# Introduction to Lab 4

## Modeling and Verification using Uppaal

Martin Stigge <martin.stigge@it.uu.se>

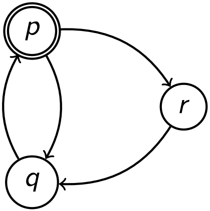
11. October 2010

# Lab 4: Modeling and Verification using UPPAAL

- Lab goals:
  - Practice formal modeling and verification of RTS
  - Work with timed automata and UPPAAL
- Lab preparation:
  - *Form groups:* 2 or 3 students each
  - Lab will be done on Tue, 12.10. in room 1515D
  - Have a look at the lab homepage
    `http://www.it.uu.se/edu/course/homepage/realtid/ht10/lab4`
  - (“Small Tutorial” is recommended reading!)
- Lab report:
  - Answers (models, queries, values) to the questions
  - To my mailbox, building 1, floor 4
  - *Deadline: Mon, 18.10., 10:00*

# Finite Automata

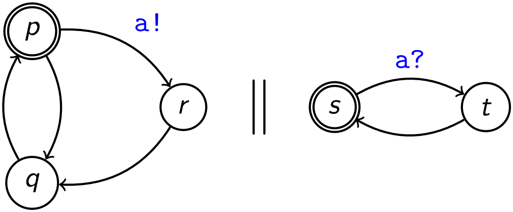
- Theoretic model for systems (or whatever else)
- *Locations* and *transitions* (drawn as nodes and edges)




- State space: Set of locations
- Trace semantics:
  - One possible trace: $p \to q \to p \to r \to q \to \dots$
  - Another one: $p \to r \to q \to p \to r \to \dots$
  - *Not* a trace: $p \to r \to p \to \dots$

# Networks of Finite Automata

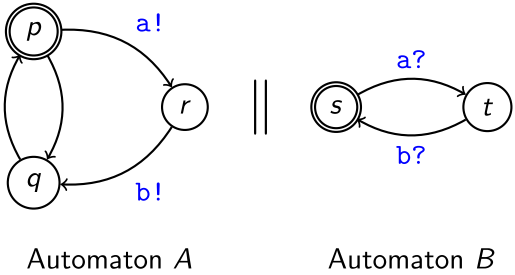
- Compose several automata into *networks*
- Use *synchronization* on edges/transitions




- State space: Product of location sets
- Trace semantics:
  - Interleaving, i.e., one automaton at a time
  - Except: Synchronized edges are taken *together*
  - E.g.: $(p, s) \rightarrow (q, s) \rightarrow (p, s) \xrightarrow{\mathtt{a}} (r, t) \rightarrow (r, s) \rightarrow \ldots$
  - *Not* a trace: $(p, s) \rightarrow (r, s) \rightarrow \ldots$

# Finite Automata: Model Checking

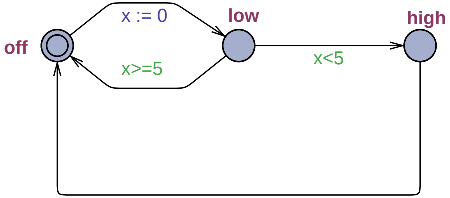
- Does a model *satisfy* some property $\varphi$?




- Property: "Does $A.r$ imply $B.t$?"
  - $\varphi :=$ `A[] (A.r imply B.t)`
  - Means: "In each state of each trace, $B$ is in $t$ whenever $A$ in $r$"
- Is *satisfied* in above example
- (Not satisfied without $b$ synchronization!)

# Timed Automata

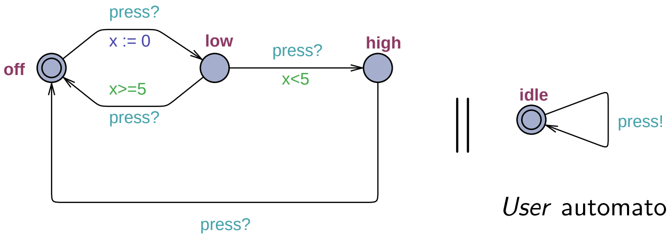
- Extend finite automata with *clocks*:




- Clocks have *real* values
  - All increasing at same pace
  - Can be reset and compared
- State space: Location $\times$ Clock valuations
- Trace semantics: Additional *delay* transitions
  - $(\mathit{off}, 0) \xrightarrow{\delta} (\mathit{off}, 1.2) \rightarrow (\mathit{low}, 0.0) \xrightarrow{\delta} (\mathit{low}, 5.7) \rightarrow (\mathit{off}, 5.7) \rightarrow (\mathit{low}, 0.0) \xrightarrow{\delta} (\mathit{low}, 2.3) \rightarrow (\mathit{high}, 2.3) \rightarrow \ldots$

# Networks of Timed Automata

- Compose just like before, using synchronized edges




*Lamp* automaton

*User* automaton

- In UPPAAL:
  - Sync. channels need to be *declared*
  - (As well as clocks and variables)

Demo

# Lab Assignment

- Part 1: Warm-Up
  - Model 3 simple automata
  - Use verification for simple properties
- Part 2: Scheduling
  - Setting: Schedule jobs to CPUs
  - One automaton *per job* and *per CPU*
  - Determine minimal execution time
- Part 3: Deadlock detection
  - Model Buffer, Producer and Consumer from Ada lab
  - Use verifier to *find deadlocks*
    - "Deadlock" means: Only time may pass (for all future)
  - Use simulator to analyze them
  - Remove all deadlocks

# The End

Questions?