CSC 255/455 Software Analysis and Improvement

Control Dependence

Instructor: Chen Ding

Chapter 7, Optimizing Compilers for Modern Architectures, Allen and Kennedy
www.cs.rice.edu/~ken/comp515/lectures/

Dependence due to Control Flow

- An example loop
  - What does the loop body (S3) depend on?

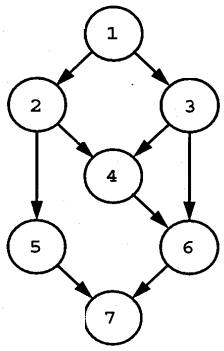
```
S1: if (B)
S2:   while (C)
S3:     some work
```

```
S1: if (B)
S2:   repeat
S3:     some work
S4:   until C
```


Control Dependence

- Two definitions: A statement y control dependent on statement x if:
  - Dominance-based definition (for a compiler)
    - there exists a non-trivial path from x to y such that every statement z≠x in the path is postdominated by y and
    - x is not postdominated by y.
  - Intuitive definition (for human)
    - one branch out of x forces execution of y and another doesn't
- A node x postdominates node y if any path from y to the exit must pass through x.
- Usually we compute control dependences between basic blocks, not statements.

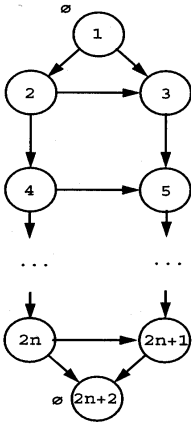
What Each Node Control Dependent On?



Which nodes can run in parallel, assuming no data dependence?

Worst Case Example

- n nodes and how many control dependences?
- Can control dependence graphs get much larger than the corresponding CFG?



Control Dependence: Loops

- Use a loop control node to represent the loop

```
10    DO I = 1, 100
20         A(I) = A(I) + B(I)
30         IF (A(I).GT.0) GO TO 50
40         A(I) = -A(I)
50         B(I) = A(I) + C(I)
   ENDDO
```

Control Dependence and Parallelization

- For simplicity, we shall only consider:
  - Forward branches - they create loop-independent control dependences
  - Control Dependences due to loops
- From Chapter 2: Most loop transformations are unaffected by loop-independent dependences
- Loop reversal, loop skewing, strip mining, index-set splitting, loop interchange do not affect independent dependences
- Might be problematic: Loop fusion, loop distribution
- However, since exit branches are excluded, loop fusion is not a problem

Loop Distribution

```
      DO I = 1, N
S1        IF (A(I).LT.B(I)) GOTO 20
S2        B(I) = B(I) + C(I)
20      CONTINUE
      ENDDO
```

$S_1\ \delta^{-1}\ S_2$

- Distributing…

```
      DO I = 1, N
S1        IF (A(I).LT.B(I)) GOTO 20
      ENDDO

      DO I = 1, N
S2        B(I) = B(I) + C(I)
      ENDDO
20    CONTINUE
```

- Incorrect!

Loop Distribution

- Problem: control dependences crossing between distributed loops
- Solution: Keep a history of the evaluated conditions (similar to if-conversion).

```
      DO I = 1, N
S1        IF (A(I).LT.B(I)) GOTO 20
S2        B(I) = B(I) + C(I)
20      CONTINUE
      ENDDO
```

- Convert to:

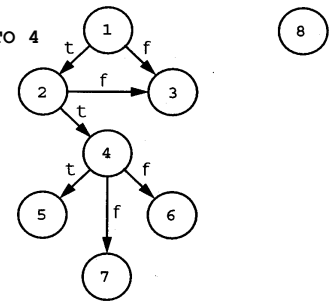
```
      DO I = 1, N
S1        e(I) = A(I).LT.B(I)
      ENDDO

      DO I = 1, N
S2        IF (e(I).EQ..FALSE.) B(I) = B(I) + C(I)
      ENDDO
```


Loop Distribution

- More complex example:

```
      DO I = 1, N
1       IF (A(I).NE.0) THEN
2        IF (B(I)/A(I).GT.1) GOTO 4
        ENDIF
3       A(I) = B(I)
        GOTO 8
4       IF (A(I).GT.T) THEN
5        T = (B(I) - A(I)) + T
        ELSE
6        T = (T + B(I)) - A(I)
7        B(I) = A(I)
        ENDIF
8       C(I) = B(I) + C(I)
      ENDDO
```



Loop Distribution

- Fusion into "like" regions
- Needs two execution variables E2(I) and E4(I) to hold result of branches at statement 2 and 4 respectively