Concurrency: Languages, Programming and Theory

– Proofs in CCS –

Session 12 – January 21, 2004

Uwe Nestmann

EPFL-LAMP

The Scheduler Problem

- informal specification
- ***specification*** as *sequential* process expression
- ***implementation*** as *concurrent* process expression
- comparison between specification and implementaton
  - proofs using ABC
  - proofs “by hand” (very close to [§ 7.3])

Scheduler, Informally [Mil99, § 3.6]

- a set of n processes $P_i, 0 \le i \le n{-}1$ is to be scheduled
- P_i starts by sync'ing on a_i with the scheduler
- P_i completes by sync'ing on b_i with the scheduler

- **(1)** each P_i *must not* run two tasks at a time
- **(2)** tasks of different P_i *may* run at the same time

- a_i are required to occur cyclically (initially, 0 starts)
- for each i, a_i and b_i must occur cyclically
- **(3)** maximal "progress":
  the scheduling must permit
  any of the "buttons" to be pressed
  at any time provided (1) and (2) are not violated.

Formal Specification [Mil99, § 3.6]

$i \in \{0 \ldots, n-1\} \quad X \subseteq \{0 \ldots, n-1\}$

$\mathsf{S}_{i,X}(\vec{a}, \vec{b}) \stackrel{\text{def}}{=}$ scheduler, where i is next and every $j \in X$ is running

(* we omit the parameters in the following *)

$$\mathsf{S}_{i,X} \stackrel{\text{def}}{=} \begin{cases} \sum_{j \in X} b_j.\mathsf{S}_{i,X-j} & (i \in X) \\ \sum_{j \in X} b_j.\mathsf{S}_{i,X-j} + a_i.\mathsf{S}_{(i+1) \bmod n, X \cup i} & (i \notin X) \end{cases}$$

$$\mathsf{Scheduler}_n \stackrel{\text{def}}{=} \mathsf{S}_{0,\emptyset}$$

- draw the transition graph for $n = 2$
- show that the scheduler is never deadlocked
- what is the difference when dropping the case for $i \in X$?

Formal "Implementation" [§ 7.3]

$$A(\,a, b, c, d\,) \stackrel{\text{def}}{=} a.c.b.\overline{d}.A$$

$$A(\,a, b, c, d\,) \stackrel{\text{def}}{=} a.C\langle\,a, b, c, d\,\rangle$$

$$C(\,a, b, c, d\,) \stackrel{\text{def}}{=} c.B\langle\,a, b, c, d\,\rangle$$

$$B(\,a, b, c, d\,) \stackrel{\text{def}}{=} b.D\langle\,a, b, c, d\,\rangle$$

$$D(\,a, b, c, d\,) \stackrel{\text{def}}{=} \overline{d}.A\langle\,a, b, c, d\,\rangle$$

$$\vec{a} := a_1 \ldots, a_n, \quad \vec{b} := b_1 \ldots, b_n \quad \vec{c} := c_1 \ldots, c_n$$

$$A_i(\,\vec{a}, \vec{b}, \vec{c}\,) \stackrel{\text{def}}{=} A\langle\,a_i, b_i, c_i, c_{i \ominus_n 1}\,\rangle$$

$$B_i(\,\vec{a}, \vec{b}, \vec{c}\,) \stackrel{\text{def}}{=} B\langle\,a_i, b_i, c_i, c_{i \ominus_n 1}\,\rangle$$

$$C_i(\,\vec{a}, \vec{b}, \vec{c}\,) \stackrel{\text{def}}{=} C\langle\,a_i, b_i, c_i, c_{i \ominus_n 1}\,\rangle$$

$$D_i(\,\vec{a}, \vec{b}, \vec{c}\,) \stackrel{\text{def}}{=} D\langle\,a_i, b_i, c_i, c_{i \ominus_n 1}\,\rangle$$

$$S(\,\vec{a}, \vec{b}\,) \stackrel{\text{def}}{=} (\boldsymbol{\nu}\vec{c})\Big(\,A_1\langle\,\vec{a}, \vec{b}, \vec{c}\rangle | D_2\langle\,\vec{a}, \vec{b}, \vec{c}\rangle | \cdots | D_n\langle\,\vec{a}, \vec{b}, \vec{c}\rangle\,\Big)$$

Formal "Implementation" (II) [§ 7.3]

$$A(\,a,b,c,d\,) \overset{\text{def}}{=} a.c.(b.\overline{d}.A + \overline{d}.b.A)$$

$$\begin{aligned}
A(\,a,b,c,d\,) &\overset{\text{def}}{=} a.C\langle\,a,b,c,d\,\rangle \\
C(\,a,b,c,d\,) &\overset{\text{def}}{=} c.E\langle\,a,b,c,d\,\rangle \\
E(\,a,b,c,d\,) &\overset{\text{def}}{=} b.D\langle\,a,b,c,d\,\rangle + \overline{d}.B\langle\,a,b,c,d\,\rangle \\
B(\,a,b,c,d\,) &\overset{\text{def}}{=} b.A\langle\,a,b,c,d\,\rangle \\
D(\,a,b,c,d\,) &\overset{\text{def}}{=} \overline{d}.A\langle\,a,b,c,d\,\rangle
\end{aligned}$$

$$\begin{aligned}
A_i &\overset{\text{def}}{=} A\langle\,a,b,c_i,c_{i-1}\,\rangle \\
\ldots & \\
S_n &\overset{\text{def}}{=} (\boldsymbol{\nu}\vec{c})\left(\,A_1|D_2|\cdots|D_n\,\right)
\end{aligned}$$

$$\boxed{\text{Scheduler}_n \overset{?}{\approx} S_n}$$

Proofs Using ABC

- model the specification for $n = 2$
- model the wrong (!) implementation for $n = 2$
- run the ABC
- analyze the transitions systems (using `step`)
- understand the problem
  w.r.t. the formal & informal specification
- model now the correct implementation for $n = 2$
- run the ABC
- understand the bisimulation relation that ABC has generated
- if time left, try out for $n = 3 \ldots$

Proofs “by Hand” (I)

$\boxed{\text{means: “guessing” a bisimulation relation !}}$

- draw the transition graph of S for $n = 2$
- generalize for greater n . . .
- Observe: every reachable state is of the form

$$(\boldsymbol{\nu}\vec{c})\left(\; Q_1|Q_2|\cdots|Q_n \;\right)$$

  where Q is one of A, B, C, D, E.
- Observe that in any state reachable from S_n **only one** of the Q is one of A, B, C, while **all other** Q are either of D, E.
- analyze the “meaning” of the those states

Proofs “by Hand” (II)

analyze the “meaning” of the following states for $n = 4$

$$(\boldsymbol{\nu}\vec{c})\left(\ D_1|E_2|A_3|E_n\ \right)$$

$$(\boldsymbol{\nu}\vec{c})\left(\ E_1|D_2|C_3|E_n\ \right)$$

Proofs "by Hand" (III)

Let $\{i\}, Y, Z$ be any partition of $\{0 \ldots, n{-}1\}$.

$$A_{i,Y,Z} \stackrel{\text{def}}{=} (\boldsymbol{\nu}\vec{c})\left(A_i \mid \prod_{j \in Y} D_j \mid \prod_{k \in Z} E_k \right)$$

$$B_{i,Y,Z} \stackrel{\text{def}}{=} (\boldsymbol{\nu}\vec{c})\left(B_i \mid \prod_{j \in Y} D_j \mid \prod_{k \in Z} E_k \right)$$

$$C_{i,Y,Z} \stackrel{\text{def}}{=} (\boldsymbol{\nu}\vec{c})\left(C_i \mid \prod_{j \in Y} D_j \mid \prod_{k \in Z} E_k \right)$$

Note that $S_n = A_{0,\{1\ldots,n-1\},\emptyset}$.

Proofs “by Hand” (IV)

Using the Expansion Law, we show that:

$$A_{i,Y,Z} \sim a_i.C_{i,Y,Z} + \sum_{k \in Z} b_k.A_{i,Y \oplus k, Z \ominus k}$$

$$B_{i,Y,Z} \sim b_i.A_{i,Y,Z} + \sum_{k \in Z} b_k.A_{i,Y \oplus k, Z \ominus k}$$

$$C_{i,Y,Z} \sim \sum_{k \in Z} b_k.A_{i,Y \oplus k, Z \ominus k} + \\ + \begin{cases} \tau.A_{i+1, Y \ominus (i+1), Z \oplus i} & \text{if } i+1 \in X \\ \tau.B_{i+1, Y, Z \ominus (i+1) \oplus i} & \text{if } i+1 \in Y \end{cases}$$

Proofs “by Hand” (V)

Let $\mathcal{R}$ be the relation containing the following pairs:

$$A_{i,Y,Z} \quad , \quad \mathsf{S}_{i,Z}$$

$$B_{i,Y,Z} \quad , \quad \mathsf{S}_{i,Z \oplus i}$$

$$C_{i,Y,Z} \quad , \quad \mathsf{S}_{i+1,Z \oplus i}$$

- $\mathcal{R}$ is a weak bisimulation (up to $\sim$).
- $\mathcal{R}$ contains the pair $(S_n, \text{Scheduler}_n)$.
- Q.E.D.