Concurrency Semantics
Week 4

Course Notes 2005
EPFL – I&C

Uwe Nestmann
Johannes Borgström

April 27, 2005

$Id: notes-4.tex,v 1.3 2005/04/25 16:16:12 uwe Exp $

3 Concurrent Processes

3.1 Definition (Concurrent Process Expressions)
The sets $\mathcal{P}$ and $\mathcal{M}$ of concurrent process expressions is defined by the following BNF-syntax:

$$\begin{array}{rcl} P & ::= & \mathsf{A}\langle\,\vec{a}\,\rangle \;\big|\; M \;\big|\; P|P \;\big|\; (\boldsymbol{\nu}a)\,P \\ M & ::= & \mathbf{0} \;\big|\; \mu.P \;\big|\; M+M \end{array}$$

under the same assumptions on process identifiers as in the sequential case. The expression $P|P$ denotes *the parallel composition*, while $(\boldsymbol{\nu}a)\,P$ denotes *name generation*, which is also called *restriction*.

If necessary, we use parentheses to clarify the scope of the various process operators in expressions. Moreover, we impose that unary operators have precedence over (i.e., bind tighter) than binary operators.

$$\begin{array}{rcl} (\boldsymbol{\nu}a)\,P \mid Q & = & ((\boldsymbol{\nu}a)\,P) \mid Q \\ a.P+M & = & (a.P)+M \\ \sigma M_1 + M_2 & = & \sigma(M_1)+M_2 \end{array}$$

3.2 Notation We also use the abbreviation

$$\prod_{i\in I} P_i \stackrel{\text{def}}{=} P_1 \mid \ldots \mid P_n$$

where I is the finite indexing set $\{1\ldots,n\}$.
Note that then the order of components is not fixed.

3.3 Definition (Free Names)
The set $\mathrm{fn}(P)$ is defined inductively by:

$$\begin{array}{lcl} \mathrm{fn}(\mu) & \stackrel{\text{def}}{=} & \begin{cases} \{b\} & \text{if } \mu = b \\ \{b\} & \text{if } \mu = \bar{b} \\ \emptyset & \text{if } \mu = \tau \end{cases} \\ \hline \mathrm{fn}(\mathbf{0}) & \stackrel{\text{def}}{=} & \emptyset \\ \mathrm{fn}(\mu.P) & \stackrel{\text{def}}{=} & \mathrm{fn}(\mu) \cup \mathrm{fn}(P) \\ \mathrm{fn}(M_1+M_2) & \stackrel{\text{def}}{=} & \mathrm{fn}(M_1) \cup \mathrm{fn}(M_2) \\ \hline \mathrm{fn}(\mathsf{A}\langle\,\vec{a}\,\rangle) & \stackrel{\text{def}}{=} & \{\vec{a}\} \\ \mathrm{fn}(P_1|P_2) & \stackrel{\text{def}}{=} & \mathrm{fn}(P_1) \cup \mathrm{fn}(P_2) \\ \mathrm{fn}((\boldsymbol{\nu}a)\,P) & \stackrel{\text{def}}{=} & \mathrm{fn}(P) \setminus \{\,a\,\} \end{array}$$

We say that a occurs **free** in P,
if it occurs without enclosing $(\boldsymbol{\nu}a)\,[\cdot]$ in P.

3.4 Definition (Bound Names)
The set $\mathrm{bn}(P)$ is defined inductively by:

$$
\begin{array}{lcl}
\mathrm{bn}(\mu) & \stackrel{\text{def}}{=} & \begin{cases} \emptyset & \text{if } \mu = b \\ \emptyset & \text{if } \mu = \bar{b} \\ \emptyset & \text{if } \mu = \tau \end{cases} \\
\hline
\mathrm{bn}(\mathbf{0}) & \stackrel{\text{def}}{=} & \emptyset \\
\mathrm{bn}(\mu.P) & \stackrel{\text{def}}{=} & \mathrm{bn}(\mu) \cup \mathrm{bn}(P) \\
\mathrm{bn}(M_1 + M_2) & \stackrel{\text{def}}{=} & \mathrm{bn}(M_1) \cup \mathrm{bn}(M_2) \\
\hline
\mathrm{bn}(\mathsf{A}\langle\, \vec{a}\,\rangle) & \stackrel{\text{def}}{=} & \emptyset \\
\mathrm{bn}(P_1|P_2) & \stackrel{\text{def}}{=} & \mathrm{bn}(P_1) \cup \mathrm{bn}(P_2) \\
\mathrm{bn}((\boldsymbol{\nu} a)\, P) & \stackrel{\text{def}}{=} & \mathrm{bn}(P) \cup \{\, a \,\}
\end{array}
$$

We say that a occurs **bound** in P,
if P has a subterm $(\boldsymbol{\nu} a)\, Q$ where a occurs free in Q.
We say that $(\boldsymbol{\nu} a)\, P$ binds (any occurrence of) a in P.

3.5 Definition A name a is called *fresh* with respect to an expression P if it does not occur in it,
i.e., if $a \notin \mathrm{fn}(P) \cup \mathrm{bn}(P)$.

3.6 Definition The process P' is a *simple α-conversion* of P if it can be obtained by replacing an instance of a subterm $(\boldsymbol{\nu} a)\, Q$ of P with $(\boldsymbol{\nu} b)\, Q'$, where Q' is obtained by replacing all occurences of a with b in Q, for some b that is fresh with respect to Q.
The relation $=_\alpha$ (of type $\mathcal{P} \times \mathcal{P}$), called *$\alpha$-congruence*, is the smallest equivalence relation containing simple α-renaming.

3.7 Definition Let $P \in \mathcal{P}$. We call P *clash-free* (or *α-free*) if $\mathrm{fn}(P) \cap \mathrm{bn}(P) = \emptyset$ and all set unions in the definition of $\mathrm{bn}(P)$ are disjoint unions (i.e., the same name is not bound twice in P).

3.8 Lemma For every expression $P \in \mathcal{P}$, there exists a clash-free expression $\widehat{P} \in \mathcal{P}$ such that $P =_\alpha \widehat{P}$.
In this case, we call $\widehat{P}$ a *clash-free version* of P.

3.9 Definition (Simultaneous Substitution)
Any substitution $\sigma : \mathcal{N} \to \mathcal{N}$ is lifted to concurrent process expressions $\mathcal{P} \to \mathcal{P}$, inductively defined by:

$$\begin{array}{rcl}
\sigma(\mathbf{0}) & \stackrel{\text{def}}{=} & \mathbf{0} \\
\sigma(\mu.P) & \stackrel{\text{def}}{=} & \sigma(\mu).\sigma(P) \\
\sigma(M_1 + M_2) & \stackrel{\text{def}}{=} & \sigma(M_1) + \sigma(M_2) \\
\sigma(\mathsf{A}\langle\, \vec{a}\,\rangle) & \stackrel{\text{def}}{=} & \mathsf{A}\langle\, \sigma(\vec{a})\,\rangle \\
\sigma(P_1|P_2) & \stackrel{\text{def}}{=} & \sigma(P_1) \mid \sigma(P_2) \\
\sigma((\boldsymbol{\nu} a)\, P) & \stackrel{\text{def}}{=} & (\boldsymbol{\nu} a)\, \sigma(P)
\end{array}$$

We say that σ avoids name-clashes on P, if $\sup(\sigma) \cap \mathrm{bn}(P) = \emptyset = \sigma(\sup(\sigma)) \cap \mathrm{bn}(P)$.

To ensure that we always avoid name-clashes when applying substitutions, we silently assume that an appropriate α-conversion is implicitly applied whenever necessary.

3.10 Definition (Operational Semantics)
The LTS $(\mathcal{P}, \mathcal{T})$ of sequential process expressions over $\mathcal{A}$ has $\mathcal{P}$ as states, and its transitions $\mathcal{T}$ are precisely generated by the following rules:

$$\text{PRE: } \mu.P \xrightarrow{\mu} P$$

$$\text{SUM}_1\text{: } \frac{M_1 \xrightarrow{\mu} M_1'}{M_1{+}M_2 \xrightarrow{\mu} M_1'} \qquad \text{SUM}_2\text{: } \frac{M_2 \xrightarrow{\mu} M_2'}{M_1{+}M_2 \xrightarrow{\mu} M_2'}$$

$$\text{DEF: } \frac{\{\vec{c}/\vec{a}\}M \xrightarrow{\mu} P'}{\mathsf{A}\langle\, \vec{c}\,\rangle \xrightarrow{\mu} P'} \quad \text{IF } \mathsf{A}(\,\vec{a}\,) \stackrel{\text{def}}{=} M$$

$$\text{PAR}_1\text{: } \frac{P_1 \xrightarrow{\mu} P_1'}{P_1|P_2 \xrightarrow{\mu} P_1'|P_2} \qquad \text{PAR}_2\text{: } \frac{P_2 \xrightarrow{\mu} P_2'}{P_1|P_2 \xrightarrow{\mu} P_1|P_2'}$$

$$\text{COM: } \frac{P \xrightarrow{\lambda} P' \qquad Q \xrightarrow{\overline{\lambda}} Q'}{P|Q \xrightarrow{\tau} P'|Q'}$$

$$\text{RES: } \frac{P \xrightarrow{\mu} P'}{(\boldsymbol{\nu} a)\, P \xrightarrow{\mu} (\boldsymbol{\nu} a)\, P'} \quad \text{IF } \mu {\notin} \{a, \overline{a}\}$$

$$\text{ALPHA: } \frac{Q \xrightarrow{\mu} Q'}{P \xrightarrow{\mu} P'} \quad \text{IF } P{=_\alpha}Q \text{ AND } P'{=_\alpha}Q'$$

where $\overline{\overline{\lambda}} \stackrel{\text{def}}{=} \lambda$.

3.11 Proposition For each $P \in \mathcal{P}$, there is a finite index set I, and for all $i \in I$ there are actions β_i and processes Q_i such that

$$P \sim \sum_{i \in I} \{\ \beta_i.Q_i \mid P \xrightarrow{\beta_i} Q_i\ \}.$$

3.12 Proposition For all $n \geq 0$ and $P_1, \ldots, P_n \in \mathcal{P}$:

$$P_1|\cdots|P_n \sim \begin{cases} \sum \{\ \beta.(\ P_1|\cdots|P_i'|\cdots|P_n\ ) \\ \quad \mid\ \exists 1{\leq}i{\leq}n : P_i \xrightarrow{\beta} P_i'\ \} \\ + \\ \sum \{\ \tau.(\ P_1|\cdots|P_i'|\cdots|P_j'|\cdots|P_n\ ) \\ \quad \mid\ \exists 1{\leq}i{<}j{\leq}n : P_i \xrightarrow{\lambda} P_i' \wedge P_j \xrightarrow{\overline{\lambda}} P_j'\ \} \end{cases}$$

3.13 Proposition For all $n \geq 0$, $P_1, \ldots, P_n \in \mathcal{P}$, and $\vec{a}$:

$$(\boldsymbol{\nu}\vec{a})\,(\ P_1|\cdots|P_n\ ) \sim \begin{cases} \sum \{\ \beta.(\boldsymbol{\nu}\vec{a})\,(\ P_1|\cdots|P_i'|\cdots|P_n\ ) \\ \quad \mid\ \exists 1{\leq}i{\leq}n : P_i \xrightarrow{\beta} P_i' \wedge \beta, \overline{\beta} \notin \vec{a}\ \} \\ + \\ \sum \{\ \tau.(\boldsymbol{\nu}\vec{a})\,(\ P_1|\cdots|P_i'|\cdots|P_j'|\cdots|P_n\ ) \\ \quad \mid\ \exists 1{\leq}i{<}j{\leq}n : P_i \xrightarrow{\lambda} P_i' \wedge P_j \xrightarrow{\overline{\lambda}} P_j'\ \} \end{cases}$$