Software Testing

Sibylle Schupp[1]

[1]Institute for Software Systems/Institut für Softwaresysteme
Hamburg University of Technology (TUHH)

Spring 2022

Lecture 7

Outline

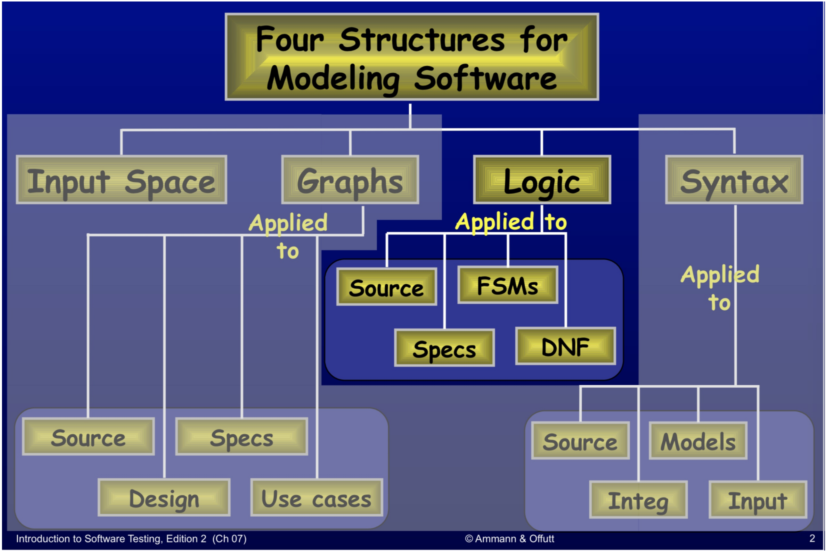
Recall

- Faults, failure, and error
  - Fault: defect. Error: internal manifestation of a fault. Failure: missed requirements. Thus, failure $\Rightarrow$ error $\Rightarrow$ fault
- RIPR model: reachability, infection, propagation, revealability
  - Graph coverage criteria focus on reachability.
  - Input space partitioning: independent of RIPR model
- Subsumption: relation between coverage criteria

  C_1 subsumes C_2 iff every test set that satisfies C_1 also satisfies C_2.

Example: FindLast

```
// Introduction to Software Testing
// Authors: Paul Ammann & Jeff Offutt
public class FindLast
{
  /**
    * Find last index of element
    *  @param x array to search
    *  @param y value to look for
    *  @return last index of y in x; -1 if absent
    *  @throws NullPointerException if x is null
    */
   public static int findLast (int[] x, int y) {
      for (int i=x.length-1; i > 0; i--) {
         if (x[i] == y)  {
            return i;
         }
      }
      return -1;
   }
}
```



Four Structures for Modeling Software
Input Space
Graphs
Logic
Syntax
Applied to
Applied to
Applied to
Source
FSMs
Specs
DNF
Source
Specs
Design
Use cases
Source
Models
Integ
Input
Introduction to Software Testing, Edition 2 (Ch 07)
© Ammann & Offutt
2

Safety-critical standards

http://www.verifysoft.com/en_do-178b.html

Catastrophic (level A), hazardous-severe (level B), major (level C), minor (level D) or no-effect (level E).

According to the DO-178B-level the following test coverage (code coverage) is required :

DO-178B Level A:
Modified Condition Decision Coverage (MC/DC)
Branch/Decision Coverage
Statement Coverage

DO-178B Level B:
Branch/Decision Coverage
Statement Coverage

DO-178B Level C:
Statement Coverage

Logic approaches

Coverage criteria for logic testing can be grouped into two classes:

- Semantic logic coverage
  - Based on the meaning of a logic expression
- Syntactic logic coverage
  - Based on the form of a logic expression
- Semantic coverage tests the intended logic meaning and allows reusing tests for equivalent predicates.
- Syntax coverage tests the particular formulation and allows for checks of common (syntactic) mistakes.

Why semantic logic criteria?

- Required
  - Example: Federal Aviation Adminstration
- Predicates without predefined syntactic form easily available
  - Decisions in programs
  - Decisions in UML activity diagrams
  - Guards in finite state machines
  - Requirements (formal, informal)
- General idea: tests choose certain kinds of truth assignments.

Logic predicates and clauses

Definition

- A predicate is an expression in a language L that evaluates to a boolean value.
  - In formal logic, predicates are mathematically defined.
  - In testing, predicates are more loosely defined. Logical operators include $\neg, \wedge, \vee, \rightarrow, \oplus, \equiv$. Predicates may contain operators from programming languages as well as boolean functions.
- A clause is a predicate with no logical operators.

Examples (predicates and clauses)

$$(a < b) \vee f(z) \wedge D \wedge (m \geq n \cdot o)$$

Clauses

- $(a < b)$: relational expression
- $f(z)$: boolean function
- D: boolean variable
- $(m \geq n \cdot o)$: relational expression

Statistics

Source: textbook, Ch. 8.1

- Most predicates have few clauses.
  - 88.5% have 1 clauses
  - 9.5% have 2 clauses
  - 1.35% have 3 clauses
  - Only 0.65% have 4 or more clauses
- Source: 63 open-source programs ($> 400,000$ predicates)

From natural language to formal language

https://criticalthinkeracademy.com/courses/2514/lectures/761246

- Some rule of thumbs exist for mapping from natural language to formal language.
  - "John and Mary went to the store": conjunction
  - "Mary will walk the dog if John agrees to maker dinner": implication
  - "If you leave before 6:30 AM, take S3 to Jungfernstieg, if you leave after 7:00 AM, take S31 to Main Station, then Main Station to Jungfernstieg." Compare:
    - $\text{time} < 6.30 \to \text{train} = \text{S3} \wedge \text{time} > 7.00 \to \text{train} = \text{S31}$
    - $\text{time} < 6.30 \to \text{train} = \text{S3} \wedge \text{time} \geq 6.30 \to \text{train} = \text{S31}$
  - "I am interested in both Software Testing and Software Verification"
    - $\text{course} = \text{SW-Testing} \vee \text{course} = \text{SW-Verification}$
- Yet, the gap between a formal and an informal language is inevitable, fundamentally.

Outline

1. Logic coverage I
   - Simple semantic coverage
   - Active clause coverage

Logic-based testing, semantically

- Two major steps
  - Model software in terms of predicates
  - Design tests that satisfy certain combinations of clauses
- Abbreviations:

P	set of predicates
p	single predicate in P
C	set of clauses
C_p	set of clauses in predicate p
c	single clause in C

Predicate and clause coverage

Definition

- For predicate coverage (PC), TR contains for every predicate p, $p \in P$, two requirements: p evaluates to true and p evaluates to false.
- For clause coverage (CC), TR contains for every clause c, $c \in C$, two requirements: c evaluates to true and c evaluates to false.

Example (predicate coverage)

Consider:

$$((a < b) \vee D) \wedge (m \geq n \cdot o)$$

Predicate coverage:

- Case true: $a = 5,\ b = 10,\ D = \mathit{true},\ m = 1,\ n = 1,\ o = 1$
  Check
  $$((5 < 10) \vee \mathit{true}) \wedge (1 \geq 1 \cdot 1) \text{ evaluates to } \mathit{true}$$
- Case false: $a = 10,\ b = 5,\ D = \mathit{false},\ m = 1,\ n = 1,\ o = 1$
  Check
  $$((10 < 5) \vee \mathit{false}) \wedge (1 \geq 1 \cdot 1) \text{ evaluates to } \mathit{false}$$

Example (clause coverage)

The same example:

$$((a < b) \vee D) \wedge (m \geq n \cdot o)$$

Clause coverage:

Clause	Valuation	Values
$(a < b)$	true	$a = 5, b = 10$
	false	$a = 10, b = 5$
D	true	true
	false	false
$(m \geq n \cdot o)$	true	$m = 1, n = 1, o = 1$
	false	$m = 1, n = 2, o = 2$

Two test cases still suffice:

1. $a = 5, b = 10, D = true, m = 1, n = 1, o = 1$
2. $a = 10, b = 5, D = false, m = 1, n = 2, o = 2$

Discussion

- Two simple coverage criteria
- PC does not subsume CC.
  - PC does not need to vary its clauses.
  - In the presence of short circuit evalution, PC does not even have to exercise all clauses.
- But also: CC does not subsume PC.
  - Doesn't seem a useful criterion, thus

Combinatorial coverage (CoC)

Definition

For combinatorial coverage (CoC) TR has for every predicate p, $p \in P$, test requirements for the clauses in C_p to evaluate to each possible combination of truth values.

- CoC is sometimes called Multiple Condition Coverage.

Example (CoC)

$$((a < b) \vee D) \wedge (m \geq n \cdot o)$$

	$(a < b)$	D	$(m \geq n \cdot o)$	$(a < b) \vee D \wedge (m \geq n \cdot o)$
1	true	true	true	true
2	true	true	false	false
3	true	false	true	true
4	true	false	false	false
5	false	true	true	true
6	false	true	false	false
7	false	false	true	false
8	false	false	false	false

Discussion

- Expensive
  - 2^N tests (N = number of clauses in a predicate)
  - Impractical for more than 3 or 4 clauses
- Informally speaking
  - Can one test the clauses not as combinations but “independently”?
  - Need to formalize “independently”: active clauses

Outline

Determination

Definition

- The clause c_i in p that is currently in focus is called the major clause. All other clauses c_j of p, $i \neq j$, are then called minor clauses.
- A major clause c_i of p determines p if the minor clauses $c_j \in p$ have values so that changing the value of c_i changes the truth value of p.

Discussion:

- It is not required that c_i and p evaluate to the same truth value.
- Counter example (determination)
  $((a < b) \vee \mathit{true}) \wedge (c \geq d)$: value of the clause $a < b$ does not determine the predicate.

Example (determination)

- $p \equiv A \vee B$
  - Let A be major clause. If B is false, A determines p. If B is true, A does not determine p.
  - Similarly, if B is major clause. Iff A is false, B determines p.
- $p \equiv A \wedge B$
  - Let A be major clause. If B is true, A determines p. If B is false, A does not determine p.
  - Similarly, if B is major clause. If A is true, B determines p.

Active clause coverage (ACC)

Definition

For each p in P and each major clause c_i in C_p, choose minor clauses c_j, $j \neq i$ so, that c_i determines p. For active clause coverage (ACC) TR has two requirements for each c_i: c_i evaluates to true and c_i evaluates to false.

- Example: $a \vee b$ (major clause in boldface)

a	b
T	f
F	f
f	**T**
f	**F**

- ACC is a form of MCDC ("modified condition decision coverage")
- Problem: ambiguity. Do the minor clauses have to have the same values for the two truth assignments to c_i ?

Example (ambiguity)

Consider

$$p \equiv a \vee (b \wedge c)$$

- Let a be the major clause. Is the following pair of tests allowed? (a=true, b=false, c=true) and (a=false, b=false, c=false)
- Possible interpretations of ACC
  - The minor clauses need to stay the same when the major clause changes.
  - The minor clauses do not need to stay the same.
  - The minor clauses force the predicate to evaluate to both true and false.

General active clause coverage (GACC)

Definition

For each p in P and each major clause c_i in C_p, choose minor clauses c_j so that c_i determines p. For general active clause coverage (GACC) TR has two requirements for each major clause c_i: c_i evaluates to true, and c_i evaluates to false. The values for the minor clauses c_j need not be the same for the two values of c_i.

Discussion

- GACC does not subsume predicate coverage.
- Ex.: $p \equiv a \leftrightarrow b$.

a	b
T	**T**
F	**F**

Restricted active clause coverage (RACC)

Definition

For each p in P and each major clause c_i in C_p, choose minor clauses c_j so that c_i determines p. For restricted active clause coverage (RACC) TR has two requirements for each major clause c_i: c_i evaluates to true, and c_i evaluates to false. The values for the minor clauses c_j must be the same for the two values of c_i.

Discussion

- RACC often leads to infeasible test requirements.
- Constraint seems artificial.

Example (RACC)

	a	b	c	$a \wedge (b \vee c)$
1	true	true	true	true
2	true	true	false	true
3	true	false	true	true
4	true	false	false	false
5	false	true	true	false
6	false	true	false	false
7	false	false	true	false
8	false	false	false	false

- Let a be major clause. When does a determine? The *TR*s for RACC for a are satisfied by one of the following pairs of rows (1,5), (2,6), (3,7).
- When does b determine?

Correlated active clause coverage (CACC)

Definition

For each p in P and each major clause c_i in C_p, choose minor clauses c_j so that c_i determines p. For correlated active clause coverage (CACC) TR has two requirements for each major clause c_i: c_i evaluates to true and c_i evaluates to false. The values for the minor clauses c_j must cause p to be true for one value of the major clause c_i and false for the other.

Discussion

- CACC subsumes PC.
- By the definition of determination, c_i and p do not have to evaluate to the same truth value.
- Minor clauses may have different values.

Example (CACC)

	a	b	c	$a \wedge (b \vee c)$
1	true	true	true	true
2	true	true	false	true
3	true	false	true	true
4	true	false	false	false
5	false	true	true	false
6	false	true	false	false
7	false	false	true	false
8	false	false	false	false

- Let a be major clause.
- CACC for a can be satisfied by choosing one test requirement from rows 1-3, and one test requirement from rows 5-7.

In-class exercise

In-class exercise (cont'd)

Inactive clause coverage (ICC)

Definition

For each p in P and each major clause c_i in C_p, choose minor clauses c_j so that c_i does **not** determine p. For inactive clause coverage (ICC) TR has four requirements for each major clause c_i:

1. c_i evaluates to true with p=true
2. c_i evaluates to false with p=true
3. c_i evaluates to true with p=false
4. c_i evaluates to false with p=false

Discussion

- Major clause does not affect the predicate.
- Two variants (general, restricted); correlation not sensible

General inactive clause coverage (GICC)

Definition

For each p in P and each major clause c_i in C_p, choose minor clauses c_j so that c_i does **not** determine p. For general inactive clause coverage (GICC) TR has the four requirements for each major clause specified in ICC. The values for the minor clauses need not be the same for the different valuations of c_i, p.

Discussion

- GICC subsumes predicate coverage.

Restricted inactive clause coverage (RICC)

Definition

For each p in P and each major clause c_i in C_p, choose minor clauses c_j so that c_i does **not** determine p. For restricted inactive clause coverage (RICC) TR has the four requirements for each major clause specified in ICC. The values for the minor clauses must be the same for the different valuations of c_i.

Discussion

- RICC subsumes predicate coverage.

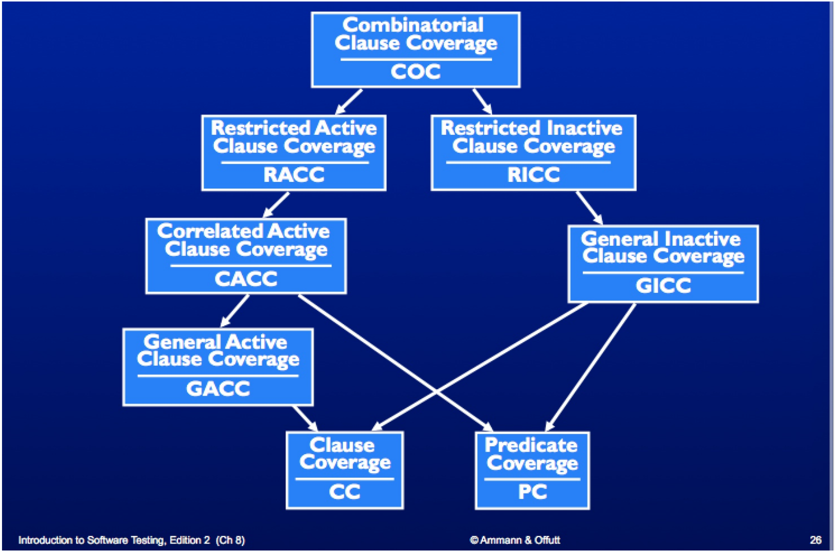
Infeasibility, again

Consider the predicate

$$(x < y \wedge y < z) \vee (z < x)$$

- 3 clauses; setting all three clauses to true is infeasible
- As before: recognizing infeasible test requirements is undecidable

Subsumption



Summary (logic coverage)

- In practice, most predicates have few clauses (< 3).
  - But important exceptions exist (e.g., control software)
- For a single clause, PC suffices.
  For 2 or 3 clauses, CoC remains practical.
- MC/DC and active clauses;
  different interpretations (GACC, RACC, CACC)

References

- AO, Ch. 8.1