The Polyhedral Model (Dependences and Transformations)

Announcements

- Project proposal is due THIS Friday

Today

- Automatic parallelization
- Transformation models/frameworks
- Polyhedral model
  - Iteration space representation
  - Data dependence problem and representation
  - Transformation representation and legality check

Protein String Matching Example (smithWaterman.c)

```
for (i=1;i<=a[0];i++) {
  for (j=1;j<=b[0];j++) {
    diag    = h[i-1][j-1] + sim[a[i]][b[j]];
    down    = h[i-1][j] + DELTA;
    right   = h[i][j-1] + DELTA;
    max=MAX3(diag,down,right);
    if (max <= 0)  {
      h[i][j]=0; xTraceback[i][j]=-1; yTraceback[i][j]=-1;
    } else if (max == diag) {
      h[i][j]=diag; xTraceback[i][j]=i-1; yTraceback[i][j]=j-1;
    } else if (max == down) {
      h[i][j]=down; xTraceback[i][j]=i-1; yTraceback[i][j]=j;
    } else  {
      h[i][j]=right; xTraceback[i][j]=i; yTraceback[i][j]=j-1;
    }
    if (max > Max){
       Max=max; xMax=i; yMax=j;
    }
 }}  // end for loops
```

Skewing smithWaterman.c

```
for (i=1;i<=a[0];i++) {
  for (j=1;j<=b[0];j++) {
    diag    = h[i-1][j-1] + sim[a[i]][b[j]];
    down    = h[i-1][j] + DELTA;
    right   = h[i][j-1] + DELTA;
    …
```

Let j'=i and i'=i+j.

Thus i=j' and j=i'-j'.

```
for (i'=2;i'<=a[0]+b[0];i'++) {
   for (j'=max(1,i'-b[0]); j'<=min(a[0],i'-1); j'++) {
     diag    = h[j'-1][i'-j'-1] + sim[a[i]][b[i'-j']];
     down    = h[j'-1][i'-j'] + DELTA;
     right   = h[j'][i'-j'-1] + DELTA;
     …
```


Automatic Parallelization

Input program has a set of operations E with a strict order

Find a partial order on E that is deterministic and results in the same output as the original strict total order.

Overall process

- Translate the code to a model
- Select a transformation/schedule
  - Determination of partial order on E, data dependence analysis
  - Ensure that the loop transformation/schedule is legal
- Transform the model and generate the transformed code

What should the model be?

Bernstein conditions

- Let u and v be operations, M(u)/M(v) be the set memory locations written by u/v, R(u)/R(v) be the set of memory locations read by u/v.
- If u precedes v and the intersection of M(u) and R(v) is non-empty, then there is a **flow** dependence.
- If u precedes v and the intersection of R(u) and M(v) is non-empty, then there is an **anti** dependence.
- If u precedes v and the intersection of M(u) and M(v) is non-empty, then there is an **output** dependence.

Problematic Example

$$M(u) = \{a^n + b^n | n > 2 \wedge a, b, n \in \mathbb{Z}\}$$

$$R(v) = \{c^n | n > 2 \wedge c, n \in \mathbb{Z}\}$$

Dependence Testing in General

General code

```
    do i1 = l1,h1
    ...
     do in = ln,hn
       ... A(f(i1,...,in))
       ... A(g(i1,...,in))
    enddo
    ...
enddo
```

There exists a dependence between iterations $I=(i_1, ..., i_n)$ and $J=(j_1, ..., j_n)$ when at least one of the accesses is a write and

- `f(I) = g(J)`
- $(l_1,...l_n) < I,J < (h_1,...,h_n)$
- `I << J or J << I, where << is lexicographically less`

Algorithms for Solving the Dependence Problem

Heuristics can say NO or MAYBE

- GCD test (Banerjee76,Towle76): determines whether integer solution is possible, no bounds checking
- Banerjee test (Banerjee 79): checks real bounds
- I-Test (Kong et al. 90): integer solution in real bounds
- Lambda test (Li et al. 90): all dimensions simultaneously
- Delta test (Goff et al. 91): pattern matches for efficiency
- Power test (Wolfe et al. 92): extended GCD and Fourier Motzkin combination

Exact solutions, exponential worst-case since integer linear programming is NP-complete

- Parametric Integer Programming (Feautrier91), based on the Simplex algorithm
- Omega test (Pugh92), based on the Fourier-Motzkin elimination algorithm

Dependence Testing

Consider the following code…

```
do i = 1,5
    A(3*i+2) = A(2*i+1)+1
enddo
```

Question

- How do we determine whether one array reference depends on another across iterations of an iteration space?

Dependence Testing: Simple Case

Sample code

```
do i = l,h
   A(a*i+c1) = ... A(a*i+c2)
enddo
```

Dependence?

- $a*i_1+c_1 = a*i_2+c_2$, or
- $a*i_1 - a*i_2 = c_2-c_1$
- Solution may exist if **a** divides c_2-c_1

GCD Test

Idea

- Generalize test to linear functions of iterators/induction variables

Code

```
do i = li,hi
   do j = lj,hj
      A(a1*i + a2*j + a0) = ... A(b1*i + b2*j + b0) ...
   enddo
enddo
```

Again

- $a_1*i_1 - b_1*i_2 + a_2*j_1 - b_2*j_2 = b_0 - a_0$
- Solution may exist if $gcd(a_1,a_2,b_1,b_2)$ divides $b_0 - a_0$

Example

Code

```
do i = l_i,h_i
   do j = l_j,h_j
      A(4*i + 2*j + 1) = ... A(6*i + 2*j + 4) ...
   enddo
enddo
```

gcd(4,-6,2,-2) = 2

Does 2 divide 4-1?

Banerjee Test

```
for (i=L; i<=U; i++) {
  x[a0 + a1*i] = ...
  ... = x[b0 + b1*i]
}
```

Does `a0 + a1*i = b0 + b1*i'` for some real `i` and `i'`?
If so then `(a1*i - b1*i') = (b0 - a0)`

Determine upper and lower bounds on `(a1*i - b1*i')`

```
for (i=1; i<=5; i++) {
   x[i+5] = x[i];
}
```

upper bound = a1*max(i) - b1 * min(i') = 4
lower bound = a1*min(i) - b1*max(i') = -4
b_0 - a_0 =

Polyhedron

(source: http://www.cse.ohio-state.edu/~pouchet/lectures/888.11.lect1.html)

Affine functions

- A function $f : \mathbb{K}^m \rightarrow \mathbb{K}^n$ is affine if there exists a vector $\vec{b} \in \mathbb{K}^n$ and a matrix $A \in \mathbb{K}^{n \times m}$ such that $\forall \vec{x} \in \mathbb{K}^m, f(\vec{x}) = A\vec{x} + \vec{b}$

Affine half spaces

- An affine half-space of $\mathbb{K}^m$ (affine constraint) is defined as a set of points $\{\vec{x} \in \mathbb{K}^m | \vec{a} \cdot \vec{x} \leq \vec{b}\}$

Polyhedron

- A set $S \in \mathbb{K}^m$ is a polyhedron if there exists a system of finite inequalities $A\vec{x} \leq \vec{b}$ such that $P = \{\vec{x} \in \mathbb{K}^m | A\vec{x} \leq \vec{b}\}$
- Equivalently it is the intersection of finitely many half-spaces.

Intersection between polyhedral sets

- When you intersect two polyhedral sets the results is a polyhedral set.
- Many questions we need to automate check whether a polyhedral set or sets are empty or not.
  - Is there a dependence at a certain loop level?
  - Is a transformation legal?

Next Time

Reading

- Advanced Compiler Optimizations for Supercomputers by Padua and Wolfe

Homework

- Project proposal due this Friday, October 17th

Lecture

- Code generation