Path Testing – Creating Test Cases

Chapter 9

CFG question

- **What is the control flow graph – DD-path graph for the following?**

```
if a < b then c = a+b ; d = a*b
         else c = a*b ; d = a+b
if c < d then x = a+c ; y = b+d
         else x = a*c ; y = b*d
```

Creating a test case – key question

- **What is the key question that needs to be answered to be able to create a test for a path?**

Create a test case – key question – 2

- The key question is:
  - **How to make the path execute, if possible.**
    - Generate input data that satisfies all the conditions on the path.

Create a test case – key items

- **What are the key items you need to generate a test case for a path?**

Create a test case – key items – 2

- Key items needed to generate a test case
  - Input vector
  - Predicate
  - Path predicate
  - Predicate interpretation
  - Path predicate expression
  - Create test input from path predicate expression

Input Vector

- **What is an input vector?**

Input Vector – 2

- An input vector is a collection of all data entities read by the routine whose values must be fixed prior to entering the routine.

Input Vector – 3

- **What are the members of an input vector?**

Input Vector – 4

- Members of an input vector can be as follows.
  - Input arguments to the routine
  - Global variables and constants
  - Files
  - Contents of registers (in Assembly language programming)
  - Network connections
  - Timers

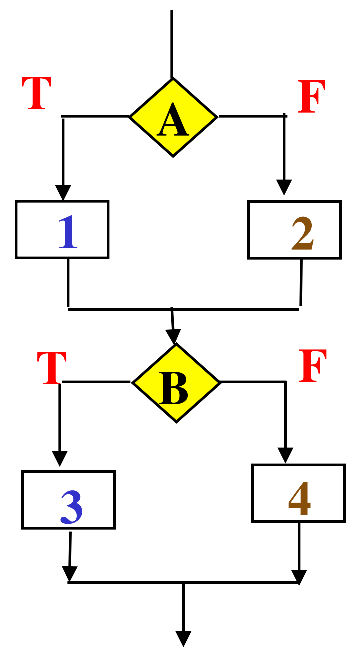
Predicate

- **What is a predicate?**

Predicate – 2

- A predicate is a logical function evaluated at a decision point.
  - Example
    - In the following each of **a < b** and **c < d** are predicates

if a < b then c = a+b ; d = a*b
else c = a*b ; d = a+b
if c < d then x = a+c ; y = b+d
else x = a*c ; y = b*d



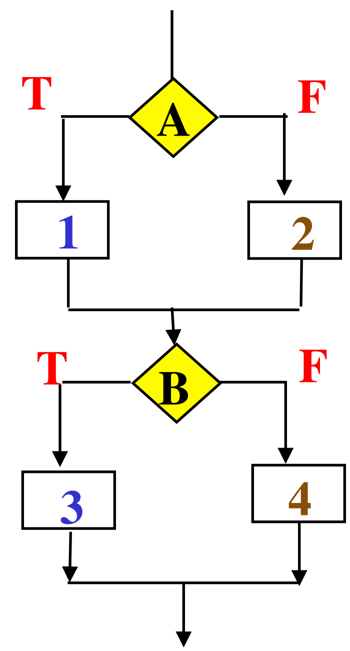
Path predicate

- **What is a path predicate?**

Path predicate – 2

- A path predicate is the set of predicates associated with a path.
  - Example
    - In the following **a < b = true & c < d = false** is a path predicate

if **a < b** **then** **c = a+b ; d = a*b**
else **c = a*b ; d = a+b**
if **c < d** **then** **x = a+c ; y = b+d**
else **x = a*c ; y = b*d**



Predicate Interpretation

- A path predicate may contain local variables.
- Local variables play no role in selecting inputs that force a path to execute.
- Local variables can be eliminated with **symbolic execution**.
  - Symbolically substituting operations along a path in order to express the predicate solely in terms of the input vector and a constant vector.
- A predicate may have different interpretations depending on how control reaches the predicate.

Path Predicate Expression

- An interpreted path predicate is called a path predicate expression.
- A path predicate expression has the following attributes.
  - It has no local variables.
  - It is a set of constraints in terms of the input vector, and, maybe, constants.
  - Path forcing inputs can be generated by solving the constraints.
  - If a path predicate expression has no solution, the path is infeasible.

Path Predicate Generating Input Values

if $a < b$ **then** $c = a+b$; $d = a*b$
else $c = a*b$; $d = a+b$
if $c < d$ **then** $x = a+c$; $y = b+d$
else $x = a*c$; $y = b*d$

- Path predicate: **a < b = true & c < d = false**
- Substitute for c and d:
  a < b = true & a + b < a * b = false
  $\rightarrow$ **a < b & a + b ≥ a * b**
- Solve for a and b: **a = 0 & b = 1**
  Solutions are not unique
- We have a feasible path, since a solution exists.
- Can have infeasible paths, if there is no solution to the constraints

Can have decision table

	A1B3	A1B4	A2B3	A2B4
A < B	T	T	F	F
C < D	T	F	T	F
A value	2	0	1	5
B value	5	1	0	2

Paths **A1B3** and **A2B4** give statement coverage
Or paths **A1B4** and **A2B3** give statement coverage

Selecting paths

- A program unit may contain a large number of paths.
  - Path selection becomes a problem. Some selected paths may be infeasible.
  - Apply a path selection strategy:
    - Select as many short paths as possible.
    - Choose longer paths.
  - Make an effort to write program text with fewer or no infeasible paths.