Lecture 4.

Monday, September 16, 2019 1:06 PM

Exaustive Testing

↳ means testing all possible inputs → impossible most of the time

→ Techniques used to reduce the number of inputs

- testing critera group input elements into class
- Select 1 input from each class
- Criteria are used to decide which test input to use

Test Data

→ specification → Black Box Testing (don't know the code just know the specification)

→ implementation → White Box testing (know the code and specifications)

Black Box

+ Check conformance with specification
+ Scales up (different techniques at different levels)
  - → works for function, class, package or system
- depends on specification and the degree of detial
- do not know how much of the system is tested
  - preform an unexpected task

White Box

+ Based on control and data flow criteria
+ it allows confidence about the amount of system tested
- does not scale up
- cannot reveal missing functionalities

Test Model Criterion

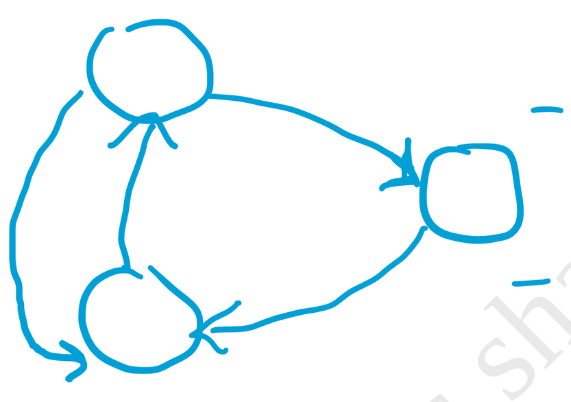
→ Given a Criterion C and model M

- the coverage ratio of a test set T is the proportion of elements in M defined by C covered by test set
- ex. N is the control flow graph function / C is all the statements results in 5/8 statement coverage ratio
- ex. M is set of use case senarios / C is all the senarios results in 12/12 senarios coverage ratio

- A test Criterion → specifies a set of test requirements / objectives
  - → Test requirements must be satisfied in order to obtain adequate test suite
  - → Issue → when applying a criterion on a test model, not all test requirements are feasible
  - → Revised notion of adequacy:
    - → The coverage ratio of a test set T is the proprtion feasible elements in M defined by C covered by the test set
    - → A test set T is said to be adequate for C, when the coverage ratio active 100%. for criterion C

Theoretical Hierarchy of Criteria

- if given C1 & C2, C1 subsumes C2 if any C1-adequate test is also C2-adequate BUT C1 is NOT a subset of C2.
- ex.



- if all nodes are tested that doesn't mean that all edges are tested
- if all edges are tested, then all nodes are tested.

Ways to use test Criteria

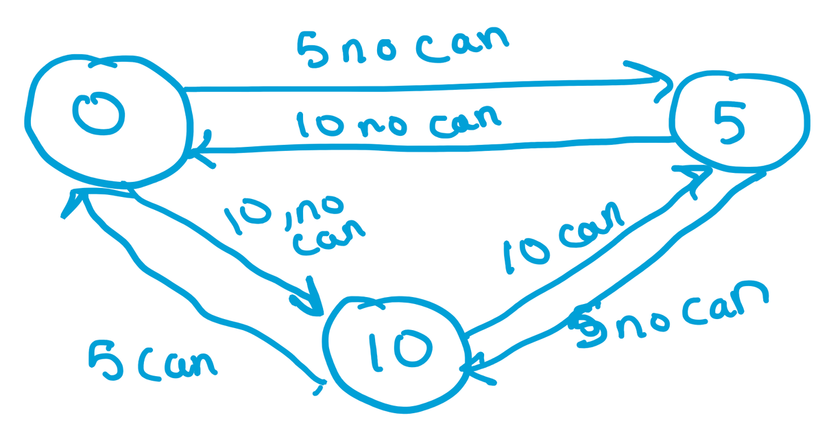
- Generate test values / cases to satisfy the criterion
  - Criterion == Selection Criterion
  - Needs a tool that generates values to satisfy criterion
- Evaluate Coverage achived by externally generated test values/cases
  - Criterion == coverage criterion
  - Needs tool, a recognizer, that automatically decides whether a set of values satisfies a criteration
  - easier to build than a generator

Using Test Adequacy Criterion

Steps:
1. Choose test model
2. Select a test criterion
3. Identify objectives
4. Create test case specifications
5. Identify test data/input
6. Identify Oracle

Ex.
1. State machine
2. All-transitions
3. Test case 1 will excersize transions t1, t2, t8 ...
   Test case 2 will excersize transion t1, t2, t8 ...
4. To execute test case 1, need to input 10.
   To execute test case 2, need to input 20
6. What do you feel you need to check during and at end of the execution of test case 1, case 2 ... and what should you expect

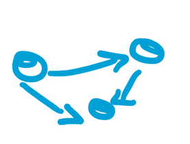
ex.



1. choose model
2. Select test criterion: all transions
3. identify objective:
   0→5 5→10 10→0
   5→0 10→5 0→10
4. Create test case specifications
   T1 = [0, 5, 10]
   T2 = [0, 10, 0]
   T3 = [0, 10, 5, 0]
5. Identify test data
   T1 = [5, 5]
   T2 = [10, 5]
   T3 = [10, 10, 10]
6. Orcales
   T1 = machine has 10, no can
   T2 = machine has 0, 1 can
   T3 = machine has 0, 2 can

1. Generate functional tests from requirements of design to try every function
   - use functional / black box adequacy (generator)
2. Check the structural coverage after functional tests are all verified to be successful
   - use a structural / white box coverage (recognizer)
3. Where the structural coverage fails, generate functional tests that include additional coverage

Test Criteria Based on Structure

- Graphs 
- Logical Expressions (not X or not Y) and A and B
- Input Domain Characterization A: {0, 1, 15} B: {...} C{...}
- Syntax Structures if (x>y){ z = x-y }else{ ...

This study resource was shared via CourseHero.com
https://www.coursehero.com/file/46209854/Lecture-4pdf/