Assignment 2 - Linguaggi e compilatori

Marco Pini (169105), Ronchetti Gian Marco (164907)

	Dataflow problem 1 - Very Busy Expressions
Domain	Expressions
Direction	Backward: IN[B] = fB(OUT[B]) OUT[B] = ∩ IN[succ(B)]
Transfer Function	fB (x)= GEN[B] U (x - KILL[B])
Meet Operation(∧)	∩
Boundary Condition	IN[EXIT]=⌀
Initial interior points	IN[B]=U

Date le espressioni a-b e b-a definiamo un bit vector composto da due bit, il primo per l'espressione a-b e il secondo per b-a. Quando uno dei due valori sarà a 1 vorrà dire che la corrispondente espressione sarà very busy

	Iterazione 1		Iterazione 2	
	IN[B]	OUT[B]	IN[B]	OUT[B]
BB1	<0,1>	<0,1>	<0,1>	<0,1>
BB2	<0,1>	<0,1>	<0,1>	<0,1>
BB3	<1,1>	<1,0>	<1,1>	<1,0>
BB4	<1,0>	⌀	<1,0>	⌀
BB5	<0,1>	⌀	<0,1>	⌀
BB6	⌀	<1,0>	⌀	<1,0>
BB7	<1,0>	⌀	<1,0>	⌀
BB8	⌀	<>	⌀	<>

	Dataflow problem 2 - Dominator Analysis
Domain	Basic Blocks
Direction	Forward: OUT[B]= fB(IN[B]) IN[B]=∩ OUT[pred[b]]
Transfer Function	fB (x)= {B} U x
Meet Operation(∧)	∩
Boundary Condition	OUT[entry]=∅
Initial interior points	OUT[B]=U

Il bit vector in questo caso sarà composto da un bit per ogni blocco in ordine alfabetico.

	IN[B]	OUT[B]
ENTRY	∅	∅
A	∅	<1,0,0,0,0,0,0>
B	<1,0,0,0,0,0,0>	<1,1,0,0,0,0,0>
C	<1,0,1,0,0,0,0>	<1,0,1,0,0,0,0>
D	<1,0,1,0,0,0,0>	<1,0,1,1,0,0,0>
E	<1,0,1,0,0,0,0>	<1,0,1,0,1,0,0>
F	<1,0,1,0,0,0,0>	<1,0,1,0,0,1,0>
G	<1,0,0,0,0,0,0>	<1,0,0,0,0,0,1>

	Dataflow problem 3 - Constant Propagation
Domain	tuple <variabile, costante>
Direction	Forward: OUT[B]=fB(IN[B]) IN[B]=∩ OUT[pred[b]]
Transfer Function	fB (x) = GEN[B] U (x - KILL[B])
Meet Operation(∧)	∩
Boundary Condition	OUT[entry]=∅
Initial interior points	OUT[B]=U

In questo caso non è stato utilizzato un bit vector ma un insieme di tuple(<<costante>,<valore>>) per ogni punto.

	Iterazione 1		Iterazione 2		Iterazione 3	
	IN[B]	OUT[B]	IN[B]	OUT[B]	IN[B]	OUT[B]
Entry	∅	∅	∅	∅	∅	∅
K= 2	∅	<<k,2>>	∅	<<k,2>>	∅	<<k,2>>
if	<<k,2>>	<<k,2>>	<<k,2>>	<<k,2>>	<<k,2>>	<<k,2>>
a=k+2	<<k,2>>	<<k,2>, <a,4>>	<<k,2>>	<<k,2>, <a,4>>	<<k,2>>	<<k,2>, <a,4>>
x=5	<<k,2>, <a,4>>	<<k,2>, <a,4>, <x,5>>	<<k,2>, <a,4>>	<<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>>	<<k,2>>	<<k,2>, <a,4>>	<<k,2>>	<<k,2>, <a,4>>
x=8	<<k,2>, <a,4>>	<<k,2>, <a,4>, <x,8>>	<<k,2>, <a,4>>	<<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>>	<<k,2>, <a,4>>	<<k,4>, <a,4>>	<<k,2>, <a,4>>	<<k,4>, <a,4>>
while	<<k,4>, <a,4>>	<<k,4>, <a,4>>	<<a,4>>	<<a,4>>	<<a,4>>	<<a,4>>
b=2	<<k,4>, <a,4>>	<<k,4>, <a,4>, <b,2>>	<<a,4>>	<<a,4>, <b,2>>	<<a,4>>	<<a,4>, <b,2>>

x=a+k	<<k,4>, <a,4>, <b,2>>	<<k,4>, <a,4>, <b,2>, <x,8>>	<<a,4>, <b,2>>	<<a,4>, <b,2>>	<<a,4>, <b,2>>	<<a,4>, <b,2>>
y=a+b	<<k,4>, <a,4>, <b,2>, <x,8>>	<<k,4>, <a,4>, <b,2>, <x,8>, <y,8>>	<<a,4>, <b,2>>	<<a,4>, <b,2>, <y,8>>	<<a,4>, <b,2>>	<<a,4>, <b,2>, <y,8>>
k++	<<k,4>, <a,4>, <b,2>, <x,8>, <y,8>>	<<k,5>, <a,4>, <b,2>, <x,8>, <y,8>>	<<a,4>, <b,2>, <y,8>>	<<a,4>, <b,2>, <y,8>>	<<a,4>, <b,2>, <y,8>>	<<a,4>, <b,2>, <y,8>>
print(a+x)	<<k,4>, <a,4>>	<<k,4>, <a,4>>	<<a,4>>	<<a,4>>	<<a,4>>	<<a,4>>
Exit	<<k,4>, <a,4>>	<<k,4>, <a,4>>	<<a,4>>	<<a,4>>	<<a,4>>	<<a,4>>