

Modeling Concurrent and Probabilistic Systems

Summer Term 09

– Series 1 –

Hand in until Mai 6 before the exercise class.

Exercise 1

(3 points)

Construct the derivation tree for the τ -step in the LTS of the parallel two place buffer (lecture 2, slide 13).

Exercise 2

(4 points)

Give the LTS of the following process definitions:

a) $A(a, b, c) = a.b.c.A(a, b, c) + a.nil$

b) $B(a) = a.\bar{a}.nil + a.nil$

c) $C(a) = a.C(a) \parallel D$
 $D = D$

d) $E = \text{new } a \ (a.nil \parallel F(a))$
 $F(a) = \bar{a}.F(a)$

Exercise 3

(4 points)

Which of the following process definitions induce an infinite LTS? Justify your answers!

a) $A = A$

b) $B(a) = (B(a) \parallel B(a)) + a.nil$

c) $C(a) = (C(a) + C(a)) + a.nil$

d) $D(a, b) = a.(D(a, b) \parallel b.nil)$

Exercise 4

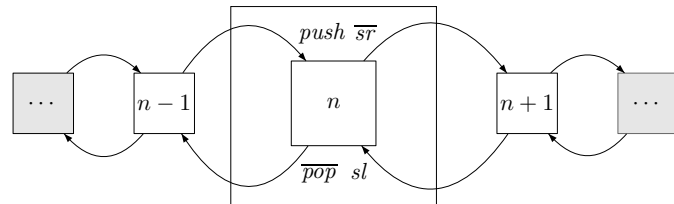
(3* points)

The aim of this exercise is to infer a finite stack data structure, i.e. a bounded LIFO queue:

To this aim, consider the following process definition of a shift register which stores any value received via actions *push* (or *sl*) and shifts its old value to the shift register on the right (on the left, resp.) via action $\bar{s}r$ (resp. $\overline{pop}$). Let $\vec{a} = (push, pop, sl, sr)$ and define

$$E(\vec{a}) = push.F(\vec{a}) + sl.F(\vec{a})$$

$$F(\vec{a}) = \overline{pop}.E(\vec{a}) + push.\bar{s}r.F(\vec{a})$$



Derive a process definition of a stack for at most three elements by appropriately composing three instances of the shift register! Construct the derivation tree of the *push*-transition of the empty stack!

Hint: You do not have to consider cases where the stack's capacity is exceeded!

*Additional exercise with extra points only.