Structural Testing Review

Chapter 11

The big question

- **When should testing stop?**

Possible stopping criteria

- Run out of time
- Continued testing causes no new failures
- Continued testing reveals no new faults
- Cannot think of any new test cases
- Reach a point of diminishing returns
- Mandated coverage has been attained
- All faults have been removed

Functional testing problems

- **What are the problems with functional testing?**

Functional testing problems – 2

- Functional testing methods may produce test suites with
  - **Serious gaps**
  - **Lots of redundancy**

Measuring gaps and redundancy

- How do know how big are the problems of gaps and redundancy?

Measuring gaps and redundancy – 2

- Structural testing analysis makes it possible to measure the extent of these problems

– graph paths
Triangle program – nominal boundary value analysis
– worst case boundary value analysis

Paths	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11
Nominal	**3**	**3**	**1**	**3**	**1**	**3**	**1**	**0**	**0**	**0**	**0**
Worst case	**5**	**12**	**6**	**11**	**6**	**12**	**7**	**17**	**18**	**19**	**12**

Measurement

- What do we need to be able to measure?

Measurement – 2

- Need a standard of measurement
  - **A metric**

Structural metric

- **What is a program structural metric?**

Structural metric – 2

- A program structural metric is a standard of measurement for the structure of a program

Structural metric – 3

- **What are the components of a structural metric?**

Structural metric – 2

- A structural metric **S** identifies **s** coverage elements in the unit under test
- A testing method **M** produces **m** test cases
- When the **m** test cases run, they visit **c** coverage elements
- By comparing **c** and **s** we have a measurement of how good is our set of test cases

Metric definitions

- **What definitions are used for structural metrics for a method M with respect to a metric S?**

Metric definitions – 2

- Coverage
- Redundancy
- Net redundancy

Coverage definition

- What the definition of coverage?

Coverage definition – 2

- **Coverage** of method **M** with respect to metric **S** is

$$C\ (M, S) = c\ /\ s$$

  - **Deals with gaps**

  - **What does the ratio tell us?**

Coverage definition – 3

- **Coverage** of method **M** with respect to metric **S** is

$$C\ (M, S) = c\ /\ s$$

  - **Deals with gaps**
    - **a ratio < 1 means there are gaps**

Redundancy definition

- **What the definition of redundancy?**

Redundancy definition – 2

- **Redundancy** of method **M** with respect to metric **S** is

$$R(M, S) = m / s$$

  - **Deals with absolute redundancy**

  - **What does the ratio tell us?**

Redundancy definition – 3

- **Redundancy** of method **M** with respect to metric **S** is

  R (M, S) = m / s

  - **Deals with absolute redundancy**
    - **Ratio of 1 is best**
    - **Larger values imply more redundancy**
    - **Smaller values imply gaps**
    - **Not so useful**
      - **WHY?**

Redundancy definition – 3

- **Redundancy** of method **M** with respect to metric **S** is

R (M, S) = m / s

  - **Deals with absolute redundancy**
    - **Ratio of 1 is best**
    - **Larger values imply more redundancy**
    - **Smaller values imply gaps**
    - **Not so useful**
      - Could have massive redundancy with massive gaps giving a small ratio

Net redundancy definition

- **What the definition of net redundancy?**

Net redundancy definition – 2

- **Net redundancy** of method **M** with respect to metric **S** is

NR (M, S) = m / c

  - **Deals with relative redundancy**
  - **What does the ratio tell us?**

Net redundancy definition – 3

- **Net redundancy** of method **M** with respect to metric **S** is

$$NR(M, S) = m / c$$

  - **Deals with relative redundancy**
    - **best is 1**
    - **Very useful, shows the redundancy of what is tested**

Metric values for triangle program

Method	m	c	s	C(M,S)	R(M,S)	NR(M,S)
Boundary Value	15	7	11	0.64	1.36	2.14
Worst Case Analysis	125	11	11	1.00	11.36	11.36
WN ECT	4	4	11	0.36	0.36	1.00
Decision Table	8	8	11	0.72	0.72	1.00

Metric values for commission program

Method	m	c	s	C(M,S)	R(M,S)
Output BVA	25	11	11	1	2.27
Decision table	2	11	11	1	0.27
DD-path	25	11	11	1	2.27
DU-path	25	33	33	1	0.76
Slice	25	40	40	1	0.63

Coverage example

- TEX (Donald Knuth) and AWK (Aho, Weinberger, Kernigan) are widely used programs with comprehensive functional test suites

- Coverage analysis shows the following percentage of items covered

System	Segment	Branch	P-use	C-use
TEX	85%	72%	53%	48%
AWK	70%	59%	48%	55%

Coverage usefulness

- 100% coverage is never a guarantee of bug-free software

- **What can coverage reports give us?**

Coverage usefulness – 2

- Coverage reports can
  - **Point out inadequate test suites**
  - **Suggest the presence of surprises, such as blind spots in the test design**
  - **Help identify parts of the implementation that require structural testing**

Coverage usefulness – 3

- All possible coverage elements **s** is very big
  - **On what basis do we select appropriate subsets?**

Coverage usefulness – 3

- Can try by selecting appropriate paths
  - **By fault type**
  - **By risk / fear**

Is 100% coverage possible?

- **Can you suggest cases that prevent 100% coverage?**

Is 100% coverage possible? – 2

- Lazy (short-circuit) evaluation
  - `a && b && c`
- Mutually exclusive conditions
  - `(x > 2) || (x < 10)`
- Redundant predicates
  - ```
    if (x == 0) do1; else do2;
    if (x != 0) do3; else do4;
    ```
- Dead code
- "This should never happen"
    ```

How to measure coverage?

- **Can you suggest ways to measure coverage; i.e. how do you determine c?**

How to measure coverage? – 2

- The source code is instrumented
- Depending on the code coverage model, code that writes to a trace file is inserted in every branch, statement etc.
- Most commercial tools measure segment and branch coverage

Questions about Coverage

- Is 100% coverage the same as exhaustive testing?
- Are branch and path coverage the same?
- Can path coverage be achieved?
- Is every path in a control flow graph testable?
- Is less than 100% coverage acceptable?
- Can I trust a test suite without measuring coverage?

Coverage counter-example vending machine

```
void give_change(int price, deposit) {
  int n_100, n_25, n_10, n_5, change_due;
  if (deposit <= price) { change_due = 0; }
  else {
    change_due = deposit - price;
    n_100      = change_due / 100;
    change_due = change_due - n_100*100;
    n_25       = change_due / 25;
    change_due = change_due - n_25*25;
    n_10       = change_due / 10;
    change_due = change_due - n_10*10;
    n_5        = change_due / 10; // Cut-and-paste bug
  }
}
```

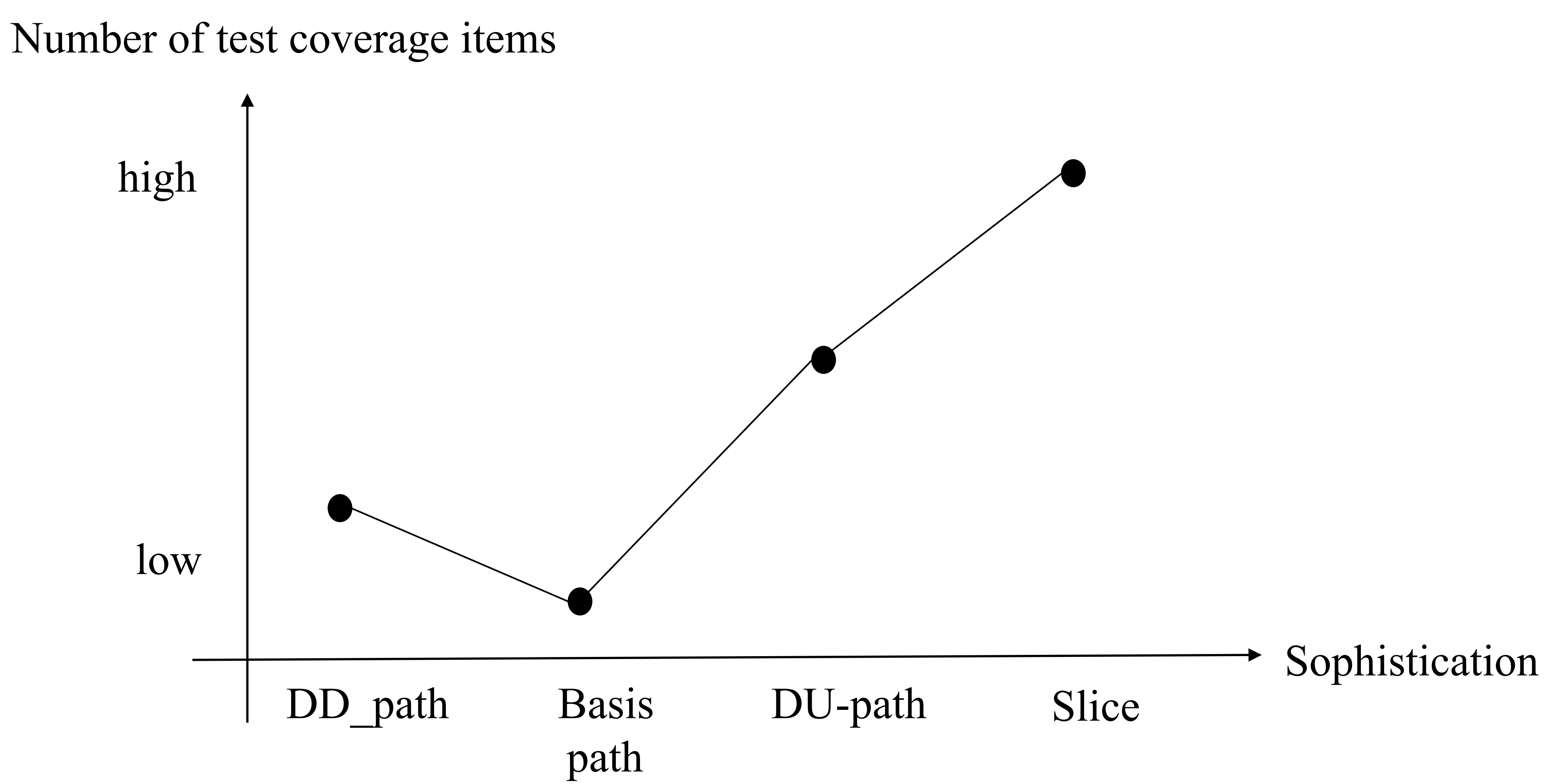
Cannot guarantee path testing will use revealing test values for deposit and price

Coverage counter-example aircraft control

```
void flight_control_event_handler (event e) {
  switch(e)
  { ...
    case RAISE_LANDING_GEAR:
      landing_gear_motor ( turn_on_until_raised );
      break;
    ...
  }
}
```

Can you find the bug?
Will any path test find the bug?
What can correct the bug?

Trend line test coverage of items



Trend line test method effort