Constraint-Based Heuristic On-line Test Generation from Non-deterministic I/O-EFSM

Danel Ahman, Marko Kääramees

Tallinn University of Technology
Eliko Competence Centre
Conformance testing

- A system should communicate to its environment according to a specification/protocol
- Black-box view: test@interface
- Embedded systems, services, communication devices
Testing non-deterministic systems

Non-deterministic system may react differently to the same input

- Non-deterministic systems
- Non-deterministic models due to abstraction

On-line testing is needed

- Test cases cannot be prepared beforehand
- Tester must decide inputs during the test based on observed outputs and active goals
- Extensive test planning is costly and not feasible on-line
  - Industrial requirements: 10-100 ms for each step

Practical non-determinism

- output-observability – next state can be determined based on the given input and observed output
Talk Outline

- Introduction and background
- Preliminaries
  - Conformance and test coverage
  - Modelling of the system and test goals
  - RPT- Reactive Planning Tester
- $\chi$RPT – Heuristic RPT
- Case studies
- Conclusions

---

Model

Spec
The vending machine
- latte for 20 kroner
- when more money given

Test Goals
Test latte for >20 kroner
Test all transitions

Symbolic, structural analysis

Strategy

Tester

Adapter

Spec

Model

Strategy

Tester
Conformance

IUT

\[ i_1 | i_2 | i_3 \rightarrow o_1 | o_2 \]

Model

\[ i_1 | i_2 \rightarrow o_1 | o_2 | o_3 \]

IOCO, alternating simulation:

- Every input of the model is acceptable by the IUT
- The resulting output is possible in the model

Hence:

- Only some aspects may be modelled
  - Some inputs (functionality) is not modelled
- IUT may be more deterministic
  - Spec/standard allows some freedom of implementation
Test coverage

When to stop testing?

- **Conformance**
  - all inputs in all states resulting all possible outputs are covered
    - Infeasible/impossible to check all combinations for a general model
- **Coverage**
  - defined structural elements of the model are covered

```python
while done?
    generate input possible in model
    output ← IUT(input)
    if output not possible in model
        return(test failed)
endwhile
return(test succeeded)
```
Modelling of IUT and test goals

- IUT is modelled by an **Input/Ooutput Extended Finite State Machines (I/O-EFSM)**
  - State space consists of locations and state variables
  - An edge (transition) has
    - input and output with its data parameters
    - guard
    - update function of the state variables

- **Background theory**
  - linear arithmetics
  - other theories possible

- Test goals modelled by **traps**
  A trap is a \(\langle\text{predicate}\rangle\) associated to an edge
The power of traps

- A trap is a \textit{predicate} associated to an edge
- Several goals can be expressed by traps
  - transition coverage: every edge has a trap \textit{true}
  - transition sequence trap with reference to other traps
  - Advanced goals using auxiliary variables consequent transitions, repeated pass, ...
- Properties not expressible by traps
  - Liveness properties but it is not possible to test for liveness anyway
  - Assertions/invariants – it never happens/always holds The model specifies only allowed behaviours
  - No LTL, CTL, but still quite powerful Many significant subsets can be modelled by aux variables
RPT – Reactive Planning Tester

- Offline symbolic test strategy generation
  - Based on backwards symbolic reachability analysis
  - Relates input and reachability of a trap
  - Predicates $\text{Strategy}_{l \rightarrow \text{trap}}(I,S)$
    - $l$ – location
    - $I$ – input with its parameters
    - $S$ – state (valuation of the state variables)

- Online test data generation
  - next trap to be covered is selected
  - the strategy predicates are used to find an input
    by model generation using SMT solver
  - input is sent to the SUT and output observed
Testing process

- Data generation based on the test strategy
  - should be done on-line for non-deterministic model
  - constraint solving / satisfying model generation used
- Several goals at the same time
  - Minimize the length of the overall testing process
  - Reset to the initial state may be expensive
    - Eg reboot of the SUT
  - The purpose is to cover as many test goals (traps) in one run as possible
Simple loop example

- Simple artificial example where the right sequence of inputs should be given to reach the goal

\[ x + y + z < 18 \]
\[ x + y + z = 18 \land z > 2 \]
\[ x + y + z = 18 \land y < 6 \]
\[ x + y + z < 18 \]
\[ x + y + z \geq 18 \]

MBT 2012
Bounded analysis

depth 8
Bounded analysis of the simple loop example

- Constraints generated with bound (traversal depth) 2

```
x=10 \land y=6 \land z=1

x=11 \land y=5 \land z=2

x=10 \land y=6 \land z=2
```

Diagram showing the states and transitions with constraints associated with each state.
χRPT - Heuristic Reactive Planning Tester

- Complementary to bounded strategy generation
  - Guides the testing process until a state is reached where a RPT strategy is applicable
- Uses an objective function to find an action that guides IUT towards some test goal
- Aims several goals (traps) at the same time to minimize the overall test time
- Based on the ideas of
  - forward, explicit state analysis
  - local search
  - tabu search
On-line test generation

while exist uncovered traps
    if RPT strategy exists in the current state for any uncovered trap
        RPT on-line testing
    else
        candidates ← Generate_Action_Candidates
        action ← Choose_Most_Promising_Action(candidates, tabu_list)
        output ← Interact_with_IUT(action)
        if the output of does not conform to the model
            stop(test_failed)
            simulate input/output on model and determine the next_state
            add next_state to the tabu_list
end while
stop(test_passed)
Search neighbourhood (candidates)

- Partitioning of the traps
  - uncovered
  - covered
  - unreachable

- Tabu lists
  - avoid the state that is explored already

- Closest locations with strategy constraints
  - a set of closest locations with strategy constraints for every pair of location and trap found off-line
  - these locations are the goals of heuristic guidance
Selecting tester action

- possible actions are simulated on the model and the result is evaluated using an objective function
- objective consists of
  - graph based distance between the simulated location and the location with a RPT strategy
  - violation degree of the RPT strategy constraint in the simulated state
    \[ f = dist^2 + viol^2 \]
- Selection of the candidate actions narrowed in 3 phases
  - most promising actions optimized for the best input parameters and the best selected for the next step
Violation degree

- Measures how much the current state (valuation) violates some constraint
  - 0 if the constraint is satisfiable in the current state
  - >0 if not satisfiable

<table>
<thead>
<tr>
<th>A, B – logical formulae, a, b – arithmetic expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\nu(a \geq b) = \text{abs}(\min(0, \nu(a) - \nu(b)))$</td>
</tr>
<tr>
<td>$\nu(a &gt; b) = \text{abs}(\min(0, -1 + \nu(a) - \nu(b)))$</td>
</tr>
<tr>
<td>$\nu(a &lt; b) = \text{abs}(\max(0, 1 + \nu(a) - \nu(b)))$</td>
</tr>
<tr>
<td>$\nu(a \leq b) = \text{abs}(\max(0, \nu(a) - \nu(b)))$</td>
</tr>
</tbody>
</table>
Telecom Billing Case-Study
Telecom Billing Case-Study

- Model: 13 locations, 47 transitions
- 7 variables of range [0 .. 32000]
- Path length to trap from initial state: 189
- Size of ASCII representation of the strategy: 34MB
- Time for test generation (symbolic analysis + input) [1 GHz Opteron]

<table>
<thead>
<tr>
<th>Strategy generation path length (time (s))</th>
<th>189 (4644)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bounded strategy depth (time (s))</td>
<td>100 (2120)</td>
</tr>
<tr>
<td>Heuristic test data path length (time (s))</td>
<td>230 (6,7)</td>
</tr>
<tr>
<td>Avg test data gen (ms)</td>
<td>51</td>
</tr>
</tbody>
</table>
Complexity issues

- Constraints limited to decidable theories
  - linear arithmetic (+ others supported by solvers)
- Theoretical limits
  - SAT problem is NP-complete
  - decision procedure of Presburger arithmetic is double-exponential
- Practical aspects
  - number of constraints is in $O($traps$\times$transitions$)$
  - Z3 does a good job in satisfiability checking and simplification in strategy generation
  - Comet used for constraint solving and violation degree calculation in $\chi$RPT
- Balancing complexity of the strategy and on-line data generation
  - feasibility can be achieved by tuning the balance
Main results

- Model-based conformance testing framework for non-deterministic I/O-EFSM models
- Computationally expensive strategy generation and neighbourhood analysis done off-line
- Efficient on-line test planning
  - selection of input for each step in 10-100 ms range
  - usable in the industrial setting