Bitvector Analyses
How to implement?

• Dataflow analyses like live-variable analysis are *bit-vector* analyses: are even more structured than regular dataflow analysis

  • Consistent lattice: powerset

  • Consistent transfer functions

  • Many sources only talk about bitvector dataflow
Bit-vector lattices

- Consider a single element, $V$, of the powerset($S$) lattice

- Each item in $S$ either appears in $V$ or does not: can represent using a single bit

  - Can represent $V$ as a bit vector

    - \{a, b, c\} = <1, 1, 1>
    - \{\} = <0, 0, 0>
    - \{b, c\} = <0, 1, 1>

  - $\sqcup$ and $\sqcap$ (which are just $\cup$ and $\cap$) are simply bitwise $\lor$ and $\land$, respectively
Eliminating merge nodes

- Many dataflow presentations do not use explicit merge nodes in CFG
- How do we handle this?
- Problem: now a node may be a statement and a merge point
- Solution: compose confluence operator and transfer functions
- Note: non-merge nodes have just one successor; this equation works for all nodes!

\[ T(s) = \text{use}(s) \cup \left( \bigcup_{X \in \text{succ}(s)} X \right) - \text{def}(s) \]
Simplifying matters

\[ T(s) = \text{use}(s) \cup (\bigcup_{X \in \text{succ}(s)} X) - \text{def}(s)) \]

- Let’s split this up into two different sets

  - OUT(s): the set of variables that are live immediately after a statement is executed
  - IN(s): the set of variables that are live immediately before a statement is executed

\[
\begin{align*}
\text{IN}(s) &= \text{use}(s) \cup (\text{OUT}(s) - \text{def}(s)) \\
\text{OUT}(s) &= \bigcup_{t \in \text{succ}(s)} \text{IN}(t)
\end{align*}
\]
Generalizing

• USE(s) are the variables that become live due to a statement—they are generated by this statement

• DEF(s) are the variables that stop being live due to a statement—they are killed by this statement

\[
IN(s) = \text{gen}(s) \cup (OUT(s) - \text{kill}(s)) \\
OUT(s) = \bigcup_{t \in succ(s)} IN(t)
\]
Bit-vector analyses

• A bit-vector analysis is any analysis that
  • Operates over the powerset lattice, ordered by $\subseteq$ and with $\cup$ and $\cap$ as its meet and join
  • Has transfer functions that can be written in the form:

    \[
    \begin{align*}
    IN(s) &= \text{gen}(s) \cup (OUT(s) - \text{kill}(s)) \\
    OUT(s) &= \bigcup_{t \in succ(s)} IN(t)
    \end{align*}
    \]

• Are these transfer functions monotonic? (Hint: if $f$ and $g$ are monotonic, is $f \circ g$ monotonic?)

• gen and kill are dependent on the statement, but not on IN or OUT

• Things are a little different for forward analyses, and some analyses use $\cap$ instead of $\cup$
next: more analyses