Collective Communication Optimization(CCO): Use cases, Requirements and Analysis

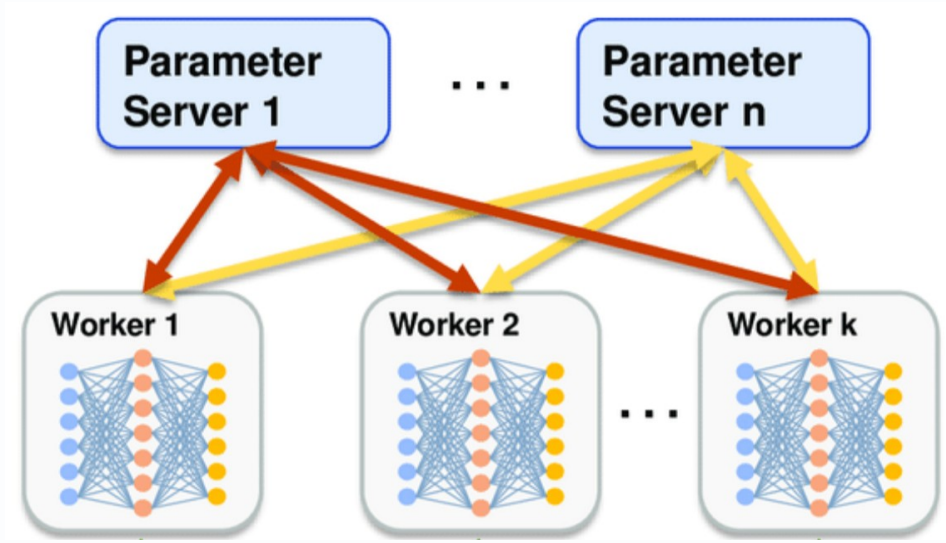
draft-yao-tsvwg-cco-problem-statement-and-usecases-00

draft-yao-tsvwg-cco-requirement-and-analysis-01

Prensenter: Kehan Yao(China Mobile)

Background: Use cases & Problems

Distributed AI Model Training



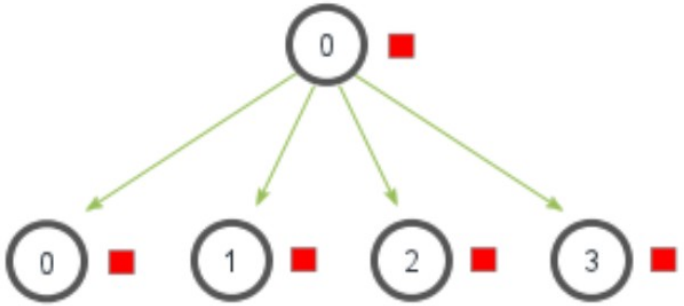
Problems

- **Incast & Large JCT:** single flow(AI traffic) could be around 10s of GB, bad data sychronization results in very large tail latency and straggler problem.
- **Low Compute & Communication overlap:** workers(i.e., GPUs) usually need to wait for network traffic, a lot of idle percentage.
- **Less efficient data distribution:** blunt data distribution, easy but frequent operations all left to endpoints to implement.
- **Scale-up & Scale-out traffic:** both have resource constraint problems for three types of parallel policies(i.e., data parallel, tensor parallel, pipeling parallel)

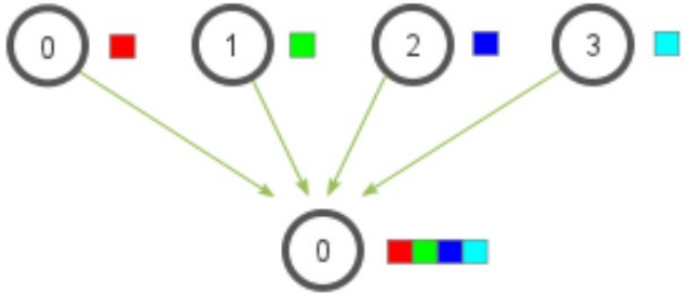
Collective Communications

- Collective communication: A set of communication patterns that application processes follow to communicate with each other in a parallel computing cluster. These patterns include One-to-Many, Many-to-one, or Many-to-Many delivery mode.

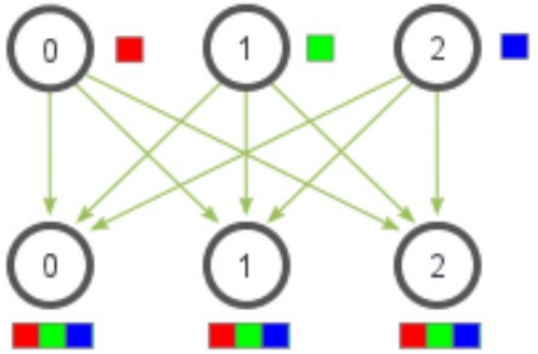
Collective Broadcast()



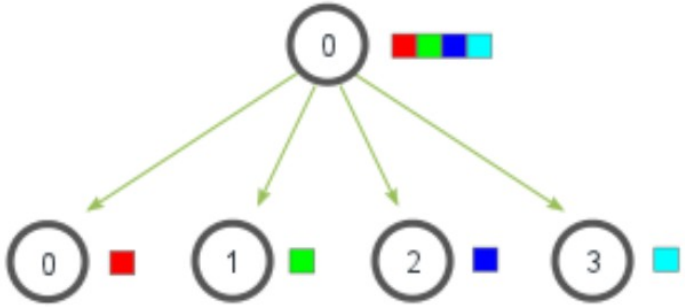
Collective Gather()



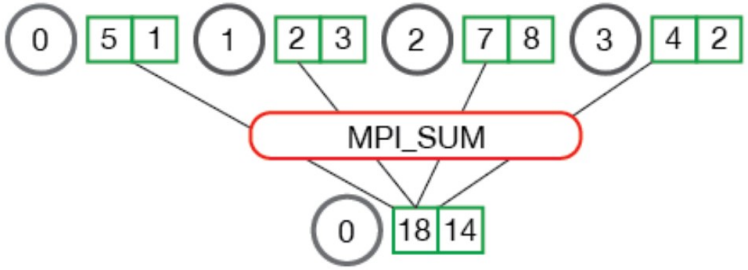
Collective Allgather()



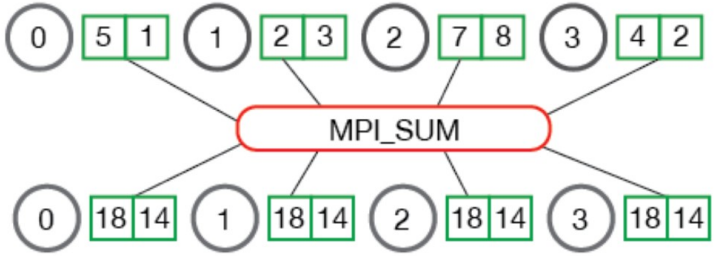
Collective Scatter()



Collective Reduce()

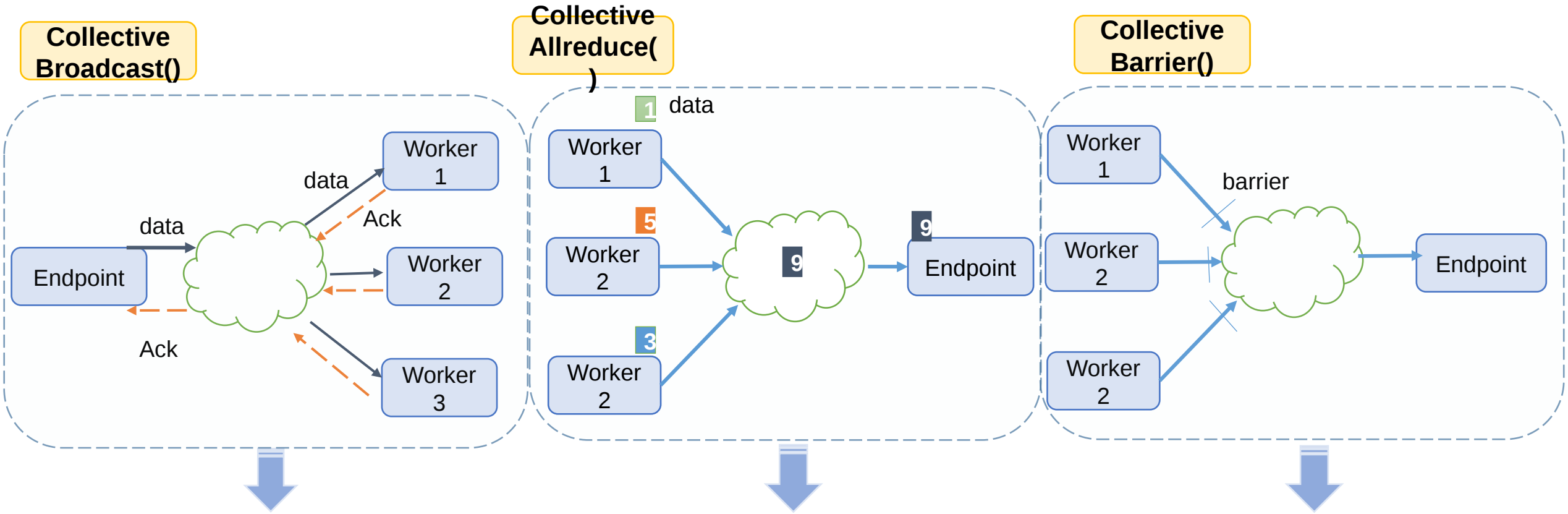


Collective Allreduce()

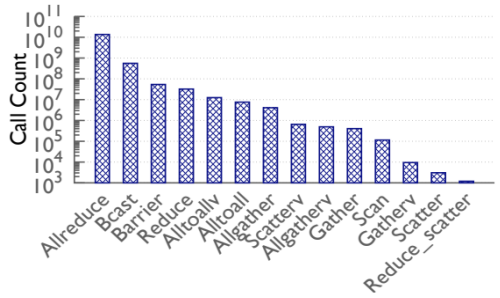


Potential Solution: Network Assisted Collectives Acceleration

- Network Assisted Collective Acceleration(NACA): Using network devices, like switches, to offload and perform collective operations, so as to improve the overall collective communication efficiency.



Allreduce(), Bcast(), and Barrier() rank in top five collectives in func calling[1], in large AI/HPC systems



[1] Characterization of MPI usage on a production supercomputer, by Sudheer Chunduri et al. (SC'18)

Analysis of Existing Solutions(1/3)

- RDMA-based transport protocols are commonly used in AI networking, IB RDMA and RoCEv2.
- There are two modes for NACA based on RDMA:
  - **NACA Based on Server-to-server RDMA Connection**
    - end-to-end RDMA connections
  - **NACA Based on Server-to-switch RDMA Connection**
    - hop-by-hop RDMA connections

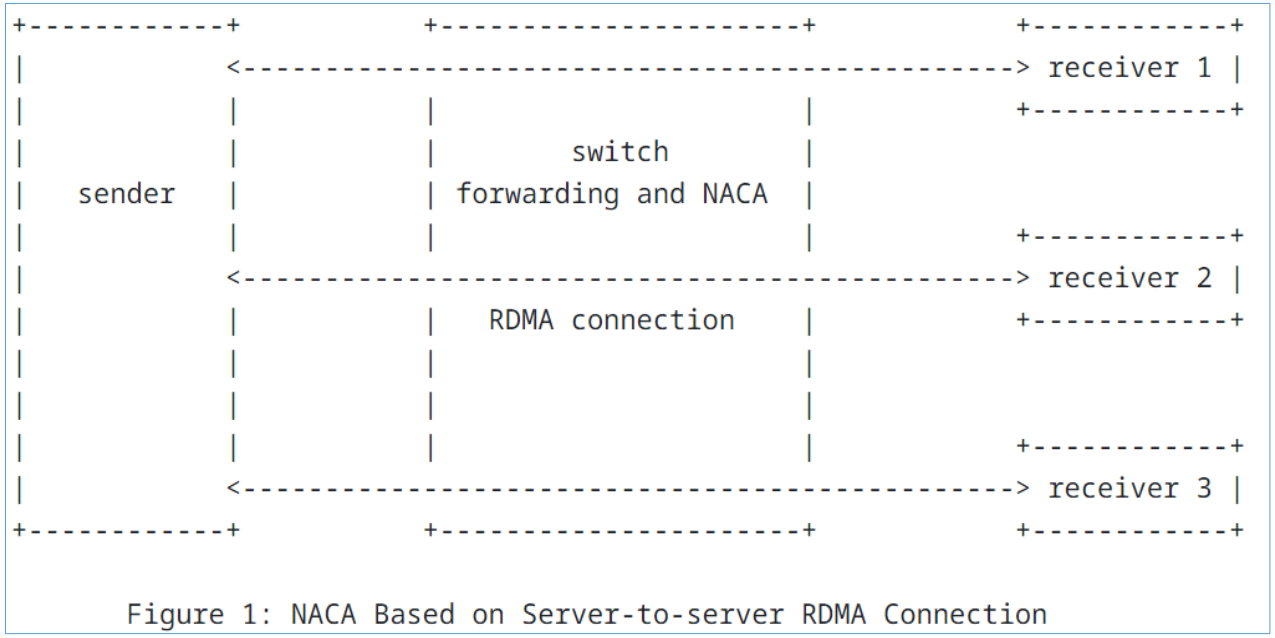


Figure 1: NACA Based on Server-to-server RDMA Connection

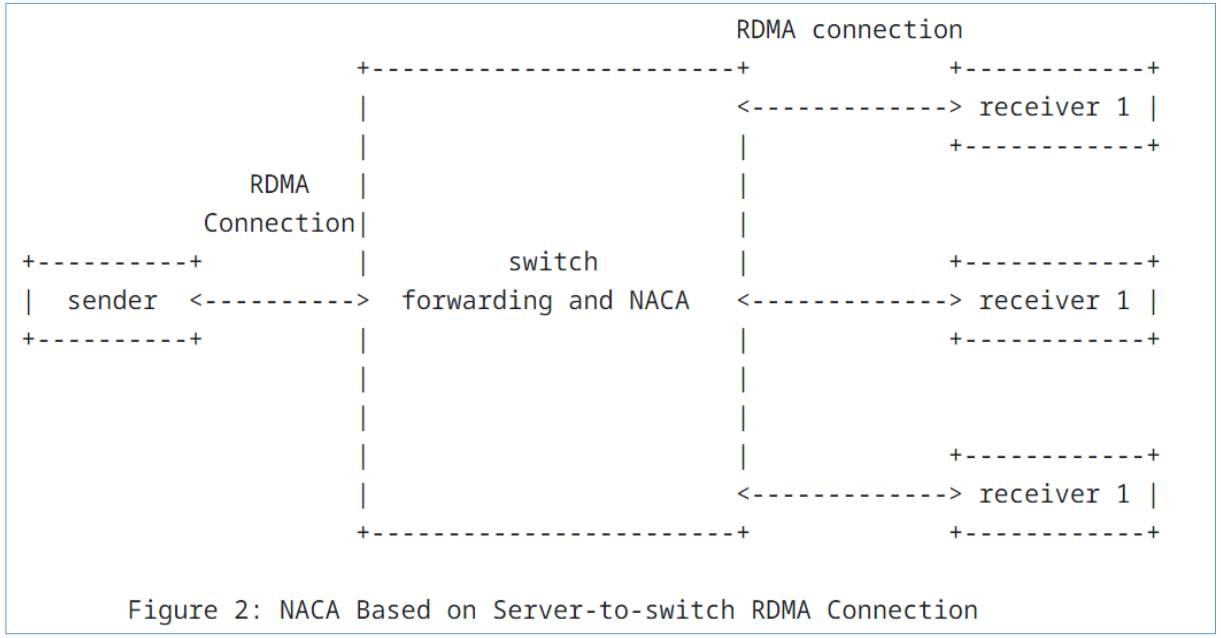
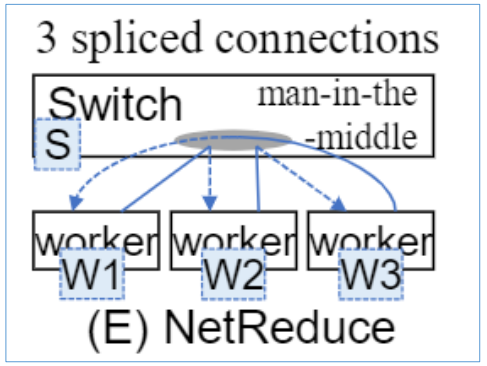


Figure 2: NACA Based on Server-to-switch RDMA Connection

Analysis of Existing Solutions(2/3)

- **NACA Based on Server-to-server RDMA Connection**
  - **NetReduce(ASPLOS 2023)**
    - based on RoCEv2
    - designs a ring algorithm for implementing ALLreduce
    - workers call RDMA send() & recv() to send and receive traffic.
    - switch performs operations following pre-cofigured rules.



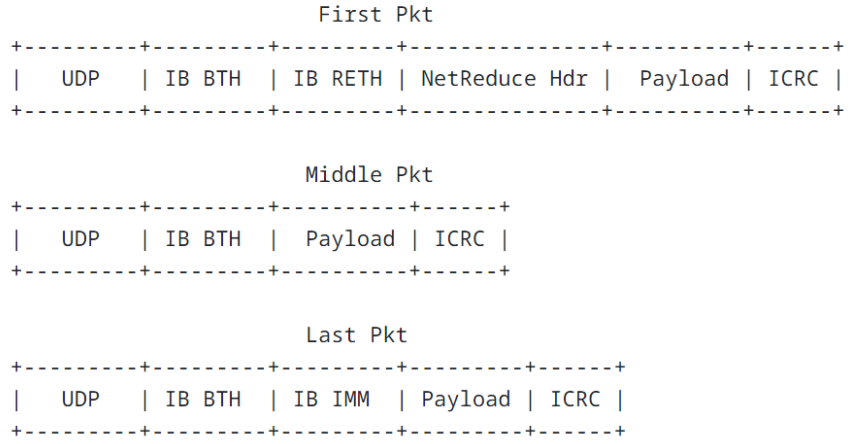
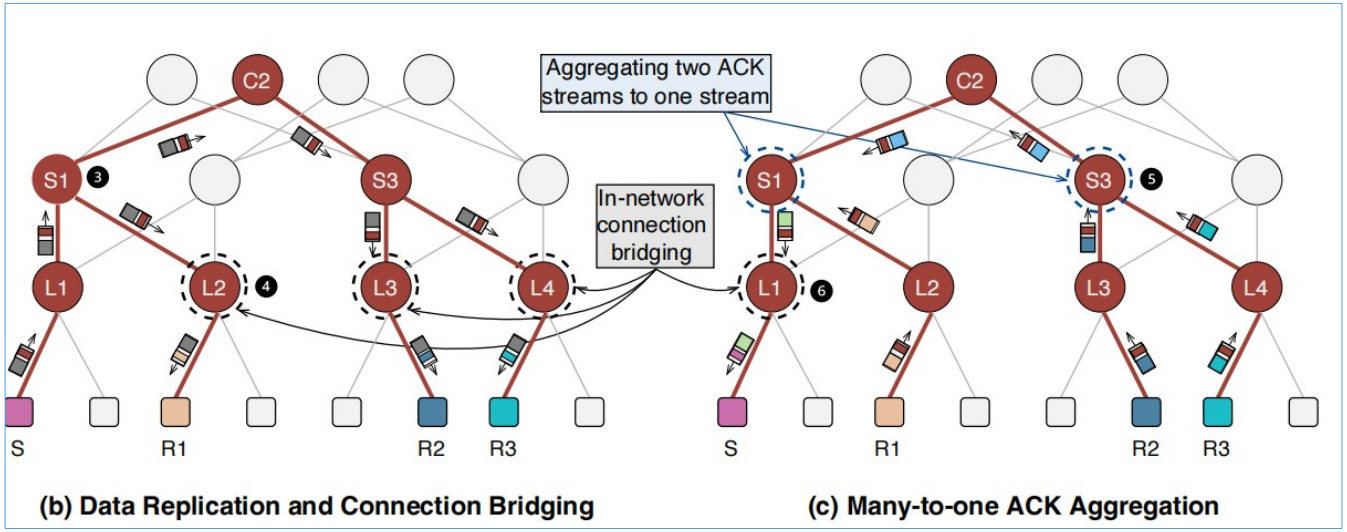


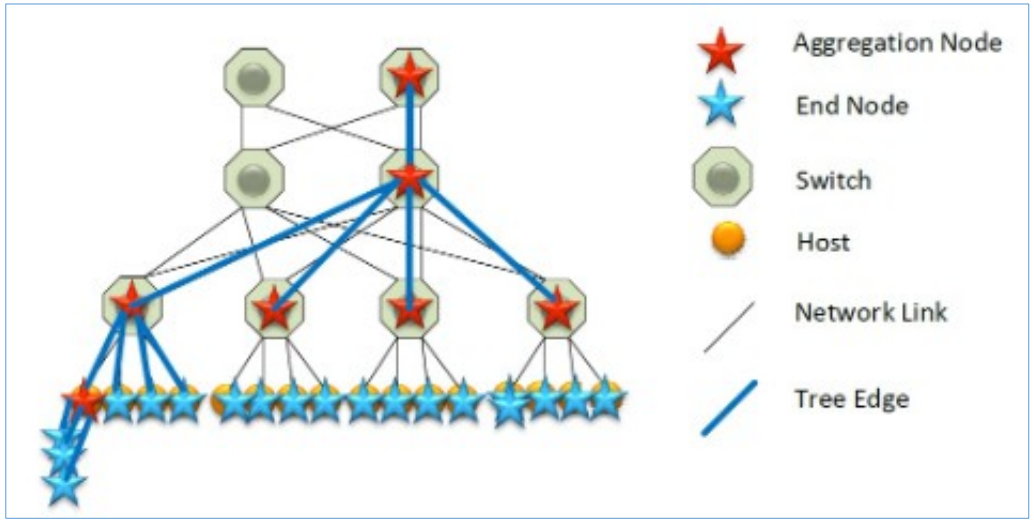
Figure 5: NetReduce Protocol

- **Cepheus(HPCA 2024)**
  - based on RoCEv2
  - end-to-end RDMA connection for connection, and in-switch RDMA bridging
  - designs a IP multicast tree for implementing Broadcast collective operation
  - in-network data replication in forward phase
  - ACK aggregation in backward phase
  - the MFT(Multicast Forwarding Table) in each network node stores dstQP(QPN) and ACKPSN information.



Analysis of Existing Solutions(3/3)

- **NACA Based on Server-to-switch RDMA Connection**
  - **SHARP**
    - Based on Infiniband RDMA
    - Designs a tree-based algorithm for implementing ALLreduce collective operation.
    - Establish an aggregation group for a single job, multiple groups can share a common aggregation tree, following hierarchical aggregation.
    - Only part of RDMA functions implemented in switches, order recovery and packet loss processing left to endpoints.



- **iWARP in IETF**
  - RDMA over TCP, currently no NACA solutions.
  - if server-to-server mode, NACA functions will break TCP reliability mechanisms.
  - if server-to-switch mode, offloading iWARP transport to switch will create state maintainence issues and much overhead.

Aggregation Request / Group Create Packet Format

IBA Headers	SHArP Base Header	Tuple Header	User Data	Operation Header	Target Header	Target Header	SHArP Payload	Invariant CRC	Variant CRC

Aggregation Response Packet Format

IBA Headers	SHArP Base Header	Tuple Header	User Data	SHArP Payload	Invariant CRC	Variant CRC

Fig. 2: SHArP Header schematic.

[1] Scalable Hierarchical Aggregation Protocol (SHArP): A Hardware Architecture for Efficient Data Reduction. (COMHPC'16)

Requirements(1/2)

- **NACA Function Design and Header Definition**
  - R1: MUST define a NACA header to indicate what collective operations that switches need to offload, together with relevant information, for example, message id, sequence number, job id etc.
  - R2: SHOULD support fallback mechanism, in case network devices are not sufficient for processing complete collective operations.

- **Bridge RDMA Transport Semantics**
  - R3: Transport layer MUST support RDMA function.
  - R4: SHOULD allow for different RDMA communication modes for NACA as described in section 3.
  - R5: In server-to-swtich mode, SHOULD clarify which part of the RDMA functions the switch supports, in order to establish a RDMA connection with the server and complete NACA.

- **RDMA Transport Related Issues**
  - R6: NACA MUST be designed with reliability, and the reliability mechanism of RoCEv2 SHOULD be modified to be more efficient.
  - R7: Flow control SHOULD be optimized in order to save more buffer and memory space for NACA functions.
  - R8: The congestion control of NACA SHOULD work compatibly with other congestion control mechanisms applied for other network traffic that runs in the same fabric.

Requirements(2/2)

- **Joint Design of NACA Task Assignment and Routing Policies**
  - R9: NACA Task assignment SHOULD be co-designed with routing policies for joint optimization.

- **Security and Traffic Isolation**
  - R10: Resources MUST be isolated on switches to ensure that different tasks do not interfere each other, and NACA functions do not operate on normal traffic.

- **Fault Tolerance**
  - R11: The mechanism of choosing alternative node for implementing NACA functions MUST be designed, to ensure system robustness and reliability.

Other considerations

- **Security Considerations**
  - data confidentiality, integrity, and authentication.
  - encrypted data brings challenges and performance issues for processing on network devices
- **Operational Considerations**
  - Limited domain[RFC 8799] deployment
  - application and infrastructure to be co-designed to reach optimization

Quick Summary

- NACA is a potential solution in AI networking, for better bandwidth-saving, Incast avoidance, compute & communication overlap, for both scale-up & scale-out networking.
- There are two modes for RDMA-based NACA, both have advantages, and also limitations and optimization space.
- IETF currently only standardize iWARP now. For AI networking, we care about that:
    - Is IETF interested in AI networking?
    - If so, will NACA/INC be in the scope? There is COINRG for a while, but it focuses on research and general terms.
    - iWARP may have some limitations in AI networking, will there be any study space of other RDMA/RoCEv2 solutions in IETF?
    - If not using RDMA, will other solutions be in the scope? RPC-based NACA?

Welcome for more discussions and contributions.

We have held two side meetings on CCO in IETF 118 & 119, please take a look:

https://github.com/CCO-IETF