Labelled Transition Systems

Luís Soares Barbosa

Universidade do Minho

HASLab
HIGH-ASSURANCE
SOFTWARE LABORATORY

UNITED NATIONS
UNIVERSITY

UNU-EGOV

Interaction & Concurrency Course Unit (Lcc)

Universidade do Minho, 12.II.2018

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 and mobility of non-terminating processes, evolving concurrently.

- observation $\equiv$ interaction

- behaviour $\equiv$ a structured record of interactions

Reactive systems

Concurrency vs interaction

$$x := 0;$$
$$x := x + 1 \mid x := x + 2$$

- both statements in parallel could read x before it is written

- which values can x take?

- which is the program outcome if exclusive access to memory and atomic execution of assignments is guaranteed?

Labelled Transition System

Definition

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

- $S = \{s_0, s_1, s_2, ...\}$ is a set of states

- $\downarrow \subseteq S$ is the set of terminating or final states

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

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

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

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 $h : S \longrightarrow S'$ st

$$s \xrightarrow{a} s' \quad \Rightarrow \quad h\,s \xrightarrow{a}{}' h\,s'$$
$$s \downarrow \quad \Rightarrow \quad h\,s \downarrow'$$

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 $\downarrow$.

LTS classification

- deterministic
- non deterministic
- finite
- finitely branching
- image finite
- ...

Reachability

Definition

The reachability relation, $\longrightarrow^* \subseteq S \times N^* \times S$, 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 $s \xrightarrow{\sigma}^* t$

Automata

Back to old friends?

$$\boxed{\text{automaton behaviour} \quad \equiv \quad \text{accepted language}}$$

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

- $L_1 + L_2 \mathrel{\widehat{=}} L_1 \cup L_2$ (union)

- $L_1 \cdot L_2 \mathrel{\widehat{=}} \{st \mid s \in L_1, t \in L_2\}$ (concatenation)

- $L^* \mathrel{\widehat{=}} \{\epsilon\} \cup L \cup (L \cdot L) \cup (L \cdot L \cdot L) \cup ...$ (iteration)

Automata

There is a syntax to specify such languages:

$$E \ ::= \ \epsilon \ | \ a \ | \ E + E \ | \ E\,E \ | \ E^*$$

where $a \in \Sigma$.

- which regular expression specifies $\{a, bc\}$?
- and $\{ca, cb\}$?

and an algebra of regular expressions:

$$(E_1 + E_2) + E_3 \ = \ E_1 + (E_2 + E_3)$$
$$(E_1 + E_2)\,E_3 \ = \ E_1\,E_3 + E_2\,E_3$$
$$E_1\,(E_2\,E_1)^* \ = \ (E_1\,E_2)^*\,E_1$$

Automata

There is a syntax to specify such languages:

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

where $a \in \Sigma$.

- which regular expression specifies $\{a, bc\}$?
- and $\{ca, cb\}$?

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 ential 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, emphasising their concurrent composition and continuous interaction with their environment

- To introduce techniques for (formal) specification, analysis and verification of reactive systems

The course

Syllabus

1. Basic models for reactive systems
   (state, behaviour, interaction, concurrency)

   1.1 Labelled transition systems
   1.2 Processes and behaviour
   1.3 Similarity and bisimilarity
   1.4 Variants:
       real-time, probabilisitic and quantum transition systems

2. Process algebra

   2.1 CCS
   2.2 mCRL2

3. Logics for reactive systems

   3.1 Hennessy-Milner logic and its extensions
   3.2 Modal, hybrid and temporal logics
   3.3 Specification and verification of logic constraints

Pragmatics

Assessment

- Training assignments (2, 15% each): 23 March and 18 May
- Written test (70%): 1 June

Interaction

- web: `arca.di.uminho.pt/ic-1718/`
- contact: `lsb@di.uminho.pt`