Chapter 11

Matrix Multiplication
Outline

• Sequential algorithms
  – Iterative, row-oriented
  – Recursive, block-oriented

• Parallel algorithms
  – Rowwise block striped decomposition
  – Cannon’s algorithm
Iterative, Row-oriented Algorithm

Series of inner product (dot product) operations
Performance as $n$ Increases

![Graph showing performance degradation as matrix size increases.](#)
Matrix B Gets Too Big for Cache

Reason:
Computing a row of C requires accessing every element of B
Block Matrix Multiplication

Replace scalar multiplication with matrix multiplication
Replace scalar addition with matrix addition
Recurse Until B Small Enough
Comparing Sequential Performance

![Graph comparing Megaflops vs Matrix Size for Block-oriented and Row-oriented approaches.](image)
First Parallel Algorithm

• Partitioning
  – Divide matrices into rows
  – Each primitive task has corresponding rows of three matrices

• Communication
  – Each task must eventually see every row of B
  – Organize tasks into a ring
First Parallel Algorithm (cont.)

• Agglomeration and mapping
  – Fixed number of tasks, each requiring same amount of computation
  – Regular communication among tasks
  – Strategy: Assign each process a contiguous group of rows
Communication of B
Communication of B
Communication of B
Communication of B
Complexity Analysis

- Algorithm has $p$ iterations
- During each iteration a process multiplies $(n / p) \times (n / p)$ block of A by $(n / p) \times n$ block of B: $\Theta(n^3 / p^2)$
- Total computation time: $\Theta(n^3 / p)$
- Each process ends up passing $(p-1)n^2/p = \Theta(n^2)$ elements of B
Isoefficiency Analysis

• Sequential algorithm: $\Theta(n^3)$
• Parallel overhead: $\Theta(pn^2)$

Isoefficiency relation: $n^3 \geq Cpn^2 \Rightarrow n \geq Cp$

$$M(Cp) / p = C^2 p^2 / p = C^2 p$$

• This system does not have good scalability
Weakness of Algorithm 1

- Blocks of B being manipulated have $\rho$ times more columns than rows
- Each process must access every element of matrix B
- Ratio of computations per communication is poor: only $\frac{2n}{\rho}$
Parallel Algorithm 2
(Cannon’s Algorithm)

- Associate a primitive task with each matrix element
- Agglomerate tasks responsible for a square (or nearly square) block of C
- Computation-to-communication ratio rises to \( n / \sqrt{p} \)
Elements of A and B Needed to Compute a Process’s Portion of C

Algorithm 1

Cannon’s Algorithm
Blocks Must Be Aligned

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0,0</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>0,1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0,2</td>
<td>2</td>
<td>0,3</td>
</tr>
<tr>
<td>0,3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1,0</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>1,1</td>
<td>1</td>
<td>1,2</td>
</tr>
<tr>
<td>1,3</td>
<td>2</td>
<td>1,3</td>
</tr>
<tr>
<td>2,0</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>2,1</td>
<td>2</td>
<td>2,2</td>
</tr>
<tr>
<td>2,3</td>
<td>3</td>
<td>2,3</td>
</tr>
<tr>
<td>3,0</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>3,1</td>
<td>3</td>
<td>3,2</td>
</tr>
<tr>
<td>3,3</td>
<td>0</td>
<td>3,3</td>
</tr>
</tbody>
</table>

Before:

- (a)

After:

- (b)
Blocks Need to Be Aligned

Each triangle represents a matrix block

Only same-color triangles should be multiplied
Rearrange Blocks

Block $A_{ij}$ cycles left $i$ positions

Block $B_{ij}$ cycles up $j$ positions
Consider Process $P_{1,2}$

Step 1
Consider Process $P_{1,2}$

Step 2
Consider Process $P_{1,2}$

Step 3
Consider Process $P_{1,2}$

Step 4
Complexity Analysis

- Algorithm has $\sqrt{p}$ iterations
- During each iteration process multiplies two $(n / \sqrt{p}) \times (n / \sqrt{p})$ matrices: $\Theta(n^3 / p^{3/2})$
- Computational complexity: $\Theta(n^3 / p)$
- During each iteration process sends and receives two blocks of size $(n / \sqrt{p}) \times (n / \sqrt{p})$
- Communication complexity: $\Theta(n^2 / \sqrt{p})$
Isoefficiency Analysis

- Sequential algorithm: $\Theta(n^3)$
- Parallel overhead: $\Theta(\sqrt{pn^2})$

Isoefficiency relation: $n^3 \geq C \sqrt{pn^2} \Rightarrow n \geq C \sqrt{p}$

\[ M(C\sqrt{p})/p = C^2 p/p = C^2 \]

- This system is highly scalable
Summary

• Considered two sequential algorithms
  – Iterative, row-oriented algorithm
  – Recursive, block-oriented algorithm
  – Second has better cache hit rate as $n$ increases

• Developed two parallel algorithms
  – First based on rowwise block striped decomposition
  – Second based on checkerboard block decomposition
  – Second algorithm is scalable, while first is not