Assignment 2

Very Busy Expressions

Domain:

Sets of Expressions X⊕Y

Direction:

Backward

Transfer Function:

$$IN[B_i] = f(OUT[B_i])$$

$$IN[B_i] = OUT[B_i] + COMPUTED[B_i] - CHANGED[B_i]$$

Per ottenere le espressioni very busy in entrata a B, prendiamo tutte le espressioni very busy in uscita da B, sottraiamo quelle con operandi che vengono modificati in B e aggiungiamo quelle che vengono computate in B.

Meet Operation:

$$\bigcap$$

Le espressioni very busy in uscita da B sono tutte quelle che sono very busy in tutti i blocchi immediatamente successivi a B.

Boundary Condition

$$OUT[B_{EXIT}] = \emptyset$$

Initial Interior Points

$$OUT[B_i] = \emptyset$$

Iterazioni

	Iterazione 1	Iterazione 1
	IN[B]	OUT[B]
BB1	a-b	a-b
BB2	a-b	a-b
BB3	a-b, b-a	a-b
BB4	a-b	null
BB5	b-a	null
BB6	null	a-b
BB7	a-b	null
BB8	null	null

Dominator Analysis

Domain:

Sets of basic blocks

Direction:

Forward

Transfer Function:

$$DOM[B_i] = \{B_i\} \cup \bigcap_{B_j \in Pred(B_i)} DOM[B_j]$$

L'insieme dei blocchi che dominano B contengono B stesso e l'intersezione di tutti gli insiemi dominatori dei blocchi immediatamente precedenti B.

Meet Operation:

$$\bigcap$$

Consegue dalla precedente dichiarazione.

Boundary Condition:

$$DOM[B_{ENTRY}] = \{B_{ENTRY}\}$$

Initial Interior Points

$$DOM[B_i] = \{All\ blocks\}$$

Iterazioni

	Iterazione 1
	DOM[B]
A (ENTRY)	A
B	A
C	A
D	A, C
E	A, C
F	A, C
G (EXIT)	A

Constant Propagation

Domain:

Sets of {variable, value}

Direction:

Forward

Transfer Function:

$$OUT[B_i] = f(IN[B_i])$$

$$OUT[B_i] = IN[B_i] - REDEF\_IN[B_i] + REDEF\_OUT[B_i] + DEF[B_i]$$

Per ottenere le coppie di variabile e costante in uscita da B, prendiamo quelle in entrata in B, sostituiamo tutte quelle che vengono ridefinite in B con le rispettive ridefinizioni e aggiungiamo tutte le coppie definite per la prima volta in B.
NOTA: Qualora vi sia una istruzione di tipo x=y, dove y è parte di una coppia (y, c) che appartiene all'insieme in entrata in B, si aggiunge all'insieme in uscita da B anche una coppia di

tipo (x, c).
Similarmente, qualora vi sia una istruzione di tipo x=y⊕z, e sia y che z siano parte di coppie analoghe (y, c) e (z, d), si aggiunge all'insieme in uscita da B anche una coppia di tipo (x, e), dove e è il risultato della computazione di c⊕d. Questo vale anche se y o z è una costante.

Meet Operation:

$$\bigcap$$

Le coppie in entrata in B sono solamente quelle che mantengono entrambi i loro membri equivalenti in tutti i blocchi immediatamente precedenti B.

Boundary Condition:

$$IN[B_{ENTRY}] = \emptyset$$

Initial Interior Points:

$$IN[B_i] = \emptyset$$

Iterazioni

	Iterazione 1	**Iterazione 1**	**Iterazione 2**	**Iterazione 2**
	IN[B]	OUT[B]	IN[B]	OUT[B]
ENTRY	null	null		
k=2	null	(k,2)		
if	(k,2)	(k,2)		
a=k+2	(k,2)	(k,2), (a,4)		
x=5	(k,2), (a,4)	(k,2), (a,4), (x,5)		
a=k*2	(k,2)	(k,2), (a,4)		
x=8	(k,2), (a,4)	(k,2), (a,4), (x,8)		
k=a	(k,2), (a,4)	(k,4), (a,4)		
while	(k,4), (a,4)	(k,4), (a,4)	(k,5), (a,4), (b,2), (x,8), (y,8)	(k,5), (a,4), (b,2), (x,8), (y,8)
b=2	(k,4), (a,4)	(k,4), (a,4), (b,2)	(k,5), (a,4), (b,2), (x,8), (y,8)	(k,5), (a,4), (b,2), (x,8), (y,8)
x=a+k	(k,4), (a,4), (b,2)	(k,4), (a,4), (b,2), (x,8)	(k,5), (a,4), (b,2), (x,8), (y,8)	(k,5), (a,4), (b,2), (x,8), (y,8)

	Iterazione 1	**Iterazione 1**	**Iterazione 2**	**Iterazione 2**
y=a*b	(k,4), (a,4), (b,2), (x,8)	(k,4), (a,4), (b,2), (x,8), (y,8)	(k,5), (a,4), (b,2), (x,8), (y,8)	(k,5), (a,4), (b,2), (x,8), (y,8)
k++	(k,4), (a,4), (b,2), (x,8), (y,8)	(k,5), (a,4), (b,2), (x,8), (y,8)	(k,5), (a,4), (b,2), (x,8), (y,8)	(k,6), (a,4), (b,2), (x,8), (y,8)
print(a+x)	(k,4), (a,4)	(k,4), (a,4)	(k,5), (a,4), (b,2), (x,8), (y,8)	(k,5), (a,4), (b,2), (x,8), (y,8)
EXIT	(k,4), (a,4)	(k,4), (a,4)	(k,5), (a,4), (b,2), (x,8), (y,8)	(k,5), (a,4), (b,2), (x,8), (y,8)

Non conoscendo la condizione del while è impossibile stabilire una convergenza completa, dato che ogni ciclo incrementa k.