System Validation

Mohammad Mousavi

2. Strong bahavioral equivalences and weak behavioral equivalences part 1.

TU Delft

Behavioral Equivalences

Mohammad Mousavi

TU/Eindhoven

System Validation, 2012-2013
TU Delft

Overview

- Organizational matters (recap)
- Motivation
- Labelled Transition Systems,
- Strong equivalences:
  1. trace equivalence,
  2. language equivalence
  3. strong bisimilarity,
  4. Exercises: 2.3.2, 2.3.9, 2.3.10

Examination(s)

Theory:

- E1 End of Quarter 1, 2-11-2012, 14:00-17:00
- E2 Resit: End of Quarter 2, 30-01-2013, 14:00-17:00

Do register using Osiris.

Practical project P (compulsory, no pass without the project)

$$M = \frac{Max(E1, E2) + P}{2}$$

Project: Procedure

- Formulate informal requirements

Project: Procedure

- Formulate informal requirements
- Define interactions with the outside world

Project: Procedure

- Formulate informal requirements
- Define interactions with the outside world
- Rephrase the requirements in terms of interactions

Project: Procedure

- Formulate informal requirements
- Define interactions with the outside world
- Rephrase the requirements in terms of interactions
- Define the system architecture and internal interactions

Project: Procedure

- Formulate informal requirements
- Define interactions with the outside world
- Rephrase the requirements in terms of interactions
- Define the system architecture and internal interactions
- Model the system behavior

Project: Procedure

- Formulate informal requirements
- Define interactions with the outside world
- Rephrase the requirements in terms of interactions
- Define the system architecture and internal interactions
- Model the system behavior
- Verify the requirements on the model

Project: Procedure

- Formulate informal requirements
- Define interactions with the outside world
- Rephrase the requirements in terms of interactions
- Define the system architecture and internal interactions
- Model the system behavior
- Verify the requirements on the model

Iterate the last two items until requirements are satisfied.

Project: Procedure

- Carried out in groups of 4; form your groups and email them to me, before September 14, 2012.

Project: Procedure

- Carried out in groups of 4; form your groups and email them to me, before September 14, 2012.
- Weekly progress meetings of 15 minutes with all group members; prepare well beforehand.

Project: Procedure

- Carried out in groups of 4; form your groups and email them to me, before September 14, 2012.
- Weekly progress meetings of 15 minutes with all group members; prepare well beforehand.
- Deadlines and deliverables:

First deliverable October 5: Report including requirements, interactions and architecture
cond deliverable October 19: Report (complete structure)
Final deliverable November 2: Report, source files for models, and reflections

Project: Short Description

- Inspired by the packet storage system, by Vanderlande Industries
- 5 controllers for elevators, conveyor belts and racks
- Several requirements: deadlock freedom, avoiding clash, maximum efficiency

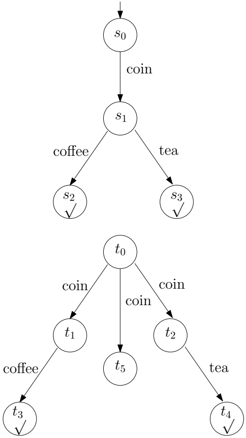
News

- The examination at the end of Q1 is moved to November 2, 2012.
- The location for weekly meetings will be LH 1.430.
- The course reader is ready to order from the printshop (order nr. 06917530021).

Behavioral Equivalences

Actions

- Atomic building blocks of models
- May denote: internal behavior or interaction with the environment
- Can be composed to obtain behavior



Motivation

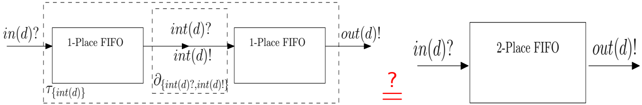
- verification: check whether an implementation conforms to the specification;
- implementation: transition system with more actions added;
- method: abstracting and comparing with spec.

Behavioral Equivalences

Motivation

- verification: check whether an implementation conforms to the specification;
- implementation: transition system with more actions added;
- method: abstracting and comparing with spec.

Example

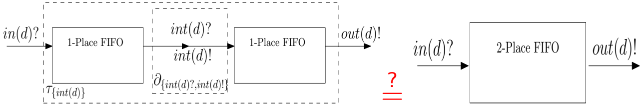


Behavioral Equivalences

Motivation

- verification: check whether an implementation conforms to the specification;
- implementation: transition system with more actions added;
- method: abstracting and comparing with spec.

Example



behavioral equivalence needed to compare behavioral models

Behavioral Equivalences

Requirements

Desired behavioral equivalence should:

- neglect immaterial differences (not too fine);
- note important differences (not too coarse);
- should be preserved under context (should be a congruence).

depends on the particular application domain.

Branching-Time Linear-Time Spectrum

There is a myriad of behavioral equivalences with different practical motivations.

Labeled Transition Systems

An LTS is a 5-tuple $\langle S, Act, \rightarrow, s, T \rangle$:

- S is a set of *states*,
- Act is a set of *(multi-)actions*,
- $\rightarrow \ \subseteq \ S \times Act \times S$ is the *transition relation*.
- $s \in S$ is the *initial* state,
- $T \subseteq S$ is the set of *terminating* states,

Labeled Transition Systems

An LTS is a 5-tuple $\langle S, Act, \rightarrow, s, T\rangle$:

- S is a set of *states*,
- Act is a set of *(multi-)actions*,
- $\rightarrow \ \subseteq \ S \times Act \times S$ is the *transition relation*.
- $s \in S$ is the *initial* state,
- $T \subseteq S$ is the set of *terminating* states,

Write $t \xrightarrow{a} t'$ for $(t, a, t') \in \ \rightarrow$.

Write $Act_{\surd}$ for $Act \cup \{\surd\}$.

Trace equivalence

Traces of a State

For state $t \in S$, Traces(t) is the minimal set satisfying:

1. $\epsilon \in$ Traces(t),

Trace equivalence

Traces of a State

For state $t \in S$, Traces(t) is the minimal set satisfying:

1. $\epsilon \in$ Traces(t),
2. $\surd \in$ Traces(t) when $t \in T$,

Trace equivalence

Traces of a State

For state $t \in S$, Traces(t) is the minimal set satisfying:

1. $\epsilon \in$ Traces(t),
2. $\surd \in$ Traces(t) when $t \in T$,
3. $\forall_{t'_0 \in S,\ a \in Act,\ \sigma \in Act_{\surd}{}^*}$ $a\sigma \in$ Traces(t) when $\exists_{t' \in S}$ $t \xrightarrow{a} t'$ and $\sigma \in$ Traces(t').

Trace equivalence

Traces of a State

For state $t \in S$, Traces(t) is the minimal set satisfying:

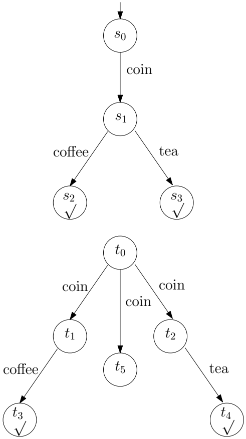
1. $\epsilon \in \text{Traces}(t)$,
2. $\surd \in \text{Traces}(t)$ when $t \in T$,
3. $\forall_{t'_0 \in S,\ a \in Act,\ \sigma \in Act_{\surd}{}^*}\ a\sigma \in \text{Traces}(t)$ when $\exists_{t' \in S}\ t \xrightarrow{a} t'$ and $\sigma \in \text{Traces}(t')$.

Trace Equivalence

For states t, t', t is trace equivalent to t' iff $\text{Traces}(t) = \text{Traces}(t')$.

1. $\epsilon \in \text{Traces}(t)$,
2. $\surd \in \text{Traces}(t)$ when $t \in T$,
3. $a\sigma \in \text{Traces}(t)$ when $t \xrightarrow{a} t'$ and $\sigma \in \text{Traces}(t')$.

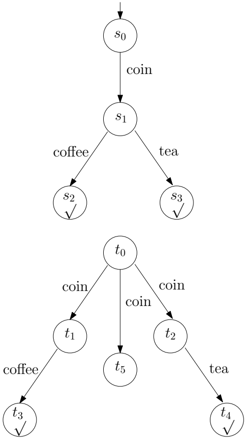
Traces: An Example



1. $\epsilon \in \mathrm{Traces}(t)$,
2. $\surd \in \mathrm{Traces}(t)$ when $t \in T$,
3. $a\sigma \in \mathrm{Traces}(t)$ when $t \xrightarrow{a} t'$ and $\sigma \in \mathrm{Traces}(t')$.

Traces: An Example

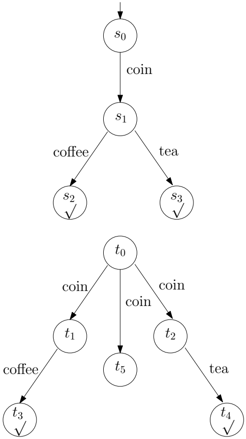
- $\mathrm{Traces}(s_2) = \mathrm{Traces}(s_3) = \mathrm{Traces}(t_3) = \mathrm{Traces}(t_4) = \{\epsilon, \surd\}$,



1. $\epsilon \in \text{Traces}(t)$,
2. $\surd \in \text{Traces}(t)$ when $t \in T$,
3. $a\sigma \in \text{Traces}(t)$ when $t \xrightarrow{a} t'$ and $\sigma \in \text{Traces}(t')$.

Traces: An Example

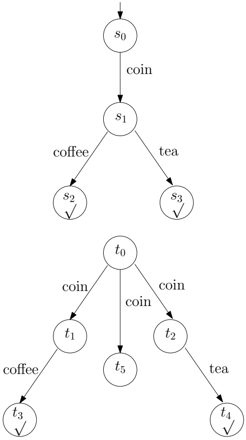
- $\text{Traces}(s_2) = \text{Traces}(s_3) = \text{Traces}(t_3) = \text{Traces}(t_4) = \{\epsilon, \surd\}$,
- $\text{Traces}(t_5) = \{\epsilon\}$,



1. $\epsilon \in \text{Traces}(t)$,
2. $\surd \in \text{Traces}(t)$ when $t \in T$,
3. $a\sigma \in \text{Traces}(t)$ when $t \xrightarrow{a} t'$ and $\sigma \in \text{Traces}(t')$.

Traces: An Example

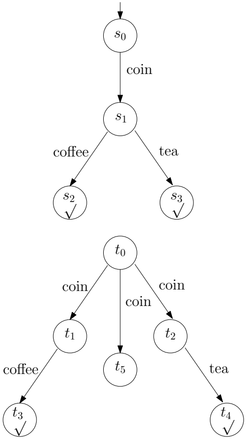
- $\text{Traces}(s_2) = \text{Traces}(s_3) = \text{Traces}(t_3) = \text{Traces}(t_4) = \{\epsilon, \surd\}$,
- $\text{Traces}(t_5) = \{\epsilon\}$,
- $\text{Traces}(s_1) = \{\epsilon, \mathit{coffee}, \mathit{tea}, \mathit{coffee}\surd, \mathit{tea}\surd\}$,



1. $\epsilon \in \text{Traces}(t)$,
2. $\surd \in \text{Traces}(t)$ when $t \in T$,
3. $a\sigma \in \text{Traces}(t)$ when $t \xrightarrow{a} t'$ and $\sigma \in \text{Traces}(t')$.

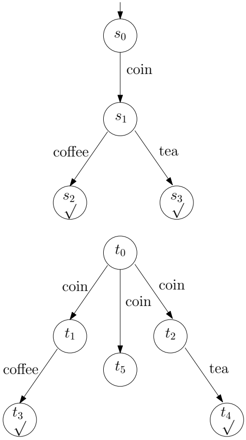
Traces: An Example

- $\text{Traces}(s_2) = \text{Traces}(s_3) = \text{Traces}(t_3) = \text{Traces}(t_4) = \{\epsilon, \surd\}$,
- $\text{Traces}(t_5) = \{\epsilon\}$,
- $\text{Traces}(s_1) = \{\epsilon, \mathit{coffee}, \mathit{tea}, \mathit{coffee}\surd, \mathit{tea}\surd\}$,
- $\text{Traces}(t_1) = \{\epsilon, \mathit{coffee}, \mathit{coffee}\surd\}$, $\text{Traces}(t_2) = \{\epsilon, \mathit{tea}, \mathit{tea}\surd\}$,



1. $\epsilon \in \text{Traces}(t)$,
2. $\surd \in \text{Traces}(t)$ when $t \in T$,
3. $a\sigma \in \text{Traces}(t)$ when $t \xrightarrow{a} t'$ and $\sigma \in \text{Traces}(t')$.

Traces: An Example



- $\text{Traces}(s_2) = \text{Traces}(s_3) = \text{Traces}(t_3) = \text{Traces}(t_4) = \{\epsilon, \surd\}$,
- $\text{Traces}(t_5) = \{\epsilon\}$,
- $\text{Traces}(s_1) = \{\epsilon, \mathit{coffee}, \mathit{tea}, \mathit{coffee}\surd, \mathit{tea}\surd\}$,
- $\text{Traces}(t_1) = \{\epsilon, \mathit{coffee}, \mathit{coffee}\surd\}$,
  $\text{Traces}(t_2) = \{\epsilon, \mathit{tea}, \mathit{tea}\surd\}$,
- $\text{Traces}(s_0) = \text{Traces}(t_0) = \{\epsilon, \mathit{coin}, \mathit{coin\ coffee}, \mathit{coin\ tea}, \mathit{coin\ coffee}\surd, \mathit{coin\ tea}\surd\}$.

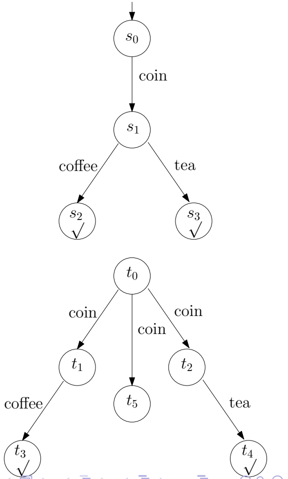
Trace Equivalence: An Observation

Observation

$\text{Traces}(s_0) = \text{Traces}(t_0) =$
$\{\epsilon, \textit{coin}, \textit{coin coffee}, \textit{coin tea}, \textit{coin coffee}\surd, \textit{coin tea}\surd\}$

Moral of the Story

Trace equivalence is usually too coarse (neglects important differences).



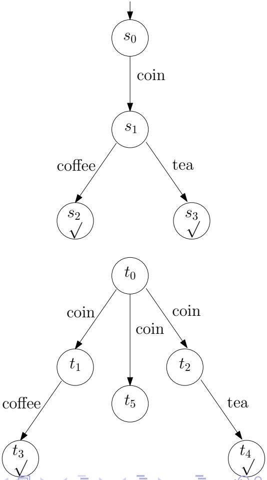
Language equivalence

Language

Lang(t):

- $\epsilon \in Lang(t)$ if $t \notin T$ and there are no $t' \in S$ and $a \in Act$ such that $t \xrightarrow{a} t'$;
- $\checkmark \in Lang(t)$ if $t \in T$; and
- if $t \xrightarrow{a} t'$ and $\sigma \in Lang(t')$ then $a\sigma \in Lang(t)$.

Two states $t, u \in S$ are *language equivalent* iff $Traces(t) = Traces(u)$ and $Lang(t) = Lang(u)$.



Bisimulation

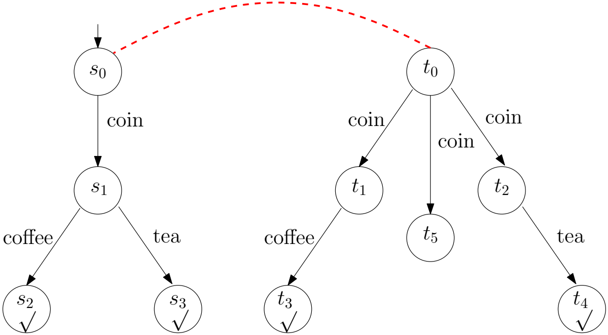
$R \subseteq S \times S$ is an (auto-)bisimulation relation when for all $\forall_{(t_0,t_1)\in R, a \in Act}$

- $\forall_{t_0' \in S}\ t_0 \xrightarrow{a} t_0' \Rightarrow \exists_{t_1' \in S}\ t_1 \xrightarrow{a} t_1' \wedge (t_0', t_1') \in R$,
- $\forall_{t_1' \in S}\ t_1 \xrightarrow{a} t_1' \Rightarrow \exists_{t_0' \in S}\ t_0 \xrightarrow{a} t_0' \wedge (t_0', t_1') \in R$, and
- $t_0 \surd \Leftrightarrow t_1 \surd$.

Bisimulation

$\forall_{(t_0,t_1)\in R}$

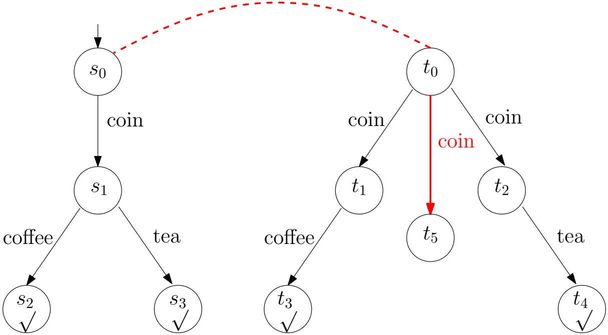
- $t_0 \xrightarrow{a} t_0' \Rightarrow \exists_{t_1'\in S}\ t_1 \xrightarrow{a} t_1' \wedge t_0'\ R\ t_1'$, and vice versa,
- $t_0\surd \Leftrightarrow t_1\surd$.



Bisimulation

$\forall_{(t_0,t_1)\in R}$

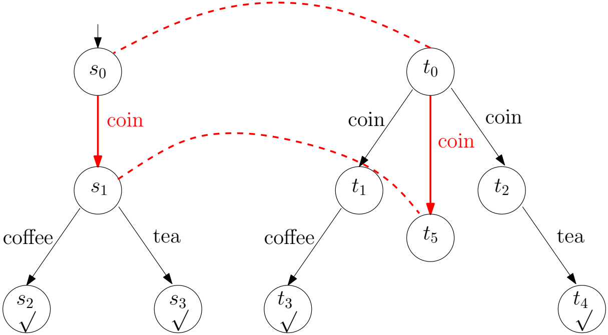
- $t_0 \xrightarrow{a} t_0' \Rightarrow \exists_{t_1' \in S}\ t_1 \xrightarrow{a} t_1' \wedge t_0'\ R\ t_1'$, and vice versa,
- $t_0\surd \Leftrightarrow t_1\surd$.



Bisimulation

$\forall_{(t_0,t_1)\in R}$

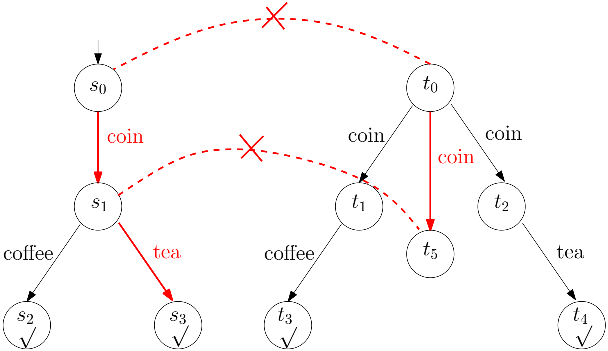
- $t_0 \xrightarrow{a} t'_0 \Rightarrow \exists_{t'_1 \in S}\ t_1 \xrightarrow{a} t'_1 \wedge t'_0\ R\ t'_1$, and vice versa,
- $t_0\surd \Leftrightarrow t_1\surd$.



Bisimulation

$\forall_{(t_0,t_1)\in R}$

- $t_0 \xrightarrow{a} t_0' \Rightarrow \exists_{t_1' \in S}\ t_1 \xrightarrow{a} t_1' \wedge t_0'\ R\ t_1'$, and vice versa,
- $t_0 \surd \Leftrightarrow t_1 \surd$.



Exercises

2.3.2

2.3.9

2.3.10