NTNU

Norwegian University of Science and Technology

Department of Engineering Cybernetics

TTK4145 - Real Time Programming

Exercise 9 - FSP and LTSA

April 21, 2014

1 Introduction

A lot of relevant information can be found around this web page:
`http://www.doc.ic.ac.uk/~jnm/book/`.
An explanation of the FSP notation can be found here:
`http://www.doc.ic.ac.uk/~jnm/LTSdocumention/FSP-notation.html`.

2 Installing LTSA

1. Download *LTSAtool.zip* from the "Files for Exercises" folder;
2. Extract the files, for instance to Desktop/;
3. Find "LTSA.jar", right click on it, and choose "Open with 'Sun Java 6 Runtime' ";

or install directly from a shell with the following commands:

```
cd ~/Desktop
unzip LTSAtool.zip
java -jar ltsatool/ltsa.jar
```

3 Introductory assignments

Read through reference manuals and experiment. You can find examples in "FSPEksempel.doc". Even more examples can be found under the File | Examples menu in LTSA.

4 Safety / Deadlock

Assignment: The absence of *deadlocks* is a built-in safety criterion in LTSA. Create a system with a *deadlock*. Does LTSA detect it?

It is easy to see from the state model that we have a *deadlock*, because there will be a state in the diagram that there is no way to get out from. How can you detect a *deadlock* from the FSP model itself?

5 Progress / Livelock

If nothing else is specified, LTSA uses the following progress criterion: From all states, all actions should be reachable within a finite number of steps.
Assignment: Create a system with a *livelock*, in other words, a system where there is a subset of states that there is no way to get out from. How is this detected by LTSA?
Assignment: Assume that the subset is not a *livelock*, but normal behaviour. Create a progress property that contains only those states that are part of the

livelock to get rid of the error.
(See `http://www.doc.ic.ac.uk/~jnm/book/pdf/ch7.pdf` for the use of "progress" and "property").

6 Dining philosophers

Assignment: Model a system with 3 philosophers and 3 forks and demonstrate a deadlock. See `http://en.wikipedia.org/wiki/Dining_philosophers` to refresh your memory of the problem.
Assignment: Extend the FSM description to handle N philosophers by using indexing, prefixing, relabeling etc. How many philosophers can LTSA handle? Be a little careful, trying a very large number can make the computer rather unresponsive.
Assignment: We want to get rid of the deadlock. Make one of the philosophers left-handed (picking up the forks in the other order). Does that solve the problem?

The left-handed philosopher has consequences for fairness. Why? Can you come up with (or Google..) fair solutions that does not have a deadlock?