Modelling using *timed automata* (*TA*)

Ing. Michal Riša, irisa@fit.vutbr.cz
supervised by
Strnadel Josef, Ing., Ph.D., strnadel@fit.vutbr.cz

18. decembra 2015

Vysoké učení technické v Brně
Fakulta informačních technologií

Contents

1. Terminology explanation,
2. introduction to real-time/embedded systems,
3. why modelling of real-time/embedded systems? ,
4. timed automata,
5. example model,
6. references.

Key ideas

- Timed automaton is a finite state machine extended with clocks. This allows us to model and analyze time-oriented systems.
- Feel free to ask questions.

  "*Teacher does bad job if the student learns nothing.*" *by Strnadel Josef, Ing., Ph.D. .*

- Except model shown later, none of information presented is my own work. The information can be found through references section.

Terminology

Real-time system (*RT*)

A real-time system is a copmputer system which deals with real time. This means that there are requirements on timing of system's actions.

For example, it is required that after you hit a stop button in a factory, production process halts within 5 seconds.

Terminology

Embedded system (ES)

An embedded system is a computer system embedded in a device. Ordinary user does not know that the device contains a computer system.

Washing machine and it's control unit are ilustrative examples.

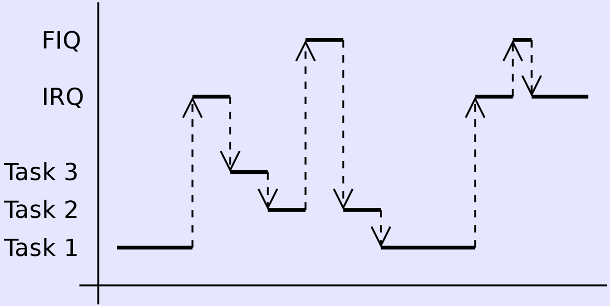
Terminology

A timing constraint

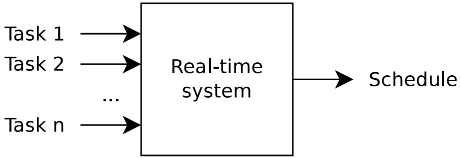
System's response for event A must occur within 15 ms after event arrival.

A task schedule

Schedule specifies "what runs when". It is created during real-time system operation by the system kernel.



RT system - logical model



- System function is divided to execution units (tasks/processes/threads).
- Execution units react to input events.
- Priorities, time constraints.
- *RT* system tries to create schedule that satisfies given time constraints.

Why modelling?

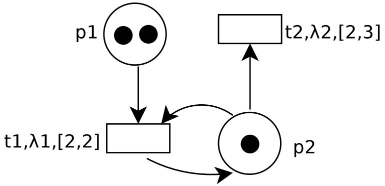
- Fast, reliable, low-power.
- Is the system I designed:
  - Fast enough to satisfy all time constraints?
  - Reliable enough to avoid damage?

  - Energy-efficient to be baterry-powered?

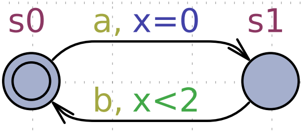
- Build it and measure. Or model it and find out before production.

Modelling *RT* systems

- Timed petri nets,



- timed process algebras,
- **timed automata** (*TA*).



TA - input

Time sequence, timed word, timed language

Let R^+ be a set of non-negative real numbers. A **time sequence** $\tau = \tau_1\tau_2...$ is an infinite sequence of time values $\tau_i \in R^+$ satisfying:

- $\tau_i > 0$
- $\tau_i \leq \tau_{i+1}$
- $\forall t \in R^+ : \exists i \geq 1 : \tau_i > t$

A **timed word** over alphabet Σ is a pair (σ, τ) where $\sigma = \sigma_1\sigma_2...$ is an infinite word over Σ and τ is a time sequence.

A **timed language** over Σ is a set of timed words over Σ.

TA - input

The **timed word** (σ, τ) is viewed as an input to an automaton.

- σ_i and τ_i represent and event and time of it's arrival.
- Time sequence τ is non-decreasing. Multiple events at the same time.

Example - a timed language

Example 1

$\Sigma = \{a, b\}$. A timed language L_1 consists of timed words (σ, τ) where there is no b after time 5.6 .

$$L_1 = \{(\sigma, \tau) | \forall i : ((\tau_i > 5.6) \Rightarrow (\sigma_i = a))\}$$

Example - a timed language

Example 2

$\Sigma = \{a, b\}$. A timed language L_2 consists of timed words (σ, τ) where a and b alternate and time difference between a and b in pairs keeps increasing.

$$L_2 = \{((ab)^\omega, \tau) | \forall i : ((\tau_{2i} - \tau_{2i-1}) < (\tau_{2i+2} - \tau_{2i+1}))\}$$

τ_{2i-1}	τ_{2i}	τ_{2i+1}	τ_{2i+2}
1	2	3	4

$$(\tau_2 - \tau_1) < (\tau_4 - \tau_3)$$

TA - clocks and clock constraints

Clock

A clock is represented by a variable whose values satisfy *time sequence* properties.

Clock constraints

For a set X of clock variables, the set $\delta = \alpha(X)$ is defined inductively by $\delta := x \leq c | c \leq x | \neg\delta | \delta_1 \wedge \delta_2$ where:

- $x \in X$ is a clock variable,
- $c \in Q^+$ is a non-negative rational constant,

and is referred to as clock constraints. Other constraint syntax definitions are also possible.

TA - clock constraints, clock interpretation

A clock interpretation

A clock interpretation is a mapping $v : X \to R^+$. The mapping assigns each clock in X a real value. We say that a clock interpretation v for X satisfies a clock constraint δ over X if δ evaluates to true using the clock values given by v.

Example

Given

$X = \{clk\}, v(X) = \{clk \to 4.7\}, \alpha(X) = \{clk < 5.1 \wedge clk < 1.7\}$,

constraint

- "$clk < 5.1$" evaluates to true, and
- "$clk < 1.7$" evaluates to false. Thus
- "$clk < 5.1 \wedge clk < 1.7$" evaluates to false.

TA - timed transition table

Timed transition table

$A = (\Sigma, S, S_0, C, E)$ is a timed transition table with following elements:

- Σ is a finite alphabet,
- S is a finite set of states,
- $S_0 \in S$ is an initial state,
- C is a finite set of clocks,
- $E \subseteq S \times S \times \Sigma \times 2^C \times \alpha(C)$ is a set of transitions. An edge (s, s', a, B, δ) represents a transition from state s to state s' on input symbol a. The set $B \subseteq C$ contains clocks to be reset with this transition and δ is a clock constraint over C.

TA - definition and operation

TA

A timed automaton $M = (\Sigma, S, S_0, C, E, F)$ is a tuple where

- (Σ, S, S_0, C, E) is the timed transition table, and
- $F \subseteq S$ is the set of accepting states.

The automaton M operation is as follows:

1. Given a timed word (σ, τ), the *TA* starts in the initial state with all clocks in C initialized to 0.
2. As time advances, the values of all clocks advance at the same rate.
3. At time τ_i the *TA* changes state from s to s' using some transition of the form $(s, s', \sigma_i, B, \delta)$ reading the input σ_i if the current values of clocks satisfy time constraint δ.
4. With this transition the clocks in B are set to 0.

TA - accepted language

Accepted language

Accepted language for a TA is a set of timed words (σ, τ) for which the automaton

- does not halt (deadlock), and
- the automaton is in one of accepting states.

$L(M) = \{(\sigma, \tau) | s \in F \wedge M \text{ does not halt}\}$

Let M be a *TA* model of some *RT* system with entire timing information. Let I be a set of all possible timed words $(\sigma, \tau) \Rightarrow I$ contains all possible event and time flows. If the *TA* M accepts all elements $\in I$, the *RT* system will satisfy it's timing requirements (if the model is correct and the *RT* system will face only timed words $\in I$).

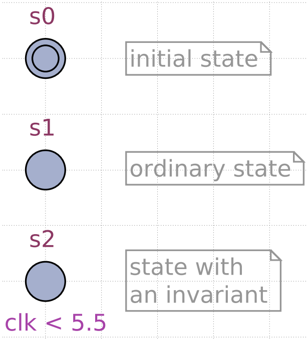
Graphical representation of a *TA*

Following slides are closely related to UPPAAL tool [**?**]. A *TA* is extended finite state machine → some symbols are inherited from *FSM* notation. The *TA* consists of

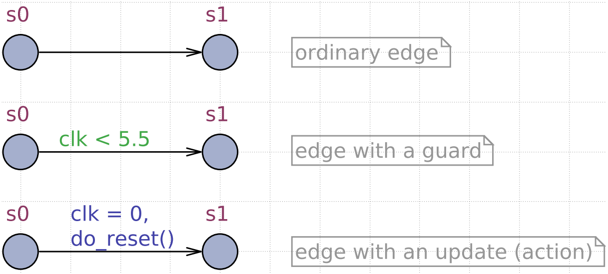
- states,
- edges,
- clocks,
- *constraints (invariants, guards),
- *actions and
- *synchronization.

* more on this later.

Graphical representation - states

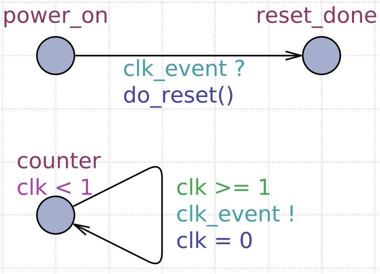


Graphical representation - edges



- *do_reset() is on next slide.

Graphical representation - synchronization




```
void do_reset() {
 ACC = 0;
 irq_mask = ~0;
 state = SFETCH;
}
```

Clocks, time constraints and synchronization notation

Clock

```
clock clk;
```

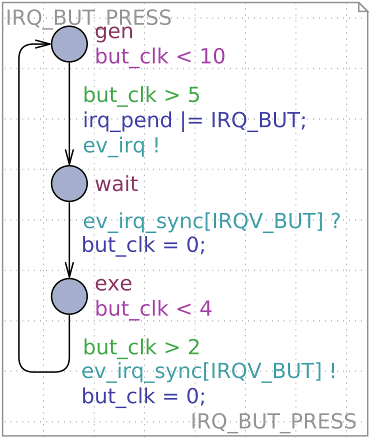
Time constraints

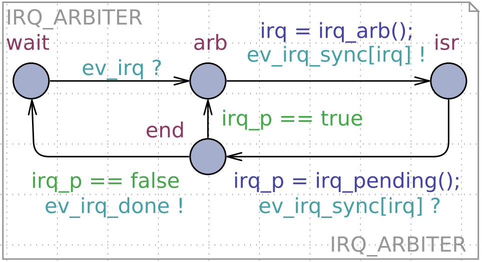
- The invariant and
- the guard.

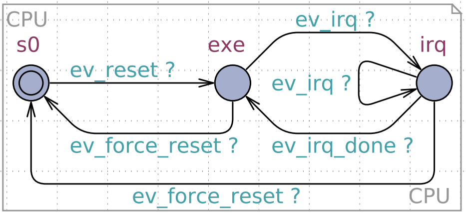
Synchronization

```
chan clk;
```

Example model







Summary

- A *TA*'s input is a timed word (σ, τ).
- *RT*-system's input events can be described by timed words.
- *RT*-system can be modelled using a *TA*.
- Behaviour of a *RT* system can be simulated and verified. See UPPAAL model checker for details.
- The models are extendable to include
  - power consumption,
  - missed time constraint condition,
  - ...

References

- UPPAAL. [cit. 09-OCT-2015].
  URL http://www.uppaal.org/
- Alur, R.; Dill, D. L.: A theory of timed automata. [cit. 16-DEC-2015].
  URL http://www.cis.upenn.edu/~alur/TCS94.pdf
- Bengtsson, J.; Yi, W.: Timed Automata: Semantics, Algorithms and Tools. [cit. 10-OCT-2015].
  URL http://www.seas.upenn.edu/~lee/09cis480/papers/by-lncs04.pdf
- Haddad, S.: Time and Timed Petri Net. [cit. 16-DEC-2015].
  URL http://www.lsv.ens-cachan.fr/~haddad/disc11-part1.pdf
- Strnadel Josef, P., Ing.: Real-time operační systémy (ROS) - Studijní opora. [cit. 16-DEC-2015].

The end - questions

Thank you for your attention. Question time!

The end - goodbye

Thanks

- Meduna Alexander, prof. RNDr., CSc. : TID and LTA organization.
- Lojda Jakub, Ing. : presentation review.
- Barošová Lenka : presentation speech review.