



CS 4120 Introduction to Compilers

Ross Tate
Cornell University

Lecture 33: Loop Optimizations II

Induction-Variable Strength Reduction

```
m = i * 8;
while (i < 10) {
    i = i + 2;
    m = i * 8; i = i + 2  $\rightarrow m = i + (i \cdot 4)$ 
    use(m + 6);
}
while (i < 10) {
    i = i + 2;
    m = m + 16;
    use(m + 6);
}
```

2

Loop Unrolling

```
m = i * 8;
while (i < 10) {
    i = i + 2;
    m = m + 16;
    use(m + 6);
}

m = i * 8;
while (i < 10-2*1) {
    i = i + 2;  $\rightarrow i = i + 4$ 
    m = m + 16;  $\rightarrow m = m + 32$ 
    use(m + 6);
    i = i + 2;
    m = m + 16;
    use(m + 6);
}
```

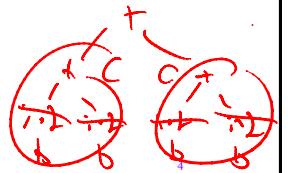
3

Common-Subexpression Elimination

$$\begin{aligned} a &:= i; \\ b &:= a * 2; \\ \text{if } (c) & \\ &\quad d := a + b; \\ &\quad b := 0; \\ \text{else} & \\ &\quad d := i + b; \\ &\quad e := b * 3; \\ &\quad f := b * 3; \\ &\quad i := i * 2; \\ &\quad \text{return } e + f + i * 2; \end{aligned}$$

Annotations:

- $a := i$: $b := i$
- $b := a * 2$: $b := i * 2$
- $d := a + b$: $d := i + i$
- $b := 0$: $b := 0$
- $d := i + b$: $d := i + b$
- $e := b * 3$: $e := b * 3$
- $f := b * 3$: $f := b * 3$
- $i := i * 2$: $i := i * 2$



Foo(n)

```
if (n > 0)
    return f(n-1) + f(n-1)
else
    return 0
```

5