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.

In Handbook of Process Algebra, ch. 12, pp. 711–765.

©2001 G. Lüttgen and R. Cleaveland
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.
System $M_1$ may be modeled in CCS as follows:

\[
M_1 \overset{\text{def}}{=} (C \mid D) \setminus \{\text{raise}\}
\]

\[
C \overset{\text{def}}{=} \text{run}.C + \text{raise}.\text{handle}.C
\]

\[
D \overset{\text{def}}{=} \text{interrupt}.\text{raise}.D
\]

Possible computation:

\[
M_1 \overset{\text{run}}{\rightarrow} \overset{\text{interrupt}}{\rightarrow} \overset{\text{run}}{\rightarrow} \ldots
\]

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:

\[
M_2 \overset{\text{def}}{=} (C \mid S) \setminus \{\text{req, resp}\}
\]

\[
C \overset{\text{def}}{=} \text{start.}\overline{\text{req.}}(\overline{\text{to.}}C + \text{resp.}\overline{\text{ok.}}C)
\]

\[
S \overset{\text{def}}{=} \text{req.}\overline{\text{run.}}\text{resp.}S
\]

Potential computation:

\[
M_2 \xrightarrow{\text{start}} \xrightarrow{\text{run}} \xrightarrow{\text{to}} \ldots
\]

Time–out happens even though response is ready because priority of time–out is too high!
Motivating Example #3: Ordered Alternatives

\[ C: \text{Controller routing jobs first to server } S_1 \text{ if it is free, else to } S_2. \]

\[ S_i: \text{Servers, for } i \in \{1, 2\}. \]
Example #3 (cont.)

System $M_3$ in CCS:

\[
M_3 \overset{\text{def}}{=} (C | S_1 | S_2) \setminus \{\text{in1, in2}\}
\]

\[
C \overset{\text{def}}{=} \text{job.}(\text{in1}.C + \text{in2}.C)
\]

\[
S_i \overset{\text{def}}{=} \text{ini}.\text{out}i.S_i
\]

Possible computation:

\[
M_3 \overset{\text{job}}{\Rightarrow} \overset{\text{out2}}{\Rightarrow} \ldots
\]

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

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

Outline

- Syntax of $\text{CCS}^{\text{prio}}$: CCS with two-level priority structure on actions
- A structural operational semantics for $\text{CCS}^{\text{prio}}$
- Strong bisimulation and observational congruence for $\text{CCS}^{\text{prio}}$
- An alternative notion of precedence for $\text{CCS}^{\text{prio}}$
- 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
- $(\lambda, \lambda', \lambda_1, \ldots \in) \Lambda$ — prioritized ports

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

- $\ell$: input on port $\ell$
- $\overline{\ell}$: output on port $\ell$
- $\tau, \overline{\tau} \not\in \Lambda \cup \Lambda$: unprioritized, prioritized internal actions.

**Notation**

Unprioritized actions: $\langle a \in \rangle A = \Lambda \cup \overline{\Lambda} \cup \{\tau\}$

Prioritized actions: $\langle a \in \rangle \overline{A} = \Lambda \cup \overline{\Lambda} \cup \{\tau\}$

All actions: $(\alpha \in) A = A \cup \overline{A}$
Syntax of CCS^{prio}: Operators

\[ P ::= 0 \quad \text{Termination} \]
\[ |\quad C \quad \text{Constant} \quad (\text{bounded by } C \overset{\text{def}}{=} P) \]
\[ |\quad \alpha.P \quad \text{Prefix} \quad (\alpha \in \mathcal{A}) \]
\[ |\quad P + P \quad \text{Choice} \]
\[ |\quad P \parallel P \quad \text{Parallel composition} \]
\[ |\quad P[f] \quad \text{Relabeling} \quad (f \in \mathcal{A} \rightarrow \mathcal{A} \text{ a relabeling}) \]
\[ |\quad P \setminus L \quad \text{Restriction} \quad (L \subseteq \Lambda \cup \Lambda) \]

**Definition**

\[ f \in \mathcal{A} \rightarrow \mathcal{A} \text{ is a relabeling if} \]
\[ f(\tau) = \tau, \quad f(\tau) = \tau, \quad f(\alpha) = \overline{f(\alpha)}, \quad f(a) \in A, \text{ and } f(a) \in A. \]

**Notation**

\( (P, P', Q, \ldots \in) \mathcal{P} \) denotes the set of processes,

i.e., closed (and guarded) terms.
Recoding Example #1 (Interrupts) in $\text{CCS}^{\text{prio}}$

\[
M_1 \overset{\text{def}}{=} (C \mid D) \setminus \{\text{raise}\}
\]

\[
C \overset{\text{def}}{=} \text{run}.C + \text{raise}.\text{handle}.C
\]

\[
D \overset{\text{def}}{=} \text{interrupt}.\text{raise}.D
\]
Recoding Example #2 (Time-outs) in CCS$^\text{prio}$

\[
M_2 \overset{\text{def}}{=} (C \parallel S) \setminus \{\text{req, resp}\}
\]

\[
C \overset{\text{def}}{=} \text{start} . \text{req} . (\text{to} . C + \text{resp} . \text{ok} . C)
\]

\[
S \overset{\text{def}}{=} \text{req} . \text{run} . \text{resp} . S
\]
Recoding Example #3 (Ordered Alternatives) in CCS\(_{\text{prio}}\)

\[
M_3 \overset{\text{def}}{=} (C \mid S_1 \mid S_2) \setminus \{\text{in1, in2}\}
\]

\[
C \overset{\text{def}}{=} \text{job}.(\text{in1}.C + \text{in2}.C)
\]

\[
S_1 \overset{\text{def}}{=} \text{in1}.\text{out1}.S_1
\]

\[
S_2 \overset{\text{def}}{=} \text{in2}.\text{out2}.S_2
\]
Operational Semantics of $\text{CCS}^{\text{prio}}$

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

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

**Answer (common)**
$P \xrightarrow{\alpha} P'$ means “$P$ performs $\alpha$ and evolves to $P'$.”

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

If $P \xrightarrow{\alpha} P'$, for some $a \in A$, then $P \xrightarrow{a}$, for any $a \in A$.

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

**Solution**
A more refined understanding of $\xrightarrow{}$. 
A Refined Understanding of \( \rightarrow \)

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

\[ P \xrightarrow{\alpha} P' \text{ means when } \alpha \text{ is enabled, } P \text{ may evolve to } P'. \]

- \( \tau \) is always enabled.

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

\( \text{CCSprio} \) builds on this point of view by only allowing enabled prioritized actions to pre-empt unprioritized transitions:

\[ \text{If } P \xrightarrow{\tau} P' \text{ then } P \xrightarrow{\alpha}, \text{ for any } \alpha \in A. \]

The SOS rules for \( \text{CCSprio} \) are intended to reflect this behavior.
Based on previous discussion:

\[ P \xrightarrow{a} P' \text{ for some } a \in A \text{ only if } P \xrightarrow{\not a} . \]

One would therefore expect a SOS rule like the following:

\[
\frac{P \xrightarrow{a} P', \ Q \xrightarrow{\not a} \quad a \in A}{P + Q \xrightarrow{a} P'}
\]

- Rule has a negative premise: \( Q \xrightarrow{\not a} \).
- Negative premises are tricky in SOS. [cf., rule formats: Bol & Groote]

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

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

\[
\begin{align*}
\mathcal{I}(a.P) &= \{a\} \\
\mathcal{I}(C) &= \mathcal{I}(P) \text{ if } C \overset{\text{def}}{=} P \\
\mathcal{I}(P + Q) &= \mathcal{I}(P) \cup \mathcal{I}(Q) \\
\mathcal{I}(P | Q) &= \mathcal{I}(P) \cup \mathcal{I}(Q) \cup \{r | \mathcal{I}(P) \cap \overline{\mathcal{I}(Q)} \neq \emptyset\} \\
\mathcal{I}(P[f]) &= \{f(a) | a \in \mathcal{I}(P)\} \\
\mathcal{I}(P \setminus L) &= \mathcal{I}(P) \setminus (L \cup \overline{L})
\end{align*}
\]

Example $\mathcal{I}((a.0 + b.c.0) | \overline{b}.0) = \{b, \overline{b}, r\}$
For some operators, the rules for $\text{CCS}^{\text{prio}}$ mirror those in CCS.

Prefix: $\alpha.P \xrightarrow{\alpha} P$

Rec: $P \xrightarrow{\alpha} P'$, $C \overset{\text{def}}{=} P$

Rel: $P \xrightarrow{\alpha} P'$, $P[f] \overset{f(\alpha)}{\rightarrow} P'[f]$

Res: $P \xrightarrow{\alpha} P'$, $P \setminus L \xrightarrow{\alpha} P' \setminus L$, $\alpha \notin L \cup \overline{L}$

©2001 G. Lüttgen and R. Cleaveland
Rules for \( \rightarrow: \) Choice

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

\[
\begin{align*}
\text{Sum1} & \quad \frac{P \xrightarrow{a} P'}{P + Q \xrightarrow{a} P'} \\
\text{Sum2} & \quad \frac{Q \xrightarrow{a} Q'}{P + Q \xrightarrow{a} Q'}
\end{align*}
\]

\[
\begin{align*}
\text{Sum1} & \quad \frac{P \xrightarrow{a} P'}{P + Q \xrightarrow{a} P'} \quad \tau \notin \mathcal{I}(Q) \\
\text{Sum2} & \quad \frac{Q \xrightarrow{a} Q'}{P + Q \xrightarrow{a} Q'} \quad \tau \notin \mathcal{I}(P)
\end{align*}
\]
Rules for \( P \rightarrow \): Parallel Composition

\[
\begin{align*}
\text{Com1} & : 
\begin{array}{c}
P \xrightarrow{a} P' \\
P | Q \xrightarrow{a} P' | Q
\end{array} \\
\text{Com2} & : 
\begin{array}{c}
Q \xrightarrow{a} Q' \\
P | Q \xrightarrow{a} P | Q'
\end{array} \\
\text{Com3} & : 
\begin{array}{c}
P \xrightarrow{a} P' \quad Q \xrightarrow{\overline{a}} Q' \\
P | Q \xrightarrow{\tau} P' | Q'
\end{array}
\]

\[
\begin{align*}
\text{Com1} & : 
\begin{array}{c}
P \xrightarrow{a} P' \\
P | Q \xrightarrow{a} P' | Q \quad \tau \notin \mathcal{I}(P | Q)
\end{array} \\
\text{Com2} & : 
\begin{array}{c}
Q \xrightarrow{a} Q' \\
P | Q \xrightarrow{a} P | Q' \quad \tau \notin \mathcal{I}(P | Q)
\end{array} \\
\text{Com3} & : 
\begin{array}{c}
P \xrightarrow{a} P' \quad Q \xrightarrow{\overline{a}} Q' \\
P | Q \xrightarrow{\tau} P' | Q' \quad \tau \notin \mathcal{I}(P | Q)
\end{array}
\]

... as does operator \( | \).
Metatheorems about $\text{CCS}^{\text{prio}}$ Semantics

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

   \[ a \in \mathcal{I}(P) \text{ if and only if } P \xrightarrow{a} \, . \]

2. The transition relation $\xrightarrow{-}$ encodes the desired notion of pre-emption, i.e.,

   \[ P \xrightarrow{a}, \text{ for some } a \in A, \text{ implies } P \xrightarrow{\tau} \, . \]
**CCSprio Semantics and Example #1**

\[
\begin{align*}
  M_1 & \overset{\text{def}}{=} (C \mid D) \setminus \text{\{raise\}} \\
  C & \overset{\text{def}}{=} \text{run}.C + \text{raise}.\text{handle}.C \\
  D & \overset{\text{def}}{=} \text{interrupt.\text{\underline{raise}}}.D
\end{align*}
\]

In the CCS encoding the following bad computation was possible:

\[
M_1 \xrightarrow{\text{run}} \text{interrupt} \xrightarrow{\text{run}} \ldots
\]

Can this happen in CCSprio? No!

\[
\begin{align*}
  M_1 & \xrightarrow{\text{run}} M_1 \\
  & \xrightarrow{\text{interrupt}} (C \mid \text{\underline{raise}}.D) \setminus \text{\{raise\}} \\
  & \xrightarrow{\text{run}} \text{since } \tau \in \mathcal{I}((C \mid \text{\underline{raise}}.D) \setminus \text{\{raise\}}))
\end{align*}
\]
\[ M_2 \overset{\text{def}}{=} (C | S) \setminus \{\text{req, resp}\} \]

\[ C \overset{\text{def}}{=} \text{start.req.}(\text{to.C + resp.ok.C}) \]

\[ S \overset{\text{def}}{=} \text{req.run.resp} . S \]

**CCS bad computation:** \[ M_2 \xrightarrow{\text{start}} \xrightarrow{\text{run}} \xrightarrow{\text{to}} \ldots \]

Can this happen in CCS\textsuperscript{prio}? No!

\[ M_2 \xrightarrow{\text{start}} (\text{req.}(\text{to.C + resp.ok.C}) | S) \setminus \{\text{req, resp}\} \]

\[ \xrightarrow{\text{run}} ((\text{to.C + resp.ok.C}) | \text{run.resp} . S) \setminus \{\text{req, resp}\} \]

\[ \xrightarrow{\text{run}} ((\text{to.C + resp.ok.C}) | \text{resp.S}) \setminus \{\text{req, resp}\} \]

\[ \xrightarrow{\text{run}} \tau \in \mathcal{I}(((\text{to.C + resp.ok.C}) | \text{resp.S}) \setminus \{\text{req, resp}\}) \]
CCSprio Semantics and Example #3

\[ M_3 \overset{\text{def}}{=} (C \mid S_1 \mid S_2) \setminus \{\text{in}_1, \text{in}_2\} \]

\[ C \overset{\text{def}}{=} \text{job.}(\overline{\text{in}_1}.C + \overline{\text{in}_2}.C) \]

\[ S_1 \overset{\text{def}}{=} \text{in}_1.\overline{\text{out}_1}.S_1 \]

\[ S_2 \overset{\text{def}}{=} \text{in}_2.\overline{\text{out}_2}.S_2 \]

CCS bad computation: \( M_3 \overset{\text{job}}{\rightarrow} \quad \overline{\text{out}_2} \quad \ldots \)

Is this possible in CCSprio? No!

\[ M_3 \overset{\text{job}}{\rightarrow} (\overline{\text{in}_1}.C + \overline{\text{in}_2}.C) \mid S_1 \mid S_2 \setminus \{\text{in}_1, \text{in}_2\} \]

\[ \overset{\rightarrow}{\Rightarrow} (C \mid \overline{\text{out}_1}.S_1 \mid S_2) \setminus \{\text{in}_1, \text{in}_2\} \]

\[ \overline{\text{out}_2} \]
Part II

$\text{CCS}^{\text{prio}}$: Bisimulation semantics  [Milner, Park]
Bisimulation Equivalences for $\text{CCS}^{\text{prio}}$

The definition of $\rightarrow$ turns $\text{CCS}^{\text{prio}}$ 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.,
  
  \[
  P \sim Q \text{ implies } K[P] \sim K[Q], \text{ for all } \text{CCS}^{\text{prio}} \text{ contexts } K[\cdot].
  \]
(Strong) Bisimulation Equivalence for CCS\textsuperscript{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.
A symmetric relation $\mathcal{R} \subseteq P \times P$ is a (strong) bisimulation if,
for all $\langle P, Q \rangle \in \mathcal{R}$ and $\alpha \in A$:
$P \xrightarrow{\alpha} P'$ implies $Q \xrightarrow{\alpha} Q'$ for some $Q'$ with $\langle P', Q' \rangle \in \mathcal{R}$.

$P \xrightarrow{\alpha} P'$

$Q \xrightarrow{\alpha} Q'$

$\forall$

$P' \xrightarrow{\alpha} Q'$

$\exists$

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 $\sim$

1. $\sim$ is the largest bisimulation.

2. $\sim$ is an equivalence relation.

3. $\sim$ is a congruence for $\text{CCS}^{\text{prio}}$.

The last fact means that $\sim$ 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$^{\text{pio}}$ 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) $t + 0 = t$

©2001 G. Lütten and R. Cleaveland
Axiomatization (cont.)

Other axioms from CCS:

(Res1) \[ 0 \setminus L = 0 \]

(Res2) \[ (\alpha.t) \setminus L = 0 \quad (\alpha \in L \cup \overline{L}) \]

(Res3) \[ (\alpha.t) \setminus L = \alpha.(t \setminus L) \quad (\alpha \notin L \cup \overline{L}) \]

(Res4) \[ (t + u) \setminus L = (t \setminus L) + (u \setminus L) \]

(Rel1) \[ 0[f] = 0 \]

(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 \( t = \sum \alpha_i . t_i \) and \( u = \sum \beta_j . u_j \). Then
\[
\begin{align*}
  t | u &= \sum \alpha_i . (t_i | u) + \\
        &\quad \sum \beta_j . (t | u_j) + \\
        &\quad \sum_{\alpha_i = \beta_j} \{ \tau . (t_i | u_j) \mid \alpha_i \in A \} + \\
        &\quad \sum_{\alpha_i = \beta_j} \{ \tau . (t_i | u_j) \mid \alpha_i \in A \}
\end{align*}
\]

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

(P) \( \tau . t + a . u = \tau . t \quad (a \in A) \)
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\textsuperscript{prio}

Relation \(\sim\) makes no special accommodation for \(\tau, \tau\); they are treated just like any other action.

To abstract from these internal actions, we will adapt \(\sim\) 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\textsuperscript{prio}. 
Defining (Naive) Weak Transitions

Idea

\[ P \xrightarrow{\alpha} \times P' \] should hold whenever

\[ P \xrightarrow{\tau} \ldots \xrightarrow{\tau} P_1 \xrightarrow{\alpha} P_2 \xrightarrow{\tau} \ldots \xrightarrow{\tau} P'. \]

Definition

1. \[ \xrightarrow{\epsilon} \times = (\xrightarrow{\tau} \cup \xrightarrow{\tau})^* \]
2. \[ P \xrightarrow{\alpha} \times P' \text{ if } P \xrightarrow{\epsilon} \times \xrightarrow{\alpha} \xrightarrow{\epsilon} \times P'. \]
3. The visible content \( \hat{\alpha} \) of action \( \alpha \in \mathcal{A} \) is defined by:

\[ \hat{\alpha} = \begin{cases} 
\epsilon & \text{if } \alpha \in \{\tau, \tau\} \\
\alpha & \text{otherwise}
\end{cases} \]
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 $\alpha \in \mathcal{A}$:

\[ P \xrightarrow{\alpha} P' \implies Q \xrightarrow{\hat{\alpha}} Q' \text{ for some } Q' \text{ with } \langle P', Q' \rangle \in \mathcal{R}. \]

**Definition (naive observational equivalence)**

\[ P \approx_{\times} Q \text{ if there is a naive weak bisimulation } \mathcal{R} \text{ with } \langle P, Q \rangle \in \mathcal{R}. \]
Facts about $\approx_x$

1. $\approx_x$ is the largest naive weak bisimulation.

2. $\approx_x$ is an equivalence relation.

However, $\approx_x$ is not a congruence for $\text{CCS}^{\text{prio}}$.

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

  $$\tau.a.0 \approx_x a.0 \quad \text{but} \quad \tau.a.0 + b.0 \not\approx_x a.0 + b.0.$$ 

- There are also problems regarding parallel composition!
Problem #1 Regarding Parallel Composition

\[
\begin{align*}
P_1 & \overset{\text{def}}{=} 0 \\
Q_1 & \overset{\text{def}}{=} \tau.Q_1 \\
R_1 & \overset{\text{def}}{=} a.0
\end{align*}
\]

Here, \( P_1 \approx_x Q_1 \), but \( P_1 | R_1 \not\approx_x Q_1 | R_1 \), because

\[
P_1 | R_1 \xrightarrow{a} P_1 | 0, \text{ while } Q_1 | R_1 \not\xrightarrow{a} x.
\]

Source of problem: Pre-emption by \( \tau^\omega \).

Infinite sequences of \( \tau \)–transitions suppress all unprioritized transitions.
Problem #2 Regarding Parallel Composition

\[ P_2 \overset{\text{def}}{=} a.0 + \tau.(a.0 + \underline{b}.0) \]
\[ Q_2 \overset{\text{def}}{=} a.0 + \underline{b}.0 \]
\[ R_2 \overset{\text{def}}{=} \underline{b}.0 \]

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

\[ P_2 \mid R_2 \xrightarrow{a} 0 \mid R_2, \text{ while } Q_2 \mid R_2 \not\xrightarrow{a} \times . \]

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 \( \approx_X \)

\( \approx_X \) is not a congruence for CCS\(^{\text{prio}} \), 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 \( \approx_X \).

**Approach**

Two stages:

- Fix the problems with parallel composition.
- Fix the problem with choice.
Fixing the “Pre–emption Potentials” Problem with

Idea

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

New relation $P \xrightarrow{a \in A} P'$, for $a \in A$ and $L \subseteq A \setminus \{\tau\}$:

\[ P \xrightarrow{\tau} P_1 \xrightarrow{\tau} P_2 \xrightarrow{\tau} \ldots \xrightarrow{\tau} P_n \xrightarrow{a} \ldots \xrightarrow{\tau} P' \]

Note: If $P \xrightarrow{a \in A} P'$ and the environment does not communicate on any action in $L$, then the transition sequence from $P$ to $P'$ cannot be pre–empted!
Let $L \subseteq A$. Then

1. $P \xrightarrow{\alpha}{L} P'$ if $P \xrightarrow{\alpha} P'$ and $\text{Tr}(P) \setminus \{\tau\} \subseteq L$

2. $\xrightarrow{\epsilon}{L} = (\xrightarrow{\tau}{L} \cup \xrightarrow{\tau})^*$; $\xrightarrow{\epsilon} = (\xrightarrow{\tau})^*$

3. $P \xrightarrow{\alpha}{L} P'$ if $P \xrightarrow{\epsilon}{L} \xrightarrow{\alpha}{L} \xrightarrow{\epsilon} P'$

4. $P \xrightarrow{\alpha} P'$ if $P \xrightarrow{\epsilon} \xrightarrow{\alpha} \xrightarrow{\epsilon} P'$

5. $\hat{\alpha} = \begin{cases} 
\epsilon & \text{if } \alpha = \tau \\
\epsilon & \text{if } \alpha = \tau \\
\alpha & \text{otherwise}
\end{cases}$
A symmetric relation $\mathcal{R} \subseteq \mathcal{P} \times \mathcal{P}$ is a prioritized weak bisimulation if, for all $\langle P, Q \rangle \in \mathcal{R}$, $a \in A$, $\hat{a} \in A$, the following holds:

1. If $\tau \notin \mathcal{I}(P)$ then $Q \xrightarrow{\mathcal{I}(P)} Q'$ for some $Q'$ with $\tau \notin \mathcal{I}(Q')$, $\langle P, Q' \rangle \in \mathcal{R}$, and $\mathcal{I}(Q') \setminus \{\tau\} \subseteq \mathcal{I}(P)$.

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

3. $P \xrightarrow{a} P'$ implies $Q \xrightarrow{\mathcal{I}(P)} Q'$ for some $Q'$ with $\langle P', Q' \rangle \in \mathcal{R}$.

**Intuition:** Condition 1 fixes problem with $\tau^\omega$, while Condition 3 fixes problem related to pre-emption potentials.

**Definition (prioritized observational equivalence)**

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

1. Recall $P_1 \overset{\text{def}}{=} 0$ and $Q_1 \overset{\text{def}}{=} \tau.Q_1$.

   $P_1 \not\preceq Q_1$ since $\tau \not\in \mathcal{I}(P_1)$ and $\tau \in \mathcal{I}(Q')$ for every $Q'$ s.t. $Q_1 \overset{c}{\rightarrow} Q'$.

2. Recall $P_2 \overset{\text{def}}{=} a.0 + \tau.(a.0 + b.0)$ and $Q_2 \overset{\text{def}}{=} a.0 + b.0$.

   Then $P_2 \not\preceq Q_2$ since $P_2 \overset{a}{\rightarrow} 0$ but $Q_2 \not\overset{a}{\rightarrow} (\emptyset = \mathcal{I}(P_2))$.

3. Note that $P_2 + b.0 \simeq Q_2$. In particular, $P_2 + b.0 \overset{a}{\rightarrow} 0$ can be matched by $Q_2 \overset{a}{\rightarrow} 0$ ($\mathcal{I}(P_2 + b.0) = \{b\}$).
Facts about $\approx$

1. $\approx$ is the largest prioritized weak bisimulation.

2. $\approx$ is an equivalence relation contained in $\approx_\times$.

3. $\approx$ is preserved by all $\text{CCS}^{\text{prio}}$ operators except choice.

4. The largest congruence $\approx_{\times}^1$ of $\approx_\times$ wrt. $\text{CCS}^{\text{prio}}$ is contained within $\approx$.

The proof requires the identification of special $\text{CCS}^{\text{prio}}$ 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 $\mathcal{R}_1^l = \mathcal{R}_2^l$. 

\[ \text{\copyright 2001 G. Lüttgen and R. Cleaveland} \]
Fixing the Problem with Choice

\[ \tau.a.0 \cong a.0, \quad \text{and yet} \quad \tau.a.0 + b.0 \not\cong a.0 + b.0. \]

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** \( P \cong^l Q \) if the following, and their symmetric counterparts, hold.

1. \( P \xrightarrow{a} P' \) implies \( Q \xrightarrow{a} Q' \) for some \( Q' \) with \( P' \cong Q' \).
2. \( P \xrightarrow{a} P' \) implies \( Q \xrightarrow{a} Q' \) for some \( Q' \) with \( P' \cong Q' \).
Remarks on the Semantic Theory of $\approx^l$

**Theorem** $\approx^l$ is the largest congruence contained in $\approx\times$.

**Axiomatizing $\approx^l$** ... is technically complex, even for finite processes.

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

- **Example:** Relabelings permitted to change priority values, e.g., \( f(a) = b \).

- Implications of prioritization and de prioritization operators on semantic theory have been studied by Cleaveland and Hennessy.

Multi–level priority structure

- **Example:** \((b \cdot 0 + a \cdot 0) \mathrel{|} (\overline{a} \cdot 0 + c \cdot 0) \mathrel{|} \overline{c}\), i.e., transitions labeled by \(b\), \(a\), \(\overline{a}\), and \(\tau\) are pre–empted by \(\tau\).

- Semantic theory can be adapted; no surprises.
Part III

A different concept of pre-emption for $\text{CCS}^{\text{prio}}$
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 \(\text{dma}i\) should be pre–empted by this prioritized communication.
- Action \(\text{dma}j\), 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.

⇒ Local pre-emption
Global vs. Local Pre-emption

- 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 ≃:

\[
P \quad \sim \quad Q
\]

m, α \quad n, α

\forall \quad \exists

P' \quad \sim \quad Q'

**Problem**

≃ is not a congruence, e.g.,

\[
a.b.0 + b.a.0 \quad \sim \quad a.0 \mid b.0 \quad \text{but} \quad (a.b.0 + b.a.0)\mid \overline{b}.0 \not\sim (a.0 \mid b.0)\mid \overline{b}.0
\]

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

Take local pre-emption into account by defining a finer bisimulation $\approx^1$.

\[
\begin{array}{c c}
\text{P} & \approx^1 & \text{Q} \\
\downarrow m,\alpha & \downarrow n,\alpha & \downarrow m,\alpha & \downarrow n,\alpha \\
\text{P}' & \approx^1 & \text{Q}' & \text{P}' & \approx^1 & \text{Q}' \\
\end{array}
\]

and $\mathcal{I}_n(Q) \setminus \{\tau\} \subseteq \mathcal{I}_m(P)$

$\mathcal{I}_m(P) = \{a \mid \exists \alpha \bowtie m. \text{P} \xrightarrow{o,\alpha} \}$ is a local prioritized initial action set.

Theorem: $\approx^1$ is the largest congruence contained in $\approx$. 

©2001 G. Lüttgen and R. Cleaveland
Extension to Multi–Level Priority Structures

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

Problem

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

- Right–associative way: \(\overline{a}\) is definitely pre–empted by the communication on \(c\), 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 \(a\) 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]
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

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.
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!