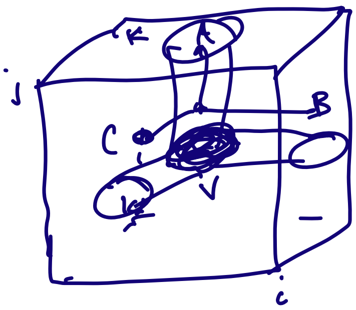
Welcome to Ma221! Lecture 10, Sep 15

Optimize Matmul: minimize comm between main mem + cache:

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



$C(i,j) \pm A(i,k) B(k,j)$

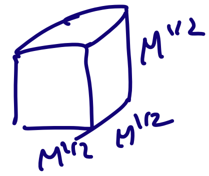
$\#\text{iterations} = |V| \le \sqrt{|V_A| \cdot |V_B| \cdot |V_C|}$

$|V_A| + |V_B| + |V_C| = M$

$|V| \lessapprox M^{3/2}$

What "shape" of V attains this bound?

a cube: $M^{\frac{1}{2}} \times M^{\frac{1}{2}} \times M^{1/2}$



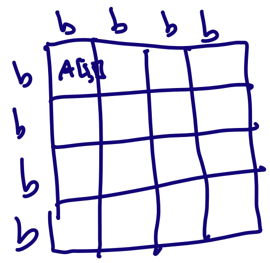
Algorithm: break A, B, C into square submatrices that we can fit 3 in Cache read 3 blocks into cache

Do local matmul, update C block

no data movement

repeat

$A^{n\times n}$ =



$A[i,j]$ is $b\times b$ block

for $i = 1$ to n/b

  for $j = 1$ to n/b

    read $C[i,j]$ into cache, ... b^2 words

    for $k = 1$ to n/b

      read $A[i,k]$, $B[k,j]$ into cache

      ... $2b^2$ words

      $C[i,j] = C[i,j] + A[i,k]\cdot B[k,j]$

      $b\times b$ matmul, 3 more loops

      no data movement

    endfor

    write $C[i,j]$ to main mem -- b^2 words

  endfor

endfor

Total words moved $= \underbrace{\frac{n}{b}\cdot\frac{n}{b}\cdot b^2}^{\text{read } C} + \overbrace{\left(\frac{n}{b}\right)^3 2b^2}^{\text{read } A, B} + \overbrace{\left(\frac{n}{b}\right)^2 b^2}^{\text{write}}$

$= 2n^2 + 2\frac{n^3}{b}$

$b \approx \sqrt{\frac{M}{3}}$

What about rectangular case? $m \times n \times k$

What if some dimension $< \sqrt{M}$?

Ex: $y = Ax$ GEMV

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

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, $i+j$, $i-2j+k$ ...

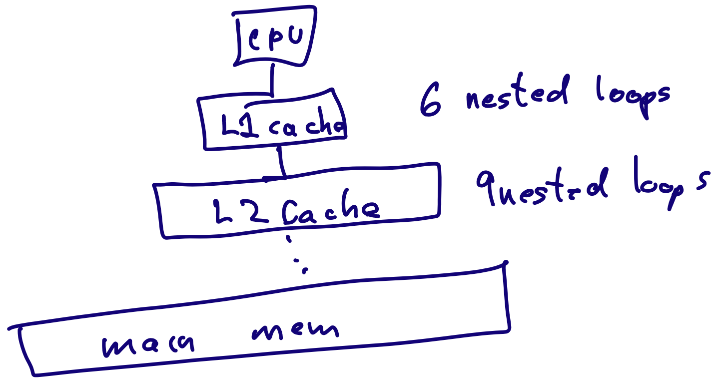
Can derive lower bound:

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

e 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

function $C = RMM(A,B)$

... RMM = Recursive Matmul

... Simple: $A^{n\times n}, B^{n\times n}, C^{n\times n}$, $n = 2^m$

... $C = \begin{array}{cc} \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} & \begin{array}{c} n/2 \\ n/2 \end{array} \\ \begin{array}{cc} n/2 & n/2 \end{array} & \end{array}$

... ditto for A, B

if $n=1$, $C = A\cdot B$, else

$C_{11} = RMM(A_{11}, B_{11}) + RMM(A_{12}, B_{21})$

... ditto for C_{12}, C_{21}, C_{22}

Correct by induction

Cost analysis:

$A(n) = \#$ arithmetic ops

$= 8\,A(\tfrac{n}{2}) + 4(\tfrac{n}{2})^2, \quad A(1) = 1$

Solve: $n = 2^m$

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

$b(m)$

$$b(m) = b(m-1) + \frac{1}{2^m}$$ ... geometric sum

get $A(n) = 2n^3$ + lower order term

$W(n)$ = # words moved

$$= 8\,W\left(\tfrac{n}{2}\right) + 4\cdot 3\left(\tfrac{n}{2}\right)^2$$

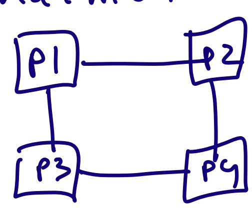
recurrence stops when matrices fit in cache

$3\cdot 2^m = M$ $\quad W(b) = 3b^2$ 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 matmul



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 procs.

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): matmul possible in $O(n^{\log_2 7})$ operations $\approx O(n^{2.81})$

trick: RMM using 7 recursive calls, not 8

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

$$\left.\begin{array}{l} P1 = (A_{12} - A_{22}) \cdot (B_{21} + B_{22}) \\ P2 = (A_{11} + A_{22}) \cdot (B_{11} + B_{22}) \\ \vdots \\ P7 = \end{array}\right\} \text{7 recursive calls}$$

$C_{11} = P1 + P2 - P4 + P6$ + 18 matrix additions

$\vdots$

$C_{22} = \cdots$

$$A(n) = 7A\left(\tfrac{n}{2}\right) + 18\left(\tfrac{n}{2}\right)^2 = O(n^{\log_2 7})$$

$$W(n) = O\left(\frac{n^w}{M^{w/2-1}}\right) \qquad w = \log_2 7$$

Thm (2010) $W(n)$ attains lower bound

Thm (2015) Extends lower bound to all "Strassen-like" algorithms

What is smallest ω? July 2023

$\omega = 2.371552$ (Williams + Xu + Xu + Zhou)

(not practical)

Nature article last year: using ML to discover new matmul algs

Thm (2008): All linear algebra can be done in $O(n^\omega)$

Error Analysis for Strassen

Classical: $|fl(A\cdot B) - A\cdot B| \le n\varepsilon |A|\cdot|B|$

Strassen-like: $\|fl(A\cdot B) - A\cdot B\| = O(\varepsilon)\|A\|\cdot\|B\|$

Gauss's Trick for complex matmul

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

usual: 4 matmuls of real matrices

$T_1 = A\cdot C \quad T_2 = B\cdot D$

$T_3 = (A+B)\cdot(C+D)$

cost: 3 real matmuls + 5 adds

classical: 4 real " + 2 "

cost $= \frac{3}{4}$ the classical formula