Program Analysis

https://www.cse.iitb.ac.in/~karkare/cs618/

Static Single Assignment (SSA) (continued)

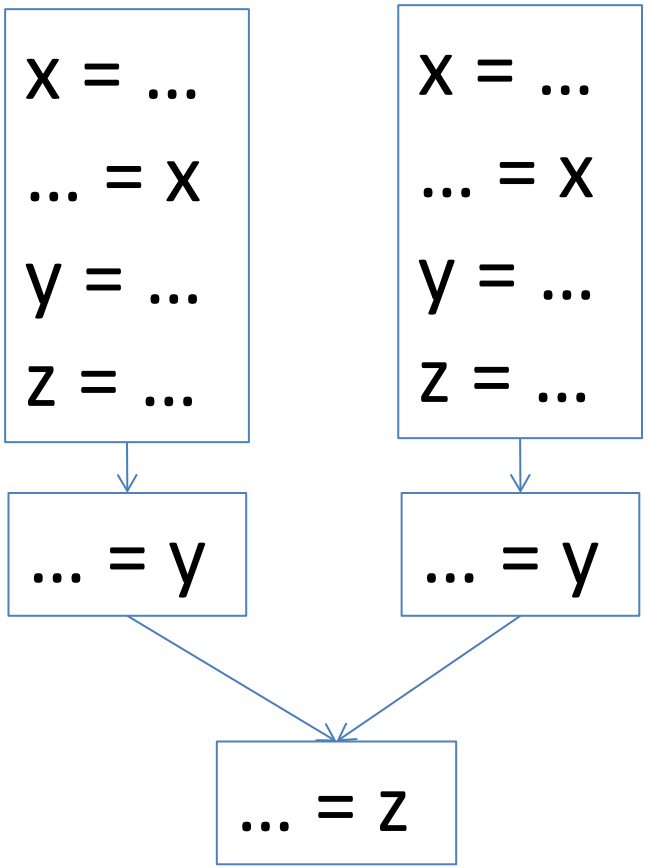
Amey Karkare
Dept of Computer Science and Engg
IIT Kanpur
Visiting IIT Bombay
karkare@cse.iitk.ac.in
karkare@cse.iitb.ac.in

Complexity of Construction

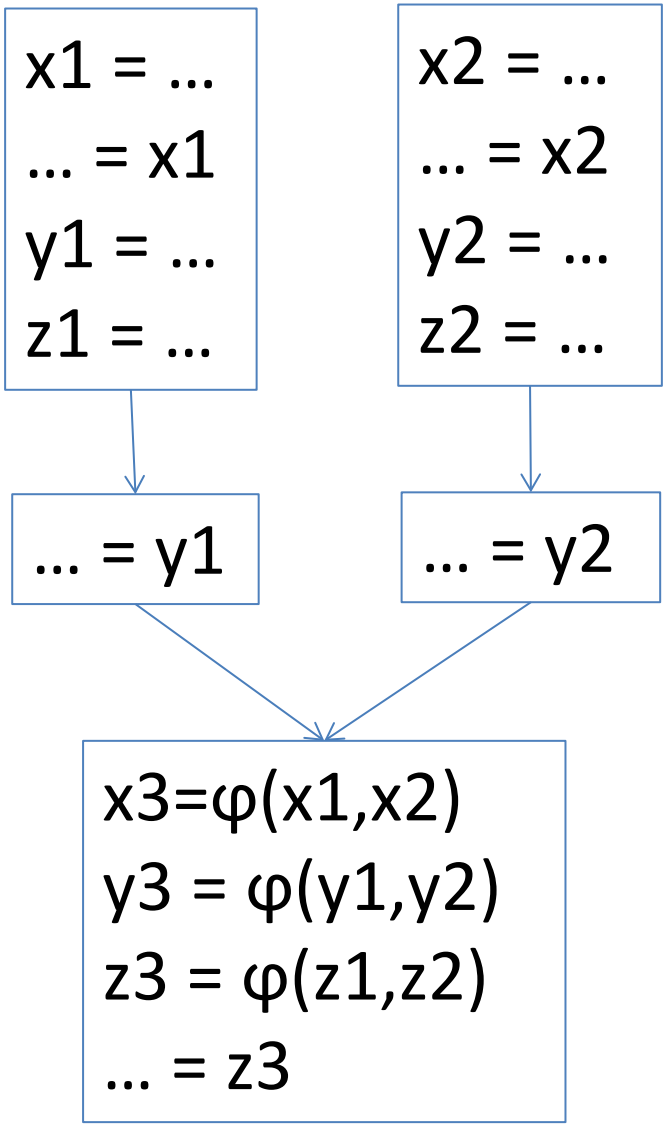
- $R = \max(N, E, A, M)$
- N: nodes, E: edges in flow graph
- A: number of assignments
- M: number of use of variables
- Computation of DF: $O(R^2)$
- Computation of SSA: $O(R^3)$
- In practice, worst case is rare.
- Practical complexity: $O(R)$

Linear Time Algo for φ-functions

- By Sreedhar and Gao, in POPL'95
- Uses a new data structure called DJ-graph
- Linear time is achieved by careful ordering of nodes in the DJ-graph
  - DF for a node is computed only once an reused later if required.



Original Program



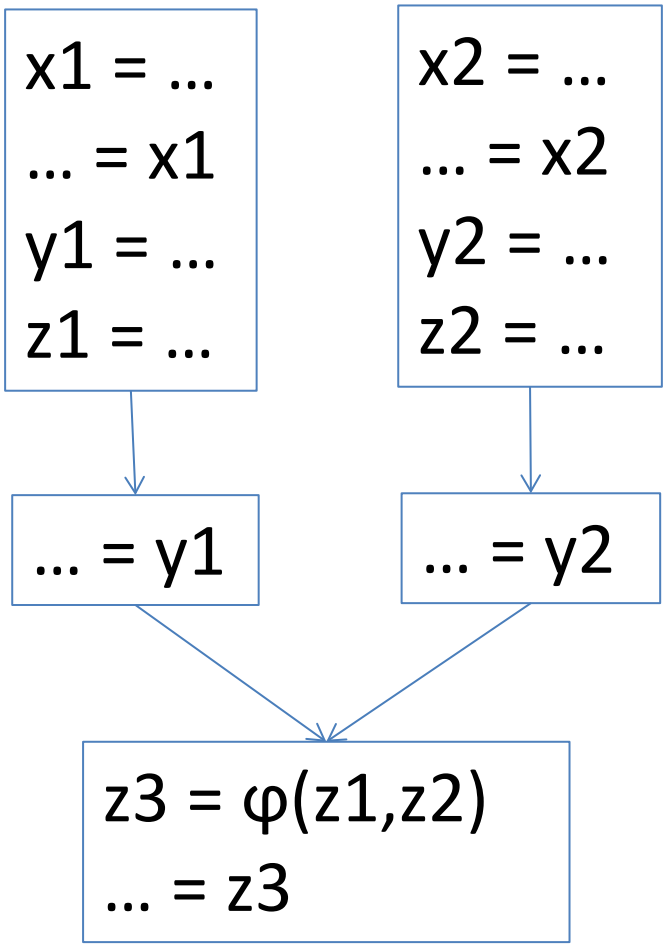
Minimal SSA form

Variants of SSA Form

- Minimal SSA still contains extraneous φ-functions
  - Inserts some φ-functions where they are dead
  - Would like to avoid inserting them
- Pruned SSA
- Semi-Pruned SSA

Pruned SSA

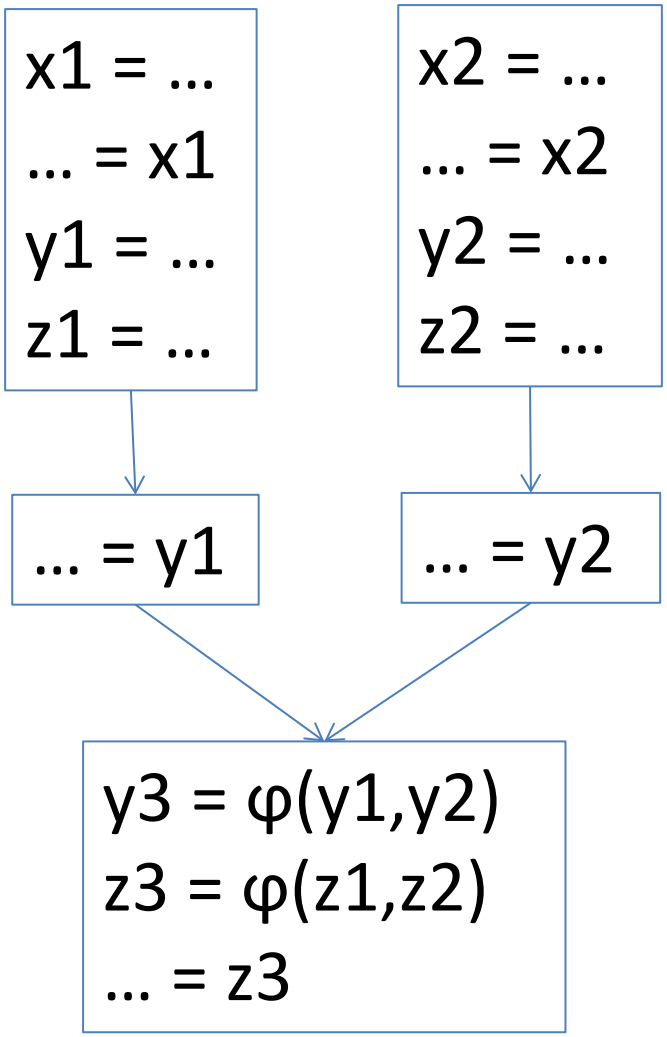
- Only insert φ-functions where their value is live
  - Inserts fewer φ-functions
  - Costs more to do
  - Requires global Live variable analysis



Pruned SSA form

Semi-pruned SSA

- *Semi-pruned SSA*: discard names used in only one block
  - Total number of φ-functions between minimal and pruned SSA
  - Needs only local Live information
  - Non-locals can be computed without iteration or elimination



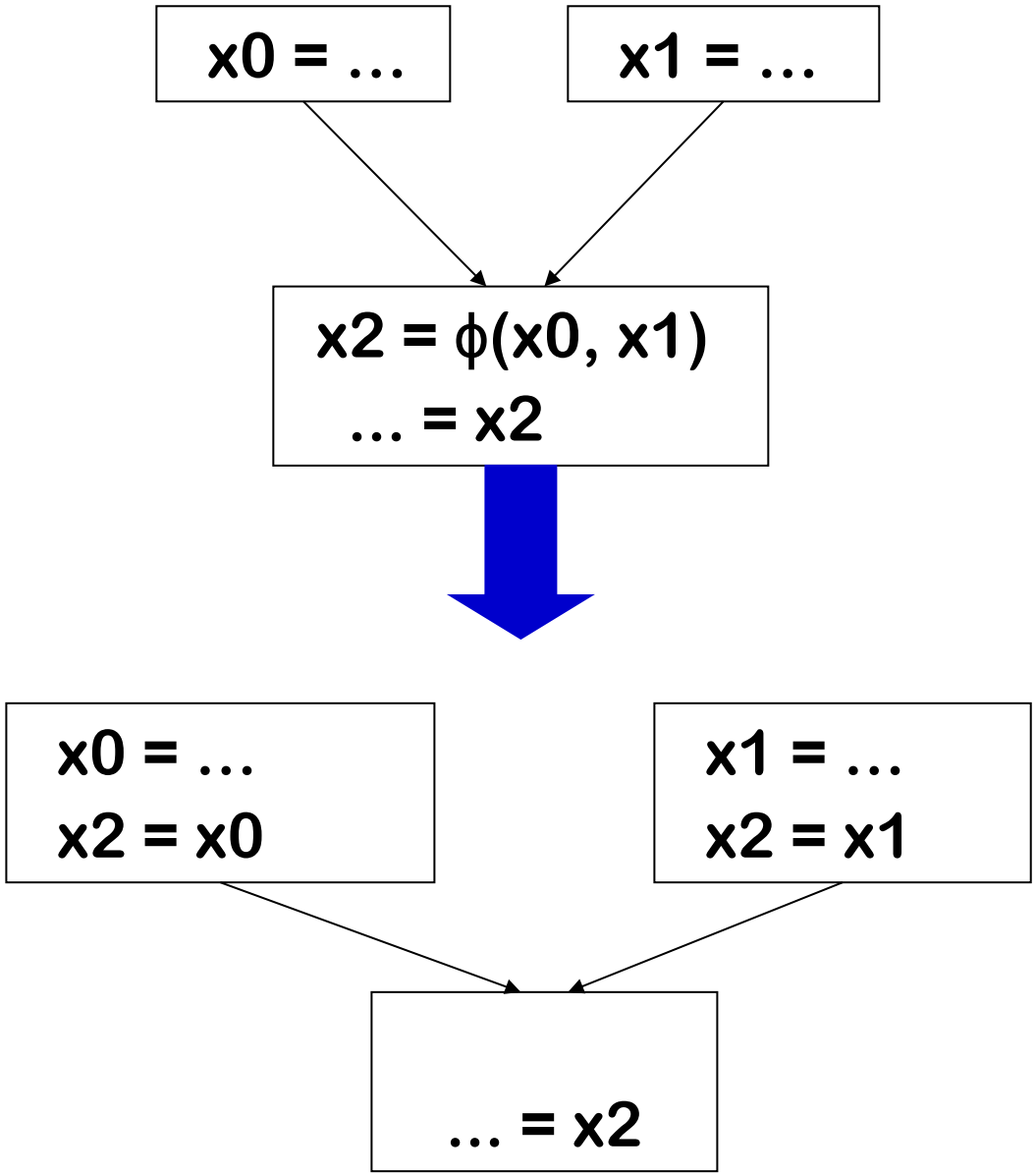
Semi-pruned SSA form

Computing Non Locals

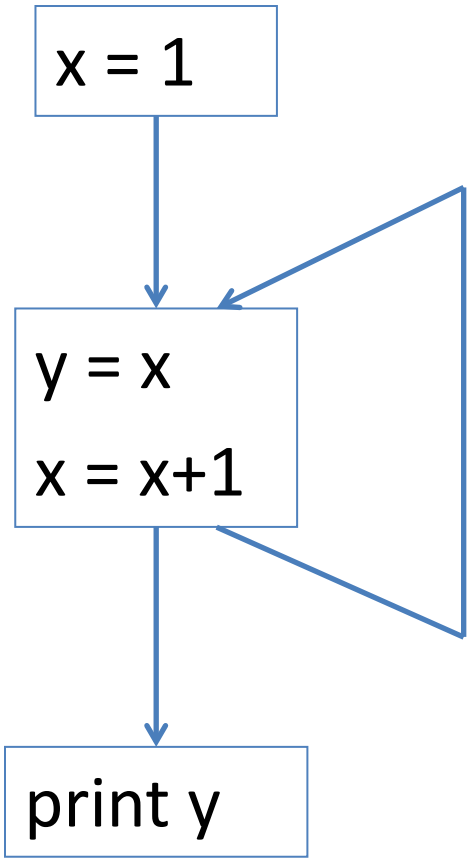
```
for each block B {
  defined = {}
  for each instruction v = x op y {
     if x not in defined
        non_locals = non_locals ∪ {x}
     if y not in defined
        non_locals = non_locals ∪ {y}
     defined = defined ∪ {v}
  }
}
```

SSA to Executable

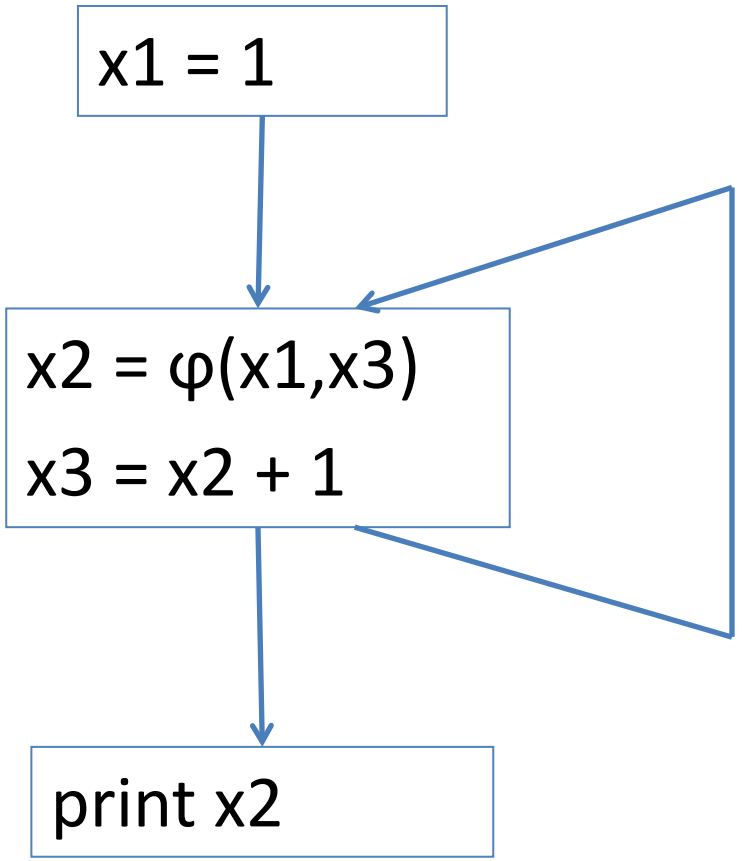
- At some point, we need executable code
  - Need to fix up the φ-function
- Basic idea
  - Insert copies in predecessors to mimick φ-function
  - Simple algorithm
    - Works in most cases, but not always
  - Adds lots of copies
    - Many of them will be optimized by later passes


x0 = …
x1 = …
x2 = ϕ(x0, x1)
… = x2
x0 = …
x2 = x0
x1 = …
x2 = x1
… = x2

Lost Copy Problem

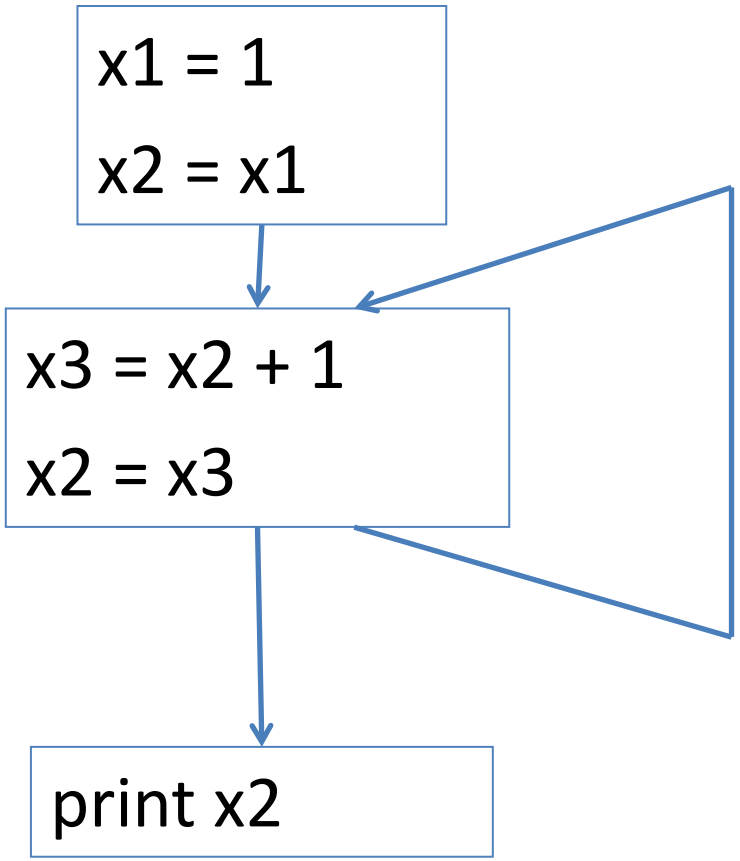


Program



SSA from with copy propagation

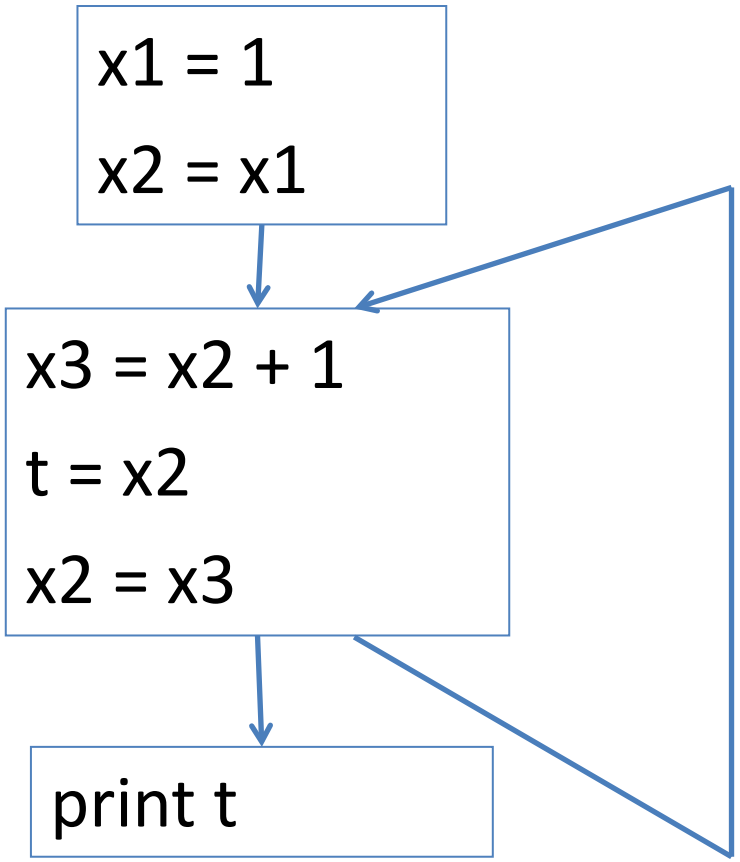
Lost Copy Problem



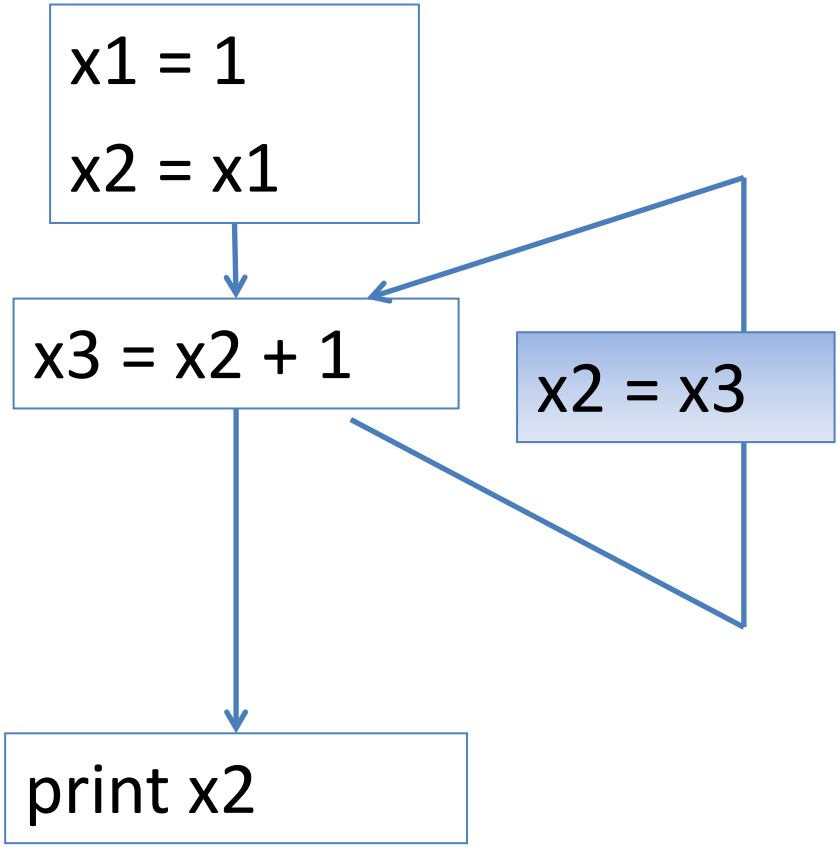
PROBLEM!!!

After φ-function removal

Lost Copy Problem: Solutions

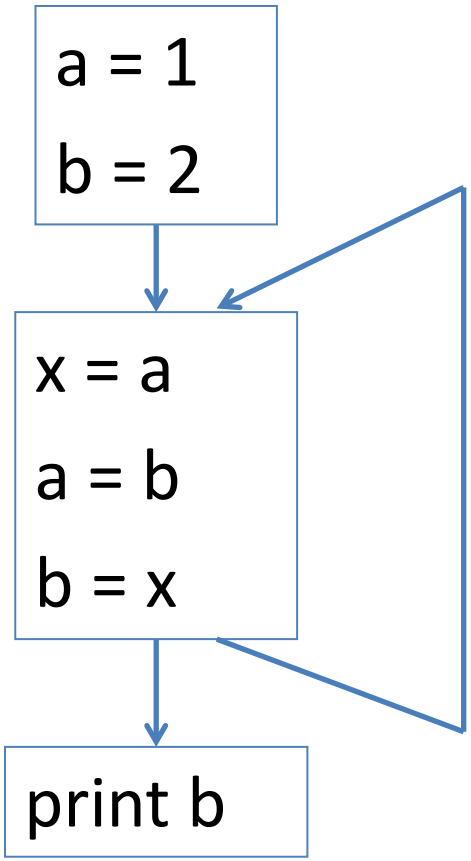


Solution 1: Temporary use

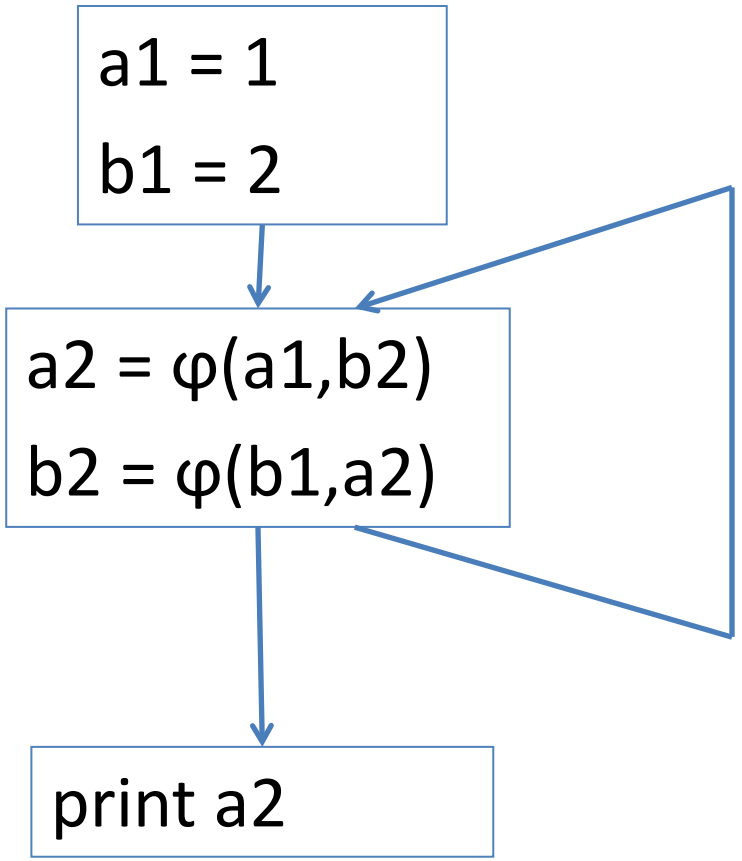


Solution 2: Critical Edge Split

Swap Problem

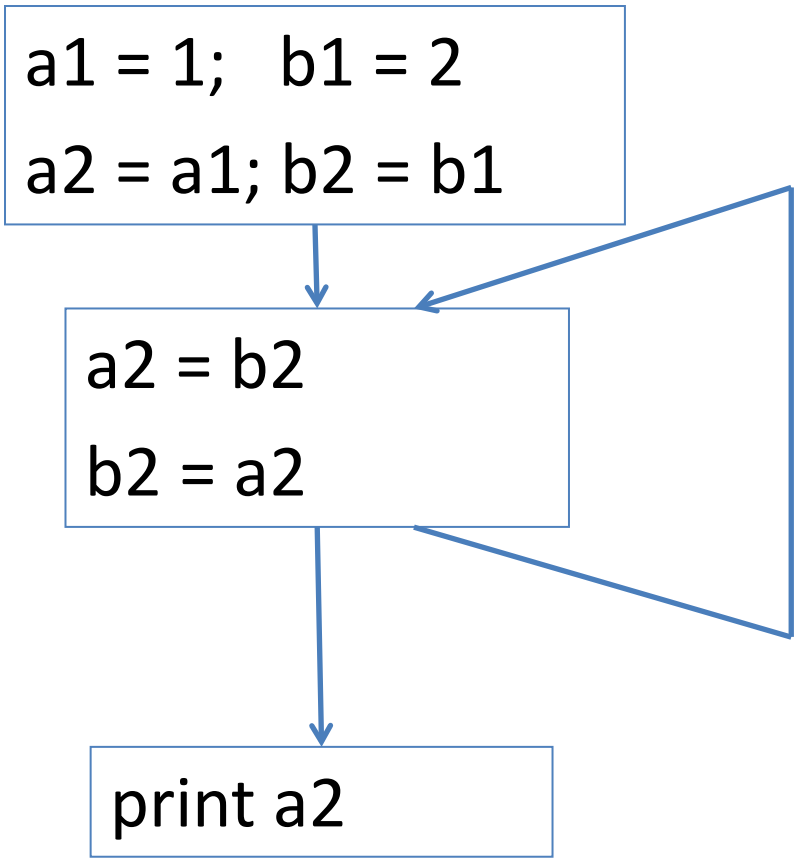


Program



SSA from with copy propagation

Swap Problem



PROBLEM!!!

Fix requires compiler to detect and break dependency from output of one φ-function to input of another φ-function. May require temporary if cyclic dependency exists.

After φ-function removal

SSA Form for Optimizations

- SSA form can improve and/or speed up many analyses and optimizations
  - (Conditional) Constant propagation
  - Dead code elimination
  - Value numbering
  - PRE
  - Loop Invariant Code Motion
  - Strength Reduction