Automatic Test Generation for LUSTRE/SCADE Programs

Virginia Papailiopoulou  (virginia.papailiopoulou@imag.fr)
Supervisor: Ioannis Parissis  (ioannis.parissis@imag.fr)

Laboratoire d’Informatique de Grenoble

Doctoral Symposium - ASE 2008
September 15-19, L’Aquila, Italy
Synchronous / safety-critical software

- Esterel
- Signal
- LUSTRE
- Lutess, LUSTRACTU
- Gatel
- Lurette
- SCADE

control/command programs

high level of criticality
Test data generation with Lutess

Informal requirements

Test data selection

Test execution

Quality evaluation
Automatic test data generation

Context

- Constrained random test generator
- Test model
- Oracle
- SUT
- synchronous programs

Testnode program(...) returns (...); let
    -- specification
tel ;

valid & suitable test cases

problem specification

formal environment description
Automatic test data generation

Proposed approach

- Composition of a testing methodology using Lutess
  - step-by-step environment construction

⇒ system tested adequately
⇒ observe many (different) states
⇒ detection of faults

⇒ Provide the guidelines in the procedure of testing with Lutess
Automatic test data generation

Contributions & ongoing work

- Definition of the basic steps of the testing methodology
  1. Domain definition
  2. System dynamics
  3. User-defined scenarios
     - Invariant
     - Probabilistic
  4. Safety property guided testing
     - Safety property specification
     - Adding hypotheses
Automatic test data generation

Contributions & ongoing work

- Evaluation on large case studies
  1. steam-boiler specification (Abrial 1994)
     - integer values
     - ~15 properties on 16 variables
     - operational profiles guided testing (steps 1-3)

  2. landing gear controller
     - boolean values
     - 20, 55, 77 properties on 25 variables (3 different models)
     - safety property guided testing (steps 1-4)
Test data quality evaluation

Informal requirements

Test data selection

Test execution

Structural Coverage Assessment (LUSTRUCTU)

1. Multiple clocks
2. Integration testing
3. data-flow + FSM

Quality evaluation
Structural coverage assessment

Context - LUSTRE overview

Declarative language

Graphical representation = operator network

Data flow

Cyclic behavior

Synchronism hypothesis

Node

\[
\text{node never } (E : \text{bool}) \quad \text{returns} (S : \text{bool});
\]

\[
\text{let} \\
S = \text{not}(E) \rightarrow (\text{not}(E) \quad \text{and} \quad \text{pre}(S));
\]

ten;

\[E = (E_1, E_2, E_3, \ldots)\]
Structural coverage assessment

**Context - Coverage model**

- operator network
- path activation conditions

**Diagram**

Path without cycles:

\[ p_1 = (RCL, L2, L3, L1, L6, S) \]

Path with cycles:

\[ p_2 = (RCL, L2, L3, L9, L8, L5, L4, L3, L1, L6, S) \]
Structural coverage assessment

**Context - Activation conditions**

- change in the input value $\Rightarrow$ change in the output

\[ (a = \text{false}) \text{ or } (b = \text{true}) \]
\[ (l = \text{true}) \text{ or } (c = \text{false}) \]

$\Rightarrow AC = (a = \text{false or b= true}) \text{ and } (l = \text{true or c = false})$
Structural coverage assessment
Context - Coverage Criteria

- adapted to the LUSTRE paradigm
- measure the test thoroughness and the test effort
- complexity w.r.t. the path length and the strength of the criterion

<table>
<thead>
<tr>
<th>Basic Criterion (BC)</th>
<th>+</th>
<th>Elementary Conditions Criterion (ECC)</th>
<th>+</th>
<th>Multiple Conditions Criterion (MCC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td></td>
<td>=</td>
<td></td>
<td>=</td>
</tr>
<tr>
<td>satisfy the</td>
<td>+</td>
<td>satisfy all input variations</td>
<td>+</td>
<td>satisfy all edges variations</td>
</tr>
<tr>
<td>activation conditions</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Structural coverage assessment

Context - Coverage Criteria

- adapted to the LUSTRE paradigm
- measure the test thoroughness and the test effort
- complexity w.r.t. the path length and the strength of the criterion
Structural coverage assessment

Context - LUSTRUCTU (LUsre STRUCTUral)
Structural coverage assessment

*Proposed approach*

- Extension of the criteria
  1. Multiple clocks
  2. Integration testing
  3. Parallel use of data flow + control flow graphs
**Multiple clocks**

- prevent useless operations
- save computational resources
- LUSTRE operators: *when*, *current*

\[
E = e_0 \ e_1 \ e_2 \ e_3 \ e_4 \ e_5 \ e_6 \ e_7 \ e_8 \ \ldots \\
B = F \ F \ T \ T \ F \ T \ F \ F \ T \ T \ \ldots \\
X=E \ \text{when} \ B = x_0=e_2 \ x_1=e_4 \ x_2=e_7 \ x_3=e_8 \ \ldots \\
Y=\text{current} \ X = \text{nil} \ \text{nil} \ e_2 \ e_2 \ e_4 \ e_4 \ e_4 \ e_7 \ e_8 \ \ldots
\]

⇒ Define the activation conditions for *when* and *current*

- ‘Extending Structural Test Coverage Criteria for LUSTRE Programs with Multi-clocks Operators’ @ FMICS 2008
Integration testing

- system coverage for large-sized systems

⇒ Define new criteria with a notion of abstraction
Data-flow diagrams & FSM

- Parallel use of data-flow and control flow graphs (SCADE V6)
Structural coverage assessment

*Contributions & ongoing work*

- Definition of the activation conditions for *when* and *current*
  - theoretical evaluation on simple programs
  - implementation in LUSTRACTU

- SIESTA project (www.siesta-project.com)
  - academic + industrial partners
Questions