Test

Rasmus Guldborg Pedersen

June 2015

Overview

1 Q 2.8: Define/Use testing

Define/Use Testing

- Data flow
- Derive test cases

Definitions

Given a program P with variables V, and program graph $G(P)$ with nodes N.

For variable $v \in V$ and node $n \in N$

- **Defining node** DEF(v, n), iff v is *defined* at n.
- **Use node** USE(v, n), iff is *used* at n.

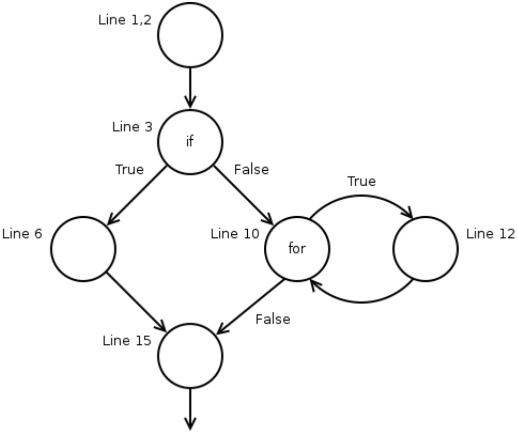
Definitions

- **Definition/use path** (du-path) is a path in $PATHS(P)$ such that for $v \in V$
  - DEF(v, m) is initial node
  - USE(v, n) is final node
- **Definition clear** (dc-path) is a path in $PATHS(P)$ with initial and final node DEF(v, m) and USE(v, m) such that no other defining node exists in the path.

Example Code

```
int Factorial(int n) {
    int f = 1;
    if (n < 0)
    {
        // -1 indicates an error
        f = -1;
    }
    else
    {
        for (int i = 1; i < n; i++)
        {
            f *= i;
        }
    }
    return f;
}
```

Example Graph



All-Uses Coverage

Var	du-path	Tests
n	$\langle 1, 3, 10\rangle$	-1, 1, 2
f	$\langle 2, 15\rangle\langle 6, 15\rangle\langle 12, 15\rangle$	-1, 2
i	$\langle 10\rangle\langle 10, 12\rangle$	1, 2

The End

> *"Testing shows the presence, not the absence of bugs."*
> *— Edsger W. Dijkstra*