Concurrency: Languages, Programming and Theory

– Equivalences for π-Calculus –

Session 13 – January 28, 2004

Uwe Nestmann

EPFL-LAMP

Derivation of Transitions (Repetition)

What is Operational Semantics about?

It provides us with a *formal* (=mechanizable) way to find out which *computations steps* (=transitions) are possible for the current state of a system.

It provides a *compiler* with a *precise specification* of what to do!

It provides the basis for the definition of *program equivalences* (and congruences!) like bisimularities.

A tool like the ABC should (=must) be able to:
(1) derive transitions according to the operational semantics,
(2) play the bisimulation game based on this information,
(3) allow us to simulate system behaviors using this information.

Derivation of Transitions: Example

$$(\boldsymbol{\nu} z)\Big(\ (\boldsymbol{\nu} y)(\underline{\overline{x}\langle y\rangle}+z(w)) \ \Big|\ (\underline{x(u)}.\overline{u}\langle v\rangle \ \Big|\ \overline{y}\langle z\rangle)\ \Big) \xrightarrow{\ \tau\ } \ldots$$

Towards Bisimulation in π-Calculus

- “standard” definition is based on *labeled transitions*

- PROBLEM: *infinite branching*
  due to infinitely many input transitions
  $\Rightarrow$ *late input transitions*

- PROBLEM: lack of congruence properties!
  - when should substitution take place?
  - how to keep track of freshness of names?

- PROBLEM: four (!) different (!!) styles of bisimulation
  ground — early — late — open
  $\Rightarrow$ which bisimulation is the “best”?

Input Transitions

$$y(\vec{x}).P \xrightarrow{y\vec{z}} [\vec{z}/\vec{x}]P \qquad \text{for all } \vec{z} \subseteq \mathcal{N}$$

generates ***infinitely many*** transitions
for each enabled input prefix.

$$y(\vec{x}).P \xrightarrow{y(\vec{x})} P$$

collapses all of them in one
by ***not yet instantiating*** the received variable.
The input is called ***late*** (or *symbolic*).
(The ... -rule should then take care of substitutions.)

$$(\text{PRE})\ \mu.P \xrightarrow{\mu} P$$

replaces the former (TAU), (OUT), and (INP).

Other Transitions ?

Now, we have transition labels

$$\mu \quad ::= \quad \tau \quad \Big| \quad y(\vec{x}) \quad \Big| \quad (\boldsymbol{\nu}\vec{w})\,\overline{y}\langle\vec{z}\rangle$$

where $\vec{w} \subseteq \vec{z}$ and $y \notin \vec{w}$. (Note that there are no more labels of the form $y\langle\vec{x}\rangle$ as we had in Session 6.)

If we change the rule for input transitions,
then what is the precise effect on the other transitions?

Note that the names $\vec{x}$ in an input label $y(\vec{x})$ arose from an input binding, and that we still need to substitute them . . .

Let us defined the bound names of a label by:

$$\mathrm{bn}(y(\vec{x})) \stackrel{\mathrm{def}}{=} \{\vec{x}\} \qquad\qquad \mathrm{bn}((\boldsymbol{\nu}\vec{w})\,\overline{y}\langle\vec{z}\rangle) \stackrel{\mathrm{def}}{=} \{\vec{w}\}$$

and, of course $\mathrm{bn}(\tau) \stackrel{\mathrm{def}}{=} \emptyset$.

Output Transitions

No input transitions involved.
No change needed, here.

$$(\text{RES})\ \frac{P \xrightarrow{\mu} P'}{(\boldsymbol{\nu} c)\, P \xrightarrow{\mu} (\boldsymbol{\nu} c)\, P'}\ \text{if } c \notin \mathrm{n}(\mu)$$

$$(\text{OPEN})\ \frac{P \xrightarrow{(\boldsymbol{\nu}\vec{b})\, \overline{a}\langle\vec{z}\rangle} P'}{(\boldsymbol{\nu} c)\, P \xrightarrow{(\boldsymbol{\nu} c\vec{b})\, \overline{a}\langle\vec{z}\rangle} P'}\ \text{if } \vec{z} \ni c \notin \{a, \vec{b}\}$$

“Uniform” Transitions

No change required.
Only non-critical access to bound names of transitions . . .

$$(\textsf{SUM})\ \frac{P \xrightarrow{\mu} P'}{P + Q \xrightarrow{\mu} P'}$$

$$(\textsf{REP})\ \frac{P \,|\, !\,P \xrightarrow{\mu} P'}{!\,P \xrightarrow{\mu} P'}$$

$$(\textsf{ALP})\ \frac{Q \xrightarrow{\mu} Q'}{P \xrightarrow{\mu} Q'} \text{ if } P =_\alpha Q$$

Transitions of Parallel Compositions

Some change & care required.
(PAR) must respect the bound input names.

$$\text{(PAR)}\ \frac{P \xrightarrow{\mu} P'}{P \,|\, Q \xrightarrow{\mu} P' \,|\, Q}\ \text{if}\ \operatorname{bn}(\mu) \cap \operatorname{fn}(Q) = \emptyset$$

$$\text{(CLOSE)}\ \frac{P \xrightarrow{a(\vec{x})} P' \qquad Q \xrightarrow{(\boldsymbol{\nu}\vec{b})\,\overline{a}\langle\vec{z}\rangle} Q'}{P \,|\, Q \xrightarrow{\tau} (\boldsymbol{\nu}\vec{b})\,([{}^{\vec{z}}/_{\vec{x}}]P' \,|\, Q')}\ \text{if}\ \{\vec{b}\} \cap \operatorname{fn}(P) = \emptyset$$

(CLOSE) must deal with the proper label and perform the substitution . . . quite at a quite late stage.

Simulating Input Transitions (I)

<u>**Definition:**</u> ("standard")

... whenever $P \mathrel{\mathcal{S}} Q$, if $P \xrightarrow{y(\vec{x})} P'$ then

there is Q' such that $Q \xrightarrow{y(\vec{x})} Q'$ with $P' \mathrel{\mathcal{S}} Q'$

Compare the following terms:

$$\begin{array}{rcl} \bar{x} \mid y & \sim & \bar{x}.y \; + \; y.\bar{x} \\ a(x).(\;\bar{x} \mid y\;) & \sim & a(x).(\;\bar{x}.y \; + \; y.\bar{x}\;) \\ a(x).(\boldsymbol{\nu} y)\,(\;\bar{x} \mid y\;) & \sim & a(x).(\boldsymbol{\nu} y)\,(\;\bar{x}.y \; + \; y.\bar{x}\;) \end{array}$$

<u>So, this kind of input simulation does not yield a congruence !</u>

Closure under input prefix means *closure under substitutions* !

Simulating Input Transitions (II)

$\ldots$ whenever $P \, \mathcal{S} \, Q$, if $P \xrightarrow{y(\vec{x})} P'$ then

ground

$\boxed{\text{there is } Q'}$

such that $Q \xrightarrow{y(\vec{x})} Q'$ with $P' \, \mathcal{S} \, Q'$

early

$\boxed{\boxed{\text{for all } \vec{z}}}$ $\boxed{\text{there is } Q'}$

such that $Q \xrightarrow{y(\vec{x})} Q'$ with $[{}^{\vec{z}}/_{\vec{x}}]P' \, \mathcal{S} \, [{}^{\vec{z}}/_{\vec{x}}]Q'$

late

$\boxed{\text{there is } Q'}$

such that $\boxed{\boxed{\text{for all } \vec{z}}}$ $Q \xrightarrow{y(\vec{x})} Q'$ with $[{}^{\vec{z}}/_{\vec{x}}]P' \, \mathcal{S} \, [{}^{\vec{z}}/_{\vec{x}}]Q'$

Simulating Input Transitions

Compare again the following terms:

$$\begin{array}{rcl} \bar{x} \mid y & \sim & \bar{x}.y \;+\; y.\bar{x} \\ a(x).(\; \bar{x} \mid y \;) & \sim & a(x).(\; \bar{x}.y \;+\; y.\bar{x} \;) \end{array}$$

So, neither early nor late input simulation yield congruences !

Open Input Simulation

$\ldots$ whenever $P\ \mathcal{S}\ Q$,

$\boxed{\boxed{\text{for all } \sigma}}$, if $\boxed{\sigma P} \xrightarrow{\mu} P'$ then

there is Q' such that $\boxed{\sigma Q} \xrightarrow{\mu} Q'$ with $P'\ \mathcal{S}\ Q'$.

Note:

- Substitution-closure is required ***before*** *each step*.
- *Open simulation* provides substitution-closure "by definition".

However, it is going a bit too far $\ldots$

Example

Compare the following terms:

$$
\begin{array}{rcl}
\bar{x} \mid y & \sim & \bar{x}.y \;+\; y.\bar{x} \\
a(x).(\boldsymbol{\nu} y)\,(\,\bar{x} \mid y\,) & \sim & a(x).(\boldsymbol{\nu} y)\,(\,\bar{x}.y \;+\; y.\bar{x}\,) \\
(\boldsymbol{\nu} x)\,\overline{a}\langle x\rangle.(\,\bar{x} \mid y\,) & \sim & (\boldsymbol{\nu} x)\,\overline{a}\langle x\rangle.(\,\bar{x}.y \;+\; y.\bar{x}\,)
\end{array}
$$

What happens after the output transition $\xrightarrow{(\boldsymbol{\nu} x)\,\overline{a}\langle x\rangle}$?

If we forget that x was freshly generated,
then it might accidentally be confused with y
when open-simulating the next (τ) transition.

Simulating Output Transitions

Under *open simulation* the approach:

> $\ldots$ whenever $P \; \mathcal{S} \; Q$,
>
> if $P \xrightarrow{(\boldsymbol{\nu} \vec{w})\, \overline{y}\langle \vec{z} \rangle} P'$ then
>
> there is Q' such that $Q \xrightarrow{(\boldsymbol{\nu} \vec{w})\, \overline{y}\langle \vec{z} \rangle} Q'$ with $P' \; \mathcal{S} \; Q'$.

is too naïve !!

Distinction

Definition:
A *distinction* D is a
finite symmetric *ir*reflexive relation on names.

A substitution σ *respects* a distinction D
if $(x, y) \in D$ implies $\sigma x \neq \sigma y$.

A *D-congruence* is . . . w.r.t. only those contexts
that do not use the names in D as “hole-binding” names.

Open Bisimilarity

Definition:
$\{\sim^D \mid D$ is a distinction$\}$ is the largest family of symmetric relations such that if $P \sim^D Q$ and σ respects D, then

- if $\sigma P \xrightarrow{\mu} P'$ and μ is not a bound output, then there is Q' such that $\sigma Q \xrightarrow{\mu} Q'$ with $P' \sim^D Q'$.
- if $\sigma P \xrightarrow{(\boldsymbol{\nu}\vec{w})\,\overline{y}\langle\vec{z}\rangle} P'$, then there is Q' such that $\sigma Q \xrightarrow{(\boldsymbol{\nu}\vec{w})\,\overline{y}\langle\vec{z}\rangle} Q'$ with $P' \sim^{D'} Q'$ where $D' := \sigma D \cup (\{\vec{w}\} \times \mathrm{fn}(\sigma P, \sigma Q))^{=}$.

The weak version is defined as usual.
Both the strong and weak bisimilarities are D-*congruences*.

Relation to the ABC

The bisimulation relation generated by the ABC are open bisimulations.

Each element of such a relation is a triple, consisting of two terms and a *distinction* ...

Some more interesting examples next week ...