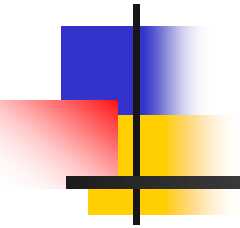


Software Testing & Quality Assurance

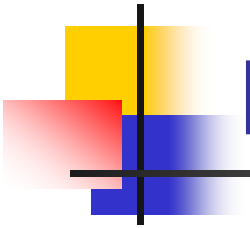
Structural Testing Review

- Structural Metrics
- Test Coverage Metrics
- Coverage Usefulness & Measurement

Credits & Readings

The material included in these slides are mostly adopted from the following books:

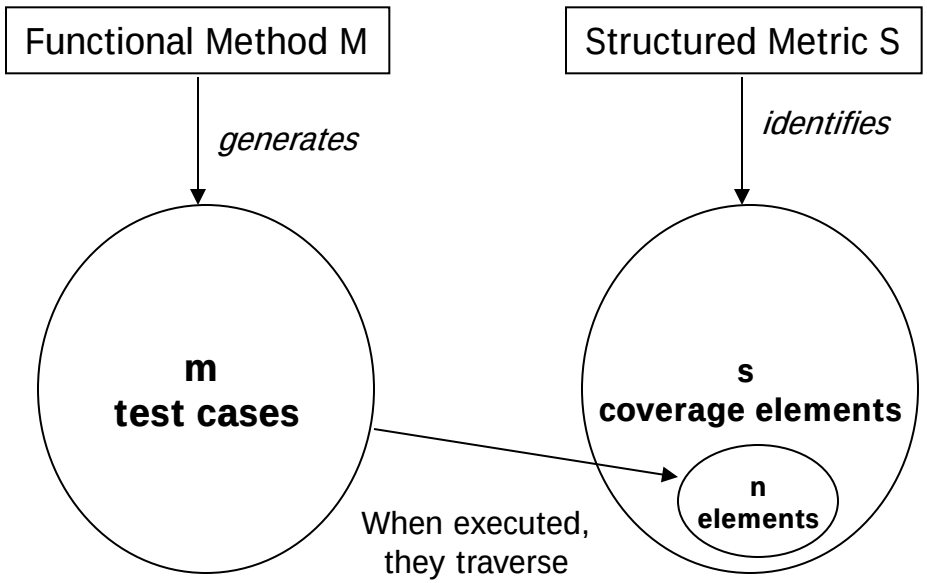
- *Software Testing: A Craftsman's Approach*, by Paul Jorgensen, CRC PRESS, third edition, ISBN: 0-8493-7475-8
- Cem Kaner, Jack Falk, Hung Q. Nguyen, *"Testing Computer Software"* Wiley (see also http://www.testingeducation.org/)
- Paul Ammann and Jeff Offutt, *"Introduction to Software Testing"*, Cambridge University Press
- Beizer, Boris, "*Software Testing and Quality Assurance"*, Van Nostrand Reinhold
- Glen Myers, *"The Art of Software Testing"*



Measuring Gaps & Redundancy

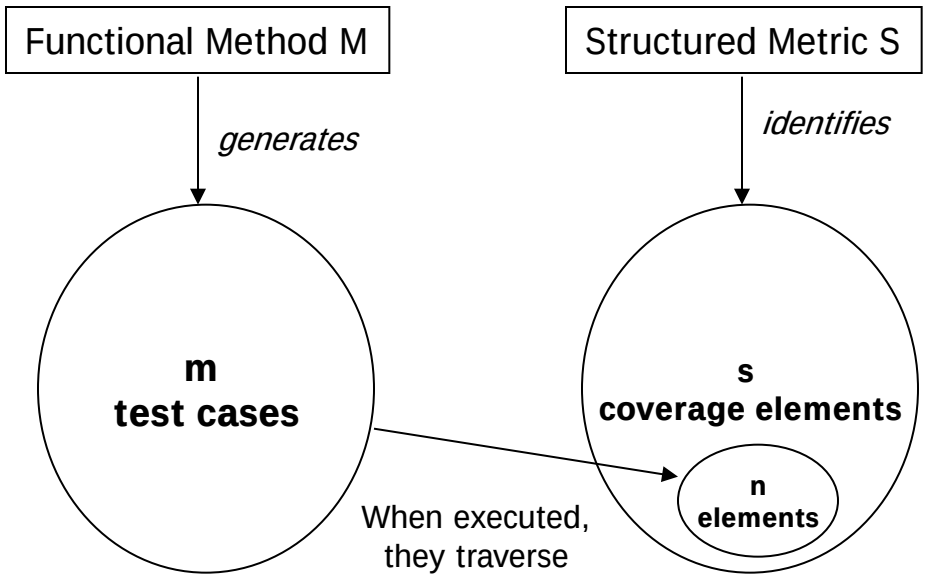
- We have seen that functional testing methods may produce test suites with serious gaps and a lot of redundancy
- Structural testing analysis allows to measure the extent of these problems
- Recall that the structural metrics are always expressed in terms of something countable, such as
  - The number of program paths
  - The number of decision-to-decision paths
  - The number of slices

Evaluation of Functional Testing Based on Structural Metrics



- We assume that
  - A functional testing method **M** produces **m** test cases which are tracked with respect to
  - a structural metric **S** that identifies **s** coverage elements in the unit under test
- When the **m** test cases are executed, they traverse **n** (of the s) coverage elements

Metrics for Functional Testing Effectiveness



All three metrics provide a quantitative way to evaluate the effectiveness of any functional testing method with respect to a structural metric

- *Coverage* of method M with respect to metric S as
  - **C(M,S) = n/s**
  - *When it is less than 1, it means that that there are gaps in the coverage with respect to that metric*
- *Redundancy* of method M with respect to metric S as
  - **R(M,S) = m/s**
  - *The bigger it is , the bigger the redundancy of test cases*
- *Net redundancy* of method M with respect to metric S as
  - **NR(M,S) = m/n**
  - *It refers to things actually traversed, not to the total space of things to be traversed (more useful)*
- Best possible value for these metrics is 1
- Note: when C(M,S) = 1, algebra forces R(M,S) = NR(M,S)

Metric Values for Triangle

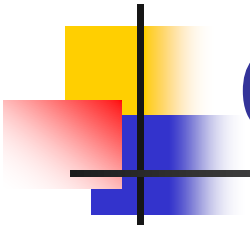
- The implementation has exactly 11 feasible paths shown in Table 11.1 of our textbook
- Table 11.2 shows the test cases generated using Boundary Value technique
  - Paths p1, p2, p3, p4, p5, p6 and p7 are covered and paths p8, p9, p10, p11 are missed
  - If Worst-Case Boundary Value testing was used, it would yield 125 test cases, providing full path coverage, but high redundancy

Method	m	n	s	C(M,S) **n/s**	R(M,S) **m/s**	NR(M,S) **m/n**
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

More Comparisons

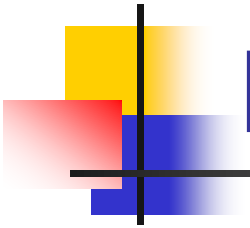
Method	m	n	s	C(M,S)	R(M,S)	NR(M,S)
Triangle Program						
BVA	15	7	11	0.64	1.36	2.14
Worst Case BVA	125	11	11	1.00	11.36	11.36
Commission Program						
Output BVA	25	11	11	1.00	2.27	2.27
Decision Table	3	11	11	1.00	0.27	0.27
DD-Path	25	11	11	1.00	2.27	2.27
DU-Path	25	33	33	1.00	0.76	0.76
Slices	25	40	40	1.00	0.63	0.63

Adopted with permission from Software Testing: A Craftsman's Approach, by Paul Jorgensen, CRC PRESS, third edition.



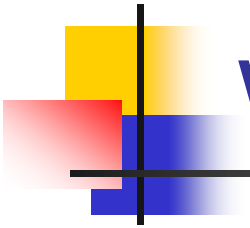
Coverage Usefulness

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



How to Measure Coverage?

- 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



When Should We Stop Testing?

- When you run out of time?
- When continued testing reveals no new faults?
- When you cannot think of any new test cases?
- When mandated coverage has been attained?
- When all faults have been removed?