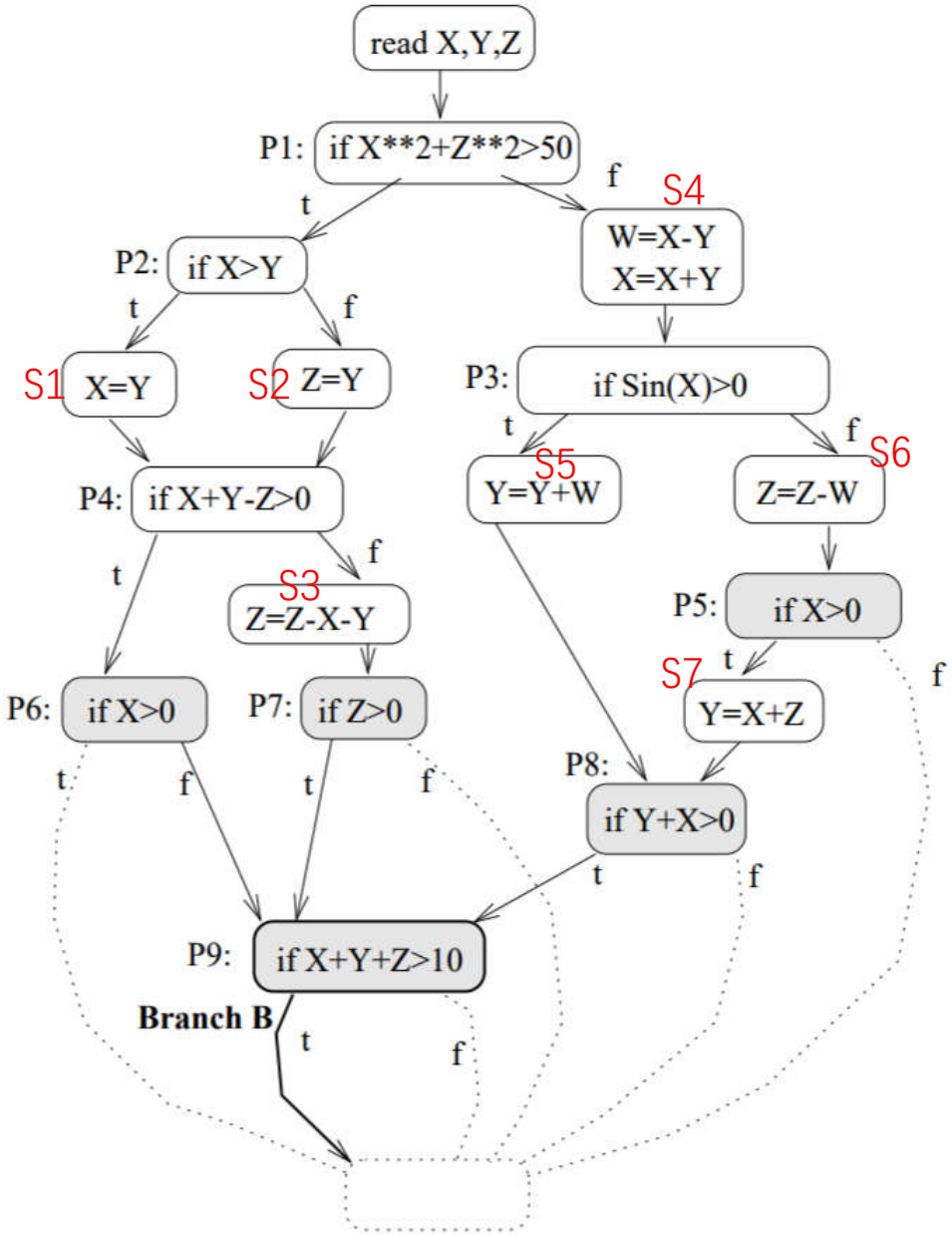
Software Testing and Reliability

Xiaoyuan Xie 谢晓园
xxie@whu.edu.cn
计算机学院E301

Lecture 7

White-box Testing (control-flow coverage)



Statement coverage?

Branch coverage?

Path coverage?

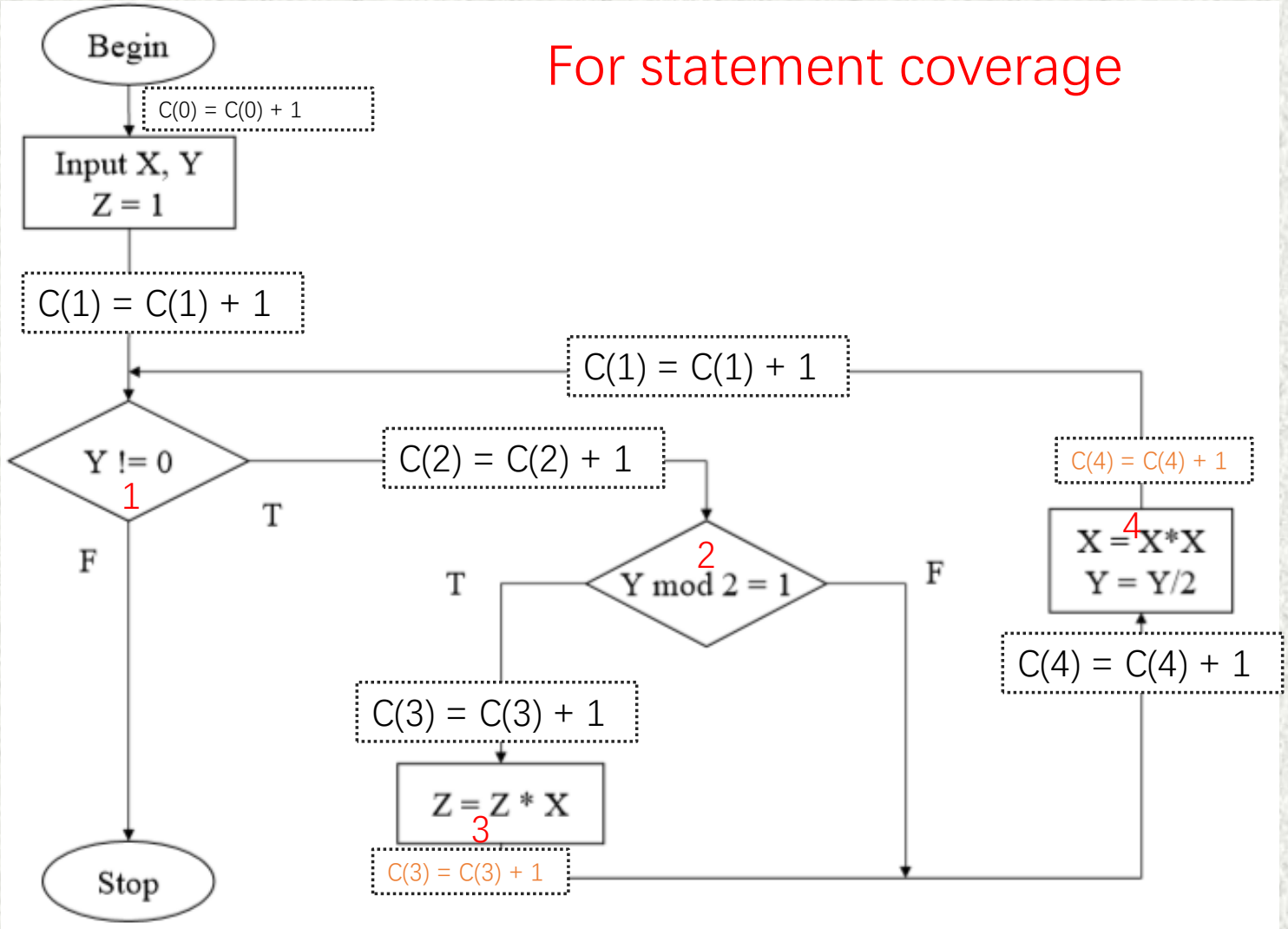
If each branch has statements, then branch coverage has no difference from statement coverage

How do we know that an item has been executed?

- By program instrumentation
  -- Insert some printing statements to trace the program execution

Can be done either after or in parallel with compilation

For statement coverage



White-box Testing (Subdomains)

Example

```
        begin
s1      read(x,y);
s2      w = x + y;
s3      if  x + y >= 2  then
s4                          z = x – (3*y);
                        else
s5                          z = 2*x – 5*y;
s6      if  z + w > 3   then
s7                          z = f(x,y,z);
                        else
s8                          z = g(x,y,z);
s9      write(z);
        end.
```

Two input variables mean the 2-dimensioal input domain

Example

```
        begin
s1      read(x,y);
s2      w = x + y;
s3      if  x + y >= 2  then
s4                      z = x – (3*y);
                      else
s5                      z = 2*x – 5*y;
s6      if  z + w > 3   then
s7                      z = f(x,y,z);
                      else
s8                      z = g(x,y,z);
s9      write(z);
        end.
```

Require *predicate interpretation*

Sample Program

```
        begin
s1      read(x,y);
s2      w = x + y;
s3      if  x + y >= 2  then
s4                        z = x – (3*y);
                        else
s5                        z = 2*x – 5*y;
s6      if  z + w > 3   then
s7                        z = f(x,y,z);
                        else
s8                        z = g(x,y,z);
s9      write(z);
        end.
```

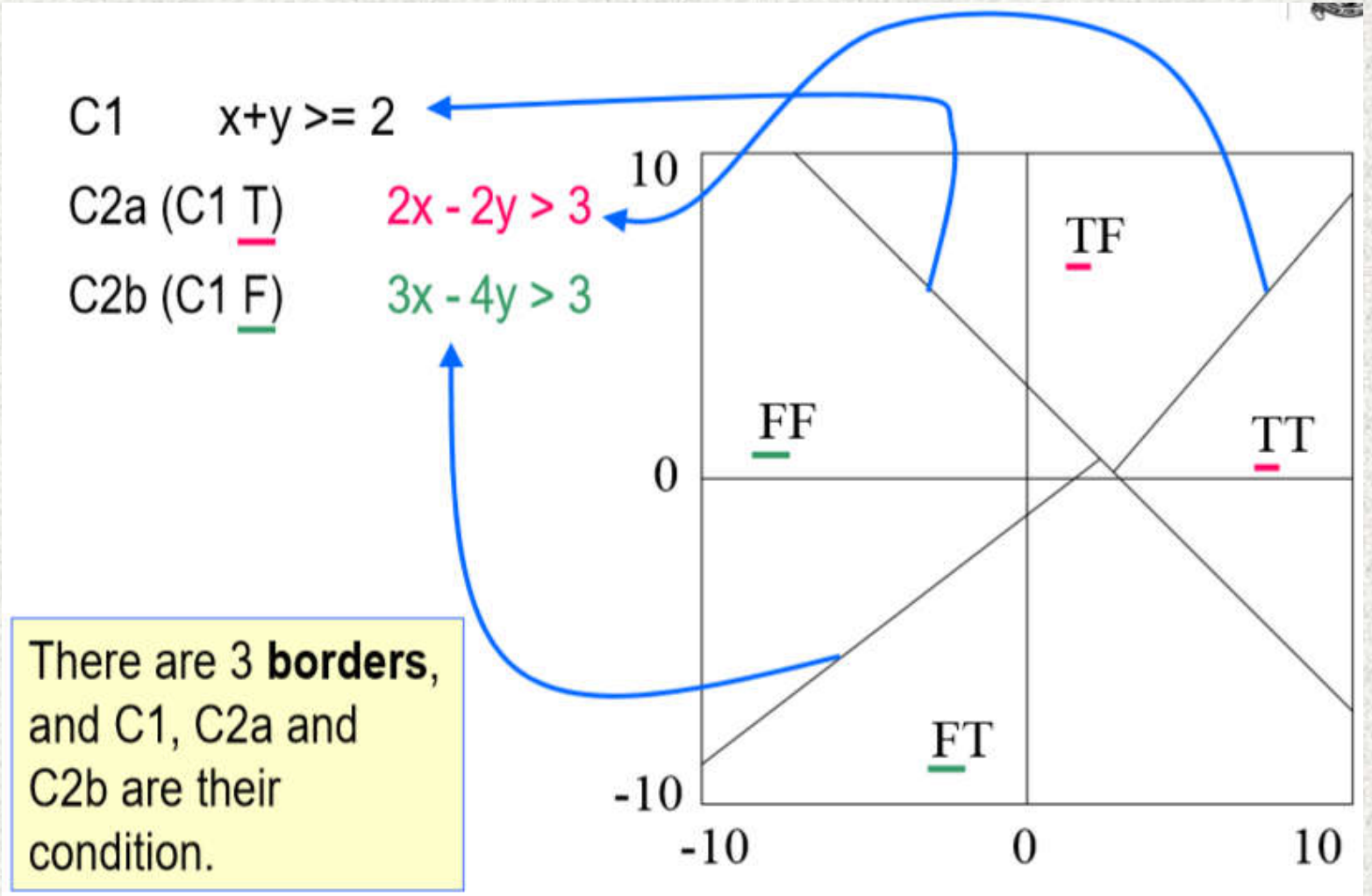
If x + y >= 2, then convert z + w > 3 to 2x - 2y > 3.

Sample Program (continued)

```
        begin
s1      read(x,y);
s2      w = x + y;
s3      if  x + y >= 2  then
s4                          z = x – (3*y);
                        else
s5                          z = 2*x – 5*y;
s6      if  z + w > 3   then
s7                          z = f(x,y,z);
                        else
s8                          z = g(x,y,z);
s9      write(z);
        end.
```

If x + y < 2, then convert z + w > 3 to 3x - 4y > 3.

Sample Program (continued)



There are 3 **borders**, and C1, C2a and C2b are their condition.

Subdomain and Function

- Partition the input domain into a set of disjoint subdomains, each of which corresponds to an executable path
- Consider the following software specification

```
INPUT A
IF A > 25 THEN B = 3A+1
ELSE B = 4A – 3
OUTPUT B
```

Subdomain and Function (continued)

- Canonical Representation of a program

  $\{ (D_1, f_1), (D_2, f_2), \ldots (D_i, f_i), \ldots \}$

  where D_i is the i^{th} subdomain;

  f_i is the computation function of D_i.

- For the program in the previous slide, we will have the following subdomains and functions:

```
INPUT A
IF A > 25 THEN B = 3A+1
ELSE B = 4A – 3
OUTPUT B
```

D_1 : A > 25
f_1 : 3A + 1
D_2 : A <= 25
f_2 : 4A - 3

Exercise

Refer to the program in Slide 4.

- Tasks
    1. Calculate the total number of subdomains
    2. Define the subdomains and functions

    Can you see that each subdomain consists of at least one border condition?

White-box Testing (Data Flow Coverage)

Data flow coverage

- Generate test cases according to the pattern of data usage inside the software
- Three types of data usage
    - -- Define
        - -- The variable is given a value, for example, **X**= 5;
    - -- Predicate use
        - -- The variable's value is used to decide the TRUE or FALSE outcome of a predicate, for example, if(**X**>3)
    - -- Computational use
        - -- The variable's value is used for computation, for example, Y = **X**+ 1

Data-flow coverage (continued)

- Notation
  - -- definition : def
  - -- predicate-use : p-use
  - -- computational-use : c-use
- One pattern of data usage
  - -- *Define* a variable, and then *use* its value

Def-Use pair coverage criterion

- Aims to cover all pairs of def-use for all variables
- Steps
    1. Find all the def-use pairs for all variables
    2. Generate test cases to execute each of these pairs at lease once

Example for Def-Use pair coverage

Example:

S1: X = ……

S2: WHILE ….. DO { ……

S3: X= X* I}

S4: Y = X

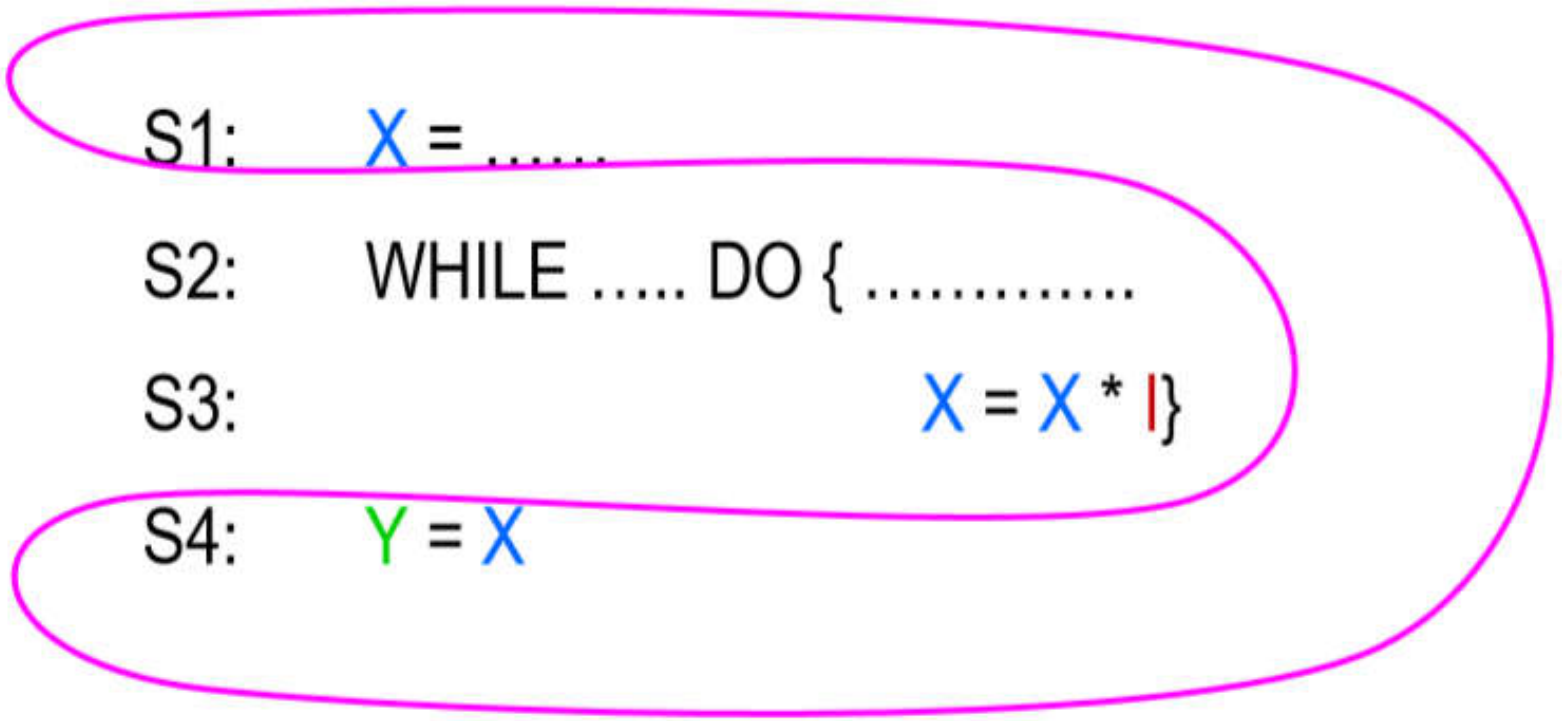
What are the def-use pairs for variable X, Iand Y?

Example for Def-Use pair coverage (continued)

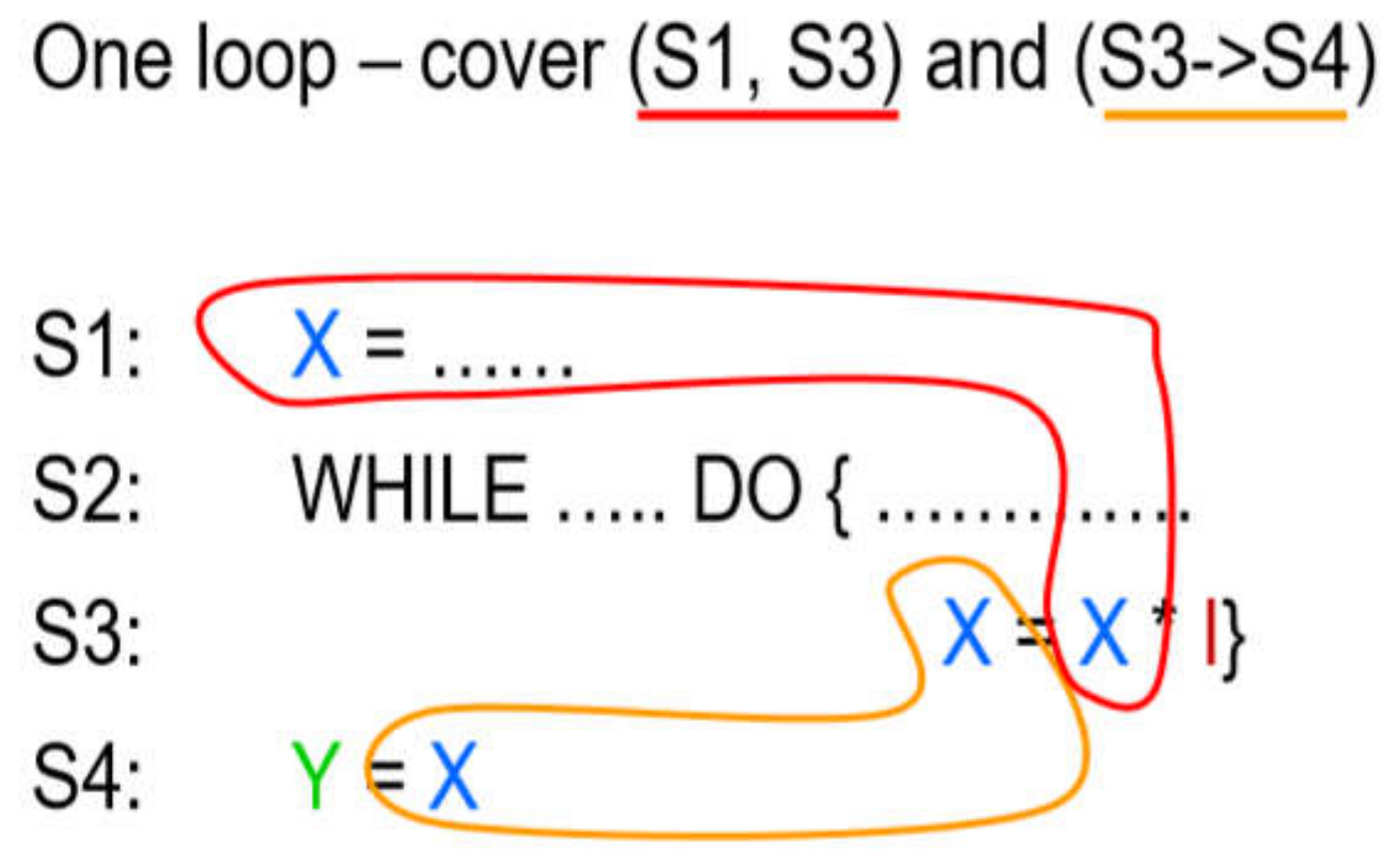
- Regarding to variable X, “def” occurs in S1 and S3, while “use” occurs in S3 and S4.
- For variable X, we have 4 pairs to cover (S1, S3), (S1, S4), (S3, S3) and (S3, S4)

Example for Def-Use pair coverage (continued)

Without entering the loop – cover (S1, S4)



Example for Def-Use pair coverage (continued)



Example for Def-Use pair coverage (continued)

More than one loops – cover (S1, S3), (S3, S3) and (S3 ->S4)

```
S1:     X = ……
S2:     WHILE ….. DO { ……
S3:                    X = X * I}
S4:     Y = X
```

Example for Def-Use pair coverage

- Without entering the loop – cover (S1, S4) √
- ~~One loop – cover (S1, S3) and (S3 ->S4)~~
- More than one loops – cover (S1, S3), (S3, S3) and (S3->S4) √

Two paths are sufficient to cover all def-use pairs of variable x

More data-flow coverage criteria

- all-defs
- all-c-uses and some-p-uses
- all-p-uses and some-c-uses
- all-uses
- all def-use pairs (d-upairs)

Partial ordering of coverage criteria