Concurrent Process Expressions

January 7, 2002, 17:43

Uwe Nestmann

EPFL-LAMP

Composition/Product of LTS

structural: *flow graphs*
behavioral: graphs (model) vs algebra (syntax)

design decisions:

- what is interaction / synchronization ?
  - *two-way* vs *multi-way*
  - *must* vs *may*
- what is external / internal ?
  - *implicitly hidden* vs *explicit hiding*

“Structural” Product

- **flow graphs** depict the structure of a system: the **linkage** among components i.e., their externally visible buttons
- only informal use, here ...

$$
\begin{aligned}
A &\stackrel{\text{def}}{=} a.A' \\
A' &\stackrel{\text{def}}{=} \overline{b}.A \\
B &\stackrel{\text{def}}{=} b.B' \\
B' &\stackrel{\text{def}}{=} \overline{c}.B
\end{aligned}
$$

“Behavioral” Product

Let $A_i = (Q_i, T_i)$ be two LTS over $\mathcal{A} = \mathcal{N} \cup \overline{\mathcal{N}}$.

$$A_1 \times A_2 \stackrel{\text{def}}{=} (\ \mathrm{st}(Q_1, Q_2), \mathrm{tr}(T_1, T_2)\ )$$

$$\mathrm{st}(Q_1, Q_2) \stackrel{\text{def}}{=} Q_1 \times Q_2 = \{\ (q_1, q_2) \mid q_1 \in Q_1 \wedge q_2 \in Q_2\ \}$$

$$\mathrm{tr}(T_1, T_2) \stackrel{\text{def}}{=} \bigcup \begin{array}{l} \{\ ((q_1, q_2), a, (q_1', q_2)) \mid (q_1, a, q_1') \in T_1\ \} \\ \{\ ((q_1, q_2), a, (q_1, q_2')) \mid (q_2, a, q_2') \in T_2\ \} \\ \{\ ((q_1, q_2), a, (q_1', q_2')) \mid \qquad \ldots \qquad \} \end{array}$$

Compute the product $A \times B$ of the example.

Concurrent Process Expressions

$\mathcal{I}$	process identifiers	$A, B \ldots$
$\mathcal{N}$	names	$a, b, c \ldots$
$\overline{\mathcal{N}}$	co-names	$\overline{a}, \overline{b}, \overline{c} \ldots$
$\mathcal{L}$	labels (buttons)	$:= \mathcal{N} \cup \overline{\mathcal{N}}$
$\mathcal{A}$	actions	metavariables $\alpha, \beta \ldots \in \mathcal{L} \underline{\cup \{\tau\}}$

- **finite sequences** $\vec{a} \ldots$
- **parametric processes** $A\langle\, a, c \,\rangle \ldots$
- **defining equations** $A(\,\vec{a}\,) \stackrel{\text{def}}{=} \ldots$

Concurrent Process Expressions (II)

<u>Definition:</u> The set $\mathcal{P}$ of conc. proc. exp. is defined (precisely) by the following BNF-syntax:

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

We use P, Q, P_i to stand for process expressions.

- $(\boldsymbol{\nu} a)\, P$ restricts the scope of a to P
- $(\boldsymbol{\nu} ab)\, P$ abbreviates $(\boldsymbol{\nu} a)\, (\boldsymbol{\nu} b)\, P$
- $\sum_{i \in \{1..n\}} \alpha_i.P_i$ abbreviates $\alpha_1.P_1 + \ldots + \alpha_n.P_n$

Concurrent Process Expressions (III)

- precedence: unary binds tighter than binary

$$\begin{aligned}(\boldsymbol{\nu} a)\, P \mid Q &= ((\boldsymbol{\nu} a)\, P) \mid Q \\ a.P + M &= (a.P) + M\end{aligned}$$

$$\begin{aligned}P|Q + R &= (P|Q) + R \\ P + Q|R &= P + (Q|R)\end{aligned}$$

$$\{^a/_b\}M_1 + M_2 = (\{^a/_b\}M_1) + M_2$$

Bound and Free Names

- $(\boldsymbol{\nu} a)\, P$ **binds** a in P
- a occurs **bound** in P, if it occurs in a subterm $(\boldsymbol{\nu} a)\, Q$ of P
- a occurs **free** in P, if it occurs without enclosing $(\boldsymbol{\nu} a)\, Q$ in P
- Define $\mathrm{fn}(P)$ and $\mathrm{bn}(P)$ inductively on $\mathcal{P}$ (sets of free/bound names of P):

$$\begin{aligned}\mathrm{fn}(P_1|P_2) &\stackrel{\text{def}}{=} \ldots \\ \mathrm{fn}((\boldsymbol{\nu} a)\, P) &\stackrel{\text{def}}{=} \ldots\end{aligned}$$

α-Conversion & Substitution

- **substitution** $\{\vec{b}/\vec{a}\}P$ (for matching $\vec{b}$ and $\vec{a}$) replaces *all* **free** occurrences of a_i in P by b_i.

  $\{^b/_a\}(\boldsymbol{\nu} b)\, b.a = ?$

- α**-conversion**, written $=_\alpha$:
  conflict-free **renaming of bound names**
  (no new name-bindings shall be generated)

- **substitution** $\{\vec{b}/\vec{a}\}P$ (for matching $\vec{b}$ and $\vec{a}$) replaces *all* **free** occurrences of a_i in P by b_i, possibly enforcing α-conversion.

Examples

$$\begin{aligned}
(\boldsymbol{\nu} a)\,(\overline{a}.\mathbf{0}|b.\mathbf{0}) &=_\alpha (\boldsymbol{\nu} c)\,(\overline{c}.\mathbf{0}|b.\mathbf{0}) \\
&=_\alpha (\boldsymbol{\nu} b)\,(\overline{b}.\mathbf{0}|b.\mathbf{0})
\end{aligned}$$

$$\begin{aligned}
\{^a/_b\}(\ (\boldsymbol{\nu} b)\,\overline{b}.\mathbf{0} \mid b.\mathbf{0}\ ) &=_\alpha (\ (\boldsymbol{\nu} b)\,\overline{a}.\mathbf{0} \mid a.\mathbf{0}\ ) \\
&=_\alpha (\ (\boldsymbol{\nu} b)\,\overline{b}.\mathbf{0} \mid a.\mathbf{0}\ )
\end{aligned}$$

$$\begin{aligned}
\{^a/_b\}(\ (\boldsymbol{\nu} a)\,\overline{b}.a.\mathbf{0} \mid b.\mathbf{0}\ ) &=_\alpha (\ (\boldsymbol{\nu} a)\,\overline{a}.a.\mathbf{0} \mid a.\mathbf{0}\ ) \\
&=_\alpha (\ (\boldsymbol{\nu} c)\,\overline{a}.c.\mathbf{0} \mid a.\mathbf{0}\ )
\end{aligned}$$

Reaction, Informally . . .

- concurrent execution of two complementary actions within a process term, i.e., within the LTS associated with it.
- reaction is non-deterministic:

$$
\begin{array}{lcl}
P & = & (\boldsymbol{\nu} a)\,(\; a.Q_1 + b.Q_2 \mid \overline{a}.\mathbf{0}\;) \mid \overline{b}.R_1 + \overline{a}.R_2 \\
P & \rightarrow & \\
P & \rightarrow & \\
P & \not\rightarrow &
\end{array}
$$

- find a way to **compute** the reactions . . .