CS6013 - Modern Compilers: Theory and Practise Dependence testing

V. Krishna Nandivada

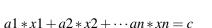
IIT Madras

for i ← 1 to 4 do $b[i] \leftarrow a[4*i] + 2.0$ a[2*i+1] ← 1.0/i endfor for i ← 1 to 4 do b[i] ← a[3*i-5] + 2.0 a[2*i+1] ← 1.0/i endfor



linear Diophantine equation

V.Krishna Nandivada (IIT Madras)



has an integer solution for $x_{1,x_{2,...}}$ iff

GCD $(a1, a2, \cdots an)$ divides c.

GCD test - intuition

- A simple and sufficient test
- if a loop carried dependency exists between X[a * i + b] and X[c * i + d], then GCD (c, a) must divide (d - b).

CS6013 - Jan 2015



GCD Test Generalization

```
for i_1 \leftarrow 1 to hi_1 do

for i_2 \leftarrow 1 to hi_2 do

for i_n \leftarrow 1 to hi_n do

...

for i_n \leftarrow 1 to hi_n do

...

\cdots x[..., a_0 + a_1 * i_1 + \cdots + a_n * i_n, ...] \cdots

...

endfor

endfor

endfor
```

- may be accessed inside loop nest using indices of multiple loops.
- Array may be multi-dimensional.
- Dependence present iff, for each subscript position in the equation

$$a_0 + \sum_{j=1}^n a_j * i_{j_1} = b_0 + \sum_{j=1}^n b_j * i_{j_2}$$

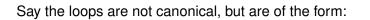
 $\forall i = 1 \cdots n$

and the following inequalities are satisfied:

V.Krishna Nandivada (IIT Madras)

 $1 \leq i_{j_1} \leq hi_j$ $1 \leq i_{j_2} \leq hi_j$ CS6013 - Jan 2015

GCD test for loops with arbitrary bounds



for $i_j \leftarrow lo_j$ by inc_j to hi_j

$$GCD\left(\bigcup_{j=1}^{n} Sep(a_j * inc_j, b_j * inc_j, j)\right) \neg / a_0 - b0 + \sum_{j=0}^{n} (a_j - b_j) * lo_j$$

GCD Test formula

- Developed by Utpal Bannerjee and Robert Towle (1976).
- Comparatively weak test (Marks too many accesses as dependent).
- If for any one subscript position

$$GCD\left(\bigcup_{j=1}^{n} Sep(a_j, b_j, j)\right) \neg / \sum_{j=0}^{n} (a_j - b_j)$$

where

• GCD - computes the Greatest common divisor for the set of numbers.

CS6013 - Jan 2015

• " $a \neg / b$ " means that a does not divide b.

 $Sep(a,b,j) = \begin{cases} \{a-b\} & \text{looking for intra iteration dependence} \\ \{a,b\} & \text{otherwise} \end{cases}$

then the two references to the array x are independent. • Other words: dependence \Rightarrow GCD divides the sum.

V.Krishna Nandivada (IIT Madras)

6/1

Dependence testing based on separability

- A pair of array references is <u>separable</u> if in each pair of subscript positions, the expressions found are of the form: a * x + b1 and a * x + b2.
- A pair of array references is <u>weakly separable</u> if in each pair of subscript positions, the expressions found are of the form: a1*x+b1 and a2*x+b2.



Dependence testing for separable array references

If the two array references are separable, then dependence exists if

- a = 0 and b1 = b2 or
- $(b1-b2)/a \leq hi_j$

Dependence testing for weakly separable array references

- For each subscript position, we have equations of the form: a1*y+b1 = a2*x+b2, or a1*y = a2*x+(b2-b1)
- Dependence exists if for a particular value of *j* has a solution that satisfies inequalities given by the loop bounds of loop *j*.
- List all such constraints for each reference.
- For any given reference if there is only one equation:
 - Say it is given by: $a_1 * y = a_2 * x + (b_2 b_1)$
 - One linear equation, two unknowns: Solution exists iff GCD(a1,a2)%(b2-b1) = 0

V.Krishna Nandivada (IIT Madras) CS6013 - Jan 2015

Dependence testing for weakly separable array references (contd)

• If the set of equations has two members of the form

 $a_{11} * y = a21 * x + (b21 - b11)$ $a_{12} * y = a22 * x + (b22 - b12)$

Two equations and two unknowns.

If a21/a11 = a22/a12 then rational solution exists: iff

$$b21 - b11)/a11 = (b22 - b12)/a12.$$

If $a21/a11 \neq a22/a12$ then there is one rational solution.

Once we obtain the solutions, check that they are integers and inequalities are satisfied.

If set of equations have n (> 2) members, either n − 2 are redundant → use previous methods.
 Else we have more equations compared to the unknowns → everdetermined



f[i] ← g[2*i,j] + 1.0
g[i+1,3*j] ← h[i,i] - 1.5
h[i+2,2*i-2] ← 1.0/i
endfor

V.Krishna Nandivada (IIT Madras)

for i ← 1 to n do

for j ← 1 to n do

Example: analyzing weak separable references

CS6013 - Jan 2015

What did we do today?

• Dependence testing.

