Test Coverage & Adequacy

- How much testing is enough?
- When to stop testing
- Test data selection criteria
- Test data adequacy criteria
  - Stopping rule
  - Degree of adequacy
- Test coverage criteria
- Objective measurement of test quality

Preliminaries

- Test data selection
  - What test cases
- Test data adequacy criteria
  - When to stop testing
- Examples
  - Statement Coverage
  - Branch coverage
  - Def-use coverage
  - Path coverage

Goodenough & Gerhart ['75]

- What is a software test adequacy criterion
  - Predicate that defines "what properties of a program must be exercised to constitute a thorough test", i.e., one whose successful execution implies no errors in a tested program

Goodenough & Gerhart ['75]

- Reliability requirement
  - "Test criterion always produces consistent test results"
  - If a program tested successfully on one test set that satisfies the criterion, then the program also tested successfully on all test sets that satisfy the criterion
- Validity requirement
  - "Test always produces a meaningful result"
  - For every error in a program, there exists a test set that satisfies the criterion and is capable of revealing the error
- There is no computable criterion that satisfies the above requirements

Uses of test adequacy

- Objectives of testing
- In terms that can be measured
  - For example branch coverage
- Two levels of testing
  - First as a stopping rule
  - Then as a guideline for additional test cases

Categories of Criteria

- Specification based
  - All-combination criterion
    - choices
  - Each-choice-used criterion
- Program based
  - Statement
  - Branch
- Note that in both the above types, the correctness of the output must be checked against the specifications

Others

- Random testing
- Statistical testing
- Interface based

Classification according to underlying testing approach

- Structural testing
  - Coverage of a particular set of elements in the structure of the program
- Fault-based testing
  - Some measurement of the fault detecting ability of test sets
- Error-based testing
  - Check on some error-prone points

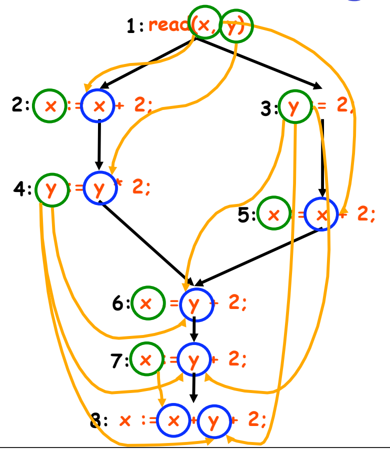
Structural Testing

- Program-based structural testing
  - Control-flow based adequacy criteria
    - Statement coverage
    - Branch coverage
    - Path coverage
      - Length-i path coverage
    - Cyclomatic number criterion
      - Set of v independent paths, where v = e -n + 1
    - Multiple condition coverage
      - All possible combinations of truth values of predicates
  - Data-flow based adequacy criteria

Structural Testing

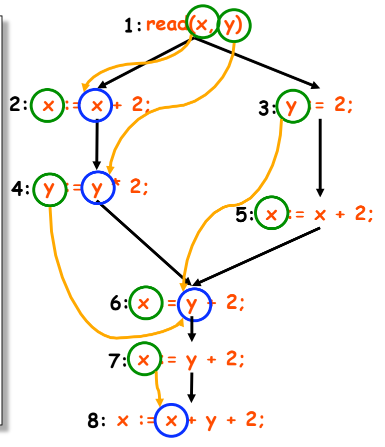
- Data-flow based adequacy criteria
  - All definitions criterion
    - Each definition to some *reachable* use
  - All uses criterion
    - Definition to each reachable use
  - All def-use criterion
    - Each definition to each reachable use

Data-flow Testing



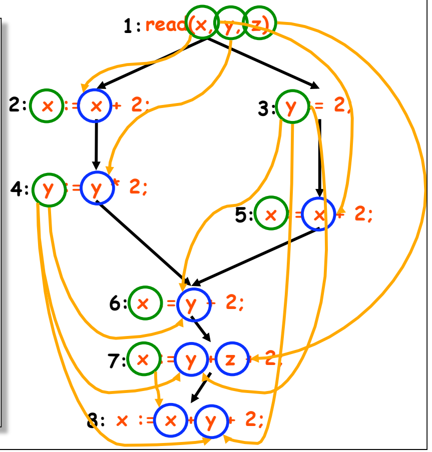
All Definitions Criterion

- A set P of execution paths satisfies the all-definitions criterion iff
  - for all definition occurrences of a variable x such that
    - there is a use of x, which is feasibly reachable from that definition,
  - there is at least one path p in P such that
    - p includes a subpath through which the definition of x reaches some use occurrence of x

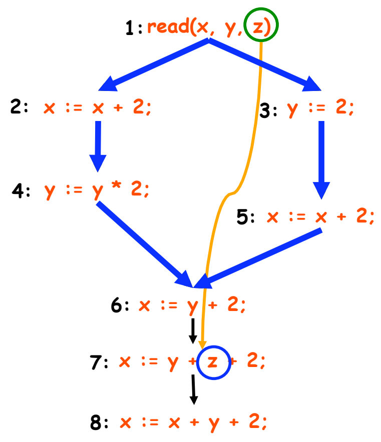


All Uses Criterion

- A set P of execution paths satisfies the all-uses criterion iff
  - for all definition occurrences of a variable x and all use occurrences of x,
    - that the definition feasibly reaches,
  - there is at least one path p in P such that
    - p includes a subpath through which that definition reaches the use



All Uses Criterion

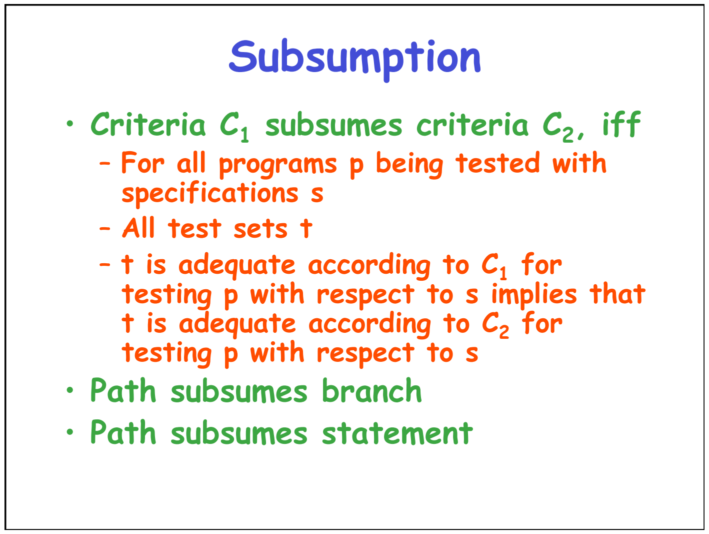


All DU-paths criterion

- A set P of execution paths satisfies the all-DU paths criterion iff
  - for all definitions of a variable x and all paths q through which that definition reaches a use of x,
  - there is at least one path p in P such that
    - q is a subpath of p and q is cycle-free

Fault-based Adequacy

- Error seeding
  - Introducing artificial faults to estimate the actual number of faults
- Program mutation testing
  - Distinguishing between original and *mutants*
    - Competent programmer assumption
      - Mutants are close to the program
    - Coupling effect assumption
      - Simple and complex errors are coupled



Subsumption

- Criteria C_1 subsumes criteria C_2, iff
  - For all programs p being tested with specifications s
  - All test sets t
  - t is adequate according to C_1 for testing p with respect to s implies that t is adequate according to C_2 for testing p with respect to s
- Path subsumes branch
- Path subsumes statement