Parallelizing GEMM within BLIS

Tyler Smith
Preview

- Opportunities for parallelism within GEMM
- First attempt at parallelism
- Failures -> More sophisticated methods
- What should parallelism within BLIS look like?
Inner Kernel vs Macro-kernel

- GotoBLAS has a monolithic inner kernel

- BLIS has a macro-kernel
  - This exposes 2 additional loops
  - Can parallelize at a finer granularity
Multiple Opportunities for Parallelism

for $j_0 = 0, \ldots, n - 1$ in steps of $n_d$
for $p_0 = 0, \ldots, k - 1$ in steps of $k_c$
for $i_0 = 0, \ldots, m - 1$ in steps of $m_c$
for $j_1 = 0, \ldots, n_d - 1$ in steps of $n_r$
  for $i_1 = 0, \ldots, m_c - 1$ in steps of $m_r$
    $C(i_1 : i_1 + m_r - 1, j_1 : j_1 + n_r - 1) \pm \ldots$
endfor
endfor
endfor
endfor

(loop around micro-kernel)
Multiple Opportunities for Parallelism

for $j_0 = 0, \ldots, n - 1$ in steps of $n_d$
for $p_0 = 0, \ldots, k - 1$ in steps of $k_c$
for $i_0 = 0, \ldots, m - 1$ in steps of $m_c$

for $j_1 = 0, \ldots, n_d - 1$ in steps of $n_r$
for $i_1 = 0, \ldots, m_c - 1$ in steps of $m_r$

$C(i_1 : i_1 + m_r - 1, j_1 : j_1 + n_r - 1) \pm \cdots$

endfor
endfor
endfor

for $j_0 = 0, \ldots, n - 1$ in steps of $n_d$
for $p_0 = 0, \ldots, k - 1$ in steps of $k_c$
for $i_0 = 0, \ldots, m - 1$ in steps of $m_c$
Multiple Opportunities for Parallelism

for $j_0 = 0, \ldots, n - 1$ in steps of $n_d$
for $p_0 = 0, \ldots, k - 1$ in steps of $k_c$

for $i_0 = 0, \ldots, m - 1$ in steps of $m_c$
for $j_1 = 0, \ldots, n_d - 1$ in steps of $n_r$
for $i_1 = 0, \ldots, m_c - 1$ in steps of $m_r$

$C(i_1:i_1 + m_r - 1, j_1:j_1 + n_r - 1) \pm \cdots$
endfor
endfor
endfor
endfor
First Attempt

- Parallelized
  - 2nd loop around micro-kernel
  - Packing routines
- Simple OpenMP pragmas
- Exploit new opportunities for parallelism
- Decent performance for several architectures
Intel Sandy Bridge E3

DGEMM ($m = n = 4000$)

- # cores=1
- # cores=2
- # cores=3
- # cores=4
ARM Cortex A9

DGEMM \( (m = n = 4000) \)

GFlops/core

\( k \)

\( \# \) cores = 1
\( \# \) cores = 2
IBM Power7

DGEMM \( (m = n = 4000) \)

Graph showing the performance of DGEMM with different number of cores.

- \# cores = 1
- \# cores = 2
- \# cores = 4
- \# cores = 8

GFlops/core vs. \( k \)
AMD A10 5800K

DGEMM \((m = n = 4000)\)

- # cores = 1
- # cores = 2
- # cores = 4

![Graph showing performance metrics for different core counts](image-url)
TI C6678 DSP

DGEMM \( (m = n = 4000) \)

![Graph showing GFlops/cores for different cores](image)
Intel Xeon Phi

DGEMM ($m = n = 14400$)

![Graph showing performance comparison between BLIS and MKL with different core counts.](image)

BLIS # cores = 1
BLIS # cores = 60
MKL # cores = 1
MKL # cores = 60
Intel Xeon Phi Problems

- Poor load balancing
  - 240 threads in the m dimension

- Poor amortization
  - Each block of A is moved into the L2 cache, only used with 7 or 8 slivers of B

- Poor use of L1 cache
Intel Xeon Phi

- Parallelize 3\textsuperscript{rd} loop around microkernel
  - Split the loop between 60 cores
- Parallelize 2\textsuperscript{nd} loop around microkernel
  - Split the loop between 4 threads
- Synchronize hardware threads
Intel Xeon Phi

DGEMM \((m = n = 14400)\)

\[
\begin{align*}
\text{GFlops/core} & \quad 0 & 5 & 10 & 15 \\
\text{k} & \quad 0 & 200 & 400 & 600 & 800 & 1,000 \\
\end{align*}
\]

- BLIS # cores = 1
- BLIS # cores = 60
- MKL # cores = 1
- MKL # cores = 60
Intel Xeon Phi

DGE MM \((m = n = 14400)\)

\[ \text{GFlops/core} \]

\[ k \]

- BLIS \# cores = 1
- BLIS \# cores = 60
- MKL \# cores = 1
- MKL \# cores = 60
What if we do the same for BG/Q

- Very similar to Xeon Phi
  - Low Power
  - Many Core
  - In-order
  - Multiple threads per core

<table>
<thead>
<tr>
<th></th>
<th>BG/Q</th>
<th>MIC</th>
</tr>
</thead>
<tbody>
<tr>
<td># Cores</td>
<td>16</td>
<td>60</td>
</tr>
<tr>
<td>Threads per Core</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td># L2 Caches</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>L2 Cache Size</td>
<td>32 MB</td>
<td>512 KB</td>
</tr>
</tbody>
</table>
IBM Blue Gene/Q

16 CORE DGEMM (M=N=10240)

\[ \text{GFLOPS/core} \]

\[ \log_{10}(k) \]

- **BLIS 1 Core**
- **BLIS 16 Cores**
- **ESSL 1 Core**
- **ESSL 16 Cores**
Blue Gene/Q

- Large, shared L2 cache
  - Problem:
    - Parallelizing 3rd loop means it holds multiple blocks of A
    - Decrease Size of A
    - Lower flops to memops ratio
  - Solution
    - Parallelize 1st and 2nd loops around micro-kernel
IBM Blue Gene/Q

16 CORE DGEMM (M=N=10240)

![Graph showing performance comparison between BLIS 1 Core, BLIS 16 Cores, ESSL 1 Core, and ESSL 16 Cores for 16 core DGEMM with M=N=10240. The graph plots GFLOPS/core against k.]
IBM Blue Gene/Q

16 Core DGEMM \( (M=N=10240) \)

![Graph showing performance comparison between BLIS 1 Core, BLIS 16 Cores, ESSL 1 Core, and ESSL 16 Cores for 16 Core DGEMM with \( M=N=10240 \).]
Requirements for Multithreaded BLIS

- Multiple levels of parallelism
- Map level of parallelism to hardware
  - We have some hypothesis on how to do this
  - Work in progress
Experimental Implementation

- Multiple levels of parallelism
- Hierarchical Groups
  - Conceptually similar to nested OpenMP parallelism
  - Use thread communicators to share data within groups
Experimental Implementation

- Thread Communicators
- Inspired by MPI Communicators
  - Provide:
    - Barriers
    - Locks
    - Broadcast
Future Work

- Extend to other BLIS operations
  - Bryan’s work within DxT
- High level interface
  - Express cache hierarchy
  - Direct BLIS on what hardware to use
- Analyze what levels to parallelize and why