Reaching Definitions and u-d Chaining

M.B.Chandak

hodcs@rknec.edu

Reaching Definitions

- A definition of any variable is “killed” if between two points along the path, there is a re-assignment.
- A definition “d” reaches point “p” if there is a path from the point immediately following “d” to “p” such that “d” is not killed along the path.
- For example: a=b+c in block B1 and a=d+e in block b4 and there is path between B1 and B4, means definition of “a” in B1 does not reaches B4.
- To compute reaching definitions, IN-OUT Analysis is performed.
- Reaching definition for block B1 is represented as IN[B1].
- Purpose: To perform optimization by computing loop invariant

Algorithm

Step 1:

For each block B do IN[B] = ϕ and OUT[B] = GEN[B]

CHANGE=TRUE

While CHANGE do

{

CHANGE = FALSE

For each block B do

{

IN [B] = Superunion OUT[P] //P = predecessors

OLDOUT = OUT[B]

OUT[B] = GEN[B] union [IN[B] – KILL[B]]

If(OUT[B] ≠ OLDOUT)

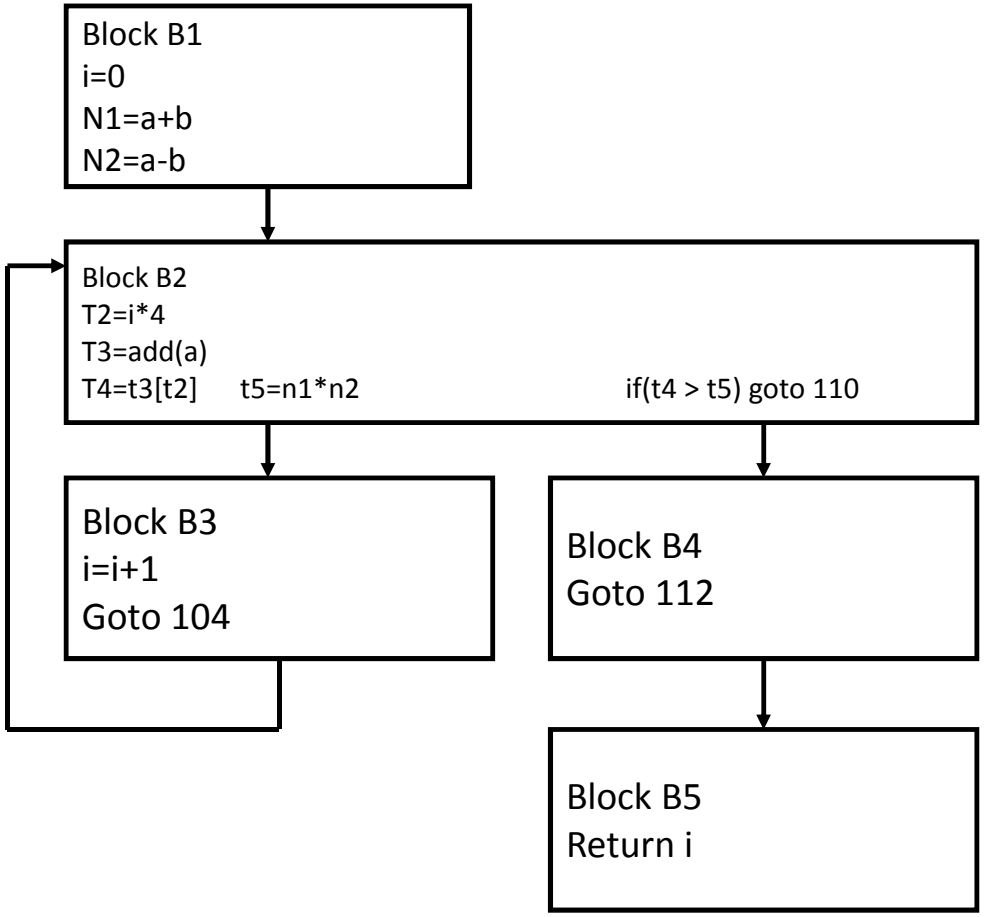
CHANGE=TRUE

}

}

Example-1: IN-OUT analysis and Reaching Definitions

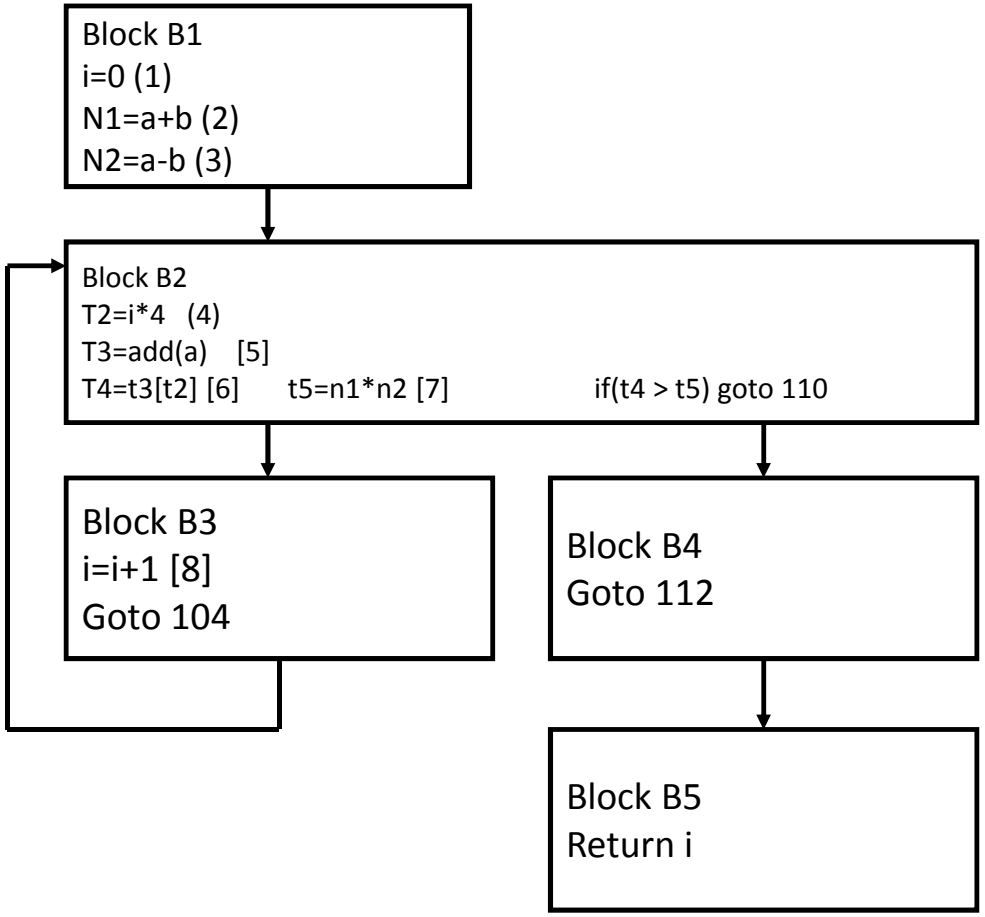
Label	Code
101	i=0
102	N1=a+b
103	N2=a-b
104	T2=i*4
105	T3 = add(a)
106	T4=t3[t2]
107	T5=n1*n2
108	If (t4 > t5) goto 110
109	Goto 112
110	i=i+1
111	Goto 104
112	Return i



Example-1: IN-OUT analysis and Reaching Definitions

BLOCK	PREDCE'R	GEN[B]	KILL[B]
B1	NIL	1,2,3	8
B2	B1,B3	4,5,6,7	NIL
B3	B2	8	1
B4	B2	Nil	NIL
B5	B4	Nil	NIL

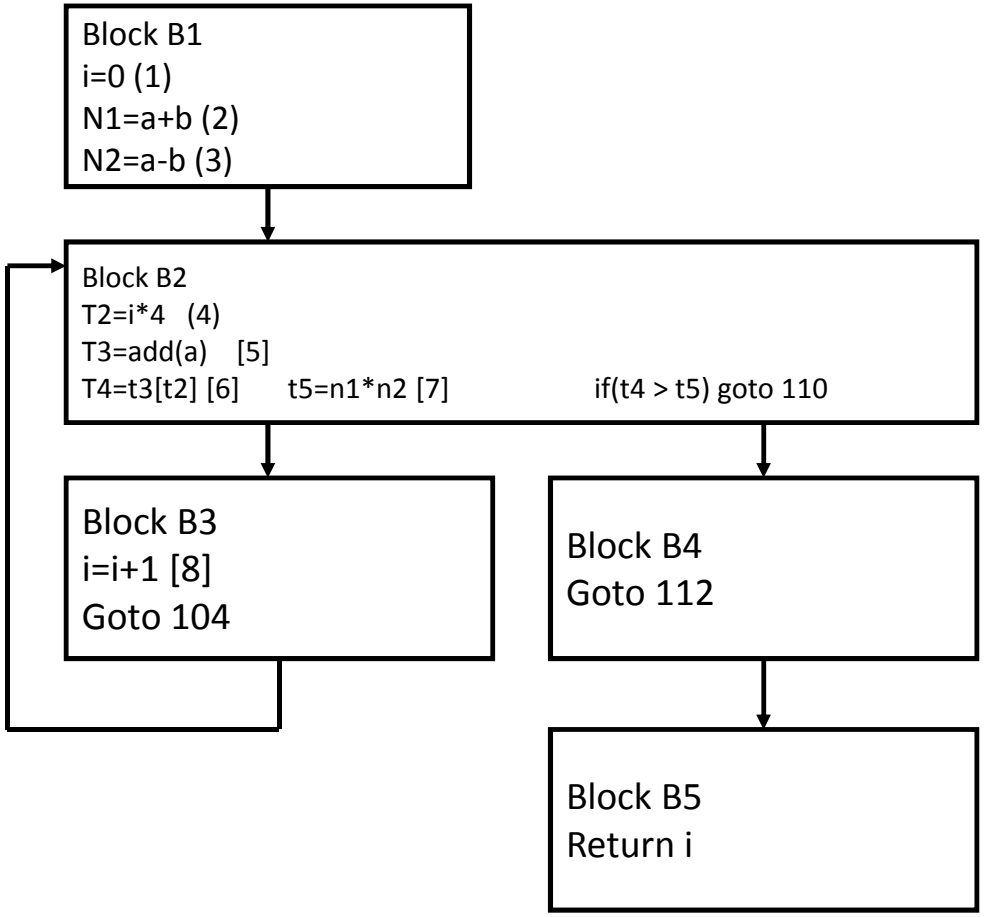
BLOCK	IN[B]	OUT[B]	IN[B]	OUT[B]
B1	Nil	1,2,3	Nil	1,2,3
B2	Nil	4,5,6,7	1,2,3,8	1,2,3,4, 5,6,7,8
B3	Nil	8	4,5,6,7	4,5,6,7, 8
B4	Nil	Nil	4,5,6,7	4,5,6,7
B5	Nil	Nil	Nil	Nil



Example-1: IN-OUT analysis and Reaching Definitions

BLOCK	PREDCE'R	GEN[B]	KILL[B]
B1	NIL	1,2,3	8
B2	B1,B3	4,5,6,7	NIL
B3	B2	8	1
B4	B2	Nil	NIL
B5	B4	Nil	NIL

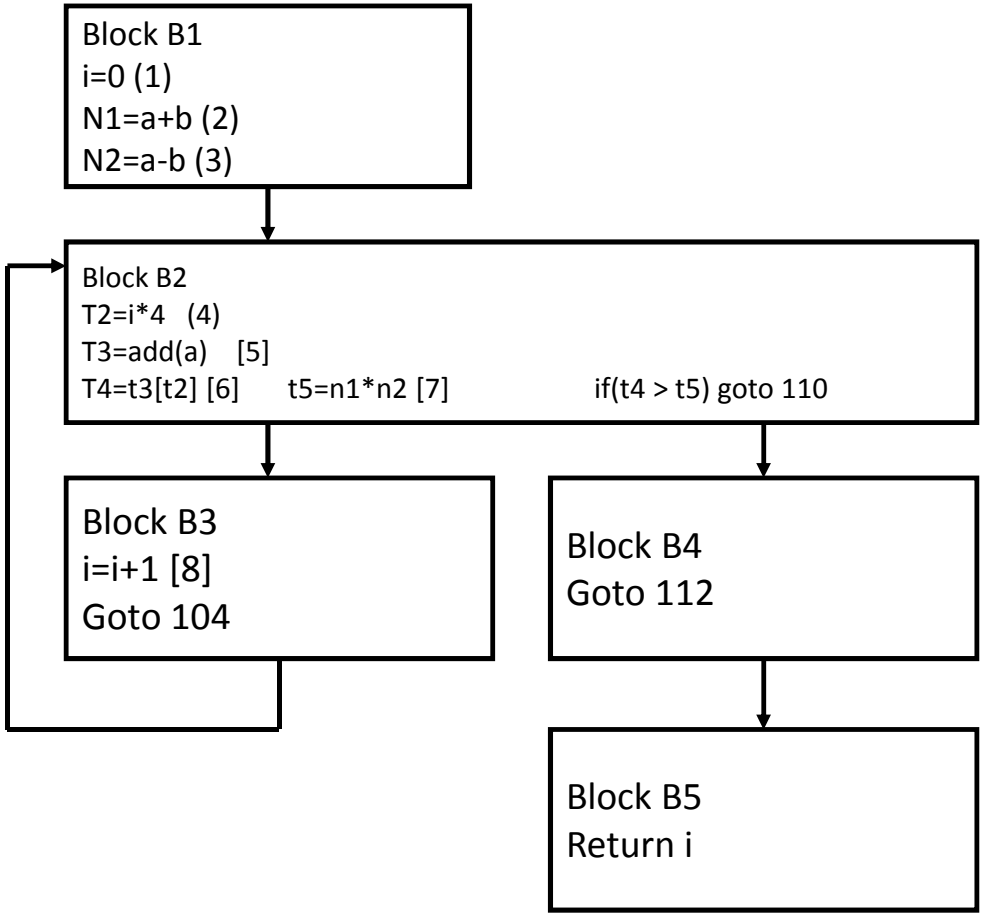
BLOCK	IN[B]	OUT[B]	IN[B]	OUT[B]
B1	Nil	1,2,3	Nil	1,2,3
B2	1,2,3,8	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8	1,2,3,4,5,,6,7,8
B3	4,5,6,7	4,5,6,7,8	1,2,3,4,5,6,7,8	2,3,4,5,6 7,8
B4	4,5,6,7	4,5,6,7	1,2,3,4,5,6,7,8	1,2,3,4,5 6,7,8
B5	Nil	Nil	4,5,6,7	4,5,6,7



Example-1: IN-OUT analysis and Reaching Definitions

BLOCK	PREDCE'R	GEN[B]	KILL[B]
B1	NIL	1,2,3	8
B2	B1,B3	4,5,6,7	NIL
B3	B2	8	1
B4	B2	Nil	NIL
B5	B4	Nil	NIL

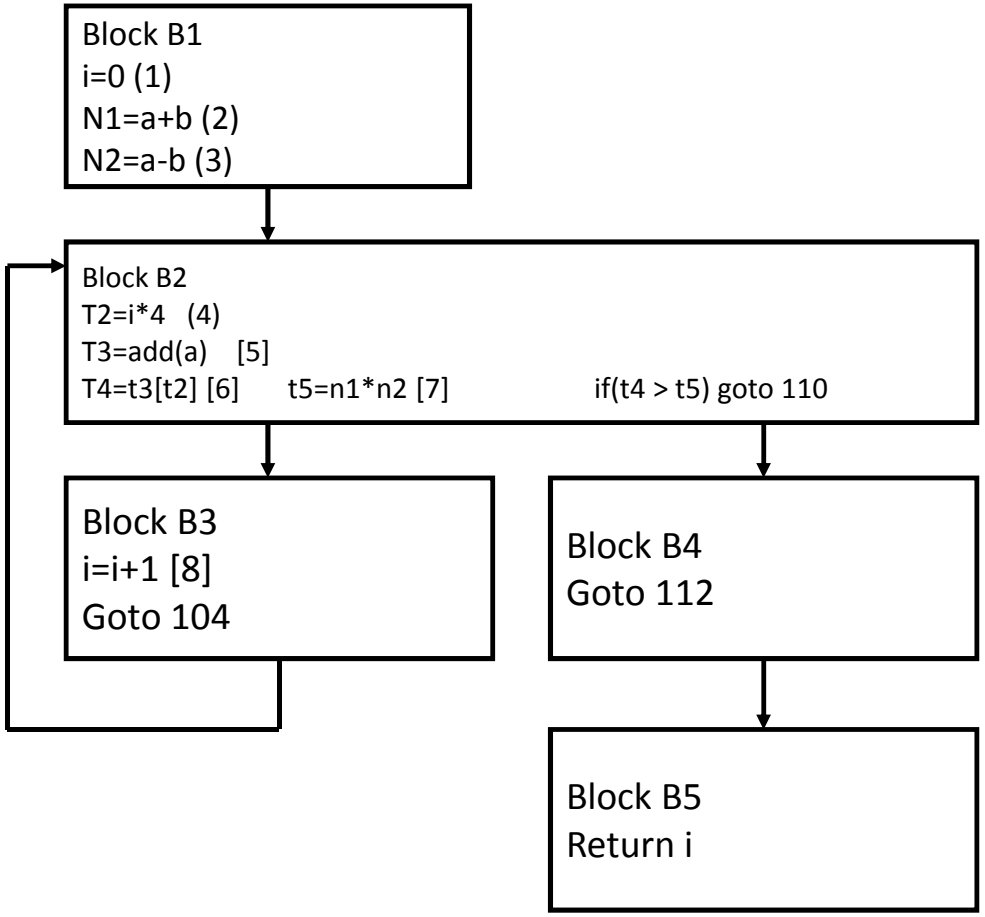
BLOCK	IN[B]	OUT[B]	IN[B]	OUT[B]
B1	Nil	1,2,3	Nil	1,2,3
B2	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8
B3	1,2,3,4,5,6,7,8	2,3,4,5,6,7,8	1,2,3,4,5,6,7,8	2,3,4,5,6,7,8
B4	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8
B5	4,5,6,7	4,5,6,7	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8



Example-1: IN-OUT analysis and Reaching Definitions

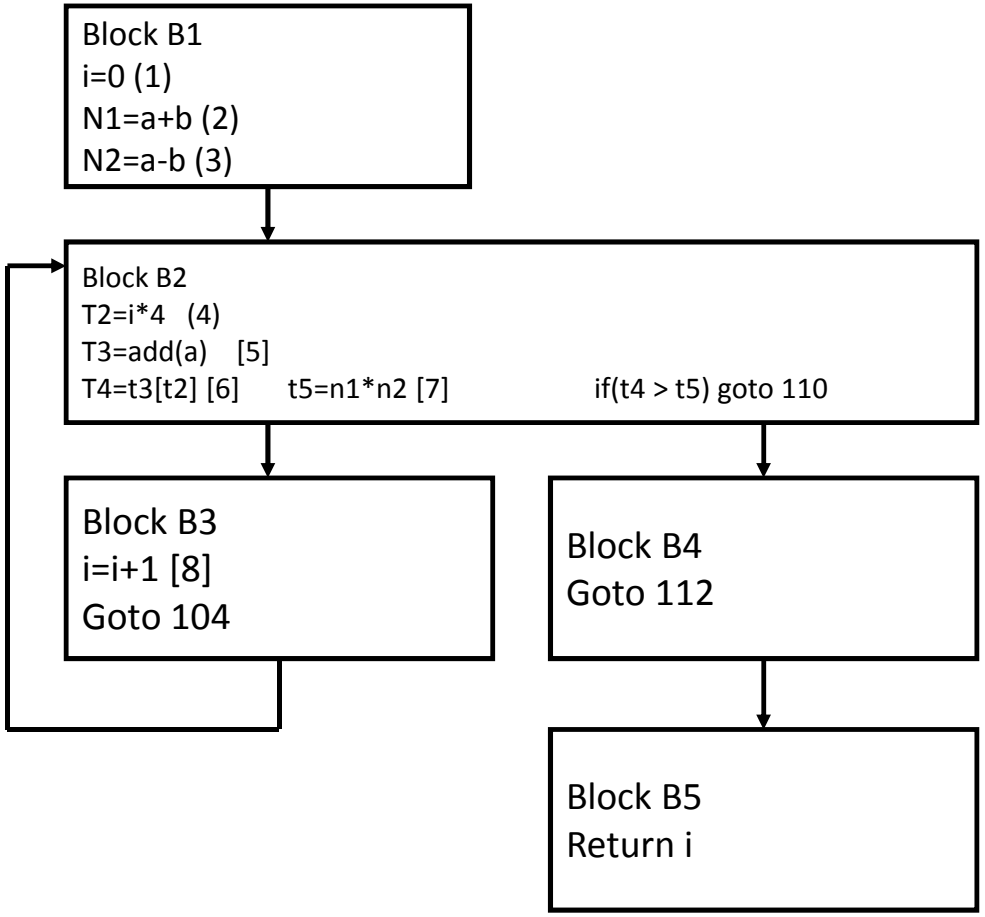
BLOCK	PREDCE'R	GEN[B]	KILL[B]
B1	NIL	1,2,3	8
B2	B1,B3	4,5,6,7	NIL
B3	B2	8	1
B4	B2	Nil	NIL
B5	B4	Nil	NIL

BLOCK	IN[B]	OUT[B]	IN[B]	OUT[B]
B1	Nil	1,2,3	**Nil**	**1,2,3**
B2	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8	**1,2,3,4,5,6 7,8**	**1,2,3,4,5, 6,7,8**
B3	1,2,3,4,5, 6,7,8	2,3,4,5,6,7 ,8	**1,2,3,4,5,6 7,8**	**2,3,4,5,6,7 ,8**
B4	1,2,3,4,5,6,7, 8	1,2,3,4,5,6 ,7,8	**1,2,3,4,5,6 7,8**	**1,2,3,4,5,6 7,8**
B5	1,2,3,4,5,6,7, 8	1,2,3,4,5,6 ,7,8	**1,2,3,4,5,6 7,8**	**1,2,3,4,5,6 7,8**



Example-1: IN-OUT analysis and Reaching Definitions

BLOCK	Reaching definitions at beginning of Block	Reaching definitions at end of Block
B1	Nil	1,2,3
B2	1,2,3,4,5,6,7,8	1,2,3,4,5, 6,7,8
B3	1,2,3,4,5,6,7,8	2,3,4,5,6,7,8
B4	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8
B5	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8



U-d Chain: Computation

- It is a data structure used to store reaching definitions
- It represents set of all definitions reaching the quad.
- For example t5= n1*n2, then to compute t5, it is necessary to get access to variable n1 and n2.
- It is expressed as:
- u-d chain(7,n1)={2}
- u-d chain(7,n2)={3}

B2	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8

Quad	Ud chain	Remarks
(4) t2 = i * 4	Udchain(4,i) = {1,8}	Definition of "i" IN[B2] are 1, 8 There is no definition of "i" in block B1 preceding quad 4, hence ud_chain (4,i) = 1,8
(5) t3 = add(a)	None	
(8) i=i+1	Ud_chain(8,i) = {1,8}	Definitions of "i" IN[B3] are 1,8. There is no definition of "i" preceding quad 8 in block B3. So ud chain = [1,8]

U-d Chain: Computation

Quad	Ud chain	Remarks
(7) t5 = n1*n2	Ud_chain(7,n1)=2 Ud_chain(7,n2)=3	The definition of variable n1 is in block B1, and there is no definition of n1 preceding to block B2 [in block B2]. Hence the definition of "n1" is from quad 2, which is in Block B1. Similar is true for variable "n2"

The loop invariant is computed using following procedure:

1. Draw the program flow graph from TAC
2. Find the Dominator information of each block
3. Compute back edges
4. Find the block involved in the back edge
5. Compute loop invariant from the blocks involved in back edges
6. Move the loop invariant after/before the loop [Refer rule base for creating new block for loop invariant]

Students can refer the class notes on computation of dominators and detection of back edge

The material in next slide is based on assumption: loop is already detected in program flow graph.

Computation of loop invariant

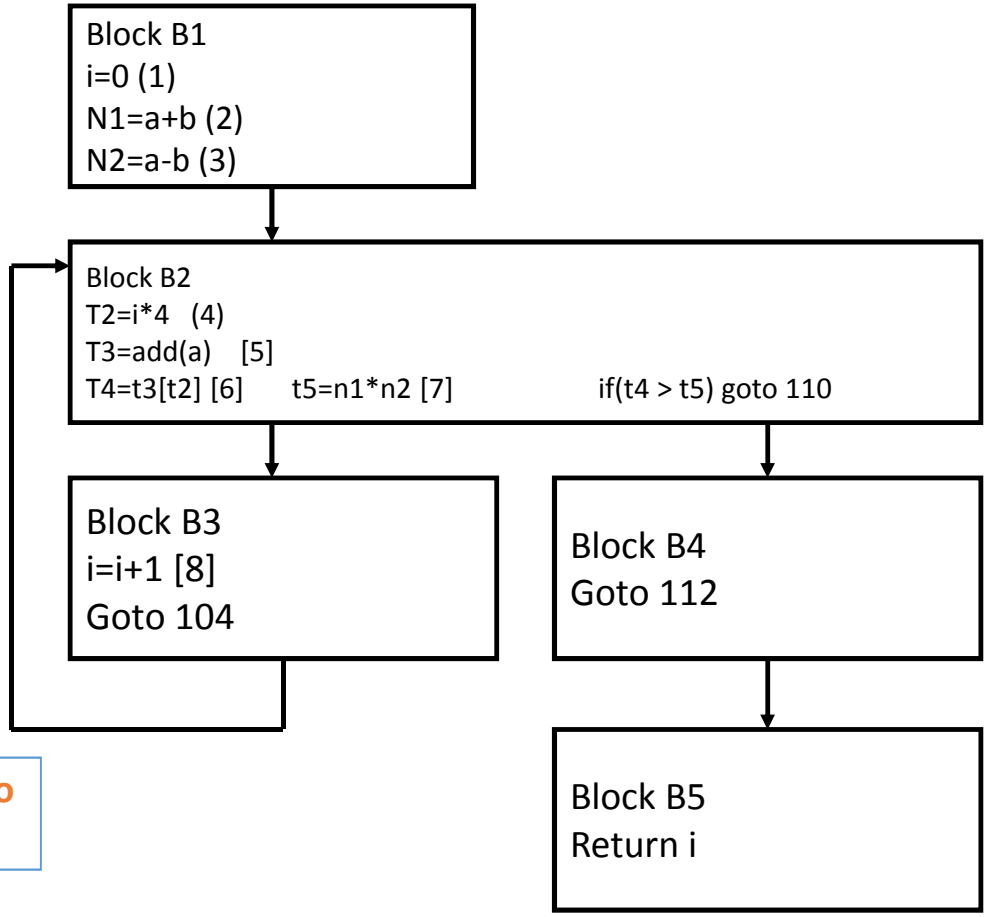
- From the information generated from computation of
- Reaching definition
- U-d chains
- And following the rules given below: Loop invariant can be computed if one of following rule is true.
- (3) x=y+z is loop invariant if

- Rule 1: All reaching definitions of “y” and “z” at quad(3) are from outside the loop, which can be checked using **u-d chain process**
- Rule 2: The operand “y” and “z” are constant.

Example: Loop invariant computation : Cont..

BLOCK	Reaching definitions at beginning of Block	Reaching definitions at end of Block
B1	Nil	1,2,3
B2	1,2,3,4,5,6,7,8	1,2,3,4,5, 6,7,8
B3	1,2,3,4,5,6,7,8	2,3,4,5,6,7,8
B4	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8
B5	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8

Quad	Ud chain	Remarks
(5) t3 = add(a)	None	Constant

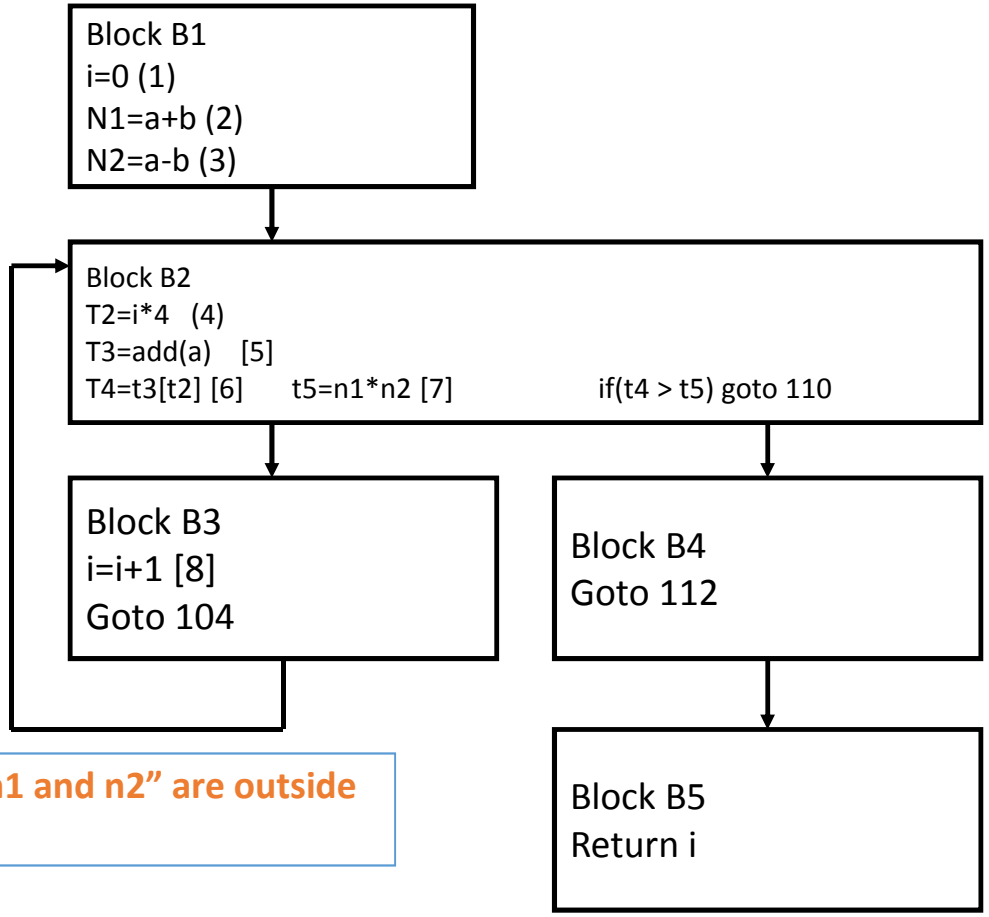


Since there is no ud-chain information for quad(5) and also it is constant. It is considered as loop invariant.

Example: Loop invariant computation : Cont..

BLOCK	Reaching definitions at beginning of Block	Reaching definitions at end of Block
B1	Nil	1,2,3
B2	1,2,3,4,5,6,7,8	1,2,3,4,5, 6,7,8
B3	1,2,3,4,5,6,7,8	2,3,4,5,6,7,8
B4	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8
B5	1,2,3,4,5,6,7,8	1,2,3,4,5,6,7,8

Quad	Ud chain	Remarks
(7) t5 = n1*n2	Ud_chain(7,n1)=2 Ud_chain(7,n2)=3	The definition of variable n1 is in block B1, and there is no definition of n1 preceding to block B2 [in block B2]. Hence the definition of "n1" is from quad 2, which is in Block B1. Similar is true for variable "n2"



The statement is loop invariant, since the definition of "n1 and n2" are outside Block B2 and there is no redefinition of variables.

Next: How to create a new block to move the loop invariant out of loop

M.B.Chandak