Testing of software and of communication systems

Richard Castanet,
LaBRI
Bordeaux
Overview

- Test positioning, definitions and norms
- Automatization and formal methods
- Test of reactive systems
- Optimizations (test on the fly, symbolic testing)
- Test of critical systems with temporal constraints
- Robustness testing
- Perspectives
Definition et positioning

• Test definitions: dynamic validation method, non exhaustive
• Need of the test: cost of systems and human life
• Live cycle: Position of the test: test of an implementation
• Types of test
  – Conformance testing, interoperability testing, robustness testing, performance testing
  – Test of unities, integration testing, test of the global system, test of acceptance
• Structural testing (white box) and functional testing (black box)
• Norms: ISO9646, DO178B, CEI 60880, CENELEC EN 50128 ...
• Need of automatization: test conception and test campaign
• Main questions about test: selection, coverage, testability, controllability
Use of formal methods

- Formal description language (well defined semantic)

- Scheme:

  - Informal specification
  - Formal specification
  - Automatic test generation
  - Verification
  - Simulation
  - Proof
  - Automatic code generation
Test of reactive systems

• Formal languages: SDL, LOTOS, LUSTRE, ESTEREL ...

• Based model: transition systems: Labeled Transition Systems (LTS), IOLTS (semantic for non deterministic reactive systems) (Tretmans, Jéron), automata ...

• Formal approach for the test: conformance relation: conf, ioco (traces and suspension inclusion)
Automatic test sequence generation

- Types of test:
  Conformance/Interoperability/Performance/Robustness/

- Conformance testing:
  Implantation conform to a specification

- Interoperability:
  Capacity several communicating system to interoperate

- Test Process:
Conformance relation: example

- After a visible behavior of the specification, the implementation is only authorized for the production of specified outputs or locking.
Test case and tester

- Principles of a tester for communication with the implementation: inversion of inputs - outputs
Test Architectures: interoperability testing

- Observation Point (OP) and Control (COP)
- 2 types of test: black box/ grey box
- Architectures for interoperability testing:
Automatic Test Generation

• Methods based on automata
  – Paths in graphs
  – Eulerian circuits
  – Fault model, mutant method
  – TT, W, UIO methods
  – Optimization by chinese postman algorithm
  – Executability problem
Generation Methods based on verification and simulation

• Construction of the reachability graph (all behaviors)

• Test sequence: path in the reachability graph
  → combinatorial explosion
  → enumerative methods, interlacing ...
  → need of reduction methods (test number, test sequences length, ...)

40ième anniversaire du LAAS
Test generation on the fly

• A method to reduce the combinatorial explosion

• Synchronous product between a test purpose and the specification

• Notion of observer

• Production of a part of all behaviors

• Optimizations with determinization of the graph
Suspension automaton

The absence of visible behaviors is modeled by an output event $!\delta$

$\Delta(M) = M + \text{loops of } !\delta$ on each quiescent states

Suspended traces of $M$: $\text{STraces}(M) = \text{Traces}(\Delta(M))$

det$(\Delta(M))$ characterizes the visible behaviors of $M$. 
An example of a specific tester related to a test purpose

- **Specification**: $S$
- **Conformance Relation**: $ioco$
- **Property**: Test purpose TP

Reachability of $\text{Det}(\Delta(S))$: suspended trace of $S$ accepted by TP
Symbolic testing

• Method used with test purpose
• Each state is linked a descriptor giving all the constraints on the variables
• From a state to the next one, the constraints are modified by the actions of the state
• Use of a constraint solver
Symbolic Reachability

Back

observation

(event or delay)

current estimate

runs matching observation

next estimate

40ième anniversaire du LAAS
Test of critical temporal systems

• Need of test for critical system
• Model for critical temporal system: temporized automata
Temporized automaton

Temporized automaton: 
\((S,L,C,S_0,T)\) with:
- \(S_0\) \(S\) : initial states,
- \(T\) \(S \times S \times L \times 2^C \times \Phi(C)\).

40ième anniversaire du LAAS
Execution of a temporized automaton

(S0, x=0 y=0) ? a \( t=0.5 \)
(S1, x=0.5 y=0) ! c \( t=0.75 \)
(S3, x=0.75 y=0.25) ? a \( t=0.85 \)
(S1, x=0.85 y=0)

! b \( t=1.85 \)
(S2, x=1.85 y=1)

? b \( t=6.85 \)
(S0, 0=x=y)

? a \( t=7.85 \)
(S1, x=1 y=0)

40ième anniversaire du LAAS
Automatic test generation

- Discrete time or continuous time
- Infinite states
- Method based on the region graph (finite graph and exponential complexity)
- Test on the fly
- Timed constraints resolution
Let \( C = \{x, y\} \) and \( \Phi(C) \) a set of constraints on \( C \) with \( C_x = 3 \) and \( C_y = 1 \).

**An example for a region graph:**

Temporal successor of a region:
The temporal successors of \( r_1 \): \( r_2, r_6, r_7 \).
Region automaton

Let \((S,L, \mathcal{C}, S_0, T)\) a temporized automaton)

State representation: \(<\text{state}, \text{region}>>\).

Transition representation: \(<<e_1, r_1>, a, <e_2, r_2>>\) with:
- \(<e_1, a, c, r, e_2> \in T\),
- \(r_2\) is a temporal successor of \(r_1\),
- \(r_2\) satisfies \(c\).

**Example with** \(\mathcal{C} = \{x, y\}\), \(C_x = 1\) et \(C_y = 1\).

- **Example with** \(\mathcal{C} = \{x, y\}\), \(C_x = 1\) et \(C_y = 1\).
(S0, x = y = 0) → ? a [δ = 0.5] → (S1, 0 ≤ y < x < 1) → ! c [δ = 0.25] → (S3, 0 < y < x < 1) → ? a [δ = 0.1] → (S1, 0 ≤ y < x < 1)  

! b [δ = 1] → (S2, 1 ≤ y < x) → ? b [δ = 5] → (S0, 0 = x = y) → ? a [δ = 1] → (S1, x = 1, 0 = y)  

40ième anniversaire du LAAS
40ième anniversaire du LAAS
Test on the fly
Approaches with test purpose

• Main idea for test with test purpose
  – Let:
    • S specification of the system to be tested,
    • O test purpose
  – Problem:
    • Find an execution of S drived by the test purpose.

• Model: temporized automaton

• Several approaches (conformance testing)
  – Region graph
  – Test purpose
  – Proof assistant
Region graph approach

• Extraction of a test sequence
  – Producing a trace driven by a test purpose on the region automaton
    1. Sequence of transitions of the region automaton (non temporized)
    2. Choice of $\delta$ : fire instants

• disadvantage: combinatory explosion
  – Abstraction of the specification (temporized automaton)
  – Equivalence of states
  – Minimization of the region graph (partition of the state space)
Method with test purpose

**Definition:** A Test purpose is an deterministic automaton without cycle automaton and with an non empty set of special states: Accept(TP).

**Goal:** Find a sequence of transitions of the specification according to the test purpose.

**Verdicts**
- (Pass): the event satisfies the Spec and the TP
- Pass: The event satisfies the Spec and the TP and TP is in an acceptance state
- Fail: the event does not verify the Spec
- Inc: the event satisfies the Spec, but not the TP.

Example of a coffee machine:

![Coffee machine diagram](image)
Synchronized product

Synchronized product on the events:

Sync : product automaton :

Rule 1:  
\[(s_1, s_3) \in S(Sync) \land (s_1, \mu, C_{t1}, C_{v1}, \rho_1, \beta_1, s_2) \in T(spec) \land (s_3, \mu, C_{t2}, C_{v2}, \rho_2, \beta_2, s_4) \in T(Ot) \land (s_2, s_3) \in S(Sync) \land ((s_1, s_3), \mu, C_{t1}, C_{v1}, \rho_1, \beta_1, (s_2, s_3)) \in T(Sync)\]

Spec:

\[\begin{array}{ccc}
1 & \rightarrow & 2 \\
A & R(t) & B \\
& t[2,5] & \\
\end{array} \quad \begin{array}{ccc}
2 & \rightarrow & 3 \\
B & C^{t[3,6]} & R(t1) \\
\end{array} \quad \begin{array}{ccc}
3 & \rightarrow & 4 \\
C^{t[3,6]} & & \\
\end{array} \quad \begin{array}{ccc}
1,5 & \rightarrow & 2,5 \\
A & R(t) & B \\
& t[2,5] & \\
\end{array} \quad \begin{array}{ccc}
2,5 & \rightarrow & 3,5 \\
B & R(t2) & \\
\end{array}\]

Product automaton
Synchronized product:

Rule 2:

\[(s_1, s_3) \in S(Sync) \land (s_1, \mu, C_{t1}, C_{v1}, \rho_1, \beta_1, s_2) \in T(spec) \land (s_3, \mu, C_{t2}, C_{v2}, \rho_2, \beta_2, s_4) \in T(Ot) \land (s_2, s_4) \in S(Sync) \land (s_1, s_3, \mu, C_{t1} \cup C_{t2}, C_{v1} \cup C_{v2}, \rho_1 \cup \rho_2, \beta_1 \cup \beta_2, s_2, s_4) \in T(Sync) \land (s_2, s_3) \in S(Sync) \land (s_1, s_3, \mu_1, C_{t1}, C_{v1}, \rho_1, \beta_1, s_2, s_3) \in T(Sync)\]

Spec:

TP:
Synchronized product

Temporal Synchronisation:

(method similar than the computation of the state classes of the temporized Petri nets)

Main idea: inequations system keeping the temporal relationship between the different clocks.

a transition is aware once all the clocks in his temporal constraint have been reset.
Synchronized product

When a transition is fired, the system of inequalities is updated in 3 stages:

1) Calculating the time remaining to make transitions sensitized

2) Remove unnecessary relations.

3) Taking into account the new transitions sensitized by resetting clocks.
Robustness testing

Robustness definitions:
IEEE: degree from one system to function properly in the presence of invalid entries or stressful environment
ability to exhibit acceptable behavior in the presence of hazards

<table>
<thead>
<tr>
<th>hazard</th>
<th>• Fault</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>– extern/intern</td>
</tr>
<tr>
<td></td>
<td>– Accidental / intentional</td>
</tr>
<tr>
<td>correct or acceptable</td>
<td>Change use profile and charge</td>
</tr>
</tbody>
</table>

• robustness stronger than conformity or orthogonal?
• sometimes, no specification of expected behavior over hazard

Classification of the hazards
• Internal, external, out of the system
• Representable or no representable
Methods based on behavior models

P : S specification, M fault model (set of potential faults and unanticipated events planned), P robustness property : an implantation I is robust iff I met P even in the presence of faults
Approach « increasing the specification » and refusal graph

- Specification S: LTS, extension with unknown or incorrect event (increase the specification)
- Construction of the refusal graph (set of refuse in each state)
- Adding fault traces (sequence of actions alternating with set of refusal)
- Construction of the robustness tester: refusal graph + fault traces
- Adding of unobservable action to go back to the initial state
- Test selection and coverage computation

40ième anniversaire du LAAS
Example

Construction of the refusal graph

Fault traces: alternating sequences of actions and set of refusal

Construction of the robustness tester
Approach based on a model on the entries

- Operational profile
- Equivalence classes
- ...

Model for the nominal behavior + model of the hazards

Overlay nominal activity nominal x hazards ?

- Problem of insignificant experiences
  Ex: injection of a fault that will not be activated in the system analysis : Online system activity, gray box analysis

- Selecting relevant case in an objective verification (= evaluation)
  Ex: heuristic optimization to guide the research of test the most "dangerous" case (≠ the most representative)
Perspectives

• Improvement of the use of formal methods in industries
• Treatment of real, complex systems
• Best selection of the tests
• Best coverage of a test suite
• Combining several methods: verification, proof, ...
An example for conformance testing

OT

0 [True]

1 [True]

2 [True]

Sender

Init [True]

Wait [True]

Finish [True]

Transmit [(S-x)<Lambda]

Retry [(S-x)<2*Sig]

S-x=2*Sig, ?Busy, x=S

S-x<=2*Sig, ?CD, x=S

S-t1, !Begin

S=$t1, !Begin, -

S=$t1+Lambda, !Begin

S=t1+Lambda, !Begin, -

S=t1+Lambda, !End

Paramètres:
Lambda=808
Sig=26

40ième anniversaire du LAAS
Calcul des intervalles de l’horloge $h(2)$

Exemple:

$h_t[0,2]$
$h_t[0,7]$
$h_t[5,13]$
$h_t[3,13]$

$h_t[0,2]$ de $h_t[0,7]$ de $h_t[5,13]$ de $h_t[3,13]$

$h_t[0,2] [0,2] [0,1] [0,1] [0,1]$
$h_t[0,7] [2,7] [2,6] [2,7] [2,6]$
$h_t[5,13] [2+2,6+6] [3,13] [4,12] [5,13]$
$h_t[3,13] [0+3,1+6] [2+3,6+7] [5,13] [5,7]$

$h_t(t_0) = [0,0]$
$h_t(t_1) = [0,2] [0,2] [0,1] [0,1]$
$h_t(t_2) = [0+2,1+5] [2,7] [2,6]$
$h_t(t_3) = [2+2,6+6] [3,13] [4,12]$
$h_t(t_4) = [0+3,1+6] [2+3,6+7] [5,13] [5,7]$

$h_t(t_1) = [0,1] [2,6] [4,12] [5,7]$
Exemple

Pour atteindre \( t_1 \)
Horloge \( C1 \): \( 0 \)  \( \tau_1 \)  \( 2 \)
Horloge \( C2 \): \( 0 \)  \( \tau_1 \)  \( 1 \)
Horloge \( h \): \( 0 \)  \( \tau_1 \)  \( 1 \)
\( RC(t_1,h) = [0,1] \)

\[
\left\langle \begin{array}{c}
h[0,1] \\
C1[0,1] \\
C2[0,1]
\end{array} \right\rangle
\]

Pour atteindre \( t_2 \)
Transition \( t_2 \)
Horloge \( C1 \): \( 0 \)  \( \tau_1 \)  \( 1 \)
Horloge \( C2 \): \( 0 \)  \( \tau_1 \)  \( 1 \)
Horloge \( h \): \( 0 \)  \( \tau_1 \)  \( 1 \)
\( RC(t_2,h) = [0,1] \)

\[
\left\langle \begin{array}{c}
h[2,6] \\
C1[2,5] \\
C2[2,6]
\end{array} \right\rangle
\]

Transition \( t_3 \)
Horloge \( C1 \): \( 0 \)  \( \tau_1 \)  \( 2 \)
Horloge \( C2 \): \( 0 \)  \( \tau_1 \)  \( 6 \)
Horloge \( h \): \( 0 \)  \( \tau_1 \)  \( 2 \)
\( RC(t_3,h) = [4,12] \)

\[
\left\langle \begin{array}{c}
h[5,13] \\
C1[3,6] \\
C2[3,7]
\end{array} \right\rangle
\]
Pour atteindre $t_4$

Transition $t_4$ :
- Horloge $h$: 5 $\tau_4$ 7
- Horloge $C1$: 3 $\tau_4 - \tau_1$ 6
- Horloge $C2$: 3 $\tau_4 - \tau_2$ 7

Transition $t_3$ :
- Horloge $h$: 4 $\tau_3$ 12
- Horloge $C2$: 2 $\tau_3 - \tau_2$ 6

Transition $t_2$ :
- Horloge $h$: 2 $\tau_2$ 6
- Horloge $C1$: 2 $\tau_2 - \tau_1$ 5

Transition $t_1$ :
- Horloge $h$: 0 $\tau_1$ 1
- Horloge $C1$: 0 $\tau_1$ 2
- Horloge $C2$: 0 $\tau_1$ 1
- $\tau_1$ $\tau_2$ $\tau_3$ $\tau_4$

$RC(t_1,h) = [0,1]$
$RC(t_2,h) = [2,4]$
$RC(t_3,h) = [4,7]$
$RC(t_4,h) = [5,7]$

Domaine de tir potentiels
- $\langle s1, C1[0,1], C2[0,1] \rangle$

Domaine de tir
- $\langle s1, C1[0,1] \rangle$
- $\langle s2, C1[2,4] \rangle$
- $\langle s2, C1[2,3] \rangle$
- $\langle s3, C1[2,5] \rangle$
- $\langle s3, C1[2,4] \rangle$
- $\langle s4, C1[4,6] \rangle$
- $\langle s4, C1[5,6] \rangle$
- $\langle s4, C2[3,5] \rangle$
- $\langle s4, C2[3,4] \rangle$