Concurrent Processes and Reaction

Overview
- External and internal actions
- Observations
- Concurrent process expressions
- Structural congruence
- Reaction

References
- Robin Milner, "Communication and Concurrency"
- Robin Milner, "Communicating and Mobile Systems"

Black Boxes

- The black box A has two ports labeled "a" and "b".
- A complementary pair (b, ˘b) of labels of Σ represents a means of interaction between black boxes.

Flow Graphs

- This a simple example of a flow graph that depicts the structure of a system composed of two sub-systems A and B.
- A flow graph depicts the linkage among its components. System A and B are connected by the complementary pair (b, ˘b). The resulting system has two open ports "a" and "c" through which the system can interact with its environment.
- A flow graph does not cover any dynamic property of a system.
Synchronized Actions

- A complementary pair \((b, \bar{b})\) denotes a synchronized action, or handshake. That is, systems can interact if they share a complementary pair of labels.
- A complementary pair of labels does not represent a buffer or channel having some capacity.
- A system \(A\) in state \(S_A\) can interact with a system \(B\) in state \(S_B\) if there exists a complementary pair of labels that enables both systems to perform a synchronous interaction.

A Concurrent System

![Diagram of concurrent system](image)

The composite system consists of the following sequential processes:

\[
\begin{align*}
A & \rightarrow a.A' & B & \rightarrow b.B' \\
A' & \rightarrow \bar{a}.A & B' & \rightarrow \bar{b}.B
\end{align*}
\]

- Internal concurrency:
  - The sub-systems \(A\) and \(B\) are running concurrently with no inter-dependency except that any action \(a\) by \(A\) must be synchronized with an action \(b\) by \(B\) and conversely.

Representing Shared Transitions

![Diagram of shared transition](image)

The bar represents the shared transition.
System Behavior

The system starts with the two sequential processes simultaneously in their initial states A and B.

The transition "a" occurs, leading to the states A' and B simultaneously.

The shared transition occurs, leading to the states A and B'.

The transition "a" and "c" occur in either order, or simultaneously, leading to the states A' and B again.

Observable Actions

- The labels "b" and "b'" represent observable actions, or observations.
- We observe "b" by interacting with it, that is, by performing its complement "b'", and conversely.
- We equate the concepts of interaction and observation.

Unobservable Actions

- Shared transactions cannot be observed.
- We can think of shared transactions as internal actions, or reactions, which is the interaction (that is, mutual observation) between two components within the system.
- Internal actions will be denoted by τ. Since τ is not observable, it does not have a complement.
- The full class of actions, both observable and internal, consists of

\[ \Sigma = L \cup \{ \tau \} \]
Concurrent Process Expressions

- Concurrent process expressions are defined by adding two new constructs to sequential process expression:
  - Composition $P \parallel Q$ to run $P$ and $Q$ concurrently, and
  - Restriction $\text{new } a P$ to restrict the scope of a name to a process.

The set $P$ of concurrent process expressions is defined by the following syntax:

$$P ::= A(a_1, \ldots, a_n) \cup \sum_{i \in I} \alpha_i. P_i \cup P_1 \cup P_2 \cup \text{new } a P$$

where $I$ is any finite indexing set. We use $P, Q, P_i, \ldots$ to stand for concurrent process expressions.

Notations

- We use $M, N$ to stand for summations. The order of terms in a summation is insignificant.
- Prefix operations $\alpha.$ and $\text{new } a$ bind more tightly than summation and composition. For example, $\text{new } a P|Q$ means $\text{new } a (P|Q)$, not $\text{new } a P|Q$.
- If a summation with more than one term occurs inside a restriction or composition, we need to use parentheses to group term: $(a.P + b.Q)|c.R$.
- We usually omit ".$0". Instead of $a.b.0$, we write simply $a.b$. Also, we write $\text{new } a,b P$ or $\text{new } ab P$ for $\text{new } a \text{ new } b P$.

First Example

The composite system consists of the following concurrent processes:

$A = a.\varepsilon A \quad B = b.c B$

$\text{new } b (A|B)$

The name $b$ occurs restricted in $A|B$. This way, the system $\text{new } b (A|B)$ can only interact with "a". Once this interaction has been consumed the system evolves silently into a state where it can interact with "a" or "c".
Bound Names

- In a restriction \( \text{new} \ a \ P \) we say that the name \( a \) is bound.
- We denote by \( \text{fn}(P) \) the set of all names occurring free, that is not bound, in \( P \).
- Changing a bound name into a fresh name is called alpha-conversion. We treat two terms as structurally congruent if one is derived from the other by alpha-conversion. For example, \( (\text{new} \ b)ab = (\text{new} \ b')ab' \).

If \( P = (\text{new} \ b)ab \), then \( (b/a)P = (\text{new} \ b)b' \).

Illustrating Reaction

- Let \( P = A'|B \) with \( A' = \exists A \) and \( B = b.B' \). Thus \( P = \exists A)(b.B') \), that is, a reaction between "b" and "A" can occur:
  \[ P \rightarrow A | B' \]

- Let \( P = \text{new} \ a ((a.Q_1 + b.Q_2) | \exists 0) | (\exists R_1 + \exists R_2) \). Then a reaction can occur either between \( a.Q_1 \) and \( \exists 0 \), or between \( b.Q_2 \) and \( \exists R_2 \). In each case, alternative choices in a summation are discarded:
  \[ P \rightarrow \text{new} \ a (Q_1 | 0 | (\exists R_1 + \exists R_2)) \]
  and
  \[ P \rightarrow \text{new} \ a (Q_2 | \exists 0 | R_2) \]

Note that \( \exists R_1 \) is not in the scope of the restriction \( \text{new} \ a \), so the \( a \)'s in \( a.Q_1 \) and \( \exists R_1 \) are different, and these two cannot react:
  \[ P \not\rightarrow \text{new} \ a (Q_1 | 0 | R_2) \]

Process Context

Since we have extended our process language, we need to revise our definition of structural congruence.

A process context \( C \) is a process expression containing a hole, represented by \( [\ast] \). Formally, process contexts are defined as follows:

\[ C ::= [\ast] | \alpha.C + M | C | P | \text{new} a.C \]

\( C(Q) \) denotes the result (that is, a process expression) of replacing the hole in the process context \( C \) with the process \( Q \).

The elementary process context are \( \alpha.[\ast] + M, [\ast] | P, | [\ast], \) and \( \text{new} a.[\ast] \). Note, \( C[Q] = Q \), that is, \( [\ast] \) is the identity context.
Process Congruence

Let $\equiv$ be an equivalence relation over $P$, that is, $\equiv$ is reflexive ($P \equiv P$), symmetric (if $P \equiv Q$ then $Q \equiv P$), and transitive (if $P \equiv Q$ and $Q \equiv R$ then $P \equiv R$). Then $\equiv$ is said to be a process congruence if it is preserved by all elementary contexts, that is, if $P \equiv Q$ then

- $a.P + M \equiv a.Q + M$
- $\text{new } a.P \equiv \text{new } a.Q$
- $P | R \equiv Q | R$
- $R | P \equiv R | Q$

Equivalence and Congruence

An arbitrary equivalence relation $\equiv$ is a process congruence if and only if, for all contexts $C$, $P \equiv Q$, implies $C[P] \equiv C[Q]$.

Question: Is $\sim$ a process congruence?

We need to verify that $P \sim Q$ implies $C[P] \sim C[Q]$. That is, we need to show that $S = \{ (C[P], C[Q]) | P \sim Q \}$ is a bisimulation.

Structural Congruence

Structural congruence, written $\equiv$, is the process congruence over $P$ determined by the following equations:

1. Change of bound names (alpha-conversion)
2. Reordering of terms in a summation
3. $P | 0 \equiv P, P | Q \equiv Q | P, P | (Q | R) \equiv (P | Q) | R$
4. $\text{new } a (P | Q) \equiv \text{new } a P | Q$ if $a \notin \text{fn}(Q)$
5. $\text{new } a 0 \equiv 0, \text{new } ab P \equiv \text{new } ba P$
6. $A(b) = (b/a)P_{\delta}$ if $A(b) = P_{\delta}$
Example

\[ P = \text{new } a \left( (a.Q_1 + b.Q_2) | \bar{a}.0 \right) | (\bar{b}.R_1 + \bar{a}.R_2) \]
\[ = \text{new } a' \left( (a'.Q_1 + b.Q_2) | \bar{a}.0 \right) | (\bar{b}.R_1 + \bar{a}.R_2) \] (1)
\[ = \text{new } a' \left( ((a'.Q_1 + b.Q_2) | \bar{a}.0 \right) | (\bar{b}.R_1 + \bar{a}.R_2)) \] (4)
\[ = \text{new } a' \left( (a'.Q_1 + b.Q_2) \bar{a}.0 | (\bar{b}.R_1 + \bar{a}.R_2) \right) \] (3)

Involves the application of \( P[0 = P] \).

Standard Form

A process expression \( \text{new } a (M_1 | \ldots | M_n) \), where each \( M_i \) is a non-empty summation, is said to be in standard form. (If \( n = 0 \) we take \( M_1 | \ldots | M_n \) to mean \( 0 \). If \( a \) is empty then there is no restriction.)

Theorem:
Every process is structurally congruent to a standard form.

Proof:
For any restriction \( \text{new } a \) inside a summation, we can bring it to the outermost by using alpha-conversion (if necessary) followed by the rule \( \text{new } a (P|Q) = \text{new } a (P|Q) \) in conjunction with some of the laws of structural congruence (3).

Process Linking

Consider a process \( P \) with two ports labeled “a” and “b”.

We often want to create a chain of such processes, linking the right port of one with the left of the next:

\[ \text{new } m (m(a)P|m(b)Q) \]

For this purpose, we can define a binary linking operator
\[ P \bot Q \rightarrow \text{new } m (m(a)P|m(b)Q) \]
where \( m \) does not occur free in \( P \) or \( Q \).
Reaction

- Reaction rules, with the help of structural congruence, define how different concurrent components of a process expression can react one with another.
- The REACT rule allows reaction to occur between an action and its complement:
  \[(a,P + M) \uparrow (P + N) \rightarrow P|Q]\]
- The TAU rule allows the internal \(\tau\)-action to occur:
  \[\tau : P + M \rightarrow P\]

In each case, zero or more alternatives represented by M and N are discarded.

When Is Reaction Possible?

- The REACT and TAU rule cannot be applied just anywhere in a process. In fact, a reaction cannot occur underneath a prefix.
- For example, in \[\pi A[a|E|Q|b,R]\] the interaction between E and R cannot occur until the action “a” has been observed.
- However, this reaction can occur in the context \[\pi A[a|E|Q|b,R]\]
  \[\equiv \pi A[a|E|Q|b,R]\]
  \[-\pi A[a|E|Q|b,R]\]
- Finally, reaction can occur inside a composition and restriction.

Reaction Rules

The reaction relation \(\rightarrow\) over \(P\) contains exactly those transitions that can be inferred from the rules in the following table:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAU:</td>
<td>(\tau : P + M \rightarrow P)</td>
</tr>
<tr>
<td>REACT:</td>
<td>((a,P + M) \uparrow (P + N) \rightarrow P</td>
</tr>
<tr>
<td>PAR:</td>
<td>(P \rightarrow P')</td>
</tr>
<tr>
<td>RES:</td>
<td>(P \rightarrow P')</td>
</tr>
<tr>
<td>new a(P) \rightarrow new a(P)</td>
<td></td>
</tr>
<tr>
<td>STRUCT:</td>
<td>(P \equiv Q)  (P \rightarrow P')  (P \equiv Q')  (Q \rightarrow Q')</td>
</tr>
</tbody>
</table>
Reaction Example

P = new a ((a.Q₁ + b.Q₂) ∈ R) ![Struct(P) = P]

Summary

- We have introduced concurrent processes, which can be built using:
  - Summation (or choice)
  - Parallel composition (concurrency)
  - Restriction
  - Process Definitions
- Complementary pairs of names represent a means of interaction between two processes.
- Reactions P → P' represent the process interactions occurring internally within the process P. Reaction does not define how a process P can interact with the environment or other processes.