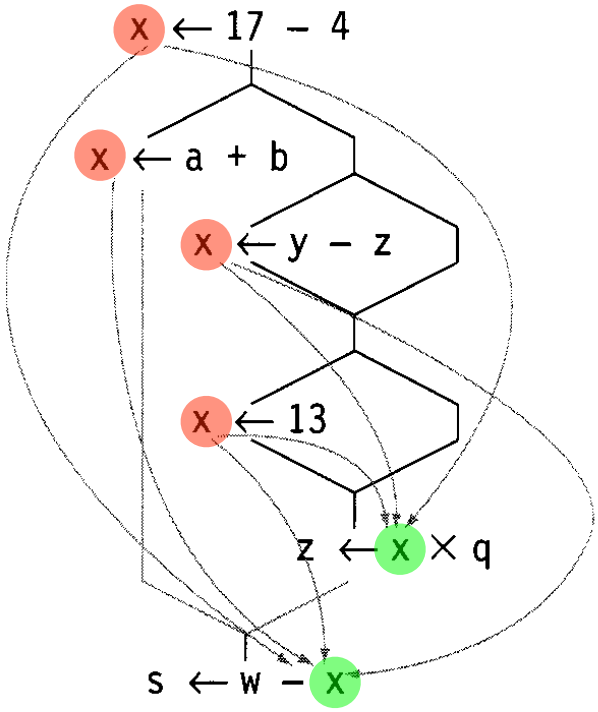
How many definitions and uses are here?

What are the possible values of x in its two uses?

How can we improve a program if we have this info?



Def-Use Chains

Reaching Definition

- Reach(n) definition (how large can a set be?)
- Local information
- Data flow equation
- Initialization
- Convergence

Dead Code Elimination

```
w: a <- a +1
x:  c <- a * b
y:  c <- b + d
z:  print(c)
```

Which statements are useless (dead)?

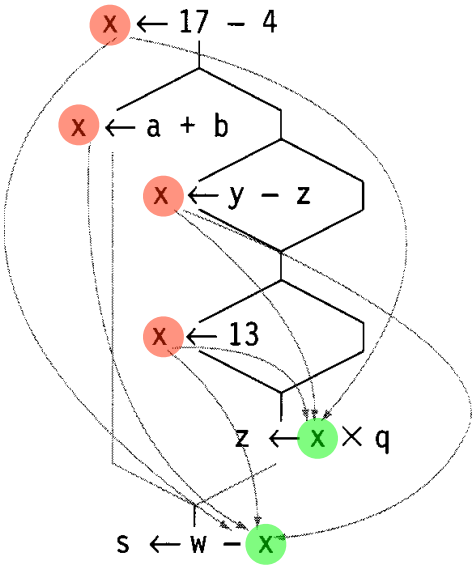
What are the def-use chain?

Can we infer the dead code automatically?

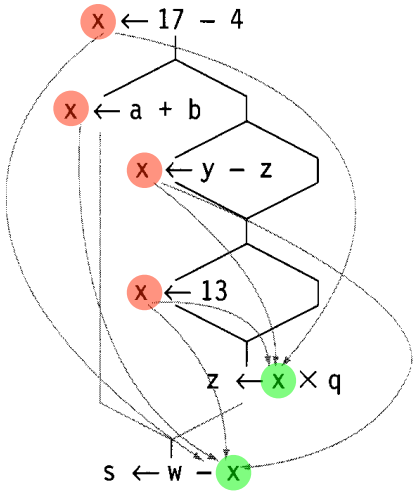
What is the cost of the algorithm?

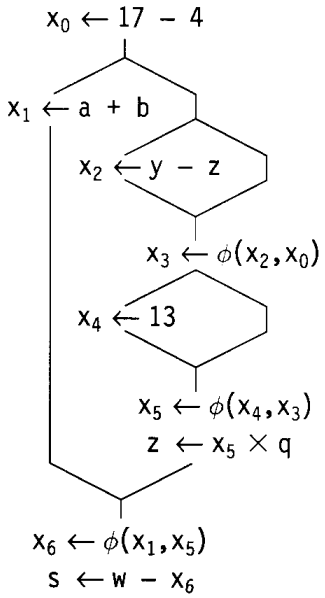
Static Single Assignment (SSA)

- Size of a def-use graph
- SSA
  - each static definition defines a new name
  - each use has a single static definition
- Meet operation
  - x <- Φ(y, z)
- Naive SSA insertion
  - Algorithm?
  - How many Φ functions are unnecessary?
  - What is the earliest point of eac meet?



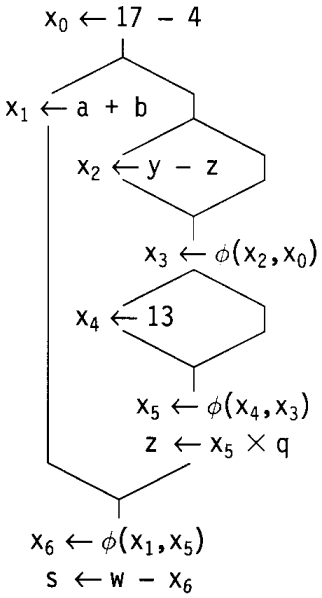
Minimal SSA





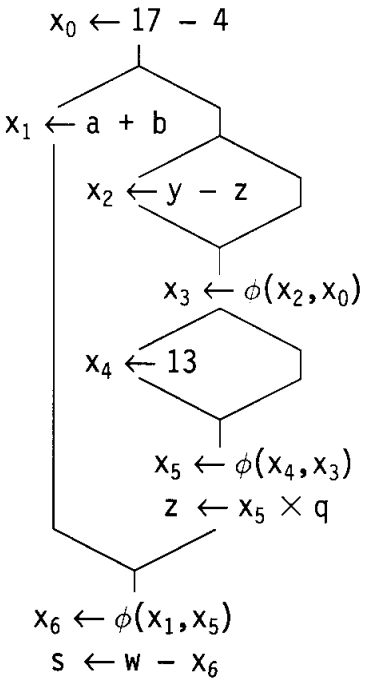
SSA Algorithm

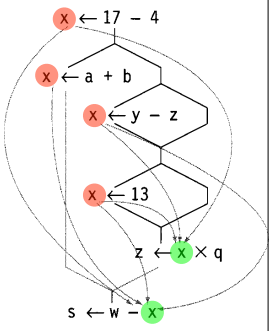
- Dominance frontiers
  - DF(n) definition
- Dominators
  - Dom(n) definition
  - Data flow equation
  - Initialization
  - Convergence
- Dom to DF
  - backward alg. [EAC, Fig. 9.10]
  - forward alg., linear time [AK, Fig. 4.9]



SSA Algorithm

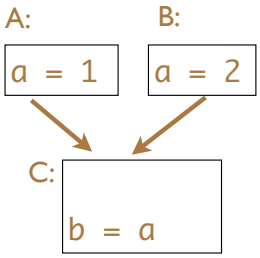
1. CFG
2. compute Dom
3. compute DF
4. insert phi
5. rename
6. reaching def
7. "destruct" SSA



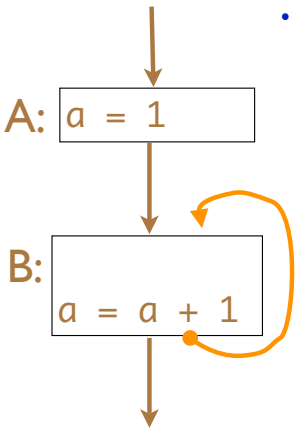


Example 1

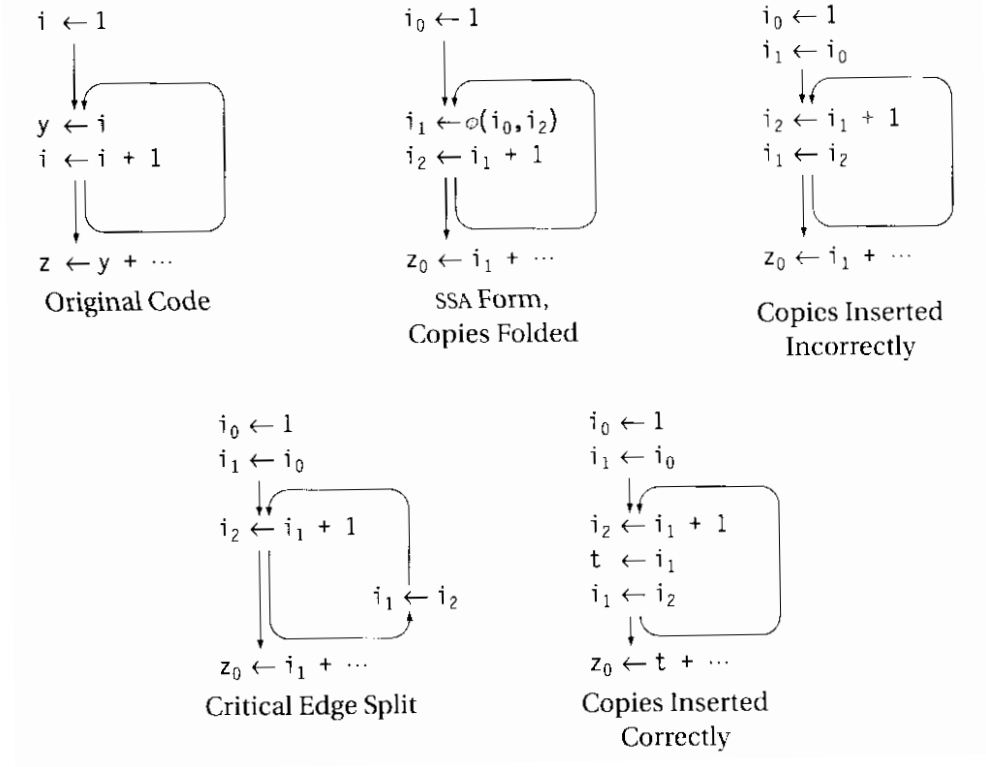
- IDOM DF phi renaming reach destruction

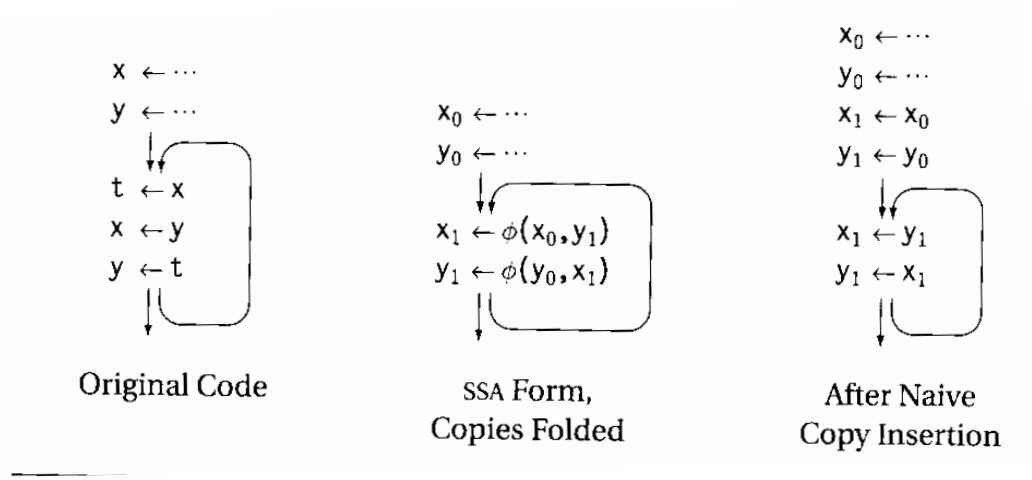


Example 2



- IDOM DF phi renaming reach destruction





15-745

SSA

http://www.cs.cmu.edu/afs/cs/academic/class/15745-s07/www/lectures/lect3.pdf

CCP