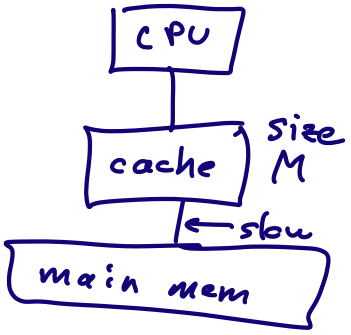
Welcome to Ma 221! Lec 7, Fall 24

Last time: considered matmul on



lower bound on # words moved between cache and main mem

$$= \Omega\left(\frac{n^3}{\sqrt{M}}\right) = \Omega\left(\frac{\#\text{ flops}}{\sqrt{M}}\right)$$

enough for linear algebra, which looks like 3 nested loops with $C(i,j) = C(i,j) + A(i,k) \cdot B(k,j)$

used Loomis-Whitney

Generalizes to any algorithm that

has nested loops (any number)

any number of arrays

any subscripts $A(i)$, $B(i, i+j)$, $C(i+2j-3k, \ldots)$

Can get a lower bound and optimal alg

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

e depends on details of subscripts, need generalization of Loomis-Whitney to get e

called Hölder-Brascamp-Lieb inequality

Last time: optimal matmul, minimized communication by "blocking", breaking up matrices into $b \times b$ submatrices

$b \sim \sqrt{M/3}$

Problems: Need to know about hardware:
depends on M, multiple levels of cache
may need 6 nested loops, or 9, or 12, ...

Goal: optimal matmul independent of HW

use recursion:

function: $C = RMM(A, B)$
... RMM = Recursive Mat Mul
... Simplicity: assume $A^{n \times n}, B^{n \times n}, n = 2^m$
if $n = 1$
$C = A \cdot B$
else ... $C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$ ($n/2$ rows, $n/2$ cols)

$C_{11} = RMM(A_{11}, B_{11}) + RMM(A_{12}, B_{21})$
ditto for C_{12}, C_{21}, C_{22}
end if

correct by induction

Cost analysis

$A(n)$ = # arithmetic ops for size n

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

solve for $n = 2^m$

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

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

geometric sum

answer is $2n^3 - n^2$ same as usual matmul

$W(n)$ = # words moved

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

$= 8\,W(\frac{n}{2}) + 3 \cdot n^2$

Base case when all 3 matrices fit in cache

$W(b) = 3b^2$ if $3b^2 \le M$

$b = \sqrt{M/3}$

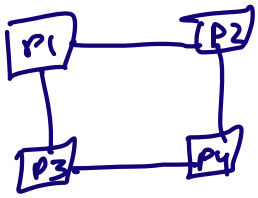
Still geometric sum, $W(n) = O\left(\frac{n^3}{\sqrt{M}}\right)$

"Cache Oblivious": works for

any cache size, any # levels of cache,

works for much of linear algebra

Extension to Parallel case



want to minimize communication over network

each processor does $\frac{1}{P}$ of flops "load balanced"

each processor stores $\frac{1}{P}$ of all data "memory balanced"

"fast memory" = local to each proc

"slow memory" = all rest of 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

Surprise: can use more memory to replicate data

→ denominator bigger

→ lower bound lower : attainable!

Going faster than n^3 flops

Strassen (1967) matmul possible in $O(n^{\log_2 7})$ flops $\sim O(n^{2.81})$

trick: recursion with 7 calls, not 8

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

(blocks of size $n/2 \times n/2$)

$$P1 = (A_{12} - A_{22}) \cdot (B_{21} + B_{22})$$
$$P2 = (A_{11} + A_{22}) \cdot (B_{11} + B_{22})$$
$$\vdots$$
$$P7 = \ldots$$

7 recursive calls

$$C_{11} = P1 + P2 - P_4 + P_6$$
$$C_{12} = P_4 + P_5$$
$$\vdots$$

total of 18 add/subs of $(\frac{n}{2}) \times (\frac{n}{2})$ matrices

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

$w = \log_2 7$

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

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

Thm (2015) Extends to all "Strassen-like" algs

Lastest record for smallest ω

Thm (2024, Williams, Xu, Xu, Zhou) $\omega = 2.371552$

not practical, Strassen is practical

Thm (2008) All linear algebra can be done in $O(n^\omega)$ ops and "stably"

Error Analysis for Strassen:

Recall usual n^3 bound (Q1.10)

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

Strassen:

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

Gauss's trick for complex matmul

$$(A + iB) \cdot (C + iD)$$

$$T_1 = A \cdot C$$

$$T_2 = B \cdot D$$

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

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

cost : 3 matmuls + 5 add/sub

vs : 4 matmuls + 2 add/sub

cost = $\frac{3}{4}$ of usual matmul

Gaussian Elimination

Additional Goals, on top of avoiding communication, $O(n^\omega)$ flops

: Backward stability: exact solution of $(A+E)\hat{x} = b+f$

$$\frac{\|E\|}{\|A\|} = O(\varepsilon), \quad \frac{\|f\|}{\|b\|} = O(\varepsilon)$$

: Exploit math. structure of A

A: symmetric, positive definite

"sparse" = depends on $\ll n^2$ parameters

so could have many 0 entries, or be dense and depend on $\ll n^2$ parameters

Eg: Vandermonde Matrix

$V_{ij} = x_i^{j-1}$ given $[x_1, \ldots, x_n]$

Ex: multiply $V \cdot c = b$

= polynomial evaluation

$b_i = \sum_j c_j x_i^{j-1}$

solving $Vc = b$ for c

polynomial interpolation

much cheaper than $O(n^3)$

Seek Matrix Factorization

A = product of simpler matrices

SVD: $A = U \Sigma V^T$
= orthog · diag · orthog

GaussElim $A = P \cdot L \cdot U$

P = permutation

L = lower triangular
("unit", $L_{ii} = 1$)

U = upper triangular

LeastSquares $A = QR$
= orthog · upper trang

Eigenproblems $A = Q \cdot T \cdot Q^T$
Q orthog
T triang.

Def: Permutation Matrix
: identity matrix with permuted rows

Facts: let P, P_1, P_2 be perms.

P has exactly one 1 in each row and each column

$P \cdot X = X$ with permuted rows

$X \cdot P = X$ with permuted columns

$P_1 \cdot P_2$ = permutation

$P^{-1} = P^T$ i.e. P is orthogonal

proof: $(P^T P)_{ii} = 1 \Rightarrow P^T P = I$

$\det(P) = \pm 1$

storing and multiplying by P is cheap,
(store indices of 1s, copy rows)

Thm: (LU decomposition)

Given any $m \times n$ full rank A, $m \geq n$

$\exists$ $m \times m$ perm P

$m \times n$ unit lower triangular $L = \begin{bmatrix} 1 & & 0 \\ & \ddots & \\ * & & 1 \\ & * & \end{bmatrix}$

$n \times n$ nonsingular U

s.t. $A = P \cdot L \cdot U$

to solve $Ax = b$, A square

(1) Factor $A = P \cdot L \cdot U$

(expensive part: #flops $= \frac{2}{3}n^3 + O(n^2)$)

(2) Solve $P \cdot L \cdot U \cdot x = b$ for $L \cdot U \cdot x = P^T \cdot b$

by permuting b, cost $= O(n)$

(3) Solve $L \cdot U \cdot x = P^T \cdot b$ for $U \cdot x = L^{-1} \cdot P^T \cdot b$

by forward substitution

cost $= n^2$

(4) Solve $U \cdot x = L^{-1} \cdot P^T \cdot b$ for $x = U^{-1} \cdot L^{-1} \cdot P^T \cdot b$
by back substitution, cost $= n^2$

Given another b' can solve $Ax' = b'$
in just $O(n^2)$ flops

Note: Should never compute A^{-1} and multiply by it because
(1) 3x more expensive in dense case
(much worse in sparse case, up to $O(n^2)$ slower)
(2) not numerically stable

Proof of $A = P \cdot L \cdot U$ (Gauss Elim)

If A full rank $\Rightarrow$ first col nonzero
$\Rightarrow \exists$ perm P s.t. $(PA)(1,1) \neq 0$

$$PA = \begin{matrix} \\ m-1 \end{matrix} \overset{\begin{matrix} 1 & n-1 \end{matrix}}{\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}} = \begin{matrix} \\ m-1 \end{matrix} \overset{\begin{matrix} 1 & m-1 \end{matrix}}{\begin{bmatrix} 1 & 0 \\ \frac{A_{21}}{A_{11}} & I \end{bmatrix}} \cdot \begin{matrix} \\ m-1 \end{matrix} \overset{\begin{matrix} 1 & n-1 \end{matrix}}{\begin{bmatrix} A_{11} & A_{12} \\ 0 & \underbrace{A_{22} - \frac{A_{21} A_{12}}{A_{11}}} \end{bmatrix}}$$

S = Schur Complement

A full rank $\Rightarrow PA$ full rank
$\Rightarrow S$ full rank: otherwise $\exists x \neq 0 : Sx = 0$
$\Rightarrow A \begin{bmatrix} -A_{12} \cdot x / A_{11} \\ x \end{bmatrix} = 0 \Rightarrow A$ not full rank
contradiction

Simpler in square case

$$0 \neq \det(A) = \pm\det(PA) = \pm \det(1^{st}\text{ factor}) \cdot \det(2^{nd}\text{ factor})$$

$$= \pm \cdot 1 \cdot A_{11} \cdot \det(S)$$

$$\Longrightarrow \det(S) \neq 0$$

Apply induction to get $S = P' \cdot L' \cdot U'$

$$PA = \left[\begin{array}{c|c} 1 & 0 \\ \hline \frac{A_{21}}{A_{11}} & I \end{array}\right] \cdot \left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline 0 & \underbrace{P' \cdot L' \cdot U'}_{S} \end{array}\right]$$

$$= \left[\begin{array}{c|c} 1 & 0 \\ \hline \frac{A_{21}}{A_{11}} & P'L' \end{array}\right] \cdot \left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline 0 & U' \end{array}\right]$$

$$= \left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & P' \end{array}\right] \cdot \left[\begin{array}{c|c} 1 & 0 \\ \hline P'^{T}\frac{A_{21}}{A_{11}} & L' \end{array}\right] \left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline 0 & U' \end{array}\right]$$

$$A = P^{T} \underbrace{\left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & P' \end{array}\right]}_{\text{perm}} \cdot \underbrace{\left[\begin{array}{c|c} 1 & 0 \\ \hline P'^{T}\frac{A_{21}}{A_{11}} & L' \end{array}\right]}_{\text{unit lower triang}} \cdot \underbrace{\left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline 0 & U' \end{array}\right]}_{\text{upper triang}}$$

QED