Lecture 25:
Data Flow and Dependence Graphs

Kenneth M. Anderson

Foundations of Software Engineering

CSCI 5828 - Spring Semester, 1999

Today’s Lecture

- White-Box Testing
  - Data Flow Graphs
- Minimum Retesting
  - Program Dependence Graphs
    - Control Dependence Graphs
    - Data Dependence Graphs

Flow Graphs

Graph representation of control flow and data flow relationships

- Control Flow

  The partial order of statement execution, as defined by the semantics of the language

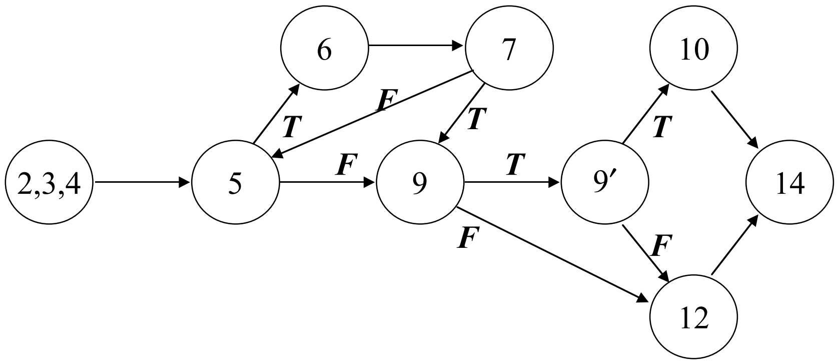
- Data Flow

  The flow of values from definitions of a variable to its uses

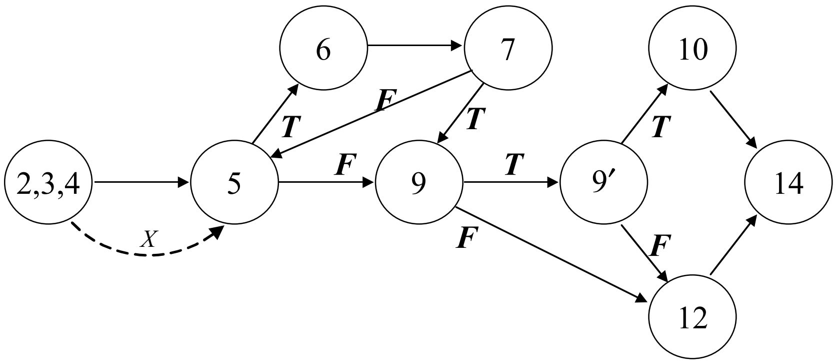
A Sample Ada Program to Test

```
function P return INTEGER is
begin
    X, Y: INTEGER;
    READ(X); READ(Y);
    while (X > 10) loop
        X := X – 10;
        exit when X = 10;
    end loop;
    if (Y < 20 and then X mod 2 = 0) then
        Y := Y + 20;
    else
        Y := Y – 20;
    end if;
    return 2 ∗ X + Y;
end P;
```

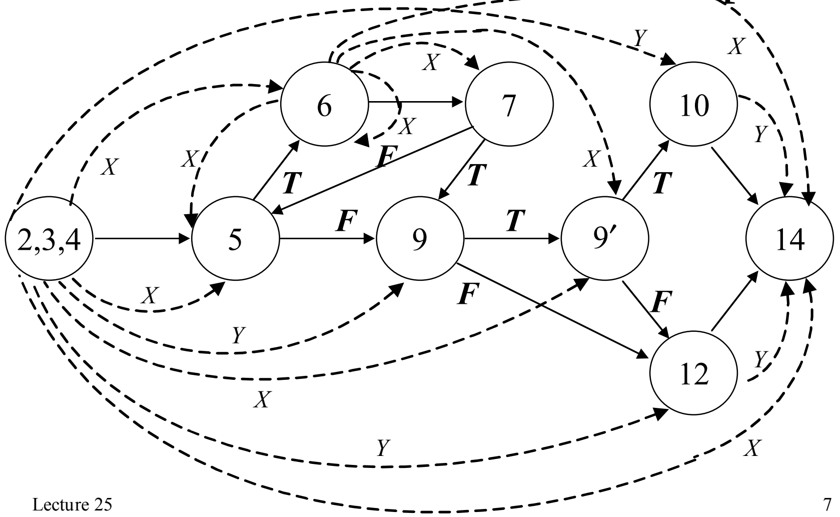
P’s Control Flow Graph (CFG)



P’s CFG with a Data Flow Edge



P’s Control/Data Flow Graph

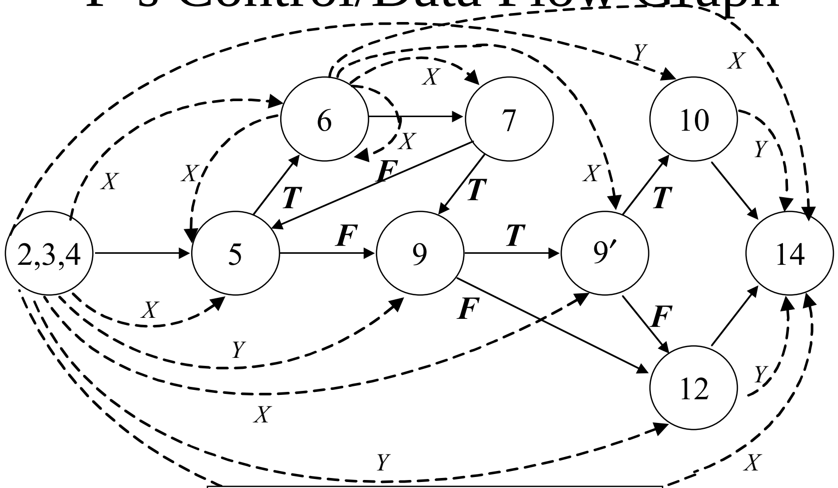


White-box Testing Criteria

- Use Coverage

  Select a test set *T* such that, by executing *P* for each *d* in *T*, all paths leading from each definition of a variable to each use of that variable in *P*’s control/data flow graph are traversed at least once

P’s Control/Data Flow Graph



How many test cases are needed?

Minimizing Retesting

- Test Only What Is Affected by a Change
- Key: Dependency Analysis

  Also used for optimization, parallelization, …
- At Coarse Level, Module Relationships

  Uses, calls, imports, includes, …
- At Fine Level, Control and Data Flow

  Program dependence graphs

Program Dependence Graph (PDG)

- Summary Representation of “Dependence”
- Nodes Are Either Statements or Predicates or the Special Node “Entry’’
- Two Kinds of Edges
  - Control dependence edge
  - Data dependence edge
- Two Subgraphs Induced by the Edges

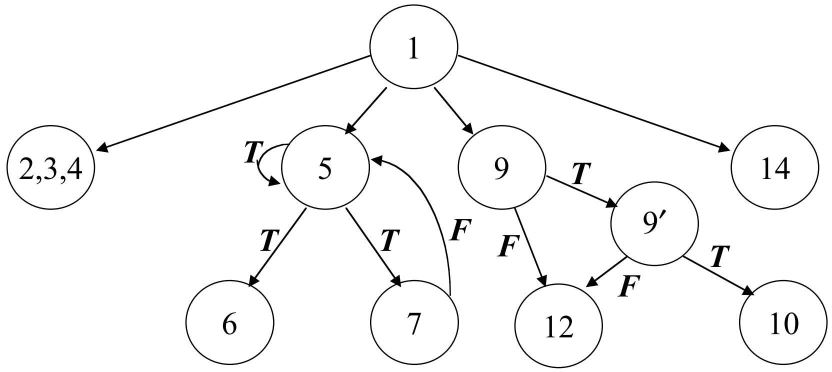
Control Dependence Graph (CDG)

- Informal Definition
  - For nodes X and Y in a CFG, Y is control dependent on X if, during execution, X can directly affect whether Y is executed

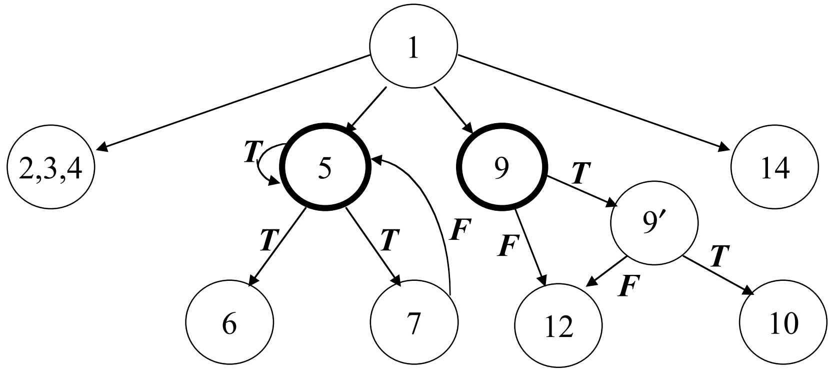
Control Dependence Graph (CDG)

- Formal Definition
  - Let X and Y be nodes in a CFG. If Y appears on every path from X to the exit node, where Y != X, then Y post-dominates X.
  - There is a control dependence from X to Y with label L iff:
    - there is a non-null path p from X to Y, starting with edge L, such that Y post-dominates every node strictly between X and Y on p; and
    - Y does not post-dominate X.

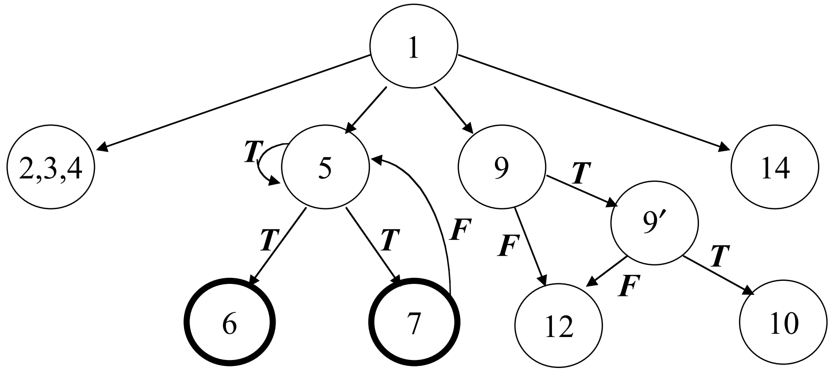
P's Control Dependence Graph



P's Control Dependence Graph



P's Control Dependence Graph



Data Dependence Graph (DDG)

- Informal Definition
  - Two statements are data dependent if they might reference the same memory location and one of the references is an assignment to the memory location

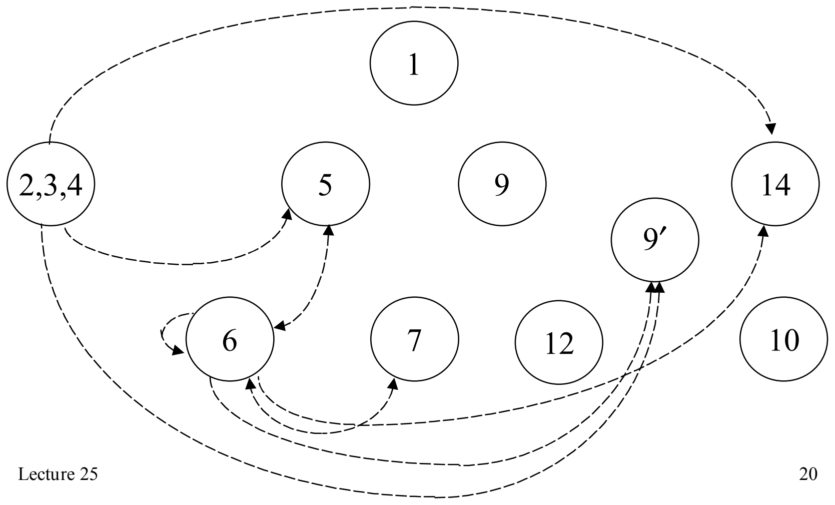
Data Dependence Graph (DDG)

- Informal Definition
  - Two statements are data dependent if they might reference the same memory location and one of the references is an assignment to the memory location
  - Intuition: If the statements cannot be switched without affecting the program, then they are data dependent

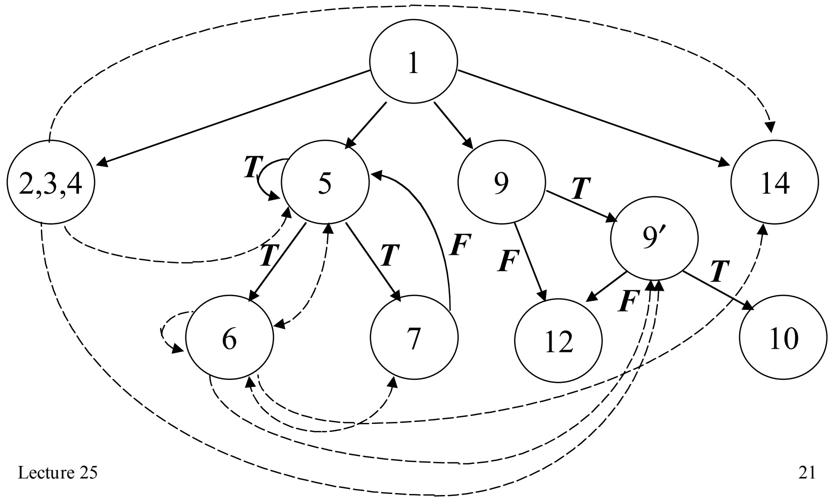
Data Dependence Graph (DDG)

- Formal Definition
  - Let X and Y be nodes in a CFG. There is a data dependence from X to Y with respect to a variable v iff there is a non-null path p from X to Y with no intervening definition of v and either:
    - X contains a definition of v and Y a use of v;
    - X contains a use of v and Y a definition of v; or
    - X contains a definition of v and Y a definition of v.

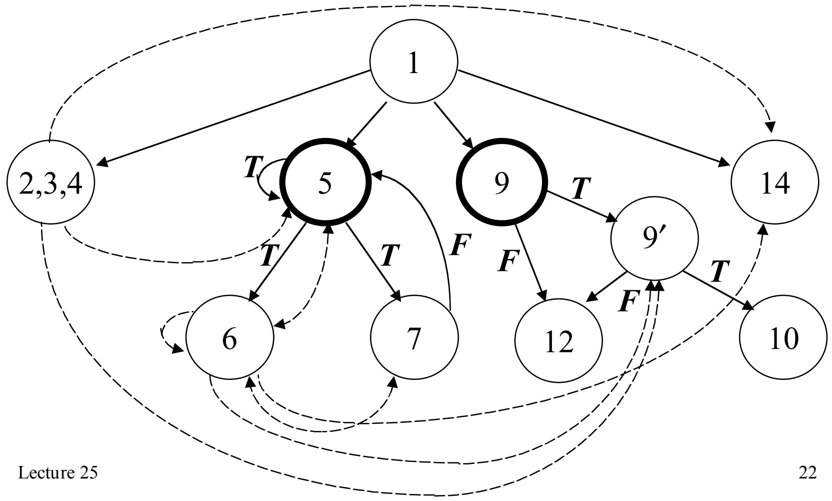
P’s Data Dependence Graph for X



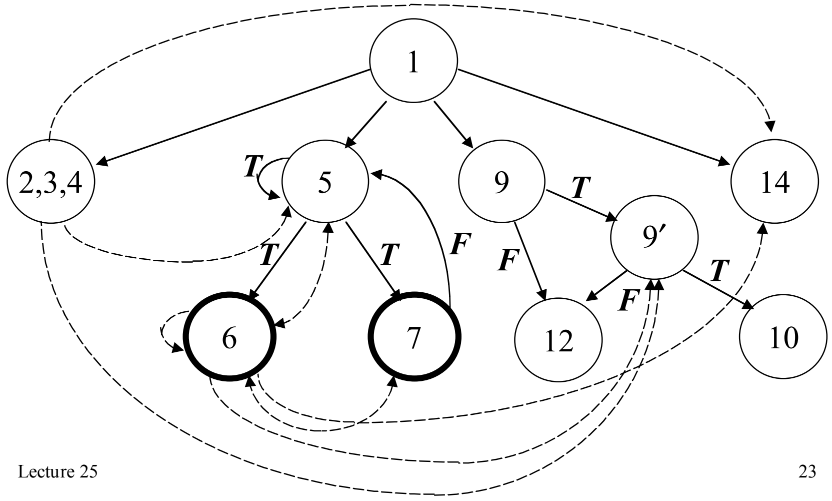
P’s PDG (DDG for X Only)



P’s PDG (DDG for X Only)



P’s PDG (DDG for X Only)



Minimum Regression Testing

Given program P, its modified version P′, and test set T used to test P, find a way, making use of T, to test P′

- Identify changes to P resulting in P′
- Select T′, a subset of T, related to changes
- Run T′ on P′

Goals

- Safety

  Every relevant test from T must be selected
- Precision

  Select only tests that exhibit different behavior
- Efficiency

  Cheap to calculate and run T′

Modifications

- Adding Statements
- Deleting Statements
- Changing Statements

- Theorem

  Need only tests in T that can traverse different *regions* of statements in P and P′, where regions are dependent-equivalent sub-CDGs

Test Selection Algorithm

```
procedure SelectTests
  Construct CDGs of P and P’, with entry nodes E1, E2
  T’ = Compare (E1, E2)

procedure Compare (N1,N2)
  mark N1 and N2 visited
  if (children of N1 and N2 differ) then
    return all tests that traverse N1
  else
    T’ = NULL
    for each region or predicate child C1 of N not yet visited do
      find C2, the corresponding child of N2
      T’ = T’ union Compare (C1,C2)
```