DSC204A: Scaleable Data Systems Winter 2025

Lecture 21: Collective Communication Cont.

Instructor: Umesh Bellur *Scribe: Aaron Wu*

1 Key Symbols for Cost Analysis

- p: Number of processors.
- $\lceil \log(p) \rceil$: Depth of the communication tree (number of rounds) for dividing in half algorithms (MST)
- : (p-1): steps for ring algorithms passing along a ring topology
- α: Startup latency per message (communication initiation time).
- β: Transfer time per data item.
- γ: Time to perform one unit of computation/operation.
- n: Number of data elements per processor.

Broadcast

Broadcast is where a single node (the root) sends the same message to all other nodes in a network.

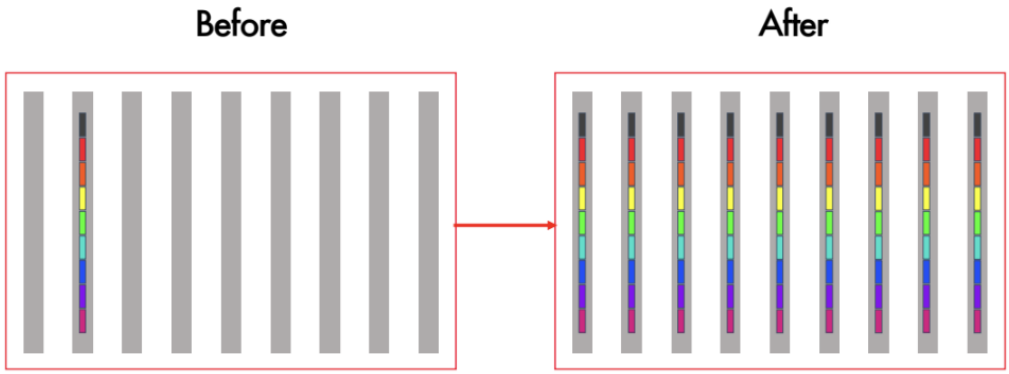

Figure 1: Broadcast Diagram

Minimum Spanning Tree (MST) Broadcast

The MST broadcast algorithm is latency-optimized, minimizing the number of message startup costs (α) at the expense of underutilizing available bandwidth.

Implementation

- From the broadcast node, divide logical linear array in half.
- Send message to the half of the network that does not contain the current node (root) that holds the message.
- Continue recursively in each of the two halves, until all nodes have the message.

The key advantage of MST is that it requires only $P - 1$ messages for P nodes, which minimizes the total number of sends.

In practice, MST-based broadcast is favored when α (latency per message) dominates over β (per-byte transmission cost), such as with small message sizes or high-latency networks.

Cost Analysis The cost of MST broadcast is given by:

$$T_{\text{broadcast}} = \lceil log(p) \rceil (\alpha + n\beta)$$

Ring Broadcast

In the ring-based broadcast algorithm, the message is split into P equal-sized chunks (one for each node), and the communication proceeds in a pipelined fashion around a logical ring of nodes.

Implementation

- The root node begins by dividing the message into P chunks: $m_0, m_1, \ldots, m_{P-1}$.
- In the first phase (**scatter**), the root sends chunk m_i to node i so that each node holds one distinct part of the message.
- In the second phase (**allgather**), each node sends its chunk to its neighbor and receives the next chunk from the opposite direction.
- After $P - 1$ rounds, each node will have received all the chunks and reconstructed the full message.

This method is efficient for large messages because the chunked, pipelined structure allows full utilization of the communication links in each round. However, it incurs higher startup latency compared to tree-based (MST) methods.

Cost Analysis The cost of broadcast is a combination of MST scatter and ring allgather:

$$T_{\text{broadcast}} = (\log(p) + p - 1)\alpha + 2\frac{p-1}{p}n\beta$$

Reduce (-to-one)

Reduce combines data from multiple nodes into a single result using a specified operation (e.g., sum, max).

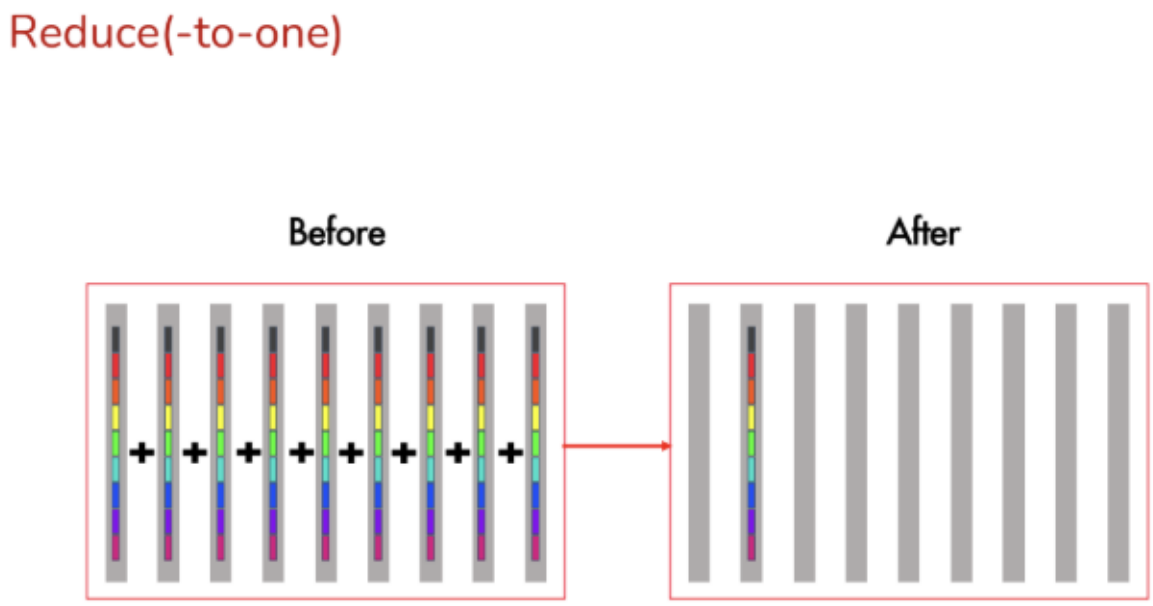

Figure 2: Reduce Diagram

1.1 MST Reduce

The MST-based reduce algorithm mirrors the structure of MST broadcast but in reverse.

Implementation

- The tree is traversed bottom-up: leaf nodes first send their data to their parent nodes.
- Each internal node waits to receive data from all of its children, applies the reduction operation (e.g., sum, max), combines the results with its own data, and sends the aggregate up to its parent.
- This continues recursively until the root node receives and reduces all data.

Cost Analysis For MST reduce:

$$T_{\text{reduce}} = \lceil \log(p) \rceil \, (\alpha + n\beta + n\gamma)$$

Ring Reduce

The ring-based reduce algorithm can be decomposed into two distinct phases: a **Reduce-Scatter** followed by a **Gather**.

Implementation

- The message is divided into P chunks of equal size.
- Each node holds its own input vector and participates in $P-1$ pipelined steps.
- In each step of the **reduce-scatter** phase:
  1. Each node sends one chunk to its neighbor in the ring.
  2. It receives a new chunk from its other neighbor.
  3. It applies the reduction operation (e.g., sum, max) to the received chunk and its local copy.
  4. It keeps the reduced result for the chunk it's responsible for and forwards others.
- After $P-1$ steps, each node holds a fully reduced chunk.
- In the subsequent **gather** phase:
  1. The reduced chunks are passed clockwise around the ring back to the designated root.
  2. Each node forwards the chunks it is not responsible for.
  3. The root assembles the full reduced message after $P-1$ steps.

Cost Analysis For ring reduce:

$$T_{\text{reduce}} = (p - 1 + \log(p))\alpha + \frac{p-1}{p} n(2\beta + \gamma)$$

Scatter

Scatter is the node of distributing different portions of a data set from a single source node to all other nodes in the network.

1.2 Minimum Spanning Tree (MST) Scatter

MST scatter divides network in half and sends node sends half of it's data to one node in the other half of the network. This is recursively done until each node in the network has a piece of the original data.

Implementation

- From the scatter node, divide logical linear array in half.
- Send **half** of the message to the half of the network that does not contain the current node (root) that holds the message.
- Continue recursively in each of the two halves, until all nodes have the $1/P$ message, where P is the number of nodes

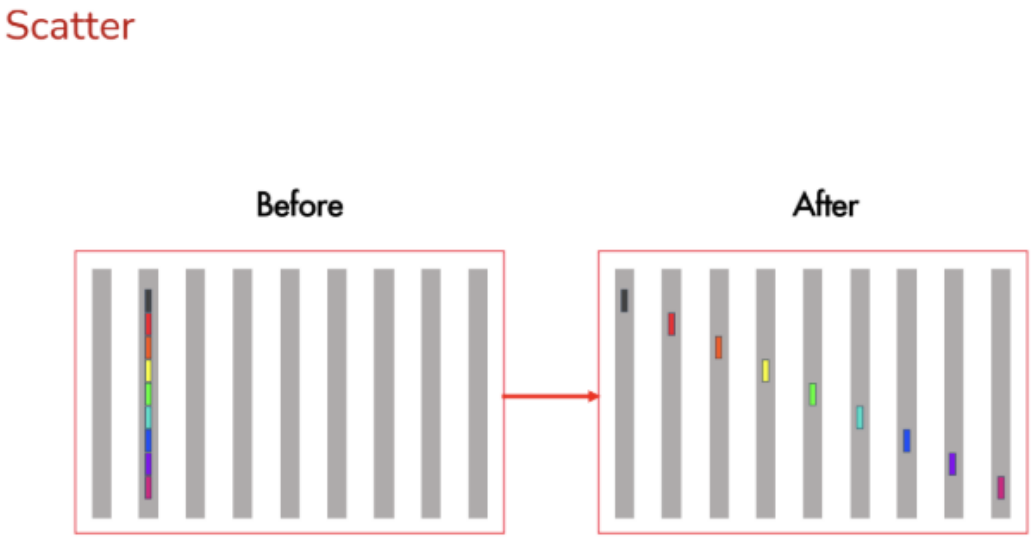

Figure 3: Scatter

Cost Analysis The cost of scatter using the MST approach is given by:

$$T_{\text{scatter}} = \sum_{k=1}^{\log(p)} \left(\alpha + \frac{n}{2^k}\beta\right) = \log(p)\,\alpha + \frac{p-1}{p} n\beta$$

1.3 Ring Scatter

Since MST is more optimal even in long vectors for scatter that is going to be used.

Gather

Gather is the inverse operation of scatter, where data from multiple nodes is collected into a single root node.

1.4 MST Gather

Implementation MST-based gather is the inverse of MST-based scatter.

- The network is logically divided in half, with one half sending data to the other at each step.
- Initially, each node holds a $1/P$ chunk of the total data.
- From the root, you call one half of the nodes to send you data.

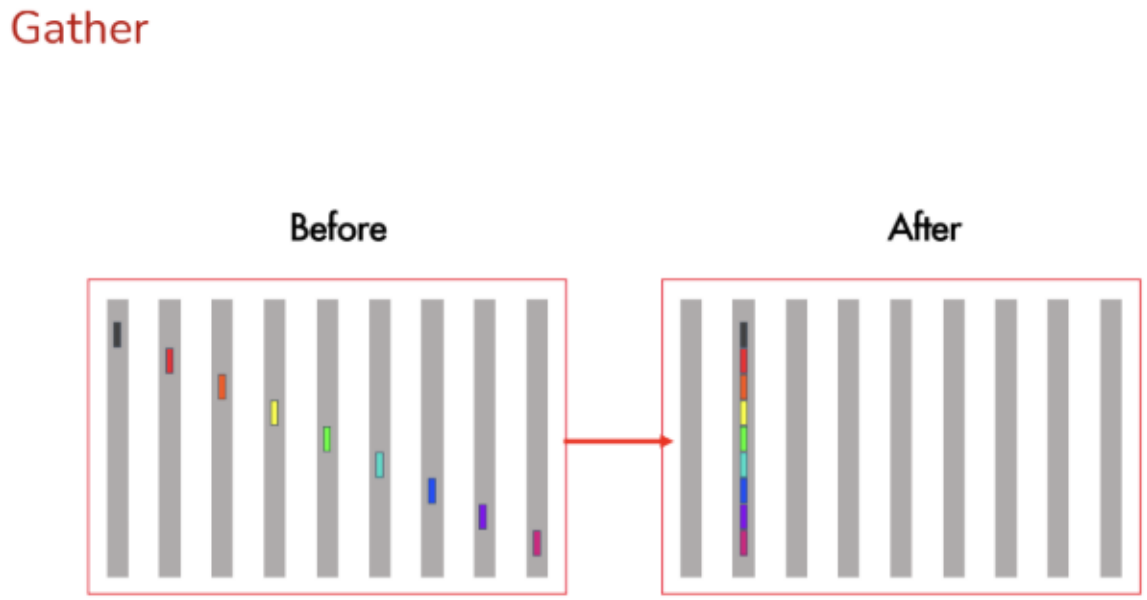

Figure 4: Gather Diagram

- This recursively repeats until a leaf node is hit.
- Then each of $\log P$ steps, every participating node sends its data to the caller node.
- The receiving node combines the incoming data with its own, accumulating a larger portion of the final message.
- This continues recursively until all data has been gathered at the root node.

Cost Analysis Similar to scatter, the cost for MST gather is:

- Assumption: power of two number of nodes

$$T_{\text{gather}} = \sum_{k=1}^{\log(p)} \left(\alpha + \frac{n}{2^k}\beta\right) = \log(p)\,\alpha + \frac{p-1}{p} n\beta$$

1.5 Ring Gather

Since MST is more optimal even in long vectors for gather that is going to be used.

Allgather

Allgather is a combination of gather and broadcast, where all nodes receive the final reduced result.

1.6 MST Allgather

Similar to MST gather but now the data is aggregated on each node.

Implementation

- The node group is recursively divided into two halves.
- At each recursion level, every node exchanges data with a partner in the opposite half of the range.
- Each node first recursively gathers data from its own half.
- After returning from the recursive call, it exchanges data with its partner in the other half. This doubles the amount of data each node holds.
- Nodes in the lower half send their local data to their partners in the upper half, and receive the partner's data in return. The upper half does the same, sending and receiving in the opposite direction.
- After $\log_2 P$ steps (where P is the number of nodes), each node has received and merged all chunks from every other node.

Cost Analysis The cost of allgather is gather + broadcast

- Assumption: power of two number of nodes

$$\begin{aligned} T_{\text{allgather}} &= T_{\text{gather}} + T_{\text{broadcast}} \\ &= \left(\log(p)\alpha + \frac{p-1}{p}n\beta\right) + (\log(p)\alpha + \log(p)n\beta) \\ &= 2\log(p)\alpha + \left(\frac{p-1}{p} + \log(p)\right)n\beta \end{aligned}$$

1.7 Ring Allgather

Cost Analysis

$$\underbrace{(p-1)}_{\text{number of steps}}\underbrace{\left(\alpha + \frac{n}{p}\beta\right)}_{\text{cost per steps}} = (p-1)\alpha + \frac{p-1}{p}n\beta$$

Note: number of steps in ring is $p-1$ and the cost per step includes n/p

1.8 BDE Allgather

- The set of node is logically divided into two halves.
- Each node identifies a unique partner in the opposite half. This partner is chosen such that the two nodes will exchange the portions of the message they currently hold.
- Before the exchange, each half recursively performs AllGather within itself. This ensures that by the time cross-group communication occurs, each node already has the data from its half.
- After the recursive calls, each node sends its known data to its partner and receives the missing portion in return.
- After the exchange, both node now hold twice as much of the full message as before.
- After $\log_2 P$ rounds, each node has the full message

Cost Analysis

$$T_{\text{BDEAllGather}}(p,n) = \sum_{k=1}^{\log(p)} \left(\alpha_3 + 2^{-k} n\beta_1\right) = \log(p)\alpha_3 + \frac{p-1}{p} n\beta_1$$

Reduce-Scatter

Reduce-scatter is a hybrid operation where each node reduces its data and then scatters the reduced results to all other nodes.

1.9 MST reduce-scatter

Implementation

- First follow **MST Reduce**
- Then follow **MST Scatter**

Cost Analysis The cost of reduce-scatter is a combination of MST reduce + MST scatter

$$\begin{aligned} T_{\text{reduce-scatter}} &= \underbrace{\log(p)(\alpha + n\beta + n\gamma)}_{\text{Reduce-to-one}} + \underbrace{\log(p)\alpha + \frac{p-1}{p} n\beta}_{\text{Scatter}} \\ &= 2\log(p)\alpha + \left(\frac{p-1}{p} + \log(p)\right) n\beta + \log(p) n\gamma \end{aligned}$$

1.10 Ring reduce-scatter

Implementation

- Each node starts with the full input vector, partitioned into P chunks.

- In each of the $P-1$ steps:
  - Each node sends the appropriate chunk to its neighbor in the ring and receives a new chunk from the other neighbor.
  - It applies the reduction operation to the received chunk and its local copy of the corresponding chunk.
  - It keeps the chunk it is ultimately responsible for and continues forwarding the others.
- After $P-1$ steps, each node ends up with the fully reduced result for its assigned chunk.

The cost of ring reduce-scatter:

$$T_{\text{reduce-scatter}} = (p-1)\alpha + \frac{p-1}{p}n(\beta+\gamma)$$

For each node it keeps $1/n$ of the message and passes it along the ring.

1.11 BDE reduce-scatter

- The set of nodes is logically divided in half at each recursive level.
- Each node determines its communication **partner** in the opposite half by offsetting its rank by half the size of the current range.
- Nodes in the left half send the right half of their data to their partners, and receive the right half from their partners to reduce into their own right portion.
- Similarly, nodes in the right half send the left half and receive the left half for reduction.
- The received chunks are element-wise reduced (e.g., summed) into the corresponding portion of the local vector.
- After the data exchange and local reduction, the nodes recurse only into the half of the range where they belong.

Cost Analysis The cost of BDE reduce-scatter:

$$T_{\text{reduce-scatter}} = (\log(p))\alpha + \frac{p-1}{p}n(\beta+\gamma)$$

Allreduce

Allreduce is a combination of reduce and broadcast, where all nodes receive the final reduced result.

1.12 MST Allreduce

Implementation

- First follow **MST Reduce-to-one**
- Then follow **MST Broadcast**

Cost Analysis Allreduce cost is a combination of MST reduce + MST broadcast:

$$\begin{aligned} T_{\text{allreduce}} &= \underbrace{\log(p)(\alpha + n\beta + n\gamma)}_{\text{Reduce-to-one}} + \underbrace{\log(p)(\alpha + n\beta)}_{\text{Broadcast}} \\ &= 2\log(p)\alpha + 2\log(p)n\beta + \log(p)n\gamma \end{aligned}$$

1.13 Ring Allreduce

Implementation

1. First follow **Ring Reduce-Scatter**
2. The follow **Ring AllGather**

Cost Analysis A combination of ring reduce-scatter and ring allgather:

$$T_{\text{allreduce}} = 2(p-1)\alpha + \frac{p-1}{p}n(2\beta + \gamma)$$

2 MST vs Ring

MST: for short vectors or small amounts of data

Pros:

- emphasizes low latency, alpha component dominates, when amount of data isn't that big

Cons:

- Some links are idle
- prioritize latency rather than bandwidth

Ring: for long vectors or large message sizes

Pros:

- emphasizes bandwidth, when $n * \beta$ dominates

Cons:

- prioritize bandwidth rather than latency

3 2D Broadcast

Choose one dimension to run an operation then run another operation on the other dimension.

- One example would be MST broadcast to columns then MST broadcast to rows.
- Another example is MST scatter across columns, MST broadcast across rows, MST allgather rows, and then MST allgather columns

Extensible to other dimensions by using the primitives generated by the dimensions below.