

CSE 260

Lecture 15

Advanced Collective Communication

Announcements

- Project Progress report, Thu 2/27, up to 5% extra credit
- No Office Hours on Thursday, make an appointment for Friday

Today's lecture

- More Collectives
- Inside MPI
- Hypercubes and spanning trees

Collective communication

- Collective operations are called by **all** processes within a communicator
- Basic collectives seen so far
 - ♦ Broadcast: distribute data from a designated root process to all the others
 - ♦ Reduce: combine data from all processes returning the result to the root process
 - ♦ How do we implement them?
- Other Useful collectives
 - ♦ Scatter/gather
 - ♦ All to all
 - ♦ Allgather
- Diverse applications
 - ♦ Fast Fourier Transform
 - ♦ Sorting

Details of the algorithms

- Broadcast
- AllReduce
- Scatter/gather
- Allgather
- All to all

Underlying assumptions

- Fast interconnect structure
 - ♦ All nodes are equidistant
 - ♦ Single-ported, bidirectional links
- Communication time is $\alpha + \beta n$ in the absence of contention
 - ♦ Determined by bandwidth β^{-1} for long messages
 - ♦ Dominated by latency α for short messages

Inside MPI-CH

- Tree like algorithm to broadcast the message to blocks of processes, and a linear algorithm to broadcast the message within each block
- Block size may be configured at installation time
- If there is hardware support (e.g. Blue Gene), then it is given responsibility to carry out the broadcast
- Polyalgorithms apply different algorithms to different cases, i.e. long vs. short messages, different machine configurations
- We'll use hypercube algorithms to simplify the special cases when $P=2^k$, k an integer

Collective communication in MPI

- Collective operations are called by **all** processes within a communicator
- Broadcast: distribute data from a designated “root” process to all the others in the communicator
MPI_Bcast(in, count, type, root, comm)
- Reduce: combine data from all processes and return to a designated root process
MPI_Reduce(in, out, count, type, op, root, comm)
- Allreduce: all processes get reduction: **Reduce + Bcast**

Broadcast

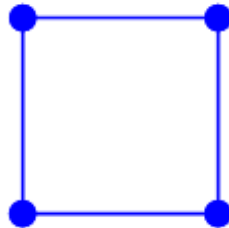
- The root process transmits of m pieces of data to all the $p-1$ other processors
- Spanning tree algorithms are often used
- We'll look at a similar algorithm with logarithmic running time: the *hypercube algorithm*
- With the linear ring algorithm this processor performs $p-1$ sends of length m
 - ♦ Cost is $(p-1)(\alpha + \beta m)$

Sidebar: what is a hypercube?

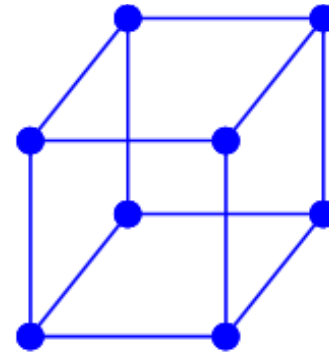
- A hypercube is a d -dimensional graph with 2^d nodes
- A 0-cube is a single node, 1-cube is a line connecting two points, 2-cube is a square, etc
- Each node has d neighbors



1D



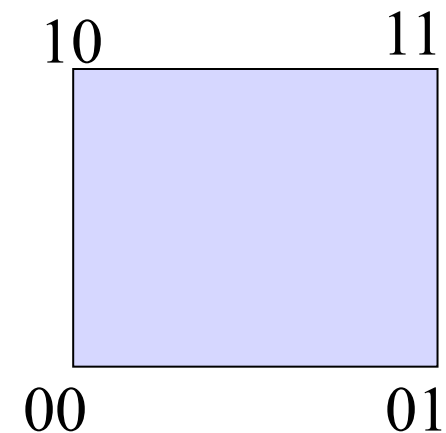
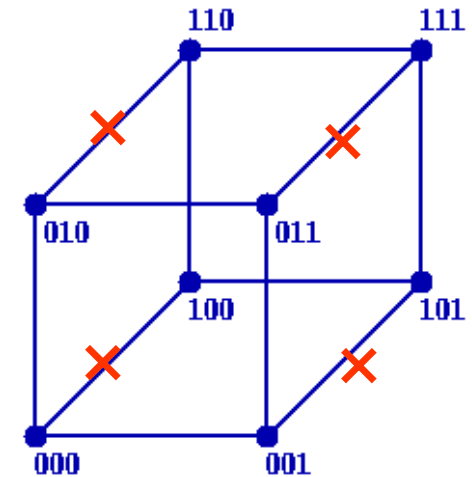
2D



3D

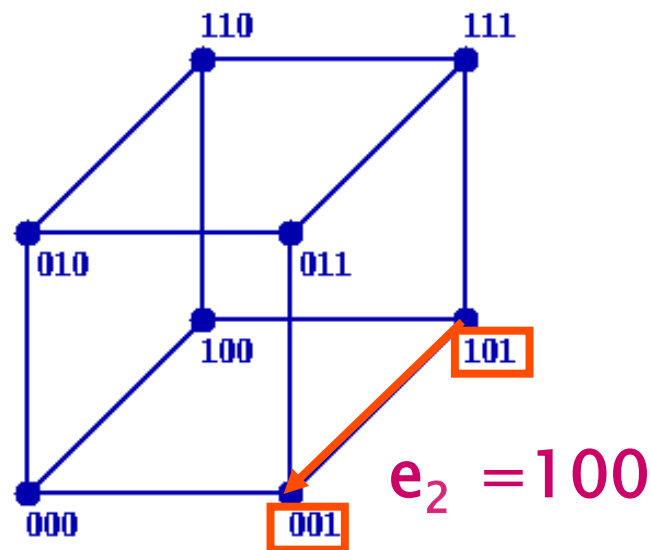
Properties of hypercubes

- A hypercube with p nodes has $\lg(p)$ dimensions
- *Inductive construction*: we may construct a d -cube from two $(d-1)$ dimensional cubes
- **Diameter**: What is the maximum distance between any 2 nodes?
- **Bisection bandwidth**: How many cut edges (mincut)



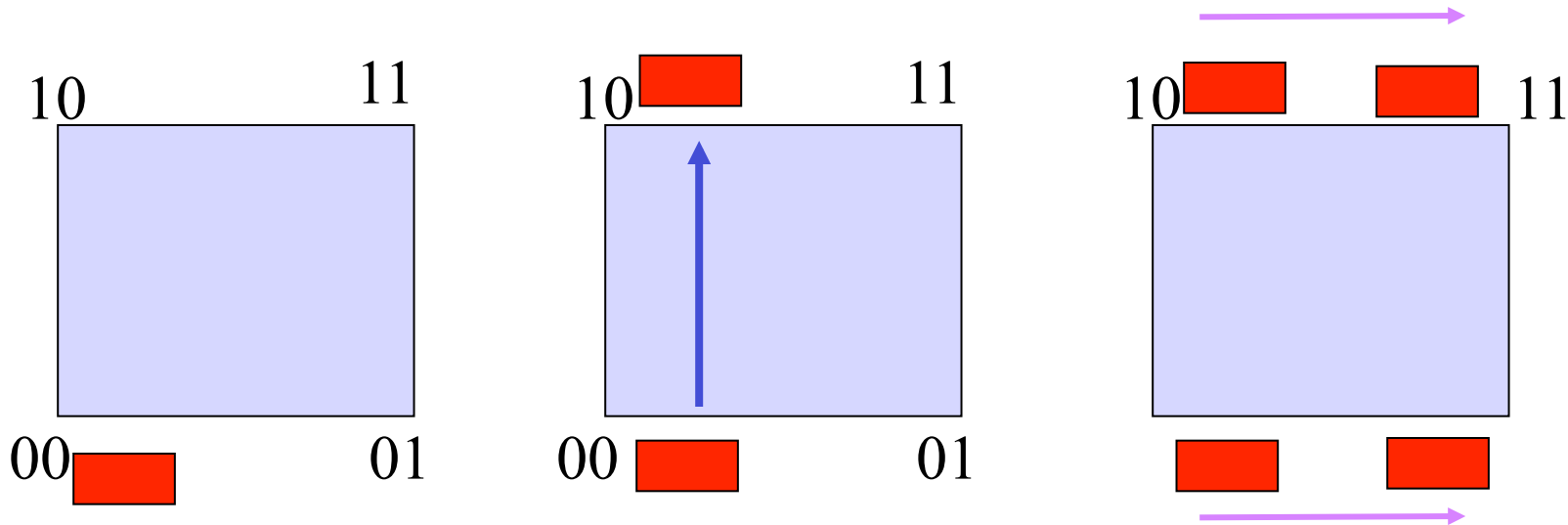
Bookkeeping

- Label nodes with a binary reflected grey code
<http://www.nist.gov/dads/HTML/graycode.html>
- Neighboring labels differ in exactly one bit position $001 = 101 \otimes e_2$, $e_2 = 100$



Hypercube broadcast algorithm with $p=4$

- Processor 0 is the root, sends its data to its hypercube “buddy” on processor 2 (10)
- Proc 0 & 2 send data to respective buddies



Reduction

- We may use the hypercube algorithm to perform reductions as well as broadcasts
- Another variant of reduction provides all processes with a copy of the reduced result

Allreduce()

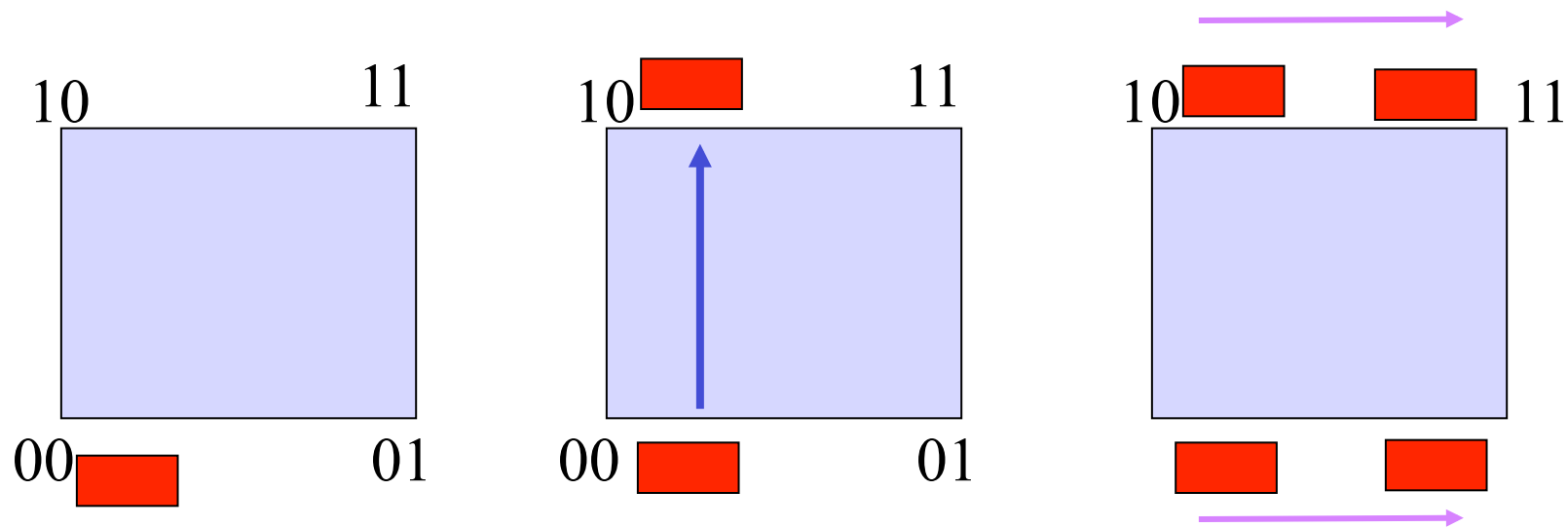
- Equivalent to a **Reduce** + **Bcast**
- A clever algorithm performs an **Allreduce** in one phase rather than having perform separate reduce and broadcast phases

Details of the algorithms

- Broadcast
- AllReduce
- Scatter/gather
- Allgather
- All to all

Allreduce

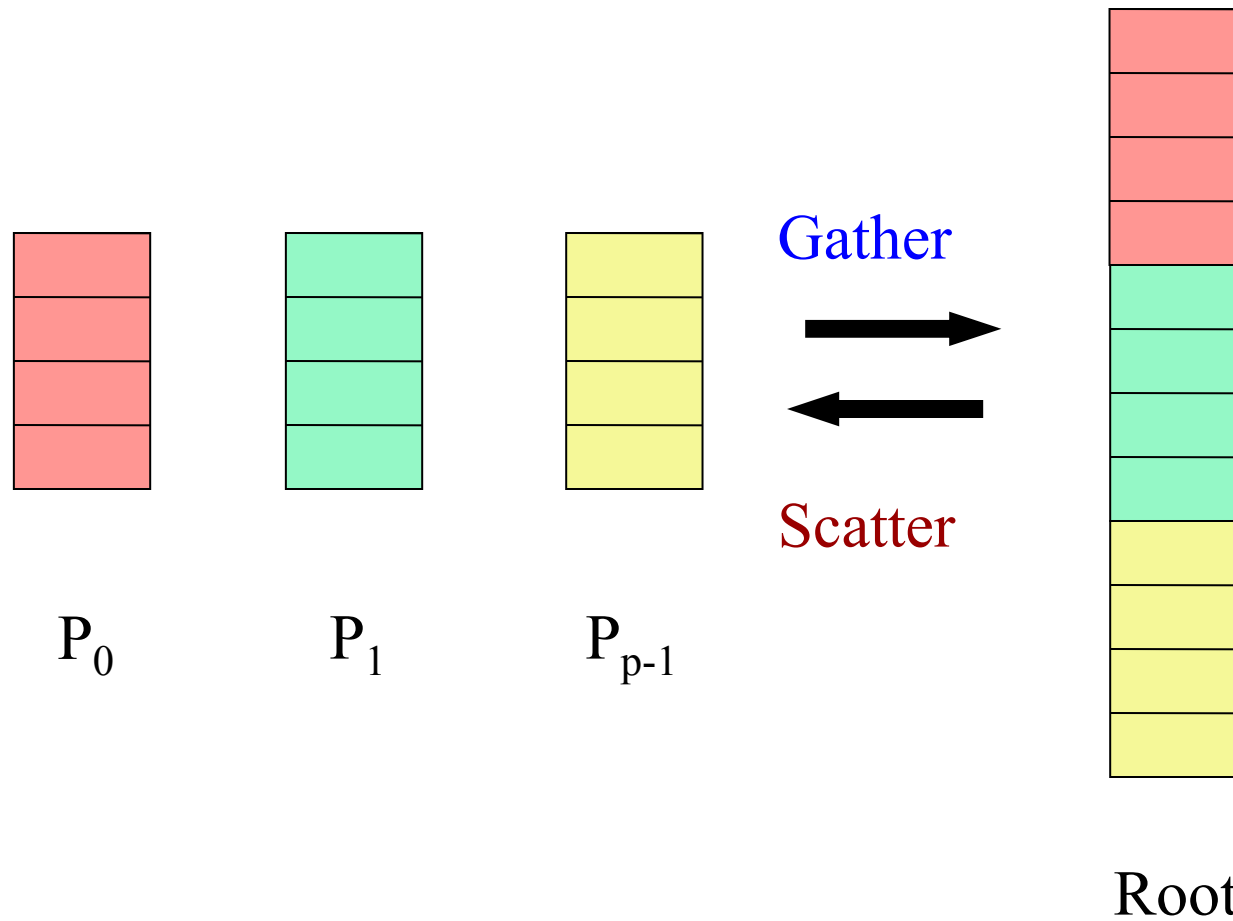
- Can take advantage of duplex connections



Details of the algorithms

- Broadcast
- AllReduce
- Scatter/gather
- Allgather
- All to all

Scatter/Gather



Scatter

- Simple linear algorithm
 - ♦ Root processor sends a chunk of data to all others
 - ♦ Reasonable for long messages

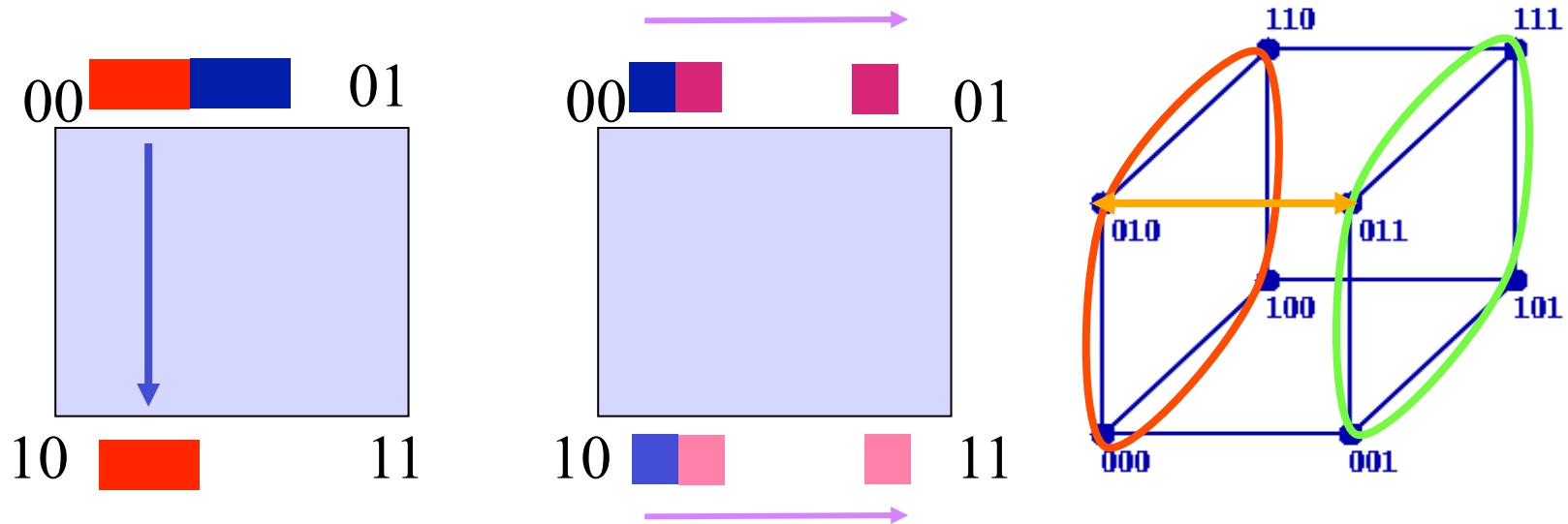
$$(p-1)\alpha + \frac{p-1}{p}n\beta$$

- Similar approach taken for Reduce and Gather
- For short messages, we need to reduce the complexity of the latency (α) term

Minimum spanning tree algorithm

- Recursive hypercube-like algorithm with $\lceil \log P \rceil$ steps
 - ♦ Root sends half its data to process $(\text{root} + p/2) \bmod p$
 - ♦ Each receiver acts as a root for corresponding half of the processes
 - ♦ MST: organize communication along edges of a minimum-spanning tree covering the nodes
- Requires $O(n/2)$ temp buffer space on intermediate nodes
- Running time:

$$\lceil \lg P \rceil \alpha + \frac{p-1}{p} n \beta$$

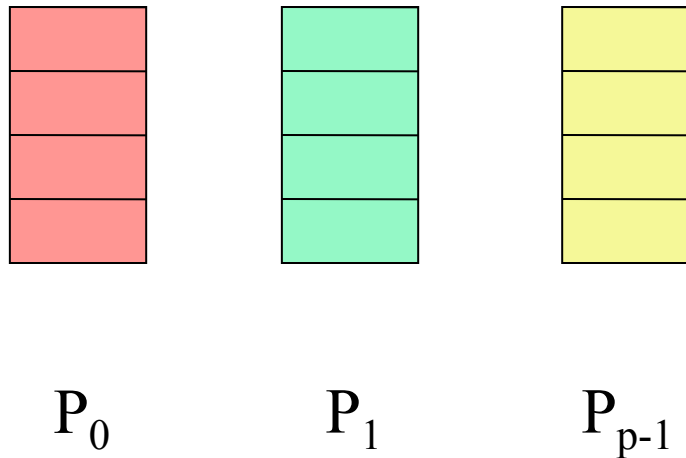


Details of the algorithms

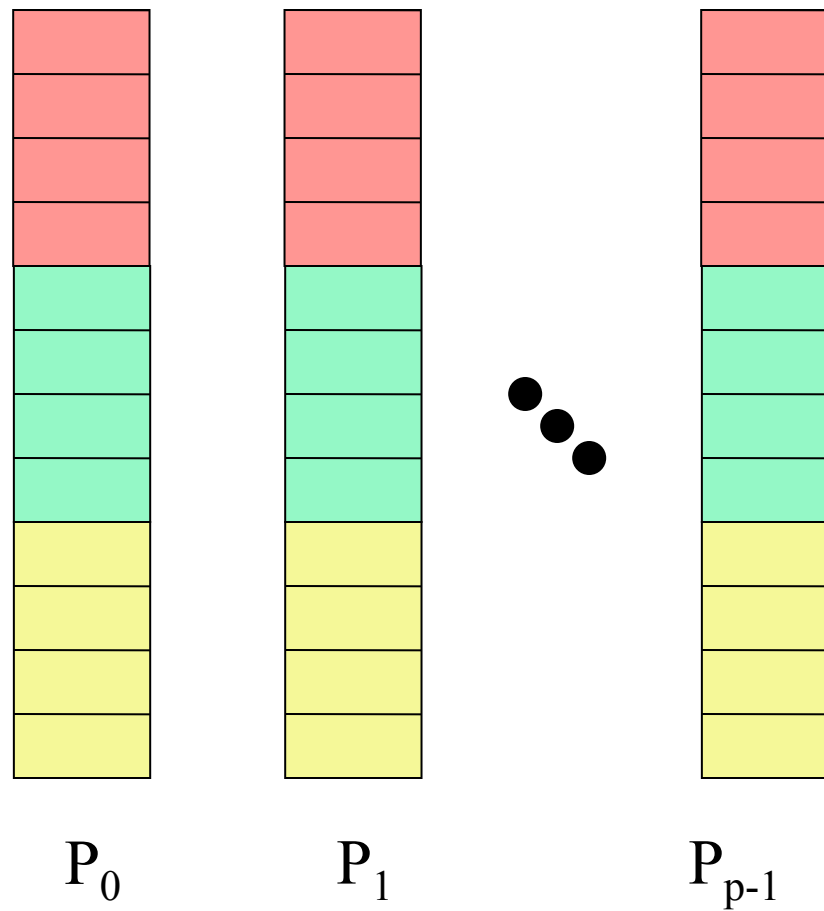
- Broadcast
- AllReduce
- Scatter/gather
- Allgather
- All to all

AllGather

- Equivalent to a gather followed by a broadcast
- All processors accumulate a chunk of data from all the others



AllGather



Allgather

- Use the all to all recursive doubling algorithm
- For P a power of two, running time is

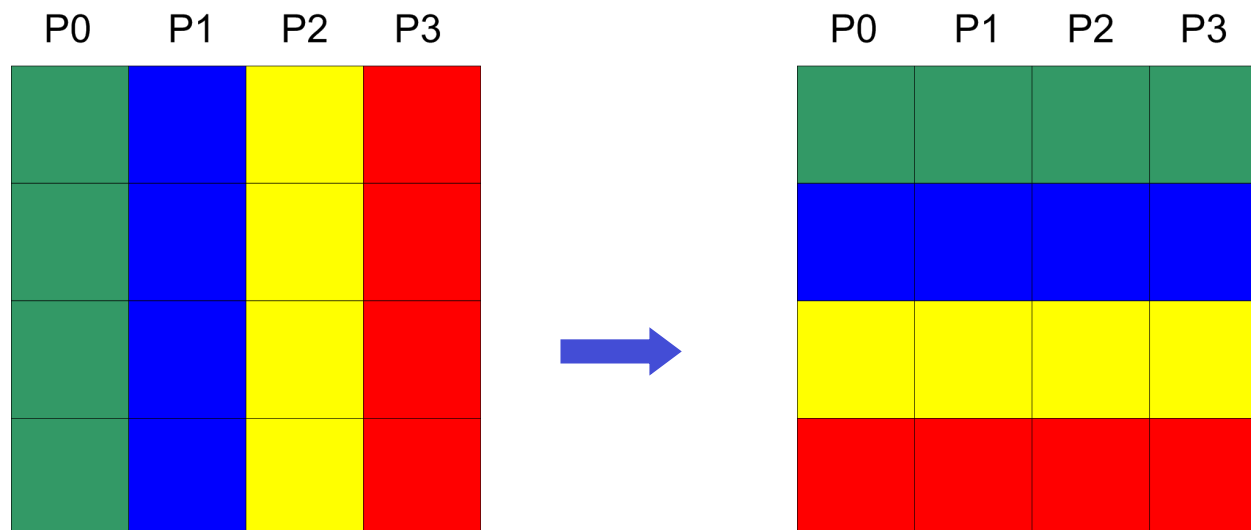
$$\lceil \lg P \rceil \alpha + \frac{p-1}{p} n \beta$$

Details of the algorithms

- Broadcast
- AllReduce
- Scatter/gather
- Allgather
- All to all

All to all

- Also called *total exchange* or *personalized communication*: a transpose
- Each process sends a different chunk of data to each of the other processes
- Used in sorting and the Fast Fourier Transform



Exchange algorithm

- n elements / processor (n total elements)
- $p - 1$ step algorithm
 - ♦ Each processor exchanges n/p elements with each of the others
 - ♦ In step i , process k exchanges with processes $k \pm i$

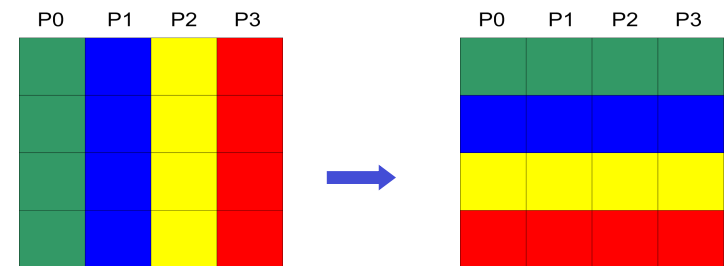
for $i = 1$ to $p-1$

src = $(\text{rank} - i + p) \bmod p$

dest = $(\text{rank} + i) \bmod p$

sendrecv(from src to dest)

end for



- Good algorithm for long messages
- Running time:

$$(p-1)\alpha + (p-1)\frac{n}{p}\beta \approx n\beta$$

Recursive doubling for short messages

- In each of $\lceil \log p \rceil$ phases all nodes exchange $\frac{1}{2}$ their accumulated data with the others
- Only $P/2$ messages are sent at any one time

$D = 1$

while ($D < p$)

 Exchange & accumulate data with rank $\otimes D$

 Left shift D by 1

end while

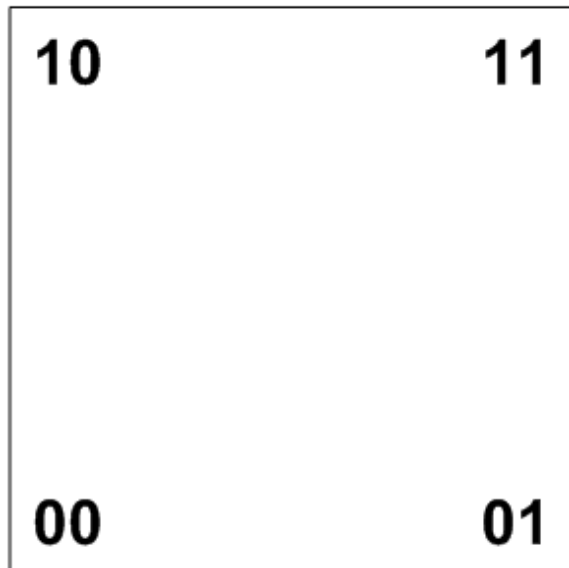
- Optimal running time for short messages

$$\lceil \lg P \rceil \alpha + nP\beta \approx \lceil \lg P \rceil \alpha$$

Flow of information

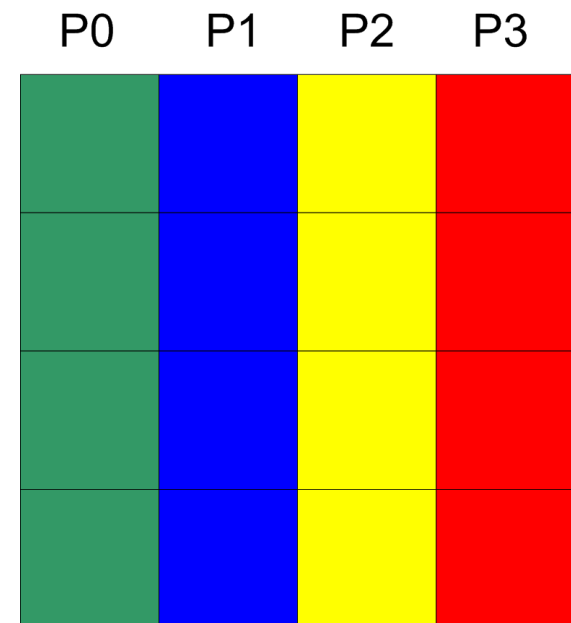
A B C D

A B C D



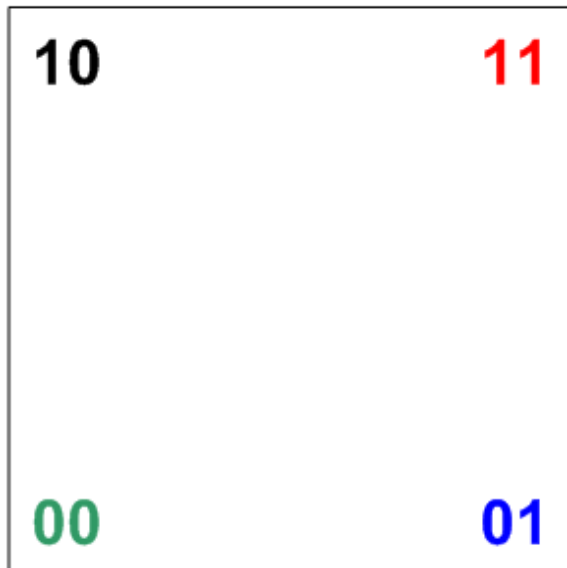
A B C D

A B C D



Flow of information

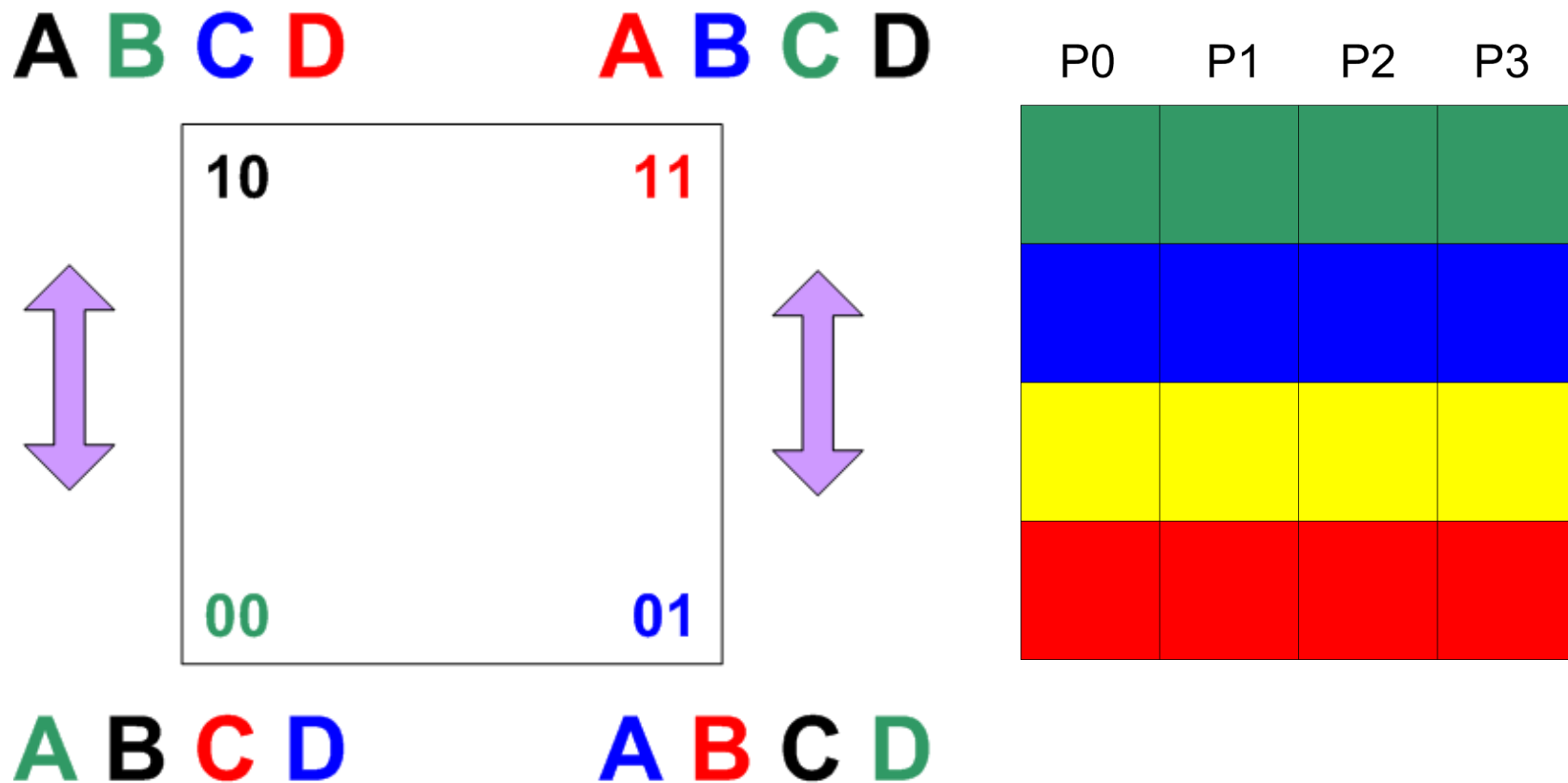
A B C D $\longleftrightarrow$ **A B C D**



A B C D $\longleftrightarrow$ **A B C D**

P0	P1	P2	P3

Flow of information



Summarizing all to all

- Short messages $\lceil \lg P \rceil \alpha$
- Long messages $\frac{P-1}{P} n \beta$

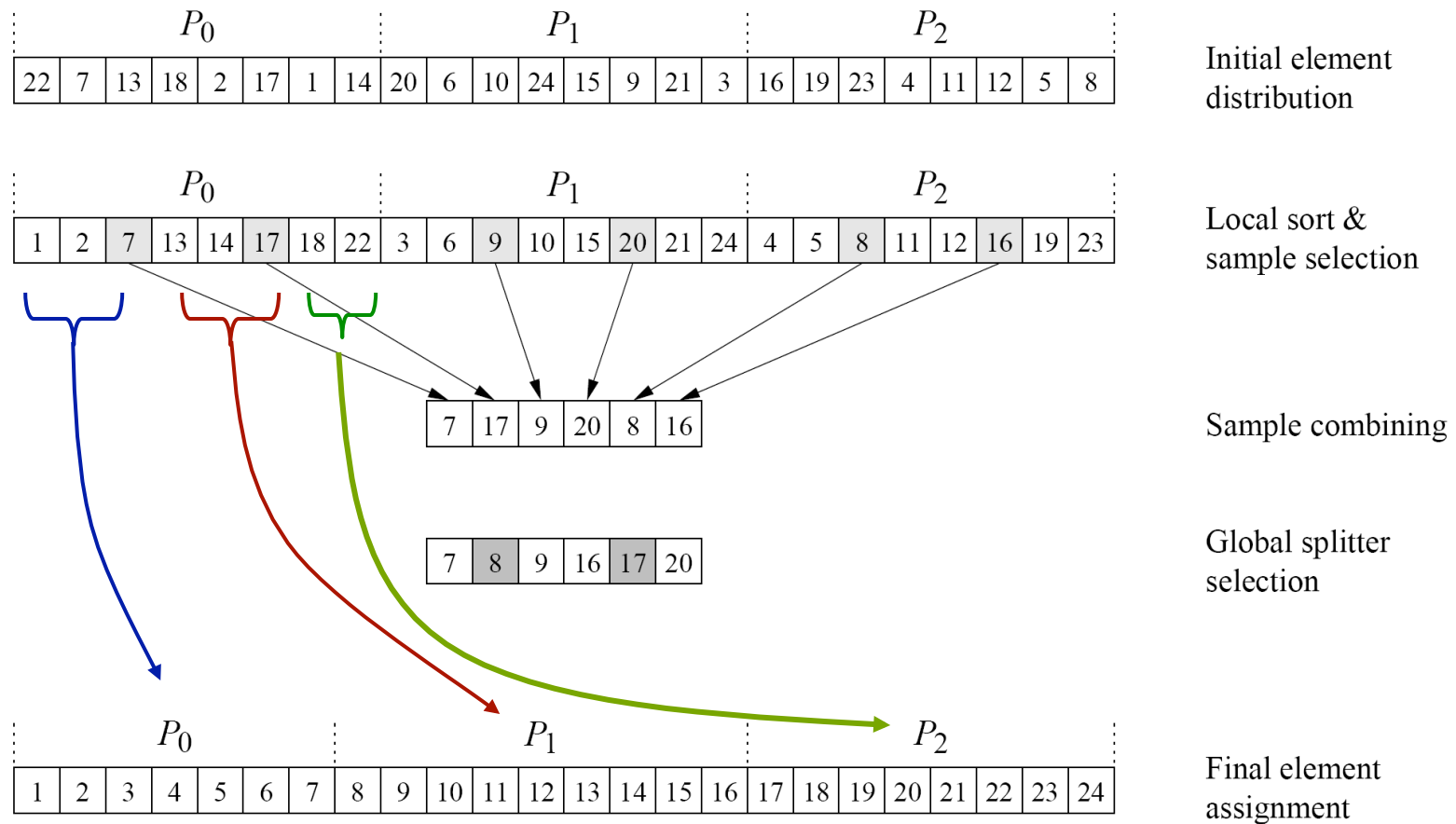
“Vector” All to All

- Generalize all-to-all, gather, etc.
- Processes supply varying length datum
- Vector all-to-all

`MPI_Alltoallv (`
 `void *sendbuf, int sendcounts[], int sDispl [],`
 `MPI_Datatype sendtype,`
 `void* recvbuf, int recvcnts[], int rDispl[],`
 `MPI_Datatype recvtype, MPI_Comm comm )`

- Used in sample sort (coming)

Alltoallv used in sample sort



Introduction to Parallel Computing, 2nd Ed., A. Grama, A. Gupta, G. Karypis, and V. Kumar, Addison-Wesley, 2003.

Details of the algorithms

- Broadcast
- AllReduce
- Scatter/gather
- Allgather
- All to all
- **Revisiting Broadcast**

Revisiting Broadcast

- P may not be a power of 2
- We use a binomial tree algorithm
- We'll use the hypercube algorithm to illustrate the special case of $P=2^k$
- Hypercube algorithm is efficient for short messages
- We use a different algorithm for long messages

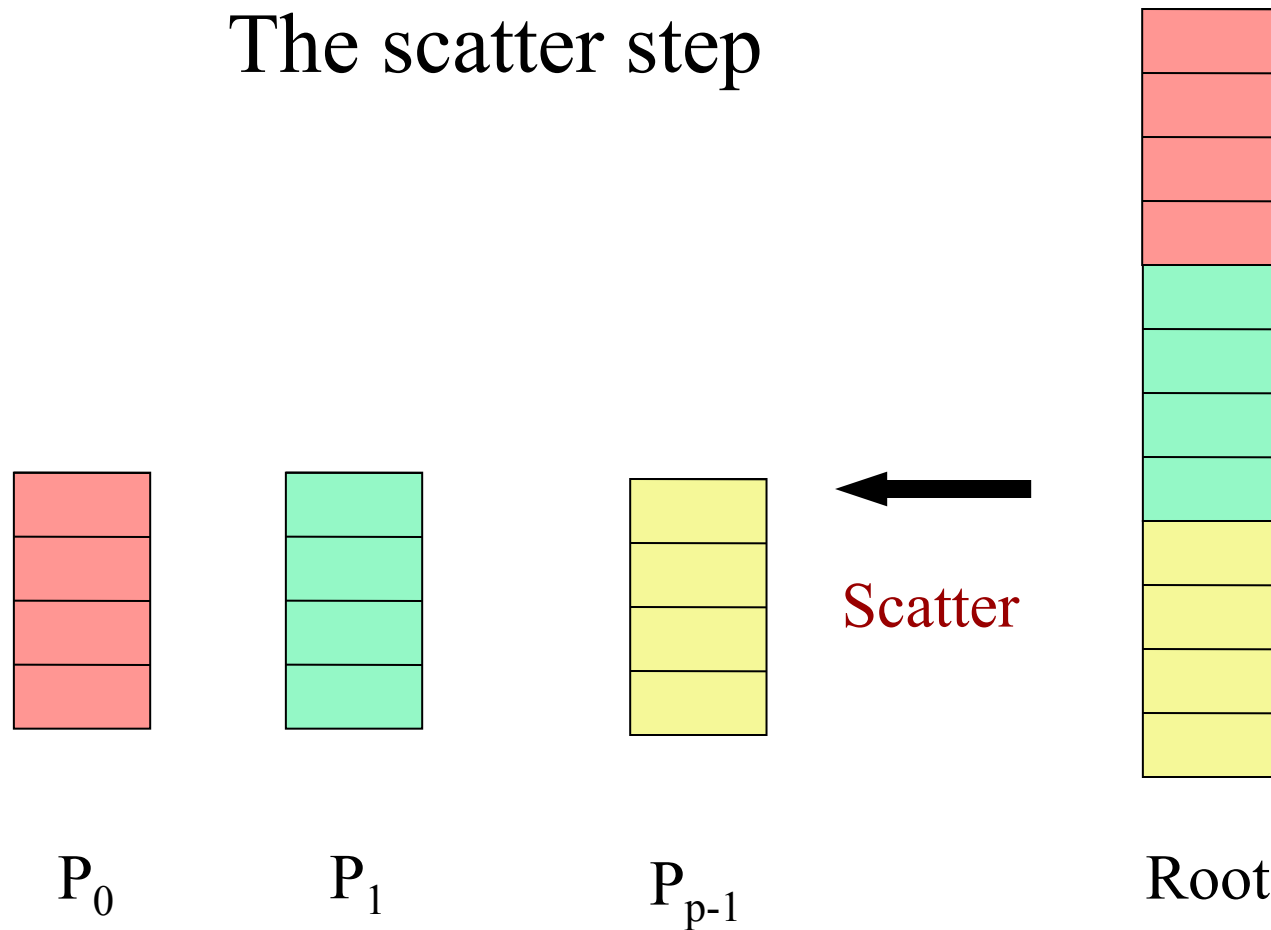
Strategy for long messages

- Based van de Geijn's strategy
- Scatter the data
 - ♦ Divide the data to be broadcast into pieces, and fill the machine with the pieces
- Do an Allgather
 - ♦ Now that everyone has a part of the entire result, collect on all processors
- Faster than MST algorithm for long messages

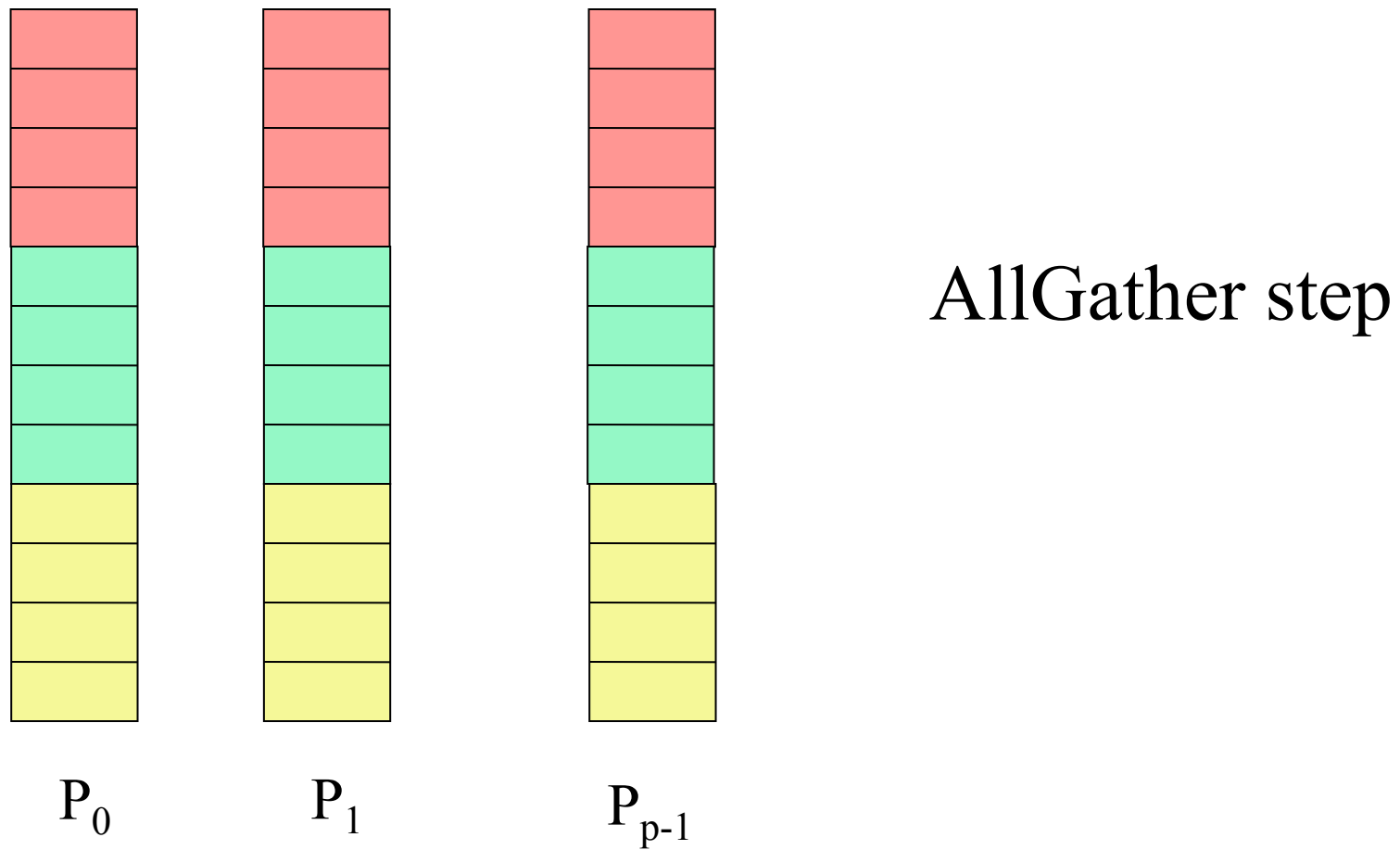
$$2 \frac{p-1}{p} n\beta \ll \lceil \lg p \rceil n\beta$$

Algorithm for long messages

The scatter step



Algorithm for long messages



Fin