

15-411: Induction Variables

Jan Hoffmann

Induction Variables

- Variables in loops whose value changes by a constant in each iteration
- *Basic induction variable:*
 - gets increase or decreased by a constant in each iteration
 - Example: $i = i + 1$
- *Derived induction variable:*
 - linear function of another induction variable
 - Example: $x = 4 * i$

Example

Check if an array is sorted

```
bool is_sorted(int[] A, int n)
//@requires 0 <= n && n <= \length(A);
{
    for (int i = 0; i < n-1; i++)
        //@loop_invariant 0 <= i;
        if (A[i] > A[i+1]) return false;
    return true;
}
```

Basic induction
variable

Derived induction
variable

```
is_sorted( $A, n$ ) :  
     $i_0 \leftarrow 0$   
    goto loop( $i_0$ )  
loop( $i_1$ ) :  
     $t_0 \leftarrow n - 1$   
    if ( $i_1 \geq t_0$ ) goto rtrue  
     $t_1 \leftarrow 4 * i_1$   
     $t_2 \leftarrow A + t_1$   
     $t_3 \leftarrow M[t_2]$   
     $t_4 \leftarrow i_1 + 1$   
     $t_5 \leftarrow 4 * t_4$   
     $t_6 \leftarrow A + t_5$   
     $t_7 \leftarrow M[t_6]$   
    if ( $t_3 > t_7$ ) goto rfalse  
     $i_2 \leftarrow i_1 + 1$   
    goto loop( $i_2$ )  
rtrue :  
    return 1  
rfalse :  
    return 0
```

Translation to SSA Form
(without array bound checks)

A first optimization

```
is_sorted( $A, n$ ) :  
     $i_0 \leftarrow 0$   
    goto loop( $i_0$ )  
loop( $i_1$ ) :  
     $t_0 \leftarrow n - 1$   
    if ( $i_1 \geq t_0$ ) goto rtrue  
     $t_1 \leftarrow 4 * i_1$   
     $t_2 \leftarrow A + t_1$   
     $t_3 \leftarrow M[t_2]$   
     $t_4 \leftarrow i_1 + 1$   
     $t_5 \leftarrow 4 * t_4$   
     $t_6 \leftarrow A + t_5$   
     $t_7 \leftarrow M[t_6]$   
    if ( $t_3 > t_7$ ) goto rfals  
     $i_2 \leftarrow i_1 + 1$   
    goto loop( $i_2$ )
```

Common
subexpression
elimination
applies

Preserve
induction
variable i

```

is_sorted( $A, n$ ) :
     $i_0 \leftarrow 0$ 
    goto loop( $i_0$ )
loop( $i_1$ ) :
     $t_0 \leftarrow n - 1$ 
    if ( $i_1 \geq t_0$ ) goto rtrue
     $t_1 \leftarrow 4 * i_1$ 
     $t_2 \leftarrow A + t_1$ 
     $t_3 \leftarrow M[t_2]$ 
     $t_4 \leftarrow i_1 + 1$ 
     $t_5 \leftarrow 4 * t_4$ 
     $t_6 \leftarrow A + t_5$ 
     $t_7 \leftarrow M[t_6]$ 
    if ( $t_3 > t_7$ ) goto rfalse
     $i_2 \leftarrow t_4$ 
    goto loop( $i_2$ )

```

```

is_sorted( $A, n$ ) :
     $i_0 \leftarrow 0$ 
    goto loop( $i_0$ )
loop( $i_1$ ) :
     $t_0 \leftarrow n - 1$ 
    if ( $i_1 \geq t_0$ ) goto rtrue
     $t_1 \leftarrow 4 * i_1$ 
     $t_2 \leftarrow A + t_1$ 
     $t_3 \leftarrow M[t_2]$ 
     $i_2 \leftarrow i_1 + 1$ 
     $t_5 \leftarrow 4 * i_2$ 
     $t_6 \leftarrow A + t_5$ 
     $t_7 \leftarrow M[t_6]$ 
    if ( $t_3 > t_7$ ) goto rfalse

    goto loop( $i_2$ )

```

Consider induction
variable t_1

$$t_1 \leftarrow 4 * i_1$$

Idea: compute t_1 from a
previous iteration of t_1

Introduce new variable
 $j = 4*i$

is_sorted(A, n) :

$$i_0 \leftarrow 0$$

goto loop(i_0)

loop(i_1) :

$$t_0 \leftarrow n - 1$$

if ($i_1 \geq t_0$) goto rtrue

$$t_1 \leftarrow 4 * i_1$$

$$t_2 \leftarrow A + t_1$$

$$t_3 \leftarrow M[t_2]$$

$$i_2 \leftarrow i_1 + 1$$

$$t_5 \leftarrow 4 * i_2$$

$$t_6 \leftarrow A + t_5$$

$$t_7 \leftarrow M[t_6]$$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2)

Consider induction variable **t1**

$$t_1 \leftarrow 4 * i_1$$

Idea: compute **t1** from a previous iteration of **t1**

Introduce new variable **j = 4*i**

is_sorted(A, n) :

$$i_0 \leftarrow 0$$

$$j_0 \leftarrow 4 * i_0$$

goto loop(i_0, j_0)

loop(i_1, j_1) :

$$t_0 \leftarrow n - 1$$

if ($i_1 \geq t_0$) goto rtrue

$$t_1 \leftarrow j_1$$

$$t_2 \leftarrow A + t_1$$

$$t_3 \leftarrow M[t_2]$$

$$i_2 \leftarrow i_1 + 1$$

$$j_2 \leftarrow 4 * i_2$$

$$t_5 \leftarrow 4 * i_2$$

$$t_6 \leftarrow A + t_5$$

$$t_7 \leftarrow M[t_6]$$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2)

@ensures $j_0 = 4 * i_0$

@requires $j_1 = 4 * i_1$

@assert $j_1 = 4 * i_1$

@ensures $j_2 = 4 * i_2$

is_sorted(A, n) :

$i_0 \leftarrow 0$

$j_0 \leftarrow 4 * i_0$

goto loop(i_0, j_0)

loop(i_1, j_1) :

$t_0 \leftarrow n - 1$

if ($i_1 \geq t_0$) goto rtrue

$t_1 \leftarrow j_1$

$t_2 \leftarrow A + t_1$

$t_3 \leftarrow M[t_2]$

$i_2 \leftarrow i_1 + 1$

$j_2 \leftarrow 4 * i_2$

$t_5 \leftarrow 4 * i_2$

$t_6 \leftarrow A + t_5$

$t_7 \leftarrow M[t_6]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2)

$$j_0 = 4 * i_0 = 0$$

is_sorted(A, n) :

$i_0 \leftarrow 0$

$j_0 \leftarrow 0$

goto loop(i_0, j_0)

loop(i_1, j_1) :

$t_0 \leftarrow n - 1$

if ($i_1 \geq t_0$) goto rtrue

$t_1 \leftarrow j_1$

$t_2 \leftarrow A + t_1$

$t_3 \leftarrow M[t_2]$

$i_2 \leftarrow i_1 + 1$

$j_2 \leftarrow j_1 + 4$

$t_5 \leftarrow 4 * i_2$

$t_6 \leftarrow A + t_5$

$t_7 \leftarrow M[t_6]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2)

$$j_2 = 4 * i_2 = 4 * (i_1 + 1) = 4 * i_1 + 4 = j_1 + 4$$

@ensures $j_0 = 4 * i_0$

@requires $j_1 = 4 * i_1$

@assert $j_1 = 4 * i_1$

@ensures $j_2 = 4 * i_2$

is_sorted(A, n) :

$i_0 \leftarrow 0$

$j_0 \leftarrow 0$

goto loop(i_0, j_0)

loop(i_1, j_1) :

$t_0 \leftarrow n - 1$

if ($i_1 \geq t_0$) goto rtrue

$t_1 \leftarrow j_1$

$t_2 \leftarrow A + t_1$

$t_3 \leftarrow M[t_2]$

$i_2 \leftarrow i_1 + 1$

$j_2 \leftarrow j_1 + 4$

$t_5 \leftarrow 4 * i_2$

$t_6 \leftarrow A + t_5$

$t_7 \leftarrow M[t_6]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2)

is_sorted(A, n) :

$i_0 \leftarrow 0$

$j_0 \leftarrow 0$

$t_0 \leftarrow n - 1$

goto loop(i_0, j_0)

if ($i_1 \geq t_0$) goto rtrue

$t_2 \leftarrow A + j_1$

$t_3 \leftarrow M[t_2]$

$i_2 \leftarrow i_1 + 1$

$j_2 \leftarrow j_1 + 4$

$t_5 \leftarrow 4 * i_2$

$t_6 \leftarrow A + t_5$

$t_7 \leftarrow M[t_6]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2)

@ensures $j_0 = 4 * i_0$

@requires $j_1 = 4 * i_1$

@ensures $j_2 = 4 * i_2$

Loop hoisting

is_sorted(A, n) :

$i_0 \leftarrow 0$

$j_0 \leftarrow 0$

$t_0 \leftarrow n - 1$

goto loop(i_0, j_0)

loop(i_1, j_1) :

if ($i_1 \geq t_0$) goto rtrue

$t_2 \leftarrow A + j_1$

$t_3 \leftarrow M[t_2]$

$i_2 \leftarrow i_1 + 1$

$j_2 \leftarrow j_1 + 4$

$t_5 \leftarrow 4 * i_2$

$t_6 \leftarrow A + t_5$

$t_7 \leftarrow M[t_6]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2)

is_sorted(A, n) :

$i_0 \leftarrow 0$

$j_0 \leftarrow 0$

$t_0 \leftarrow n - 1$

goto loop(i_0, j_0)

loop(i_1, j_1) :

if ($i_1 \geq t_0$) goto rtrue

$t_2 \leftarrow A + j_1$

$t_3 \leftarrow M[t_2]$

$i_2 \leftarrow i_1 + 1$

$j_2 \leftarrow j_1 + 4$

$t_6 \leftarrow A + j_2$

$t_7 \leftarrow M[t_6]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2)

@ensures $j_0 = 4 * i_0$

@requires $j_1 = 4 * i_1$

@ensures $j_2 = 4 * i_2$

is_sorted(A, n) :

$i_0 \leftarrow 0$

$j_0 \leftarrow 0$

$t_0 \leftarrow n - 1$

goto loop(i_0, j_0)

loop(i_1, j_1) :

if ($i_1 \geq t_0$) goto rtrue

$t_2 \leftarrow A + j_1$

$t_3 \leftarrow M[t_2]$

$i_2 \leftarrow i_1 + 1$

$j_2 \leftarrow j_1 + 4$

$t_6 \leftarrow A + j_2$

$t_7 \leftarrow M[t_6]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2)

t2 is another
induction variable.

We introduce k to
keep track of $k = A + j$

is_sorted(A, n) :

$i_0 \leftarrow 0$

$j_0 \leftarrow 0$

$k_0 \leftarrow A + j_0$

$t_0 \leftarrow n - 1$

goto loop(i_0, j_0, k_0)

loop(i_1, j_1, k_1) :

if ($i_1 \geq t_0$) goto rtrue

$t_2 \leftarrow k_1$

$t_3 \leftarrow M[t_2]$

$i_2 \leftarrow i_1 + 1$

$j_2 \leftarrow j_1 + 4$

$k_2 \leftarrow k_1 + 4$

$t_6 \leftarrow A + j_2$

$t_7 \leftarrow M[t_6]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2, k_2)

@ensures $j_0 = 4 * i_0$

@ensures $k_0 = A + j_0$

@requires $j_1 = 4 * i_1 \wedge k_1 = A + j_1$

@ensures $j_2 = 4 * i_2$

@ensures $k_2 = A + j_2$

$$k_2 = A + j_2 = A + j_1 + 4 = k_1 + 4$$

is_sorted(A, n) :

$i_0 \leftarrow 0$

$j_0 \leftarrow 0$

$k_0 \leftarrow A + j_0$

$t_0 \leftarrow n - 1$

goto loop(i_0, j_0, k_0)

loop(i_1, j_1, k_1) :

if ($i_1 \geq t_0$) goto rtrue

$t_2 \leftarrow k_1$

$t_3 \leftarrow M[t_2]$

$i_2 \leftarrow i_1 + 1$

$j_2 \leftarrow j_1 + 4$

$k_2 \leftarrow k_1 + 4$

$t_6 \leftarrow A + j_2$

$t_7 \leftarrow M[t_6]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2, k_2)

is_sorted(A, n) :

$i_0 \leftarrow 0$

$j_0 \leftarrow 0$

$k_0 \leftarrow A$

$t_0 \leftarrow n - 1$

goto loop(i_0, j_0, k_0)

loop(i_1, j_1, k_1) :

if ($i_1 \geq t_0$) goto rtrue

$t_3 \leftarrow M[k_1]$

$i_2 \leftarrow i_1 + 1$

$j_2 \leftarrow j_1 + 4$

$k_2 \leftarrow k_1 + 4$

$t_7 \leftarrow M[k_2]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, j_2, k_2)

@ensures $j_0 = 4 * i_0$

@ensures $k_0 = A + j_0$

@requires $j_1 = 4 * i_1 \wedge$

@ensures $j_2 = 4 * i_2$

@ensures $k_2 = A + j_2$

One more round of constant propagation, dead code elim., and CSE

```

is_sorted( $A, n$ ) :
   $i_0 \leftarrow 0$ 
   $j_0 \leftarrow 0$ 
   $k_0 \leftarrow A$ 
   $t_0 \leftarrow n - 1$ 
  goto loop( $i_0, j_0, k_0$ )
loop( $i_1, j_1, k_1$ ) :
  if ( $i_1 \geq t_0$ ) goto rtrue
   $t_3 \leftarrow M[k_1]$ 
   $i_2 \leftarrow i_1 + 1$ 
   $j_2 \leftarrow j_1 + 4$ 
   $k_2 \leftarrow k_1 + 4$ 
   $t_7 \leftarrow M[k_2]$ 
  if ( $t_3 > t_7$ ) goto rfalse
  goto loop( $i_2, j_2, k_2$ )

```

$j_1, j_2,$ and j_3 are no longer needed.

```

is_sorted( $A, n$ ) :
   $i_0 \leftarrow 0$ 
   $k_0 \leftarrow A$ 
   $t_0 \leftarrow n - 1$ 
  goto loop( $i_0, k_0$ )
loop( $i_1, k_1$ ) :
  if ( $i_1 \geq t_0$ ) goto rtrue
   $t_3 \leftarrow M[k_1]$ 
   $i_2 \leftarrow i_1 + 1$ 
   $k_2 \leftarrow k_1 + 4$ 
   $t_7 \leftarrow M[k_2]$ 
  if ( $t_3 > t_7$ ) goto rfalse
  goto loop( $i_2, k_2$ )

```

@ensures $k_0 = A + 4 * i_0$

@requires $k_1 = A + 4 * i_1$

@ensures $k_2 = A + 4 * i_2$

is_sorted(A, n) :

$i_0 \leftarrow 0$

$k_0 \leftarrow A$

$t_0 \leftarrow n - 1$

goto loop(i_0, k_0)

loop(i_1, k_1) :

if ($i_1 \geq t_0$) goto rtrue

$t_3 \leftarrow M[k_1]$

$i_2 \leftarrow i_1 + 1$

$k_2 \leftarrow k_1 + 4$

$t_7 \leftarrow M[k_2]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, k_2)

is_sorted(A, n) :

$i_0 \leftarrow 0$

$k_0 \leftarrow A$

$t_0 \leftarrow n - 1$

goto loop(i_0, k_0)

loop(i_1, k_1) :

if ($k_1 \geq A + 4 * t_0$) goto rtrue

$t_3 \leftarrow M[k_1]$

$i_2 \leftarrow i_1 + 1$

$k_2 \leftarrow k_1 + 4$

$t_7 \leftarrow M[k_2]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, k_2)

$i_1 \geq t_0$ iff $A + 4 * i_1 \geq A + 4 * t_0$

is_sorted(A, n) :

$i_0 \leftarrow 0$

$k_0 \leftarrow A$

$t_0 \leftarrow n - 1$

goto loop(i_0, k_0)

loop(i_1, k_1) :

if ($k_1 \geq A + 4 * t_0$) goto rtrue

$t_3 \leftarrow M[k_1]$

$i_2 \leftarrow i_1 + 1$

$k_2 \leftarrow k_1 + 4$

$t_7 \leftarrow M[k_2]$

if ($t_3 > t_7$) goto rfalse

goto loop(i_2, k_2)

$i_0, i_1,$ and i_2 are no longer needed.

is_sorted(A, n) :

$k_0 \leftarrow A$

$t_0 \leftarrow n - 1$

$t_8 \leftarrow 4 * t_0$

$t_9 \leftarrow A + t_8$

goto loop(k_0)

loop(k_1) :

if ($k_1 \geq t_9$) goto rtrue

$t_3 \leftarrow M[k_1]$

$k_2 \leftarrow k_1 + 4$

$t_7 \leftarrow M[k_2]$

if ($t_3 > t_7$) goto rfalse

goto loop(k_2)

Unrolling the loop once can remove one memory access per iteration.