Labelled Transition Systems

Luís Soares Barbosa

Architecture & Calculi Course Unit

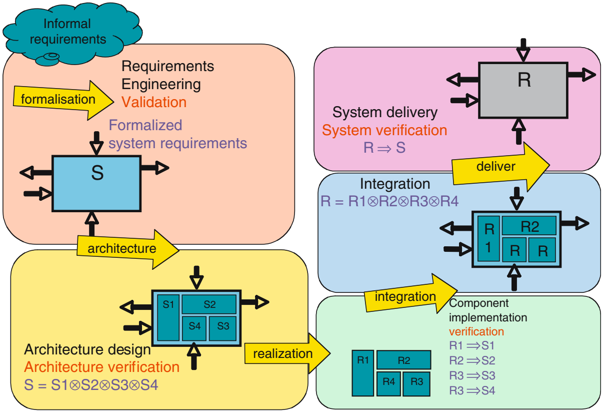
Universidade do Minho

Introduction to the Architecture & Calculi course unit

Software development as one of the most complex but at the same time most effective tasks in the engineering of innovative applications:

- Software drives innovation in many application domains
- Appropriate software provides engineering solutions that can calculate results, communicate messages, control devices, animate and reason about all kinds of information
- Actually software is becoming everyware ...

Introduction to the Architecture & Calculi course unit



Software Engineering (illustration from [Broy, 2007])

Introduction to the Architecture & Calculi course unit

So, ... yet another module in the MFES profile?

Models and analysis of reactive systems

characterised by

- a methodological shift: an architectural perspective (compositionality; interaction; focus on observable behaviour)
- a focus: on reactive systems — nondeterministic, probabilistic, timed, cyber-physical

Introduction to the Architecture & Calculi course unit

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

Labelled Transition System

Definition

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

- $S = \{s_0, s_1, s_2, ...\}$ is a set of states
- $\longrightarrow \subseteq S \times N \times S$ is the transition relation, often given as an N-indexed family of binary relations

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

In some contexts the definition is extended with a set $\downarrow \subseteq S$ of terminating or final states and a characterisitic predicate

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

Labelled Transition System

Morphism

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

$$s \xrightarrow{a} s' \quad \Rightarrow \quad h(s) \xrightarrow{a}' h(s')$$

i.e.

$$\boxed{\text{morphisms preserve transitions}}$$

... and termination, whenever applicable:

$$\downarrow \quad \Rightarrow \quad h(s) \downarrow'$$

Labelled Transition System

System

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

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 $s \xrightarrow{\sigma}^* t$

Labelled Transition System

Alternative characterization (coalgebraic)

A morphism $h : \langle S, \mathsf{next}\rangle \longrightarrow \langle S', \mathsf{next}'\rangle$ is a function $h : S \longrightarrow S'$ st the following diagram commutes

$$\begin{tikzcd} S \times N \arrow[r, "\mathsf{next}"] \arrow[d, "h \times id"'] & \mathcal{P}S \arrow[d, "\mathcal{P}h"] \\ S' \times N \arrow[r, "\mathsf{next}'"] & \mathcal{P}S' \end{tikzcd}$$

i.e.,

$$\mathcal{P}h \cdot \mathsf{next} \quad = \quad \mathsf{next}' \cdot (h \times id)$$

or, going pointwise,

$$\{h(x) \mid x \in \mathsf{next}\ \langle s, a\rangle\} \quad = \quad \mathsf{next}'\ \langle h(s), a\rangle$$

Labelled Transition System

Alternative characterization (coalgebraic)

A morphism $h : \langle S, \text{next}\rangle \longrightarrow \langle S', \text{next}'\rangle$

- preseves transitions:

$$s' \in \text{next}\ \langle s, a\rangle \Rightarrow h(s') \in \text{next}'\ \langle h(s), a\rangle$$

- reflects transitions:

$$r' \in \text{next}'\ \langle h(s), a\rangle \Rightarrow \langle \exists\ s' \in S\ :\ s' \in \text{next}\ \langle s, a\rangle :\ r' = h(s')\rangle$$

(why?)

Comparison

- Both definitions coincide at the object level:

$$\langle s, a, s' \rangle \in T \quad \equiv \quad s' \in \mathsf{next}\ \langle s, a \rangle$$

- Wrt morphisms, the relational definition is more general, corresponding, in coalgebraic terms, to

$$\mathcal{P}h \cdot \mathsf{next} \quad \subseteq \quad \mathsf{next}' \cdot (h \times id)$$

A taxonomy of simple transition systems

$\alpha : S \longrightarrow \mathcal{P}(S)$	unlabelled TS
$\alpha : S \longrightarrow \mathbb{N} \times S + \mathbf{1}$	partial LTS (generative)
$\alpha : S \longrightarrow (S + \mathbf{1})^{\mathbb{N}}$	partial LTS (reactive)
$\alpha : S \longrightarrow \mathcal{P}(\mathbb{N} \times S)$	non deterministic LTS (generative)
$\alpha : S \longrightarrow \mathcal{P}(S)^{\mathbb{N}}$	non deterministic LTS (reactive)

Recall the following notation for sets

$A \times B$ Cartesian product

$A + B$ disjoint union

B^A function space

$\mathbf{1}$ Singular set: $\mathbf{1} \cong \{*\}$

A zoo of transition systems

Simple transition systems can be extended with actions and suited to different sorts of behaviours (e.g. partial, non deterministic, etc).
... but the zoo is much broader, capturing

- probabilistic transitions (Prism)
- timed transitions (Uppaal, mCRL2)
- continuous evolutions (e.g. of physical processes) (KeYmaera)
- ... and several combinations thereof

(typical support tools are indicated in brown)

Going further: how to put order into this picture?

The taxonomy is driven by the structure on the codomain of function α which computes the next state(s), thus specifying the structure of the system's evolution or behaviour.

Going generic, the essential part of such a structure can be captured by a monad $\mathcal{B}$:

$$\alpha : S \longrightarrow \mathcal{B}(S)$$

or

$$\alpha : S \longrightarrow \mathcal{F}(\cdots \mathcal{B}(S) \cdots)$$

Monads

A monad

$$(\mathcal{B}, \mu : \mathcal{B}\mathcal{B} \Longrightarrow \mathcal{B}, \eta : Id \Longrightarrow \mathcal{B})$$

is a functor and two natural transformations such that the following diagrams commute:

$$\begin{tikzcd}
\mathcal{B}^3 \arrow[r, "\mathcal{B}\mu"] \arrow[d, "\mu_{\mathcal{B}}"'] & \mathcal{B}^2 \arrow[d, "\mu"] \\
\mathcal{B}^2 \arrow[r, "\mu"'] & \mathcal{B}
\end{tikzcd}
\qquad
\begin{tikzcd}
\mathrm{Id}_C\mathcal{B} \arrow[r, "\eta_{\mathcal{B}}"] \arrow[dr, equal] & \mathcal{B}^2 \arrow[d, "\mu"] & \mathcal{B}\mathrm{Id}_C \arrow[l, "\mathcal{B}\eta"'] \arrow[dl, equal] \\
& \mathcal{B} &
\end{tikzcd}$$

that is,

$$\mu \cdot \eta_{\mathcal{B}} = \mu \cdot \mathcal{B}\eta = id \tag{1}$$

$$\mu \cdot \mathcal{B}\mu = \mu \cdot \mu_{\mathcal{B}} \tag{2}$$

Monads

A monad in Set can be thought of as a monoid in $\mathit{Set}^{\mathit{Set}}$, the category of Set-endofunctors.

Thinking of $\mathcal{B}$ as the encapsulation of a computational structure,

- its unit η represents the minimal such structure when a value $s \in S$ is embedded in $\mathcal{B}(S)$;
- Multiplication μ flattens computations, providing a way to view a $\mathcal{B}$-effect of a $\mathcal{B}$-effect still as a $\mathcal{B}$-effect.

Monads

The powerset monad - for nondeterministic LTS

$$\left(\mathcal{P}, \bigcup, sing\right)$$

i.e.

- $\eta = sing = \lambda x \, . \, \{x\}$
- $\mu = \bigcup$

The maybe monad - for partial LTS

$$(\mathsf{Id} + \mathbf{1}, [id, \iota_1], \iota_1)$$

The discrete distribution monad

$\mathcal{D}$ maps a set S to the set of finitely supported functions $\phi : S \longrightarrow [0,1]$ such that $\sum \phi(s) = 1$, where the sum is taken over the support of ϕ, i.e.

- $\mathcal{D}(S)$ consists of weights across S which sum to 1 and for which cofinitely many weights are zero.
- $\mathcal{D}$ acts on morphisms $h : S \longrightarrow S'$ by extending them linearly:

$$\mathcal{D}(h)\left(\sum_i s_i[w_i]\right) = \sum_i h(s_i)[w_i]$$

where notation $\sum s[\phi(s)]$ is a handy way to refer to elements $\phi \in \mathcal{D}(S)$, as in e.g.

$$a\left[\frac{1}{4}\right] + b\left[\frac{1}{2}\right] + c\left[\frac{1}{4}\right]$$

Note that the support of $\mathcal{D}(h)$ is finite because the support of h, i.e. $\{s \in S \mid \phi(s) > 0\}$ is finite as well.

The discrete distribution monad

with

- The unit η assigns maximum weight to its argument:

$$\eta(s) \;=\; s[1]$$

  i.e. the Dirac distribution at point s.

- Multiplication μ transforms weights on weights on S into weights on S by averaging

$$\mu(F)(s) \;=\; \left(\sum_{\phi \in \mathrm{supp}(F)} F(\phi) \cdot \phi(s) \right)$$

Probabilistic transition systems

Markov chains

$$\boxed{\alpha : S \longrightarrow \mathcal{D}(S)}$$

A Markov chain goes from a state s to a state s' with probability p if

$$\alpha(s) = \phi \;\text{ with }\; \phi(s') = p > 0$$

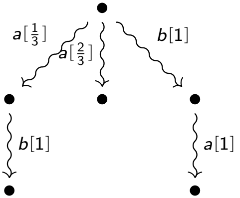
Notation

- $s \rightsquigarrow \phi$: the system evolves from s according to proabability distribution ϕ.
- $s \overset{p}{\rightsquigarrow} s'$: goes from s to s' with probability p computed as $p = (\phi(s))(s')$.

Reactive PTS

$$\boxed{\alpha : S \longrightarrow (\mathcal{D}(S) + \mathbf{1})^N}$$

- $s \overset{a}{\rightsquigarrow} \phi_a$ if $\alpha(s)(a) = \phi_a$
- $s \overset{a[p]}{\rightsquigarrow} s'$ if additionally s' in the support of ϕ_a and $\phi_a(s') = p$
- $s \not\rightsquigarrow$ if $\alpha(s)(a) = *$
- Note the role of $\mathbf{1}$ (cf $\emptyset$ in the non deterministic LTS)



Reactive PTS

$$\boxed{\alpha : S \longrightarrow (\mathcal{D}(S) + \mathbf{1})^N}$$

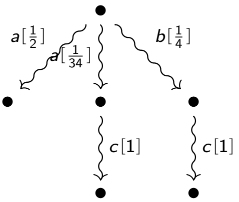
- In a reactive system probabilities are distributed over the outgoing transitions labeled with the same action.
- Actions correspond to input stimuli from the environment. On receiving a stimulus it chooses the next state probabilistically.

  There are no probabilistic assumptions over the behaviour of the environment.
- In a reactive system there is only external non-determinism

Generative PTS

$$\boxed{\alpha : S \longrightarrow \mathcal{D}(N \times S) + \mathbf{1}}$$

- $s \rightsquigarrow \phi$ if $\alpha(s) = \phi$
- $s \overset{a[p]}{\rightsquigarrow} s'$ if additionally (a, s') is in the support of ϕ and $\phi(a, s') = p$
- $s \not\rightsquigarrow$ if $\alpha(s) = *$



Generative PTS

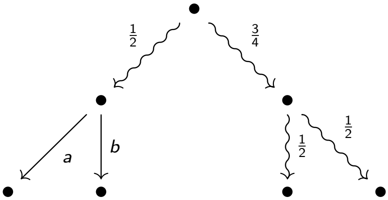
$$\boxed{\alpha : S \longrightarrow \mathcal{D}(N \times S) + \mathbf{1}}$$

- In a generative system probabilities are distributed over all outgoing transitions.
- Actions are regarded as outputs generated by the system. It chooses the next pair state and action according to the distribution probability associated to the state in the origin of the transition. The transition being chosen, the system moves to another state while generating the output action.
- No non-determinism is involved.

A taxonomy of probabilistic transition systems

$\alpha : S \longrightarrow \mathcal{D}(S)$	simple PTS (Markov chain)
$\alpha : S \longrightarrow \mathcal{D}(N \times S) + \mathbf{1}$	generative PTS
$\alpha : S \longrightarrow (\mathcal{D}(S) + \mathbf{1})^N$	reactive PTS
$\alpha : S \longrightarrow \mathcal{D}(S) + (N \times S) + \mathbf{1}$	alternating PTS

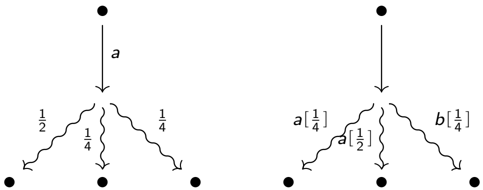
Alternating PTS



Adding non determinism

$\alpha : S \longrightarrow \mathcal{P}(N \times \mathcal{D}(S))$	simple Segala PTS
$\alpha : S \longrightarrow \mathcal{P}(\mathcal{D}(N \times S))$	strict Segala PTS
$\alpha : S \longrightarrow \mathcal{P}(\mathcal{D}(\mathcal{P}(N \times S)))$	Pnueli-Zuck PTS

Transitions for simple and strict Segala PTS



What's next?

- When are two states equivalent? Or two labelled transition systems?
- Can labelled transition systems be refined?
- Can they be combined?