Formal techniques for software and hardware verification

Lecturers:

Vladimir Zakharov

Vladislav Podymov

e-mail:

valdus@yandex.ru

2020, fall semester

Seminar 10–11

UPPAAL

exercises

Exercise 1: learning UPPAAL by example

Open, understand, simulate, and verify the following examples in the UPPAAL package:

<*uppaal*>/demo

1. `bridge`
2. `2doors`
3. `fischer`

Exercise 2: smart lamp

The lamp has three regimes: **off**, **dim**, **bright**

Initially the lamp is **off**

The regimes are switched with a button
which can be pressed and then released (*just like a usual button*)

When the button is released,
the lamp immediately switches its regime:

- if the button was pressed for less than a second:
  - **off**$\rightarrow$**dim**
  - **dim**$\rightarrow$**off**
  - **bright**$\rightarrow$**off**
- if the button was pressed for at least a second:
  - **off**$\rightarrow$**bright**
  - **dim**$\rightarrow$**bright**
  - **bright**$\rightarrow$**dim**

Create a model for the lamp, and ensure that the model is correct

Exercise 3: chinese juggler

Exercise 3: chinese juggler

N plates are spinning before the juggler

When (if) a plate stops spinning, it falls and breaks

Initially (when the system starts) each of the plates
is guaranteed to spin for at least 5 seconds

The juggler is allowed to spin the plates as he wishes,
at most one plate at a time

While the juggler spins a plate,
the plate cannot stop (and fall, and break)

If the juggler have spun a plate for ...

- ... at least 1 second,
  then the plate is guaranteed to spin for at least 3 more seconds
- ... more than 2 seconds,
  then the plate is guaranteed to spin for at least 5 more seconds

How many plates can the juggler keep spinning indefinitely?

Exercise 4: debugging an ATM model

(*the required* `atm` *system can be found in a zip-file on the course page*)

Eric wants to withdraw some cash from his bank account via an ATM, but something went horribly wrong

Try to fix the model in such a way that Eric is able to withdraw as much money as his bank account and the ATM's till allow, and not a single cent more

Exercise 4: debugging an ATM model

The `bank_card` channel models insertion and return of a bank card

The `request` channel models a request to get 10 currency units

The `cash` channel models a cash issue

When Eric gets cash, he stores it in a pocket (`cash_in_pocket`)

If the ATM has enough cash `in_till`,
it requests the bank for a permission to issue (`ask_permission`),
and the permission is either granted (`OK`) or denied (`not_OK`)
depending on Eric's `balance`

The card should be returned in any case
after all the operations are complete

Exercise 5: debugging a train crossing

(*the required* `traingate` *system can be found in a zip-file on the course page*)

N trains are crossing a bridge with one railway line

On arrival, each train obeys the dispatcher who might order to `stop` or to go

When a train arrives at the bridge (`appr`), but is still far enough, it can be stopped

When a train is near, it takes the bridge line until it crosses the bridge and leaves (`leave`)

Fix the model in such a way that all the formulas in the verification tab are sarisfied