CSC2/455 Software Analysis and Improvement
Foundations of Data Flow Analysis

Sreepathi Pai
February 23, 2021

URCS
Outline

Review

The Meaning of It All

Data flow analysis

Postscript
Outline

Review

The Meaning of It All

Data flow analysis

Postscript
Analyses so far

- Live variables
- Available expressions
- Very busy expressions
- Dominators
  - Dominance frontiers (direct)
- Solved using Iterative data flow analysis
Outline

Review

The Meaning of It All

Data flow analysis

Postscript
The Reaching Definitions Problem

if (x > 5)
  x#0 = 5;
else
  x#1 = 0;

if (x)
  x#2 = 3;

u = x;

Which definitions reach u = x?
Compute reaching definitions for each *possible execution* path to $u = x$, and combine the results using $\cup$:

**Execution Path 1**

- $x#0 = 5$
- $x#2 = 3$
- $u = x$

**Execution Path 2**

- $x#1 = 0$
- $u = x$

So only $\{x#2, x#1\}$ reach $u$. 
The Ideal Solution

- Let $P = ENTRY, B0, B1, ..., B_k$ be an execution path to the entry of block $B_k$
- Let $f_P$ be the *composite* transfer function we want to evaluate
- The ideal solution is defined as the *meet* $(\land)$ over all $P$

$$IDEAL[B_k] = \bigwedge_{P \text{ is path from ENTRY to } B_k} f_P(P)$$

Note, for reaching definitions $\land = \cup$.

What is the problem with trying to find all execution paths?
Meet over (all possible flow-graph) paths (MOP)

- IDEAL can be undecidable
  - “does this loop on the path to $B_k$ terminate?”
- MOP approximates IDEAL
  - Only all paths in control flow graph are explored
- Calculate $f$ over all paths in the CFG, one path at a time.
- Combine using $\land$, as in IDEAL
• The MOP solution is \( \{x_0, x_1, x_2\} \)
• Note IDEAL \( \subset \) MOP
Maximum Fixed Point (MFP)

- CFGs with cycles will have an unbounded number of paths
- Hence, our use of the iterative dataflow analysis framework
  - Compute function $f$ at individual basic blocks (not on paths)
  - Use meet operator at join nodes (or “confluence” points)
  - Terminate when functions reach a fixpoint
  - (Ignore “maximum” for now)
MFP solution

- MFP solution is \( \{x#0, x#1, x#2\} \)
- In this case, MFP = MOP
• For reaching definitions (and this example):
  • IDEAL ⊂ MOP = MFP

• Precision also decreases from IDEAL to MFP

• Can we generalize?
  • IDEAL ⊆ MOP ⊆ MFP
  • When would it be a proper subset (⊂)?
  • When would it be identical (=)?
  • Do our conclusions apply only to reaching definitions?
Outline

Review

The Meaning of It All

Data flow analysis

Postscript
This slide intentionally blank.
More formally

- A data flow analysis is a \((D, V, \wedge, F)\) where:
  - \(D\) is direction (backwards or forwards)
  - \(V\) is the domain of solutions (i.e. set) to the dataflow problem
  - \(\wedge\) is the binary meet operator over \(V\)
  - \(F : V \rightarrow V\) is the family of transfer functions
- \(V\) and \(\wedge\) must define a semilattice
- A semilattice is a lattice with only a meet $\land$ operator (or a join $\lor$ operator)
  - A lattice has both
- A semilattice $(V, \land)$ is a set $V$ and an operator $\land$ such that:
  - $x \land x = x$ (idempotent)
  - $x \land y = y \land x$ (commutative)
  - $(x \land y) \land z = x \land (y \land z)$ (associative)
- Two elements of $V$ are denoted as $\top$ (top) and $\bot$ (bottom) such that:
  - $\forall x \in V \top \land x = x$
  - $\forall x \in V \bot \land x = \bot$
Reaching Definitions

- $D = ?$
- $V = ?$
- $\land = ?$
- $T = ?$
- $\bot = ?$
- $D =$ forward
- $V =$ powerset of definitions
- $\land = \cup$
- $\top = \emptyset$ (no definitions)
- $\bot = U$ (all definitions)
Semilattice for our Reaching Definitions Example

\[
\begin{array}{c}
\{\} \\
\{x#0\} \quad \{x#1\} \quad \{x#2\} \\
\{x#0, x#1\} \quad \{x#0, x#2\} \quad \{x#1, x#2\} \\
\{x#0, x#1, x#2\} (\bot)
\end{array}
\]
Partial Orders

- A semilattice \((V, \land)\) also induces a partial order among elements of \(V\).
- A partial order \(\leq\) for values \(x, y, z \in V\) has the following properties:
  - \(x \leq x\) (reflexive)
  - if \(x \leq y\) and \(y \leq x\), then \(x = y\) (anti-symmetric)
  - if \(x \leq y\) and \(y \leq z\), then \(x \leq z\) (transitive)
- A partial order \(\leq\) for a semilattice is defined as:
  - \(x \leq y\) if and only if \(x \land y = x\)
- The pair \((V, \leq)\) is known as a partially ordered set or poset.
What is the partial order \((x \leq y \text{ iff } x \land y = x)\) between:

- \{x#0\} and \{x#0, x#1\}?
- \{x#0\} and \{x#1\}?

What is the name of partial order when:

- \(\land = \cup\)? (i.e. \(x \cup y = x\))
- \(\land = \cap\)?
Informally, the greatest lower bound ($glb$) is the first element $\leq$ than both $x$ and $y$
  - it can be $x$ or $y$ as well (note $=$ in $\leq$)

The greatest lower bound $g$ of $x$ and $y \in V$ is
  - $g = x \land y$

See the textbook for the definition and proof
Putting it all together

Think about how values at each basic block change as we go about performing an iterative data flow analysis, especially how the values are related to each other in the lattice.
Transfer functions

- $F : V \rightarrow V$, $F$ is a set of functions $f$
- $f(x) = G \cup (x - K)$
- $F$ must satisfy two conditions:
  - must contain an identity function: $I(x) = x$
  - must be closed under composition $h(x) = f(g(x))$ is also in $F$
- E.g. reaching definitions transfer functions satisfy these conditions
  - See textbook for proof.
A given \((D, V, \land, F)\) is monotone if for all \(x, y \in V\), and \(f \in F\):

- \(x \leq y \rightarrow f(x) \leq f(y)\)
- equivalently, \(x \leq y \rightarrow f(x \land y) \leq f(x) \land f(y)\)

In addition, the framework is *distributive* if:

- \(f(x \land y) = f(x) \land f(y)\)

Note that these properties do not necessarily arise automatically, \(F\) must be designed to have these properties
- And proofs must be written to show that \(F\) does.
General Iterative Algorithm

forwards(IN, OUT, meet, top, v_entry, f_transfer)
    OUT[entry] = v_entry

    for each basic block B except ENTRY:
        OUT[B] = top

    do {
        for each basic block B except ENTRY:
            # this calculates the meet over predecessors, \( \bigwedge_p OUT[p] \)
            IN[B] = reduce(meet, [OUT[p] for p in B.predecessors])
            OUT[B] = f_transfer(IN[B])
    } while(some OUT changes value)

- Does this calculate the solution to the dataflow problem?
- Does this algorithm terminate?
- Does this algorithm calculate the maximum fixed point – i.e. the most precise solution admissible?
• Proofs that explore the three questions
• Relationships between IDEAL, MOP and MFP in terms of the framework
• Examples of:
  • a non-distributive framework (read Dragon 9.4, Constant Propagation)
  • lattices containing infinite values
  • possibly some proof writing exercises (read 9.3)
Outline

Review

The Meaning of It All

Data flow analysis

Postscript
• Chapter 9 of the Dragon book
  • Section 9.3
  • 4xx students are expected to be able to solve exercises therein