

Value Dependence Analysis

Announcement

- Need to make up November 14th lecture, November 30th class will be 2:10-3:50

Last time

- Data dependences for loops

Today

- Value dependence analysis

Versions of the Dependence Problem

Is there a data/memory dependence?

- in general this question is undecidable, for example indirect memory references (ie. $x[l[i]]$) throw a wrench into things
- equivalent to integer programming when the following assumptions are made:
 - structured procedure (i.e. reducible)
 - subscripts, loop bounds, and conditions are affine functions of the loop variables and loop-independent variables
 - loop steps are constant
- Input: IP problem
- Output: yes or no, is there a solution

Are integer programming problems in P or NP?

Versions of the Dependence Problem (cont...)

What is the distance or direction vector of the dependences?

- may require an exponential number of calls to a dependence testing algorithm that only returns yes/no
- Input: IP problem
- Output: distance or direction vector for dependences
- Example outputs: (1,0), (<), (<=), (<=>), (>=), (0,3)
- Which one of the above dependence vectors is not legal?

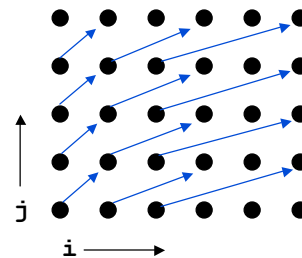
What is the dependence relation?

- mapping from one iteration space to another
- Input: Presburger formula (ie. affine constraints, existential and universal quantifiers, logical operators)
- Output: simplified presburger formula representing dependence relation
- Example input: $\{ [i, j] \rightarrow [i', j'] \mid 1 \leq i, j, i', j' \leq 10 \ \& \ i=i'-1 \ \& \ j=j' \ \& \ i < i' \ \& \ j < j' \}$
- Example output: $\{ [i, j] \rightarrow [i+1, j] \mid 1 \leq i, j \leq 10 \}$

Example

Sample code

```
do i = 1, 6
  do j = 1, 5
    A(2i, j) = A(i, j-1)
  enddo
enddo
```



Dependence

- $2i_1 - i_2 = 0, j_1 = j_2 - 1$, solution: YES

Distance/Direction Vector

- $(i_1, j_1) + (d_i, d_j) = (i_2, j_2), d_j = 1, d_i = ?, d = (<, 1)$

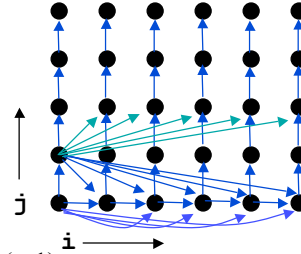
Dependence Relation

- $\{ [i, j] \rightarrow [2i, j+1] \mid 1 \leq i \leq 3 \ \& \ 1 \leq j \leq 4 \}$

We are done right?

Sample code

```
do i = 1,6
  do j = 1,5
    A(j) = A(j-1)+A(j)+A(j+1)
  enddo
enddo
```



Memory Deps:

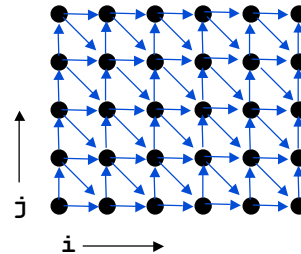
- distance/direction vectors: (0,1), (<,-1), (<,0),(+,1)
- dependence relations: $\{ [i, j] \rightarrow [i, j+1] : 1 \leq i \leq 6 \ \&\& \ 1 \leq j \leq 4 \}$,
 $\{ [i, j] \rightarrow [i', j-1] : 1 \leq i < i' \leq 6 \ \&\& \ 2 \leq j \leq 5 \}$,
 $\{ [i, j] \rightarrow [i', j] : 1 \leq i < i' \leq 6 \ \&\& \ 1 \leq j \leq 5 \}$,
 $\{ [i, j] \rightarrow [i', j+1] : 1 \leq i < i' \leq 6 \ \&\& \ 1 \leq j \leq 4 \}$

Can we remove all anti, output, and transitive flow dependences?

Improve Precision with Array Expansion

Sample code

```
do i = 1,6
  do j = 1,5
    A(i,j) = A(i,j-1)+A(i-1,j)
             +A(i-1,j+1)
  enddo
enddo
```



Memory Deps:

- distance/direction vectors: (0,1), (1,0), (1,-1)
- dependence relations: $\{ [i, j] \rightarrow [i, j+1] : 1 \leq i \leq 6 \ \&\& \ 1 \leq j \leq 4 \}$,
 $\{ [i, j] \rightarrow [i+1, j] : 1 \leq i \leq 5 \ \&\& \ 1 \leq j \leq 5 \}$,
 $\{ [i, j] \rightarrow [i+1, j-1] : 1 \leq i \leq 5 \ \&\& \ 2 \leq j \leq 5 \}$

Are these also value dependences?

Value Dependence Analysis

Get rid of false dependences without doing array expansion

Memory Dependence versus Value Dependence [Pugh & Wonnocott 93]

- There is a flow dependence from array access A(I) to A(J) iff
 - A is executed with iteration vector I
 - B is executed with iteration vector J
 - A(I) writes to the same location as is read by B(J)
 - A(I) is executed before B(J)
- there is no write to the location read by B(J) between the execution of A(I) and B(J)

Build Dependence Relation for Memory Dependence

Sample code

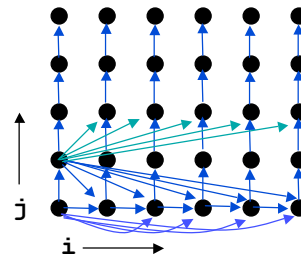
```
do i = 1,6
  do j = 1,5
    A(j) = A(j-1)+A(j)+A(j+1)
  enddo
enddo
```

A(j) write to A(j-1) read (+,1):

```
{[i,j]-> [i',j']} :
  1<=i,i'<=6 and 1<=j,j'<=5  \\ loop bounds
  and (i<i' or (i==i' && j<j')) \\ write before read
  and j=j'-1 }              \\ access same location
```

Simplified:

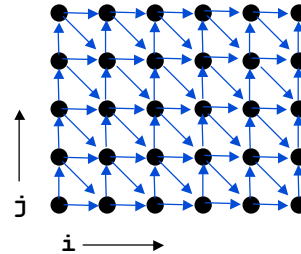
```
{[i,j] -> [i',j+1] : 1 <= i < i' <= 6 && 1 <= j <= 4}
union {[i,j] -> [i,j+1] : 1 <= i <= 6 && 1 <= j <= 4}
```



Build Dependence Relation for Value Dependence

Sample code

```
do i = 1,6
  do j = 1,5
    A(j) = A(j-1)+A(j)+A(j+1)
  enddo
enddo
```



A(j) write to A(j-1) read, (0,1):

```
{[i,j]-> [i',j']} :
  1<=i,i'<=6 and 1<=j,j'<=5  \\ loop bounds
  and (i<i' or (i==i' && j<j'))  \\ write before read
  and j=j'-1                    \\ access same location
  and ~(exists [i',j']st(1<=i'<=6 && 1<=j'<=5 // no intervening
    and (i,j)<<(i',j')<<(i'',j'')
    and (j'=j''-1) )
```

Simplified:

```
{[i,j] -> [i,j+1] : 1 <= i <= 6 && 1 <= j <= 4}
```

Omega Calculator Manipulates Relations

```
>../omega/bin/oc
# Omega Calculator v1.2 (based on Omega Library 1.2, August, 2000):
R := { [i,j] -> [i',j'] : 1<=i,i'<=6 && 1<=j,j'<=5
# R := { [i,j] -> [i',j'] : 1<=i,i'<=6 && 1<=j,j'<=5
&& (i<i' || (i=i' && j<j'))
# && (i<i' || (i=i' && j<j'))
&& j=j'-1};
# && j=j'-1};
#
R;
# R;
{[i,j] -> [i',j+1] : 1 <= i < i' <= 6 && 1 <= j <= 4} union
{[i,j] -> [i,j+1] : 1 <= i <= 6 && 1 <= j <= 4}
range R;
# range R;
{[i,j]: 1 <= i <= 6 && 2 <= j <= 5}
#
domain R;
# domain R;
{[i,j]: 1 <= i <= 6 && 1 <= j <= 4}
```

Petit Analyses Tiny Programs

```
>../omega/bin/petit -Rsimple.petit.out  
-b simple.t
```

```
>more simple.petit.out  
flow      1: Entry          -->  6: a(i-1,j+1)  
{[-1,In_2] -> [0,In_2-1] : 1 <= In_2 <= N} unid  
{[In_1,N] -> [In_1+1,N-1] : 0 <= In_1 <= N-2}  
flow      1: Entry          -->  6: a(i-1,j+1) [ M]  
{[In_1,In_2] -> [In_1+1,In_2-1] : -1 <= In_1 <= N-2 && 1 <= In_2 <= N}  
output    1: Entry          -->  6: a(i,j) [ M]  
{[In_1,In_2] -> [In_1,In_2] : 0 <= In_1 < N && 0 <= In_2 < N}  
flow      1: Entry          -->  4: N [ MV]  
{ -> TRUE }  
flow      1: Entry          -->  5: N [ MV]  
{ -> [i] : 0 <= i < N}  
flow      6: a(i,j)         -->  6: a(i-1,j+1) (1,-1) [ MVO]  
{[i,j] -> [i+1,j-1] : 0 <= i <= N-2 && 1 <= j < N}  
exact dd: {[1,-1]}  
flow      6: a(i,j)         -->  8: Exit [ MV]  
{[i,j] -> [i,j] : 0 <= i < N && 0 <= j < N}
```

```
simple.t  
-----  
integer i,j,N  
real a(0:99,0:99)  
  
for i=0,(N-1) do  
  for j=0,(N-1) do  
    a(i,j) = a(i-1,j+1)  
  endfor  
endfor
```

Concepts

Data Dependence Problems

- is there a dependence?
- what is the dependence distance or direction vector?
- what is the dependence relation?

Value Dependence Analysis gets the same results as full array expansion

Omega tools manipulate data dependence relations

Next Time

Lecture

- Loop transformations