6. Test-Adequacy

Assessment Using Control Flow and Data Flow

Andrea Polini

Software Engineering II – Software Testing
MSc in Computer Science
University of Camerino

January 22nd, 2015

What is test adequacy?

Consider a program P developed to satisfy a set of requirements (P,R)
It is necessary to know if the system has been tested throughly
Two different classes of criteria:

- Black-box: based on model and requirements
- White-box: based on code

Example

- **R1**: Input two integers, x, y, from the standard input device
- **R2**: Find and print to the standard output the sum if $x < y$
- **R3**: Find and print to the standard output the product of the two numbers if $x \geq y$

Adequacy criteria push the improvements of test sets

```
begin
  int x,y;
  int product, count;
  input(x,y);
  if (y >= 0) {
    product = 1; count = y;
    while (count > 0) {
      product = product * x;
      count = count - 1;
    }
    output(product);
  }
  else
    output("Input does not match its specification");
}
```

Criteria

C1 : A test set is considered adequate if it tests the program for at least one zero and one nonzero value of each of the two inputs x and y

C2 : A test set is considered adequate if it tests all paths. In case the program contains a loop, then it is adequate to traverse the loop body zero times and once.

It is clearly possible that some criteria could be infeasible given P structure

Criteria based on control flow

Statement coverage

The statement coverage of T with respect to (P,R) is computed as $|S_c|/(|S_e| - |S_i|)$ where S_c is the set of statements covered, S_i the set of unreacheable statements, and S_e the set of statements in the program, that is the coverage domain. T is considered adequate with respect to the statement coverage criterion if the statement coverage of T with respect to (P,R) is 1.

Block coverage

The block coverage of T with respect to (P,R) is computed as $|B_c|/(|B_e| - |B_i|)$ where B_c is the set of blocks covered, B_i the set of unreacheable blocks, and B_e the blocks in the program, that is the block coverage domain. T is considered adequate with respect to the block coverage criterion if the block coverage of T with respect to (P,R) is 1.

Conditions and decisions

Decision Coverage

The decision coverage of T with respect to (P,R) is computed as $|D_c|/(|D_e| - |D_i|)$ where D_c is the set of decisions covered, D_i the set of unfeasible decision, and D_e the set of decision in the program, that is the decision coverage domain. T is considered adequate with respect to the decision coverage criterion if the decision coverage of T with respect to (P,R) is 1.

Condition Coverage

The condition coverage of T with respect to (P,R) is computed as $|C_c|/(|C_e| - |C_i|)$ where C_c is the set of simple conditions covered, D_i the set of unfeasible simple conditions, and C_e is the set of simple conditions in the program, that is the condition coverage domain. T is considered adequate with respect to the decision coverage criterion if the decision coverage of T with respect to (P,R) is 1.

Condition vs. decision coverage

Condition coverage does not guarantee decision coverage

Condition/decision coverage

The condition/decision coverage of T with respect to (P,R) is computed as $(|C_c| + |D_c|)/((|C_e| - |C_i|) + (|D_e| - |D_i|))$ where variable as defined as before. T is considered adequate with respect to the condition/decision coverage criterion if the condition/decision coverage of T with respect to (P,R) is 1.

Multiple Condition Coverage

Multiple condition coverage

The multiple condition coverage of T with respect to (P,R) is computed as $|C_c|/(|C_e| - |C_i|)$ where $|C_c|$ denotes the set of combinations covered, $|C_i|$ denotes the set of infeasible simple combinations, and $|C_e|$ is the total number of combinations in the program. T is considered adequate with respect to the multiple-condition coverage criterion if the multiple-condition coverage of T with respect to (P,R) is 1.