CS3211 Parallel and Concurrent Programming – Guidelines for tutorial 3 (8-12 Feb 2010)

Content:

Discussion on Processes Threads, Concurrent Execution (Lecture 4).

The prefix operator (->), choice operator (|), recursion operator and parallel composition operator (||) have been discussed so far in class.

Sample Exercises:

[Please conduct these as an interactive discussion, rather than an evaluation. Please also make it clear to the students that they are not being evaluated for their performance in these exercises, so that they are not afraid to make mistakes while answering.]

1. A digital circuit receives a sequence of trigger inputs and alternately outputs 0 and 1. Model the process CIRCUIT using FSP, and check that it produces the required output, i.e., it should perform the actions given by the trace:
   trigger ->1->trigger->0->trigger->1->trigger->0…

   ```
   CIRCUIT =  trigger -> 1 -> trigger -> 0 -> CIRCUIT
   ```

[Discuss the state machine description as well during the tutorial, it can be easily constructed from the above description.]

2. A) A variable stores values in the range 0…N and supports the actions read and write. Model the variable as a single process using FSP. You may assume that the variable is initialized to 0. For N=2, check that it can perform the actions given by the trace
    write.2 ->read.2->read.2->write.1->write.0->read.0

```
VARIABLE = VAR[0].

VAR[v:0..N] =  ( (read[v] -> VAR[v] )  |  (write[u:0..N]  -> VAR[u])).
```

[Discuss the state machine description as well during the tutorial, it can be easily constructed from the above description.]

2 B) What would change in the above if the variable initialization is not specified.

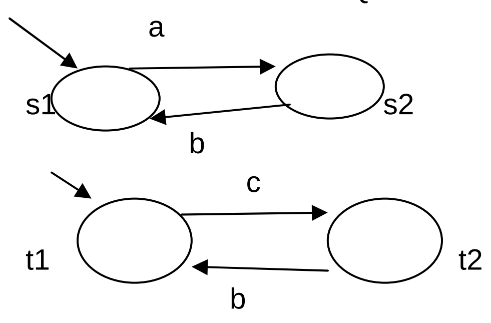
```
VARIABLE = (write[v:0..N] -> VAR[v]).

VAR[v:0..N] =  ( (read[v] -> VAR[v] )  |  (write[u:0..N]  -> VAR[u])).
```

3. Consider the following concurrent process descriptions
   P = (a -> b-> P).
   Q = (c -> b->Q).
   S1 = (P || Q)
   S2 = (a->c->b->S2 | c -> a -> b -> S2).

   How can you argue that the processes S1 and S2 are equivalent in terms of allowed execution traces?

   Answer: Construct the state machine description of S1 and S2 [Do so from the state machine of P and Q – they are shown below. I have marked the states of the state machine for P,Q as well. The states of P||Q will be the composition of states from P, Q – that is, of the form, (s1,t1), (s1,t2) and so on.]

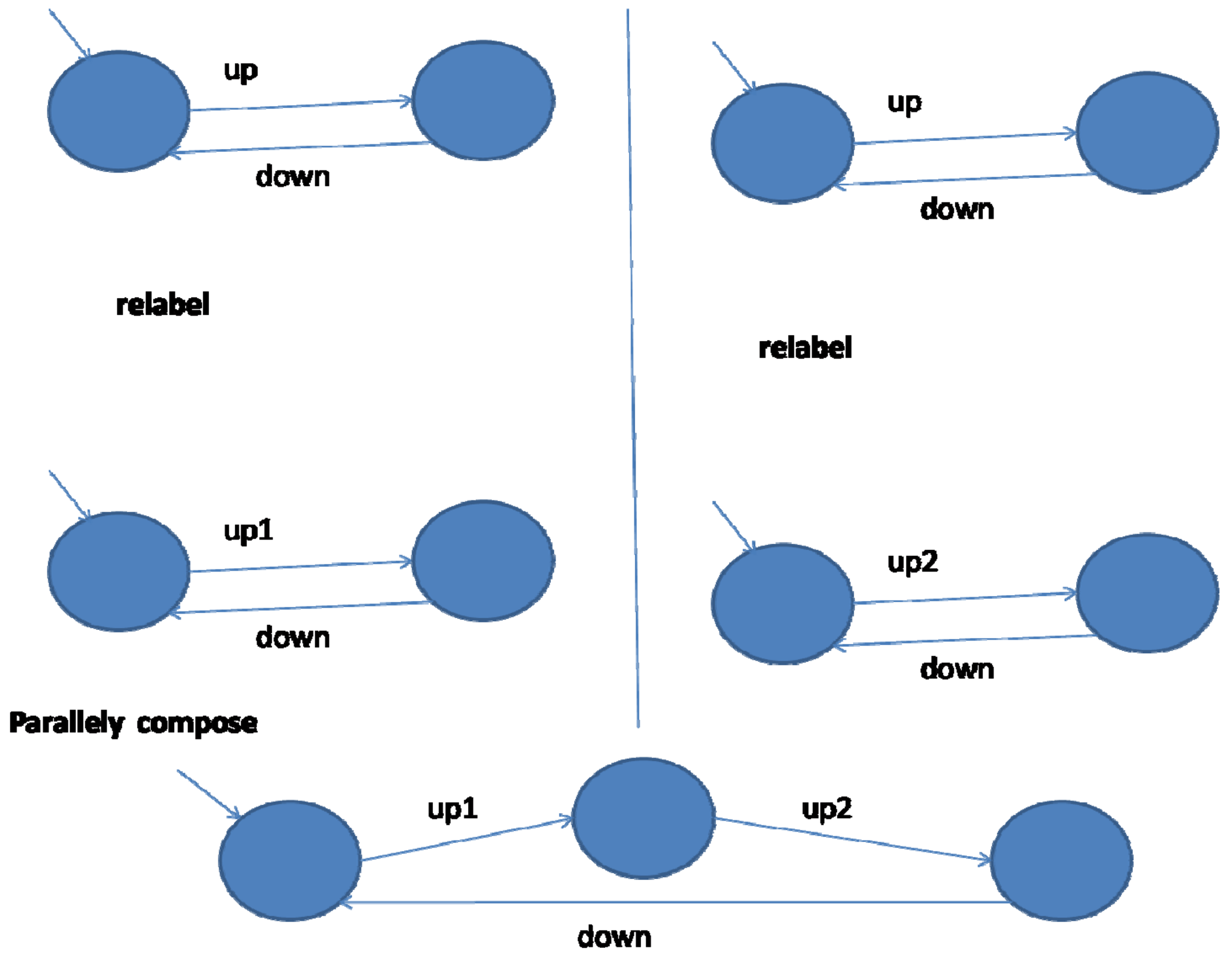
   State machine for P and Q are as follows.



   For S1, note that b is a shared action across P and Q.

4. Consider the process ELEMENT = (up -> down -> ELEMENT)
   Using parallel composition and the ELEMENT process describe a model that can accept FOUR up actions before a down action.

The solution for a process which performs 2 up actions prior to a down action is shown in the following. It is shown by state machine manipulations. The same can be written as process equations.


up
down
up
down
relabel
relabel
up1
down
up2
down
Parallely compose
up1
up2
down