Welcome to Mu 221! Lecture 9, Sep 15

Goal: Understand real cost of algs.
Moving data, not flops, most expensive

Simple Model of Comm. Costs
Bandwidth (bw), Latency

Intuition: Freeway from Berkeley to Sacramento

\[ BW = \text{# cars/hour that can go from B \rightarrow S} \]

\[ \text{# cars/hour} = \text{density (# cars/mile/ lane)} \times \text{velocity (miles/hr)} \times \text{lanes} \]

Latency = how long it takes 1 car to
go from B \rightarrow S

\[ \text{time (hrs)} = \frac{\text{distance (miles)}}{\text{velocity (miles/hour)}} \]

So minimum time to move n cars from B \rightarrow S.
assuming they all travel in one “convoy” close together

time (hrs) = time for 1st car + time for remaining cars
= latency + \frac{n}{BW}

Same formula for moving data:
w words from DRAM to cache
= latency + \frac{w}{BW}
assuming all words in one “message”

Moving w words in m messages costs
\[ m \cdot (\text{latency} + \frac{w}{BW}) \]

Notation \[ m \cdot \alpha + w \cdot \beta = \text{comm cost} \]
\[ \alpha = \text{latency} \]
\[ \beta = \frac{1}{BW} \]
\[ \gamma = \text{time per flop} \]
\[ f = \# \text{flops} \]

Total time:
\[ fg + m \cdot \alpha + w \cdot \beta \]

Today \[ f \ll \beta \ll \alpha \] (picture!)

Flops dominate: \[ fg \geq m \alpha + w \cdot \beta \]
Comm dominates: \[ fg < m \alpha + w \cdot \beta \]

Notation: Computational Intensity
\[ g = \frac{f}{w} = \text{“flops per word moved”} \]
$g$ needs to be large to run fast:
\[ f - g \geq w^B \Rightarrow g = \frac{f}{w} \geq \frac{B}{g} \gg 1 \]

History of how this has influenced alg.

**BLAS-1** Basic Linear Algebra Subroutines

- Standard Library of 15 ops mostly on vectors
  - (1) $y = \alpha \cdot x + y$ "array" $x,y$ vectors
  - $\alpha$ scalar
  - inner loop of Gauss Elim
  - (2) dot product
  - (3) $\|x\|_2 = \sqrt{\sum \|x_i\|^2}$
  - (4) find largest entry $|x_i|$ in $x$

**Motivation:** easier programming, readable, avoid over/underflow, efficiency

Poor comp intensity: $q = \frac{2n}{2n} = 1$ (dot prod)

**BLAS-2** (mid 1980s)

- standard library of 25 ops on Matrix-vector pairs
  - (i) $y = \alpha y + \beta \cdot A \cdot x$ $A$ matrix, $x,y$ vecs
  - $\alpha, \beta$ scalars
  - "GEMV"
  - lots of variations, $A = A^T$, triangular...
(2) $A = A + \alpha \cdot x \cdot y^T$ rank-one update
   "GER"

(3) Solve $T x = b$, $T$ triangular "TRSV"
   Motivation: similar to BLAS-1
   + opportunities to optimize on vector computers

Not much improvement on $g = \frac{\alpha}{w} = \frac{2n^2}{n^2} = 2$
   for GEMV $n \times n$

BLAS-3 library (late 1980s)
   9 operations on pairs of matrices
   (1) $C = \alpha C + B \cdot A \cdot B$ $A, B, C$ matrices
      "GEMM" $\alpha, B$ scalars
   (2) $C = \alpha C + \beta \cdot A \cdot A^T$ $C = C^T$, $A$ rectangular
      "SYRK"
   (3) Solve $T X = B$, $T$ triangular, B matrix

$g$ for GEMM $= \frac{\alpha}{w} = \frac{2n^3}{4n^2} = \frac{n}{2}$
   But usual 3 nested loops as slow as GEMV

BLAS 3 led community to rewrite all
dense linear algebra using BLAS
resulted in LAPACK, Scalapack,...
Goal: Prove comm lower bound for
\( nxn \) matmul for
\[
\begin{array}{c}
\text{CPU} \\
\text{Cache} \\
\text{main mem}
\end{array}
\]
\( \text{size}=M \)

minimize this data movement

Easy case: all 3 matrices fit in cache: \( 3n^2 \leq M \)
- read \( A, B, C \) into cache
- do all arithmetic
- write \( C \) back to main mem
- # words moved \( \geq 4n^2 \) obvious lower bound

Hard Case: \( 3n^2 > M \)

Thm (Hong, Kung 1981): To multiply
\[
C = C + A \cdot B \quad \text{(or } A \cdot B) \text{ using } 2n^3 \text{ flops}
\]
in correct order, # words moved = \( \Omega \left( \frac{n^3}{UM} \right) \)

- Extends to rectangular, sparse matrices
  \( \Omega \left( \frac{\#\text{flops}}{UM} \right) \)

Proof Sketch:
1) Suppose we fill cache with \( M \) words
2) do as many flops as possible
3) store results back to main mem

Upper bound \# flops we can do in 2) by \( G \)
\[ \Rightarrow \text{Doing } G \text{ flops costs } 2M \text{ words moved} \]
\[ \Rightarrow \text{Since we do } 2n^3, \text{ need to repeat } \frac{2n^3}{G} \text{ times} \]
\[ \Rightarrow \# \text{ words moved} = \frac{2n^3}{G} \cdot 2M \]
Need G, or an upper bound

use geometric model: represent alg as 3D cube (lattice)

\[
\begin{align*}
\text{for } i &= 1:n \\
\text{for } j &= 1:n \\
\text{for } k &= 1:n \\
C(i,j) &= A(i,j) \cdot B(i,j)
\end{align*}
\]

Goal: bound
\|W\| in terms
\|V_A\| = \#A entries
\|V_B\| = \#B entries
\|V_C\| = \#C entries

assumption: \|V_A\| + \|V_B\| + \|V_C\| \leq M

Intuition:
\[
\begin{align*}
\|V\| &= x \cdot y \cdot z \\
\|V_A\| &= x \cdot y \\
\|V_B\| &= y \cdot z \\
\|V_C\| &= x \cdot z
\end{align*}
\]

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

Thm (Loomis & Whitney, 1949)
\[\|V\| \leq \sqrt{|V_A| \cdot |V_B| \cdot |V_C|}\] for any V

\[G = \|V\| \leq \sqrt{\frac{M^2}{M^2}} \cdot M = M^{3/2}\]
\[ \text{\# words moved} \geq \frac{2n^3 \cdot 2M}{G} = \frac{4n^3 M}{M^{3/2}} \leq \Omega\left(\frac{n^3}{\sqrt{M}}\right) \]

If careful with constants:
\[ \text{\# words moved} \geq \frac{2n^3}{\sqrt{M}} - 2M \]

attainable!