Dataflow Analysis

October 28 2022

Partial Order

Properties for meet operator

1. *idempotent*: x $\wedge$ x = x
2. *commutative*: x $\wedge$ y = y $\wedge$ x
3. *associative*: x $\wedge (y \wedge z) = (x \wedge y) \wedge z$

Partial order: $x \leq y$ *iff* $x \wedge y = x$

Properties for partial order

1. *reflexive*: $x \leq x$
2. *anti-symmetry*: $x \leq y$ *and* $y \leq x$ *then* $x = y$
3. *transitive*: $x \leq y$ *and* $y \leq z$ *then* $x \leq z$

Prove the transitivity property of the partial order

Monotonicity

1.What is the point of monotonicity?

2.Is the result optimal?

3.Prove the equivalence of equation (1) and (2)

$$x \leq y \textit{ implies } f(x) \leq f(y) \tag{1}$$

$$f(x \wedge y) \leq f(x) \wedge f(y) \tag{2}$$

Note: The greatest lower bound for x, and y can only be $x \wedge y$

Dataflow Analysis

A uniform framework for computing properties of basic blocks useful for optimization. A type of **static analysis**

Checklist

- Domain: Inputs for the analysis
- Direction:
  - Forward Flow: What can **happen** before a given point
  - Backward Flow: What can happen **after** the given point.
- May / Must:
  - May Analysis: What property holds on **some path**
    - Meet Operator: $\cup$
    - Initial value: min
  - Must Analysis: What property holds on **all paths**
    - Meet Operator: $\cap$
    - Initial value: max
- Transfer Function: How the value changes among the statement
  - $IN(b)$: What properties hold **on entry to** basic block b
  - $OUT(b)$: What properties hold **on exit from** basic block b
  - $Gen(b)$: The properties that are **generated in** basic block b
  - $Kill(b)$: The properties that are **invalidated in** basic block b

Reaching Definition

Problem Definition

- A definition d reaches a point p if there **is a path** from the point immediately following d to p, such that **d is not killed along that path**
- $Gen(B)$: Definitions made in B
- $Kill(B)$: Definitions invalidate by B

Usage

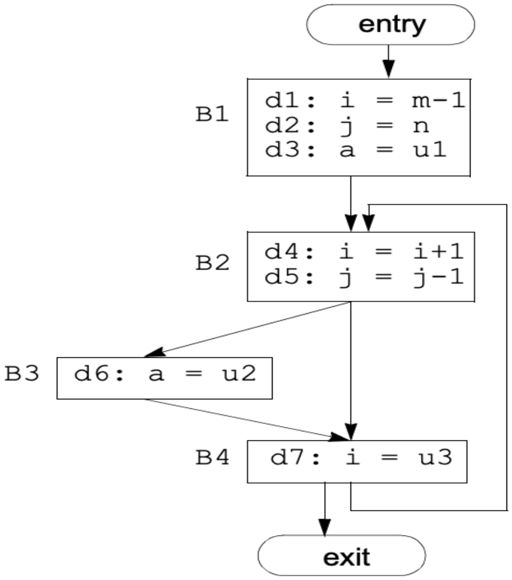
- Determine possible usage of uninitialized value

Summary

Domain
Direction
Transfer Function
Meet Operator($\wedge$)
IN Equation
OUT Equation
Initial Condition
Boundary Condition

Table 1: Summary for Reaching Definition

Example


entry
B1
d1: i = m-1
d2: j = n
d3: a = u1
B2
d4: i = i+1
d5: j = j-1
B3
d6: a = u2
B4
d7: i = u3
exit

Live Variable

Problem Definition

- A variable v is live at point p if the **value** of v is used **along some path starting at p**
- $Def(B)$: Variables defined in B
- $Use(B)$: Variables whose values may be used in B

Usage

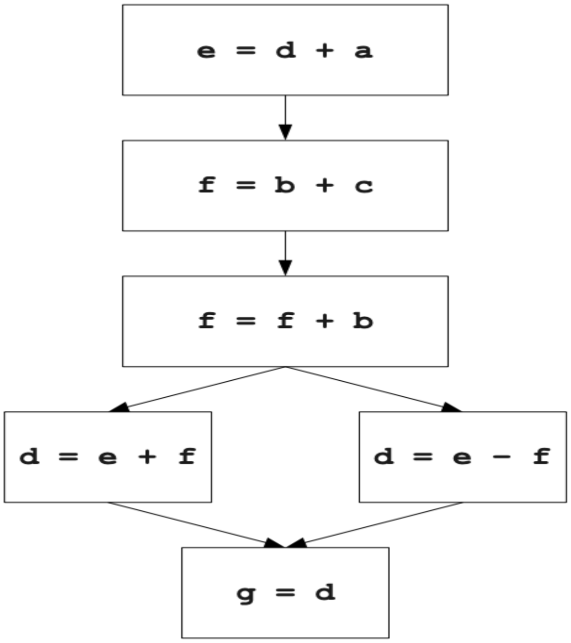
- Build inference graph in register allocation

Summary

Domain	
Direction	
Transfer Function	
Meet Operator($\wedge$)	
IN Equation	
OUT Equation	
Initial Condition	
Boundary Condition	

Table 2: Summary for Live Variable

Example



Available Expression

Problem Definition

- An expression is available at a point p if **every path leading to p** contains a definition of the expression which is not subsequently killed
- $Gen(B)$: Expression generated in B
- $Kill(B)$: Expression x $\oplus$ y is killed if it assigns x or y and does not subsequently recompute x $\oplus$ y

Usage

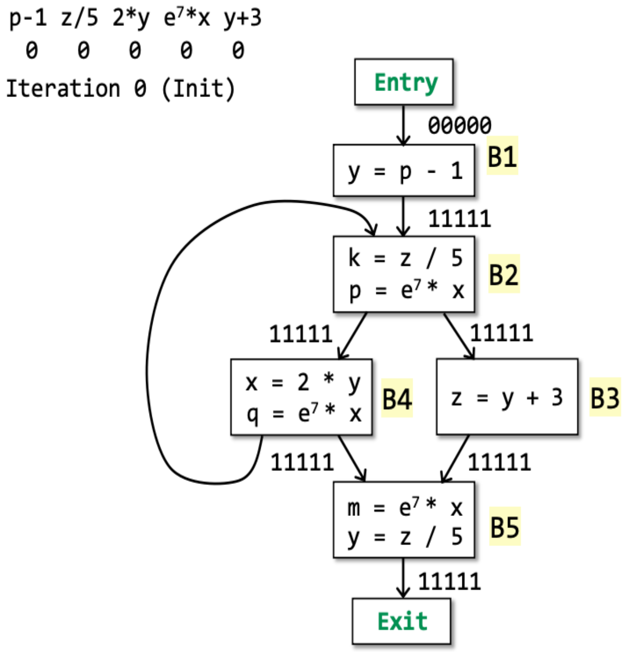
- Determine sub-expression for global common sub-expression elimination

Summary

Domain	
Direction	
Transfer Function	
Meet Operator($\wedge$)	
IN Equation	
OUT Equation	
Initial Condition	
Boundary Condition	

Table 3: Summary for Available Expression

Example


p-1 z/5 2*y e7*x y+3
0 0 0 0 0
Iteration 0 (Init)
Entry
00000
y = p - 1
B1
11111
k = z / 5
p = e7 * x
B2
11111
11111
x = 2 * y
q = e7 * x
B4
z = y + 3
B3
11111
11111
m = e7 * x
y = z / 5
B5
11111
Exit

Busy/Anticipated Expression

Problem Definition

- An expression e is very busy at point p **if not matter what path is taken from p**, the expression e will be evaluated before any of its operands are redefined

Usage

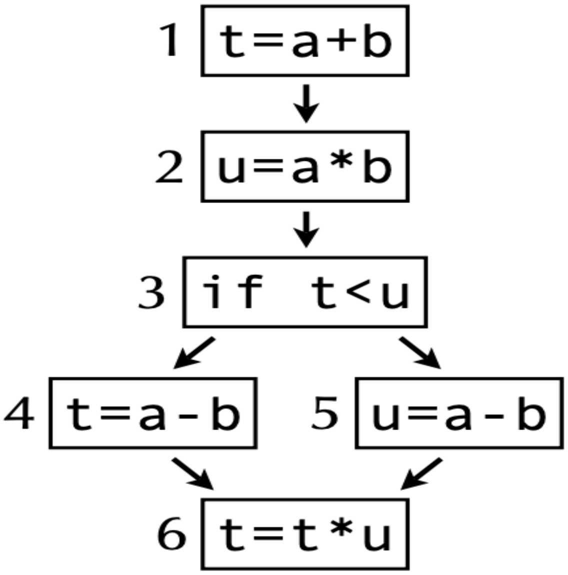
- Determine candidates for hoisting

Summary

Domain
Direction
Transfer Function
Meet Operator($\wedge$)
IN Equation
OUT Equation
Initial Condition
Boundary Condition

Table 4: Summary for Busy Expression

Example


1
t=a+b
2
u=a*b
3
if t<u
4
t=a-b
5
u=a-b
6
t=t*u