Software Modeling and Verification
Lehrstuhl für Informatik 2
RWTH Aachen University

Prof. Dr. Ir. J.-P. Katoen
Priv.-Doz. Dr. T. Noll
T. Han, M. Neuhäußer

# Modeling Concurrent and Probabilistic Systems

Winter Term 07/08

---

# – Solution 4 –

## Exercise 1 **(6 points)**

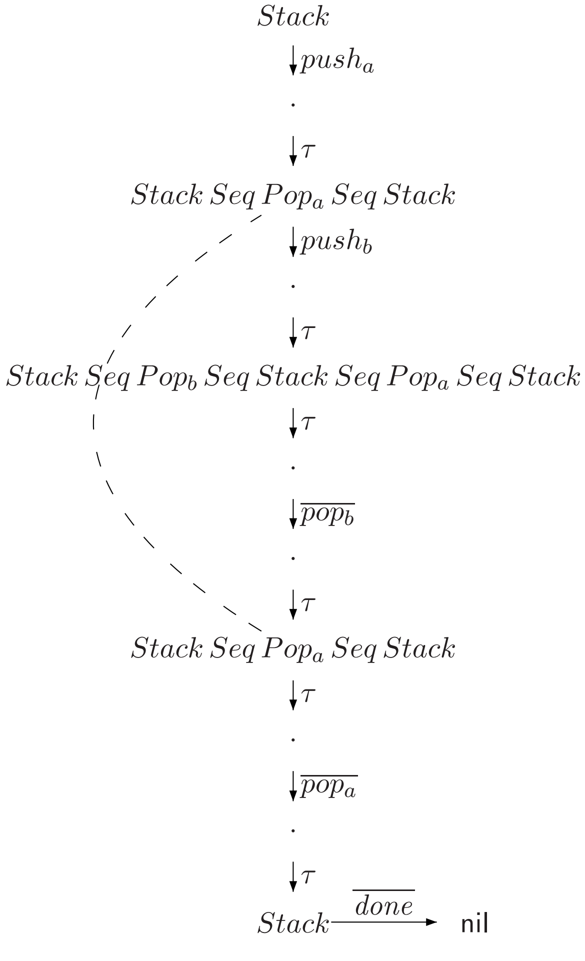
"Specification": (note: infinitely many equations, thus not a valid CCS process definition)

$$
\begin{aligned}
Stack_\varepsilon(\vec{a}) &= push_a.Stack_a(\vec{a}) + push_b.Stack_b(\vec{a}) + \overline{empty}.Stack_\varepsilon(\vec{a}) + Done(done)\\
Stack_{xs}(\vec{a}) &= push_a.Stack_{axs}(\vec{a}) + push_b.Stack_{bxs}(\vec{a}) + \overline{pop_x}.Stack_s(\vec{a})\\
&\quad \text{where } x \in \{a,b\}, s \in \{a,b\}^*, a = (push_a, push_b, pop_a, pop_b, empty, done)\\
Done(done) &= \overline{done}.\mathsf{nil}
\end{aligned}
$$

"Implementation":

$$
\begin{aligned}
Stack(\vec{a}) &= Push_a(push_a, done)\, Seq\, (Stack(\vec{a})\, Seq\, (Pop_a(pop_a, done)\, Seq\, Stack(\vec{a})))\\
&+ Push_b(push_b, done)\, Seq\, (Stack(\vec{a})\, Seq\, (Pop_b(pop_b, done)\, Seq\, Stack(\vec{a})))\\
&+ Done(done) \quad \text{where } \vec{a} = (push_a, push_b, pop_a, pop_b, done)\\
Push_x(push_x, done) &= push_x.Done(done) \quad (x \in \{a,b\})\\
Pop_x(pop_x, done) &= pop_x.Done(done) \quad (x \in \{a,b\})
\end{aligned}
$$

Example (according to our specification):
$Stack_\varepsilon \xrightarrow{push_a} Stack_a \xrightarrow{push_b} Stack_{ba} \xrightarrow{pop_b} Stack_a \xrightarrow{pop_a} Stack_\varepsilon \xrightarrow{\overline{done}} \mathsf{nil}$.
Implementation example:



**Exercise 2** **(4 points)**

First, recall the definition of a (deterministic) Turing machine. A deterministic Turing machine is a tuple:

$$\mathcal{A} = (Q, \Sigma, \Gamma, \delta, q_0, \Box, E)$$

where

- $Q$ is a finite set of states
- $\Sigma$ is the input alphabet
- $\Gamma \supset \Sigma$ is the working alphabet
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, N, R\}$ is the transition function,
- $q_0$ is the initial state,
- $\Box \in \Gamma \setminus \Sigma$ is the blanc symbol and
- $E \subseteq Q$ is the set of final states.

Here, without loss of generality, we assume that $\Sigma = \Gamma \setminus \{\Box\}$ and that $\delta$ has the form

$$\delta : Q \times \Gamma \to Q \times \Sigma \times \{L, N, R\},$$

i.e. that the blanc symbol $\Box$ can only be read and overwritten, but not written.

Now we provide a reduction from the problem whether a given Turing machine $\mathcal{M}$ on every of its computations visits only finitely many configurations to the problem whether a given CCS process definition induces a finite LTS.

We use the following idea to transform a deterministic Turing machine to a CCS process definition:

- Each state of $\mathcal{A}$ is represented by a process identifier
- $\mathcal{A}$'s tape is split into two stacks: *LStack* and *RStack*.
- The current position of the head is the top of stack *LStack*.

Intuitively, *LStack* contains the content of the tape up to (including) the current position of the head; *RStack* contains the remaining tape contents. For $x \in \Gamma$, we let

$$\begin{aligned} Pid &= \{TM, LStack(\vec{b}), RStack(\vec{c})\} \cup \{Control_q(\vec{a}) \mid q \in Q\} \text{ and} \\ TM_{\text{CCS}} &= \mathsf{new}\, \vec{a}\, (Control_{q_0}(\vec{a}) \| LStack(\vec{b}) \| RStack(\vec{c})) \text{ where} \\ LStack(lpush_x, lpop_x, lempty) &= Stack(lpush_x, lpop_x, lempty) \\ RStack(rpush_x, rpop_x, rempty) &= Stack(rpush_x, rpop_x, rempty) \end{aligned}$$

The transitions of $\mathcal{A}$ are represented in our CCS processes as follows:
Let $q, q' \in Q, x \in (\{\Box\} \cup \Sigma), a \in \Sigma$ and $d \in \{L, N, R\}$. For every transition

$$\delta(q, x) = (q', a, d)$$

of the deterministic Turing machine $\mathcal{A}$, introduce a corresponding nondeterministic choice in the process definition $Control_q$ that corresponds to state $q$ of $\mathcal{A}$ as follows:

$$Control_q(\vec{a}) = \ldots + \alpha.P + \ldots.$$

Here, $\alpha$ and $P$ reflect the semantics of $\mathcal{A}$'s transition as follows:

$$\alpha = \begin{cases} \mathit{lpop}_x & \text{if } x \in \Sigma \\ \mathit{lempty} & \text{if } x = \Box \end{cases}$$

$$P = \begin{cases} \overline{\mathit{lpush}}_a.\mathit{Control}_{q'}(\vec{a}) & \text{if } d = N \\ \overline{\mathit{rpush}}_a.\mathit{Control}_{q'}(\vec{a}) & \text{if } d = L \\ \overline{\mathit{lpush}}_a.\big(\sum_{b\in\Sigma} \mathit{rpop}_b.\overline{\mathit{lpush}}_b.\mathit{Control}_{q'}(\vec{a}) + \mathit{rempty}.\mathit{lpush}_{\Box}.\mathit{Control}_{q'}(\vec{a})\big) & \text{if } d = R \end{cases}$$

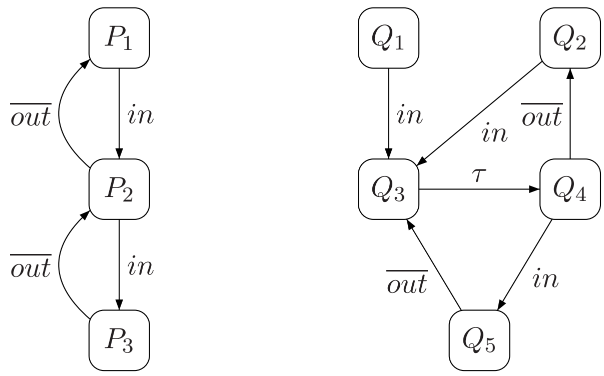
For a given Turing machine $\mathcal{M}$, the problem whether every computation of $\mathcal{M}$ visits only finitely many configurations is undecidable.

This completes our reduction as we now have: $TM$ induces finite LTS $\Leftrightarrow$ every computation of $\mathcal{A}$ visits finitely many configurations.

## Exercise 3 (4 points)

Recall the LTS of the two buffer implementations:

Specification: Implementation:




Apply the partition algorithm:

**(1)** Initial partition $\pi = \{S\} = \{\{P_1, P_2, P_3, Q_1, \ldots, Q_5\}\}$

**(2,3)** Successor blocks:

| $P$ | $P_1$ | $P_2$ | $P_3$ | $Q_1$ | $Q_2$ | $Q_3$ | $Q_4$ | $Q_5$ |
|---|---|---|---|---|---|---|---|---|
| $in(P)$ | $\{S\}$ | $\{S\}$ | $\emptyset$ | $\{S\}$ | $\{S\}$ | $\emptyset$ | $\{S\}$ | $\emptyset$ |
| $\overline{out}(P)$ | $\emptyset$ | $\{S\}$ | $\{S\}$ | $\emptyset$ | $\emptyset$ | $\emptyset$ | $\{S\}$ | $\{S\}$ |
| $\tau(P)$ | $\emptyset$ | $\emptyset$ | $\emptyset$ | $\emptyset$ | $\emptyset$ | $\{S\}$ | $\emptyset$ | $\emptyset$ |

**(4,5)** Decomposition:

$$\pi = \{\underbrace{\{P_1, Q_1, Q_2\}}_{B_1}, \underbrace{\{P_2, Q_4\}}_{B_2}, \underbrace{\{P_3, Q_5\}}_{B_3}, \underbrace{\{Q_3\}}_{B_4}\}$$

**(2,3)** Successor blocks of $B_1$:

| $P$ | $P_1$ | $Q_1$ | $Q_2$ |
|---|---|---|---|
| $in(P)$ | $\{B_2\}$ | $\{B_4\}$ | $\{B_4\}$ |
| $\overline{out}(P)$ | $\emptyset$ | $\emptyset$ | $\emptyset$ |
| $\tau(P)$ | $\emptyset$ | $\emptyset$ | $\emptyset$ |

**(4,5)** Decompose $B_1$ into $\{P_1\}$ and $\{Q_1, Q_2\}$

$\Rightarrow P_1 \not\sim Q_1$