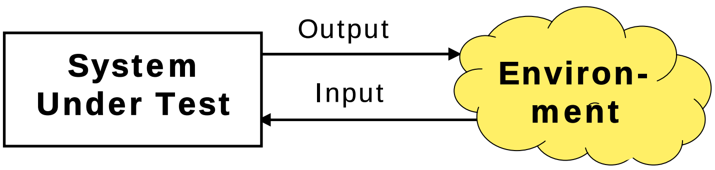
UPPAAL Tutorial

CoVer – Test-Case Selection and Generation for Real-Time Systems

Alexandre David
Paul Pettersson
RTSS'05

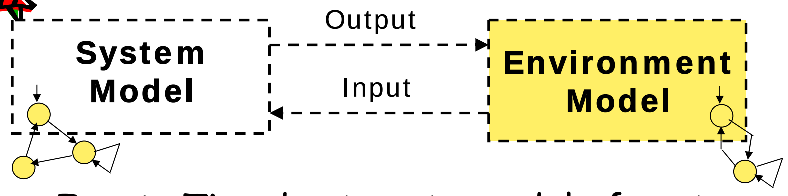
Testing

- Primary validation technique used in industry
- Part of system development life-cycle



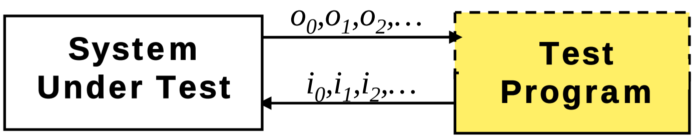
- Expensive, error prone, time consuming (for Real-Time Systems)
- UPPAAL model can be used to generate test specifications

Model-Based Testing



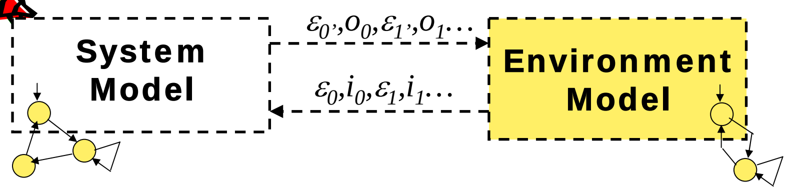
- **Input:** Timed automata model of system and environment
- Test-suite generation algorithm
- **Output:**
  - Test input/output sequence $\sigma_{\phi p} = i_0, i_1, i_2, \ldots$
  - Test suite $T = \{\sigma_1, \ldots, \sigma_n\}$,
    minimized w.r.t global time or global length

Testing Verdict



- Test specification $\sigma_{\phi p} = i_0, i_1, i_2, \ldots$
- Test in/output $\delta_{\phi p} = i_0, o_0, i_1, o_1, i_2, i_3, \ldots$
- Test Verdict:
  - OK, if $\delta_{\phi p} = i_0, o_0, i_1, o_0, i_2, i_3, \ldots$ run of system model
  - NOK, otherwise

Testing Real-Time Systems



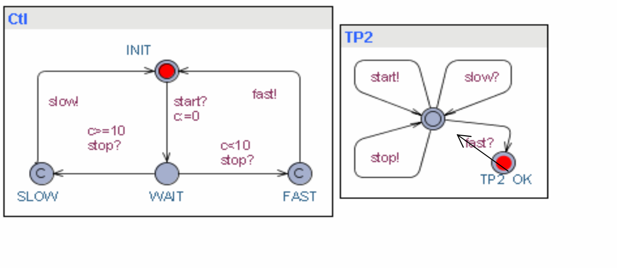
- Test input sequence $\sigma_{\phi p} = \varepsilon_0,i_0,\varepsilon_1,i_1,\varepsilon_2,i_2,\ldots$
- Test in/output $\delta_{\phi p} = \varepsilon_0,i_0,\varepsilon_1,o_0,\varepsilon_1,i_1,o_1,\ldots$
- Test Verdict:
  - OK, if $\delta_{\phi p} = \varepsilon_0,i_0,\varepsilon_1,o_0,\varepsilon_1,i_1,o_1,\ldots$ run of system model
  - NOK, otherwise
- **Timed Automata?**

Controllable Timed Automata

- **Input Enabled**: all inputs can always be accepted
- **Output Urgent**: enabled outputs will occur immediately
- **Determinism**: two transitions with same input/output leads to the same state
- **Isolated Outputs**: if an output is enabled, no other output is enabled

Test Purpose Generation

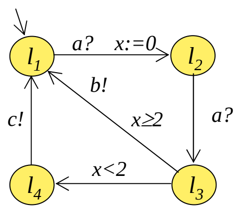
- Generate input sequence for a single test purpose
- Reachability analysis
- Example: “fast” output can be sent by system



- TP2.OK reachable?
- Test input: start!.delay(0).stop!.delay(0).fast?

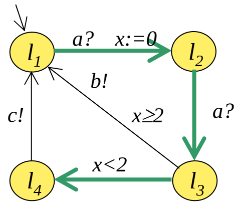
Coverage Based Test Generation

- Systematic testing
- Coverage measurement
- Examples:
  - Location coverage,
  - Edge coverage,
  - Definition/use pair coverage



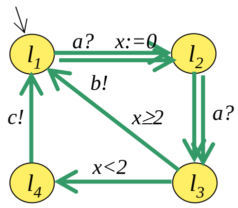
Coverage Based Test Generation

- Systematic testing
- Cover measurement
- Examples:
  - **Location coverage,**
  - Edge coverage,
  - Definition/use pair coverage



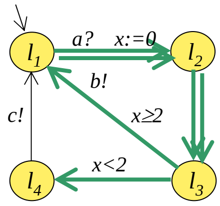
Coverage Based Test Generation

- Systematic testing
- Cover measurement
- Examples:
  - Location coverage,
  - **Edge coverage,**
  - Definition/use pair coverage



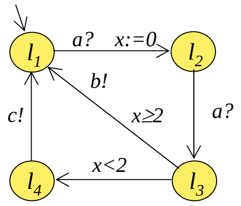
Coverage Based Test Generation

- Systematic testing
- Cover measurement
- Examples:
  - Location Coverage,
  - Edge Coverage,
  - **Definition/Use Pair Coverage**



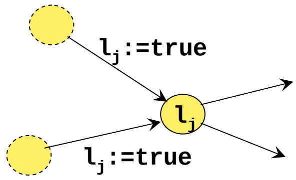
Coverage Based Test Generation

- Systematic testing
- Cover measurement
- Examples:
  - Locations coverage,
  - Edge coverage,
  - Definition/use pair coverage
  - All Definition/Use pairs
- Generated by modified model-checking algorithm in UPPAAL



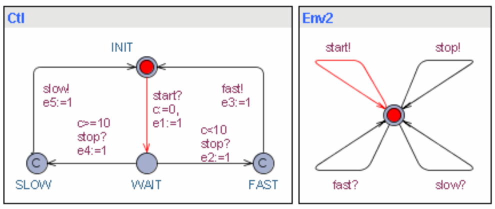
Location Coverage

- Test sequence traversing all locations
- Encoding:
  - Enumerate locations $l_0, \ldots, l_n$
  - Add an auxiliary variable l_i for each location
  - Label each ingoing edge to location i l_i:=true
  - Mark initial visited l_0:=true
- Check: **EF(** l_0=true ∧ … ∧ l_n=true **)**



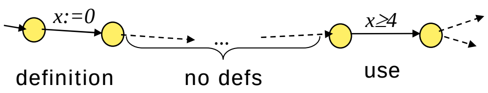
Edge Coverage

- Test sequence traversing all edges
- Encoding:
  - Enumerate edges $e_0, \ldots, e_n$
  - Add auxiliary variable e_i for each edge
  - Label each edge e_i:=true
- Check: **EF(** e_0=true ∧ … ∧ e_n=true **)**



Definition/Use Pair Coverage

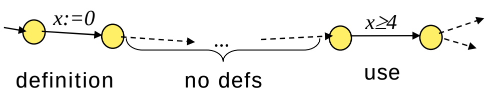
- Dataflow coverage technique
- Def/use pair of variable x:

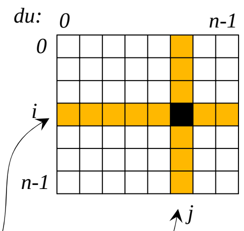


- Encoding:
  - $v_d \in \{ false \} \cup \{ e_0, \ldots, e_n \}$, initially false
  - Boolean array du of size $|E| \times |E|$
  - At definition on edge i: $v_d := e_i$
  - At use on edge j: if(v_d) then $du[v_d, e_j] := true$

Definition/Use Pair Coverage

- Dataflow coverage technique
- Def/use pair of variable x:



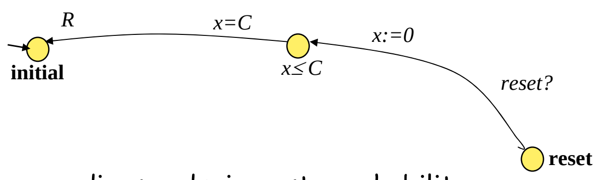


- Encoding:
  - $v_d \in \{ false \} \cup \{ e_0, \ldots, e_n \}$, initially false
  - Boolean array du of size $|E| \times |E|$
  - At definition on edge i: $v_d := e_i$
  - At use on edge j: if(v_d) then $du[v_d, e_j] := true$

- Check:
  - EF(all $du[i,j] = true$)

Test Suite Generation

- In general a set of test cases is need to cover a test criteria
- Add global reset of SUT and environment model and associate a cost (of system reset)

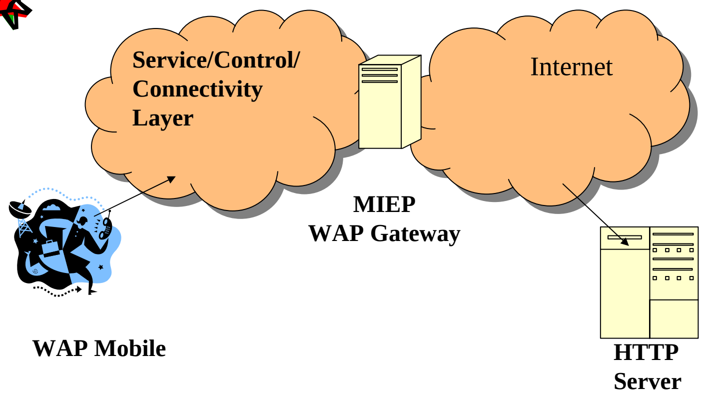


- Same encodings and min-cost reachability
- Test sequence $\sigma = \varepsilon_0, i_0, \ldots, \varepsilon_1, i_1, \textbf{reset}\ \underbrace{\varepsilon_2, i_2, \ldots, \varepsilon_0, i_0}_{\sigma_i}, \textbf{reest}, \varepsilon_1, i_1, \varepsilon_2, i_2, \ldots$
- Test suite $T = \{\sigma_1, \ldots, \sigma_n\}$ with minimum cost

CoVer Tool

- Extension of UPPAAL
  - Graphical user interface
  - Timed Automata models
  - Symbolic on-the-fly algorithms
- Graphical input language for coverage criteria, a.k.a. Observers
- Generation of test specifications

Application w. Ericsson



- Automatic generation of test programs
- Systematic tests with coverage