Lecture 11
Loop Transformations for Parallelism and Locality

1. Examples
2. Affine Partitioning: Do-all
3. Affine Partitioning: Pipelining

Readings: Chapter 11-11.3, 11.6-11.7.4, 11.9-11.9.6
Shared Memory Machines

Performance on Shared Address Space Multiprocessors: Parallelism & Locality

Interconnect
Parallelism and Locality

- Parallelism DOES NOT imply speed up!

- Parallel performance:
  Improve locality with loop transformations
  - Minimize communication
  - Operations using the same data are executed on the same processor

- Sequential performance:
  Improve locality with loop transformations
  - Minimize cache misses
  - Operations using the same data are executed close in time.
Loop Permutation (Loop Interchange)

for $I = 1$ to $4$
for $J = 1$ to $3$
$Z[I,J] = Z[I-1,J]$

for $I = 1$ to $4$
for $J = 1$ to $3$
$Z[I,J] = Z[I-1,J]$

for $J' = 1$ to $3$
for $I' = 1$ to $4$
$Z[I',J'] = Z[I'-1,J']$

for $J' = 1$ to $3$
for $I' = 1$ to $4$
$Z[I',J'] = Z[I'-1,J']$

$\begin{bmatrix}
  j' \\
  i'
\end{bmatrix} = \begin{bmatrix}
  0 & 1 \\
  1 & 0
\end{bmatrix} \begin{bmatrix}
  i \\
  j
\end{bmatrix}$
Loop Fusion

for I = 1 to 4
    T[I] = A[I] + B[I] (s1)
for I' = 1 to 4
    C[I'] = T[I'] x T[I'] (s2)

for J = 1 to 4
    C[J] = T[J] x T[J] (s2)
Loop Transformations

• Unimodular transforms on loop nests
  – Interchange
  – Skewing
  – Reversal
• Cross statement transforms
  – Loop fusion
  – Loop fission
  – Re-indexing
• How to combine them to get parallelism and locality?
Affine Partitioning: An Contrived but Illustrative Example

\begin{align*}
\text{FOR } j = 1 \text{ TO } n \\
\text{FOR } i = 1 \text{ TO } n \\
A[i,j] &= A[i,j] + B[i-1,j]; \quad (S_1) \\
B[i,j] &= A[i,j-1] \times B[i,j]; \quad (S_2)
\end{align*}
Best Parallelization Scheme

Algorithm finds **affine partition mappings** for each instruction:

S1: Execute iteration \((i, j)\) on processor \(i-j\).

S2: Execute iteration \((i, j)\) on processor \(i-j+1\).

**SPMD code:** Let \(p\) be the processor’s ID number

```plaintext
if (1-n <= p <= n) then
  if [1 <= p) then
    B[p,1] = A[p,0] * B[p,1];  \hspace{1cm} (S_2)
  for i_1 = max[1,1+p) to min[n,n-1+p) do
    A[i_1,i_1-p] = A[i_1,i_1-p] + B[i_1-1,i_1-p]; \hspace{1cm} (S_1)
    B[i_1,i_1-p+1] = A[i_1,i_1-p] * B[i_1,i_1-p+1]; \hspace{1cm} (S_2)
  if (p <= 0) then
    A[n+p,n] = A[n+p,N] + B[n+p-1,n]; \hspace{1cm} (S_1)
```
2. Iteration Space

FOR $i = 0$ to $5$
    FOR $j = i$ to $7$
        ...

- $n$-deep loop nests: $n$-dimensional polytope
- Iterations: coordinates in the iteration space
- Assume: iteration index is incremented in the loop
- Sequential execution order: lexicographic order
  - $[0,0]$, $[0,1]$, $\ldots$, $[0,6]$, $[0,7]$, $[1,1]$, $\ldots$, $[1,6]$, $[1,7]$, $\ldots$
Maximum Parallelism & No Communication

For every pair of data dependent accesses $F_{11} + f_1$ and $F_{22} + f_2$

Find $C_1$, $c_1$, $C_2$, $c_2$:

$\forall i_1, i_2 \quad F_{11} + f_1 = F_{22} + f_2 \rightarrow C_{11} + c_1 = C_{22} + c_2$

with the objective of maximizing the rank of $C_1$, $C_2$
Rank of Partitioning = Degree of Parallelism

Affine Mapping

\[
\begin{bmatrix}
0 & 0 \\
0 & 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
0 & 1 \\
0 & 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
1 & 0 \\
0 & 1
\end{bmatrix}
\]

Rank

0

1

2

Mapped to same processor
Example 1: Loop Transform

Find affine partitioning: $c_1, c_2, c_0$ such that

$$p = \begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix} i \\ j \end{bmatrix} + c_0$$

Suppose iteration $i,j$ & $i', j'$ refer to same location

$$i = i' - 1$$
$$j = j'$$

No communication means:

$$c_1 i + c_2 j + c_0 = c_1 i' + c_2 j' + c_0$$

$$c_1(i'-1) + c_2 j' + c_0 = c_1 i' + c_2 j' + c_0$$

$$c_1 = 0$$
$$p = c_2 j + c_0$$

Pick simplest $c_2, c_0$: $c_2 = 1, c_0 = 0$

$$p = j$$
Code Generation

- Naive
  - Each processor visits all the iterations
  - Executes only if it owns that iteration
- Optimization
  - Removes unnecessary looping and condition evaluation
for I = 1 to 4
    for J = 1 to 3
        Z[I,J] = Z[I-1,J]

SPMD (single program multiple data) code:
for I = 1 to 4
    Z[I,P] = Z[I-1,P]
Loop Permutation (Loop Interchange)

for I = 1 to 4
  for J = 1 to 3
    Z[I,J] = Z[I-1,J]

for P = 1 to 3
  for I = 1 to 4
    Z[I,P] = Z[I-1,P]

\[
\begin{bmatrix}
  p' \\
  i'
\end{bmatrix} = \begin{bmatrix}
  0 & 1 \\
  1 & 0
\end{bmatrix} \begin{bmatrix}
  i \\
  j
\end{bmatrix}
\]
Example 2: Loop Fusion

Find affine partitioning: $c_{1,1}, c_{1,0}, c_{2,1}, c_{1,0}$, such that

$$s1: \begin{bmatrix} p \end{bmatrix} = \begin{bmatrix} c_{1,1} \end{bmatrix}[i] + c_{1,0}$$

$$s2: \begin{bmatrix} p \end{bmatrix} = \begin{bmatrix} c_{2,1} \end{bmatrix}[i'] + c_{2,0}$$

Suppose iteration $i$ & $i'$ refer to the same location

$$i = i'$$

No communication means:

$$c_{1,1} i + c_{1,0} = c_{2,1} i' + c_{2,0}$$

$$c_{1,1} = c_{2,1}$$
$$c_{1,0} = c_{2,0}$$

Pick simplest values: $c_{1,1} = c_{2,1} = 1$, $c_{1,0} = c_{2,0} = 0$

$p = i$; $p = i'$
for $I = 1$ to $4$

$T[I] = A[I] + B[I]$ \hspace{1cm} (s1)

for $I' = 1$ to $4$

$C[I'] = T[I'] \times T[I']$ \hspace{1cm} (s2)

$s1: \quad [p] = [1][i]$

$s2: \quad [p] = [1][i']$

for $P = 1$ to $4$

$T[P] = A[P] + B[P]$ \hspace{1cm} (s1)

$C[P] = T[P] \times T[P]$ \hspace{1cm} (s2)
Example 3: 2 Nested, Parallel Loops

for I = 1 to 4
  for J = 1 to 3
    Z[I,J] = Z[I,J]+1

Find affine partitioning: $c_1, c_2, c_0$ such that

$$p = \begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix} i \\ j \end{bmatrix} + c_0$$

Suppose iteration $i, j$ & $i', j'$ refer to same location

$$i = i'$$
$$j = j'$$

No communication means:

$$c_1 i + c_2 j + c_0 = c_1 i' + c_2 j' + c_0$$

$$c_1 i' + c_2 j' + c_0 = c_1 i + c_2 j + c_0$$

No constraints

Two basis vectors: $[c_1 c_2]=[1 0]$, or $[c_1 c_2]=[0 1]$

Two answers for $p$: two degrees of parallelism

$$\begin{bmatrix} p_1 \\ p_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} i \\ j \end{bmatrix}$$
Example 3: 2 Nested, Parallel Loops

for I = 1 to 4
  for J = 1 to 3
    Z[I,J] = Z[I,J]+1

I

J

for p1 = 1 to 4
  for p2 = 1 to 3
    Z[p1,p2] = Z[p1,p2]+1

for I = 1 to 4
  for J = 1 to 3
    if (I==p1 & J == p2)
      Z[I,J] = Z[I,J]+1

for p1 = 1 to 4
  for p2 = 1 to 3
    Z[p1,p2] = Z[p1,p2]+1
DO 1 J = 0, N
I0 = MAX ( -M, -J )
DO 2 I = I0, -1
   DO 3 JJ = I0 - I, -1
   DO 3 L = 0, NMAT
   DO 2 L = 0, NMAT
   A(L,I,J) = A(L,I,J) * A(L,0,I+J)
   DO 4 L = 0, NMAT
   EPSS(L) = EPS * A(L,0,J)
   DO 5 JJ = I0, -1
   DO 5 L = 0, NMAT
   A(L,0,J) = A(L,0,J) - A(L,JJ,J) ** 2
   DO 4 L = 0, NMAT
   A(L,0,J) = 1. / SQRT ( ABS (EPSS(L) + A(L,0,J)) )
   DO 6 I = 0, NRHS
   DO 7 K = 0, N
   DO 8 L = 0, NMAT
   B(I,L,K) = B(I,L,K) * A(L,0,K)
   DO 7 JJ = 1, MIN (M, N-K)
   DO 8 L = 0, NMAT
   B(I,L,K+JJ) = B(I,L,K+JJ) - A(L,-JJ,K+JJ) * B(I,L,K)
   DO 6 K = N, 0, -1
   DO 9 L = 0, NMAT
   B(I,L,K) = B(I,L,K) * A(L,0,K)
   DO 6 JJ = 1, MIN (M, K)
   DO 9 L = 0, NMAT
   B(I,L,K-JJ) = B(I,L,K-JJ) - A(L,-JJ,K) * B(I,L,K)
Chotst: Results with Affine Partitioning + Blocking

(Unimodular: a subset of affine partitioning for perfect loop nests)
Summary of Affine Partitioning

Communication-Free

Loops

Array

Processor ID

$F_1i_1 + f_1$

$F_2i_2 + f_2$

$C_1i_1 + c_1$

$C_2i_2 + c_2$
Advanced topic: Pipelining

SOR (Successive Over-Relaxation): An Example

for $i = 1$ TO $m$
  for $j = 1$ TO $n$
Finding the Maximum Degree of Pipelining

For every pair of data dependent accesses $F_1 i_1 + f_1$ and $F_2 i_2 + f_2$
Let $B_1 i_1 + b_1 \geq 0, B_2 i_2 + b_2 \geq 0$ be the corresponding loop bound constraints,
Find $C_1, c_1, C_2, c_2$:
$$\forall \ i_1, i_2 \quad B_1 i_1 + b_1 \geq 0, \ B_2 i_2 + b_2 \geq 0$$
$$(i_1 \leq i_2) \land (F_1 i_1 + f_1 = F_2 i_2 + f_2) \rightarrow C_1 i_1 + c_1 \leq C_2 i_2 + c_2$$
with the objective of maximizing the rank of $C_1, C_2$
**Key Insight**

- Choice in time mapping => (pipelined) parallelism
- Rank(C) - 1 degree of parallelism with 1 degree of synchronization
- Can create blocks with Rank(C) dimensions

- Find time partitions is not as straightforward as space partitions
  - Need to deal with linear inequalities
  - Solved using Farkas Lemma - no simple intuitive proof
Summary of Affine Partitioning

**Communication-Free**

<table>
<thead>
<tr>
<th>Processor ID</th>
<th>Array</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_1i_1+f_1$</td>
<td></td>
</tr>
<tr>
<td>$F_2i_2+f_2$</td>
<td></td>
</tr>
<tr>
<td>$C_1i_1+c_1$</td>
<td></td>
</tr>
<tr>
<td>$C_2i_2+c_2$</td>
<td></td>
</tr>
</tbody>
</table>

**Pipelining**

$\text{Loops} \quad i_1 \leq i_2$

<table>
<thead>
<tr>
<th>Array</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_2i_2+f_2$</td>
</tr>
<tr>
<td>$F_1i_1+f_1$</td>
</tr>
<tr>
<td>$C_1i_1+c_1$</td>
</tr>
<tr>
<td>$C_2i_2+c_2$</td>
</tr>
</tbody>
</table>

Time Stage