Communication [Lower] Bounds for Heterogeneous Architectures

Julian Bui

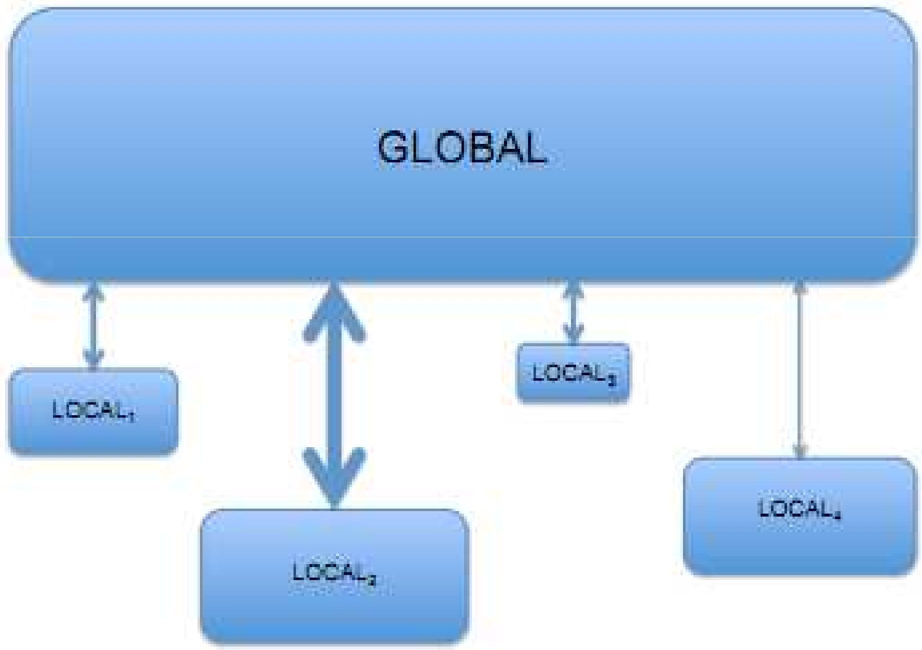
Outline

- Problem
- Goal
- MV multiplication w/i constant factor of the lower bound
- Matrix-matrix multiplication w/i constant factor of the lower bound
  - With Examples!
- If there’s time: derivation of proofs of lower bounds for N^2 and N^3 problems

Problem

- N^2 problems like MV multiply
  - Dominated by the communication to read inputs and write outputs to main memory
- N^3 problems like matrix-matrix multiply
  - Dominated by communication between global and local memory (Loomis-Whitney)

Architecture



Goal

- Let’s create a model based on the communication scheme to minimize the overall time
- Execution time becomes a function of processing speed, bandwidth, message latency, and memory size.

Optimization Function

Time won't be faster than the fastest heterogeneous element (could be a GPU, for example)

Num. FLOPS times computation speed

Comm. cost of reading inputs and writing outputs

Num. bytes & msgs passed between global and shared memory

$$T \geq \min_{\sum F_i = G} \max_{1 \leq i \leq P} \gamma_i F_i + \beta_i \max\left\{ I_i + O_i, \frac{F_i}{8\sqrt{M_i}} \right\} + \alpha_i \max\left\{ \frac{I_i + O_i}{M_i}, \frac{F_i}{8 M_i^{3/2}} \right\}.$$

Inv. Bandwidth

Latency to

shared memory

Overall execution time

The slowest processor is the weakest link

Heterogeneous MV multiply

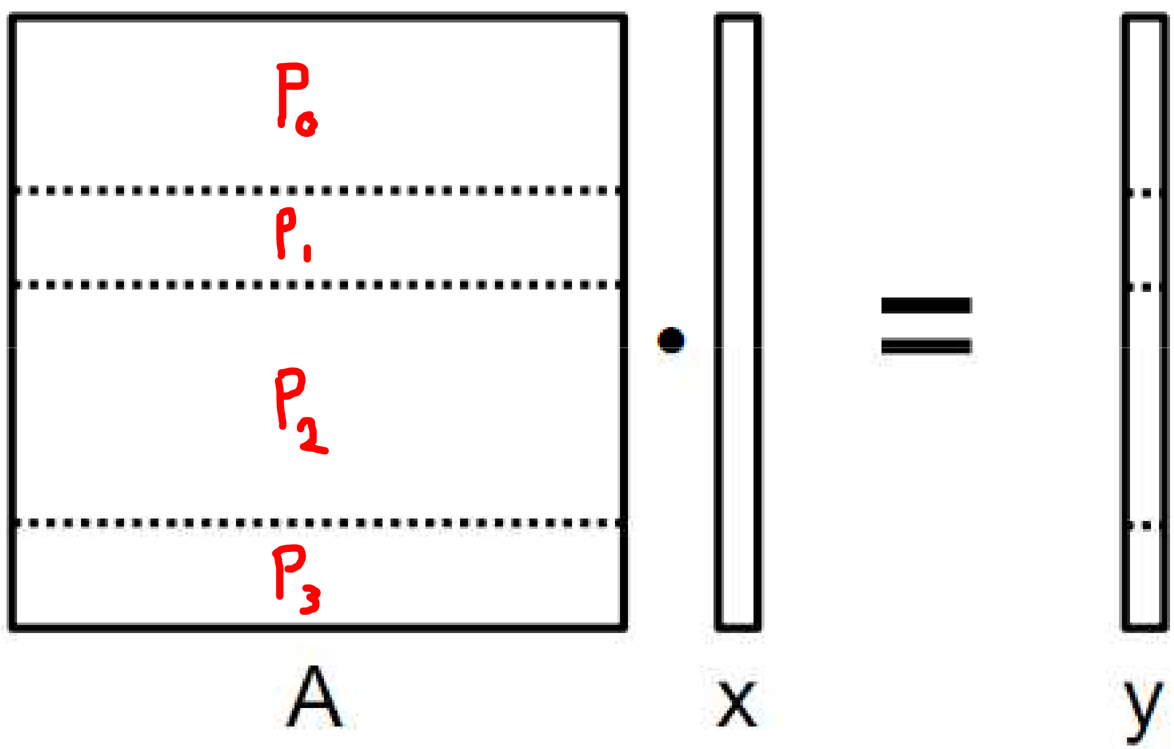


Figure 3: HGEMV splitting

Heterogeneous, recursive MM

Algorithm 2 Heterogeneous matrix-matrix multiplication

Require: Matrices $A, B \in \mathbb{R}^{n \times n}$, stored in block-recursive order, n is a power of two

Measure $\alpha_i, \beta_j, \gamma_i, M_i$ and set δ_i according to equation (9) for each $1 \leq i \leq P$
for $i = 1$ to P **do**
  Set F_i according to equation (10) where $G = n^3$
  Set k_i to be the largest integer such that $3(n/2^{k_i})^2 \geq M_i$
  Convert F_i/G into octal and round[1] to k_i^{th} digit: $0.d_1^{(i)} d_2^{(i)} \cdots d_{k_i}^{(i)}$
end for
Initialize $S = \{A \cdot B\}$
for $j = 1$ to $\max k_i$ **do**
  Subdivide all problems in S into 8 subproblems according to square recursive GEMM
  Assign $d_j^{(i)}$ subproblems to proc_i and remove subproblems from S
end for
for all proc_i **parallel do**
  Compute assigned subproblems using square recursive GEMM
end for

Ensure: Matrix $C = AB$, stored in block-recursive order

Solve the optimization equation for each processor, then assign it an appropriate amount of work based in units of work that are multiples of 8 sub MM multiplication blocks

Recursively solve each sub-block using a divide and conquer method

Recursive, Parallel MM

x

$$
\begin{array}{|c|c|}
\hline
C & D \\
\hline
E & F \\
\hline
\end{array}
\times
\begin{array}{|c|c|}
\hline
G & H \\
\hline
I & J \\
\hline
\end{array}
=
$$

$$
\begin{array}{|c|c|}
\hline
CG & CH \\
\hline
EG & EH \\
\hline
\end{array}
+
\begin{array}{|c|c|}
\hline
DI & DJ \\
\hline
FI & FJ \\
\hline
\end{array}
$$

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

x

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

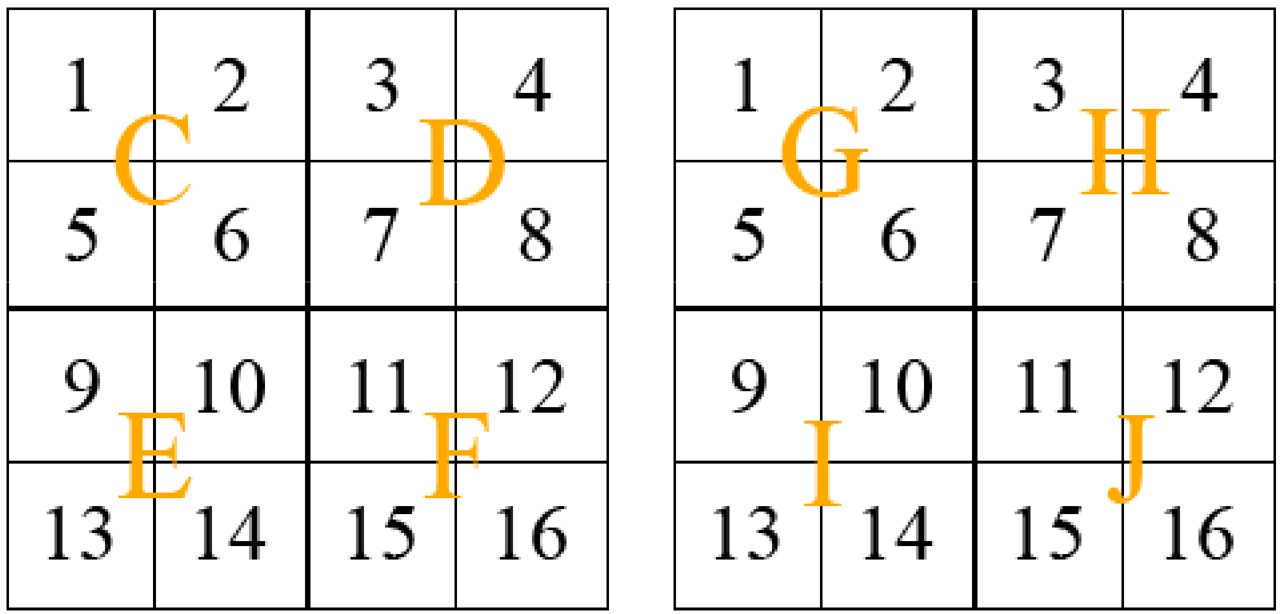
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

x

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

C	D
E	F

G	H
I	J


1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
C D E F
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
G H I J

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

x

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

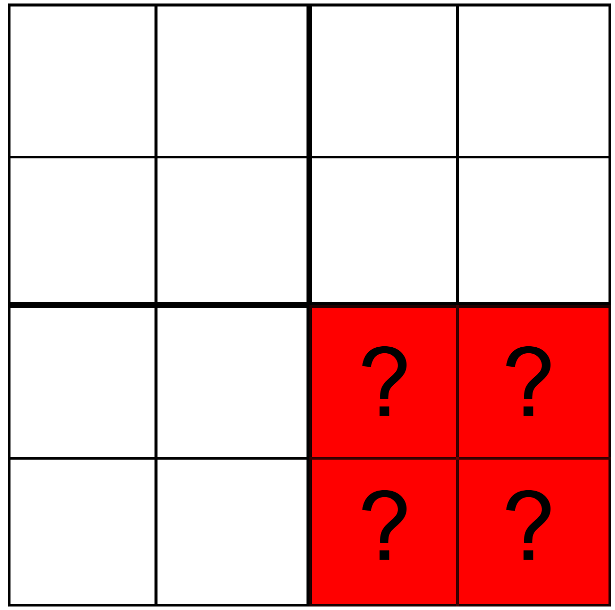
=

		?	?
		?	?

Review

$$\begin{bmatrix} C & D \\ E & F \end{bmatrix} \times \begin{bmatrix} G & H \\ I & J \end{bmatrix} =$$

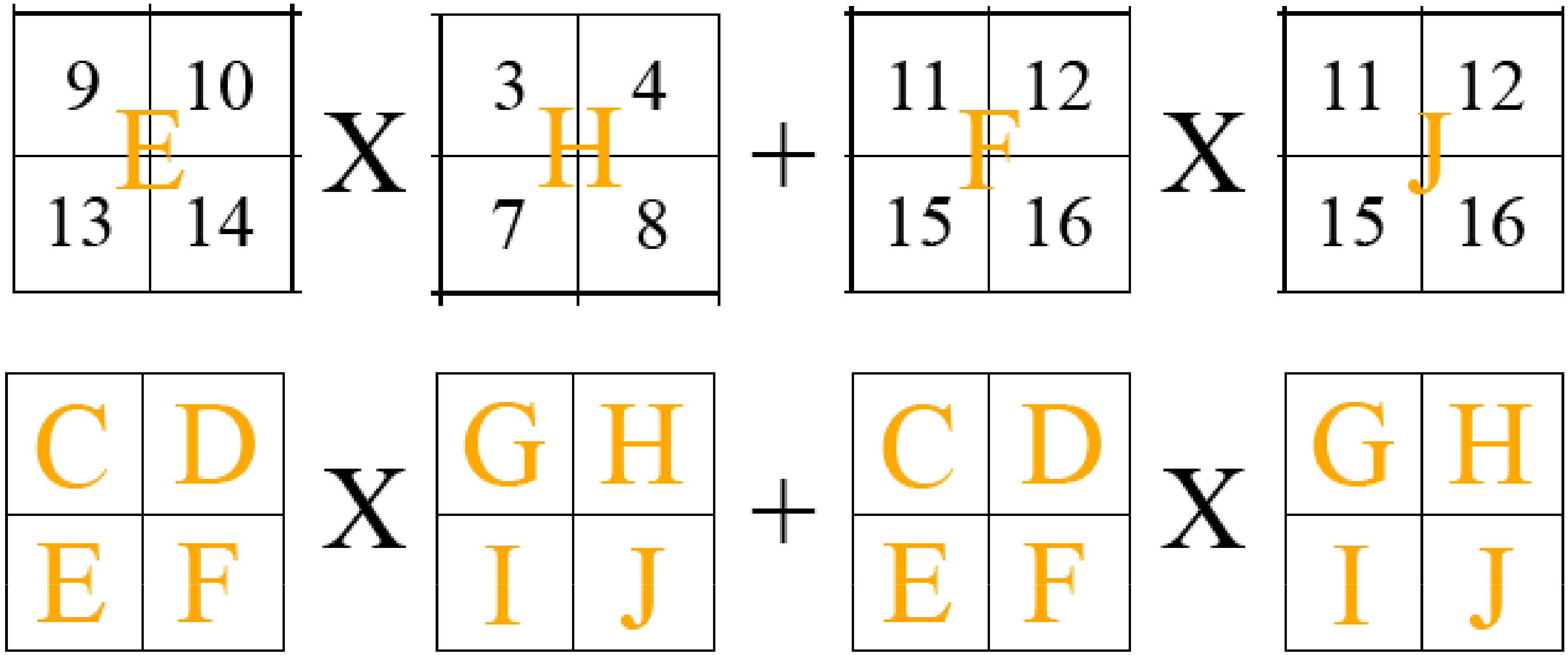
$$\begin{bmatrix} CG & CH \\ EG & EH \end{bmatrix} + \begin{bmatrix} DI & DJ \\ FI & FJ \end{bmatrix}$$



$$\begin{bmatrix} C & D \\ E & F \end{bmatrix} \times \begin{bmatrix} G & H \\ I & J \end{bmatrix} = \begin{bmatrix} CG & CH \\ EG & \boxed{EH} \end{bmatrix} + \begin{bmatrix} DI & DJ \\ FI & \boxed{FJ} \end{bmatrix}$$

So we need EH + FJ

$$\underset{E}{\begin{bmatrix} 9 & 10 \\ 13 & 14 \end{bmatrix}} \; X \; \underset{H}{\begin{bmatrix} 3 & 4 \\ 7 & 8 \end{bmatrix}} + \underset{F}{\begin{bmatrix} 11 & 12 \\ 15 & 16 \end{bmatrix}} \; X \; \underset{J}{\begin{bmatrix} 11 & 12 \\ 15 & 16 \end{bmatrix}}$$



CG = 9 * 3 = 27	DI = 10 * 7 = 70	CG = 11 * 11 = 121	DI = 12 * 15 = 180
CH = 9 * 4 = 36	DJ = 10 * 8 = 80	CH = 11 * 12 = 132	DJ = 12 * 16 = 192
EG = 13 * 3 = 39	FI = 14 * 7 = 98	EG = 15 * 11 = 165	FI = 16 * 15 = 240
EH = 13 * 4 = 52	FJ = 14 * 8 = 112	EH = 15 * 12 = 180	FJ = 16 * 16 = 256

CG = 9 * 3 = 27	DI = 10 * 7 = 70	CG = 11 * 11 = 121	DI = 12 * 15 = 180
CH = 9 * 4 = 36	DJ = 10 * 8 = 80	CH = 11 * 12 = 132	DJ = 12 * 16 = 192
EG = 13 * 3 = 39	FI = 14 * 7 = 98	EG = 15 * 11 = 165	FI = 16 * 15 = 240
EH = 13 * 4 = 52	FJ = 14 * 8 = 112	EH = 15 * 12 = 180	EH = 15 * 12 = 180

CG = 9 * 3 = 27

CH = 9 * 4 = 36

EG = 13 * 3 = 39

EH = 13 * 4 = 52

DI = 10 * 7 = 70

DJ = 10 * 8 = 80

FI = 14 * 7 = 98

FJ = 14 * 8 = 112

CG = 11 * 11 = 121

CH = 11 * 12 = 132

EG = 15 * 11 = 165

EH = 15 * 12 = 180

DI = 12 * 15 = 180

DJ = 12 * 16 = 192

FI = 16 * 15 = 240

EH = 15 * 12 = 180

$$\begin{bmatrix} C & D \\ E & F \end{bmatrix} \times \begin{bmatrix} G & H \\ I & J \end{bmatrix} + \begin{bmatrix} C & D \\ E & F \end{bmatrix} \times \begin{bmatrix} G & H \\ I & J \end{bmatrix} =$$

CG = 9 * 3 = 27

CH = 9 * 4 = 36

EG = 13 * 3 = 39

EH = 13 * 4 = 52

DI = 10 * 7 = 70

DJ = 10 * 8 = 80

FI = 14 * 7 = 98

FJ = 14 * 8 = 112

CG = 11 * 11 = 121

CH = 11 * 12 = 132

EG = 15 * 11 = 165

EH = 15 * 12 = 180

DI = 12 * 15 = 180

DJ = 12 * 16 = 192

FI = 16 * 15 = 240

EH = 15 * 12 = 180

$$\begin{bmatrix} C & D \\ E & F \end{bmatrix} \mathrm{X} \begin{bmatrix} G & H \\ I & J \end{bmatrix} + \begin{bmatrix} C & D \\ E & F \end{bmatrix} \mathrm{X} \begin{bmatrix} G & H \\ I & J \end{bmatrix} =$$

$$\begin{bmatrix} CG & CH \\ EG & EH \end{bmatrix} + \begin{bmatrix} DI & DJ \\ FI & FJ \end{bmatrix} + \begin{bmatrix} CG & CH \\ EG & EH \end{bmatrix} + \begin{bmatrix} DI & DJ \\ FI & FJ \end{bmatrix}$$

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

x

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

=

		398	440
		542	600

Derivation of Proofs

- For a MM, the a node can only do
  O(N * $\sqrt{N}$) useful arithmetic operations per phase
  - If a single processor accesses rows of matrix A and some of these rows have at least $\sqrt{N_A}$ elements, then the processor will touch at most $\sqrt{N_A}$ rows of matrix A.
  - Since each row of C is a product of a row in A and all of B, the number of useful multiplications on a single processor that involve rows of matrix A are bounded by O($N_B\sqrt{N_A}$) or O($N\sqrt{N}$)

Derivation Cont'd

- Number of total phases = the total amt. of work per processor divided by the amt. of work a processor can do per phase
  O($\frac{G}{M * \sqrt{M}}$)
- Total amt. of communication = # phases times memory used per phase (~M)
  $\frac{G}{\sqrt{M}}$

Derivation Cont'd

- Total # Words, W = $\frac{G}{\sqrt{M}}$
- G / sqrt(M) = W = G * M / (M * sqrt(M))
- W >= ((G / (M * sqrt(M)) – 1) * M
- W >= (G / sqrt(M)) - M