CS 168
Spring 2025

Multicast and Collectives

Discussion 12

1 True or False

1.1 IP multicast is widely deployed across different domains for global Internet communication.

False: IP multicast is primarily used within a single domain, such as enterprise networks or ISPs, and is rarely deployed across multiple autonomous systems due to challenges like scalability, billing disputes, and inter-domain coordination required for protocols like PIM and MSDP.

1.2 In the DVMRP protocol, the multicast forwarding table requires one entry per source, per multicast group.

True: DVMRP builds a source-specific shortest-path tree for each (source, multicast group) pair, requiring multicast forwarding table entries that specify the Reverse Path Forwarding (RPF) interface and outgoing interfaces (children) for each pair, updated via flood-and-prune mechanisms.

1.3 Core-Based Trees (CBT) guarantee that packets are forwarded along the least-cost paths to all group members.

False: CBT uses a single shared tree per multicast group rooted at a core router, prioritizing scalability over optimality. Paths depend on the core's location and may be suboptimal compared to shortest unicast paths, unlike source-specific trees in protocols like DVMRP.

1.4 Overlay multicast requires routers to implement specialized multicast protocols, unlike IP multicast.

False: Overlay multicast operates at the application layer, with end hosts or proxy servers managing multicast routing using unicast packets. Routers only need standard unicast protocols, unlike IP multicast, which requires specialized protocols like PIM or DVMRP. Examples include P2P streaming or ALM protocols like Narada.

1.5 The AllReduce collective operation can be implemented efficiently using a ring topology, where each node sends and receives data to/from its neighbors.

True: Ring-based AllReduce involves nodes exchanging partial sums in a logical ring, completing in 2(n-1) steps (reduce and broadcast phases) for n nodes. This reduces bandwidth compared to full-mesh approaches, though performance depends on the logical ring's mapping to the physical network. It is widely used in MPI and AI training frameworks like Horovod.

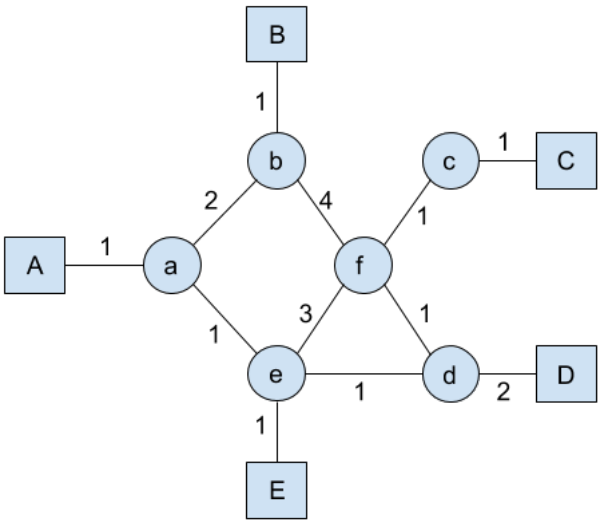
1.6 The stretch factor in overlay multicast measures the ratio of the overlay path cost to the underlay path cost, with lower stretch indicating better performance.

True: The stretch factor is the ratio of the overlay path cost (latency or hops in the overlay tree) to the underlay path cost (optimal unicast path). A lower stretch, closer to 1, indicates that the overlay topology closely matches the underlay, reducing latency and improving packet forwarding performance.

1.7 Collective operations in AI training, such as Broadcast and Reduce, are unrelated and cannot be combined to form other operations like AllReduce.

False: Broadcast and Reduce are complementary operations in collective communication. AllReduce can be implemented as a Reduce phase (aggregating data, e.g., summing gradients) followed by a Broadcast phase (distributing the result), commonly using tree-based algorithms in frameworks like MPI or Horovod.

2 Multicast



Consider the network topology above for both parts of this question. Hosts are squares with capital letters, routers are circles with lower case letters. Link cost is latency in seconds.

DVMRP

All parts of this section are cumulative. At the start, A and E are members of group 1.

2.1 (a) A sends a multicast packet to Group 1. After all Non-Membership Reports (NMRs) have propagated, which links remain active (i.e., used for forwarding packets) in the source-specific multicast tree for (Source A, Group 1)?

A-a, a-e, e-E

(b) Suppose C adds itself to group 1. Immediately after this, who knows C joined group 1?

c. Only the first hop router is notified at first.

(c) E sends a message to group 1. Which links remain active for (Source E, Group 1)?

A-a, a-e, e-E, d-e, d-f, c-f, c-C

(d) B sends a message to group 1, is this possible?

Yes, non-members can send messages to a group.

(e) What is the route that the packet takes from B to E?

B-b-a-e-E

(f) How long does it take for B's message to reach all members of group 1?

7 seconds, the path to C (B-b-f-c-C) is the longest.

(g) Suppose that the pruning information expires. A, B, and D form a new group 2. A then sends messages to groups 1 and 2, B sends a message to group 2, and D sends a message to group 1. What state does router f have? (Give a list of (Source x, Group y)).

(A, 1), (D, 1)

(h) What message(s) took the longest to send?

$B \rightarrow D$ took 7 seconds.

(i) What message(s) took the least time to send?

$A \rightarrow E$ took only 3 seconds.

CBT

All parts of this section are cumulative, but independent from the previous section. a is the core for group 1.

2.2 (a) C decides to join group 1. What routers does its request go through?

c, f, d, e, a

(b) D decides to join group 1. What routers does its request go through?

None

(c) B sends a message to group 1. How long does it take to reach all members?

8 seconds. 3 seconds to reach a from B, and then another 5 to reach C from a (reaching D only took 4 seconds).

(d) C sends a message to group 1. How long does it take to reach all members?

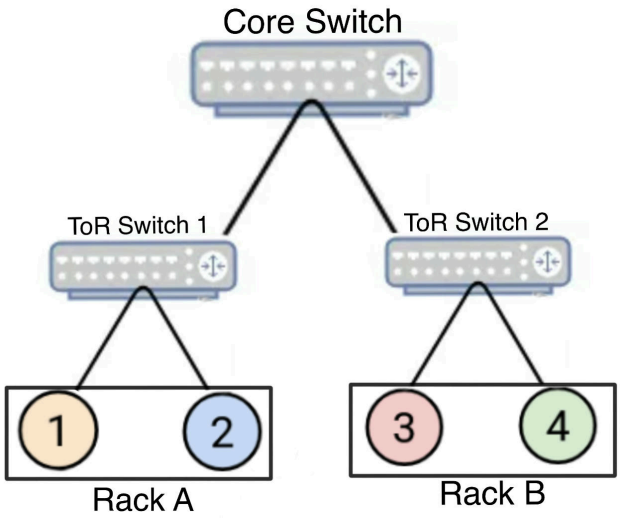
5 seconds. C $\rightarrow$ c $\rightarrow$ f $\rightarrow$ d $\rightarrow$ D.

(e) The link between A and a goes down, but no other changes in control plane state occur. Who can still send/receive messages to/from group 1, if any?

B, C, D, E can send messages to Group 1. C, D can receive messages from Group 1. A Cannot send or receive.

3 Collective

You are designing a distributed AI training system in a datacenter with 4 GPUs (nodes 1, 2, 3, 4) performing collective operations. Each node holds a 4-element vector of data (total size D bytes). The datacenter uses a Clos topology, with nodes 1 and 2 in Rack A, and nodes 3 and 4 in Rack B, connected via a core switch. Intra-rack links (e.g., 1↔2) have 1 hop (stretch = 1); inter-rack links (e.g., 1↔3) have 3 hops (stretch = 3). Assume intra-rack links have unlimited bandwidth, inter-rack links have a bandwidth capacity of B bytes per time step, and nodes can send/receive on all links simultaneously unless otherwise specified.



The Clos topology has nodes 1 and 2 in Rack A, nodes 3 and 4 in Rack B, connected via a core switch. Intra-rack links have 1 hop (stretch = 1); inter-rack links have 3 hops (stretch = 3).

Part A: Full-Mesh AllReduce Analysis

Consider an AllReduce operation where each node computes the element-wise sum of all vectors and receives the sum vector.

3.1 (a) Explain why a full-mesh topology is suitable for AllReduce in AI training, and discuss its scalability limitations for large p.

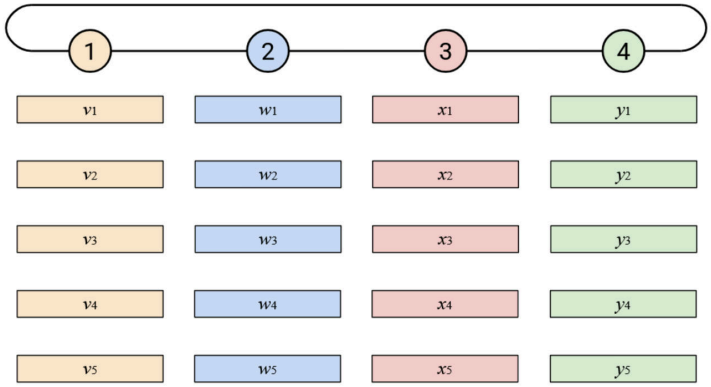
A full-mesh topology enables each node to send its vector directly to all other nodes simultaneously, minimizing latency to one time step, which is ideal for AI training's predictable, high-bandwidth needs. However, it requires $O(p^2)$ links, making it impractical for large p due to high wiring complexity and resource demands.

(b) Calculate the total bandwidth and number of steps for a full-mesh AllReduce with $p = 4$. If inter-rack link bandwidth is limited to $B = D$ bytes per time step and intra-rack links have unlimited bandwidth, how many time steps are required?

Each node sends its vector (D bytes) to $p - 1 = 3$ other nodes, so each node sends $D \cdot 3$ bytes. With $p = 4$, total bandwidth is $4 \cdot D \cdot 3 = 12D$ bytes, completed in 1 step without bandwidth limits. With inter-rack link bandwidth limited to $B = D$ bytes per step and intra-rack links unlimited, each node sends to 1 intra-rack node (D bytes, 1 step) and 2 inter-rack nodes (D bytes each, 1 step each). Since nodes can send on all links simultaneously, the operation completes in 1 step.

Part B: Naive Ring AllReduce

Assume a naive ring-based AllReduce (e.g., Node 1 ⟶ Node 2 ⟶ Node 3 ⟶ Node 4 ⟶ Node 1).



The ring topology connects nodes 1⟶2⟶3⟶4⟶1 in a closed loop.

3.2 (a) How many steps are required for a naive ring AllReduce with $p = 4$? Describe the data exchange process.

The naive ring AllReduce requires $2(p-1) = 2(4-1) = 6$ steps. The process consists of two phases, each taking $p - 1 = 3$ steps:

- **Reduction Phase (Steps 1–3):** In each step, each node sends its current vector (D bytes) to its neighbor (e.g., Node 1 sends to Node 2, Node 2 to Node 3, etc.) and receives a vector from its other neighbor (e.g., Node 1 receives from Node 4). The received vector is added element-wise to the node's current vector. After $p - 1 = 3$ steps, each node has computed the sum of all vectors (e.g., Node 1 has $v_1 + w_1 + x_1 + y_1$).
- **Distribution Phase (Steps 4–6):** Each node now sends the computed sum to its neighbor in the same manner. After another $p - 1 = 3$ steps, all nodes have received the same final sum, completing the AllReduce operation.

(b) Calculate the total bandwidth and bandwidth per node per step. If inter-rack links are limited to $B = \frac{D}{2}$ bytes per step, how does this affect the number of steps?

Each node sends one vector (D bytes) per step. The full AllReduce takes $2(p-1) = 6$ steps with $p = 4$ nodes, so the total bandwidth (bytes sent) is $4 \cdot D \cdot 6 = 24D$ bytes (4 nodes send D bytes each step for 6 steps). The bandwidth per node per step is D bytes (sent only).

With inter-rack links limited to $B = \frac{D}{2}$ bytes per step, sending D bytes over an inter-rack link (e.g., 2⟶3) takes $\frac{D}{\frac{D}{2}} = 2$ steps. The ring has 2 intra-rack links (1⟶2, 3⟶4, 1 step each) and 2 inter-rack links (2⟶3, 4⟶1, 2 steps each). For one vector to traverse the ring once (one phase), the steps are: 1 (1⟶2) + 2 (2⟶3) + 1 (3⟶4) + 2 (4⟶1) = 6 steps. Since the AllReduce has two phases (reduction and distribution), the total steps increase to $6 \times 2 = 12$ steps.

Part C: Optimized Ring vs. Tree-Based AllReduce

Compare naive ring, optimized ring, and tree-based AllReduce implementations for $p = 16$.

3.3 (a) For an optimized ring AllReduce, describe how it differs from the naive ring and calculate its total bandwidth and steps for $p = 16$.

The optimized ring performs Reduce-Scatter (each node gets one summed element) followed by AllGather (each node gets all summed elements), using $p - 1 = 15$ steps per phase, totaling $2(p - 1) = 30$ steps. Each node sends/receives $\frac{D}{p} = \frac{D}{16}$ bytes per step, unlike the naive ring, which sends the entire vector (D bytes) per step over two phases (reduction and distribution, $2(p - 1) = 30$ steps). For $p = 16$, each node sends $(p - 1) \cdot \frac{D}{p} \cdot 2 = 15 \cdot \frac{D}{16} \cdot 2 = 15\frac{D}{8}$ bytes over both phases. Total bandwidth is $p \cdot \left(15\frac{D}{8}\right) = 16 \cdot 15\frac{D}{8} = 30D$ bytes, and steps are $2(p - 1) = 30$ (15 for Reduce-Scatter, 15 for AllGather).

(b) For a tree-based AllReduce, calculate the total bandwidth and steps for $p = 16$, assuming a balanced binary tree.

In a tree-based AllReduce, leaf nodes send vectors up to the root, which computes the sum, then sends it down. For $p = 16$, a balanced binary tree has height $\log_2(16) = 4$. Each edge carries D bytes up and down, and there are $p - 1 = 15$ edges, so total bandwidth is $2 \cdot (p - 1) \cdot D = 2 \cdot 15 \cdot D = 30D$ bytes. Steps are $2 \cdot \log_2(p) = 8$ (4 up, 4 down).

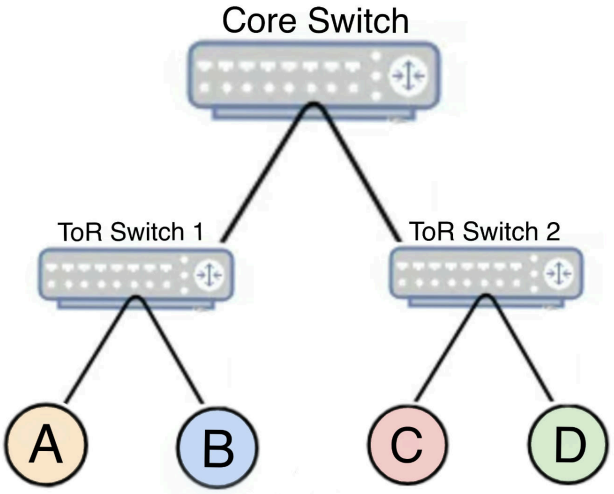
(c) Compare the three implementations (naive ring, optimized ring, tree-based) for $p = 16$, discussing trade-offs for AI training.

- **Naive Ring**: Bandwidth = $p \cdot 2D \cdot 2(p - 1) = 16 \cdot 2D \cdot 30 = 960D$ bytes, steps = $2(p - 1) = 30$. Simple but slow due to high bandwidth per step ($2D$) and sequential exchanges.
- **Optimized Ring**: Bandwidth = $30D$ bytes, steps = 30. More steps but lower bandwidth per step ($\frac{D}{16}$), better for bandwidth-constrained links and scalability.
- **Tree-Based**: Bandwidth = $30D$ bytes, steps = 8. Fastest due to fewer steps, but the root can be a bottleneck and tree construction may be complex.

For AI training, tree-based is preferred for low latency (fewer steps), but optimized ring is better for scalability and fault tolerance in large systems due to lower bandwidth per step and no single point of failure.

Part D: Overlay Topology Optimization

Optimize an overlay ring for AllReduce over the Clos network, with nodes A, B in Rack A, and C, D in Rack B.



Nodes A and B are in Rack A, nodes C and D are in Rack B, with intra-rack links (stretch = 1) and inter-rack links (stretch = 3).

3.4 (a) Propose an overlay ring topology by numbering nodes (A, B, C, D) as Nodes 1, 2, 3, 4 to minimize the average stretch factor. Calculate the stretch for each link and the average.

Number nodes as: Node 1 = A, Node 2 = B, Node 3 = C, Node 4 = D, forming A⟶B⟶C⟶D⟶A. Stretches: A⟶B (1, intra-rack), B⟶C (3, inter-rack), C⟶D (1, intra-rack), D⟶A (3, inter-rack). Average stretch = $\frac{1+3+1+3}{4} = 2$.

(b) Consider two alternative topologies: (i) A⟶C⟶D⟶B⟶A, and (ii) A⟶C⟶B⟶D⟶A. Calculate their average stretches and compare to your proposed topology. If inter-rack links have bandwidth $B = \frac{D}{4}$ bytes per step, which topology minimizes total latency for an optimized ring AllReduce, assuming each hop adds 1 time unit of latency?

- **Alternative (i) A⟶C⟶D⟶B⟶A**: Stretches: A⟶C (3, inter-rack), C⟶D (1, intra-rack), D⟶B (3, inter-rack), B⟶A (1, intra-rack). Average = $\frac{3+1+3+1}{4} = 2$.
- **Alternative (ii) A⟶C⟶B⟶D⟶A**: Stretches: A⟶C (3, inter-rack), C⟶B (3, inter-rack), B⟶D (3, inter-rack), D⟶A (3, inter-rack). Average = $\frac{3+3+3+3}{4} = 3$.
- **Comparison**: The proposed topology (A⟶B⟶C⟶D⟶A) and alternative (i) both have stretch = 2, alternating intra- and inter-rack links, while alternative (ii) has higher stretch = 3, using only inter-rack links, making it less efficient.

- **Latency Analysis**: For an optimized ring AllReduce ($p = 4$), total steps are $2 \cdot (p - 1) = 6$ (3 for ReduceScatter, 3 for AllGather). Each step sends $\frac{D}{p} = \frac{D}{4}$ bytes. With inter-rack bandwidth $B = \frac{D}{4}$, intra-rack links (unlimited) send $\frac{D}{4}$ in 1 step (1 hop = 1 time unit), and inter-rack links send $\frac{D}{4}$ in $\frac{\frac{D}{4}}{\frac{D}{4}} = 1$ step (3 hops = 3 time units). For A⟶B⟶C⟶D⟶A (links: intra, inter, intra, inter), steps are: 1 (1 hop), 3 (3 hops), 1 (1 hop), 3 (3 hops), 1 (1 hop), 3 (3 hops), total latency = $1 + 3 + 1 + 3 + 1 + 3 = 12$ time units. Alternative (i) has the same sequence (intra, inter, intra, inter), yielding 12 time units. Alternative (ii) (all inter-rack) has steps: 3, 3, 3, 3, 3, 3, total = 18 time units. The proposed topology and alternative (i) minimize latency equally; the proposed topology is chosen for its sequential ordering (A, B in Rack A; C, D in Rack B), simplifying configuration.

Part E: ReduceScatter Implementation

Consider a ReduceScatter operation, where each node receives one summed element of the AllReduce output.

3.5 (a) Explain how ReduceScatter differs from AllReduce and how it can be implemented using a naive ring for $p = 4$. Calculate the total bandwidth and steps.

ReduceScatter sums each element across nodes, with node i receiving the ith summed element, unlike AllReduce, where all nodes receive the full summed vector. For a naive ring ($p = 4$), each node sends its entire vector (D bytes) to its neighbor in each of $p - 1 = 3$ steps, accumulating the ith element's sum at node i. Total bandwidth = $D \cdot (p - 1) \cdot p = 3 \cdot 4 \cdot D = 12D$ bytes (each of 4 nodes sends D bytes for 3 steps). Steps = 3.

(b) If you implement ReduceScatter using the optimized ring topology from Part D.1 (A⟶B⟶C⟶D⟶A) with inter-rack bandwidth $B = \frac{D}{4}$, calculate the total latency (in steps) and compare to the naive ring.

For an optimized ring ReduceScatter ($p = 4$), only the ReduceScatter phase is needed ($p - 1 = 3$ steps), sending $\frac{D}{p} = \frac{D}{4}$ bytes per step. With inter-rack bandwidth $B = \frac{D}{4}$, intra-rack links (unlimited) send $\frac{D}{4}$ in 1 step, and inter-rack links (e.g., B⟶C, D⟶A) send $\frac{D}{4}$ in $\frac{\frac{D}{4}}{\frac{D}{4}} = 1$ step. The ring (A⟶B⟶C⟶D⟶A) has 2 intra-rack (1⟶2, 3⟶4) and 1 inter-rack (2⟶3) links in 3 steps, yielding 3 steps total. The naive ring also takes 3 steps but sends D bytes per step, increasing congestion. The optimized ring is more efficient due to lower bandwidth per step ($\frac{D}{4}$ vs. D), reducing congestion in bandwidth-constrained settings.