PURDUE UNIVERSITY

STyGIANet

Collective Communication

CS59000: AI/DC Networking

Fall 2025

Vamsi Addanki
vaddank@purdue.edu

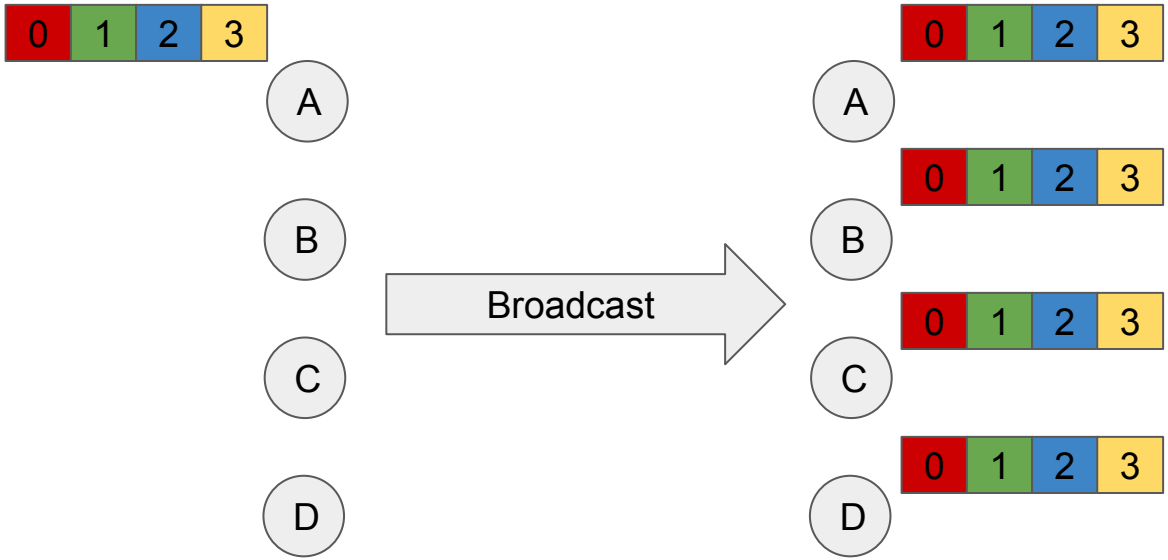
https://stygianet.cs.purdue.edu

Recap Motivation: Parallel Computing

- Communication is an essential part of massively parallel distributed computing
- This lecture:
  - Collective communication primitives
  - Cost model for analyzing collectives
  - Collective communication algorithms (Optimizing the comm. time)
- Fun fact: Collective communication is not new. GPU clusters are new. These algorithms have been widely studied for decades e.g., Message Passing Interface (MPI)
  - Nvidia's NCCL, AMD's RCCL,.... are all basically reincarnations of MPI
  - "CCL" stands for Collective Communication Library

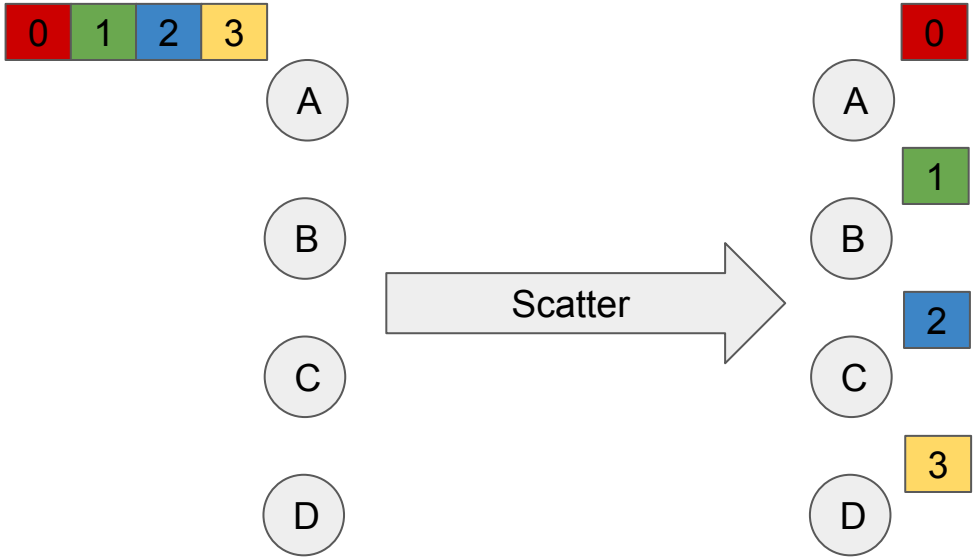
Recap Collective Communication Primitives: Broadcast

- A single node transmits its entire data to all other nodes



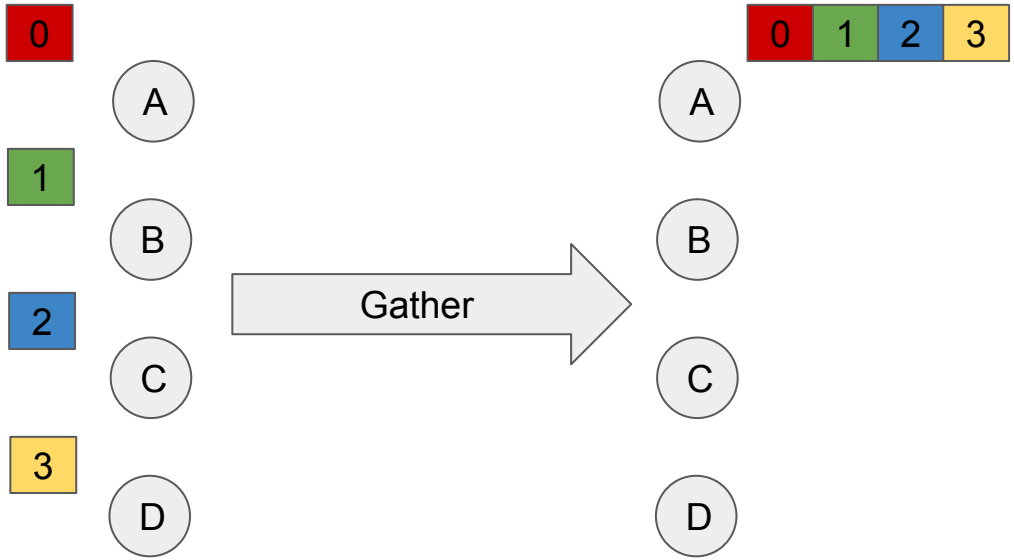
Recap Collective Communication Primitives: Scatter

- A single node transmits *distinct* chunks of its data to all other nodes



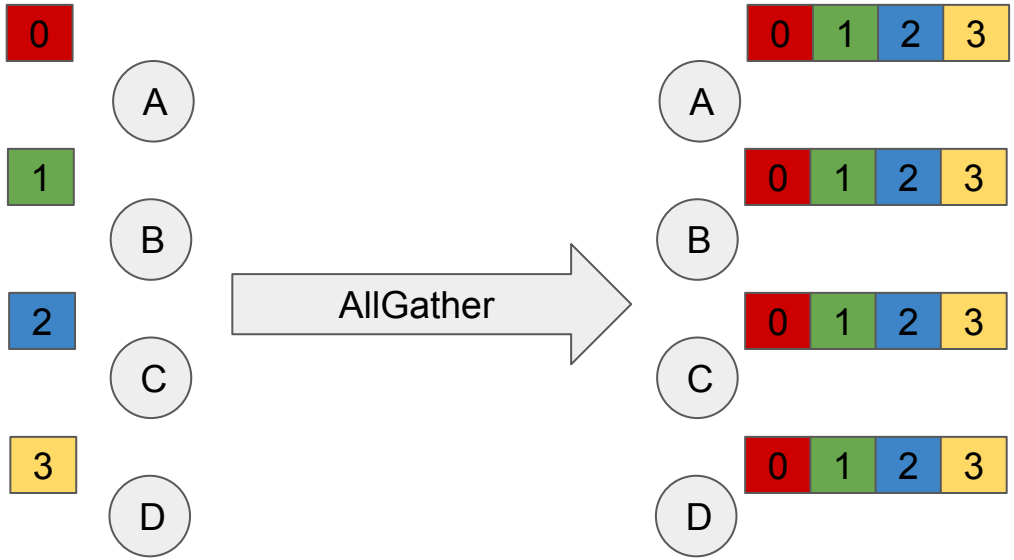
Recap Collective Communication Primitives: Gather

- Every node transmits *distinct* chunks of its data to a single (root) node



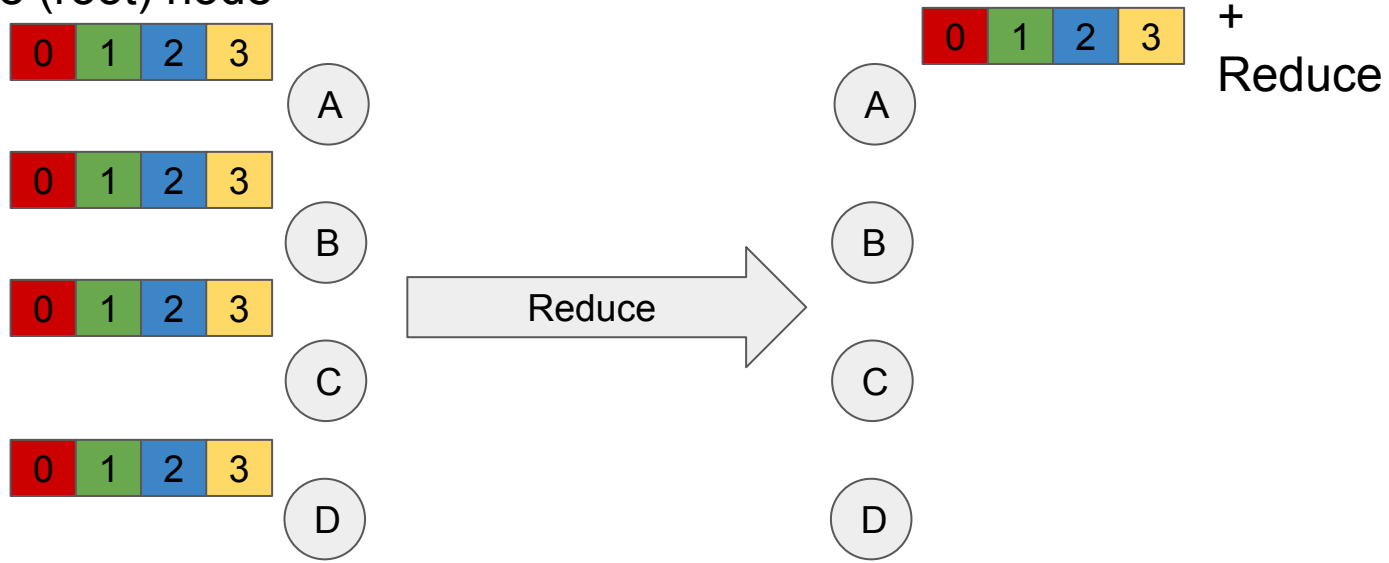
Recap Collective Communication Primitives: AllGather

- Every node transmits *distinct* chunks of its data to all other nodes



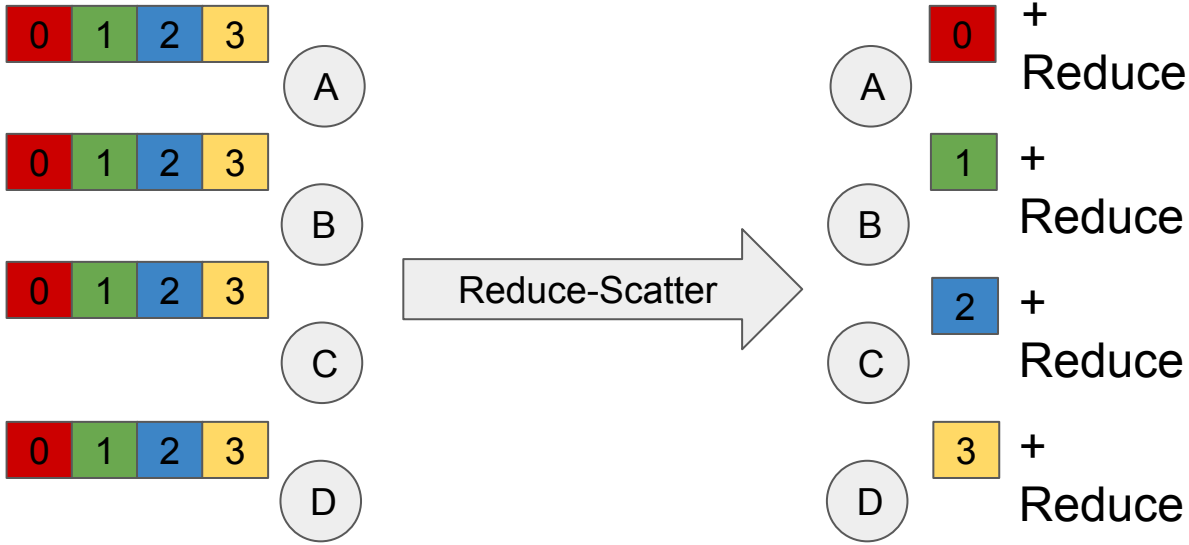
Recap Collective Communication Primitives: Reduce

- Data across all the nodes is globally aggregated (reduced e.g., sum, max) at a single (root) node



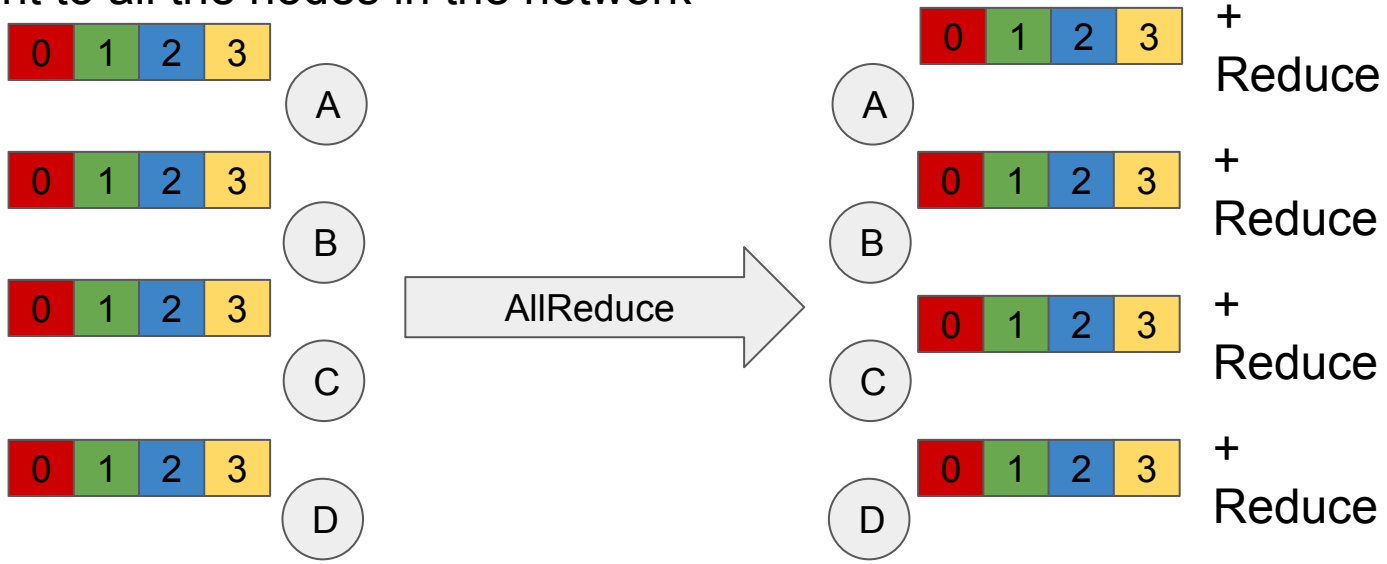
Recap Collective Communication Primitives: Reduce-Scatter

- Data across all the nodes is globally aggregated (reduced e.g., sum, max) and distinct chunks of the reduced results are scattered across the nodes



Recap Collective Communication Primitives: AllReduce

- Data across all the nodes is globally aggregated (reduced e.g., sum, max) and sent to all the nodes in the network



α,β Cost Model

- **Latency (α)**: This is the fixed latency incurred in transmitting a message. α can be interpreted as the latencies such as initiating a communication operation, propagation delays etc., that are independent of the size of the data being transmitted
- **Bandwidth (1/β)**: This is the rate at which data can be transmitted between two nodes. β represents the time per unit of data transferred across the network
- Cost of sending a message of size m on a single link:

$$T_{\text{total}} = \alpha + \beta \cdot m$$

α,β Cost Model

Assumptions:

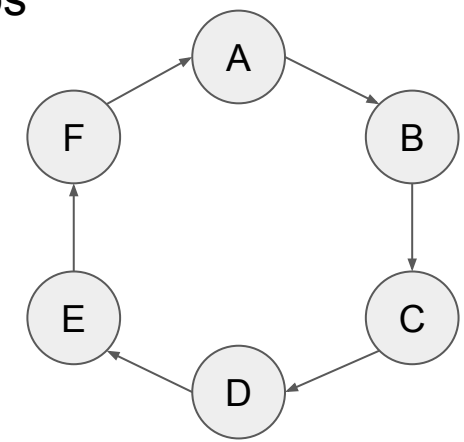
- All the nodes are equidistant from each other i.e., the latency is the same across all pairs of nodes
- The bandwidth is uniform across all links (bidirectional)
- Nodes store-and-forward messages i.e., the message is completely received before it is forwarded to the next node
- Nodes can send, receive, and process messages concurrently

Algorithms for Broadcast

- Ring
- Binary tree
- Binomial tree
- many more …

Ring Broadcast

- A single node initiate the communication and sends all its data to a neighbor
- Broadcast progresses along the ring:
  - In each step one node transmits data to its neighbor
  - It takes O(n) steps

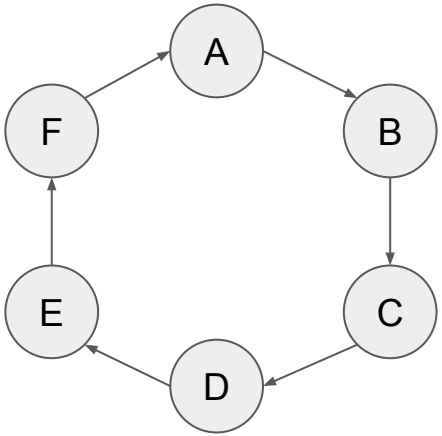


PURDUE UNIVERSITY

STyGIANet

Ring Broadcast

- Communication cost per step: $\alpha + m\,\beta$
- Total cost: $(n-1)\cdot(\alpha + m\,\beta)$
- Cost increases linearly with the number of nodes *(not desirable)!*



PURDUE UNIVERSITY

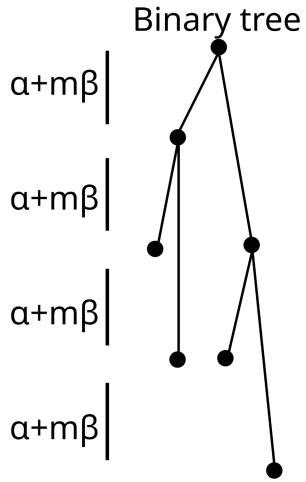
STyGIANet

Binary Tree Broadcast

- Broadcast begins at the root node
- Each node has two children (hence binary tree)
- Each node transmits data to its two children until all the nodes received data

Binary Tree Broadcast

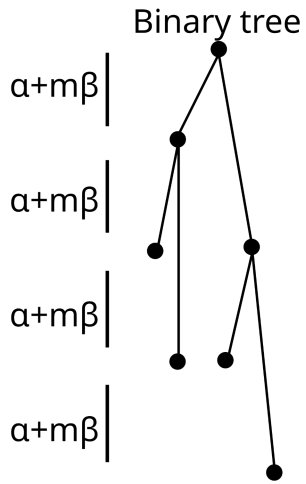
- Communication cost per step: $\alpha + m\beta$
- Total cost: $2\log(n)(\alpha + m\beta)$
- log comes from height



PURDUE UNIVERSITY
STyGIANet

Binary Tree Broadcast

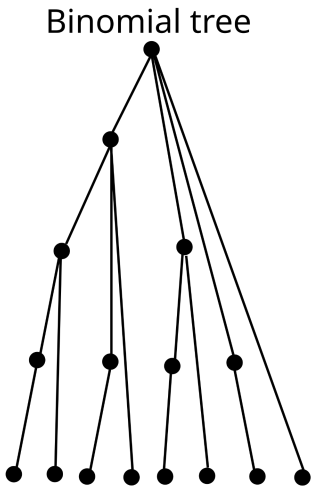
- Communication cost per step: $\alpha + m\beta$
- Total cost: $2\log(n)(\alpha + m\beta)$
- log comes from height
- Proof sketch:
  - Consider that each node transmits to its left child first
  - The right-most node in the tree receives data last, after 2 log(n) steps



PURDUE UNIVERSITY
STyGIANet

Binomial Tree Broadcast

- Binomial tree of order k is constructed by connecting the root with the roots of binomial trees from 0 to k-1 (a recursive construction)

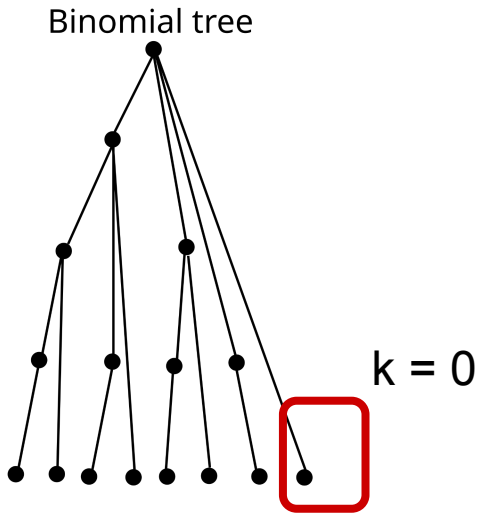


Binomial Tree Broadcast

- Binomial tree of order k is constructed by connecting the root with the roots of binomial trees from 0 to k-1 (a recursive construction)

Number of nodes:

0: 1



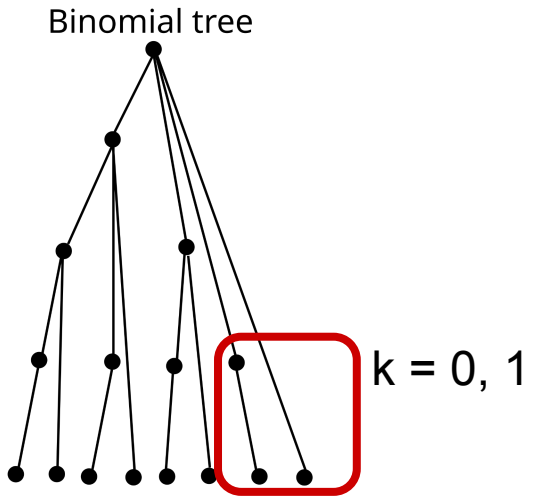
Binomial Tree Broadcast

- Binomial tree of order k is constructed by connecting the root with the roots of binomial trees from 0 to k-1 (a recursive construction)

Maximum number of nodes:

0: 1

1: 1 + 1 = 2



Binomial Tree Broadcast

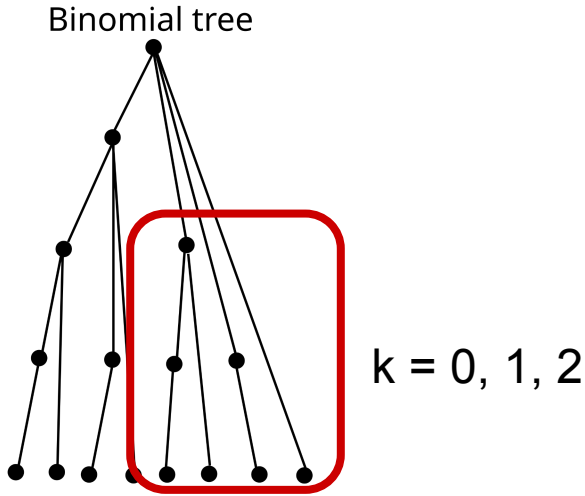
- Binomial tree of order k is constructed by connecting the root with the roots of binomial trees from 0 to k-1 (a recursive construction)

Maximum number of nodes:

0: 1

1: 1 + 1 = 2

2: 1 + 2 + 1 = 4



Binomial Tree Broadcast

- Binomial tree of order k is constructed by connecting the root with the roots of binomial trees from 0 to k-1 (a recursive construction)

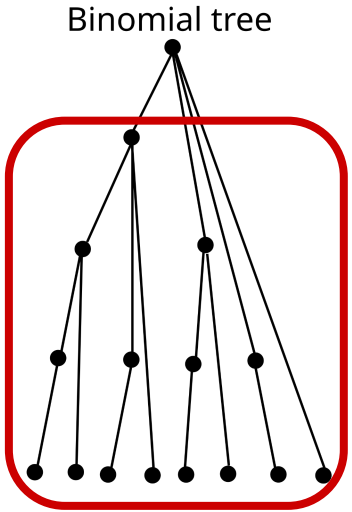
Maximum number of nodes:

0: 1

1: 1 + 1 = 2

2: 1 + 2 + 1 = 4

3: 1 + 2 + 4 + 1 = 8



k = 0, 1, 2, 3

PURDUE UNIVERSITY

STyGIANet

Binomial Tree Broadcast

- Binomial tree of order k is constructed by connecting the root with the roots of binomial trees from 0 to k-1 (a recursive construction)

Maximum number of nodes:

0: 1

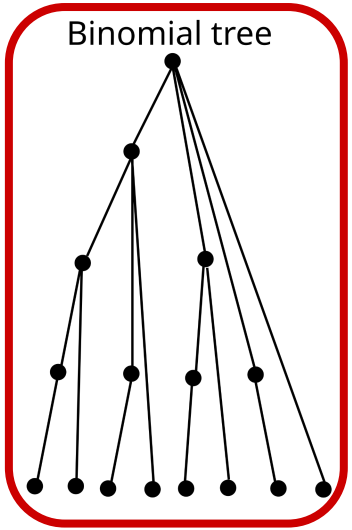
1: 1 + 1 = 2

2: 1 + 2 + 1 = 4

3: 1 + 2 + 4 + 1 = 8

4: 1 + 2 + 4 + 8 + 1 = 16

Geometric



k = 4

PURDUE UNIVERSITY

STyGIANet

Binomial Tree Broadcast

- Binomial tree of order k is constructed by connecting the root with the roots of binomial trees from 0 to k-1 (a recursive construction)

Number of nodes at depth: Binomial Coeff.

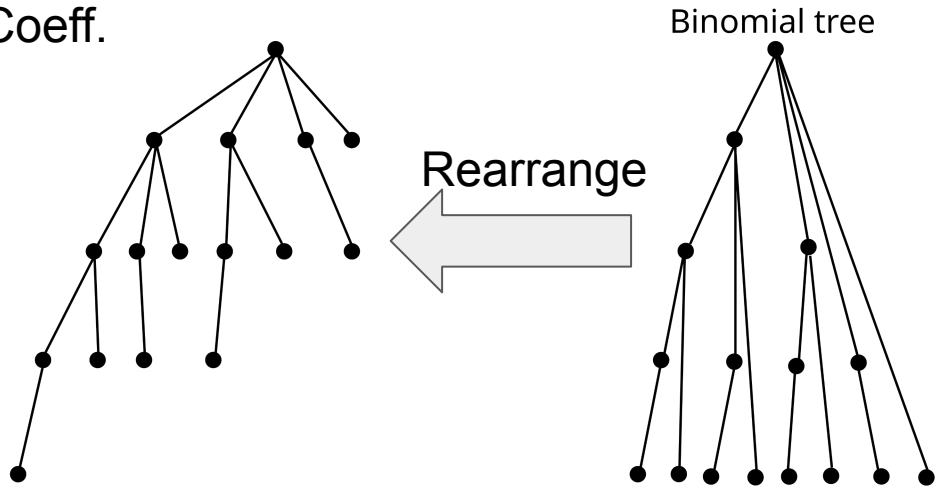
Depth 0: $\binom{4}{0} = 1$

Depth 1: $\binom{4}{1} = 4$

Depth 2: $\binom{4}{2} = 6$

Depth 3: $\binom{4}{3} = 4$

Depth 4: $\binom{4}{4} = 1$

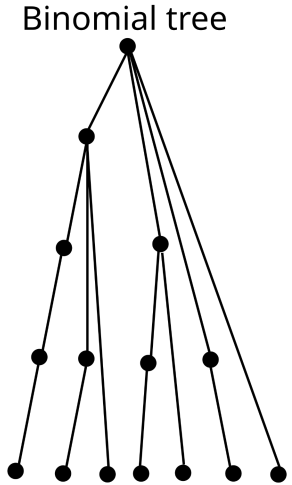


PURDUE UNIVERSITY

STyGIANet

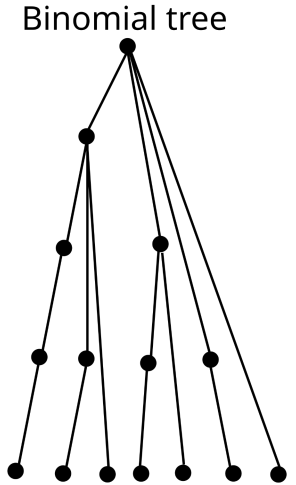
Binomial Tree Broadcast

- Communication cost per step: $\alpha + m\,\beta$
- Total cost: $\log(n)\,(\alpha + m\,\beta)$
- log comes from height
- Factor of 2 improvement



Binomial Tree Broadcast

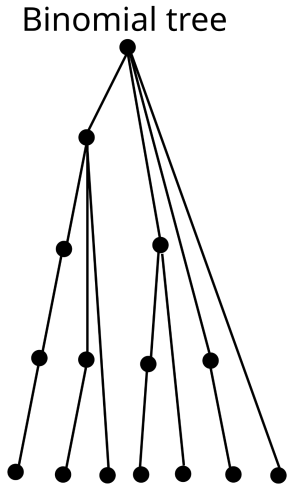
- Proof sketch:
  - Each node always transmits in the decreasing order k of the sub-trees corresponding to each child
  - Step 0: Root sends to sub-tree of order k-1
  - Next step: Broadcast begins in the sub-tree of order k-1 and the root sends to sub-tree of order k-2
  - Given the height of the tree is k = log n, the proof follows



Binomial Tree Broadcast

- Total cost: log(n) α + log(n) m β

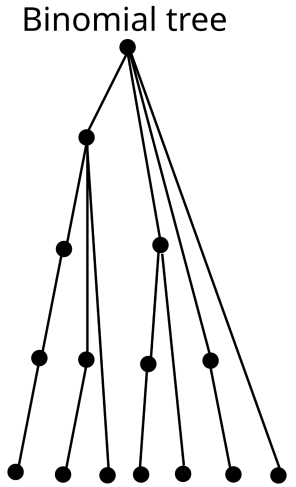
Good for short messages
O(log n)



Binomial Tree Broadcast

- Total cost: log(n) α + log(n) m β

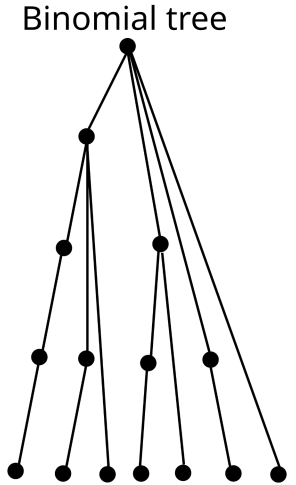
Bad for large message size m
O(m log n)



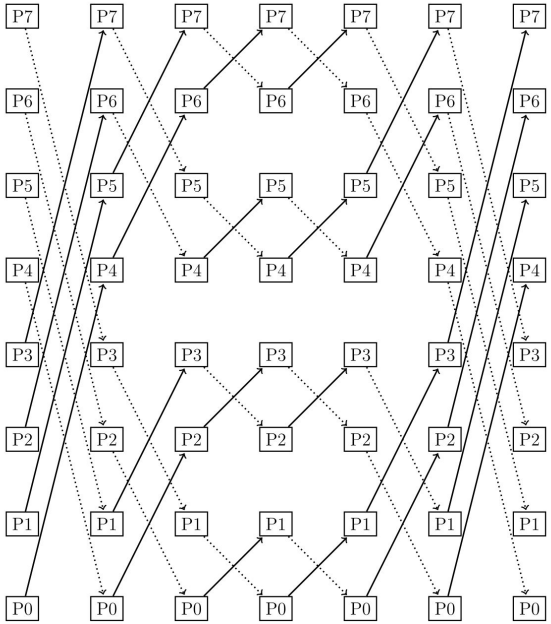
Binomial Tree Broadcast

- Total cost: log(n) α + log(n) m β

Is this optimal?
Can we do better?



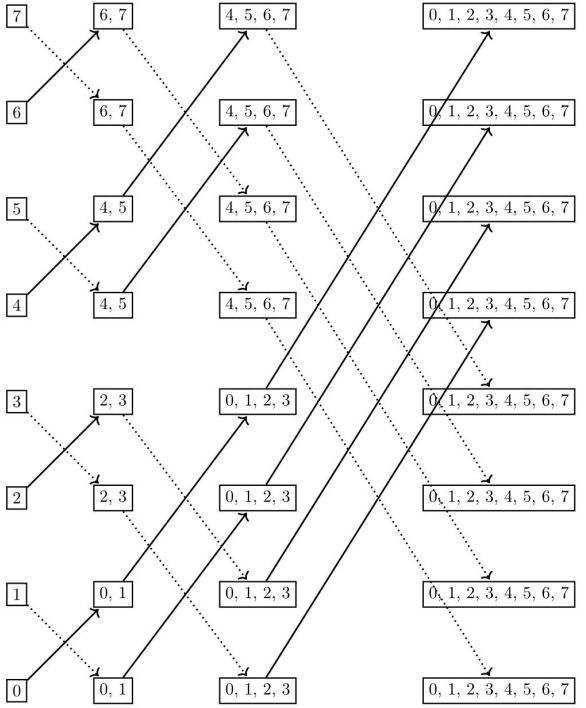
Butterfly-based Algorithms



Butterfly-based Algorithms

- Significant cost improvement in asymptotics
- Logarithmic latency factor
- Linear in m, bandwidth factor
- All primitives can be expressed as a combination of AllGather and Scatter
  - e.g., Broadcast can be implemented as scatter first and then AllGather
  - Improves bandwidth cost by log n

Butterfly AllGather (Recursive Doubling)

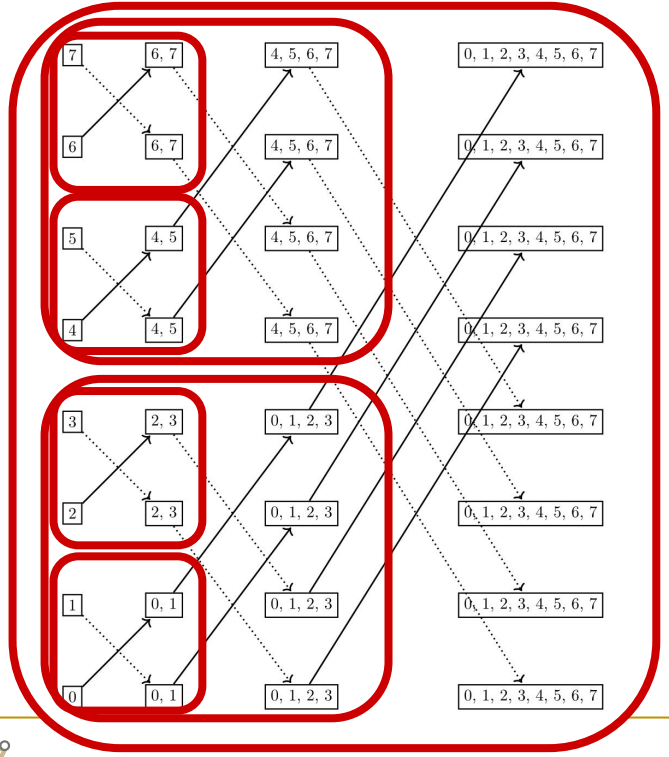


PURDUE UNIVERSITY

STyGIANet

Butterfly AllGather (Recursive Doubling)

- Notice the recursive structure
- AllGather for message size m with n nodes can be expressed as a function of allgather for message size m/2 with n/2 nodes



PURDUE UNIVERSITY

STyGIANet

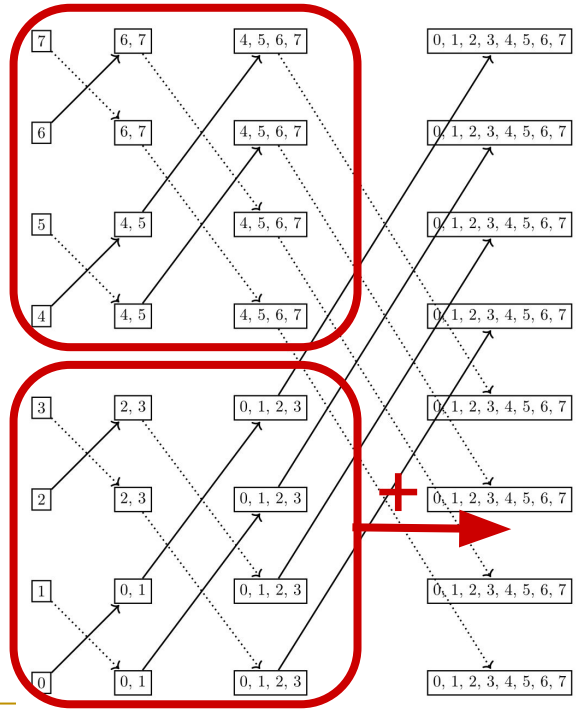
Butterfly AllGather (Recursive Doubling)

- Notice the recursive structure
- AllGather for message size m with n nodes can be expressed as a function of allgather for message size m/2 with n/2 nodes

$$T_{\text{allgather}}(m,n) = T_{\text{allgather}}\left(\frac{m}{2},\frac{n}{2}\right) + \alpha + \left(\frac{m}{2}\right)\cdot\beta$$

$$\approx \log_2(n)\cdot\alpha + \sum_{i=1}^{\log_2(n)} \frac{m}{2^i}\cdot\beta$$

$$\approx \log_2(n)\cdot\alpha + m\cdot\beta$$



PURDUE UNIVERSITY

STyGIANet

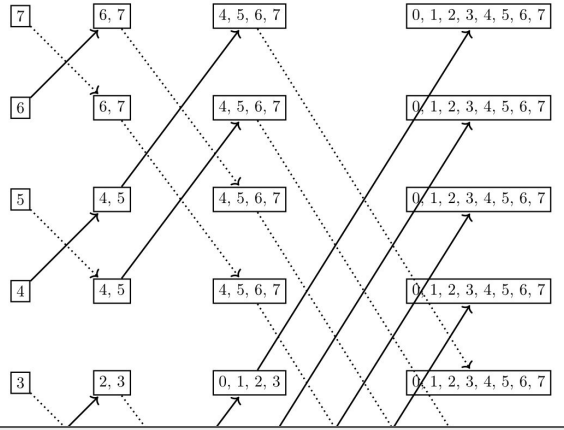
Butterfly AllGather (Recursive Doubling)

- Notice the recursive structure
- AllGather for message size m with n nodes can be expressed as a function of allgather for message size m/2 with n/2 nodes

$$T_{\text{allgather}}(m,n) = T_{\text{allgather}}\left(\frac{m}{2},\frac{n}{2}\right) + \alpha + \left(\frac{m}{2}\right)\cdot\beta$$

$$\approx \log_2(n)\cdot\alpha + \sum_{i=1}^{\log_2(n)} \frac{m}{2^i}\cdot\beta$$

$$\approx \log_2(n)\cdot\alpha + m\cdot\beta$$

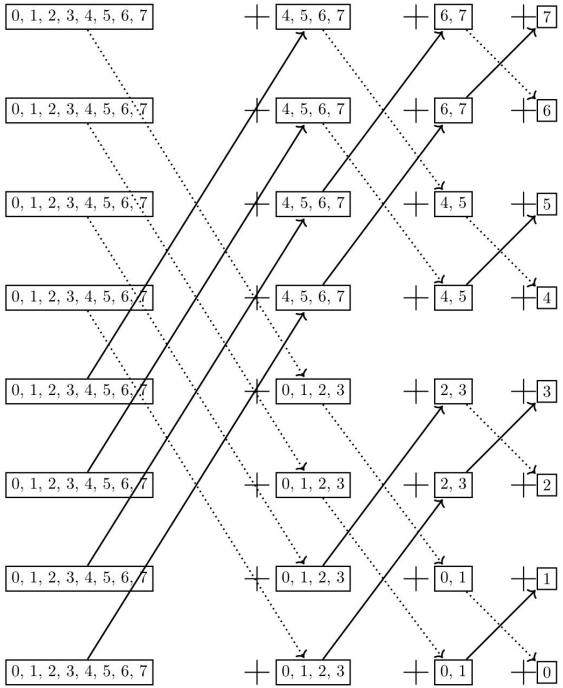


Linear in message size.
A characteristic of the geometric sum due to recursive doubling (or halving) of the message size

PURDUE UNIVERSITY
STyGIANet

Butterfly Reduce-Scatter

- Reverse of AllGather



PURDUE UNIVERSITY

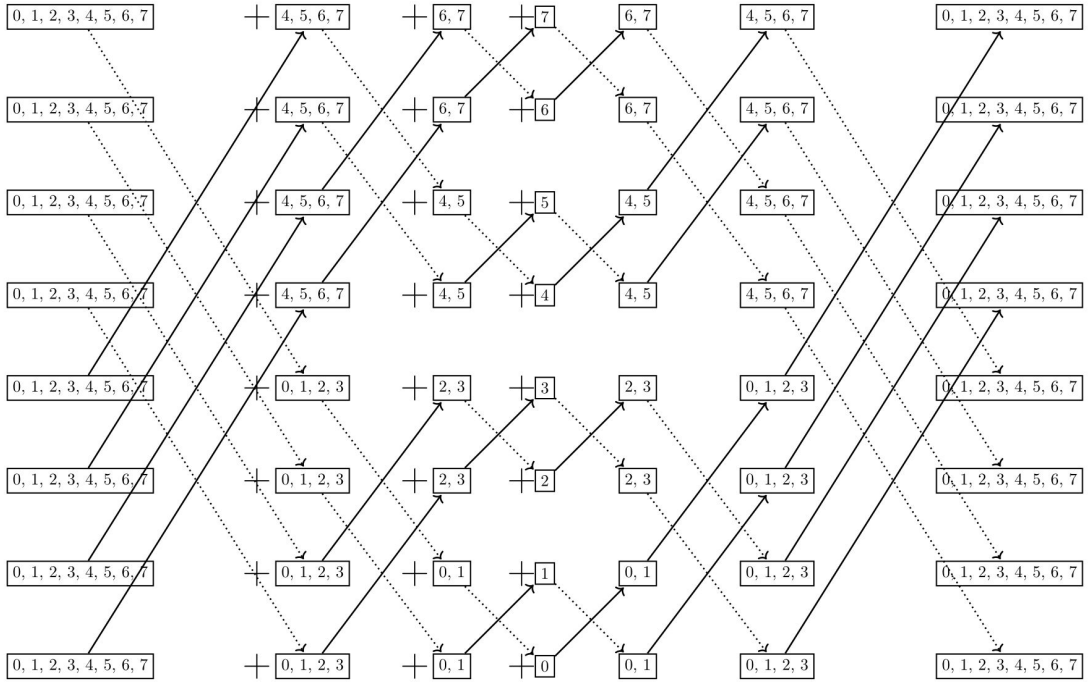
STyGIANet

Butterfly Reduce-Scatter

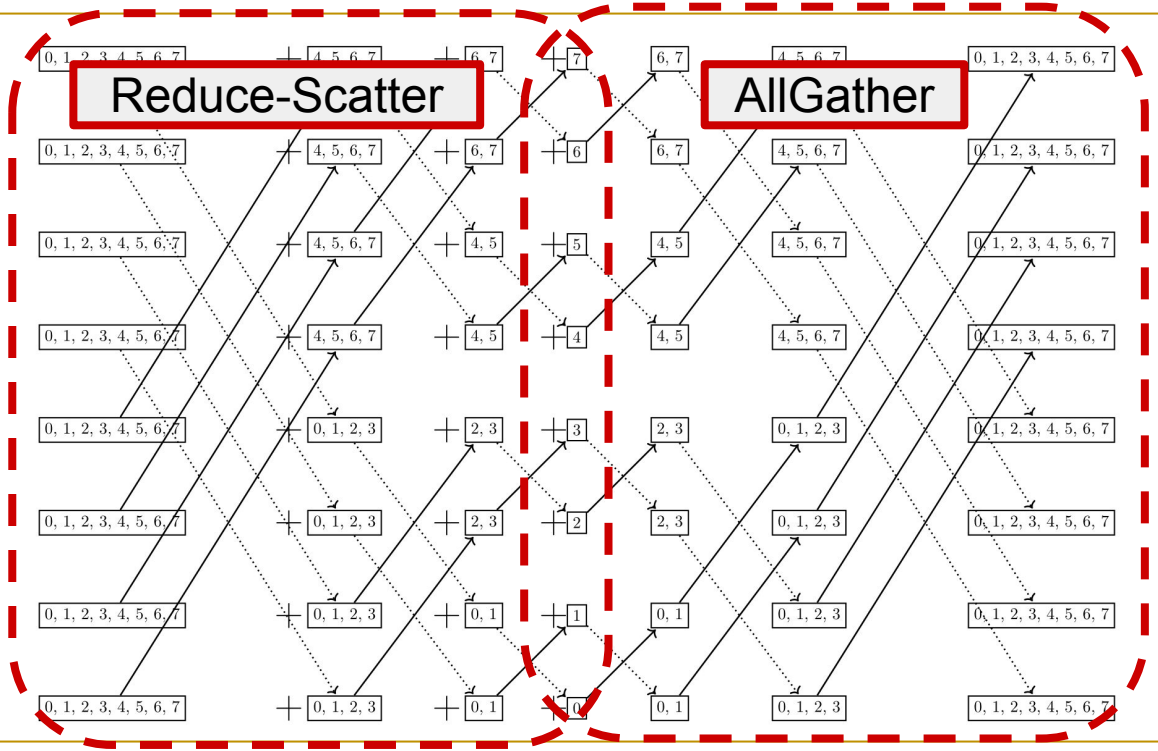
Reduce-Scatter: Running butterfly allgather in the reverse direction is equivalent to reduce-scatter (Recursive Halving). The communication cost is the same as allgather, with an additional cost for performing local reduction.

$$T_{Reduce-Scatter} = T_{AllGather} + cost\ of\ reduction$$

Butterfly AllReduce



Butterfly AllReduce



PURDUE UNIVERSITY

STyGIANet

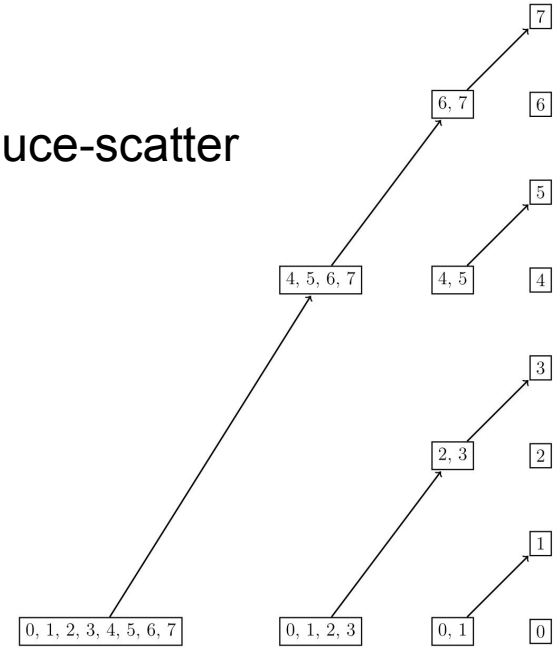
Butterfly AllReduce

AllReduce: Running reduce-scatter followed by allgather is equivalent to allreduce. The communication cost is the sum of reduce-scatter and allgather.

$$T_{AllReduce} = T_{Reduce-Scatter} + T_{AllGather}$$

Butterfly Scatter

- Recall the structure!
  - Binomial tree
- The cost is same as reduce-scatter (without reduction cost) = cost of AllGather



PURDUE UNIVERSITY

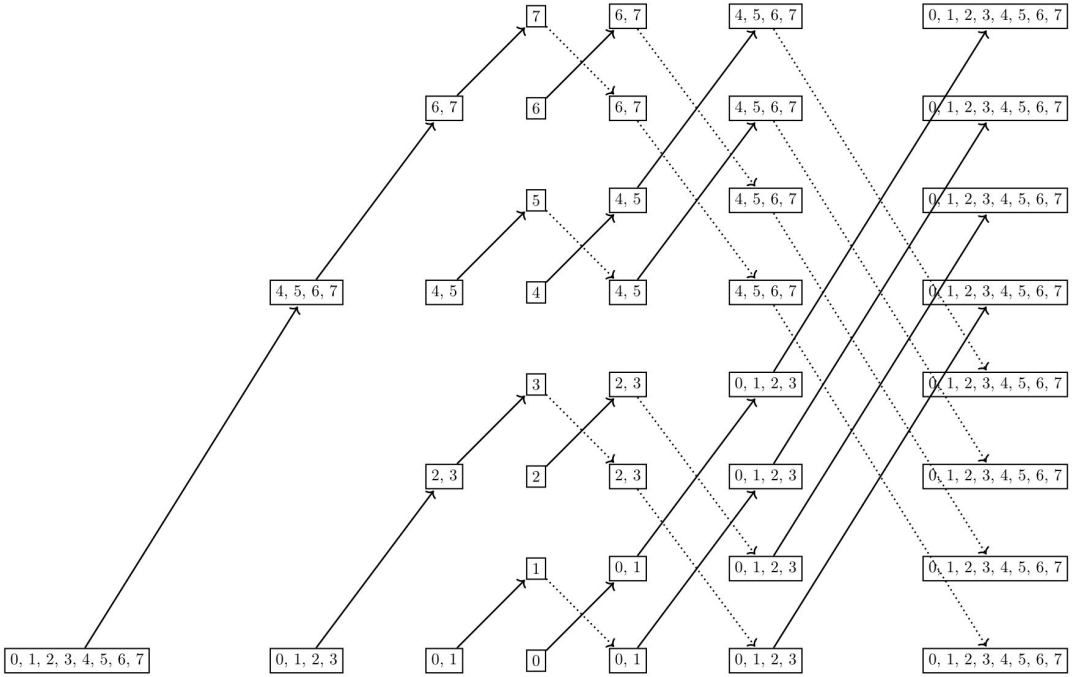
STyGIANet

Butterfly Scatter

Scatter: A single node has all the data. At each stage, the number of nodes sending data doubles. Notice that the butterfly structure of scatter operation is equivalent to a binomial tree.

$$T_{Scatter} = T_{AllGather}$$

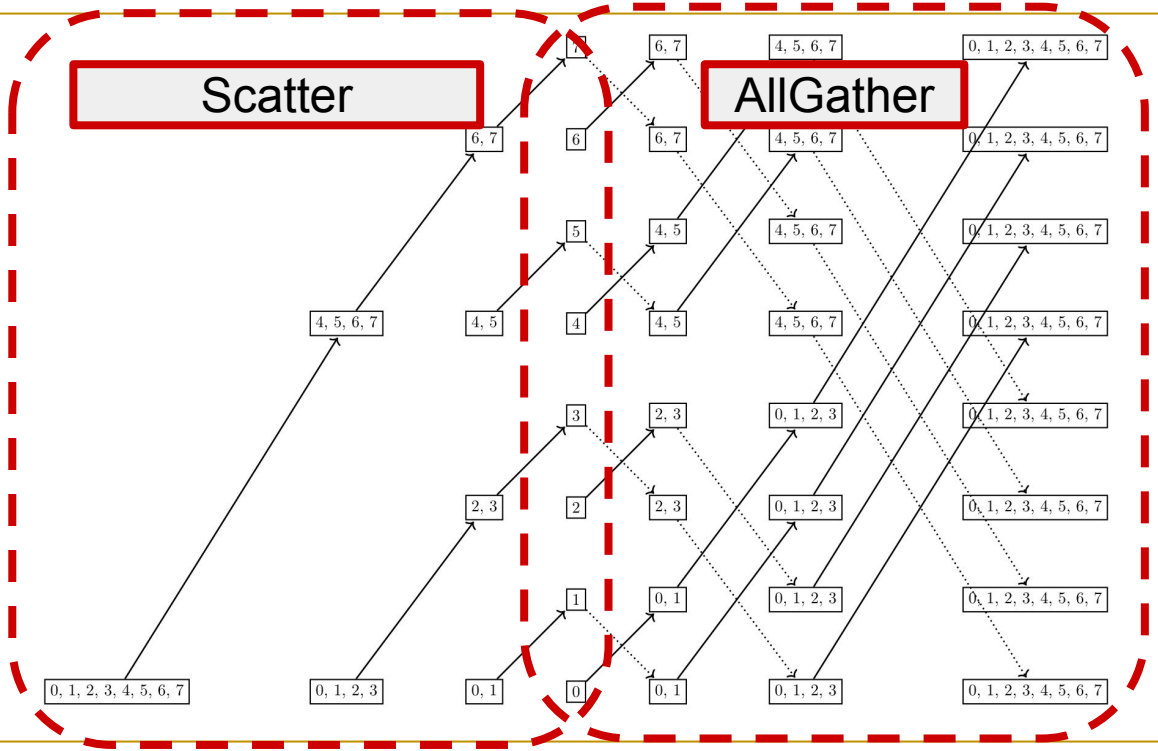
Butterfly Broadcast



PURDUE UNIVERSITY

STyGIANet

Butterfly Broadcast



PURDUE UNIVERSITY

STyGIANet

Butterfly Broadcast

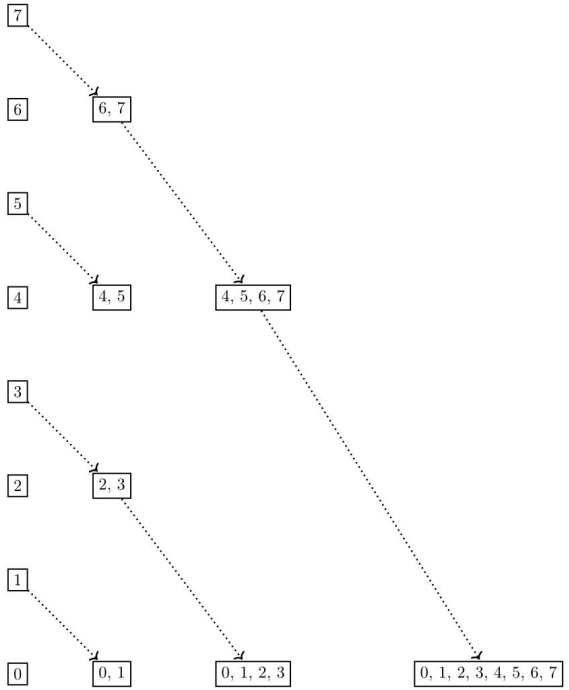
Broadcast: Running scatter, followed by allgather is equivalent to broadcast. The communication cost is the sum of scatter and allgather.

$$T_{Broadcast} = T_{Scatter} + T_{AllGather} = 2 \cdot log_2(n) \cdot \alpha + 2 \cdot m \cdot \beta$$

Bandwidth cost: Linear in message size!
log n improvement over binomial

Butterfly Gather

- Reverse of scatter



PURDUE UNIVERSITY

STyGIANet

Butterfly Gather

Gather: Running scatter in the reverse direction is equivalent to a gather operation. The communication cost is the same as scatter.

$$T_{Gather} = T_{Scatter}$$

Further Reading

[Collective communication: theory, practice, and experience]

[Synthesizing optimal collective algorithms]

[Optimization of Collective Communication Operations in MPICH]

[Swing: Short-cutting Rings for Higher Bandwidth Allreduce]

The End