# 1 Theory

## 1.1 Control flow graphs

In figure 1 I assume that the code is:

for ( a ; b ; c) {}
d;
e;

not:

for ( a ; b ; c)
    d;
e;

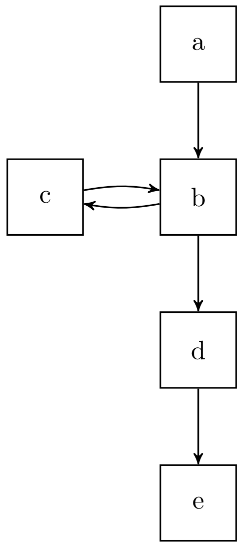


Figure 1: for ( a ; b ; c ) d ; e ;

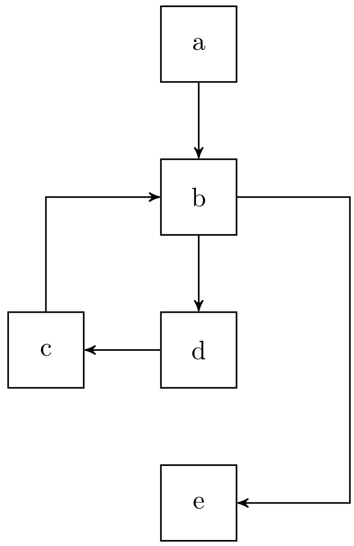


Figure 2: a ; while ( b ) { d ; c ; } e ;

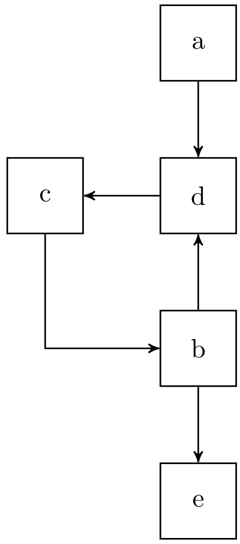


Figure 3: a ; do { d ; c ; } while ( b ); e ;

## 1.2 Optimizations

### 1.2.1 Copy propagation

1. a=1
2. b=2
3. c=3
4. d=a+x
5. e=b+c
6. f=e
7. **g = f**
8. g=d+y
9. a=b+c

Figure 4: Before

1. a=1
2. b=2
3. c=3
4. d=a+x
5. e=b+c
6. f=e
7. **g = e**
8. g=d+y
9. a=b+c

Figure 5: After

### 1.2.2 Common subexpression elimination

1. a=1
2. b=2
3. c=3
4. d=a+x
5. **e = b + c**
6. f=e
7. g=f
8. g=d+y
9. **a = b + c**

Figure 6: Before

1. a=1
2. b=2
3. c=3
4. d=a+x
5. **t1 = b + c**
6. **e = t1**
7. f=e
8. g=f
9. g=d+y
10. **a = t1**

Figure 7: After

### 1.2.3 Constant propagation

Assuming only constant propagation, not constant folding.

1. $a = 1$
2. $b = 2$
3. $c = 3$
4. $\mathbf{d = a + x}$
5. $\mathbf{e = b + c}$
6. $f = e$
7. $g = f$
8. $g = d + y$
9. $\mathbf{a = b + c}$

Figure 8: Before

1. a=1
2. b=2
3. c=3
4. $\mathbf{d = 1 + x}$
5. $\mathbf{e = 2 + 3}$
6. f=e
7. g=f
8. g=d+y
9. $\mathbf{a = 2 + 3}$

Figure 9: After