

A Roadmap

- Traditional Static Program Analysis
 - Theory
 - Compiler Optimizations; Control Flow Graphs, Properties of Control Flow Graphs
 - **Data-flow Analysis – today's class**
 - Data-flow Frameworks
 - Specific Analyses, Applications, etc.
- Software Testing
- Formal Static Program Analysis

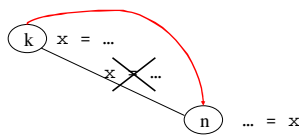
Four Classical Dataflow Problems

- Reaching Definitions (*Reach*)
- Live uses of variables (*Live*)
- Available expressions (*Avail*)
- Very Busy Expressions (*VeryB*)
- Def-use and Use-def chains, built from *Reach* and *Live*, are used in many optimizations
- *Avail* enables global common subexpression elimination
- *VeryB* is used for conservative code motion

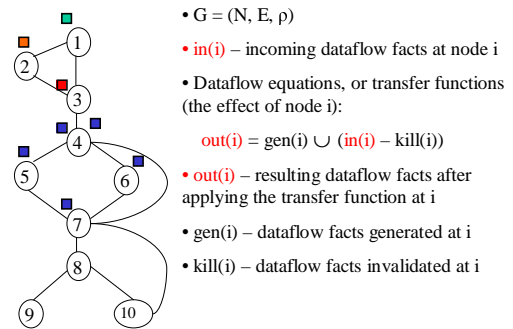
Reaching Definitions: Def-use Relations

- *Def-use chain* links a definition to an use that it reaches \rightarrow

Q: What definitions reach node n ?
 A: The definition of x at node k reaches n : (x,k)

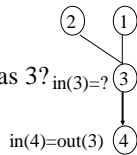


Dataflow analysis



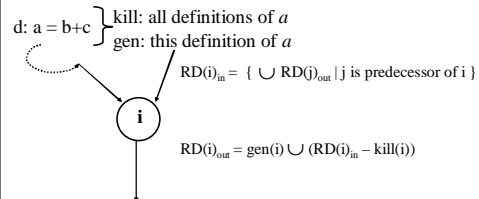
Two important questions

- How do we combine incoming dataflow facts at merge nodes such as 3? $in(3)=?$
- What are the $gen(i)$ and $kill(i)$ at a node i ?



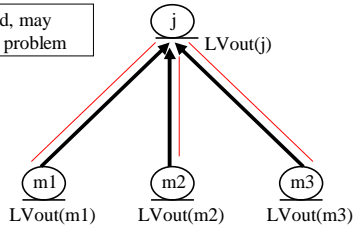
Problem 1: Reaching Definitions

- For each program point, what definitions *may* have been made and not overwritten, when program execution reaches this point.



Live Uses of Variables

Backward, may dataflow problem



Is this set of equations the same?

$$LVout(j) = \bigcup_{m \in succ(j)} \{(LVout(m) \cap pres(m)) \cup gen(m)\}$$

where:

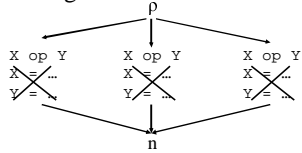
pres(m) is the set of uses preserved through node m (these will correspond to variables whose defs are preserved)

gen(m) is the set of uses generated at node m

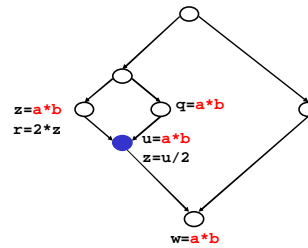
succ(j) is the set of immediate successors of node j

Problem 3: Available Expressions

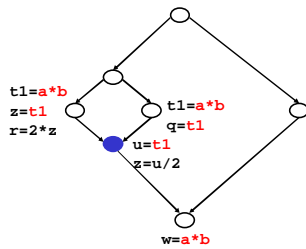
- An expression $X \text{ op } Y$ is *available* at program point n if **every** path from entry to n evaluates $X \text{ op } Y$, and after every evaluation prior to reaching n , there are NO subsequent assignments to X or Y



Global Common Subexpressions

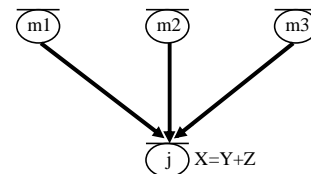


Global Common Subexpressions



Can we eliminate $w=a*b$?

Available Expressions



Forward, must dataflow problem

$AE_{in}(j) = ?$
 $AE_{out}(j) = ?$
 $gen(j) = ?$
 $kill(j) = ?$

Example

1. $x = a + b$
2. $y = a * b$
3. if $y \leq a + b$ then goto 7
4. $a = a + 1$
5. $x = a + b$
6. goto 3
7. ...