Prof. Dr. Andreas Podelski
Tanja Schindler

Hand in until November 28th, 2018
15:59 via the post boxes
Discussion: December 3rd/4th, 2018

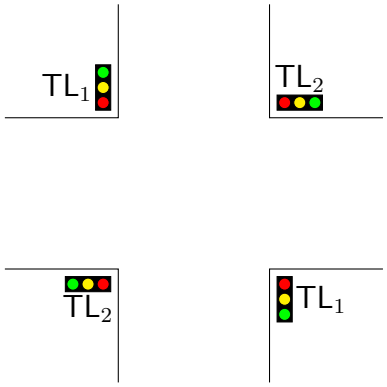
Tutorial for Cyber-Physical Systems - Discrete Models
Exercise Sheet 6

The goal of this sheet is to train your understanding of different synchronization and communication mechanisms in Cyber-Physical Systems. You have already worked on communication via shared variables on the last sheet; the exercises on this sheet deal with synchronization via handshaking and with synchronization via communication over channels.

Exercise 1: Traffic Lights

The goal of this exercise is to train your ability to model a system where the processes synchronize via handshaking. This exercise also serves as a warm-up for the second exercise. The situation in the second exercise will be more complicated, because there, we take into account a controller.

Consider the crossing of two roads with four traffic lights as depicted on the right. The two traffic lights labelled with TL_1 always show the same color, and likewise the two traffic lights labelled with TL_2 always show the same color. The traffic lights have three modes: `red`, `yellow`, and `green`, and they switch from `green` to `yellow`, from `yellow` to `red`, and from `red` to `green`.



(a) Create two transition systems TS_1 and TS_2 for the traffic lights, one for each direction of a crossing.

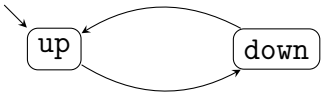
Insert suitable actions on which these system can synchronize so that at least one of the lights are in the `red` mode in each state of the transition system $TS_1 \| TS_2$.

(b) Compute the transition system $TS_1 \| TS_2$. Is the system safe? An informal argument is sufficient.

Exercise 2: Railroad Crossing

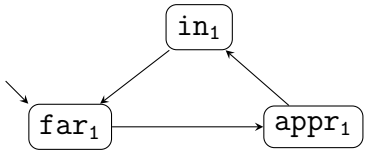
In the lecture you have already seen a railroad crossing system. That system was wrong. Now you are going to design a correct system. In the lecture, we also mentioned another wrong railroad crossing system where the error appeared only in the situation where we had two trains. This is why here we take into account two tracks.

In this exercise we build a model for the controller of a railroad crossing. Our railroad crossing has one gate and two train tracks, one track for each direction.
The transition system of the gate has two states and the following graph structure.

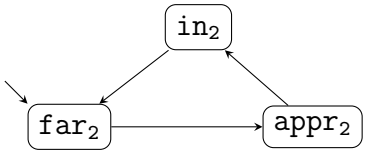


The transition systems of the train tracks have three states. In the first state all trains on this track are far away, in the second state one train is approaching, in the third state one train is in the railroad crossing and no other train is approaching on this track. The transition systems of the train tracks have the following graph structure.

Track 1:



Track 2:



Describe a controller (in the form of a transition system) that controls the gate such that whenever a train is in the railroad crossing (state $\texttt{in}_1$ or state $\texttt{in}_2$),the gate is down (state `down`). Your controller may temporarily stop a train in the sense that a train may only move from $\texttt{appr}_i$ to $\texttt{in}_i$ if the controller agrees.
Complete the transition system descriptions of train tracks and gate by adding suitable actions to the graphs given above.
The system should have the property that every train can pass the gate eventually and that the gate is not always down. Hence, e.g., the trivial controller that just stops every train or the trivial controller the keeps the gate down are not valid solutions here.

Exercise 3: Modeling a Channel System

The goal of this exercise is to train your abilities to model a Cyber-Physical system where the components use channels for the communication.

Consider the following leader election algorithm:
For $n \in \mathbb{N}$, n processes $P_1, \ldots, P_n$ are located in a ring topology where each process is connected by an unidirectional channel to its neighbor in a clockwise manner. To distinguish the processes, each process is assigned a unique identifier $id \in \{1, \ldots, n\}$. The aim is to elect the process with the highest identifier as the leader within the ring. Therefore each process executes the following algorithm:

```
send(id);                                  ▷ initially the process sends its id
while true do
    receive(m);
    if m = id then
        stop;                              ▷ process is the leader
    end if
    if m > id then
        send(m);                           ▷ forward identifier
    end if
end while
```

(a) Model the leader election protocol for n processes as a channel system. That is, give a graph that represents the way the processes communicate with each other, and give a graph that represents the behavior of each single process. Do this twice: for $n = 3$ and (using the $\ldots$ notation) for general n.

(b) Give an initial execution fragment of $\mathcal{T}([P_1|P_2|P_3])$ such that at least one process has executed the send statement within the body of the while loop. Assume for $0 < i \leq 3$, that process P_i has identifier $id_i = i$.