Welcome to Ma 221! Lecture 10, Sep 15

Optimize Matmul: minimize comm between main mem + cache:

Do as many flops as possible given M words of data in cache

\[ C(i,j) \pm A(i,k)B(k,j) \]

\[ |V| \leq \sqrt{|V_A| \cdot |V_B| \cdot |V_C|} \]

\[ |V_A| + |V_B| + |V_C| = M \]

\[ |V| \leq M^{3/2} \]

What "shape" of V attains this bound?

A cube: \( M^{\frac{3}{2}} \times M^{\frac{3}{2}} \times M^{\frac{1}{2}} \)

Algorithm: break A, B, C into square submatrices that we can fit 3 in Cache
read 3 blocks into cache
Do local matmul, update C block
   no data movement
repeat
   \[
   A^{n \times n} = \begin{bmatrix}
   b & b & b & b \\
   b & & & \\
   b & & & \\
   b & & & 
   \end{bmatrix}
   \]
   \[A[i,j] \text{ is } b \times b \text{ block}\]
   for \(i = 1 \text{ to } n/b\)
   for \(j = 1 \text{ to } n/b\)
       read \(C[i,j]\) into cache, ... 1^2 \text{ words}
       for \(k = 1 \text{ to } n/b\)
           read \(A[i,k], B[k,j]\) into cache
           ... 2b^2 \text{ words}
           \[
           C[i,j] = C[i,j] + A[i,k] \times B[k,j]
           \]
       b \times b \text{ matmul, 3 more loops}
       no data movement
   end for
   write \(C[i,j]\) to main mem -- b^2 \text{ words}
   end for
end for

Total words moved = \(\frac{n^2}{b} \cdot \frac{n}{b} \cdot b^2 + \left(\frac{m}{b}\right)^2 2b^2 + \left(\frac{n}{b}\right)^2 b^2\)

\[= 2n^2 + 2 \frac{n^3}{b}\]

\(b \approx \sqrt[3]{\frac{m}{3}}\)
What about rectangular case? \( m \times n \times k \)
What if some dimension \( \leq \sqrt{M} \)?
Ex: \( y = Ax \) GEMV

Optimal Lower Bound = \( \Omega \left( \max \left( \frac{m \cdot n \cdot k}{M}, \frac{\text{size input}}{\text{size output}} \right) \right) \)

Idea of using Loomis-Whitney works for all dense LA, also extends to any algorithm that looks like
nested loops
accessing arrays
"any" subscripts like \( i, i+j, i-2j+k \)...

Can derive lower bound:

\( \# \text{words moved} = \Omega \left( \frac{\# \text{loop iterations}}{M^c} \right) \)

\( c \) depends on details of alg, uses generalization of Loomis-Whitney called Brascamp-Lieb (extension by Christ, Tao et al)
still open questions...

One problem: need to know \( M \)
What if there is more than one cache?
Goal: optimal matmul independent of HW

Use recursion

\[
\text{function } C = \text{RMM}(A, B) \\
\text{... RMM = Recursive Matmul} \\
\text{... Simple: } A^{n \times n}, B^{n \times n}, C^{n \times n}, n = 2^m \\
\text{... } C = \begin{bmatrix}
C_{11} & C_{12} \\
C_{21} & C_{22}
\end{bmatrix}^{n/2}
\begin{bmatrix}
C_{11} & C_{12} \\
C_{21} & C_{22}
\end{bmatrix}^{n/2}
\]

\[\text{ditto for } A, B\]

if \( n = 1 \), \( C = A \cdot B \), else

\[C_{ii} = \text{RMM}(A_{ii}, B_{ii}) + \text{RMM}(A_{12}, B_{21})\]

\[\text{ditto for } C_{12}, C_{21}, C_{22}\]

Correct by induction

Cost analysis:

\[A(n) = \# \text{ arithmetic ops} = 8 \cdot A(\frac{n}{2}) + 4 \left(\frac{n}{2}\right)^2, \quad A(1) = 1\]

Solve: \( n = 2^m \)

\[a(m) = A(2^m) = 8 \cdot a(m-1) + \frac{2^m}{8^m} \]

\[\frac{a(m)}{8^m} = \frac{a(m-1)}{8^{m-1}} + \frac{2^m}{8^m}\]
\[ b(m) \]
\[ b(m) = b(m-1) + \frac{1}{2^m} \quad \text{...geometric sum} \]
get \[ A(n) = 2n^3 + \text{lower order term} \]

\[ W(n) = \# \text{words moved} = 8W(\frac{n}{2}) + 4.3(\frac{n}{2})^2 \]

recurrence stops when matrices fit in cache \[ 3 \cdot 2^m = M \]
\[ W(b) = 3b^2 \quad \text{if } 3b^2 = M \]

Still geometric sum \[ W(n) = O\left(\frac{n^3}{\sqrt{M}}\right) \]

"Cache oblivious algorithm" works for any cache size \( M \), any number of levels of cache

Lower bound on comm extends to parallel \text{\texttt{mutmul}}

\[ \begin{array}{c}
\text{P1} \\
\downarrow \quad \downarrow \\
| \quad | \\
\text{P3} \\
\uparrow \quad \uparrow \\
\text{P2} \\
\text{P4}
\end{array} \]
\( P \) processors each stores \( \frac{1}{P} \) of all data each does \( \frac{1}{P} \) of all work

"fast memory" = local on processor
"slow memory" = memory on other proc.

Same lower bound \( \Omega\left(\frac{\text{\#flops per proc}}{\sqrt{\text{mem per proc}}}\right) \)
\[ = \Omega\left(\frac{n^3/p}{\sqrt{3n^2/p}}\right) = \Omega\left(\frac{n^2}{\sqrt{p}}\right) \]
attainable by "SUMMA"
can be beaten by using more memory
and replicating data (see CS267)

Going faster than $O(n^3)$ FLOPS
Strassen (1967): matrix possible in $O(n^{\log_2 7})$
operations $\approx O(n^{2.81})$
Trick: RMM using 7 recursive calls, not 8
$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$
$P_1 = (A_{12} - A_{22}) \cdot (B_{21} + B_{22})$
$P_2 = (A_{11} + A_{22}) \cdot (B_{11} + B_{22})$
$\vdots$
$P_7 = \vdots$
$C_{11} = P_1 + P_2 - P_4 + P_6 + 18 \text{ matrix additions}$
$C_{22} = \vdots$
$A(n) = 7A(C_{22}^\frac{1}{2}) + 18(C_{22})^2 = O(n^{\log_2 7})$
$W(n) = O\left(\frac{n^w}{m^{w_1 w_2 - 1}}\right)$
$w = \log_2 7$
Thm (2010): WC(n) attains lower bound
Thm (2015): Extends lower bound to all "Strassen-like" algorithms
What is smallest \( w \)? July 2023

\[ w = 2.371552 \quad (\text{Williams} + Xu + Xu + Zhou) \]

(not practical)

Nature article last year: using ML to discover new matrix alg.

Thr, (2008): All linear algebra can be done in \( O(n^w) \)

Error Analysis for Strassen

Classical: \[ \| f(A \cdot B) - A \cdot B \| \leq \varepsilon \| A \| \cdot \| B \| \]

Strassen-like: \[ \| f_l(A \cdot B) - A \cdot B \| = O(\varepsilon) \| A \| \cdot \| B \| \]

Gauss's Trick for matmul

complex

\[ (A + iB) \cdot (C + iD) = (T_1 - T_2) + i(T_3 - T_1 - T_2) \]

usual: 4 matrix-multiplications of real matrices

\[ T_1 = A \cdot C \quad T_2 = B \cdot D \]

\[ T_3 = (A + B) \cdot (C + D) \]

\( \text{cost: 3 real matmuls + 5 adds} \)

classical: 4 real " + 2 "

\( \text{cost} = \frac{3}{4} \text{ the classical formula} \)