Really Basic Stuff

Flow Graphs

Constant Folding

Global Common Subexpressions

Induction Variables/Reduction in Strength

Dawn of Code Optimization

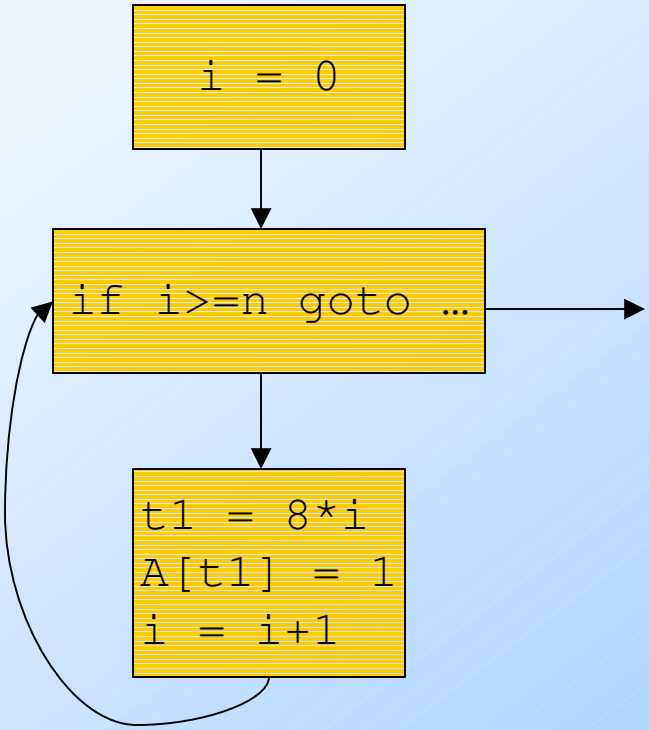
- A never-published Stanford technical report by Fran Allen in 1968.
- Flow graphs of intermediate code.
- Key things worth doing.

Intermediate Code

```
for (i=0; i<n; i++)
  A[i] = 1;
```

- Intermediate code exposes optimizable constructs we cannot see at source-code level.
- Make flow explicit by breaking into *basic blocks* = sequences of steps with entry at beginning, exit at end.

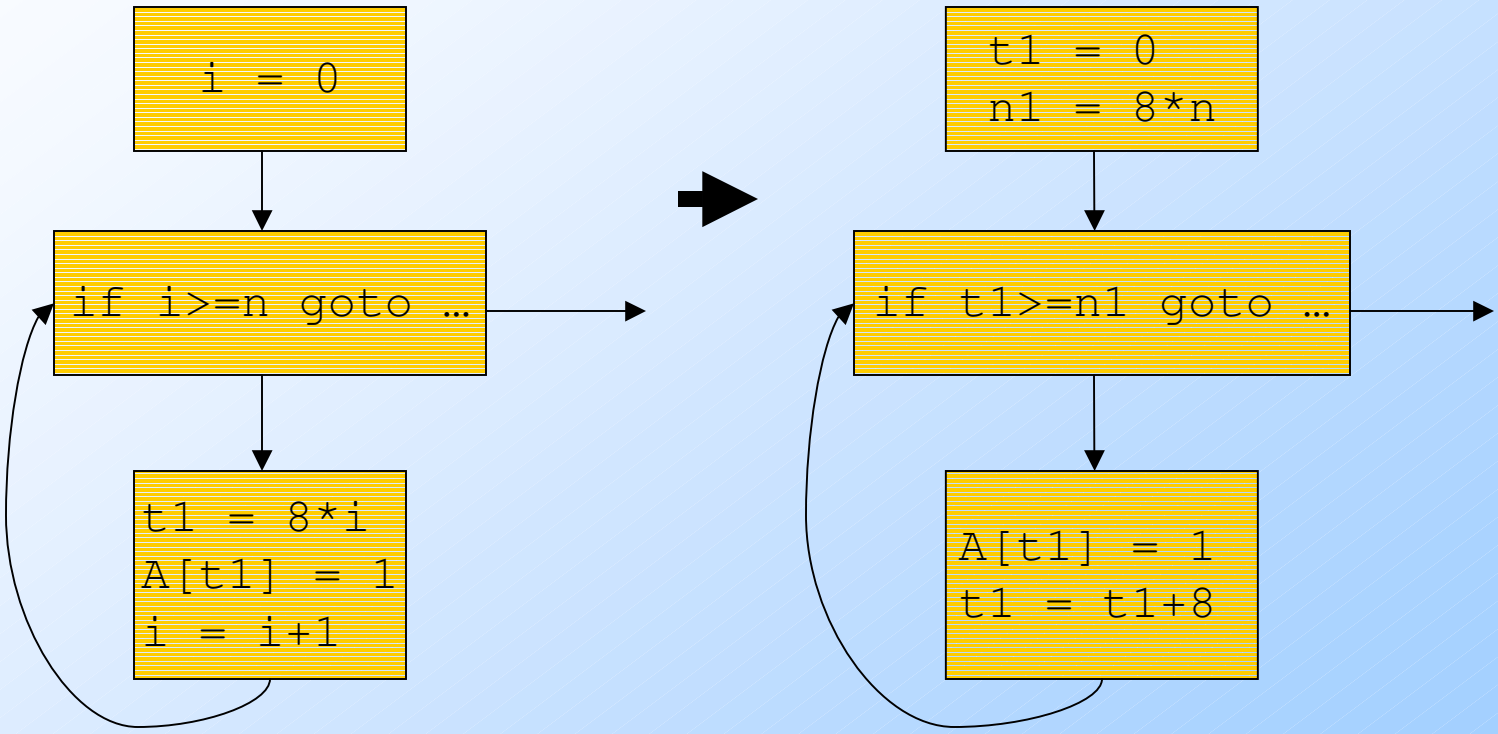
Basic Blocks



Induction Variables

- x is an *induction variable* in a loop if it takes on a linear sequence of values each time through the loop.
- Common case: loop index like i and computed array index like t1.
- Eliminate "superfluous" induction variables.
- Replace multiplication by addition (*reduction in strength*).

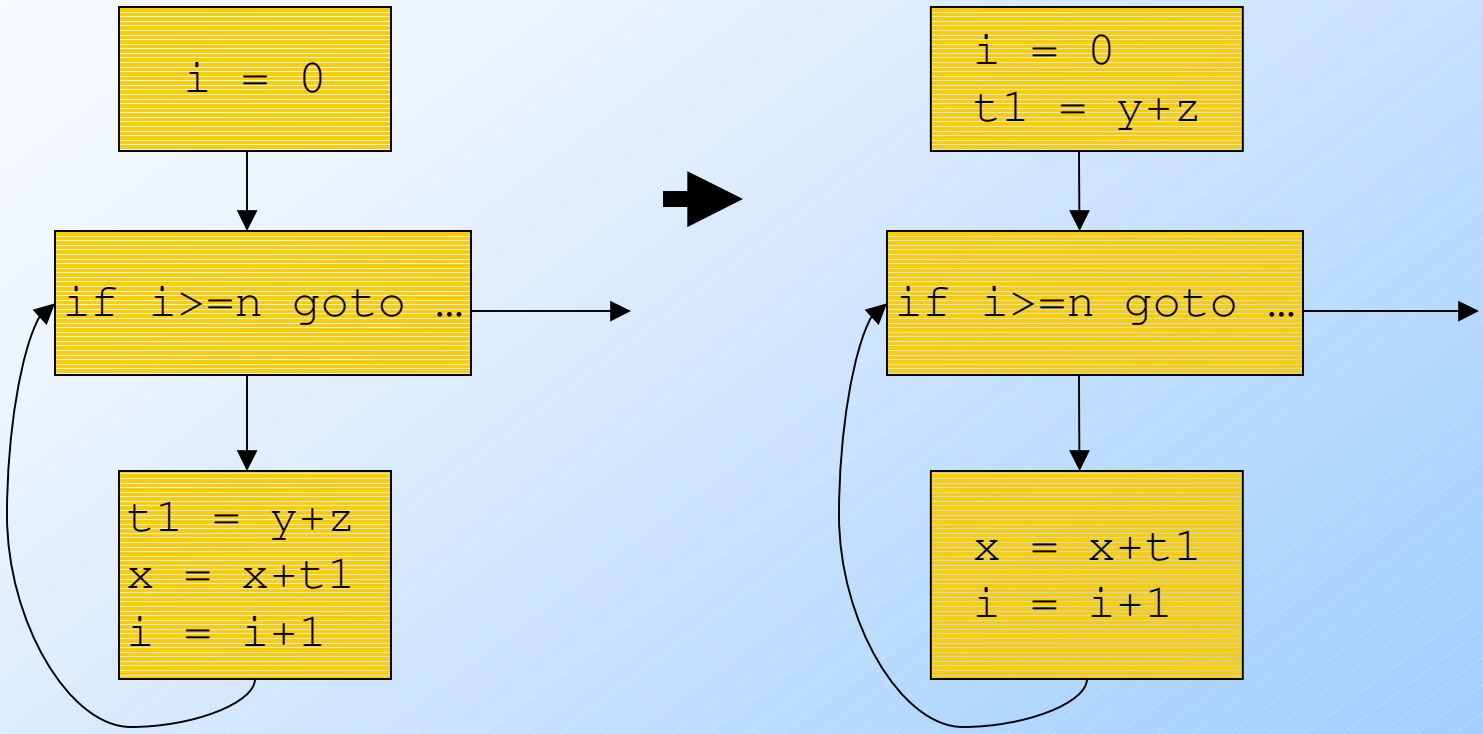
Example



Loop-Invariant Code Motion

- Sometimes, a computation is done each time around a loop.
- Move it before the loop to save n-1 computations.
  - Be careful: could n=0? I.e., the loop is typically executed 0 times.

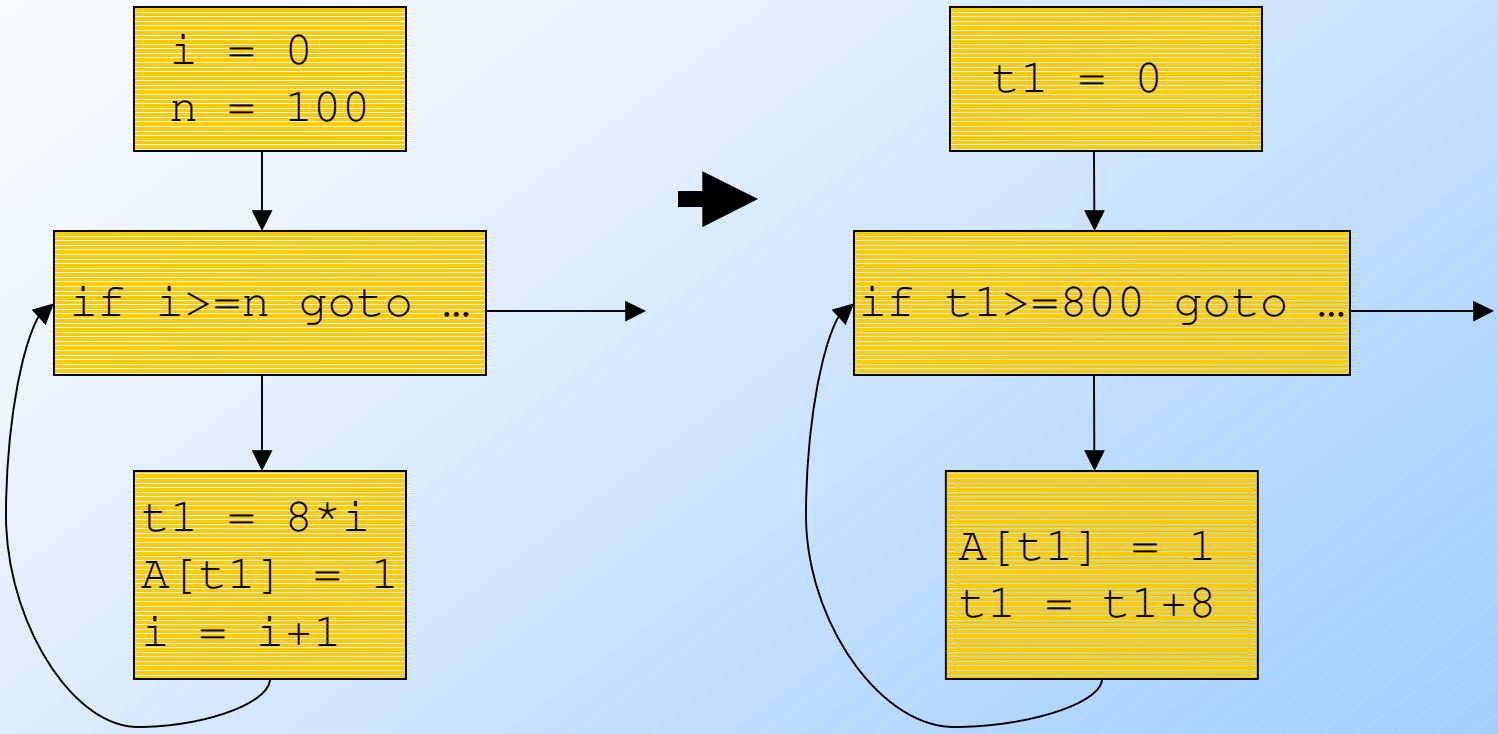
Example



Constant Folding

- Sometimes a variable has a known constant value at a point.
- If so, replacing the variable by the constant simplifies and speeds-up the code.
- Easy within a basic block; harder across blocks.

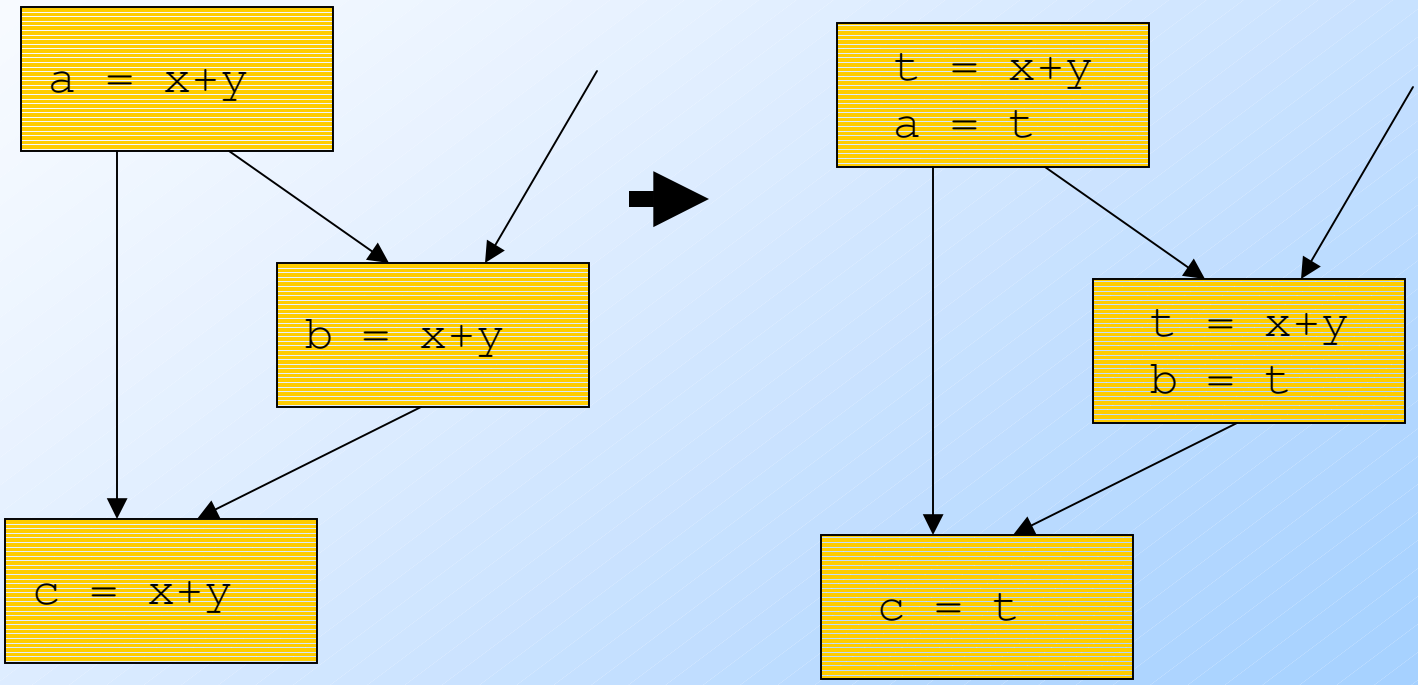
Example



Global Common Subexpressions

- Suppose block B has a computation of x+y.
- Suppose we are sure that when we reach this computation, we are sure to have:
  1. Computed x+y, and
  2. Not subsequently reassigned x or y.
- Then we can hold the value of x+y and use it in B.

Example



Example --- Even Better