Lecture 3: Cost of Communications

September 27, 2022

How Long Does It Take to Communicate?

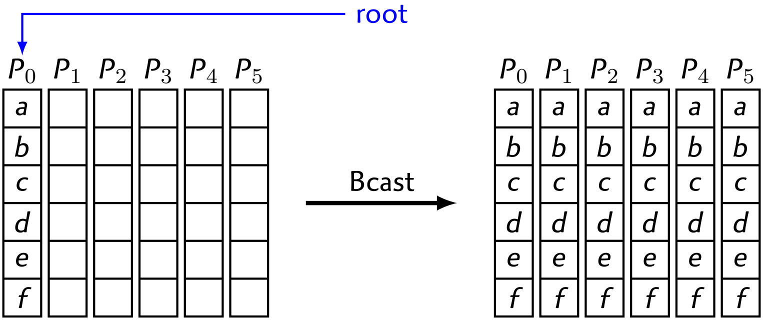
Sending a message along a network link

- Reasonable to assume $T = \alpha + n \times \beta$, where
  - n = message size (bits)
  - α = latency (s)
    - MPI layer, OS, NICs, Switches, ...
  - $\beta = 1/$ bandwidth (s / bit).
- Assumption: full-duplex
  - NICs can send & receive at the same time

Latency VS Bandwidth

Cluster	Year	Interconnect	$1/\beta$	α (μs)	$n\beta = \alpha$
PPTI	20??	1Gbit ethernet	64MB/s	50.0	3.2K
sagitaire	2006	1Gbit ethernet	116MB/s	65.0	7.5K
taurus	2012	10Gbit ethernet	1156MB/s	25.9	30K
gros	2019	25Gbit ethernet	2480MB/s	8.9	22K
grcinq	2013	56Gbit InfiniBand	2488MB/s	1.2	3K
grimoire	2016	56Gbit InfiniBand	4248MB/s	1.1	4.6K
drac	2015	100Gbit InfiniBand	7760MB/s	1.7	13K
grvingt	2018	100Gbit OmniPath	12100MB/s	0.9	11K

Broadcast

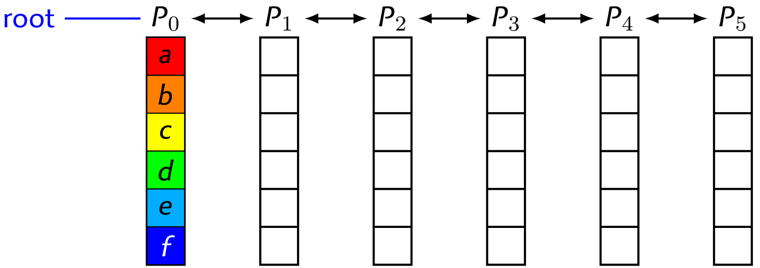


Lower bound

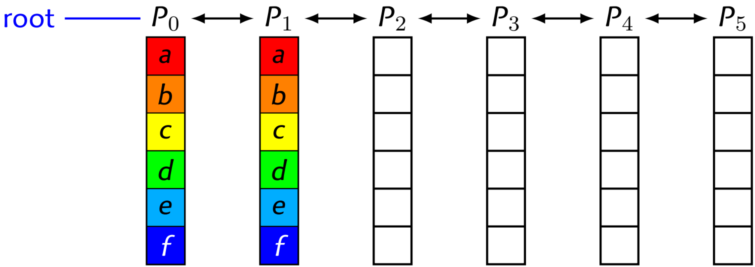
1. n elements exit from the root
   - $T \geq n\beta$ ("bandwidth term")
2. Reach everybody ("*disease propagation*")
   - $\lceil \log_2 p \rceil$ successive messages ("latency term")

$\Longrightarrow$ $T \geq \lceil \log_2 p \rceil \alpha + n\beta$

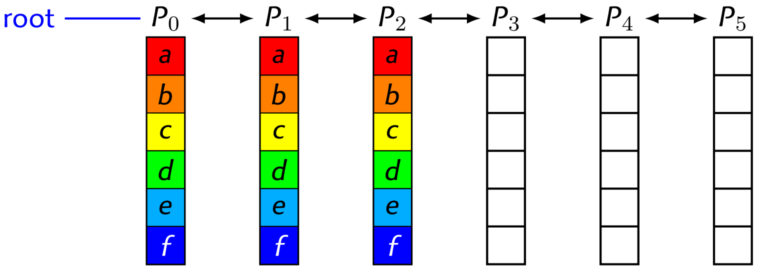
Naive Broadcast With Linear Network



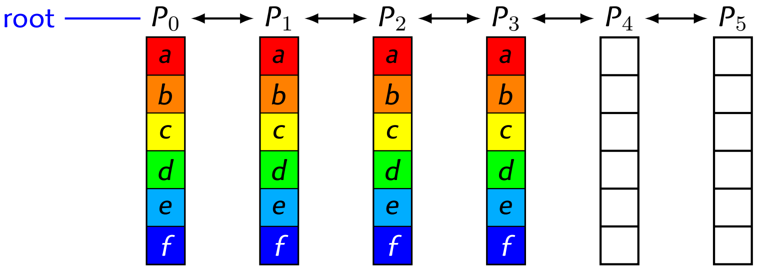
Naive Broadcast With Linear Network



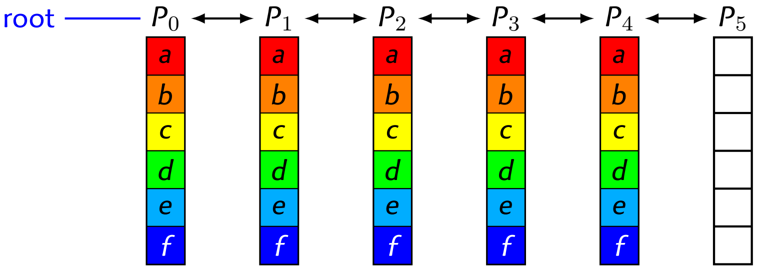
Naive Broadcast With Linear Network



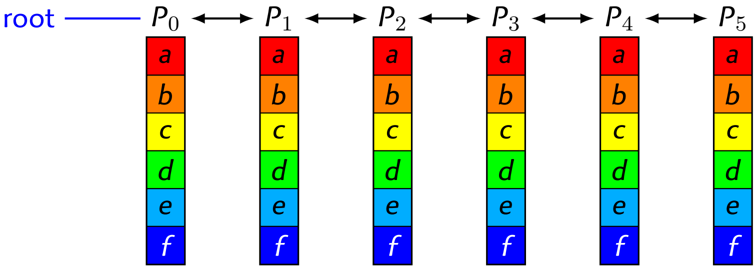
Naive Broadcast With Linear Network



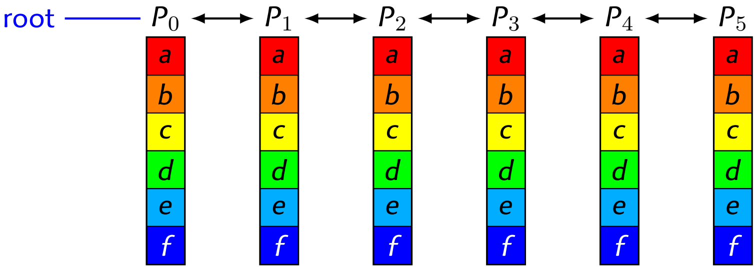
Naive Broadcast With Linear Network



Naive Broadcast With Linear Network



Naive Broadcast With Linear Network

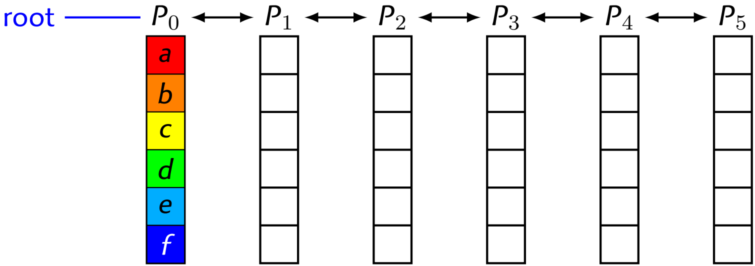


Summary

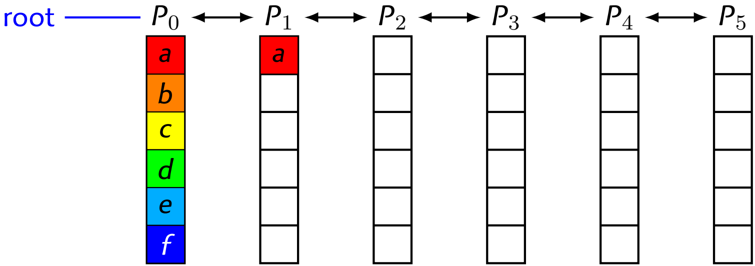
- $p-1$ successive messages of size n

$$T = (p-1)\alpha + (p-1)n\beta$$

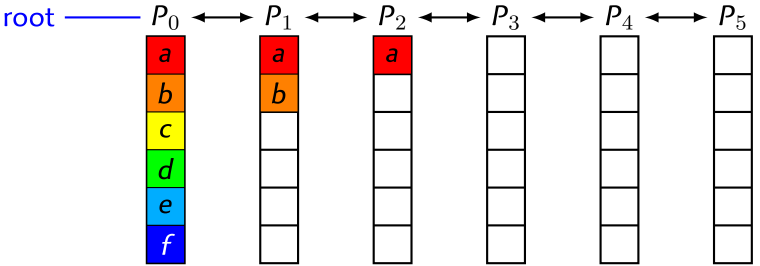
Pipelined Broadcast With Linear Network



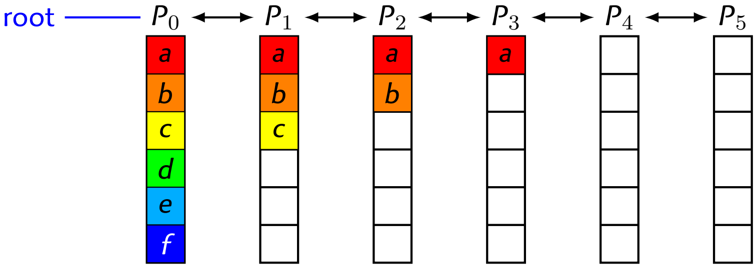
Pipelined Broadcast With Linear Network



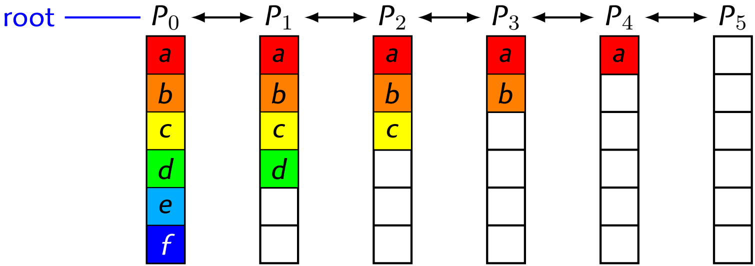
Pipelined Broadcast With Linear Network



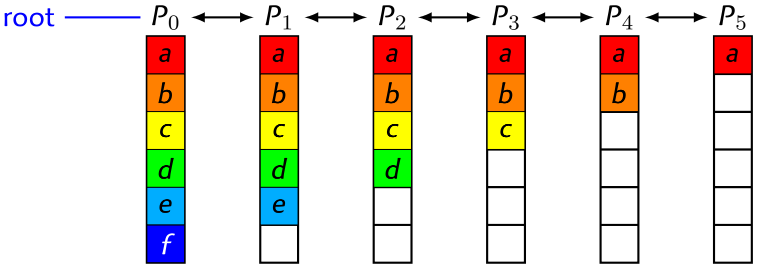
Pipelined Broadcast With Linear Network



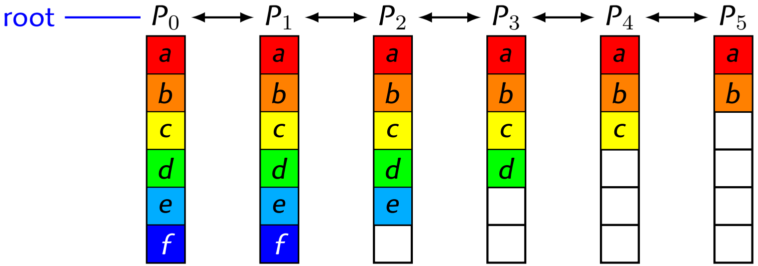
Pipelined Broadcast With Linear Network



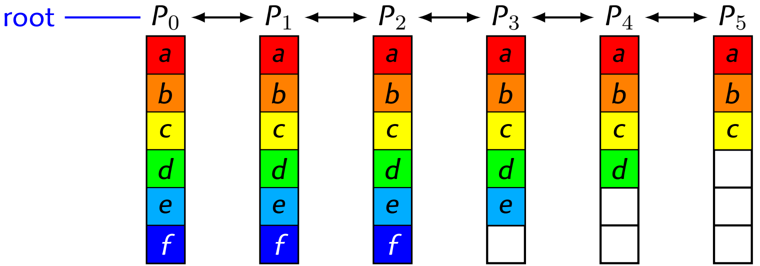
Pipelined Broadcast With Linear Network



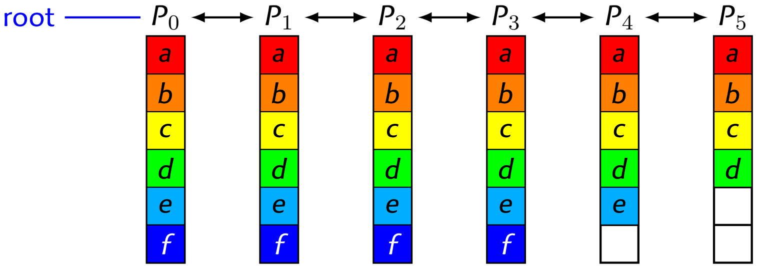
Pipelined Broadcast With Linear Network



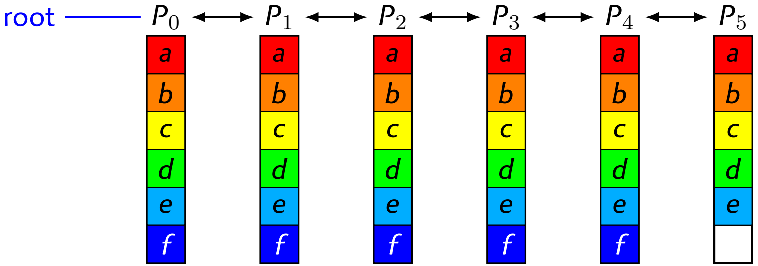
Pipelined Broadcast With Linear Network



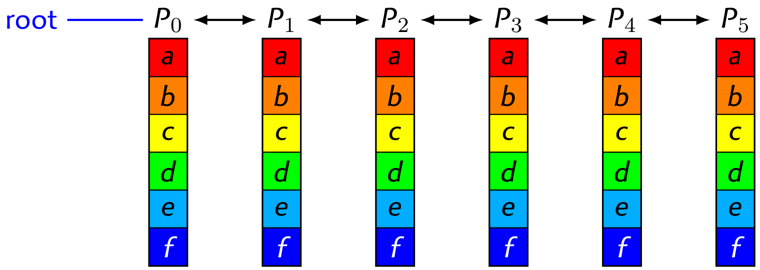
Pipelined Broadcast With Linear Network



Pipelined Broadcast With Linear Network



Pipelined Broadcast With Linear Network

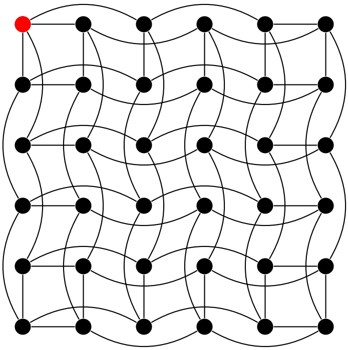


- Pipelining: k "phases"
  - Naive broadcasts of size n/k
- $k + p - 2$ successive messages of size n/k

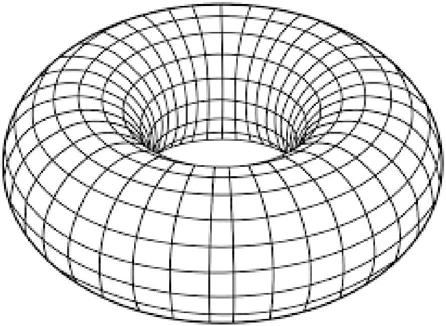
$$T = (k + p - 2)\alpha + \left(1 + \frac{p-2}{k}\right) n\beta$$

$k = p - 2$ makes sense; best choice: $k = \sqrt{(p-2)\beta n/\alpha)}$

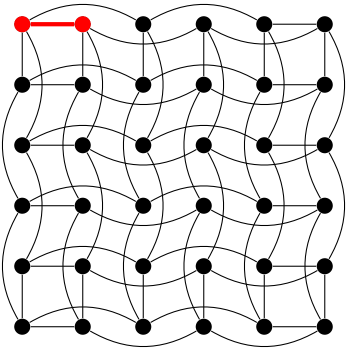
Naive Broadcast With 2D Torus Network



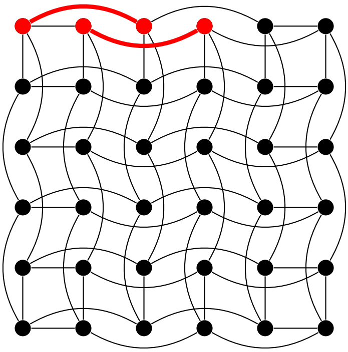
Naive Broadcast With 2D Torus Network



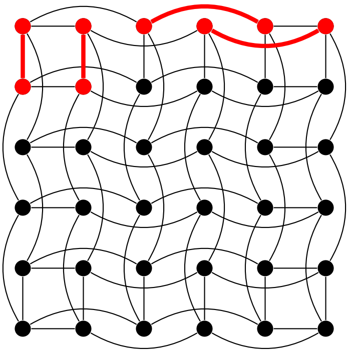
Naive Broadcast With 2D Torus Network



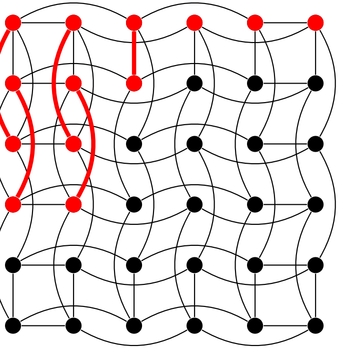
Naive Broadcast With 2D Torus Network



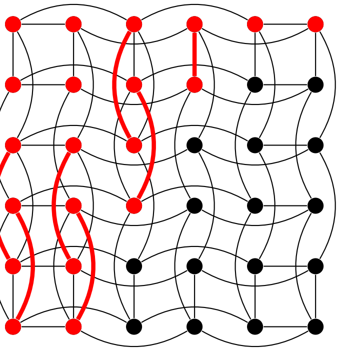
Naive Broadcast With 2D Torus Network



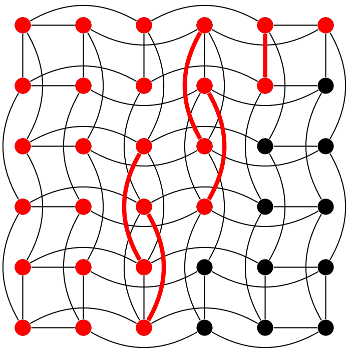
Naive Broadcast With 2D Torus Network



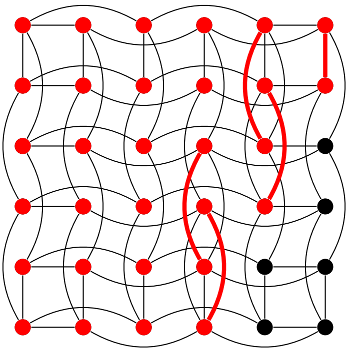
Naive Broadcast With 2D Torus Network



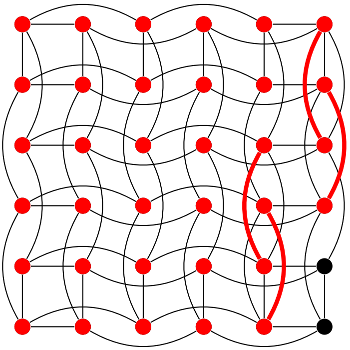
Naive Broadcast With 2D Torus Network



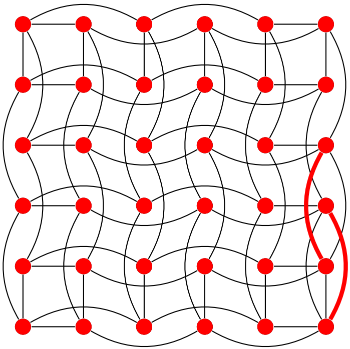
Naive Broadcast With 2D Torus Network



Naive Broadcast With 2D Torus Network



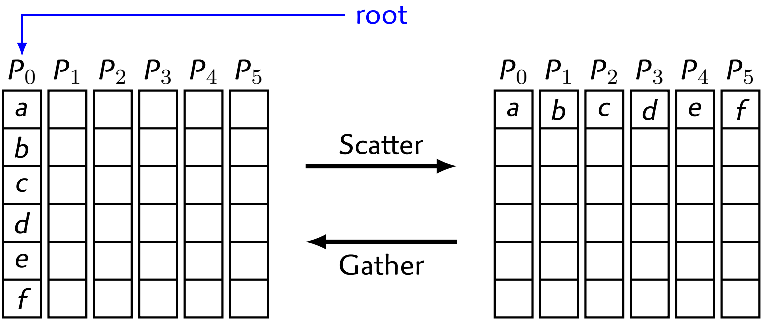
Naive Broadcast With 2D Torus Network



- $2\sqrt{p}$ successive messages of size n

$$T = 2\sqrt{p}\alpha + 2\sqrt{p}n\beta$$

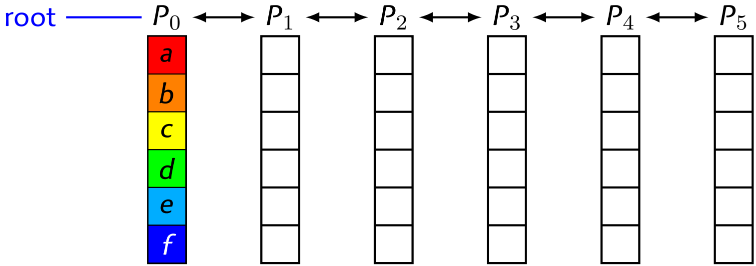
Scatter/Gather: Lower Bound



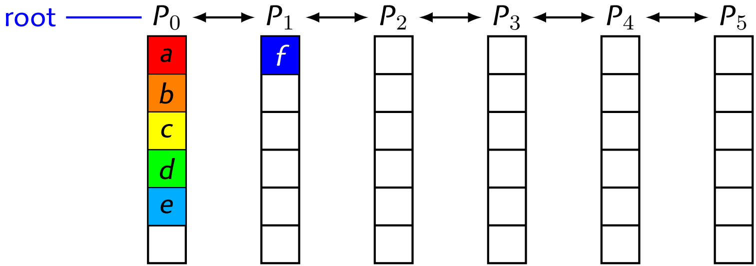
1. $(p-1)\frac{n}{p}$ elements exit from (resp. enter into) root
   - $T \geq (p-1)\frac{n}{p}\beta$ ("bandwidth term")
2. Reach everybody ("*disease propagation*")
   - $\lceil \log_2 p \rceil$ successive messages ("latency term")

$$\Longrightarrow T \geq \lceil \log_2 p \rceil \alpha + (p-1)\frac{n}{p}\beta$$

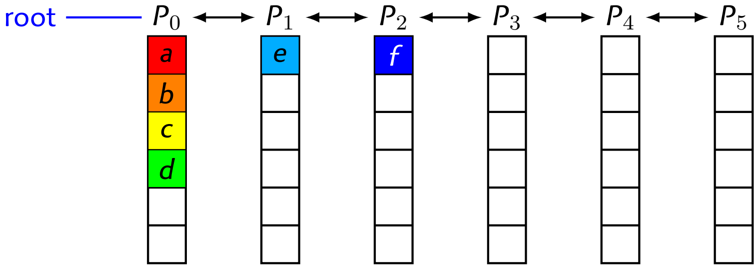
Scatter With Linear Network



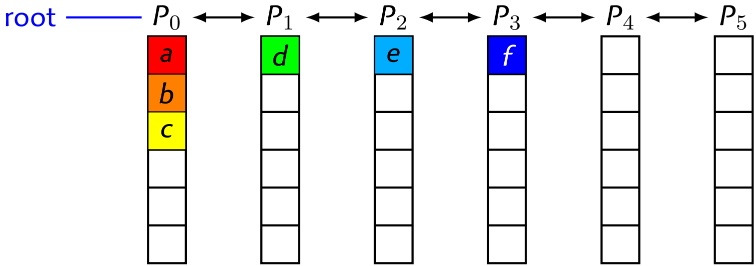
Scatter With Linear Network



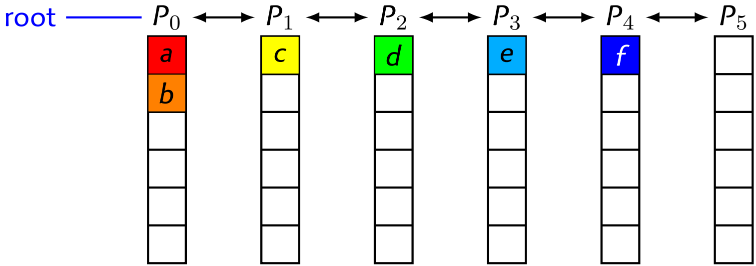
Scatter With Linear Network



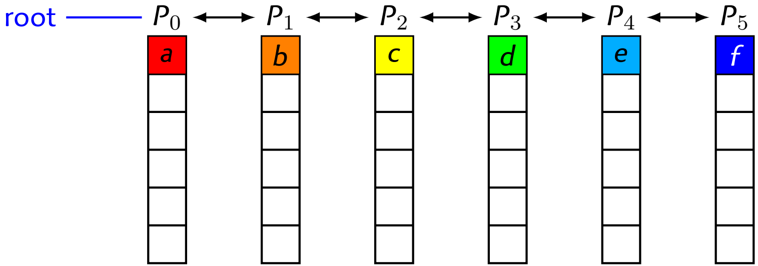
Scatter With Linear Network



Scatter With Linear Network



Scatter With Linear Network

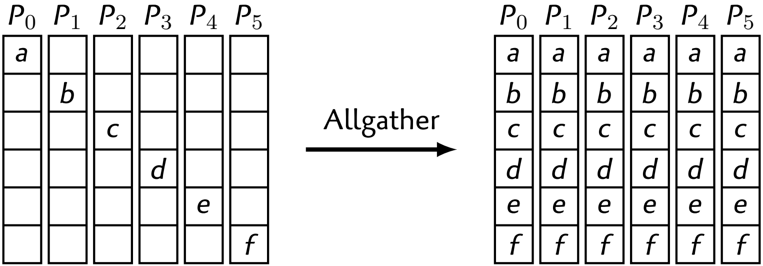


Summary

- $p-1$ successive messages of size n/p
- Parallel communications (“pipeline”)

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

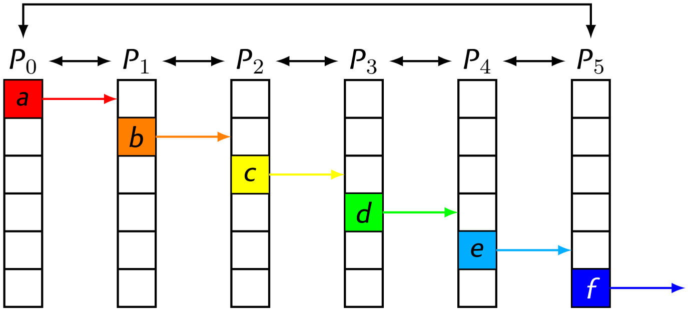
Gather-to-All



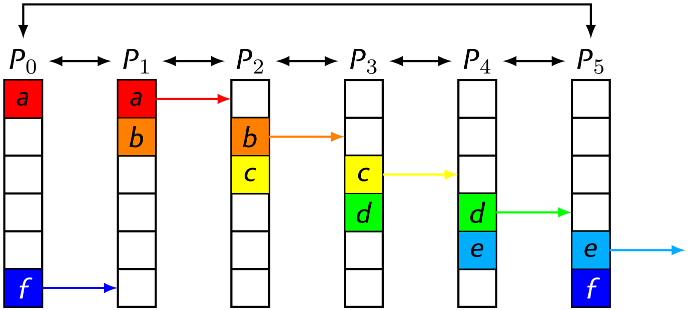
1. $(p-1)\frac{n}{p}$ elements exit from each node
   - $T \geq (p-1)\frac{n}{p}\beta$ ("bandwidth term")
2. Reach everybody ("*disease propagation*")
   - $\lceil \log_2 p \rceil$ successive messages ("latency term")

$$\Longrightarrow T \geq \lceil \log_2 p \rceil \alpha + (p-1)\frac{n}{p}\beta$$

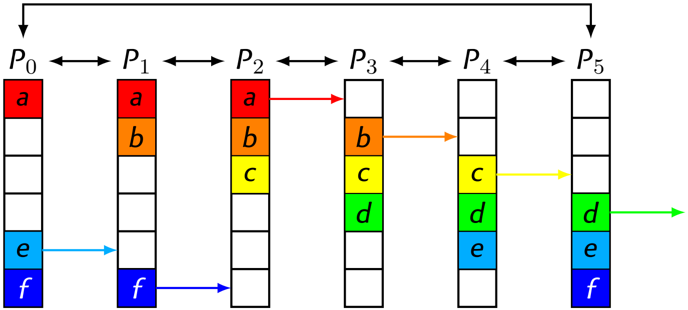
Allgather on 1D Torus Network



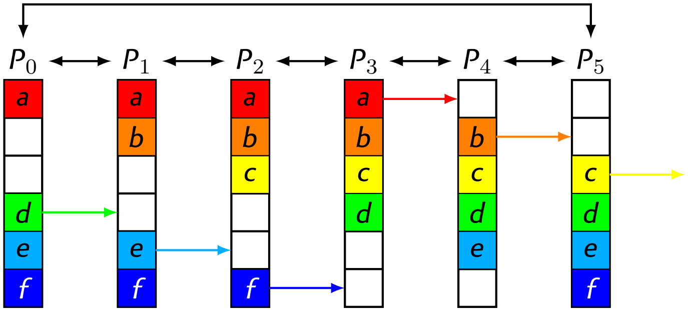
Allgather on 1D Torus Network



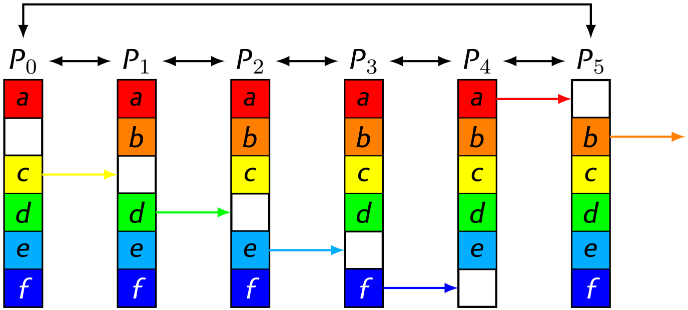
Allgather on 1D Torus Network



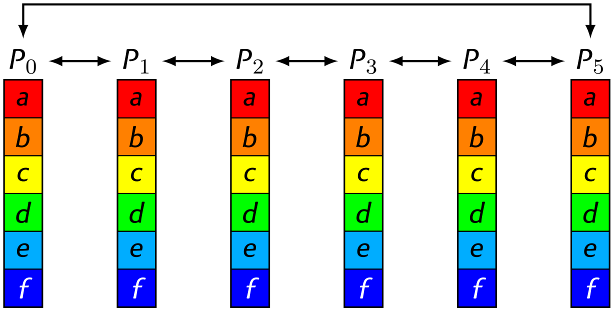
Allgather on 1D Torus Network



Allgather on 1D Torus Network



Allgather on 1D Torus Network



- Step 1: send you own data; step $i+1$: send what you just received
- $p-1$ successive messages of size n/p

$$T = (p-1)\alpha + \left(\frac{p-1}{p}\right) n\beta$$

Performance of collective operations…

… heavily affected by **network topology**

Topology

Bipartite graph with (NIC, Switch) nodes and link edges

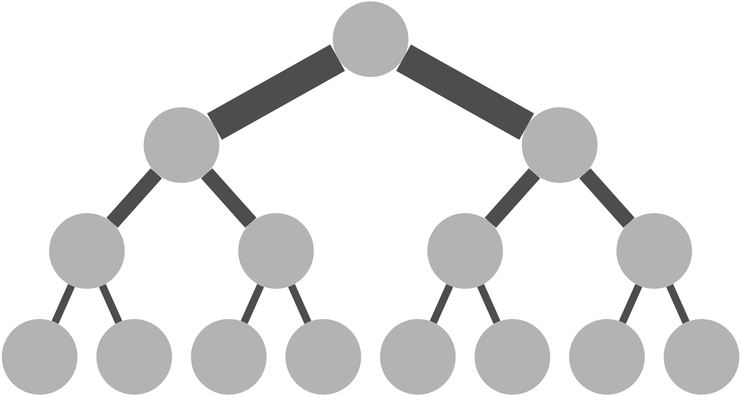
How Long Does It Take to Communicate (redux)?

P_i sends a message to **any other** process P_j

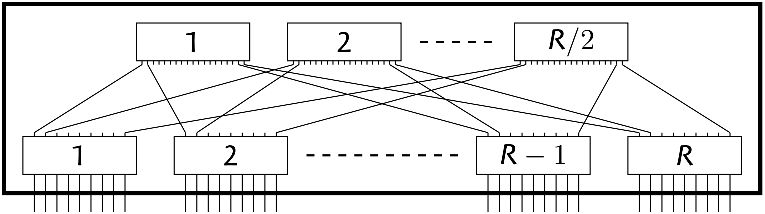
- Reasonable to assume $T = \alpha + n \times \beta$, where
  - n = message size (bits)
  - α = latency (s)
  - $\beta = 1/$ bandwidth (s / bit).
- Assumption: full-duplex
  - Processes can send & receive at the same time
- T independent of i, j (no weird topology)
- T independent from actions of other processes
  - No congestion

Realistic?

Fat Tree (Infiniband / Omnipath)



Fat tree (2 levels)



- **non-blocking** R-port "basic" switches
- $(R^2/2)$-port switch using $1.5R$ basic switches
- Infiniband : $R = 36 \rightsquigarrow$ 648 ports

Infiniband 648-port “Director” switch

image: (c) Mellanox

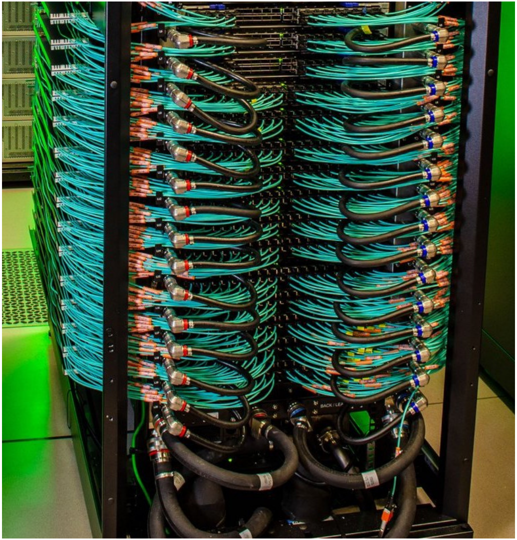
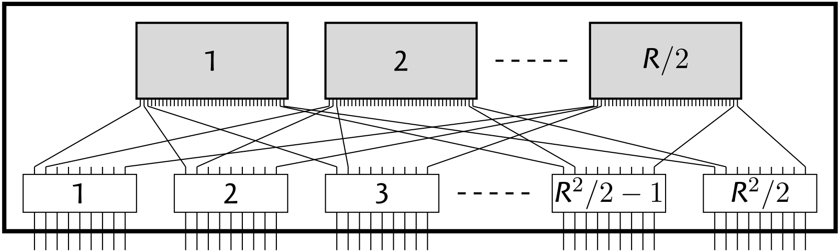


image: (c) UTexas

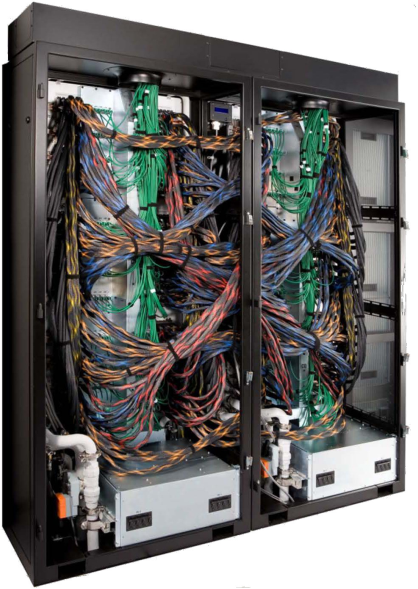
Fat tree (3 levels)



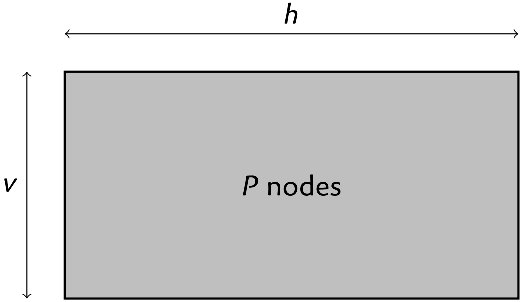
- Same technique, recursively
- $(R^3/4)$-ports with $1.25R^2$ basic R-port switches
- Infiniband : $R = 36 \rightsquigarrow 11\,644$ ports
- Used in `Summit` with $9\,216$ CPUs
  - 1 rack = 18 nodes = 36 CPUs = 36 NICs
  - Two 36-port switches per rack
  - 256 racks
  - 15 “director” (level-2) switches

Fat tree — High Cabling Costs

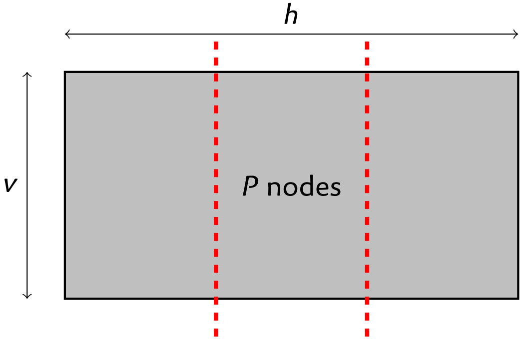
It ends up looking like this



Fat tree — High Cabling Costs

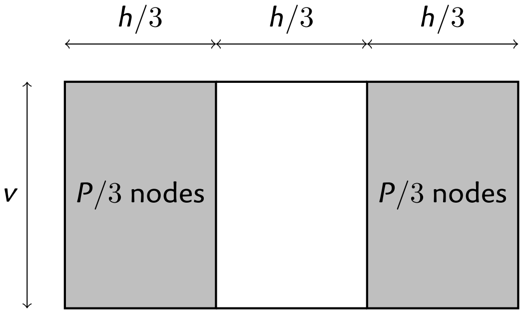


Fat tree — High Cabling Costs



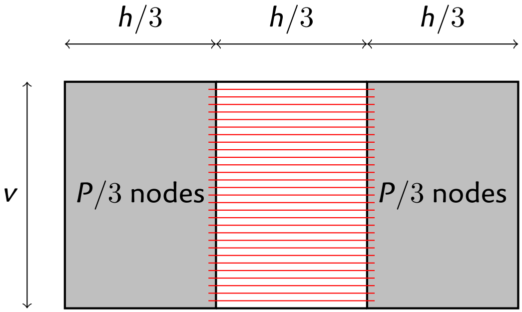
- Split machine in 3

Fat tree — High Cabling Costs



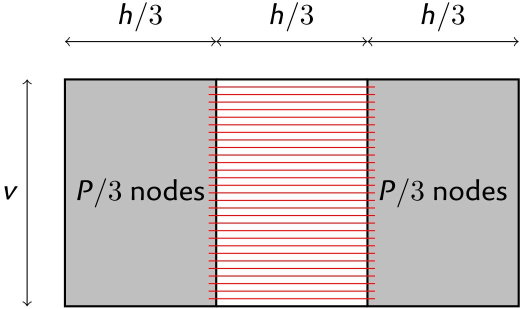
- Split machine in 3
- All nodes on the left send messages to nodes on the right

Fat tree — High Cabling Costs



- Split machine in 3
- All nodes on the left send messages to nodes on the right
- Assumption = no congestion
  - ⇝ Need $P/3$ parallel network links across the middle
  - ⇝ Length $\geq h/3$

Fat tree — High Cabling Costs



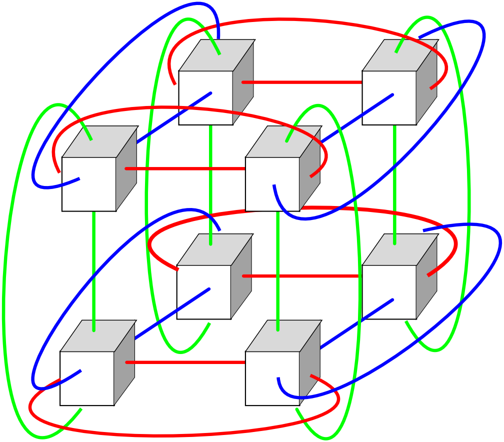
- Split machine in 3
- All nodes on the left send messages to nodes on the right
- Assumption = no congestion
  - ⇝ Need $P/3$ parallel network links across the middle
  - ⇝ Length $\geq h/3$

$\Longrightarrow$ cable length $\geq hP/9 = \Omega\left(P^{1.5}\right)$

- Fat tree a viable options with "small" #nodes ($\leq$ 10k)

Cost-Saving Alternatives

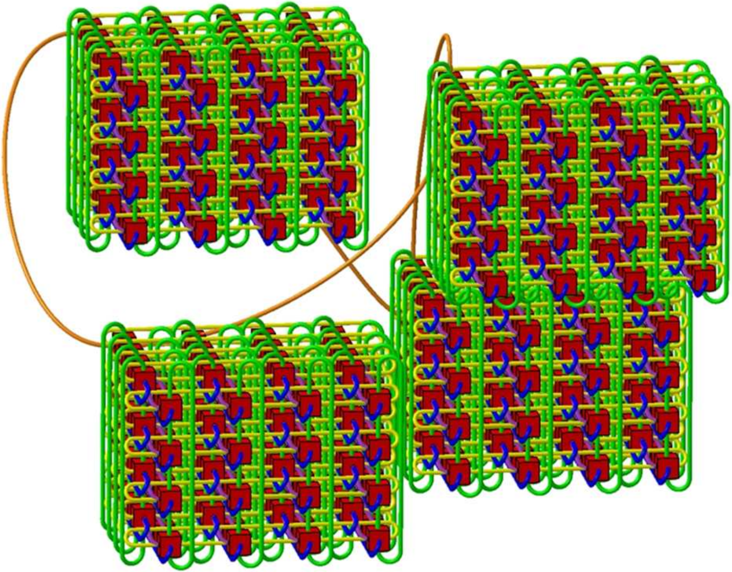
3D Torus (IBM BlueGene/P, Cray XT3, ...)



- Cable length proportional to #Nodes
- Used on very large machines
  - E.g. $48 \times 72 \times 24 \approx 100$k Nodes

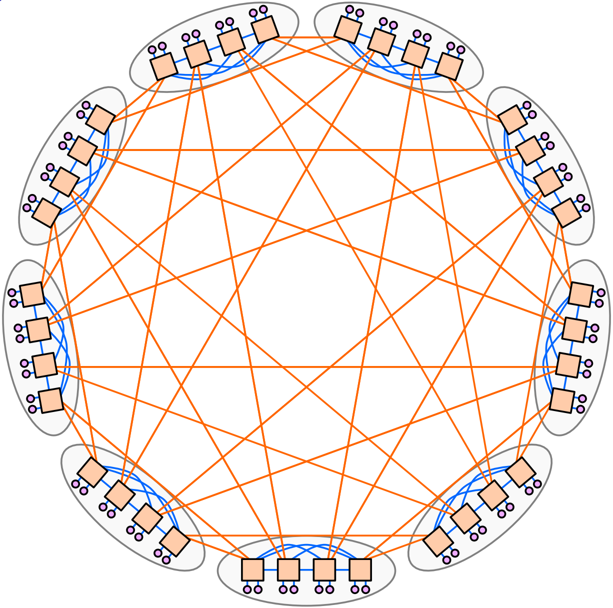
Cost-Saving Alternatives

5D Torus (IBM BlueGene/Q)



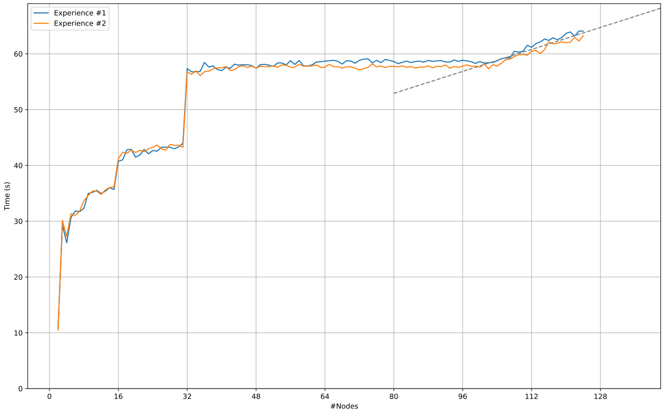
Compromise

Dragonfly (Cray "Aries" Interconnect)



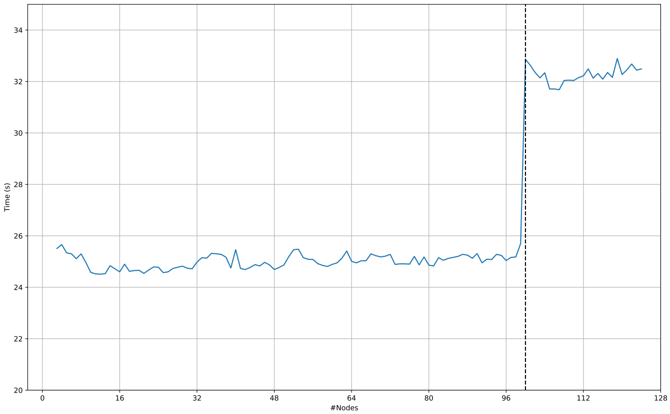
On Grid5000 ?

Except in special case, all nodes of a cluster connected to a switch
MPI All-to-All:

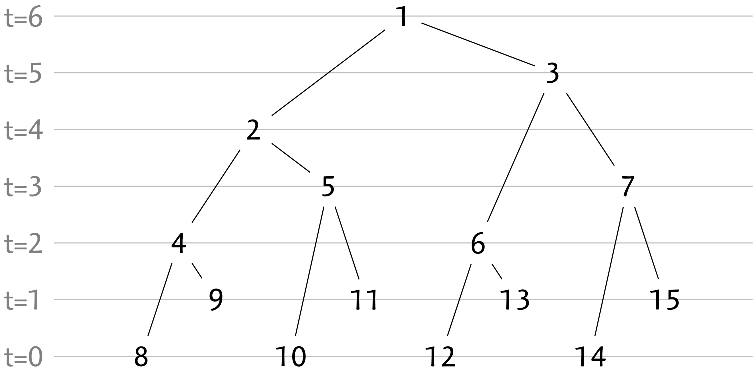


On Grid5000 ?

Ring algorithm:



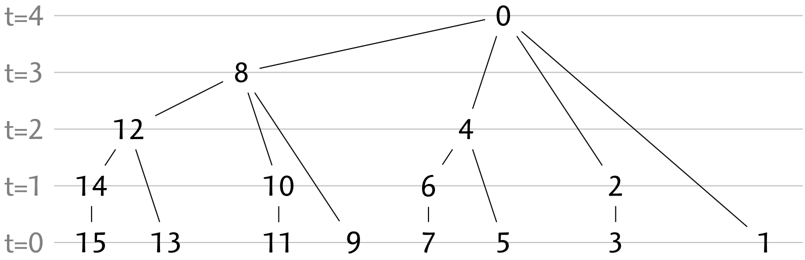
Scatter/Gather Again: Binary Tree Algorithm



- $2^{h+1} - 1$ nodes in total ; $2h$ successive messages
- 2^{h-i} nodes of height i. Each transfers:
  - Upwards $(2^{i+1} - 1)n/p$ items
  - Downwards $2 \times (2^{i} - 1)n/p$ items

$\Rightarrow$ $T \leq 2\lceil \log_2 P \rceil \alpha + 2(p-1)/pn\beta$

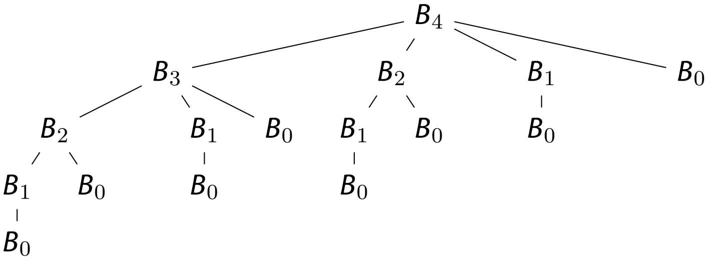
Scatter/Gather Again: **Binomial** Tree Algorithm



- 2^h nodes ; h successive messages
- Nodes of height i transfer $2^i n/p$ items (synchronously)

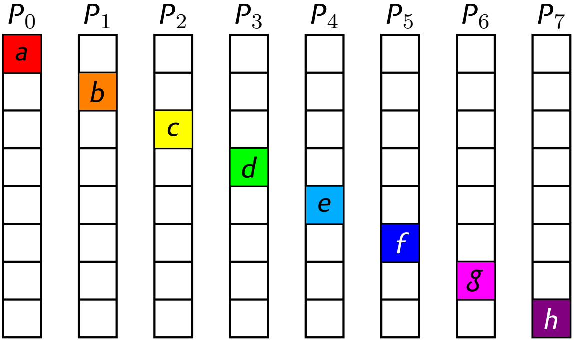
$$T = \text{message of size } 1 + \cdots + \text{message of size } n/2$$
$$= \alpha\lceil \log_2 p \rceil + (p-1)\frac{n}{p}\beta \qquad \text{(optimal)}$$

Interlude: Binomial Tree



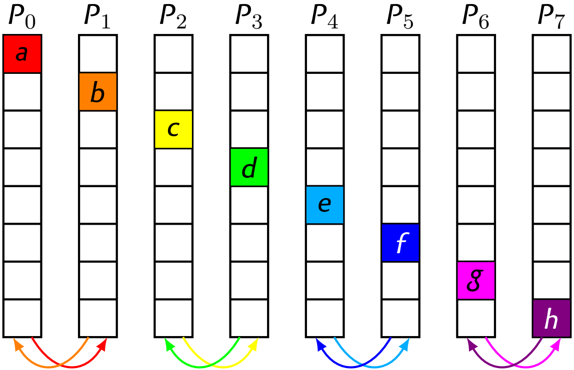
- B_i = root + $B_0 + B_1 + \ldots B_{i-1}$
- $B_i = B_{i-1} + B_{i-1}$
- $\binom{h}{i}$ nodes at height i

AllGather again: Recursive Doubling Algorithm



$$T = \text{msg of size } n/p + \cdots + \text{msg of size } n/4 + \text{msg of size } n/2$$
$$= \alpha \log_2 p + (p-1)\frac{n}{p}\beta \qquad \text{(optimal)}$$

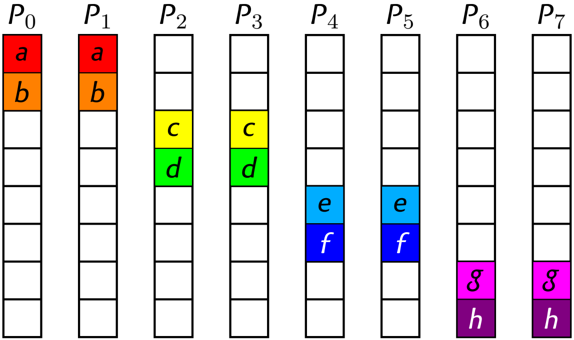
AllGather again: Recursive Doubling Algorithm



$$T = \text{msg of size } n/p + \cdots + \text{msg of size } n/4 + \text{msg of size } n/2$$

$$= \alpha \log_2 p + (p-1)\frac{n}{p}\beta \qquad \text{(optimal)}$$

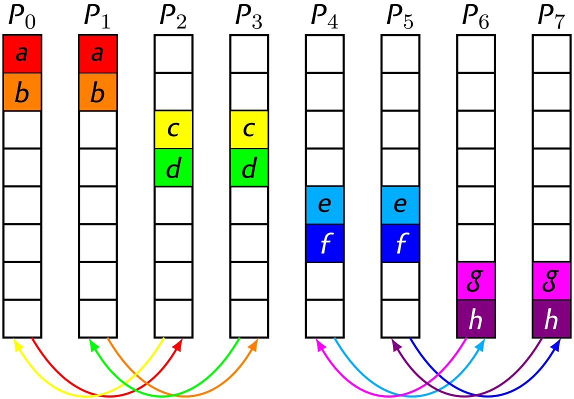
AllGather again: Recursive Doubling Algorithm



$$T = \text{msg of size } n/p + \cdots + \text{msg of size } n/4 + \text{msg of size } n/2$$

$$= \alpha \log_2 p + (p-1)\frac{n}{p}\beta \qquad \text{(optimal)}$$

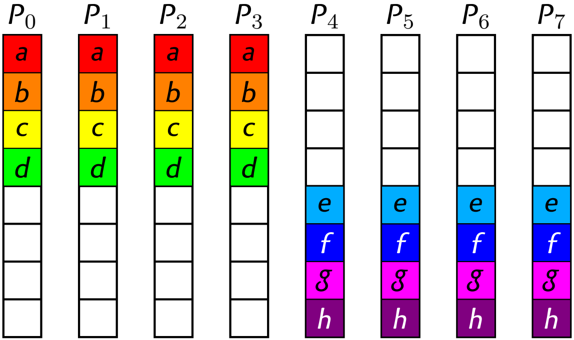
AllGather again: Recursive Doubling Algorithm



$$T = \text{msg of size } n/p + \cdots + \text{msg of size } n/4 + \text{msg of size } n/2$$

$$= \alpha \log_2 p + (p-1)\frac{n}{p}\beta \qquad \text{(optimal)}$$

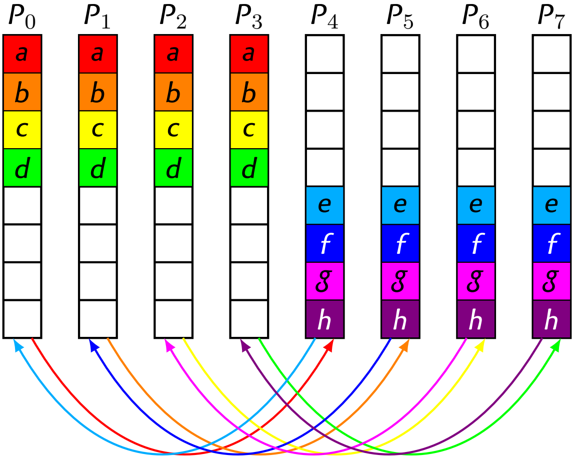
AllGather again: Recursive Doubling Algorithm



$$T = \text{msg of size } n/p + \cdots + \text{msg of size } n/4 + \text{msg of size } n/2$$

$$= \alpha \log_2 p + (p-1)\frac{n}{p}\beta \qquad \text{(optimal)}$$

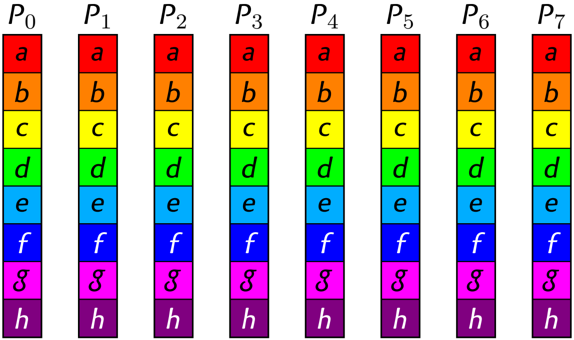
AllGather again: Recursive Doubling Algorithm



$$T = \text{msg of size } n/p + \cdots + \text{msg of size } n/4 + \text{msg of size } n/2$$

$$= \alpha \log_2 p + (p-1)\frac{n}{p}\beta$$ (optimal)

AllGather again: Recursive Doubling Algorithm



$$T = \text{msg of size } n/p + \cdots + \text{msg of size } n/4 + \text{msg of size } n/2$$

$$= \alpha \log_2 p + (p-1)\frac{n}{p}\beta \qquad \text{(optimal)}$$

Algorithms for Broadcast / Reduce

Binomial Tree

- $T = \lceil \log_2 p \rceil (\alpha + n\beta)$
- "Bandwidth" term sub-optimal
- (bad for large messages)

Van de Geijn

- Broadcast = AllGather ∘ Scatter
  - Recursive doubling / Binomial tree
- $T = 2(\log_2 p)\alpha + 2\left(1 - \frac{1}{p}\right) n\beta$

$\rightarrow$ $2\times$ lower bound