Communication-Optimal Parallel Algorithm for Strassen’s Matrix Multiplication

Grey Ballard, James Demmel, Ben Lipshitz, Oded Schwartz

Summary

- Our algorithm is communication optimal
  - matches the recently proved communication lower bounds
  - moves asymptotically less data than all existing algorithms
- Our implementation is faster
  - than any classical algorithm can be
  - than any Strassen implementation we are aware of
- Strassen’s matrix multiplication is faster than classical
  - not just computation, but also communication
  - not just in theory, but also in practice
  - not just sequentially, but also in parallel

Asymptotics: Lower Bounds and Algorithms

<table>
<thead>
<tr>
<th>Classical</th>
<th>Flops</th>
<th>Bandwidth</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower Bound [4]</td>
<td>$\frac{n^2}{\omega}$</td>
<td>$\max\left{\frac{n^2}{\log P}, \frac{n^2}{\log P}\right}$</td>
<td>$\frac{n^2}{\log P}$</td>
</tr>
<tr>
<td>Cannon [2]</td>
<td>$\frac{n^2}{\omega}$</td>
<td>$\max\left{\frac{n^2}{\log P}, \frac{n^2}{\log P}\right}$</td>
<td>$\frac{n^2}{\log P}$</td>
</tr>
<tr>
<td>2.5D [6]</td>
<td>$\frac{n^2}{\omega}$</td>
<td>$\max\left{\frac{n^2}{\log P}, \frac{n^2}{\log P}\right}$</td>
<td>$\frac{n^2}{\log P}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Strassen</th>
<th>Flops</th>
<th>Bandwidth</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower Bound [1]</td>
<td>$\frac{n^2}{\omega}$</td>
<td>$\max\left{\frac{n^2}{\log P}, \frac{n^2}{\log P}\right}$</td>
<td>$\frac{n^2}{\log P}$</td>
</tr>
<tr>
<td>Cannon-Strassen [5]</td>
<td>$\frac{n^2}{\omega}$</td>
<td>$\max\left{\frac{n^2}{\log P}, \frac{n^2}{\log P}\right}$</td>
<td>$\frac{n^2}{\log P}$</td>
</tr>
<tr>
<td>Strassen-Cannon [3, 5]</td>
<td>$\frac{n^2}{\omega}$</td>
<td>$\max\left{\frac{n^2}{\log P}, \frac{n^2}{\log P}\right}$</td>
<td>$\frac{n^2}{\log P}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>New Algorithm</th>
<th>Flops</th>
<th>Bandwidth</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{n^2}{\omega}$</td>
<td>$\max\left{\frac{n^2}{\log P}, \frac{n^2}{\log P}\right}$</td>
<td>$\frac{n^2}{\log P}$</td>
<td></td>
</tr>
</tbody>
</table>

- $n$ is Matrix dimension
- $P$ is Number of processors
- $\omega$ is Exponent of matrix multiplication
- $\log 7 \approx 2.81$ for Strassen

Architectural implications

- Strassen reduces both computation and communication
- To remain compute bound: $\beta \leq \gamma M^{3/2-1}$ vs. $\beta \leq \gamma M^{1/2}$ for classical
- $\phi$ is the inverse bandwidth
- $\gamma$ is time per flop

Strassen-Winograd Algorithm

- Requires 7 multiplies and 15 additions for $2 \times 2$ matrix multiplication
- Requires $O(n^2)$ flops for $n \times n$ matrix multiplication
- Hidden constant is better than Strassen’s original algorithm

The Parallel Algorithm

- Key parallelization decision is to choose how to compute 7 subproblems
  - breadth first search ordering or depth first search ordering
- Independent decision can be made at each level of recursion tree

Breadth First Search (BFS) Step

- Runs all 7 multiplies in parallel
  - each uses $P/7$ processors
- Requires $7/4$ as much extra memory
- Requires communication
- Reduces future communication

Depth First Search (DFS) Step

- Runs all 7 multiplies sequentially
  - each uses all $P$ processors
- Requires $1/4$ as much extra memory
- No immediate communication
- Increases future communication

Perfect Strong Scaling Range

- Within range, both flops and communication scale linearly with $P$
- For largest problem that fits on $P_0$ processors, ranges up to $P_0^{3/2}$ processors
- For $P > P_0^{3/2}$, communication can no longer scale perfectly

Implementation Details

- Interleaving BFS and DFS
- Data Layout
- Local shared memory Strassen
- Hiding communication

Open Problems

- Analyze contention and optimize for it
- An efficient algorithm for arbitrary number of processors
- Fast parallel dense linear algebra: LU, QR, etc.
- Other practical fast matrix multiplication algorithms

References


Performance Data

Performance on Franklin (Cray XT4)

Breakdown of time for $n = 94080$

Perfect Strong Scaling Range

- Running on $P = m \cdot 7^k$
- Hybrid BFS steps for $m > 1$
- Hiding communication

Implementation Details

- Interleaving BFS and DFS
- Data Layout
- Local shared memory Strassen
- Hiding communication

Open Problems

- Analyze contention and optimize for it
- An efficient algorithm for arbitrary number of processors
- Fast parallel dense linear algebra: LU, QR, etc.
- Other practical fast matrix multiplication algorithms

Global Schemes

- Unlimited memory: do $k = \log_2 P$ BFS steps, then local computation
  - requires $O\left(\frac{n^3}{\sqrt{P}}\right)$ local memory footprint
- Limited memory: do $k = \log_2 \frac{n}{\alpha n}$ DFS steps, then $k$ BFS steps, then local computation
  - requires $O\left(\frac{n^3}{\sqrt{P}}\right)$ local memory footprint

Implementation Details

- Interleaving BFS and DFS
- Data Layout
- Local shared memory Strassen
- Hiding communication

Open Problems

- Analyze contention and optimize for it
- An efficient algorithm for arbitrary number of processors
- Fast parallel dense linear algebra: LU, QR, etc.
- Other practical fast matrix multiplication algorithms

References