UPPAAL MODELLING LANGUAGE

Modeling Real-Time Systems

15.02.2018

Deepak Pal

Basic Definitions

- **Template**
  - The idea is to define templates (like in C++) for processes that are instantiated to have a complete system.
  - The motivation for the templates is that system often have several processes that are very alike. The control structure (i.e., the locations and edges) is the same, only some constant or variable is different.
  - A template may also have have local variables and clocks.

- **Constants** are declared as const name value. Constants by definition cannot be modified and must have an integer value.
  - **Const** int x= 5 ;

Basic Definitions

- **Binary synchronisation**
  - channels are declared as **chan c**. An edge labelled with **c!** synchronises with another labelled **c?.**
  - A synchronisation pair is chosen non-deterministically if several combinations are enabled.

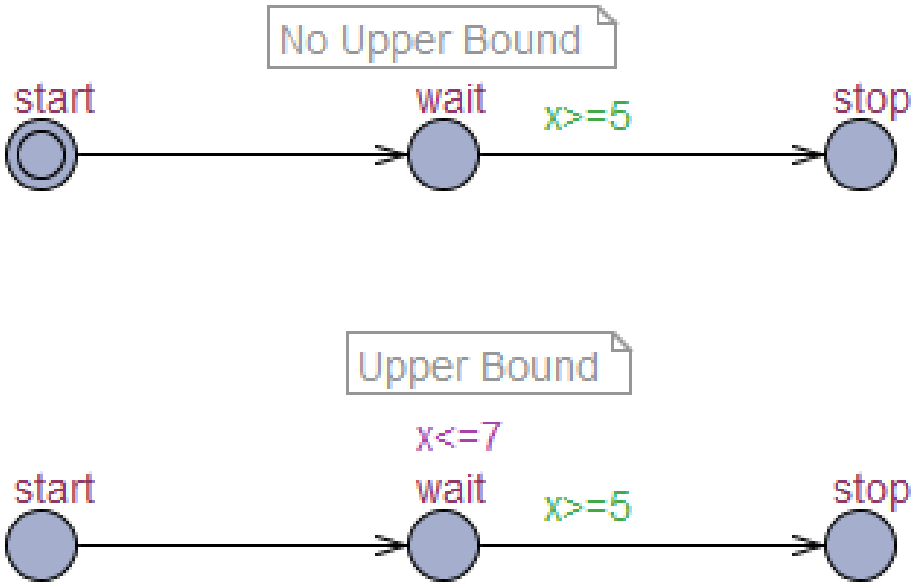
- **Broadcast channels**
  - are declared as **broadcast chan c**.
  - In a broadcast synchronisation one **sender c!** can synchronise with an arbitrary number of **receivers c?.**
  - **A**ny receiver than can synchronise in the current state must do so.
  - If there are no receivers, then the sender can still execute the c! action, i.e. **broadcast sending is never blocking**

Basic Definitions

- **Guard**: A guard is a particular expression
  - satisfying the following conditions:
  - it is side-effect free; it evaluates to a boolean;
  - only clocks, integer variables, and constants are referenced (or arrays of these types)

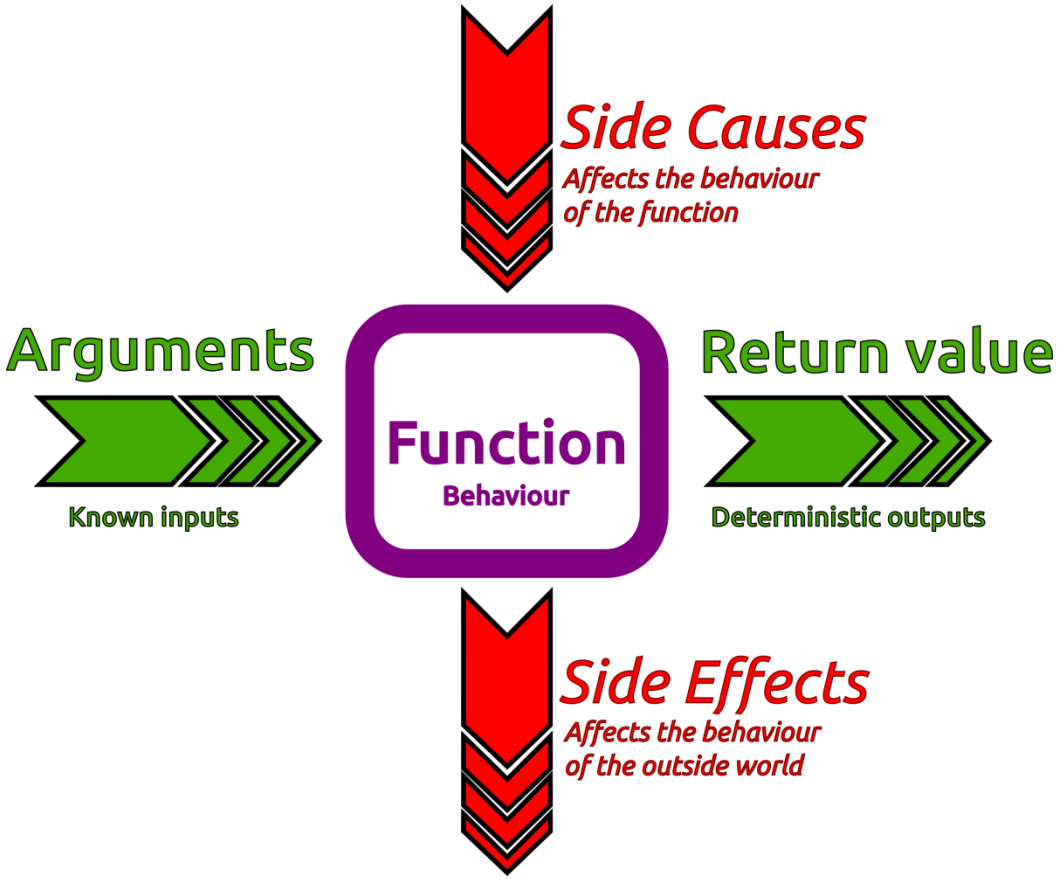
- **Update**
  - An update label is a comma separated list of expressions with a sideeffect; expressions must only refer to clocks, integer variables, and constants and only assign integer values to clocks. They may also call functions.

Time and clocks



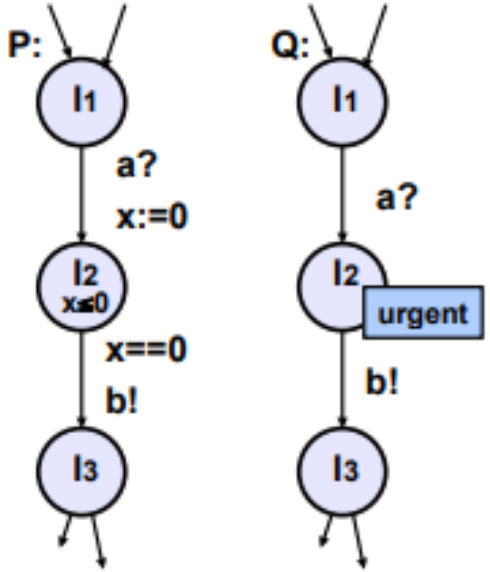
Side effects

A side effect is something that creates a change outside of the currrent code scope, or something external that affects the behaviour or result of executing the code in the current scope.



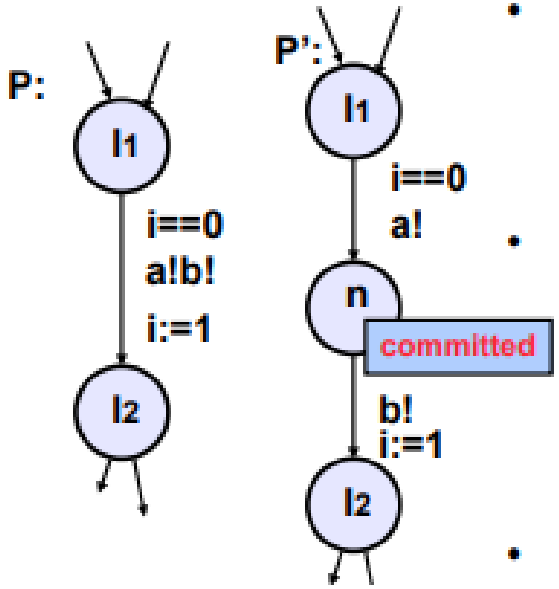
Basic Definitions

- **Urgent locations**
  - are semantically equivalent to adding an extra clock x, that is reset on all incoming edges, and having an invariant x<=0 on the location. Hence, time is not allowed to pass when the system is in an urgent location



Basic Definitions

- **Committed locations**
  - are even more restrictive on the execution than urgent locations. A state is committed if any of the locations in the state is committed. A committed state cannot delay and the next transition must involve an outgoing edge of at least one of the committed locations.



Why Committed locations ?

- A trick of modeling (e.g. to model multi-way synchronization using handshaking)

- **More importantly**, To reduce the size of the state space by reducing interleaving using committed locations, thus speeding up the verification.
  - In fact, it is a simple form of 'partial order reduction'

- It is used to avoid intermediate states, interleavings:
  - Committed states are not stored in the passed list
  - Interleavings of any state with a committed location will not be explored

State Explosion Problem

state-explosion problem

- **The number of states of a model can be enormous.**
  - **For example, consider a system composed by n processes, each having m states. Then, the asynchronous composition of these processes may have m^n.**
  - **In model checking we refer to this problem as the state explosion problem. All model checkers suffer from it**

Exponential Growth of …
… global state space in number of concurrent components.
… memory states in memory size.

ATM


bank_card
request
bank_card
cash
ATM
24 HOUR BANKING
ask_permission
OK
not_OK

A JOBSHOP

We suppose that two people are sharing the use of two tools — a hammer and a mallet — to manufacture objects from simple components.

Each object is made by driving a peg into a block. We call a pair consisting of a peg and a block a job; the jobs arrive sequentially on a conveyor belt, and completed objects depart on a conveyor belt.

To make the example more specific, we shall assume that the nature of the job influences the jobber's actions in a particular way. We suppose that he may use three predicates **easy, average and hard over jobs,** to determine whether a job is **easy or hard or average**. He will do easy jobs with his hands, hard jobs with the hammer, and mallet and average jobs with either hammer or mallet.