Compiler Construction

BITS Pilani
Pilani Campus

Vinti Agarwal
April 2021

innovate achieve lead

BITS Pilani
Pilani Campus

CS F363, Compiler Construction

Lecture topic: Global Optimization

In previous lecture

- Constant Propagation ✓
- Dead Code elimination ✓

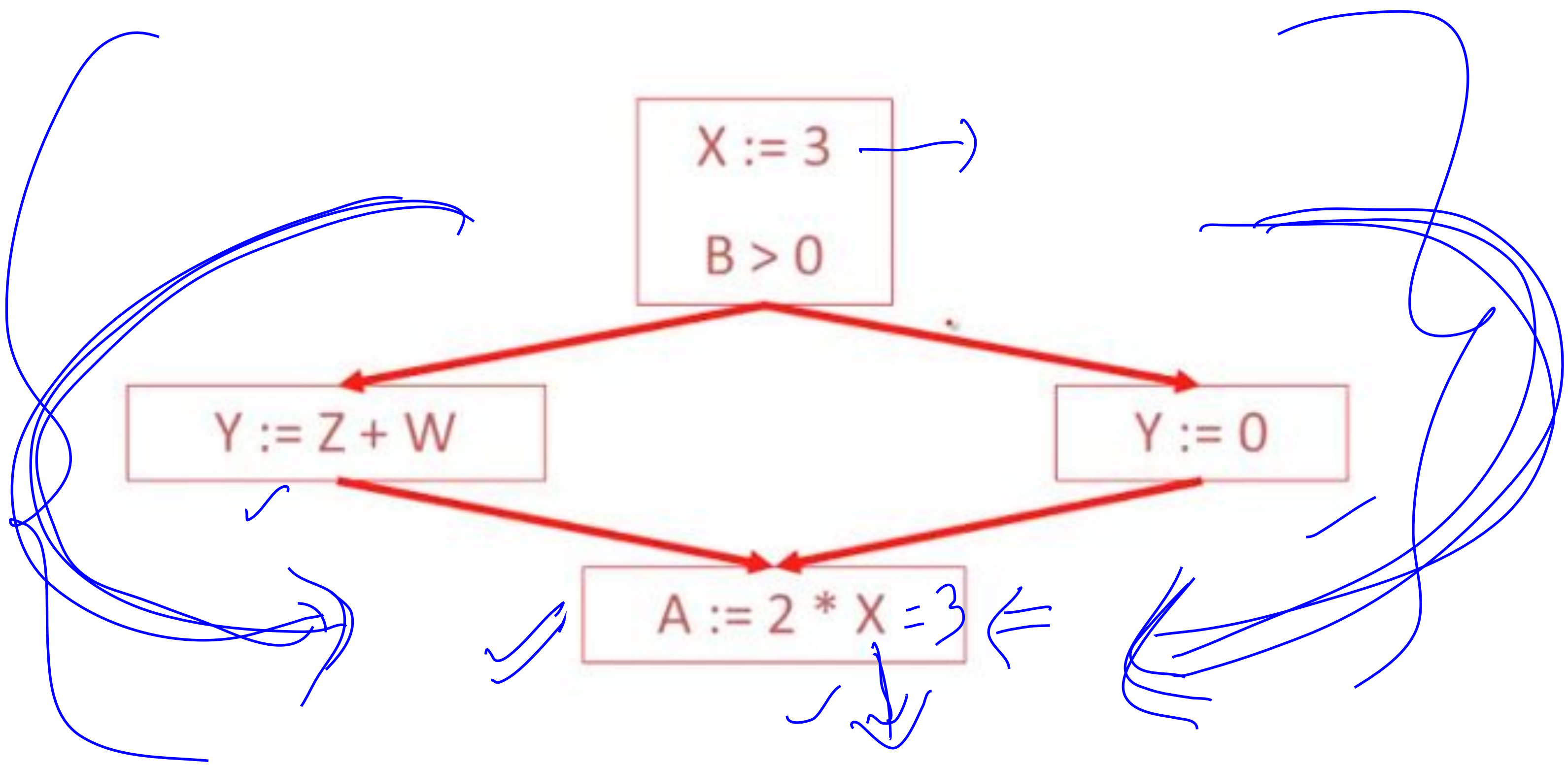
$x = 3$
$y = z * w$
$q = x + y$

$\Rightarrow$

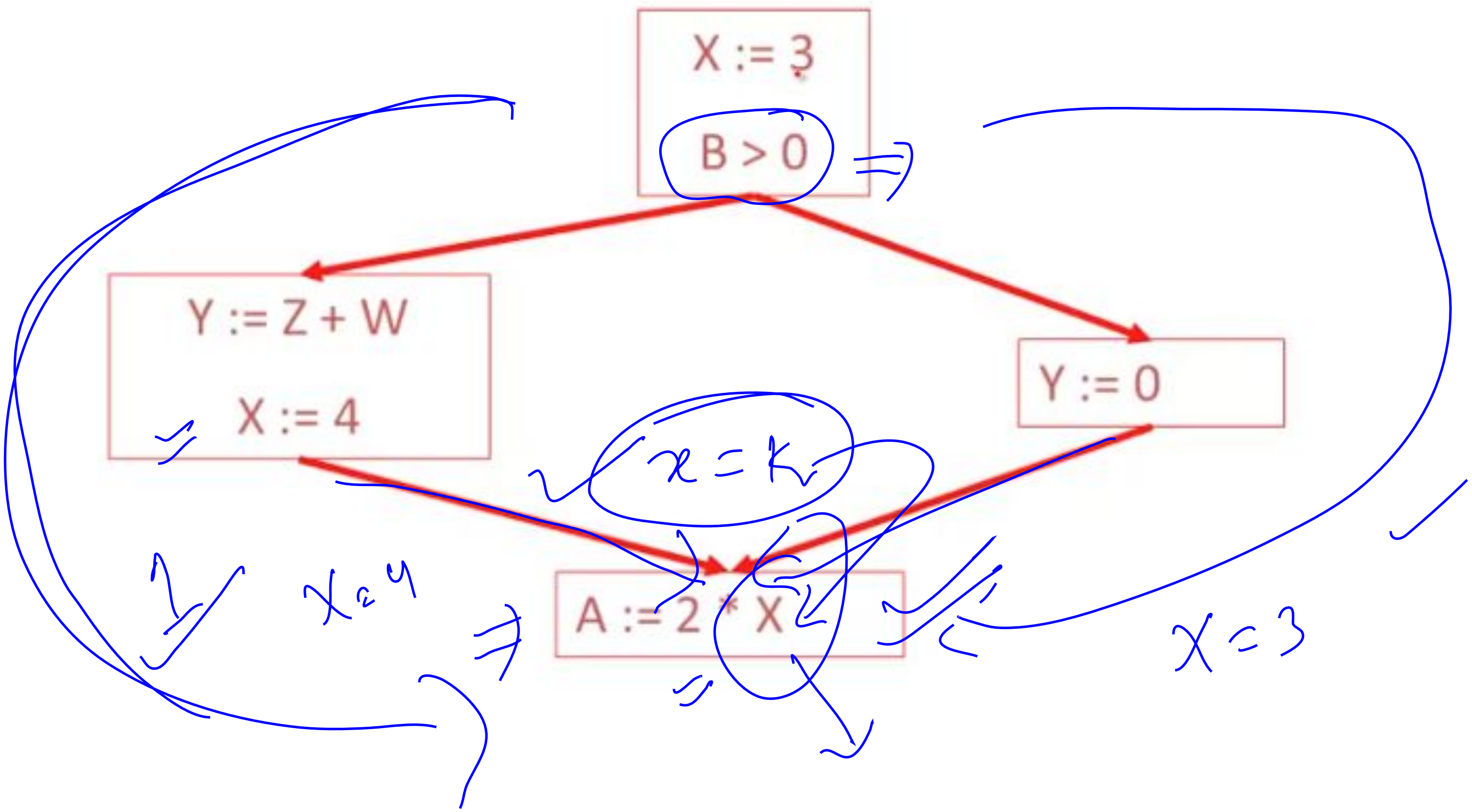
$$\boxed{\begin{aligned} y &= z * w \\ q &= 3 + y \end{aligned}}$$

Dataflow Analysis

- These optimization can be extended to entire control flow graph



Dataflow Analysis



Dataflow Analysis

- To replace a use of x by a constant k we must know:

- on every path to the use of x, the last statement to x is

$$x := k$$

- The correctness condition is not trivial to check.
- “All paths” include paths around loops and through branches of conditionals.
- Checking the condition requires global dataflow analysis
    - An analysis of the entire control flow graphs

~~Copy~~ propagation
Constant

- Global optimization task shares several traits
  - The optimization depends on knowing a property of x at a particular point of program
  - proving x at any point requires knowledge of the entire program
  - it is OK to be conservative. If the optimization requires X to be true, then want to know either
    - X is definitely true
    - don;t know if X is true
    - it is always safe to say don't know

- Global dataflow analysis is a standard technique for solving problems with these characteristics
- Global constant propagation is one example of an optimization that requires global dataflow analysis

Global Constant propagation

- To replace a use of x by a constant k we must know:

- on every path to the use of x, the last assignment to x is

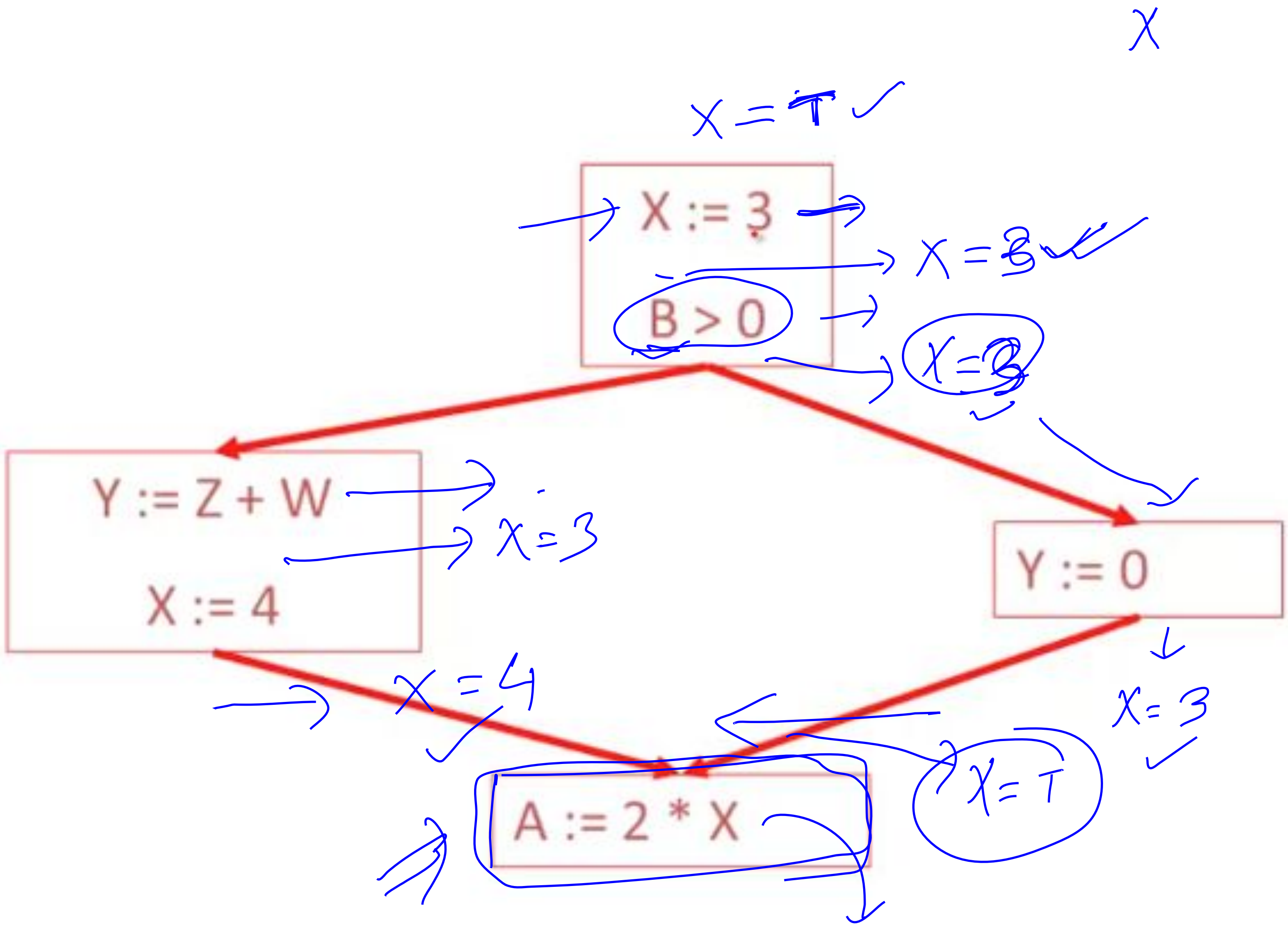
  X := k **

- global constant propagation can be performed at any point where ** holds
- consider the case of computing ** for a single variable X at all points of the program

Global Constant Propagation

- To make the problem precise, we attach one of the following values with X at every program point

Value	Interpretation
$\bot$ = "Bottom"	Statement never executes
C	X = constant
$\top$ = "Top"	X = is not a constant


X
X = T
X := 3
X = 3
B > 0
X = 3
Y := Z + W
X = 3
X := 4
Y := 0
X = 4
X = 3
X = T
A := 2 * X

Global Constant Propagation

- Given global constant information, it is easy to perform the optimization
  - simply inspect X = ? associated with a statement using X
  - If X is constant at that point replace the use of X by that constant.

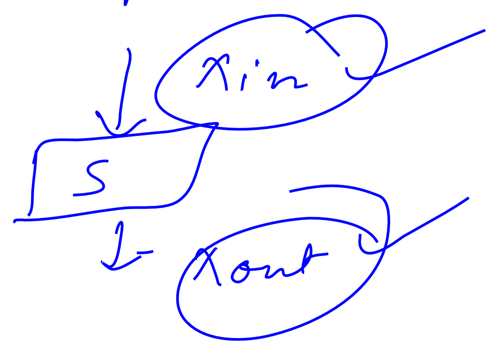
How do we compute the properties X= ?

The analysis of a complicated program can be expressed as a combination of simple rules relating the change in information between adjacent statements

- The idea is to "push" or "transfer" information from statement to the next
- for each statement s, we compute information of x immediately before or after s

$C(x, s, in) =$ value of x before s

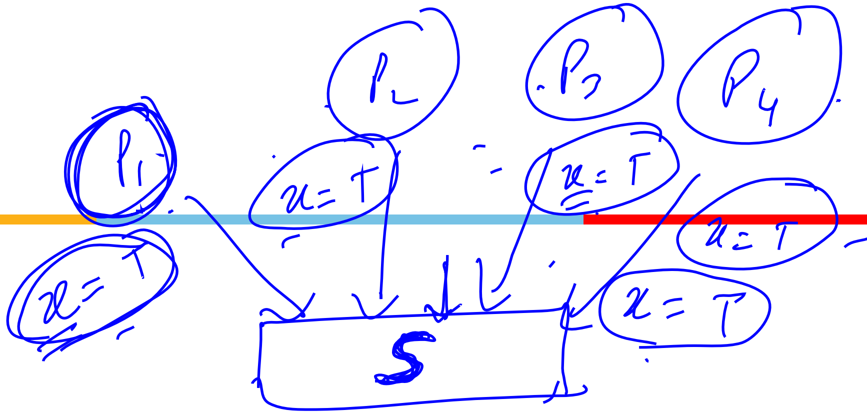
$C(x, s, out) =$ value of x after s

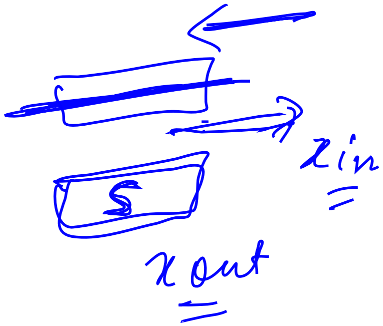


- Define a transfer function that transfer information one statement to other

- In the following rules, let statement s have immediate predecessor statements p1, p2, ….pn.

Rule 1:

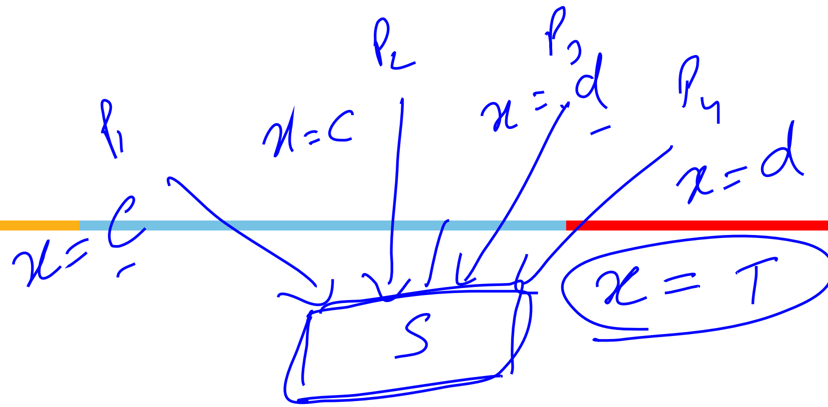




if $C(P_i, x, \text{out}) = T$ for any i

then $C(S, x, \text{in}) = T$

Rule 2:

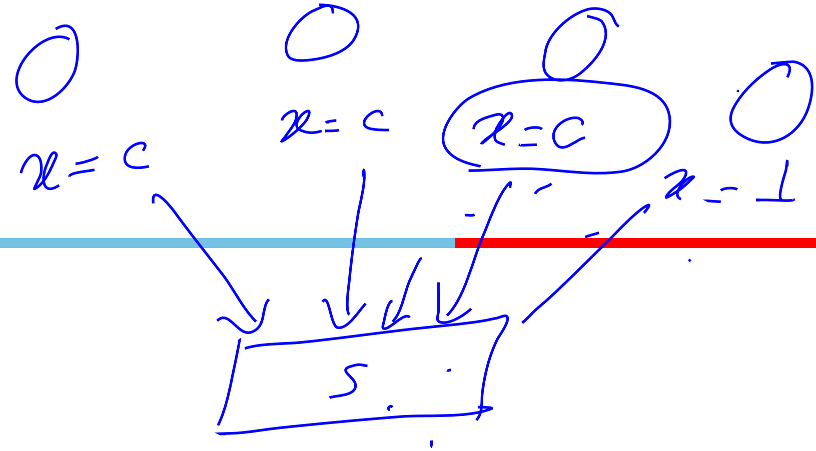


$c \neq d$

if $c(p_i; x, out) = c$ & $c(p_j; x, out) = d$

then $c(S, x, in) = T$

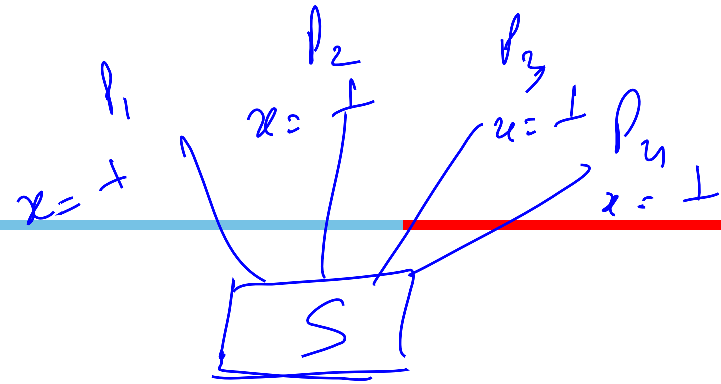
Rule 3:



if $c(P_i, x, out) = c$ or $\perp$ for all i

then $c(S, x, in) = c$

Rule 4:



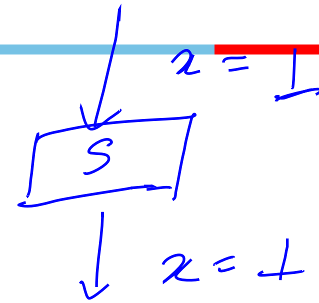
$C(P_i, x, out) = \perp$ for all i

$C(S, x, in) = \perp$

- Rules 1–4 relate out of one statement to the "in" of the next statement.

- Now we need rules that relate the "in" of a statement to the "out" of the same statement.

Rule 5:



if $c(S, x, in) = \perp$ then $c(S, x, out) = \perp$

Rule 6:

$x = T$

$\boxed{x := c}$

$x = c$

$c(x := c, x, \text{out}) = c$

Rule 7:

$x := c$ or $x := T$

$\boxed{x := f(\,)}$

$x := y * g$

$x := T$

$\boxed{C(x := f(x), x, out) = T}$

Rule 8:

$x = k$

$y := \cdot$

$C(y := \ldots, x, out) = k$

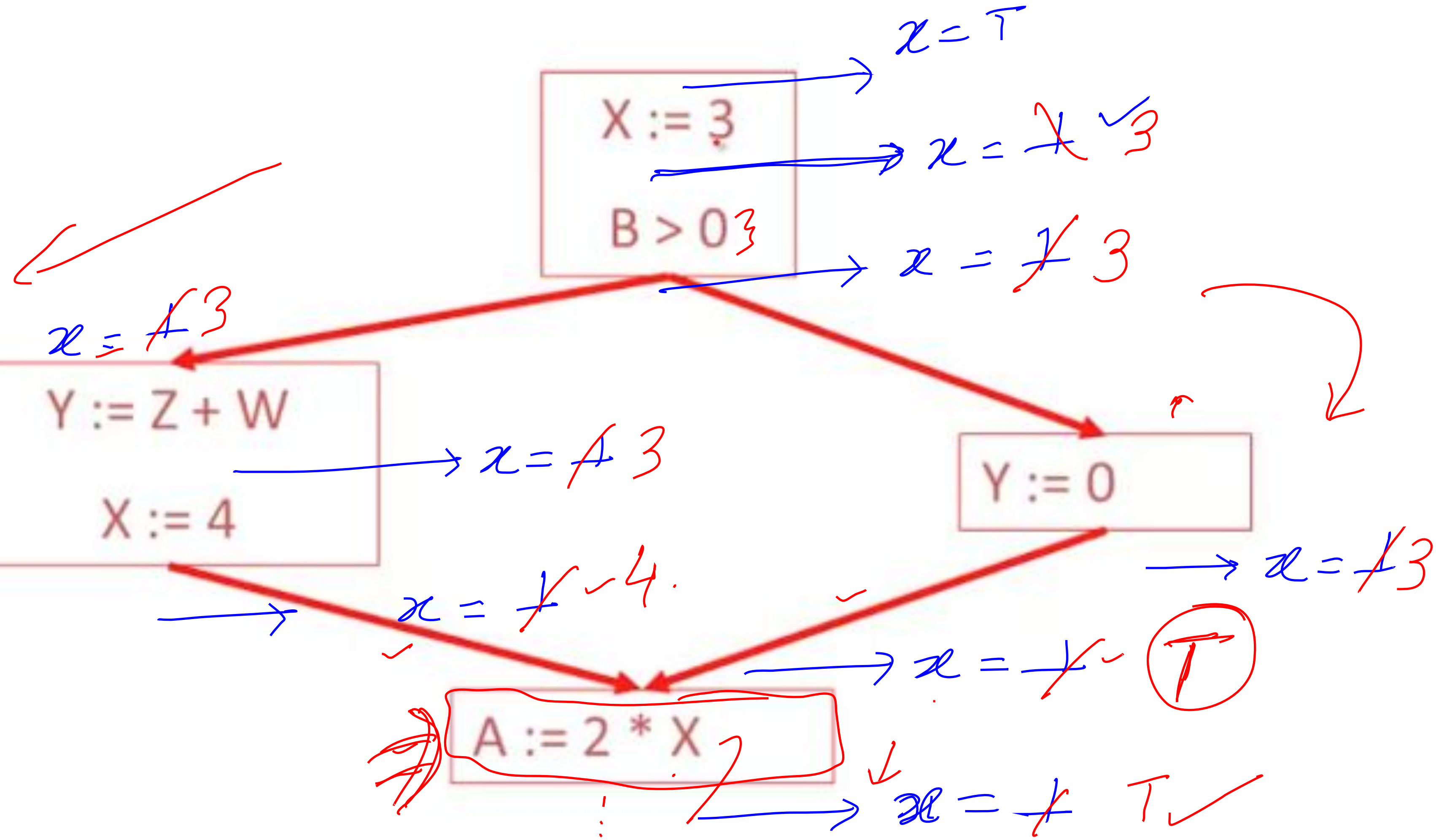
Summary

- For every entry s to the program, set C(s,x,in) =T
- set C(s,x,in) =C(s,x,out) = $\perp$ everywhere else
- repeat until all points satisfy 1-8

  pick s not satisfying 1-8 and update using the appropriate

  rule

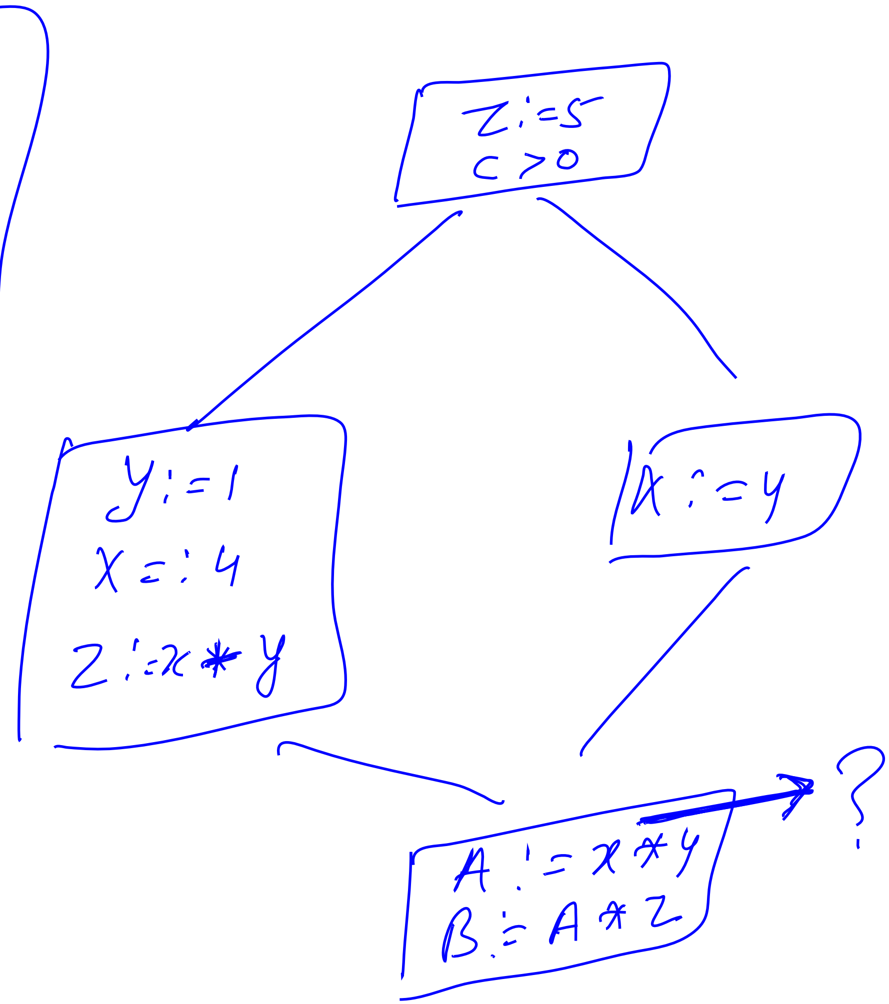
Global constant propagation



Example

X =

X	Y	Z
4	1	5
4	T	5



Thank You!