

Effects of Assignments on Branches

```

sumodd = sumeven = 0;
odd = quit = 0;
for (i = 0; !quit &&
     i < 1000; i++)
    if (a[i] == 0)
        quit = 1;
    else if (odd) {
        sumodd += a[i];
        odd = 0;
    }
    else {
        sumeven += a[i];
        odd = 1;
    }

```

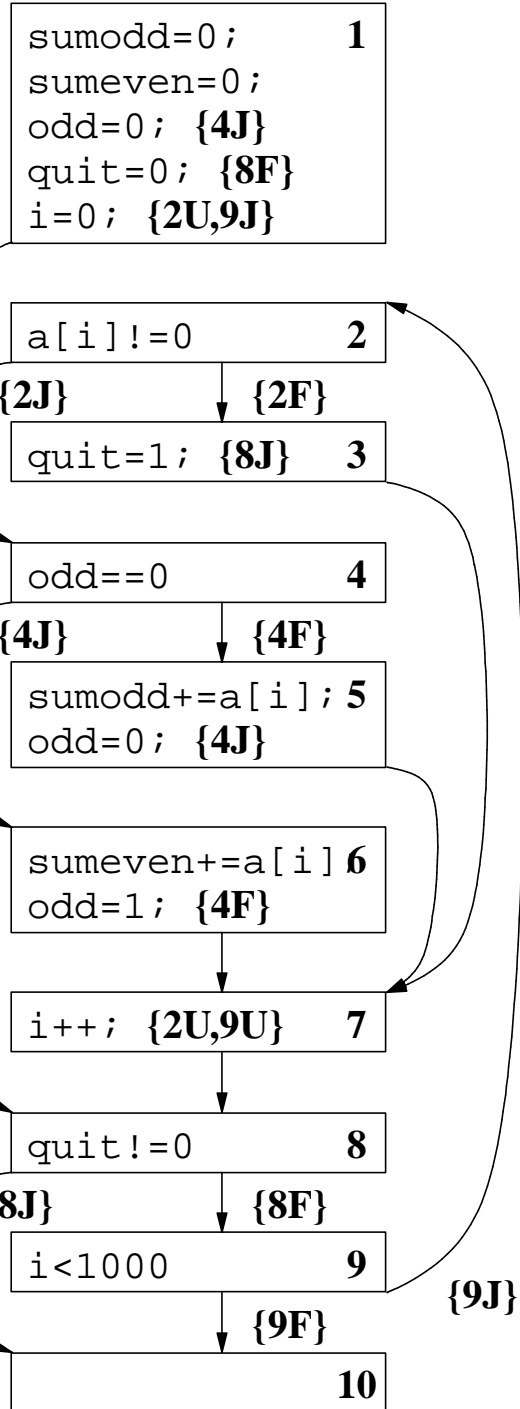
(a) Source Code

- (1) blk 1 nullifies blk 2
- (2) blk 1 makes blk 4 jump
- (3) blk 1 makes blk 8 fall thru
- (4) blk 1 makes blk 9 jump
- (5) blk 3 makes blk 8 jump
- (6) blk 5 makes blk 4 jump
- (7) blk 6 makes blk 4 fall thru
- (8) blk 7 nullifies blks 2,9

(c) Explicit Constraints

- (1) 8
- (2) 8→9
- (3) 8→9→2→3→7
- (4) 8→9→2→4→5→7
- (5) 8→9→2→4→6→7

(d) Paths in Loop



(b) Control Flow

Logical Correlation between Branches

```

sumneg = sumall = 0;
sumpos = 0;
for (i = 0; i < 1000;
    i++) {
    if (a[i] < 0)
        sumneg += a[i];
    sumall += a[i];
    if (a[i] > 0)
        sumpos += a[i];
}
    
```

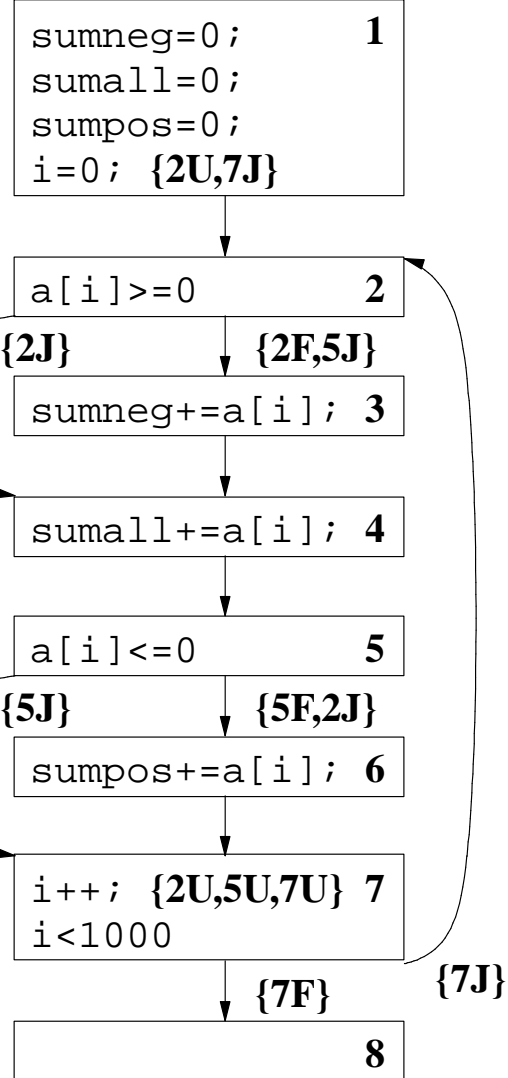
(a) Source Code

- (1) blk 1 nullifies blk 2
- (2) blk 1 makes blk 7 jump
- (3) blk 2 fall thru makes blk 5 jump
- (4) blk 5 fall thru makes blk 2 jump
- (5) block 7 nullifies blocks 2,5,7

(c) Explicit Constraints

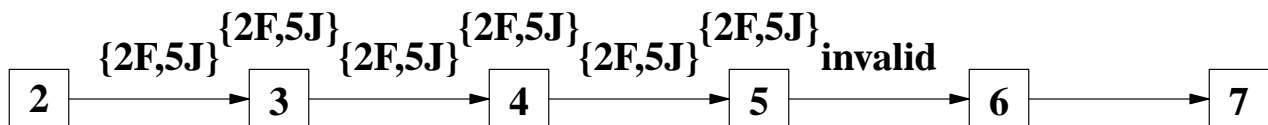
- (1) 2→4→5→7
- (2) 2→3→4→5→7
- (3) 2→4→5→6→7
- (4) 2→3→4→5→6→7

(d) Paths in Loop



(b) Control Flow

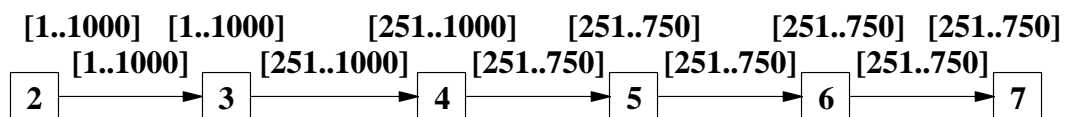
- Path 4 is not feasible.



Ranges of Iterations and Branch Outcomes

<pre> summid = sumall = 0; for (i = 0; i < 1000; i++) { if (i != m && 249 < i && i < 750) summid += a[i]; sumall += a[i]; } </pre> <p>(a) Source Code</p>	<p>(b) Control Flow</p>
<p>(1) blk 1 makes blks 3,7 jump (2) blk 1 makes blk 4 fall thru (3) blk 2 will jump at most once (4) blk 3 jump makes blk 4 fall thru (5) blk 3 fallthru in iters [251..1000] (6) blk 3 jump in iters [1..250] (7) blk 4 fallthru in iters [1..750] (8) blk 4 jump in iters [751..1000] (9) blk 4 jump makes blk 3 fall thru (10) blk 7 nullifies blks 2,3,4,7</p> <p>(c) Explicit Constraints</p>	<p>(1) 2→6→7 (2) 2→3→6→7 (3) 2→3→4→6→7 (4) 2→3→4→5→6→7</p> <p>(d) Paths in Loop</p>

- Iteration constraints propagated thru path 4.



Worst-Case Loop Analysis Algorithm

iterations_handled = 0.

while (iterations_handled < N) **do**

do

Find longest path available for execution.

Use this longest path for 1 iteration.

iterations_handled += 1.

while (longest path's caching behavior changes).

iters_to_do =

longest path's required iterations +

min (longest path's nonrequired iterations,
loop's nonrequired iterations).

Use this longest path for iters_to_do iterations.

iterations_handled += iters_to_do.