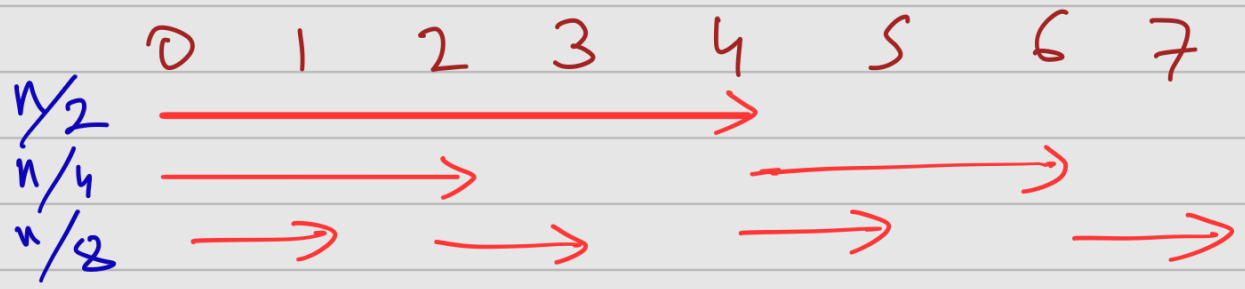
Lec 14

- MPI Blocking Collectives

Barrier, Reduce, Gather, Scatter,
Bcast, Allgather, Allreduce, All to all

→ **Scatter** Recursive Halving
Vector Halving
Distance Halving
(finally each process needs n/p)

$$\log p \times L + (p-1) \times \left(\frac{n}{p}\right) \times \left(\frac{1}{B}\right)$$



→ **All Gather**

① Ring

- every comm b/w direct neighbour
- first cycle: left nei → right nei apna data
  second cycle: left nei → right nei ko

n Bytes p processes ⇒ $\frac{n}{p}$ bytes

send/recv every time

$$(p-1)\left(L + \frac{n}{p} \times \frac{1}{B}\right)$$

(2) Recursive Doubling

$\left(2^{k-1}\right) \times \frac{n}{p}$ Bytes at k^{th} step.

(Data size, dist double every step)

0 ↔ 1 2 ↔ 3 4 ↔ 5 6 ↔ 7
0 ↔ 2 1 ↔ 3 4 ↔ 6 5 ↔ 7
0 ↔ 4 1 ↔ 5 2 ↔ 6 3 ↔ 7

$$\log p \times L + (p-1)\left(\frac{n}{p}\right) \times \frac{1}{B}$$

→ Allgather Short msg = 80KB
Long msg = 512 KB

S < 80KB M < 512KB

	Power of 2	Not Power of 2
S	Rec Doub	Bruck
M	Rec Doub	Ring
L	Ring	Ring

→ **Bcast — Van de Geijn**

Binomial : $\log p \times \left(L + \frac{n}{B}\right)$

Scatter + Allgather

↳ $\log p\, L + \frac{(p-1)}{p} \frac{n}{B}$

Allg (ring) : $(p-1)\,L + (p-1)\frac{n}{pB}$

Allg (rec D) $\log p\, L + (p-1)\frac{n}{pB}$

Bcast_Short_Msg_Size (X)
Bcast_Long_Msg_Size (Y)

$S < X$ — binomial
M — scatter + allg (rec dou)
$L > Y$ — scatter + allg (ring)

→ **Reduce**

Rec Doub ⇒ $\log p \left(L + \frac{n}{B} + n \times C\right)$

(compute cost per Byte)

- each process has 'n' Bytes data

(2) Raben Seifner

Recursive Vector Halving & Dist Doubling

then Gather

$$\log p \times L + \frac{(p-1)}{p}\frac{n}{B} + \frac{(p-1)}{p} \times n \times c \quad \text{(reduce + scatter)}$$

$$+ \log p \times L + \frac{(p-1)}{p} \times \frac{n}{B} \quad \text{(gather using binomial)}$$

→ at each step processes P, & comm. reduce first half of data at P and second half of data at Q.

at k^{th} step, each process has $\frac{n}{2^k}$ data

after $\log p$ steps, gather at root.

0 ↔ 1	2 ↔ 3	4 ↔ 5	6 ↔ 7
0 ↔ 2	1 ↔ 3	4 ↔ 6	5 ↔ 7
0 ↔ 4	1 ↔ 5	2 ↔ 6	3 ↔ 7

(like all gather)

→ All reduce (Rabenseifner)

reduce - scatter using recursive vector halving and dist doubling

then

all gather using vector doubling, dist

halving

$$\log p \times L + \frac{p-1}{p} \times \frac{n}{B} + \frac{p-1}{p} \times n \times c$$

$$+ \log p \times L + \frac{p-1}{p} \times \frac{n}{B}$$

- Short / Medium :

Reduce (Rec Doub) + Bcast (Binomial)

$$\log p \times L + \log p \times \frac{n}{B} + \log p \times n \times c +$$

$$\log p \times L + \log p \frac{n}{B}$$

Long

Reduce Scatter + allgather (rec doub)

$$\log p \times L + \frac{p-1}{p} \times \frac{n}{B} + \frac{p-1}{p} \times n \times c$$

$$+ \log p \times L + (p-1) \frac{n}{p} \times \frac{1}{B}$$

Lec 15

- col-wise dgroup ⇒ cols are cutters

- row-wise decomp ⇒ rows are cutters

decomp along x axis ⇒ col wise decomp
" " y axis ⇒ row wise "

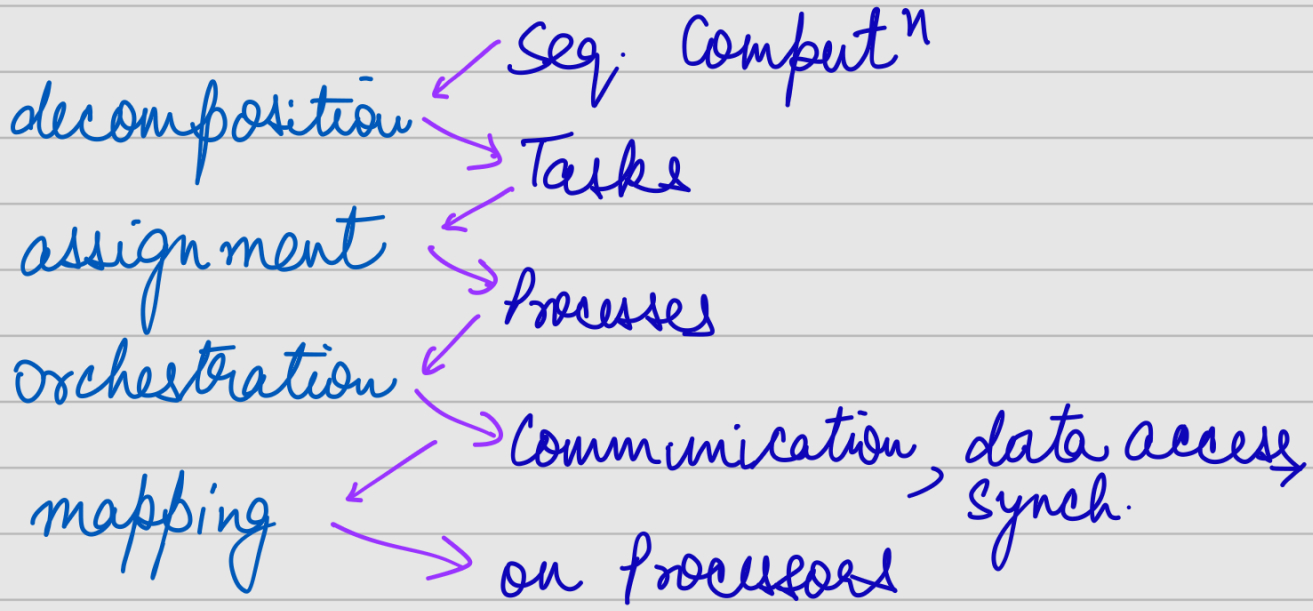
Gatherv — recvcounts[i], displ[i]

Scatterv — sendcount[i], displ[i]

Allgatherv — recvct[i], displ[i]

Alltoallv — sendct[i], displs[i]
recvct[i], displr[i]

Lec 16



→ **Performance Goals**

i) Expose concurrency

ii) Reduce interprocess comm.

iii) Load-balance

iv) Reduce synchronisation

v) Reduce idling

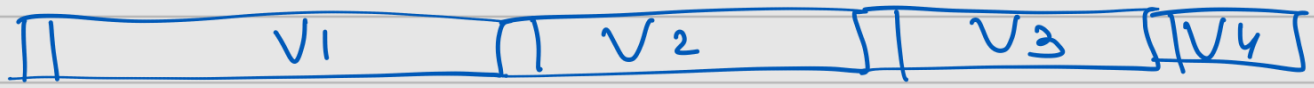
vi) Reduce management overhead

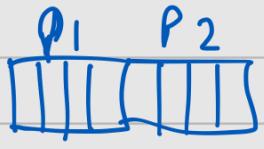
vii) Preserve data locality

viii) Exploit network topology

Lec 17

M x N → 4 vars each, P x Q processes

(1) 

(2) 

. . . . .

MPI_File_open (CommWorld, filename, create+rdwr, Null, fh)

for (i = 0 to 4) {

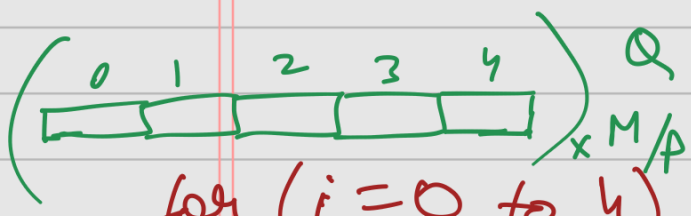
MPI_Offset

offset[i] = i x M x N x sizeof(float) + (((rank / Q) x Q + (rank % Q)) x M x N x sizeof(float))

P x Q

MPI_File_Seek (fh, offset[i], ———)

MPI_File_write (fh, buf[i], ———) }



for (i = 0 to 4) { for (idx = 0 to $\frac{M \times N}{P \times Q}$) {

off [i] [idx] = $\left(\frac{rank}{Q}\right) \times \left(\frac{M}{P} \times N\right) \times sizeof(float)$

$+ \frac{(idx)}{N/Q} \times N \times sizeof(float)$

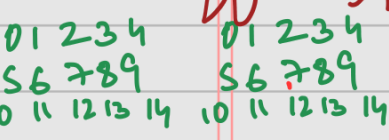
$+ \left((rank) \% Q\right) \times \left(\frac{N}{Q}\right) \times (float)$

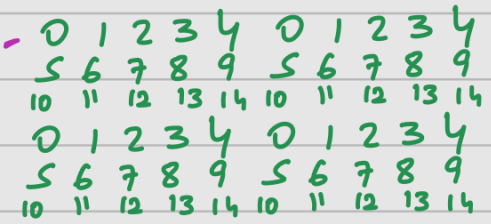
$+ \left((idx) \% Q\right) \times (float)$

$+ i \times (M \times N) \times (float)$

idx % $\left(\frac{N}{Q}\right)$

Fwrite (offset[i][idx], 1, float,);





off [4] [M/P]

off [0] [0] = $\left(\left(\frac{rank}{Q}\right) \times \frac{M}{P} \times N + \left(rank \% Q\right) \times \frac{N}{Q}\right)$

↓ +N x f

$\hookrightarrow + N \times M \times f$

Part 2 :

for $(id = 0$ to $\frac{M \times N}{P \times Q})$ {

offset = $\left[\left(\frac{rank}{Q} \times \frac{M \times N}{P} + (rank \,\%\, Q) \times \frac{N}{Q} \right) \right.$

$\left. + \left((id / (N/Q)) \times N + (id \,\%\, (\frac{N}{Q})) \right) \right]$

$4 \times sizeof(int) \times$

for $(i = 0$ to $4)$

Fwrite_at $(fh,$ offset + $i \times sizeof(int),$ var[id][i],$)$