Processos e Concorrência 2016/17
Bloco de Slides 1

Alexandre Madeira
HASLab INESC TEC, DI UMINHO

Febrary 10, 2016

Reactive systems

Reactive system

system that computes by reacting to stimuli from its environment along its overall computation

Reactive systems

Reactive system

system that computes by reacting to stimuli from its environment along its overall computation

- in contrast to sequential systems whose meaning is defined by the results of finite computations, the behaviour of reactive systems is mainly determined by interaction of non-terminating processes evolving concurrently.
- observation $\equiv$ interaction
- behaviour $\equiv$ a structured record of interactions

Labelled Transition System

Definition

A LTS over a set N of names is a tuple $\langle S, N, \downarrow, \rightarrow \rangle$ where

- $S = \{s_0, s_1, s_2, ...\}$ is a set of states
- $\downarrow \subseteq S$ is the set of terminating or final states

$$\downarrow s \;\equiv\; s \in \downarrow$$

- $\rightarrow \subseteq S \times N \times S$ is the transition relation, often given as an N-indexed family of binary relations

$$s \xrightarrow{a} s' \;\equiv\; \langle s, a, s' \rangle \in \rightarrow$$

Labelled Transition System

Morphism

A morphism relating two LTS over N, $\langle S, N, \downarrow, \longrightarrow \rangle$ and $\langle S', N, \downarrow', \longrightarrow' \rangle$, is a function $h : S \to S'$ st

$$\begin{array}{rcl} s \xrightarrow{a} s' & \Rightarrow & h(s) \xrightarrow{a}{}' h(s') \\ s \downarrow & \Rightarrow & h(s) \downarrow' \end{array}$$

morphisms preserve transitions and termination

Labelled Transition System

System

Given a LTS $\langle S, N, \downarrow, \longrightarrow \rangle$, each state $s \in S$ determines a system over all states reachable from s and the corresponding restrictions of $\longrightarrow$ and $\downarrow$.

Reachability

Definition

The reachability relation, $\rightarrow^* \subseteq S \times N^* \times S$, is defined inductively

- $s \xrightarrow{\epsilon}{}^* s$ for each $s \in S$, where $\epsilon \in N^*$ denotes the empty word;
- if $s \xrightarrow{a} s''$ and $s'' \xrightarrow{\sigma}{}^* s'$ then $s \xrightarrow{a\sigma}{}^* s'$, for $a \in N, \sigma \in N^*$

Reachability

Definition

The reachability relation, $\rightarrow^* \subseteq S \times N^* \times S$, is defined inductively

- $s \xrightarrow{\epsilon}{}^* s$ for each $s \in S$, where $\epsilon \in N^*$ denotes the empty word;
- if $s \xrightarrow{a} s''$ and $s'' \xrightarrow{\sigma}{}^* s'$ then $s \xrightarrow{a\sigma}{}^* s'$, for $a \in N, \sigma \in N^*$

Reachable state

$t \in S$ is reachable from $s \in S$ iff there is a word $\sigma \in N^*$ st $s \xrightarrow{\sigma}{}^* t$

LTS classification

An LTS $\langle S, N, \downarrow, \longrightarrow \rangle$ is said

LTS classification

An LTS $\langle S, N, \downarrow, \longrightarrow \rangle$ is said

deterministic if for each $s \in S$, $a \in N$, there is at most an $s' \in S$ such that $s \xrightarrow{a} s'$, i.e., **if $s \xrightarrow{a} s'$ and $s \xrightarrow{a} s''$, then $s' = s''$.**

LTS classification

An LTS $\langle S, N, \downarrow, \longrightarrow \rangle$ is said

deterministic if for each $s \in S$, $a \in N$, there is at most an $s' \in S$ such that $s \xrightarrow{a} s'$, i.e., **if $s \xrightarrow{a} s'$ and $s \xrightarrow{a} s''$, then $s' = s''$.**

non deterministic if it is not deterministic

LTS classification

An LTS $\langle S, N, \downarrow, \longrightarrow \rangle$ is said

deterministic if for each $s \in S$, $a \in N$, there is at most an $s' \in S$ such that $s \xrightarrow{a} s'$, i.e., **if $s \xrightarrow{a} s'$ and $s \xrightarrow{a} s''$, then** $s' = s''$.

non deterministic if it is not deterministic

finite if $\{s \xrightarrow{a} s' | s \in S, a \in N\}$ is finite

LTS classification

An LTS $\langle S, N, \downarrow, \longrightarrow \rangle$ is said

deterministic if for each $s \in S$, $a \in N$, there is at most an $s' \in S$ such that $s \xrightarrow{a} s'$, i.e., **if $s \xrightarrow{a} s'$ and $s \xrightarrow{a} s''$, then $s' = s''$.**

non deterministic if it is not deterministic

finite if $\{s \xrightarrow{a} s' | s \in S, a \in N\}$ is finite

finitely branching if each node has only finitely many outgoing transitions, i.e., for any $s \in S$, $\{s' \xrightarrow{a} s | a \in A, s' \in S\}$ is finite

LTS classification

An LTS $\langle S, N, \downarrow, \longrightarrow \rangle$ is said

deterministic if for each $s \in S$, $a \in N$, there is at most an $s' \in S$ such that $s \xrightarrow{a} s'$, i.e., **if $s \xrightarrow{a} s'$ and $s \xrightarrow{a} s''$, then** $s' = s''$.

non deterministic if it is not deterministic

finite if $\{s \xrightarrow{a} s' | s \in S, a \in N\}$ is finite

finitely branching if each node has only finitely many outgoing transitions, i.e., for any $s \in S$, $\{s' \xrightarrow{a} s | a \in A, s' \in S\}$ is finite

image finite if, for each $a \in N$, each node has only finitely many outgoing a-transitions, i.e., for each $s \in S$, $a \in N$, $\{s' | s \xrightarrow{a} s'\}$ is finite

Back to old friends?

automaton behaviour $\equiv$ accepted language

Recall that finite automata recognize **regular** languages, i.e. generated by

- $L_1 + L_2 := L_1 \cup L_2$ (union)
- $L_1 \cdot L_2 := \{st | s \in L_1, t \in L_2\}$ (concatenation)
- $L^* := \{\epsilon\} \cup L \cup (L \cdot L) \cup (L \cdot L \cdot L) \cup \ldots$ (iteration)

Automata

There is a syntax to specify such languages:

$$E \quad ::= \quad \epsilon \mid a \mid E + E \mid E\,E \mid E^*$$

where $a \in \Sigma$.

Automata

There is a syntax to specify such languages:

$$E \quad ::= \quad \epsilon \mid a \mid E + E \mid E\,E \mid E^*$$

where $a \in \Sigma$.
and an algebra of regular expressions:

$$\begin{aligned}
(E_1 + E_2) + E_3 &= E_1 + (E_2 + E_3) \\
(E_1 + E_2)\,E_3 &= E_1\,E_3 + E_2\,E_3 \\
E_1\,(E_2\,E_1)^* &= (E_1\,E_2)^*\,E_1
\end{aligned}$$

After thoughts

... need more general models and theories:

- **Several interaction points** ($\neq$ functions)
- Need to distinguish **normal from anomolous termination** (eg deadlock)
- **Non determinisim** should be taken seriously: the notion of **equivalence** based on accepted language is **blind** wrt non determinism
- Moreover: the **reactive** characters of systems entails that not only the generated language is important, but also **the states traversed during an execution of the automata**.

The course

Aims

- To become familiar with **reactive systems**, emphasizing their **concurrent composition** and **continuous interaction with their environement**
- To introduce techniques for (formal) specification, analysis and verification of reactive systems

The course

1. Basic models for reactive systems
   (state, behaviour, interaction, concurrency)
   1. Labelled transition systems
   2. Processes and behaviour
   3. Similarity and bisimilarity
2. Process algebras
   1. CCS
   2. (Overview in Abstract Data Types specification)
   3. mCRL2
3. Logics for reactive systems
   1. Hennessy-Milner logic and its extensions
   2. Modal, hybrid and temporal logics
   3. Specification and verification of logic constraints
   4. Introduction to model-checking techniques

The course

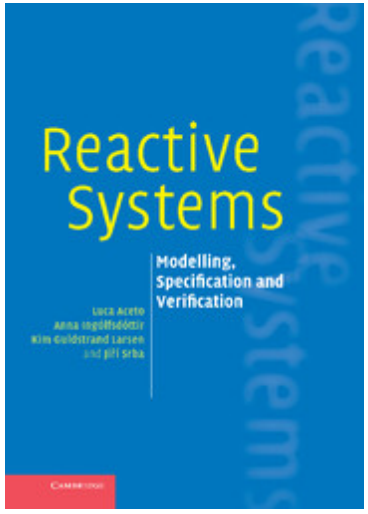
Assignment

- final test 70%
- project in mCRL2 30%

The course

bibliography

Main reference:

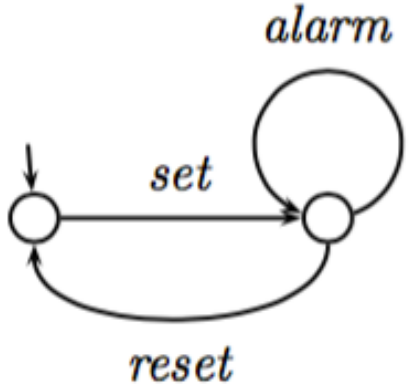


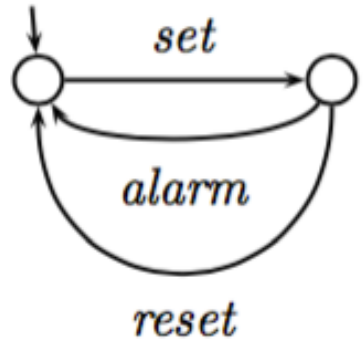
you can get a free preprint of the first at http://www.win.tue.nl/∼jfg/educ/2IW26/lente2014/mcrl2-book.pdf

Course web site:

http://alfa.di.uminho.pt/∼madeira/IntConc.html

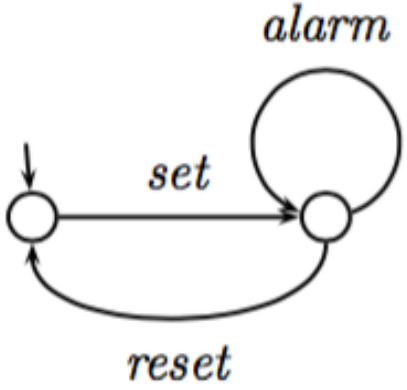
Exercise

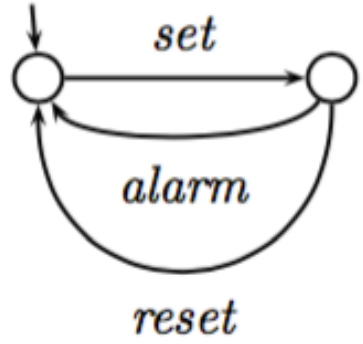




- Describe each behaviour and distinguish between the two alarm clocks.
- Describe these graphical specifications in the form of a labelled transition system conforming to the formal definition.
- Modify the previous specification to express a situation in which it is unclear how often the alarm can be repeated.

Exercise





- Draw the behaviour of an alarm clock where it is always possible to do a set or a reset action.
- Draw the behaviour of an alarm clock with unreliable buttons. When pressing the set button the alarm clock can be set, but this does not need to be the case. Similarly for the reset button. Pressing it can reset the alarm clock, but the clock can also stay in a state where an alarm is still possible.
- Draw the behaviour of an alarm clock where the alarm sounds at most three times when no other action interferes.