Evaluating Testing Methods by Delivered Reliability

Frankl, Hamlet, Littlewood, Strigini
IEEE TOSE Aug 98

Partition Testing

◆ What is a partition?
◆ How do you choose the partitions?
◆ What is a good partition?

Fault Detection Probability

◆ Probability of a testing methodology finding a fault (if it existed)

Operational Profile

◆ What is an operational profile?
◆ How do you construct an operational profile?
◆ What are some difficulties with operational profiles?

Tests, Specifications, meets

◆ Test or test case
  ◆ single value of program input
  ◆ functional program - one input produces an output
◆ Specification - S
  ◆ set of input-output pairs
◆ Program meets specification
  ◆ iff for all x in spec, actual output matches spec output

Failures, failure points

◆ “A program P with specification S fails on input x iff P does not meet S at x.”
◆ “the event is called a failure”
◆ “the input responsible is a failure point”
◆ “The program’s failure set is the collection of all failure points”
Faults

- “the part of a source program that causes a failure” (IEEE glossary)
- Not well defined
- usually multiple ways to correct a failure

Fixing failures

- “assume that all testers, upon observing a test failure, choose fixes that eliminate exactly the same failure region”
- Why is this assumption important?

Operational Testing

- Q - probability distribution over input domain
  - Q:D -> [0,1] and Σ Q(t) = 1
- labels φ-phi(failure) and σ-sigma(success)
- δ(t) = 1 if φ and 0 if σ
- Θ - failure probability for a randomly drawn point is Σ Qδ

Cost of testing

- Fixed budget
  - T_d debug tests or T_r operational tests
  - T_d log(1-d) < T_k log(1-q)
  - for small q and d
    - dT_d > qT_r
  - Why does he assume that T_d > T_r?

3.3 single failure region, with sub

- Debug with subdomains
  - E(Θ) = q(1-d)^T
- Operational
  - E(Θ) = q(1-q)^T

Triangle Code

```cpp
cin >> a >> b >> c;
if (a >= b || a >= c || b >= c) type = “isosceles”;  
if (a == b & & b == c) type = “equilateral”;  
if (a+b+c) type = “not a triangle”;  
if (a<=0||b<=0||c<=0) type = “bad inputs”;  
cout<< type
```
Triangle subdomains

- subdomains

- \((1-d_i^1)T_1 \cdots (1-d_i^n)T_n < (1-q)^T\)

Class discussion

- What would be better subdomains

Multiple Failure, debug w/o sub

- \(E(\Theta) = \sum q_i (1-d_i)^T\)

- \(E(\Theta) = \sum q_i (1-q_i)^T\)

When is ….

- A. debug better than operational?

- B. vice-versa?

- C. reliability worse but detection better?