences of internal computation steps.

Definition $P \underline{\approx}^l Q$ if the following, and their symmetric counterparts, hold.

1. $P \xrightarrow{\underline{a}} P'$ implies $Q \overset{\underline{a}}{\Longrightarrow} Q'$ for some Q' with $P' \underline{\approx} Q'$.
2. $P \xrightarrow{a} P'$ implies $Q \underset{\underline{\mathcal{I}}(P)}{\overset{a}{\Longrightarrow}} Q'$ for some Q' with $P' \underline{\approx} Q'$.

Remarks on the Semantic Theory of $\underline{\approx}^l$

Theorem $\underline{\approx}^l$ is the largest congruence contained in $\approx_\times$.

Axiomatizing $\underline{\approx}^l$ ... is technically complex, even for finite processes.

- Handbook chapter contains sound and complete axiomatizations of observational congruence for $\mathrm{CCS}^{\mathrm{prio}}$ extended with prioritization and deprioritization operators.
- Equations require introduction of an auxiliary relation, dealing with prioritized initial action sets, that is also characterized axiomatically.

Variants of CCS^{prio}

Operators changing priority levels

- Example: Relabelings permitted to change priority values, e.g., $f(a) = \underline{b}$.
- Implications of prioritization and deprioritization operators on semantic theory have been studied by Cleaveland and Hennessy.

Multi–level priority structure

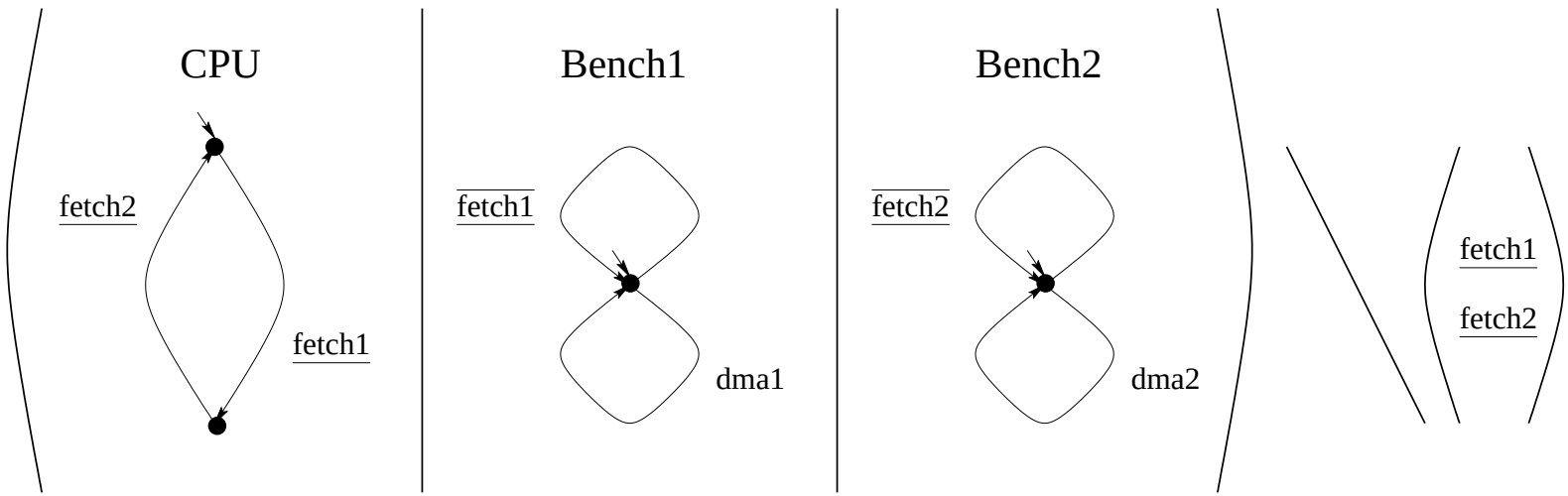
- Example: $(b.\mathbf{0} + \underline{a}.\mathbf{0}) \mid (\overline{\underline{a}}.\mathbf{0} + \underline{\underline{c}}.\mathbf{0}) \mid \overline{\underline{\underline{c}}}$, i.e.,
  transitions labeled by b, $\underline{a}$, $\overline{\underline{a}}$, and $\underline{\tau}$ are pre–empted by $\underline{\underline{\tau}}$.
- Semantic theory can be adapted; no surprises.

Part III

A different concept of pre–emption for CCS^{prio}

Is the Pre–emption Concept in CCS^{prio} Always Adequate?

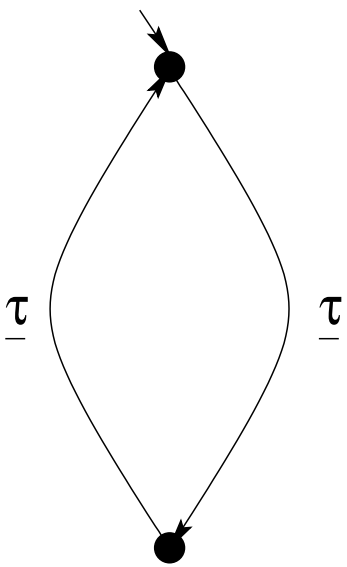
Example A direct–memory–access (DMA) system



Intuitively, if CPU and Bench$i$, for some $i \in \{1, 2\}$, can communicate then

- Action $\mathtt{dma}i$ should be pre–empted by this prioritized communication.
- Action $\mathtt{dma}j$, for $j \neq i$, should be offered to the environment.

CCS^{prio} Semantics of the DMA System



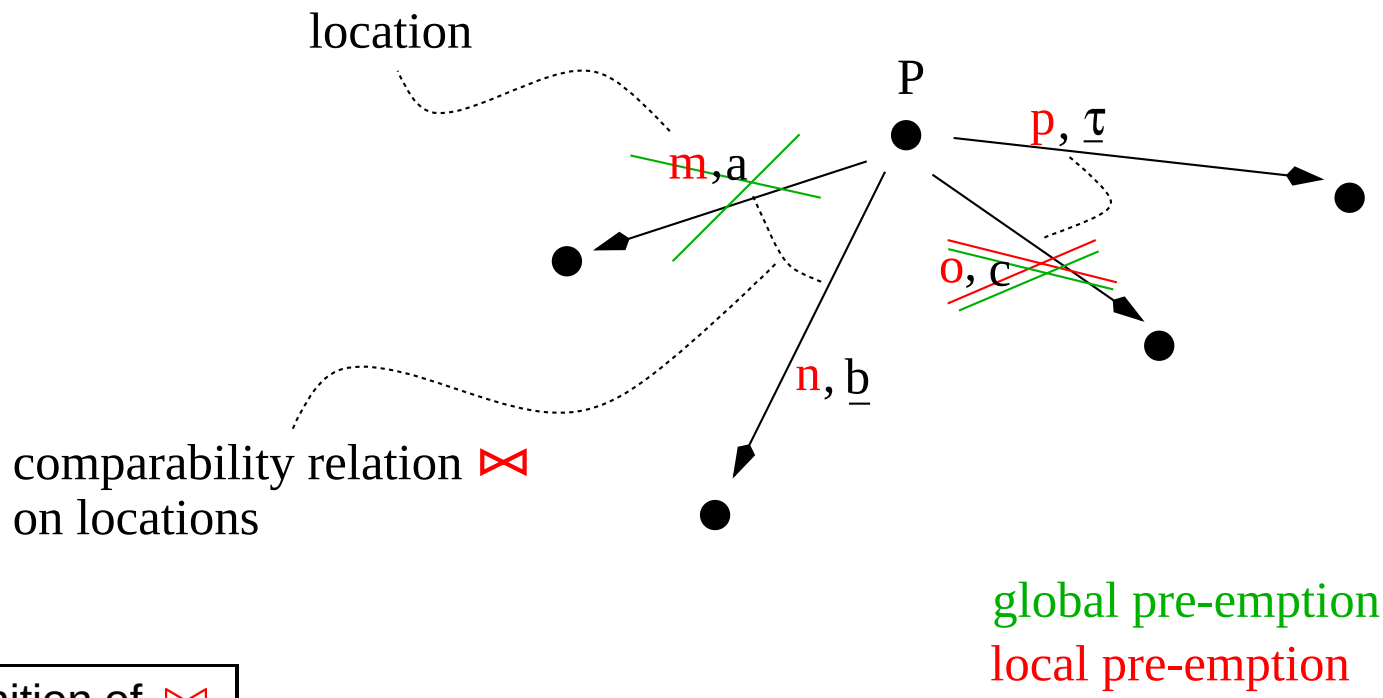
Actions dma1 and dma2 are always pre–empted!

Hence

- Global pre–emption is sometimes too strong.
- Take locations into account, specifying the subterm(s) which is (are) responsible for a transition.

$\Longrightarrow$ Local pre–emption

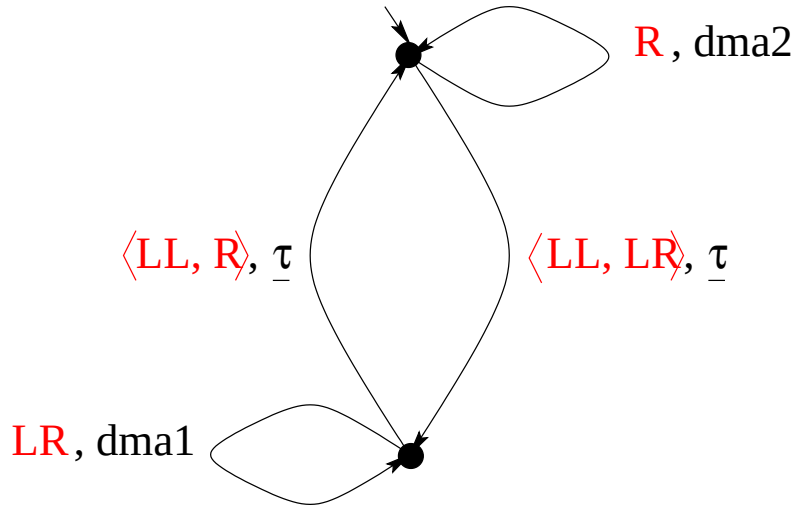
Global vs. Local Pre–emption



Definition of ⋈

- Locations on different sides of $+$ are comparable.
- Locations on different sides of $|$ are incomparable.

DMA System Revisited



- Desired operational behavior because of local pre–emption.
- Introducing a distributed view of priority.

Bisimulation and Local Pre–emption

... naive adaptation of (strong) bisimulation $\simeq$:

$$\begin{tikzcd}
P \arrow[r, phantom, "\simeq"] \arrow[d, "\mathrm{m},\alpha", "\forall"'] & Q \arrow[d, "\mathrm{n},\alpha"', "\exists"] \\
P' \arrow[r, phantom, "\simeq"] & Q'
\end{tikzcd}$$

Problem $\simeq$ is not a congruence, e.g.,

$$a.\underline{b}.\mathbf{0} + \underline{b}.a.\mathbf{0} \simeq a.\mathbf{0} \,|\, \underline{b}.\mathbf{0} \quad \text{but} \quad (a.\underline{b}.\mathbf{0} + \underline{b}.a.\mathbf{0}) \,|\, \overline{\underline{b}}.\mathbf{0} \not\simeq (a.\mathbf{0} \,|\, \underline{b}.\mathbf{0}) \,|\, \overline{\underline{b}}.\mathbf{0}$$

i.e., the standard expansion theorem is invalid here!

Achieving Compositionality

Take local pre–emption into account by defining a finer bisimulation $\underline{\simeq}^{\mathrm{l}}$.

$$\begin{tikzcd} P \arrow[d, "\mathrm{m},\underline{\alpha}"'] & \underline{\sim}^{\mathrm{l}} & Q \arrow[d, "\mathrm{n},\underline{\alpha}"'] \\ P' & \underline{\sim}^{\mathrm{l}} & Q' \end{tikzcd} \qquad \begin{tikzcd} P \arrow[d, "\mathrm{m},\alpha"'] & \underline{\sim}^{\mathrm{l}} & Q \arrow[d, "\mathrm{n},\alpha"'] \\ P' & \underline{\sim}^{\mathrm{l}} & Q' \end{tikzcd}$$

and $\underline{\mathcal{I}}_{[n]}(Q) \setminus \{\underline{\tau}\} \subseteq \underline{\mathcal{I}}_{[m]}(P)$

$\underline{\mathcal{I}}_{[m]}(P) = \{\underline{a} \mid \exists o \bowtie m.\ P \xrightarrow{o,\underline{a}}\}$ is a local prioritized initial action set.

Theorem: $\underline{\simeq}^{\mathrm{l}}$ is the largest congruence contained in $\underline{\simeq}$.

Extension to Multi–Level Priority Structures

Is not quite as easy as for global pre–emption...

Problem When reading, e.g., the process $(b.\mathbf{0} + \underline{a}.\mathbf{0}) \mid (\overline{\underline{a}}.\mathbf{0} + \underline{\underline{c}}.\mathbf{0}) \mid \overline{\underline{\underline{c}}}$ in a

- Right–associative way: $\overline{\underline{a}}$ is definitely pre–empted by the communication on $\underline{\underline{c}}$, whence b is enabled.
- Left–associative way: b might or might not be pre–empted, since it is initially not clear whether a communication on $\underline{a}$ is actually possible.

Possible solution Require all initial actions in the scope of an occurrence of $+$ to be input actions, i.e., prioritized choice only over input actions.

[cf., Camilleri & Winskel's priority calculus]

Lecture Summary

Priority in process algebra

- Helps one to model certain aspects of system behavior more accurately, such as interrupts and time–outs.
- Allows one to encode different levels of precedence among transitions.
- Restricts nondeterminism via prioritized choices.
  [cf., the PRIALT construct in the programming language occam]
- Relies on the technical concept of pre–emption.

Classification of Approaches to Priority

Dimension discussed in this lecture | Global pre–emption vs. local pre–emption

- Global pre–emption: [cf., Baeten, Bergstra & Klop, and Cleaveland & Hennessy]
  - Priority values have a global scope.
  - For modeling centralized systems.
- Local pre–emption: [cf., Camilleri & Winskel]
  - Priority values have a local (component–based) scope.
  - For modeling distributed systems.

Classification (cont.)

Another dimension Static priority vs. dynamic priority

- Static priority: [discussed in this lecture]
  - Priority values of transitions do not change during computations.
  - For modeling interrupts and prioritized choice constructs.
- Dynamic priority: [included in the handbook chapter]
  - Priority values may change as systems evolve.
  - For modeling some scheduling algorithms and real–time semantics.

Some Open Issues

Specific Axiomatizations of prioritized observational congruences

- (Partly) done for finite processes.

  [cf., work by Natarajan and Jensen wrt. global and local pre–emption, respectively]

- Studied in restricted settings for regular processes.

  [cf., work by Gorrieri & Bravetti, and Hermanns & Lorey]

General

- Priority and non–interleaving semantics (e.g., asynchr. transition systems).
- Approaches to priority in calculi for mobility (e.g., the pi–calculus).

Thanks!