Cause-Effect Graphs for Test Models
Based on UML and OCL

Stephan Weißleder, Dehla Sokenou
Table of Content

- Introduction to model-based testing
- Example: freight elevator
- Automatic test suite generation
  - Algorithm
  - Quality measurement with coverage criteria
- Relations between OCL expressions
- Introduction to cause-effect-graphs
- Using cause-effect-graphs to clarify OCL relationships
Model-Based Testing
Model-Based Testing

![Diagram of Model-Based Testing](image)
Model-Based Testing

- Requirements Specification
  - System Modeling
    - Base
      - Model
      - 1
    - ...
    - ...
  - System Implementation
    - System
Model-Based Testing
Model-Based Testing

- Requirements Specification → Test Specification
- System Modeling
- Test Case Specification → Test Modeling
- System Implementation

Stephan Weißleder, HU Berlin
Dehla Sokenou, GEBIT Solutions
Model-Based Testing

![Diagram of the Model-Based Testing process](image.png)
Model-Based Testing

Cause-Effect-Graphs for Test Models
Based on UML and OCL

Stephan Weißleder, HU Berlin
Dehla Sokenou, GEBIT Solutions
Example of a Freight Elevator

- **idle**
  - Conditions: \([\text{actualWeight} > \text{maxWeight}]\)
  - Transitions:
    - **removeWeight** \((w) / \text{subtractFromWeight}(w)\)
    - **insertWeight** \((w) / \text{addToWeight}(w)\)

- **button pressed**
  - Conditions: \([\text{actualWeight} <= \text{maxWeight}]\)
  - Transitions:
    - **pressButton** \((b, r) [(b <> \text{currentFloor}) \text{ and } (b > \text{basement} \text{ or } r > \text{minRank})]\)
    - **/setButton** \((b)\)

- **start moving**
  - Conditions: \([\text{actualWeight} = \text{minWeight}]\)
    - **/actualWeight** \((\text{minWeight})\)
    - \([\text{actualWeight} > \text{minWeight}]\)

- **move fast**
  - Transition: **reachFloor**

- **move slow**
  - Transition: **reachFloor**
Automatic Test Case Generation from UML and OCL

• Model-based test generation from behaviour specifications:
  – Search forward from initial node
  – Search backward from test goals
• Test goal:
  – Coverage criterion applied to a concrete model
  – Example: one state for All-States
• Generate abstract test case
  – Find a path
• Generate concrete test case
  – Find concrete input values
Automatic Test Case Generation from UML and OCL

- Generation of test cases:
  - Path from initial node to test goal contains conditions (e.g. OCL)
  - Due to conditions not each found path is feasible
  - Consequence: include conditions into search algorithm
  - Deal with the relations between OCL conditions along the path
Automatic Test Case Generation from UML and OCL

• Generation of Test Cases:
  – Classify all variables used in the OCL expressions
    • Which variables can change?
  – Algorithm - for each guard:
    • try to find postconditions that influence the result of the guard
    • Combine guard and postcondition to a new condition
    • If there are changeable variables in the condition: continue search
  – Basic Idea:
    • Transform conditions on system attributes into conditions on input parameters
    • Use them as input partitions
Quality of Test Suites

- Satisfying coverage criteria:
  - Algorithm generates tests that satisfy all guard conditions
  - Several coverage criteria also need negative cases:
    - Condition Coverage
    - Decision Coverage
    - MC/DC
  - Consequence:
    - Add test goals for transformed guard conditions
    - Find test cases that satisfy them

\[
\text{[actualWeight} > \text{minWeight]} \\ 
\text{[actualWeight} \leq \text{minWeight]} 
\]
Classification of OCL Elements

- Why classify OCL elements at all?

Postcondition:

\[ X = Y \text{@pre} \quad X := Y \]
\[ Y := X \]
\[ X := Z \]
\[ Y := Z \]
Classification of OCL Elements

- Examples for Condition Transformations
  - Guard: $\text{guard} = \text{actualWeight} > \text{minWeight}$
    (passive) (passive)
  - Post: $\text{post} = \text{actualWeight} = \text{actualWeight}_{pre} + w$
    (active) (passive) (passive)
  - New: $\text{new} = \text{actualWeight}_{pre} + w > \text{minWeight}$
  - Satisfaction of guard condition is now influenced by input parameters

- Is this relationship easily visible?
- Supporting readability by using cause-effect-graphs?
Introduction to Cause-Effect-Graphs

- Graphical representation of
  - Logical expressions
  - Logical dependencies
- Easier to read than textual representation

Example: $C = A \lor B$
Introduction to Cause-Effect-Graphs

- \((b \not< currentFloor) \text{ and } (b > \text{basement or } r > \text{minRank})\)
  
  
  \[\begin{array}{c}
  A \quad B \quad C \\
  \end{array}\]

- \(E = \text{true}, \ F = \text{false}\)
Using Cause-Effect-Graphs to clarify OCL Relationships

- Expressing relationships between OCL expressions
- Guard: \([\text{actualWeight} > \text{minWeight}]\)
- Relation 1
  - Post of \text{addToWeight}: \text{actualWeight} = \text{actualWeight}@pre + w
  - New1: \text{actualWeight}@pre + w > \text{minWeight}
- Relation 2
  - Post of \text{subtractFromWeight}: \text{actualWeight} = \text{actualWeight}@pre - w
  - New2: \text{actualWeight}@pre - w > \text{minWeight}
- Combination
Discussion

- Improved readability by using cause-effect-graphs?
- Case studies needed
- Cause-Effect-Graphs are related to the test case
  - The order of input stimuli has effects on the cause-effect-graph
  - Examples:
    - Execute addToWeight AND removeFromWeight
    - Execute addToWeight two times
  - Result:
    - Adapt causes depending on the structure of the cause-effect-graph
- Include information about input events in cause-effect-graph?
- Same principle used in prototype ParTeG (http://parteg.sf.net)