Conformance testing for input/output symbolic transition systems

Pascale LE GALL

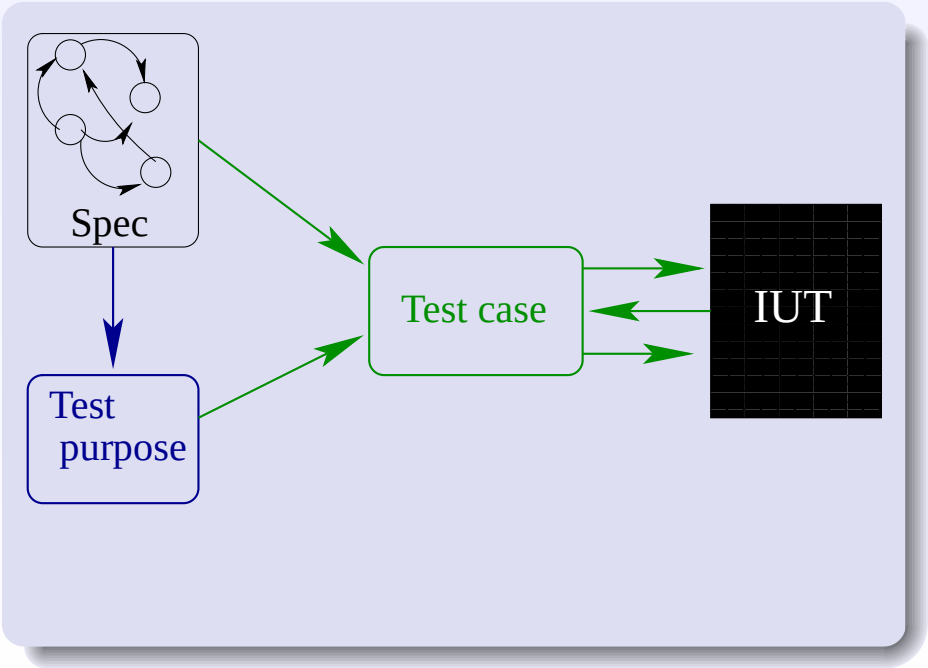
IBISC FRE 2873 - Université d'Évry Val d'Essonne

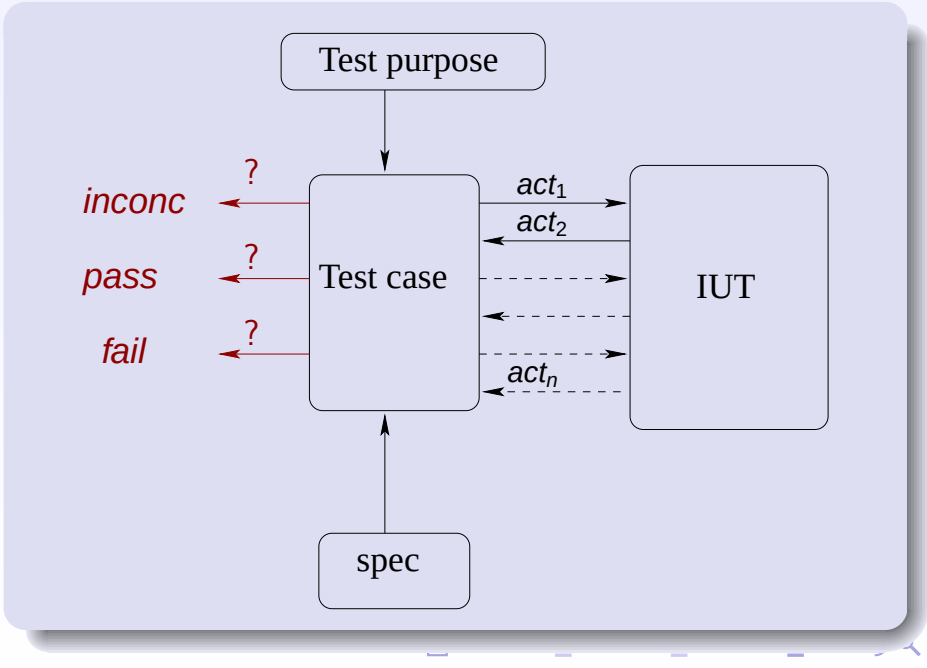
IFIP -June 2006

Conformance testing for reactive systems : “model-based testing”

- **Spec :** automata based description of expected behaviours
- **IUT :** “black-box view” of the implementation under test
- **Test purpose :** selected behaviours from Spec to be the test target (expertise)
- **Test case :** verdict computation by analysis of traces

$$t = act_1\ act_2\ \ldots act_n \text{ observed on IUT}$$





General process

Hypotheses

- Spec defines a set of models by means of a conformance relation *conf*

$$MOD(Spec) = \{i \in MOD \mid i \ conf \ Spec\}$$

- IUT is a “black-box” which can be modelled by a model i_{IUT}

$$\forall IUT, \exists i_{IUT} \in MOD, \forall t \in Traces, t \in Tr(IUT) \Longleftrightarrow t \in Tr(i_{IUT})$$

Goal of a testing method

- recognise i_{IUT} among all the possible models in MOD
- Correctness (non bias) and completeness (exhaustivity)

Symbolic transition systems

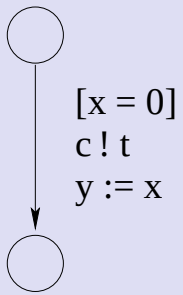
Input/Ouptut symbolic system transitions :

- Finite representation of large or infinite state-based systems
- Data are exchanged in input/output messages

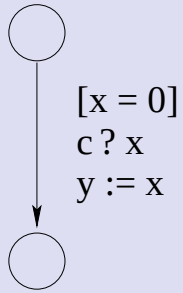
Transitions

- Guards on attribute variables
- emission/reception of data on a channel
- state modification

Emission of a message denoted by a term



Reception of a message on an attribute variable



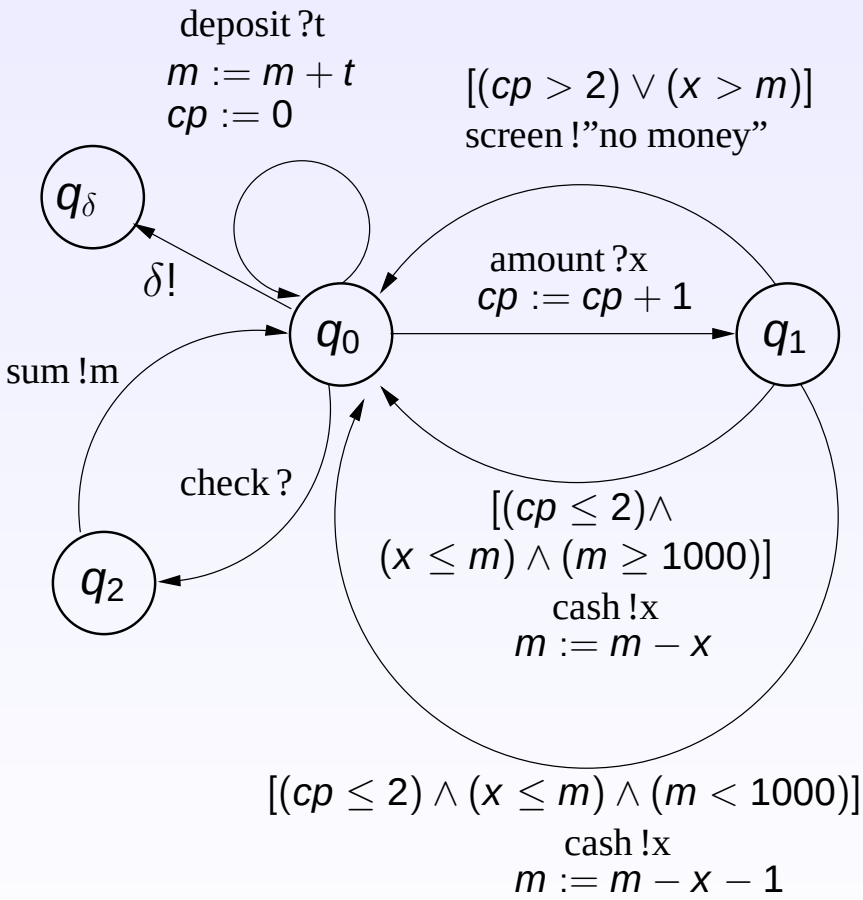
Input/Output Symbolic Transition Systems : IOSTS

$G = (Q, q_0, Trans)$

- Q set of states
- q_0 initial state
- $Trans$ set of transitions

Enrichment by quiescence

- source state q
- action : emission $\delta!$
- guard : $\wedge_e (\neg guard_e)$ for all emissions e of source q
- substitution id
- target state : state q_δ



Our goal

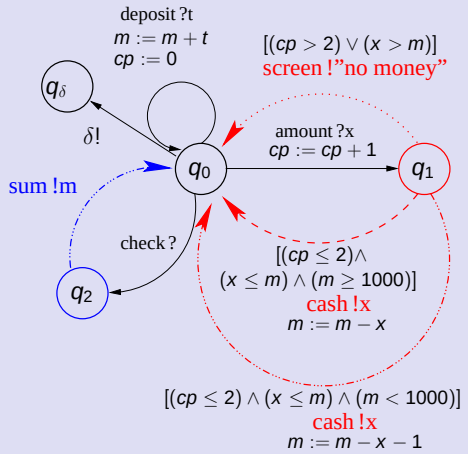
- defining appropriate test puposes Obj
- proposing selection criteria for test purposes
- Algorithm for selecting data in order to stimulate IUT with respect to Obj and emitting a verdict.
- Which verdicts ?
  1. **PASS** IUT satisfies Obj
  2. **FAIL** IUT does not conform to Spec
  3. **INCONC** IUT does not satisfy to Obj but conforms to Spec
  4. **WeakPASS** IUT not only satisfies Obj but also satisfies some other behaviours of Spec
- Non determinism in Spec $\Rightarrow$ verdicts INCONC, WeakPASS

Our approach : using Symbolic execution technique

Symbolic execution tree = description of all expected behaviours
a symbolic path = *an intensional class of similar concrete behaviours*

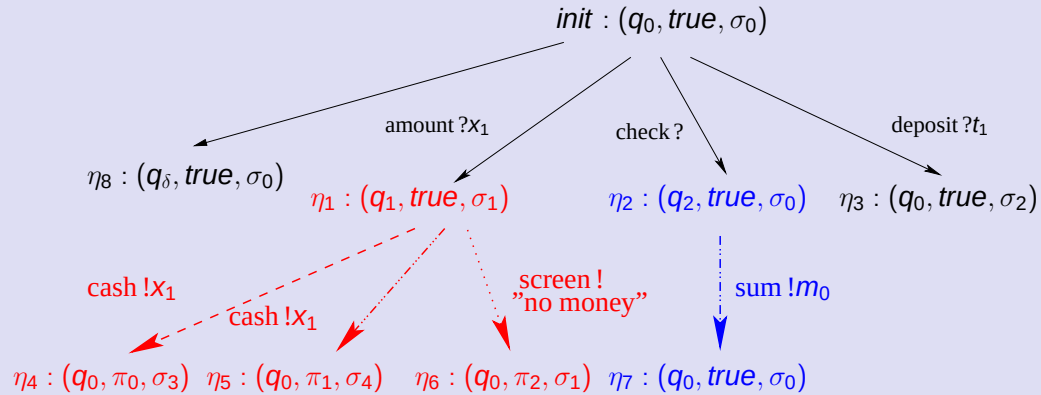
Symbolic execution

IOSTS



Symbolic state (q, π, σ)

- q state
- π path predicate
- σ assignment of attribute variables by using symbolic variables



$$\sigma_3 = x \rightarrow x_1, m \rightarrow m_0 - x_1, t \rightarrow t_0, cp \rightarrow cp_0 + 1$$
$$\sigma_4 = x \rightarrow x_1, m \rightarrow m_0 - x_1 - 1, t \rightarrow t_0, cp \rightarrow cp_0 + 1$$
$$\sigma_1 = x \rightarrow x_1, m \rightarrow m_0, t \rightarrow t_0, cp \rightarrow cp_0 + 1$$

$$\pi_0 = (cp_0 + 1 \leq 2) \wedge (x_1 \leq m_0) \wedge (m_0 \geq 1000)$$
$$\pi_1 = (cp_0 + 1 \leq 2) \wedge (x_1 \leq m_0) \wedge (m_0 < 1000)$$
$$\pi_2 = (cp_0 + 1 > 2) \vee (x_1 > m_0)$$

Conformance testing

IUT ioco Spec

$\forall tr \in Trace(Spec) \cap Trace(IUT)$ if it exists $c!v$ such that $tr.c!v \in Tr(IUT)$, then $tr.c!v \in Tr(Spec)$

"input enable" hypothesis

IUT accepts all inputs in all states

Test purpose

Finite subtree of the symbolic execution tree

- leaves labelled by the key word *accept*
- intermediate nodes labelled by the key word *skip*

Observable traces between the test case and IUT

- Messages sent to IUT : $stim(c?t)$ [stimulations/controllability]
- Messages sent by IUT : $obs(c!t)$ [observations]

Test case

Goal of test execution :

- depending on (the beginning of) the trace, computing the possible current states of the IUT according Spec
- computing concrete stimulations in order to aim an accept state (given by the test purpose) or to produce a verdict

Description of a possible current state = a context as (η, φ)

- η symbolic state
- φ predicate on symbolic variables (of symbolic execution tree) taking into account the first elements of the testing trace between the test case and IUT

Principle of the algorithm

SC : set of current contexts

- Initial set of contexts : $SC_0 = \{(q_0, true)\}$
- Steps of the algorithm expressed by rules of the form :

$$\frac{SC}{Result}\ cond$$

with :

- *cond* set of conditions including $stim(act)$ or $obs(act)$

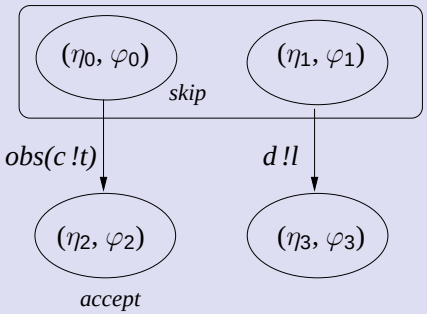
  Notation : $SC \xrightarrow{act} Result$
- *Result* of the form
  - a new current set of contexts
  - or a verdict among PASS, FAIL, WeakPASS and INCONC

Testing trace : $[act_1, act_2, \ldots, act_n \mid Verdict]$

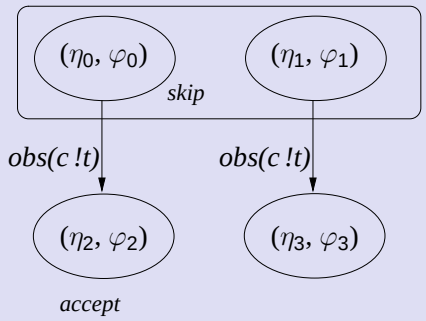
$$SC_0 \xrightarrow{act_1} SC_1 \xrightarrow{act_2} SC_2 \ldots SC_{n_1} \xrightarrow{act_n} Verdict$$

Verdicts

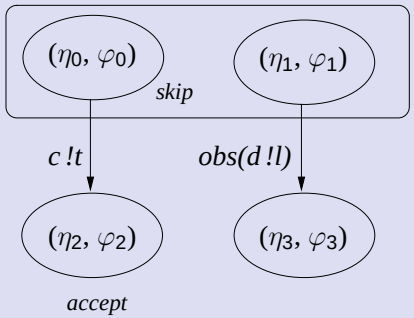
PASS



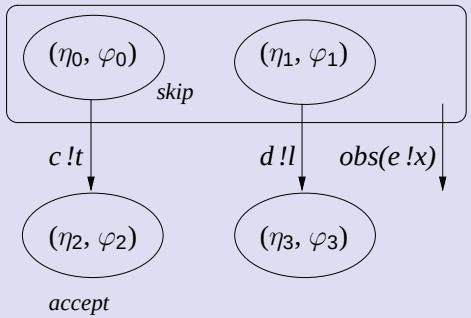
WeakPASS



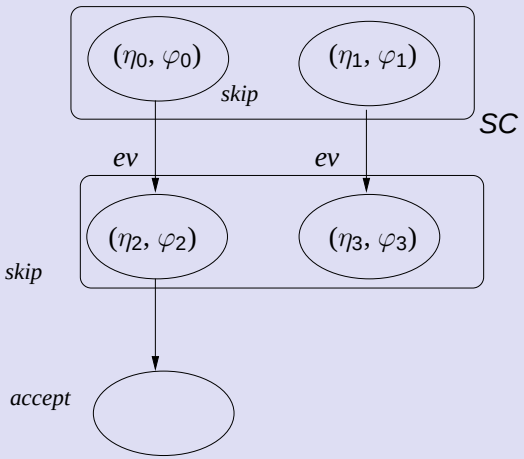
INCONC



FAIL



Auxiliary functions



$Next(ev, SC) = \{(\eta_2, \varphi_2), (\eta_3, \varphi_3)\}$

$NextSkip(ev, SC) = \{(\eta_2, \varphi_2)\}$

$NextPass(ev, SC) = \emptyset$

With $\triangle \in \{?, !\}$, if

- $\eta_0 = (q, \pi, \sigma)$
- $\eta_2 = (q', \pi', \sigma')$
- $(\eta_0, c \triangle u, \eta_2)$ transition of symbolic execution tree
- ev is $c \triangle t$ (resp. $c \triangle$)

then

- φ_2 is $\varphi_0 \wedge (t = u)$ (resp. φ_0)
- and $\varphi_2 \wedge targetCond(\eta_2)$ is satisfiable where $targetCond(\eta_2)$ is

$$\bigvee_{\pi} \pi$$

π path condition
of an *accept* state
reachable from η_2

Rules for the observations (ev of the form $c!t$ or $c!$)

Rule 1 : Emission compatible with Obj but no *accept* is reached.

$$\frac{SC}{Next(ev, SC)} \; obs(ev), NextSkip\ (ev, SC) \neq \emptyset, NextPass(ev, SC) = \emptyset$$

Rule 2 : Emission not expected with respect to Spec

$$\frac{SC}{FAIL} \; obs(ev), Next(ev, SC) = \emptyset$$

Rule 3 : Emission not compatible with Obj, but compatible with Spec

$$\frac{SC}{INCONC} \; obs(ev), Next(ev, SC) \neq \emptyset, NextSkip(ev, SC) = \emptyset$$

Rule 4 : All next contexts are *accept* ones.

$$\frac{SC}{PASS} \; obs(ev), Next(ev, SC) = NextPass(ev, SC), Next(ev, SC) \neq \emptyset$$

Rule 5 : Some next contexts are *accept* ones, but not all of them

$$\frac{SC}{WeakPASS} \; obs(ev), NextPass(ev, SC) \neq \emptyset, NextPass(ev, SC) \subsetneq Next(ev, SC)$$

Other rules

Rule 6 : Stimulation of the *SUT* (*ev* of the form $c?t$ or $c?$)

$$\frac{SC}{Next(ev, SC)} \quad stim(ev), NextSkip\ (ev, SC) \neq \emptyset$$

Initialisation rule (**Rule 0**)

$$\frac{}{\{(init, true)\}}$$

Illustration :*WeakPASS* verdict

Trace *amount*?25 *cash*!25

$$SC_0 = \{(init, true)\} \xrightarrow{amount?25} SC_1 = \{(\eta_1, (x_1 = 25))\} \xrightarrow{cash!25} WeakPASS$$

Current set of contexts after *cash*!25 contains two contexts, but only one has a state lallebed by *accept*

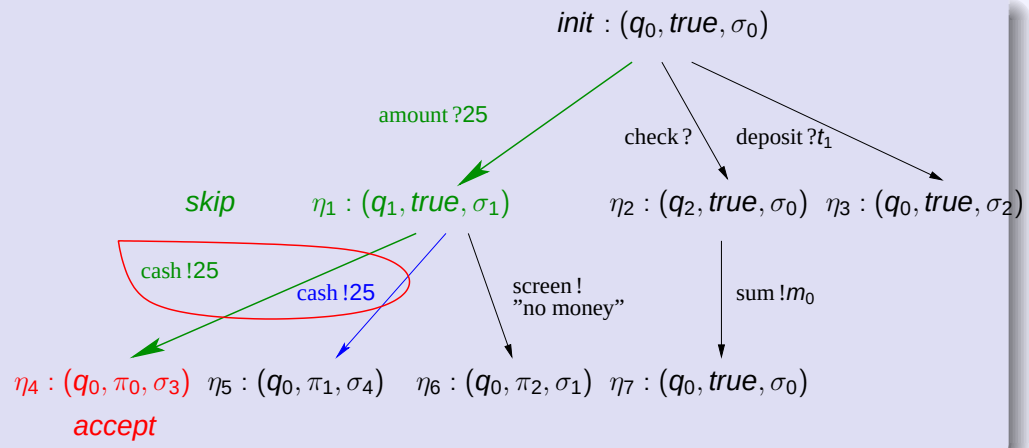


Illustration : INCONC verdict

Trace *amount*?25 *screen*!''*nomoney*''

$$SC_0 = \{(init, true)\} \xrightarrow{amount?25} SC_1 = \{(\eta_1, (x_1 = 25))\} \xrightarrow{screen!''nomoney''} INCONC$$

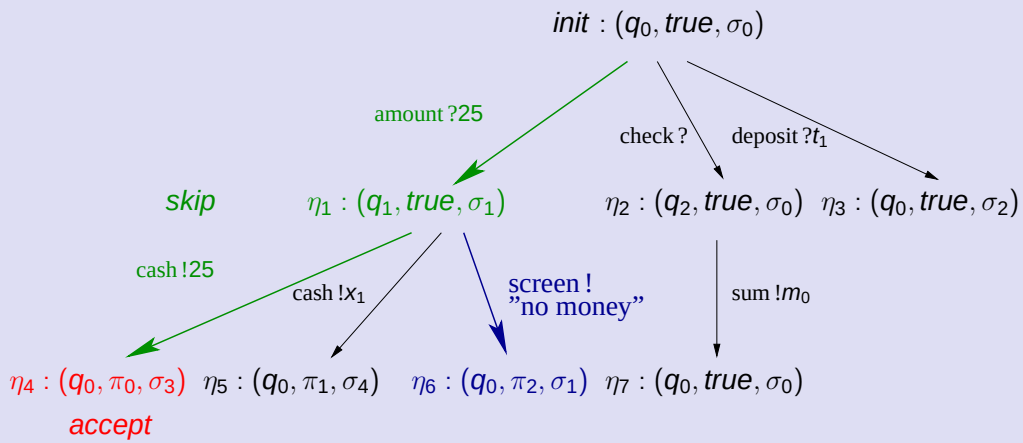
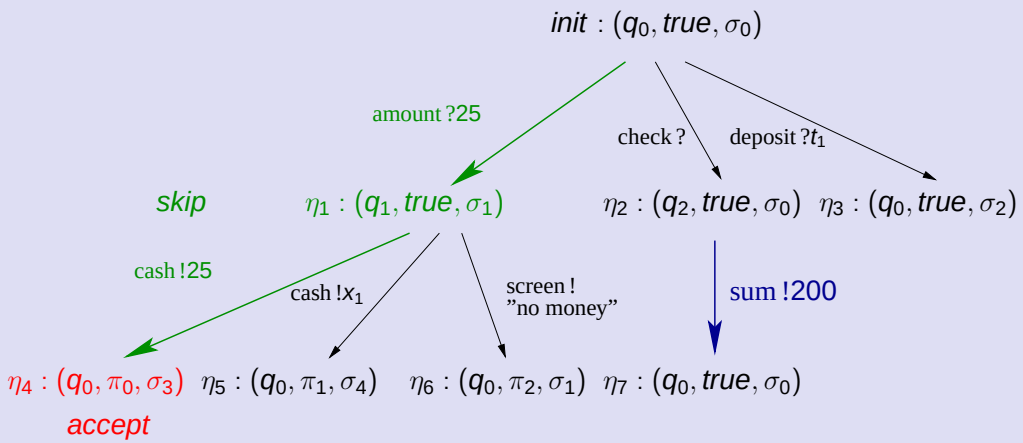


Illustration : FAIL verdict

Trace *amount*?25 *sum*!200

$$SC_0 = \{(init, true)\} \xrightarrow{amount?25} SC_1 = \{(\eta_1, (x_1 = 25))\} \xrightarrow{sum!200} FAIL$$



Results

Correctness and completeness

Inclusion criterion

- End of the computation of the symbolic execution tree when a redondancy is detected :
  all concrete value s of the new encountered symbolic state are included in the one of a previous computed one
- Good candidate for the selection of test purposes

Implementation within the AGATHA tool (CEA)

- Need a strategy to apply rules
- Decision procedure (inclusion criterion) and constraint solving (stimulation computation) in the Presburger arithmetic (Omega Library)

Extension with structuration

Design

- Reactive systems described by synchroneous composition of IOSTS

$$Spec = Spec1 \times Spec2$$

$$\Longrightarrow \text{compositional testing}$$

- Incremental design by refinement

$$ConcreteSpec \text{ refines } AbstractSpec$$

$$\Longrightarrow \text{refinement testing}$$

Compositional testing : $Spec = Spec1 \times Spec2$

Intuition

intensively testing each component according to the operational profile of the global system
(some behaviours are priviledged while others cannot be performed)

Key points

- Restriction : $Spec1$ and $Spec2$ are input enable
- Hypothesis : Synchroneous communication between IUT_1 and IUT_2 is supposed correctly implemented
- take a test purpose Obj for $Spec$
- build $Obj_{|Spec1}$ the test purpose for $Spec1$ by projection of Obj on $Spec1$
- test IUT_1 with $Obj_{|Spec1}$ as test purpose

Testing refinement : *SpecC refines SpecA*

Intuition

- *SpecC refines SpecA*

  iff all the traces of *SpecA* are traces of *SpecC*, up to the fact that actions introduced by the refinement step are forgotten
- Testing refinement : partial verification by selecting a finite number of traces of *SpecA*

Key points

- Symbolically execution of a selected trace of *SpecA* on *SpecC*
- While executing the trace on *SpecC*, compute the associated path condition
- If *SpecC refines SpecA* then all path conditions have to be satisfiable with the same initial values for the attribute variables.