Concurrency problems class

Catuscia Palamidessi, Jean-Jacques Lévy, James J. Leifer
Catuscia@lix.polytechnique.fr, Jean-Jacques.Levy@inria.fr, James.Leifer@inria.fr

11 December 2003

1 Definition: CCS processes

\[ P ::= 0 \quad \text{empty} \]
\[ \alpha.P \quad \text{prefixing} \]
\[ P|P \quad \text{parallel composition} \]
\[ (\nu L)P \quad \text{hiding} \]
\[ P + P \quad \text{summation} \]

\[ \begin{array}{ll}
K & \text{constant (for expressing recursion)} \\
!P & \pi\text{-calculus-style replication (for expressing recursion)} \\
\mu X.P & \text{fixed-point (for expressing recursion)}
\end{array} \]

2 Definition: CCS alphabetic conventions

\[ a \quad \text{name} \]
\[ \overline{a} \quad \text{co-name} \]
\[ \ell \quad \text{label (ranges over names and co-names)} \]
\[ L \quad \text{label set} \]
\[ f \quad \text{label map} \]
\[ \alpha \quad \text{action (ranges over labels and } \tau) \]

3 Definition: CCS labelled transitions rules

- input: \[ a.P \overset{a}{\rightarrow} P \]
- output: \[ \overline{a}.P \overset{\overline{a}}{\rightarrow} P \]
- synchronization: \[ P \overset{\ell}{\rightarrow} P' \quad Q \overset{\tau}{\rightarrow} Q' \quad \frac{P|Q \overset{\alpha}{\rightarrow} P'|Q'}{P|Q \overset{\alpha}{\rightarrow} P'|Q'} \]
- choice: \[ P \overset{\alpha}{\rightarrow} P' \quad Q \overset{\alpha}{\rightarrow} Q' \quad \frac{P + Q \overset{\alpha}{\rightarrow} P'|Q'}{P + Q \overset{\alpha}{\rightarrow} P'|Q'} \]
- parallel composition: \[ P \overset{\alpha}{\rightarrow} P' \quad Q \overset{\alpha}{\rightarrow} Q' \quad \frac{P|Q \overset{\alpha}{\rightarrow} P'|Q'}{P|Q \overset{\alpha}{\rightarrow} P'|Q'} \]
- hiding: \[ (\nu L)P \overset{\alpha}{\rightarrow} (\nu L)P' \quad \text{if } \alpha, \overline{\alpha} \notin L \]
- and others, for example...
- constant: \[ K \overset{\alpha}{\rightarrow} P' \quad \text{if } K = P \]
- replication (many possible): \[ P\|P \overset{\alpha}{\rightarrow} P' \]

\[ 1P \overset{\alpha}{\rightarrow} P' \]
• fixed-point (many possible): \( \frac{P\{μX.P/X\}}{μX.P} \xrightarrow{α} P' \)

4 Definition: CCS operational equivalences

• strong simulation: a relation \( R \) is a strong simulation if for all \( (P, Q) \in R \) and \( P \xrightarrow{α} P' \), there exists \( Q' \) such that \( Q \xrightarrow{α} Q' \) and \( (P', Q') \in R \).

• strong bisimulation: a relation \( R \) is a strong bisimulation if it and its inverse are strong simulations.

• strong bisimilarity: \( \sim \) is the largest strong bisimulation.

• weak simulation: a relation \( R \) is a weak simulation if for all \( (P, Q) \in R \) we have:

1. if \( P \xrightarrow{τ} P' \) then there exists \( Q' \) such that \( Q \xrightarrow{τ} Q' \) and \( (P', Q') \in R \).
2. if \( P \xrightarrow{τ} P' \) then there exists \( Q' \) such that \( Q \xrightarrow{τ} Q' \) and \( (P', Q') \in R \).

• weak bisimulation: a relation \( R \) is a weak bisimulation if it and its inverse are weak simulations.

• weak bisimilarity (also known as bisimilarity, also known as observational equivalence): \( \approx \) is the largest weak bisimulation.

• observational congruence: \( \cong \) is the largest symmetric relation satisfying the following property: if \( P \cong Q \) and \( P \xrightarrow{α} P' \) then there exists \( Q' \) such that \( Q \xrightarrow{α} Q' \) and \( P' \cong Q' \).

5 Exercise (CCS): unreliable transmission medium

A transmitter \( T \), an unreliable transmission medium \( M \), and a receiver \( R \) are modelled as follows:

\[
\begin{align*}
T & \overset{\text{def}}{=} \text{in}.i.T' \\
T' & \overset{\text{def}}{=} r.\overline{i}.T' + a.T \\
M & \overset{\text{def}}{=} i.M' \\
M' & \overset{\text{def}}{=} \overline{σ}.M + τ.\overline{σ}.M \\
R & \overset{\text{def}}{=} o.\overline{\text{out}}.\overline{a}.R
\end{align*}
\]

\( M \) is an unreliable medium: having received an input message from \( T \) (action \( i \)) it either outputs the message to \( R \) (action \( σ \)), or loses it (action \( τ \)) and then sends a request for retransmission (action \( \overline{τ} \)). If \( R \) does receive the message, it delivers it (action \( \overline{\text{out}} \)) and sends an acknowledgement directly to \( T \) (action \( \overline{a} \)).

1. Calculate the transition graph of \( \nu i, o, r, a)(T|M|R) \) and hence show that this process is observationally equivalent to a simple reliable buffer \( B \) defined by:

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

2. Are \( \nu i, o, r, a)(T|M|R) \) and \( B \) observationally congruent?

3. Do the two have the same behavior with respect to divergence, that is can either perform a series of actions ending in an infinite sequence of \( τ \) actions?
6 Exercise (CCS): semaphores

1. A semaphore is a mechanism to prevent more than a certain number \( n \) of clients from simultaneously entering their critical sections to access a precious resource. A client “brackets” its critical section by requesting entry permission (action \( \text{wait} \)) and then signaling when it is finished (action \( \text{signal} \)):

\[
\text{wait} \quad \ldots \text{critical section} \ldots \quad \text{signal}
\]

Note that a mutual exclusion lock (mutex) is a special case (when \( n = 1 \)) of a semaphore.

Define a CCS process to model a semaphore of capacity \( n \). Hint: create a constant \( \text{Sem}_k \), for \( 0 < k \leq n \), that represent a semaphore in the state when \( k \) clients are in their critical sections. You will need to treat the cases \( k = 0 \) and \( k = n \) specially.

7 Exercise (CCS): deadlock

We say that a process can deadlock if it can perform a sequence of actions to enter a state that is observationally congruent (\( \equiv \)) to 0.

Let

\[
C \overset{\text{def}}{=} g_0 \cdot g_1 \cdot p_0 \cdot p_1 \cdot C \\
D \overset{\text{def}}{=} g_1 \cdot g_0 \cdot p_1 \cdot p_0 \cdot D \\
S_0 \overset{\text{def}}{=} [g_0 \cdot p_0] \cdot S_0 \\
S_1 \overset{\text{def}}{=} [g_1 \cdot p_1] \cdot S_1
\]

1. For each of the following processes, determine whether or not it can deadlock:

\[
(\nu g_0, p_0, g_1, p_1)(C|C|S_0|S_1) \\
(\nu g_0, p_0, g_1, p_1)(C|D|S_0|S_1)
\]

2. Prove that \( P \equiv 0 \) iff \( P \) can do no action.

3. Prove that \( T \approx 0 \) where \( T \overset{\text{def}}{=} \tau. T \).

4. Hence show that it is possible for a process that can deadlock to be observationally congruent to one that cannot deadlock.

8 Exercise (\( \pi \)-calculus): arithmetic

We can define a process \( N_n \) for representing the natural number \( n \) as follows:

\[
N_0(s, z) \overset{\text{def}}{=} z \\
N_{n+1}(s, z) \overset{\text{def}}{=} s.N_n(s, z)
\]

Thus \( N_n(s, z) \) outputs \( n \) times on \( s \) and then outputs on \( z \).

Our goal is define a process \( A(s_0, z_0, s_1, z_1, s, z) \) for adding numbers which has the property that

\[
(\nu s_0, z_0, s_1, z_1)(N_{n_0}(s_0, z_0) | N_{n_1}(s_1, z_1) | A(s_0, z_0, s_1, z_1, s, z)) \approx N_{n_0+n_1}(s, z)
\]

\( (*) \)

- First define a processes \( C(s, z, s', z') \) for copying a number from \((s, z)\) to \((s', z')\) and prove that

\[
(\nu s, z)(N_n(s, z)|C(s, z, s', z')) \approx N_n(s', z')
\]

- Then define addition \( A(s_0, z_0, s_1, z_1, s, z) \) and prove \( (*) \) above.