Constant Propagation

Uday Khedker

(www.cse.iitb.ac.in/~uday)

Department of Computer Science and Engineering,

Indian Institute of Technology, Bombay

Dec 2019

Part 1

About These Slides

Copyright

These slides constitute the lecture notes for CS618 Program Analysis course at IIT Bombay and have been made available as teaching material accompanying the book:

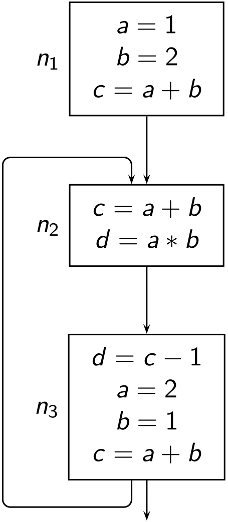
- Uday Khedker, Amitabha Sanyal, and Bageshri Karkare. *Data Flow Analysis: Theory and Practice*. CRC Press (Taylor and Francis Group). 2009.

  (Indian edition published by Ane Books in 2013)

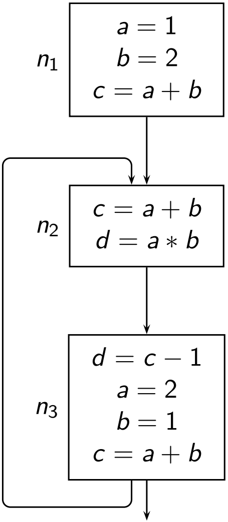
Apart from the above book, some slides are based on the material from the following book

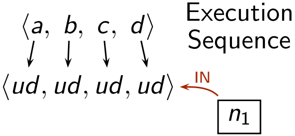
- M. S. Hecht. *Flow Analysis of Computer Programs*. Elsevier North-Holland Inc. 1977.

An Introduction to Constant Propagation

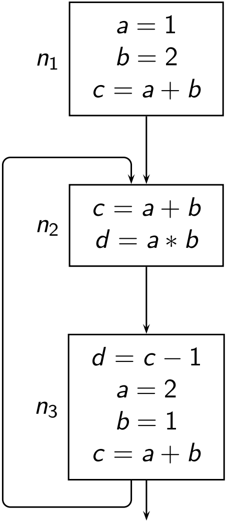


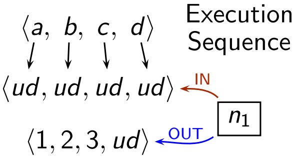
An Introduction to Constant Propagation



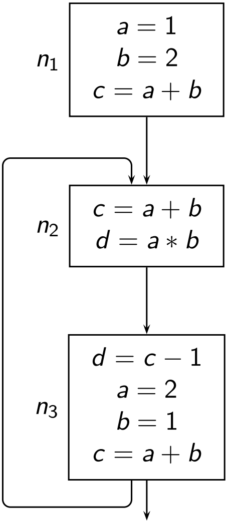


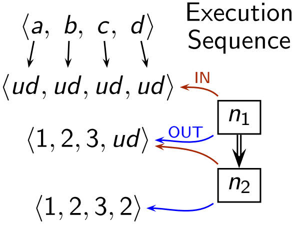
An Introduction to Constant Propagation



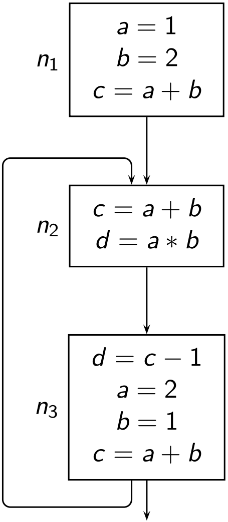


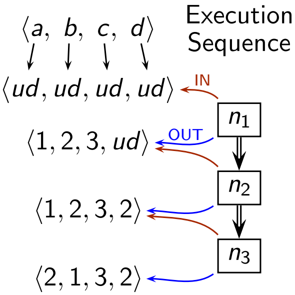
An Introduction to Constant Propagation



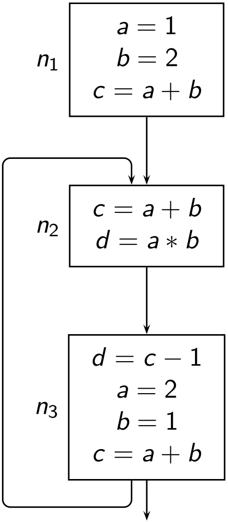


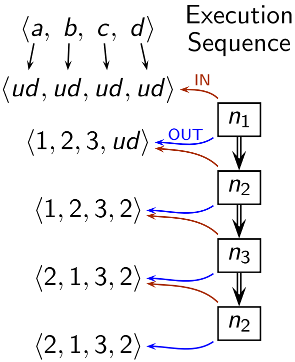
An Introduction to Constant Propagation



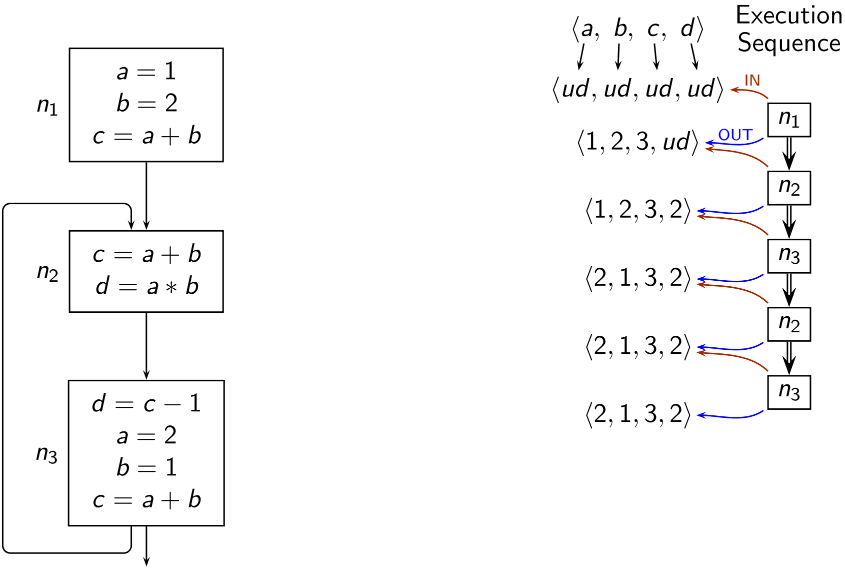


An Introduction to Constant Propagation

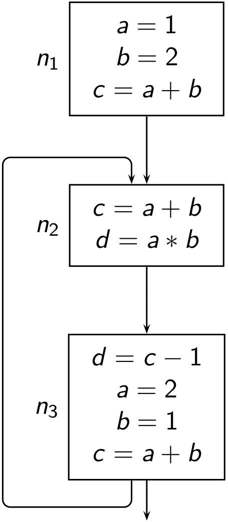


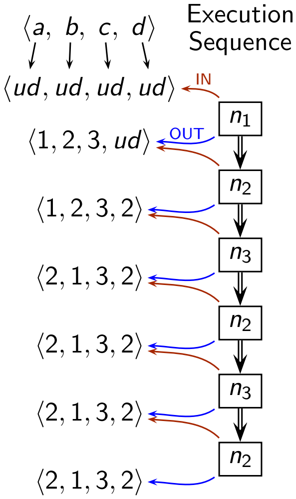


An Introduction to Constant Propagation

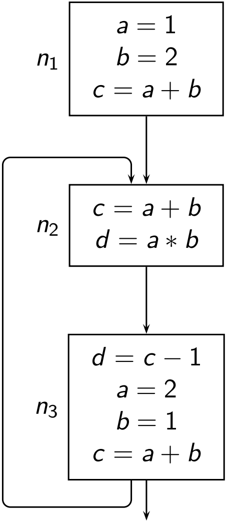


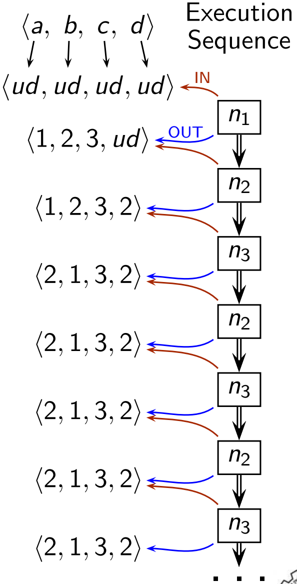
An Introduction to Constant Propagation



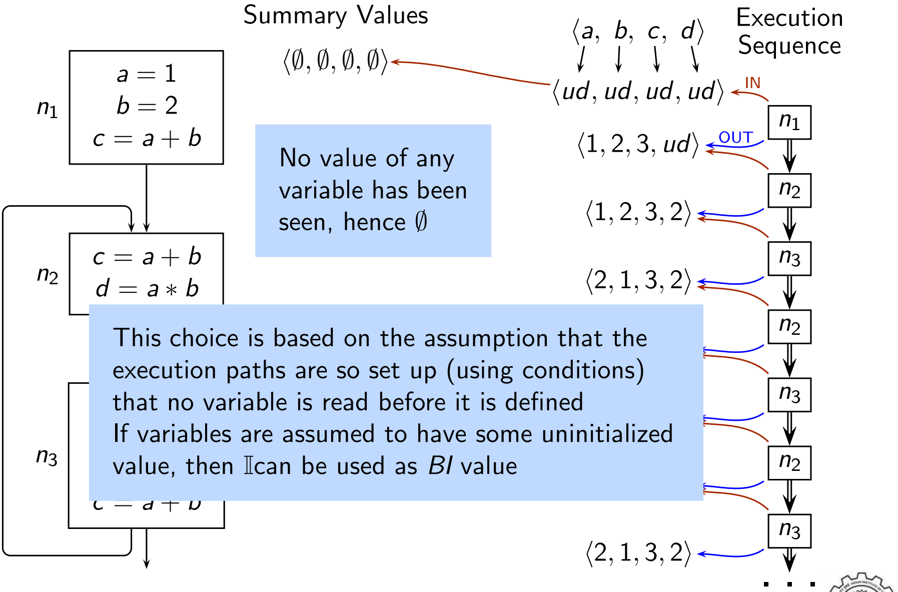


An Introduction to Constant Propagation





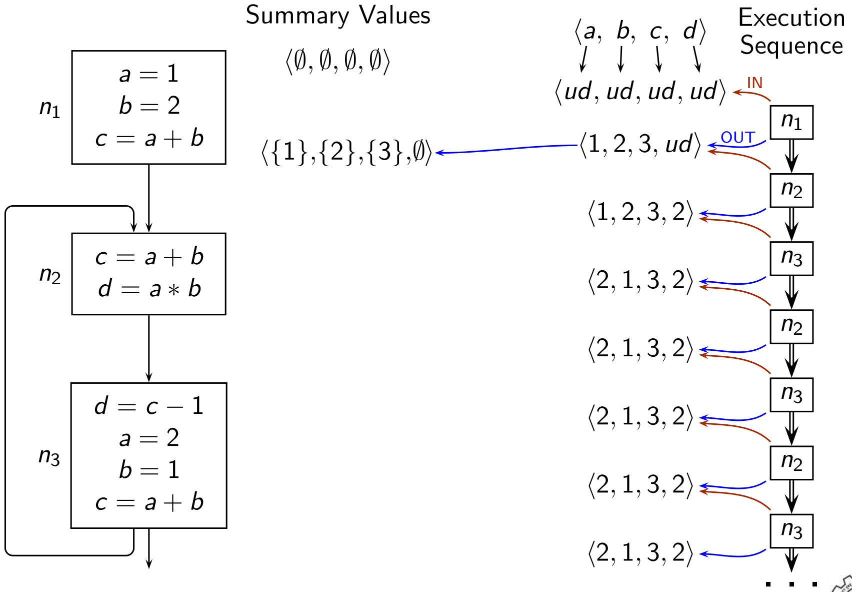
An Introduction to Constant Propagation



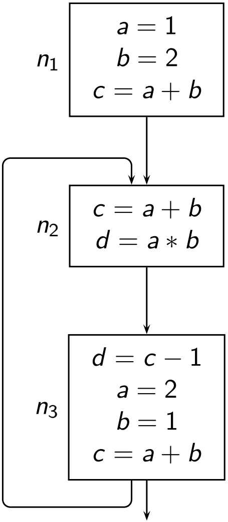
No value of any variable has been seen, hence $\emptyset$

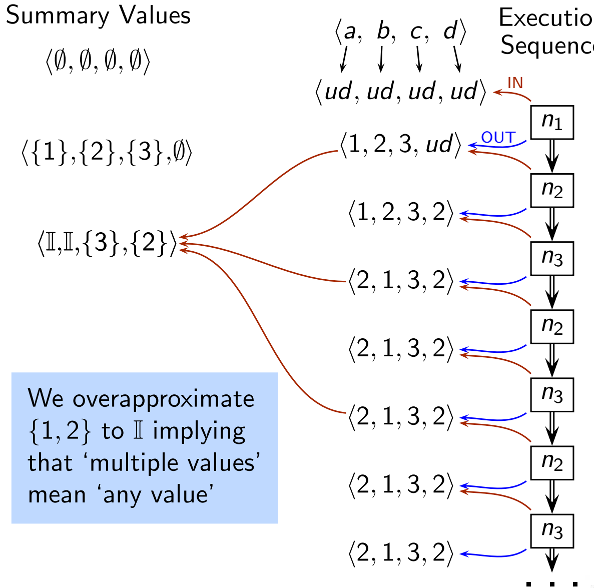
This choice is based on the assumption that the execution paths are so set up (using conditions) that no variable is read before it is defined
If variables are assumed to have some uninitialized value, then $\mathbb{I}$can be used as *BI* value

An Introduction to Constant Propagation



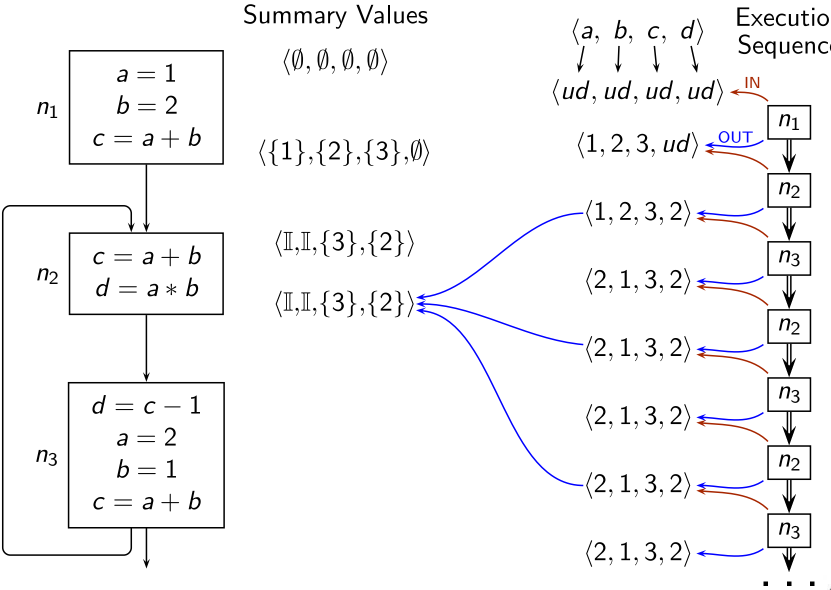
An Introduction to Constant Propagation



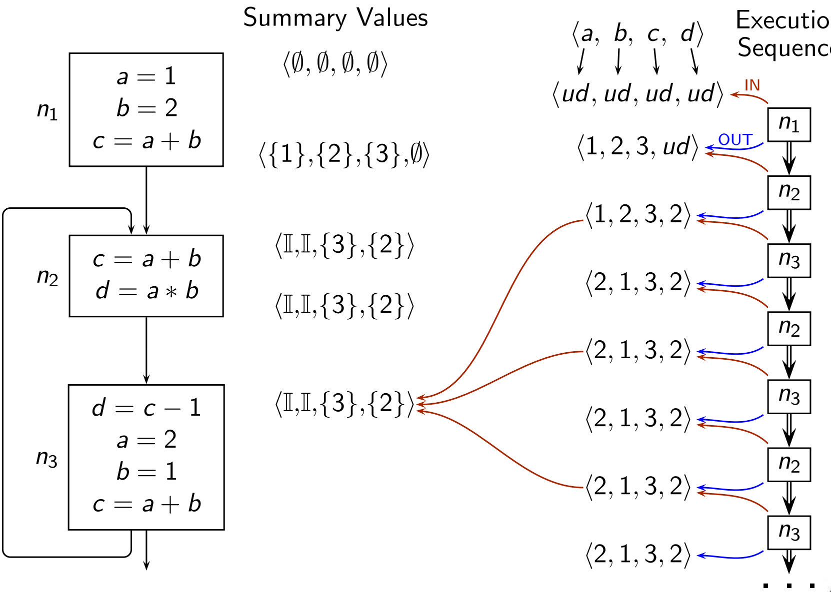


We overapproximate $\{1, 2\}$ to $\mathbb{I}$ implying that 'multiple values' mean 'any value'

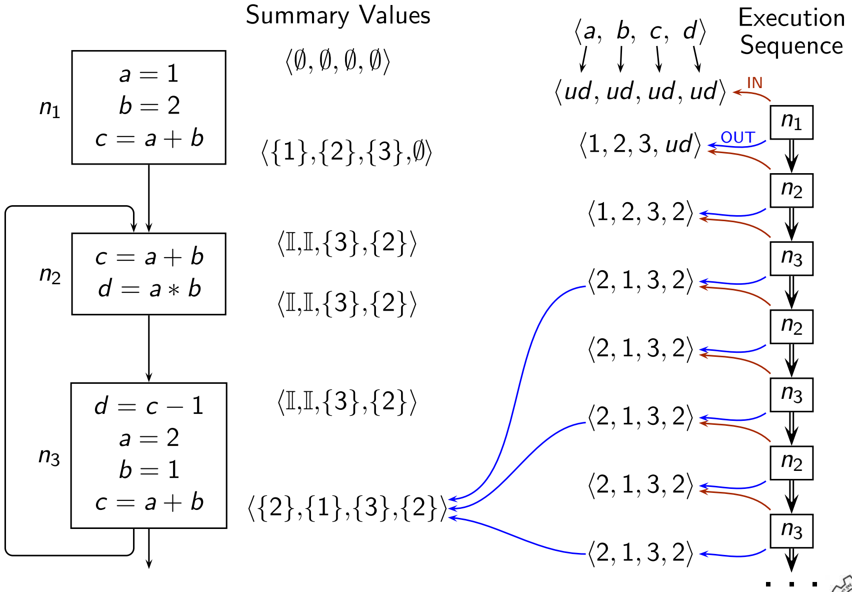
An Introduction to Constant Propagation



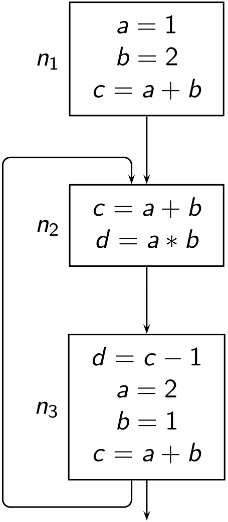
An Introduction to Constant Propagation



An Introduction to Constant Propagation



An Introduction to Constant Propagation



Summary Values

$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$

$\langle \{1\},\{2\},\{3\},\emptyset \rangle$

$\langle \mathbb{I},\mathbb{I},\{3\},\{2\} \rangle$

$\langle \mathbb{I},\mathbb{I},\{3\},\{2\} \rangle$

$\langle \mathbb{I},\mathbb{I},\{3\},\{2\} \rangle$

$\langle \{2\},\{1\},\{3\},\{2\} \rangle$

Desired Solution

Difference #1: Data Flow Values

- Tuples of the form $\langle \eta_1, \eta_2, \ldots, \eta_k \rangle$
  Or sets of pairs (v_i, η_i) or $(v_i \mapsto \eta_i)$ where
  η_i is the data flow value for i^{th} variable

  Unlike live variables analysis, value η_i is not 0 or 1 (i.e. true or false). Instead, it is one of the following:

  - $\emptyset$ indicating that no values is known for v_i
  - $\mathbb{I}$ indicating that variable v_i could have multiple values
  - Set $\{c_1\}$ if the value of v_i is known to be c_1 at compile time

Difference #2: Dependence of Data Flow Values Across Entities

- In (simple) live variables analysis, data flow values of different entities are independent

  Liveness of variable b does not depend on that of any other variable

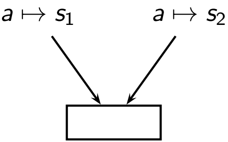
- Given a statement $a = b * c$, can the constantness of a be determined independently of the constantness of b and c?

  No

  This is similar to strong liveness analysis

Difference #3: Merging Information

- Merging the pairs $a \mapsto s_1$ and $a \mapsto s_2$



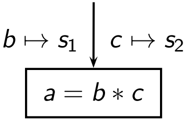
$\sqcap$	$a \mapsto \emptyset$	$a \mapsto \mathbb{I}$	$a \mapsto \{c_1\}$
$a \mapsto \emptyset$	$a \mapsto \emptyset$	$a \mapsto \mathbb{I}$	$a \mapsto \{c_1\}$
$a \mapsto \mathbb{I}$	$a \mapsto \mathbb{I}$	$a \mapsto \mathbb{I}$	$a \mapsto \mathbb{I}$
$a \mapsto \{c_2\}$	$a \mapsto \{c_2\}$	$a \mapsto \mathbb{I}$	If $c_1 = c_2$ $a \mapsto \{c_1\}$ Otherwise $a \mapsto \mathbb{I}$

- The merge (or technically, the "meet") operator is neither $\cap$ nor $\cup$

  What are its properties?

Difference #4: Flow Functions for Constant Propagation

- Flow function for $a = b * c$



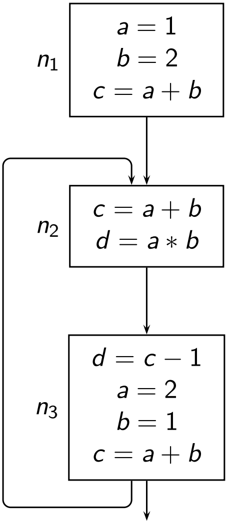
$mult$	$b \mapsto \emptyset$	$b \mapsto \mathbb{I}$	$b \mapsto \{c_1\}$
$c \mapsto \emptyset$	$a \mapsto \emptyset$	$a \mapsto \mathbb{I}$	$a \mapsto \emptyset$
$c \mapsto \mathbb{I}$	$a \mapsto \mathbb{I}$	$a \mapsto \mathbb{I}$	$a \mapsto \mathbb{I}$
$c \mapsto \{c_2\}$	$a \mapsto \emptyset$	$a \mapsto \mathbb{I}$	$a \mapsto \{c_1 * c_2\}$

- This cannot be expressed in the form

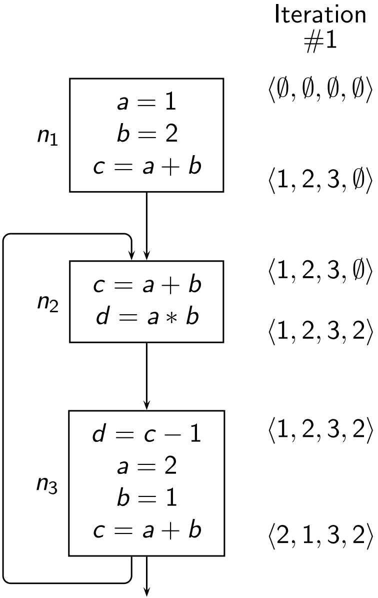
$$f_n(X) = \text{Gen}_n \cup (X - \text{Kill}_n)$$

where Gen_n and Kill_n are constant effects of block n

Difference #5: Solution Computed by Iterative Method

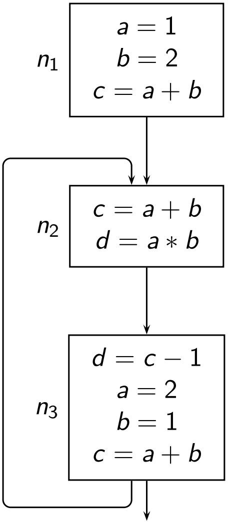


Difference #5: Solution Computed by Iterative Method



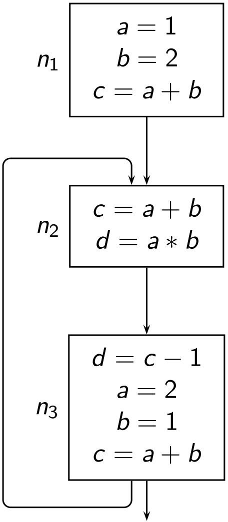
For convenience, we omit the braces for singleton sets

Difference #5: Solution Computed by Iterative Method



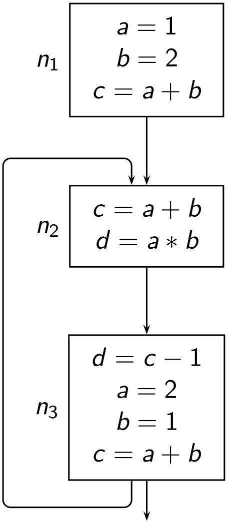
Iteration #1	Iteration #2
$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$	$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$
$\langle 1, 2, 3, \emptyset \rangle$	$\langle 1, 2, 3, \emptyset \rangle$
$\langle 1, 2, 3, \emptyset \rangle$	$\langle \mathbb{I}, \mathbb{I}, 3, 2 \rangle$
$\langle 1, 2, 3, 2 \rangle$	$\langle \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I} \rangle$
$\langle 1, 2, 3, 2 \rangle$	$\langle \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I} \rangle$
$\langle 2, 1, 3, 2 \rangle$	$\langle 2, 1, 3, \mathbb{I} \rangle$

Difference #5: Solution Computed by Iterative Method



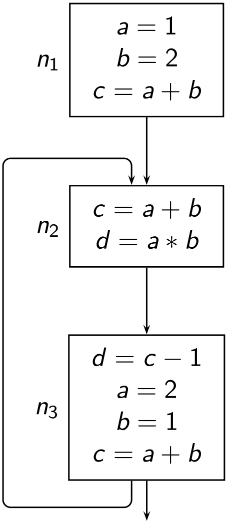
	Iteration #1	Iteration #2	Iteration #3
n_1 in	$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$	$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$	$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$
n_1 out	$\langle 1, 2, 3, \emptyset \rangle$	$\langle 1, 2, 3, \emptyset \rangle$	$\langle 1, 2, 3, \emptyset \rangle$
n_2 in	$\langle 1, 2, 3, \emptyset \rangle$	$\langle \mathbb{I}, \mathbb{I}, 3, 2 \rangle$	$\langle \mathbb{I}, \mathbb{I}, 3, \mathbb{I} \rangle$
n_2 out	$\langle 1, 2, 3, 2 \rangle$	$\langle \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I} \rangle$	$\langle \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I} \rangle$
n_3 in	$\langle 1, 2, 3, 2 \rangle$	$\langle \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I} \rangle$	$\langle \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I} \rangle$
n_3 out	$\langle 2, 1, 3, 2 \rangle$	$\langle 2, 1, 3, \mathbb{I} \rangle$	$\langle 2, 1, 3, \mathbb{I} \rangle$

Difference #5: Solution Computed by Iterative Method



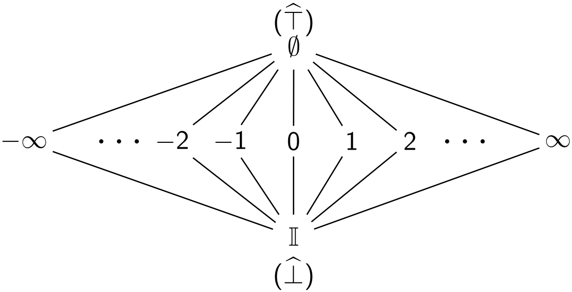
	Iteration #1	Iteration #2	Iteration #3	Desired solution
n_1 entry	$\langle\emptyset,\emptyset,\emptyset,\emptyset\rangle$	$\langle\emptyset,\emptyset,\emptyset,\emptyset\rangle$	$\langle\emptyset,\emptyset,\emptyset,\emptyset\rangle$	$\langle\emptyset,\emptyset,\emptyset,\emptyset\rangle$
n_1 exit	$\langle 1,2,3,\emptyset\rangle$	$\langle 1,2,3,\emptyset\rangle$	$\langle 1,2,3,\emptyset\rangle$	$\langle 1,2,3,\emptyset\rangle$
n_2 entry	$\langle 1,2,3,\emptyset\rangle$	$\langle \mathbb{I},\mathbb{I},3,2\rangle$	$\langle \mathbb{I},\mathbb{I},3,\mathbb{I}\rangle$	$\langle \mathbb{I},\mathbb{I},3,2\rangle$
n_2 exit	$\langle 1,2,3,2\rangle$	$\langle \mathbb{I},\mathbb{I},\mathbb{I},\mathbb{I}\rangle$	$\langle \mathbb{I},\mathbb{I},\mathbb{I},\mathbb{I}\rangle$	$\langle \mathbb{I},\mathbb{I},3,2\rangle$
n_3 entry	$\langle 1,2,3,2\rangle$	$\langle \mathbb{I},\mathbb{I},\mathbb{I},\mathbb{I}\rangle$	$\langle \mathbb{I},\mathbb{I},\mathbb{I},\mathbb{I}\rangle$	$\langle \mathbb{I},\mathbb{I},3,2\rangle$
n_3 exit	$\langle 2,1,3,2\rangle$	$\langle 2,1,3,\mathbb{I}\rangle$	$\langle 2,1,3,\mathbb{I}\rangle$	$\langle 2,1,3,2\rangle$

Difference #5: Solution Computed by Iterative Method



Iteration #1	Iteration #2	Iteration #3	Desired solution
$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$	$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$	$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$	$\langle \emptyset, \emptyset, \emptyset, \emptyset \rangle$
$\langle 1, 2, 3, \emptyset \rangle$	$\langle 1, 2, 3, \emptyset \rangle$	$\langle 1, 2, 3, \emptyset \rangle$	$\langle 1, 2, 3, \emptyset \rangle$
$\langle 1, 2, 3, \emptyset \rangle$	$\langle \mathbb{I}, \mathbb{I}, 3, 2 \rangle$	$\langle \mathbb{I}, \mathbb{I}, 3, \mathbb{I} \rangle$	$\langle \mathbb{I}, \mathbb{I}, 3, 2 \rangle$
$\langle 1, 2, 3, 2 \rangle$	$\langle \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I} \rangle$	$\langle \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I} \rangle$	$\langle \mathbb{I}, \mathbb{I}, 3, 2 \rangle$
$\langle 1, 2, 3, 2 \rangle$	$\langle \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I} \rangle$	$\langle \mathbb{I}, \mathbb{I}, \mathbb{I}, \mathbb{I} \rangle$	$\langle \mathbb{I}, \mathbb{I}, 3, 2 \rangle$
$\langle 2, 1, 3, 2 \rangle$	$\langle 2, 1, 3, \mathbb{I} \rangle$	$\langle 2, 1, 3, \mathbb{I} \rangle$	$\langle 2, 1, 3, 2 \rangle$

Component Lattice for Integer Constant Propagation



$\widehat{\sqcap}$	$\langle v, \widehat{\top} \rangle$	$\langle v, \widehat{\bot} \rangle$	$\langle v, c_1 \rangle$
$\langle v, \widehat{\top} \rangle$	$\langle v, \widehat{\top} \rangle$	$\langle v, \widehat{\bot} \rangle$	$\langle v, c_1 \rangle$
$\langle v, \widehat{\bot} \rangle$	$\langle v, \widehat{\bot} \rangle$	$\langle v, \widehat{\bot} \rangle$	$\langle v, \widehat{\bot} \rangle$
$\langle v, c_2 \rangle$	$\langle v, c_2 \rangle$	$\langle v, \widehat{\bot} \rangle$	If $c_1 = c_2$ then $\langle v, c_1 \rangle$ else $\langle v, \widehat{\bot} \rangle$

Overall Lattice for Integer Constant Propagation

- In_n/Out_n values are mappings $\mathbb{V}\text{ar} \to \widehat{L}$: $In_n, Out_n \in \mathbb{V}\text{ar} \to \widehat{L}$
- Overall lattice L is a set of mappings $\mathbb{V}\text{ar} \to \widehat{L}$: $L = \mathbb{V}\text{ar} \to \widehat{L}$
- $\sqcap$ and $\widehat{\sqcap}$ get defined by $\sqsubseteq$ and $\widehat{\sqsubseteq}$
  - Partial order is restricted to data flow values of the same variable
    Data flow values of different variables are incomparable

$$(x, v_1) \sqsubseteq (y, v_2) \;\Leftrightarrow\; x = y \wedge v_1 \widehat{\sqsubseteq} v_2$$

$$OR \qquad x \mapsto v_1 \sqsubseteq y \mapsto v_2 \;\Leftrightarrow\; x = y \wedge v_1 \widehat{\sqsubseteq} v_2$$

  - For meet operation, we assume that X is a total function
    Partial functions are made total by using $\widehat{\top}$value

$$X \sqcap Y = \big\{(x, v_1 \widehat{\sqcap} v_2) \mid (x, v_1) \in X, (x, v_2) \in Y\big\}$$

$$OR \qquad X \sqcap Y = \big\{x \mapsto v_1 \widehat{\sqcap} v_2 \mid x \mapsto v_1 \in X, x \mapsto v_2 \in Y\big\}$$

Notations for Mappings as Data Flow Values

Accessing and manipulating a mapping $X \subseteq A \to B$

- $X(a)$ denotes the image of $a \in A$
  $X(a) \in B$
- $X[a \mapsto v]$ changes the image of a in X to v

$$X[a \mapsto v] = (X - \{(a, u) \mid u \in B\}) \cup \{(a, v)\}$$

Defining Data Flow Equations for Constant Propagation

$$
In_n = \begin{cases} BI = \left\{\left(y, \widehat{\top}\right) \mid y \in \mathbb{V}\text{ar}\right\} & n = Start \\ \displaystyle\bigsqcap_{p \in pred(n)} Out_p & \text{otherwise} \end{cases}
$$

$$
Out_n = f_n(In_n)
$$

$$
f_n(X) = \begin{cases} X\left[y \mapsto c\right] & n \text{ is } y = c, y \in \mathbb{V}\text{ar}, c \in \mathbb{C}\text{onst} \\ X\left[y \mapsto \widehat{\bot}\right] & n \text{ is } input(y), y \in var \\ X\left[y \mapsto X(z)\right] & n \text{ is } y = z, y \in \mathbb{V}\text{ar}, z \in \mathbb{V}\text{ar} \\ X\left[y \mapsto eval(e, X)\right] & n \text{ is } y = e, y \in \mathbb{V}\text{ar}, e \in \mathbb{E}\text{xpr} \\ X & \text{otherwise} \end{cases}
$$

Defining Data Flow Equations for Constant Propagation

$$In_n = \begin{cases} BI = \left\{ \left(y, \widehat{\top}\right) \mid y \in \mathbb{V}\text{ar} \right\} & n = Start \\ \displaystyle\bigsqcap_{p \in pred(n)} Out_p & \text{otherwise} \end{cases}$$

$$Out_n = f_n(In_n)$$

$$f_n(X) = \begin{cases} X\left[y \mapsto c\right] & n \text{ is } y = c, y \in \mathbb{V}\text{ar}, c \in \mathbb{C}\text{onst} \\ X\left[y \mapsto \widehat{\bot}\right] & n \text{ is } input(y), y \in var \\ X\left[y \mapsto X(z)\right] & n \text{ is } y = z, y \in \mathbb{V}\text{ar}, z \in \mathbb{V}\text{ar} \\ X\left[y \mapsto eval(e, X)\right] & n \text{ is } y = e, y \in \mathbb{V}\text{ar}, e \in \mathbb{E}\text{xpr} \\ X & \text{otherwise} \end{cases}$$

$$eval(e, X) = \begin{cases} \widehat{\bot} & a \in Opd(e) \cap \mathbb{V}\text{ar}, X(a) = \widehat{\bot} \\ \widehat{\top} & a \in Opd(e) \cap \mathbb{V}\text{ar}, X(a) = \widehat{\top} \\ -X(a) & e \text{ is } -a \\ X(a) \oplus X(b) & e \text{ is } a \oplus b \end{cases}$$

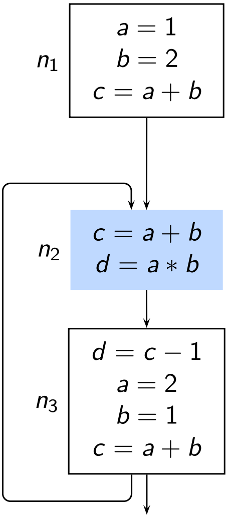
Tutorial Problem for Constant Propagation

- Construct a CFG corresponding to this program and perform constant propagation

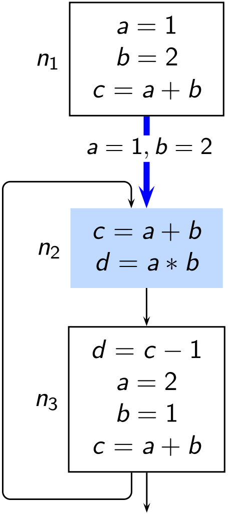
```
int f ()
{   int a, b, c, d;
    for (i=0; i<n; i++)
    {   if (i==m+3)
            a=b+1;
        else if (i==m+2)
            b=c+1;
        else if (i==m+1)
            c=d+1;
        else if (i==m)
            d=2;
    }
}
```

- Repeat the analysis assuming *BI* to be $\widehat{\bot}$ for all variables

Non-Distributivity of Constant Propagation

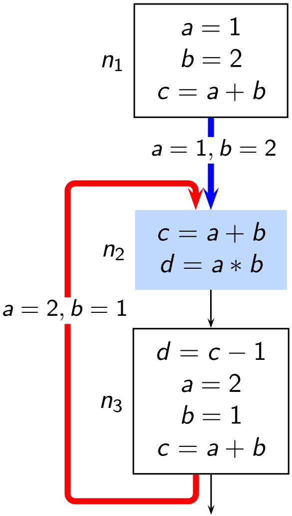


Non-Distributivity of Constant Propagation



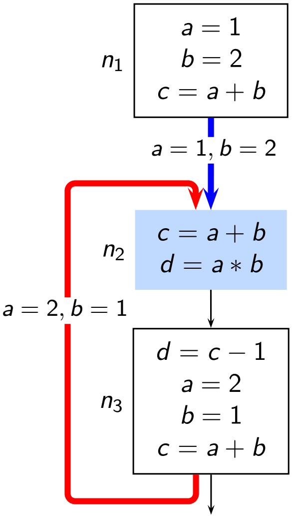
- $x = \langle 1, 2, 3, \widehat{\top} \rangle$ (Along $Out_{n_1} \rightarrow In_{n_2}$)

Non-Distributivity of Constant Propagation



- $x = \langle 1, 2, 3, \widehat{\top} \rangle$ (Along $Out_{n_1} \rightarrow In_{n_2}$)
- $y = \langle 2, 1, 3, 2 \rangle$ (Along $Out_{n_3} \rightarrow In_{n_2}$)

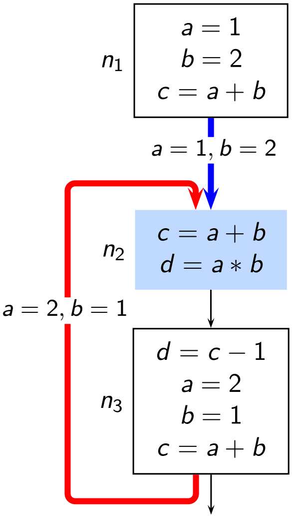
Non-Distributivity of Constant Propagation



- $x = \langle 1, 2, 3, \widehat{\top} \rangle$ (Along $Out_{n_1} \rightarrow In_{n_2}$)
- $y = \langle 2, 1, 3, 2 \rangle$ (Along $Out_{n_3} \rightarrow In_{n_2}$)
- Function application for block n_2 before merging

$$
\begin{aligned}
f(x) \sqcap f(y) &= f(\langle 1, 2, 3, \widehat{\top} \rangle) \sqcap f(\langle 2, 1, 3, 2 \rangle) \\
&= \langle 1, 2, 3, 2 \rangle \sqcap \langle 2, 1, 3, 2 \rangle \\
&= \langle \widehat{\bot}, \widehat{\bot}, 3, 2 \rangle
\end{aligned}
$$

Non-Distributivity of Constant Propagation



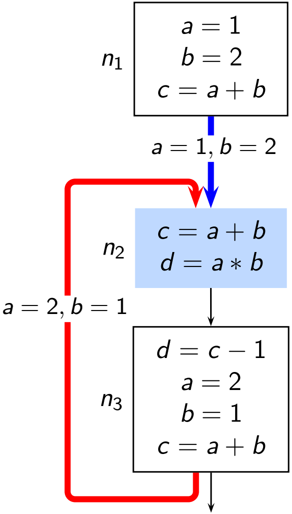
- $x = \langle 1, 2, 3, \widehat{\top} \rangle$ (Along $Out_{n_1} \rightarrow In_{n_2}$)
- $y = \langle 2, 1, 3, 2 \rangle$ (Along $Out_{n_3} \rightarrow In_{n_2}$)
- Function application for block n_2 before merging

$$\begin{aligned} f(x) \sqcap f(y) &= f(\langle 1, 2, 3, \widehat{\top} \rangle) \sqcap f(\langle 2, 1, 3, 2 \rangle) \\ &= \langle 1, 2, 3, 2 \rangle \sqcap \langle 2, 1, 3, 2 \rangle \\ &= \langle \widehat{\bot}, \widehat{\bot}, 3, 2 \rangle \end{aligned}$$

- Function application for block n_2 after merging

$$\begin{aligned} f(x \sqcap y) &= f(\langle 1, 2, 3, \widehat{\top} \rangle \sqcap \langle 2, 1, 3, 2 \rangle) \\ &= f(\langle \widehat{\bot}, \widehat{\bot}, 3, 2 \rangle) \\ &= \langle \widehat{\bot}, \widehat{\bot}, \widehat{\bot}, \widehat{\bot} \rangle \end{aligned}$$

Non-Distributivity of Constant Propagation



- $x = \langle 1, 2, 3, \widehat{\top} \rangle$ (Along $Out_{n_1} \rightarrow In_{n_2}$)
- $y = \langle 2, 1, 3, 2 \rangle$ (Along $Out_{n_3} \rightarrow In_{n_2}$)
- Function application for block n_2 before merging

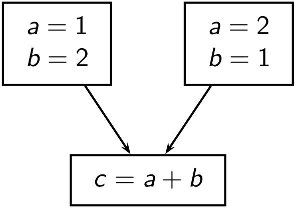
$$\begin{aligned} f(x) \sqcap f(y) &= f(\langle 1, 2, 3, \widehat{\top} \rangle) \sqcap f(\langle 2, 1, 3, 2 \rangle) \\ &= \langle 1, 2, 3, 2 \rangle \sqcap \langle 2, 1, 3, 2 \rangle \\ &= \langle \widehat{\bot}, \widehat{\bot}, 3, 2 \rangle \end{aligned}$$

- Function application for block n_2 after merging

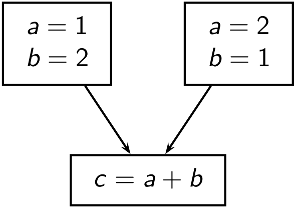
$$\begin{aligned} f(x \sqcap y) &= f(\langle 1, 2, 3, \widehat{\top} \rangle \sqcap \langle 2, 1, 3, 2 \rangle) \\ &= f(\langle \widehat{\bot}, \widehat{\bot}, 3, 2 \rangle) \\ &= \langle \widehat{\bot}, \widehat{\bot}, \widehat{\bot}, \widehat{\bot} \rangle \end{aligned}$$

- $f(x \sqcap y) \sqsubset f(x) \sqcap f(y)$

Why is Constant Propagation Non-Distributive?



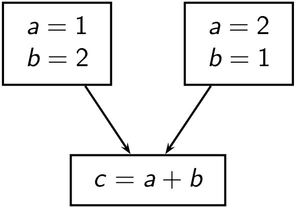
Why is Constant Propagation Non-Distributive?



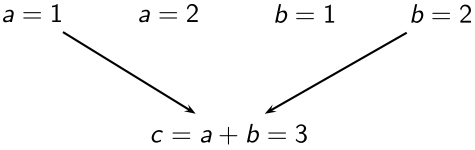
Possible combinations due to merging

$a = 1$ $\quad$ $a = 2$ $\quad$ $b = 1$ $\quad$ $b = 2$

Why is Constant Propagation Non-Distributive?

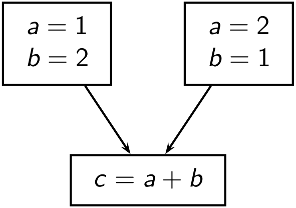


Possible combinations due to merging

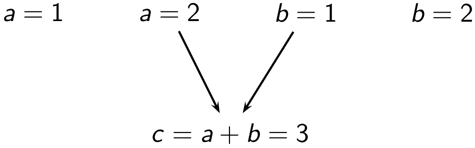


- Correct combination.

Why is Constant Propagation Non-Distributive?

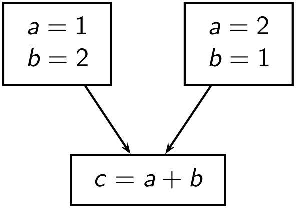


Possible combinations due to merging

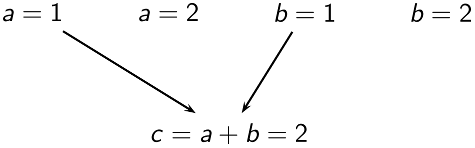


- Correct combination.

Why is Constant Propagation Non-Distributive?

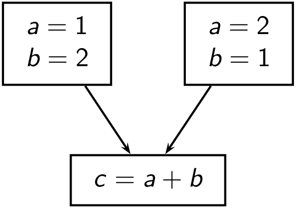


Possible combinations due to merging

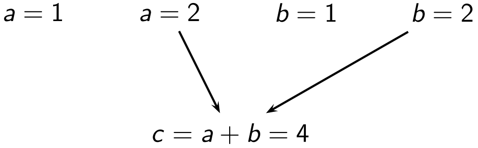


- Wrong combination.
- Mutually exclusive information.
- No execution path along which this information holds.

Why is Constant Propagation Non-Distributive?



Possible combinations due to merging



- Wrong combination.
- Mutually exclusive information.
- No execution path along which this information holds.