CSE in SSA and LLVM (PA2)

Available Expressions: Possible approach
– Going to miss out on some possibilities.
– Representation for expressions, define operator <
– If gen’ ed expressions not in avail_in set
  – Map each gen’ ed expression to instruction pointer that creates it
  – Put expression in avail_out set
– If gen’ ed expression is in avail_in set, then current instruction is unnecessary
– Special set intersection, expressions with same define are equivalent

Lattice Theoretic Framework Approach
– Effect of instruction n: \( x = e \)
  – \( \text{kill}[n] = \{(e, d) | \forall d\} \)
  – \( \text{gen}[n] = \{(x, n)\} \)

\[
\begin{align*}
i.1 & := j.1 + x \\
a.1 & := 4*i.1 \\
i.2 & := i.1 + 1 \\
b & := 4*i.2 \\
i.3 & := \phi(i.1, i.2) \\
c & := 4*i.3 \\
d & := j.1 + x
\end{align*}
\]