Technische Universität München (I7) **Winter Semester 2021-22**
J. Křetínský / A. Evangelidis / K. Grover **28.10.2021**

Quantitative Verification – Exercise sheet 2

Exercise 2.1

Consider the scenario of traffic lights (green and red) at a T-junction between a highway and a local road. The lights on the highway will always be green and light on the local road will always be red unless a vehicle is detected by a sensor in the local road. The sensor on the local road will be turned off when the light there is green. The following points describe the complete behaviour of the traffic lights:

- Assume that the vehicles do not go from highway to local road.
- When the sensor detects a vehicle on the local road, the light there should become green within 30 seconds and should stay green for at least 20 seconds.
- The lights on the highway should be green for at least 30 seconds in each cycle.

1. Model this scenario using Timed Automata.
2. Model your automaton in UPPAAL and check that both the lights should never be green at the same time. (Hint: Use different automata for highway traffic light and local road traffic light, and synchronize them.)

Exercise 2.2

Construct a system modeling trains from multiple tracks crossing a bridge with a single track. The expected behaviour of the train is elaborated below.

- When the train approaches a bridge, it sends a signal to the controller. If the bridge is occupied, the controller sends a stop signal to the train.
- If the train doesn't receive a stop signal within 10 time units, it enters the bridge within 20 time units. It takes the train 3 to 5 time units to cross the bridge.
- If the train receives a stop signal within 10 time units, it comes to a stop. When it receives a go signal from the controller, it starts moving within 15 time units and it takes atleast 7 time units to enter the bridge.

Design a controller which uses an FCFS strategy to process requests. If the bridge is free and a train requests to use it, add it to a queue. When the train leaves, remove it from the queue. If the bridge is being used, always add the train to the queue and ask it to wait until its turn. Verify the following properties

- Gate can receive (and store in queue) msg's from approaching trains.
- Train 1 can reach crossing.
- Train 0 can be crossing bridge while Train 1 is waiting to cross.
- Train 0 can cross bridge while the other trains are waiting to cross.
- There is never more than one train crossing the bridge (at any time instance).
- There can never be N elements in the queue (thus the array will not over ow).
- Whenever a train approaches the bridge, it will eventually cross.
- The system is deadlock-free.

Source: UPPAAL Demos

Exercise 2.2

The problem consists of a number of aircrafts that need to land on limited number of runways. Each aircraft has a preferred target landing time, corresponding to arriving at the runway with cruise speed and landing immediately. However, the aircraft can speed up and land earlier or stay longer in the air and land later if necessary. Due to extra gas used in accelerating, landing earlier has an added cost which grows linearly until some fixed earliest landing time. Late landings have an immediate constant penalty for landing late and a cost growing linearly with the amount of gas used for staying in the air until all latest possible (controlled) landing time. Furthermore, aircrafts cannot land back to back on the same runway due to wake turbulence. Thus, there are certain minimum constraints on the separation delay between aircrafts of different types. Your task is to use UPPAAL to model a flight and a runway and run in parallel, a system which is composed of 10 flights and 2 runways. You may use *https://www.in.tum.de/fileadmin/w00bws/i07/2021ws/qv/apl-system.txt* as your Systems declaration. Now, the problem is to assign landing times and runways to all aircrafts while respecting individual timing constraints for the aircrafts and the wake turbulence constraints.

Source: UPPAAL Demos