Preliminary Transformations
Chapter 4, part 2

Vivek Pai
Comp 515
March 14, 1995

Topics

Forward Expression Substitution

Induction Variable Substitution

Forward Substitution

```
DO I = 1, 100
   K = I-2
   A(K) = A(K) + 5
ENDDO
```

Forward Substitution

```
DO I = 1, 100
   CONTINUE
   A(I-2) = A(I-2) + 5
ENDDO
```

Induction Var. Sub.

```
DO I = 1, 100
   KI = 0
   DO J = 1, 100
      KI = KI + 2
      A(KI) = A(KI) + B(J)
   ENDDO
ENDDO
```

Induction Var. Sub.

```
DO I = 1, 100
   KI = 0
   DO J = 1, 100
      CONTINUE
      A(J*2) = A(J*2) + B(J)
   ENDDO
ENDDO
```

Control Flow

Forward substitution

statement must always precede "victim"

Induction variable sub.

statement must be executed on every loop iteration

Dominators

Statement S1 dominates statement S2 with respect to statement S0 if every possible execution path from S0 to S2 contains S1

Using Dominators

Forward Substitution

S1 can be sub'ed into S2 if
S1 dominates S2 wrt first statement

Induction Variable Sub.

S can be an induction variable
if S dominates last wrt first

Forward Substitution

```
DO I =
   X = A + B + C + I
```

All defs of RHS must come from outside loop

RHS can't have side effects

For. Sub. Algorithm

For each use of variable X

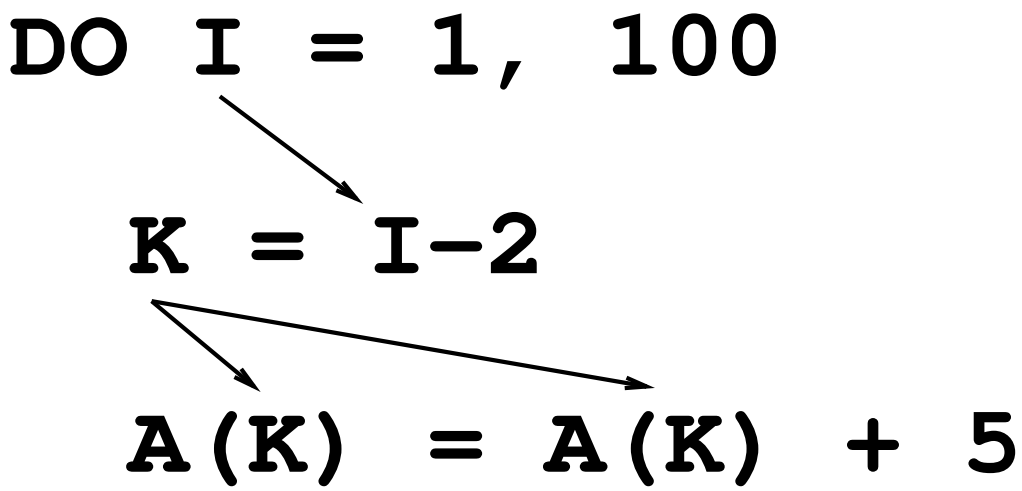
verify statement is dominated

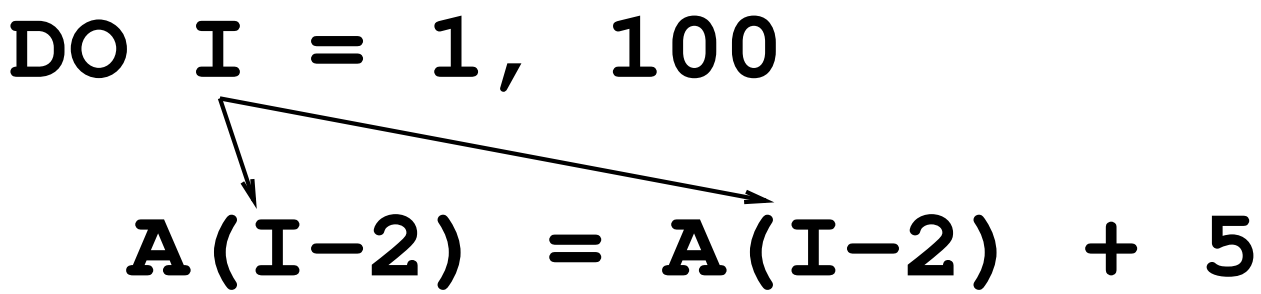
replace use with RHS of X

add new def–use edges into new code

remove old edges from X

New Def–Use Chains





Cleanup

If assignment has no uses, delete it

If it has uses, but only outside the loop, hoist it

Algorithm Code

Induction Var. Sub.

2 Parts

Testing – is it an IV?

Replacement

Testing

It's an IV if (generally)
X = X + (extra)

Executed on every iteration
Self–edge
IV dominated by addition
extra dominated by add or subtract
extra is loop–invariant

IV Testing

IV Substitution

For all uses of IV in loop

replace use with equivalent expression based on iter #

add def–use edges to the new uses of (extra)

add def–use edges to uses of IV

New Def–Uses

```
DO I = 1, 100
  KI = 0
  DO J = 1, 100
    KI = KI + 2
    A(KI) = A(KI) + B(J)
  ENDDO
ENDDO
```

```
DO I = 1, 100
  KI = 0
  DO J = 1, 100
    CONTINUE
    A(KI+J*2) = A(KI+J*2) + B(J)
  ENDDO
ENDDO
```

Cleanup

Move IV outside of loop

Add appropriate value to IV

Remove IV's self–edge if we know it can't reach

IV Sub Algorithm

Tying it together

Observations

IV Sub can create new IVs in outer loops

Forward sub can create new IVs

IVDrive

Closing Comments

IV Sub on unnormalized loops can be very bad

The forward substitution algorithm is a simplification