Automated Generation of Parameterizable TTCN-3 Test Cases for Embedded Systems

Jens R. Calamé
Natalia Iouustinova
Jaco van de Pol
CWI Amsterdam

Natalia Sidorova
TU Eindhoven
Agenda

1. Embedded Systems and their Environments
2. Test Generation using Test Purposes
3. Test Case Parameterization and TTCN-3
4. Case Study CEPS
5. Conclusion
Embedded Systems and their Environments

- Embedded systems communicate with their environments.
- The latter is underspecified or not specified at all.
Data Abstraction

- Regarding all possible communication with the environment leads to a large/infinite state space of the system.
- Abstracting the possible input from environment to one abstract value forecures this problem.
Data Abstraction – Example
Agenda

1. Embedded Systems and their Environments
2. Test Generation using Test Purposes
3. Test Case Parameterization and TTCN-3
4. Case Study CEPS
5. Conclusion
Our Test Generation Process

System Specification (e.g. UML)

Test Purposes

Automated Test Case Generation

Param. Testcases (TTCN-3)

Automated Test Data Selection

Test Data

Automated Test Execution
Test Generation using Test Purposes

System Model

Test Purpose
Test Generation with Test Purposes cont’d
Agenda

1. Embedded Systems and their Environments
2. Test Generation using Test Purposes
3. Test Case Parameterization and TTCN-3
4. Case Study CEPS
5. Conclusion
Test Case Parameterization

- To be solved after data abstraction:
  - Pruning of traces introduced by overapproximation
  - Finding possible value ranges for test data

- Rule System (Prolog)
  - Represents SUT specification
  - Defines rules on data (addition, substraction; and, or etc.)
  - Defines rules for process behavior of Spec

- Query (Prolog)
  - Represents the abstract test case
    (one rule per trace)
  - All transitions in trace taken as query body
  - Alternative: one query for whole CTG
Testing with Abstraction – Rule System

\[
\begin{align*}
&0 \xrightarrow{\text{?initPin}(p)} 1 \xrightarrow{\text{?initBalance}(b)} 2 \\
&\quad \xrightarrow{?\text{getPin}(x)} 3 \\
&\quad \xrightarrow{(x \neq \text{pin}) \rightarrow !\text{pinIncorrect}} 3 \\
&\quad \xrightarrow{(x = \text{pin}) \rightarrow !\text{pinCorrect}} 4 \\
&\quad \xrightarrow{(y > b) \rightarrow !\text{LowBalance}(b)} 8 \\
&\quad \xrightarrow{(y \leq b) \rightarrow !\text{Money}(y); \ b := b-y} 7 \\
&\quad \xrightarrow{?\text{getAmount}(y)} 6 \\
&\quad \xrightarrow{?\text{getAmount}(y)} 5 \\
&\quad \xrightarrow{!\text{Balance}(b)} 6 \\
&\text{...}
\end{align*}
\]

getPin(state(2,P,B,X,Y), state(3,P,B,X1,Y), param(X1)).

pinIncorrect(state(3,P,B,X,Y), state(8,P,B,X,Y),param(_)) :- X \neq P.

pinCorrect(state(3,P,B,X,Y), state(4,P,B,X,Y), param(_)) :- X = P.
Testing with Abstraction – Query

oracle(P,B,X,Yin,Yout) :-
initPin(state(0,0,0,0,0), G1, param(P)),
initBalance(G1, G2, param(B)),
getPin(G2, G3, param(X)),
pinCorrect(G3, G4, _),
getAmount(G4, G5, param(Yin)),
money(G5, _, param(Yout)).

P=P{-1.0inf..1.0inf}
B=B{-1.0inf..1.0inf}
X=P{-1.0inf..1.0inf}
Yin=Yout{-1.0inf..1.0inf}
Yout=Yout{-1.0inf..1.0inf}

Yout{-1.0inf..1.0inf}-B{-1.0inf..1.0inf}=<0
Generation of TTCN-3

- Test case parameters
- Actual test code
- Variables
- TestCase
- GenericComponents
- Port definitions
- DataTypes
- Signatures
- Non-primitive data types
- Procedure signatures, Message templates
- <<import>>
- <<import>>
- <<import>>
- <<import>>
- <<import>>
Test Execution with Online Constraint Solving

Diagram:
- MTC (Test case)
- SUT
- Constraint Solver
- Spec

Connections:
- MTC (Test case) to SUT
- MTC (Test case) to Constraint Solver
- Constraint Solver to Spec
Agenda

1. Embedded Systems and their Environments
2. Test Generation using Test Purposes
3. Test Case Parameterization and TTCN-3
4. Case Study **CEPS**
5. Conclusion
Case Study CEPS

- **Common Electronic Purse Specifications**: protocol for electronic payment using a multi-currency smart-card

- Input and output parameters of card actions mainly natural numbers $\rightarrow$ (in)finite domain

- Variables partially arrays (up to 16 elements in simplified realization)

- Specification consisted of:
  - 54 transition edges
  - 44 process variables (netto)
  - 207 process variables (brutto), due to arrays of structures
Case Study CEPS (cont'd)

- Instantiation and reduction (abstracted specification): ca. 9.5 min. on five 2.2GHz Athlon 64bit single CPU computers (1 GB RAM each)
- Generation of two test cases: 594 and 109 states, resp. in less than one second (one 2.2GHz AMD Athlon XP 32 bit CPU and 1 GB RAM)
- Constraint solving produces results in negligible time
Agenda

1. Embedded Systems and their Environments
2. Test Generation using Test Purposes
3. Test Case Parameterization and TTCN-3
4. Case Study CEPS
5. Conclusion
Conclusion

- Data abstraction makes state-based test generation applicable to systems with large data domains
- Approach successfully tested on CEPS case study
- Test cases in TTCN-3 are generated from formal specifications with a binding to UML
- On-the-fly constraint-solving during test execution is ongoing work
- Website: [www.cwi.nl/~calame/dataabstr.html](http://www.cwi.nl/~calame/dataabstr.html)