Poster: Behavioral Model-Based Testing of Variability Intensive Systems
Devroey, Xavier

Publication date: 2015

Document Version
Peer reviewed version

Link to publication

Citation for published version (HARVARD):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 29. Dec. 2018
Behavioral Model-Based Testing of Variability Intensive Systems
Xavier Devroey, Gilles Perrouin, Pierre-Yves Schobbens, Patrick Heymans

- Focus on Variability-Intensive Behavioural Models (software product lines, plugin-based systems, etc.)
- Formal Foundations (Featured Transition Systems) suitable for Analysis and Checking
- Combining Model-checking and Test-case Generation techniques in VIBeS Framework
- Cover various Testing Activities (test-case selection, prioritization, mutation assessment, etc.)

Test Model

Model-Based Testing Process
1. Models definition (Java DSL)
2. Selection criteria definition
   - Maximizing
   - Usage coverage (based on a usage model) [4]
   - Number of mutants killed by the tests [1]
   - Minimizing
   - Dissimilarity measure (based on distance def.)
3. Concrete mapping definition (TBD)
4. Test results analysis (TBD)

Models Def.
Featured Transition System Java DSL
public class CPTerminalFts extends FeaturedTransitionSystemDefinition {
  @Override
  protected void define() {
    from("i").action("insert_card")
        .to("cardin");
    from("cardin").action("init_schema")
        .fexpr("dd || cr").to("App_init");
    ...}
}
Feature Model
Boolean Conjunctive Normal Form (CNF) in DIMACS format used as input to SAT and BDD solvers

Selection Criteria Def.
Structural coverage
SolverFacade solver = new Sat4JSolverFacade(dimacsFile);
TestSet set = allStatesSelection(fts, solver);
Dissimilarity measure maximization
set = dissimilarLocalMaximumDistance(fts, solver, rbrTestCases, time);
set = dissimilarGlobalMaximumDistance(fts, solver, rbrTestCases, time);

Model Based Mutation Testing
- Goals: Quality Assessment (QA)
  - Tester: QA of a set of abstract test cases
  - Researcher: QA of a test case selection method
  - Mutant = Test Model with injected fault(s)
- Result of the application of a mutation operator
- Execution of the test cases on the mutants
  - Number of mutants detected
  - Mutation score = Number of mutants detected / Total number of mutants

Future Work
- Higher order and equivalent mutants detect./gen.
- Test cases generation using counter examples in ProVeLines FTS model checker
- Abstract test cases concretization
- Empirical assessment on an industrial case study

Mutant Execution using Featured Mutants Model

Bibliography

Contact: xavier.devroey@unamur.be
https://projects.info.unamur.be/vibes/