Collective Communication Optimization(CCO): Use cases, Problem Space, and Requirement

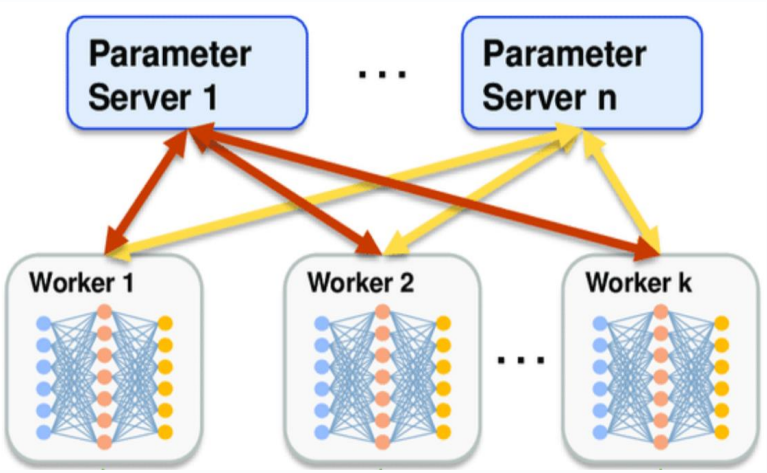
Kehan Yao, China Mobile

IETF 118

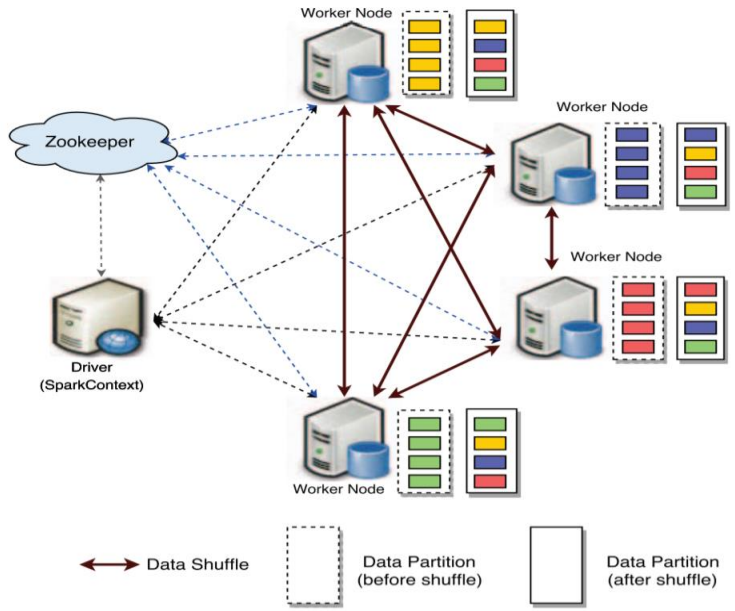
Concept:

Collective communication is an inter-process communication model which plays a key role in high performance computing and modern distributed AI model training workloads such as recommender systems and natural language processing. It involves a group or groups of processes participating in collective operations like AllReduce or AllGather. The communication model can be one-to-all, all-to-one or all-to-all and is usually realized by a sequence of unicast messages.

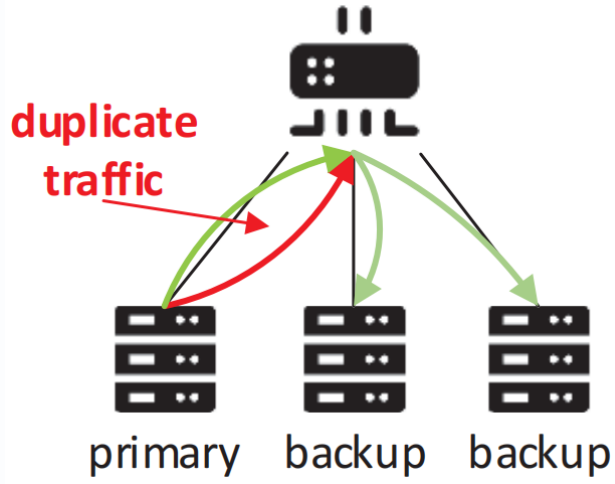
Use cases:



Distributed AI Model Training



Spark Shuffle
in Big Data Anlaysis



Distributed Storage

Major Problems:

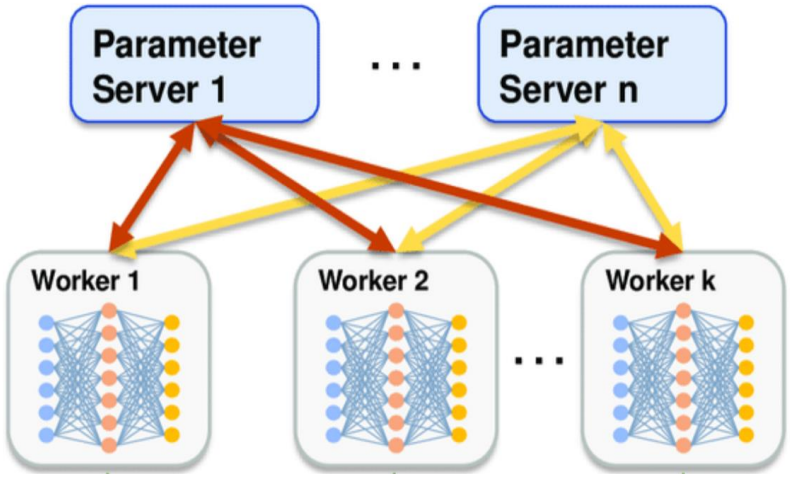
- P2P implementation of Collective Communication incurs much overhead, reflected in:
  - **large bandwidth occupancy(duplications & redundancy)**
    - **Trillions of parameters** of large AI model take up much bandwidth[1].
  - **much data movement across computing nodes(end-to-end transmission)**
    - **Over 30%** of the training time spent on I/O intensive operations[2].
  - **large number of data copies at endpoints(sending one pkt needs to copy at least one time).**

Communication bottleneck & performance degradation

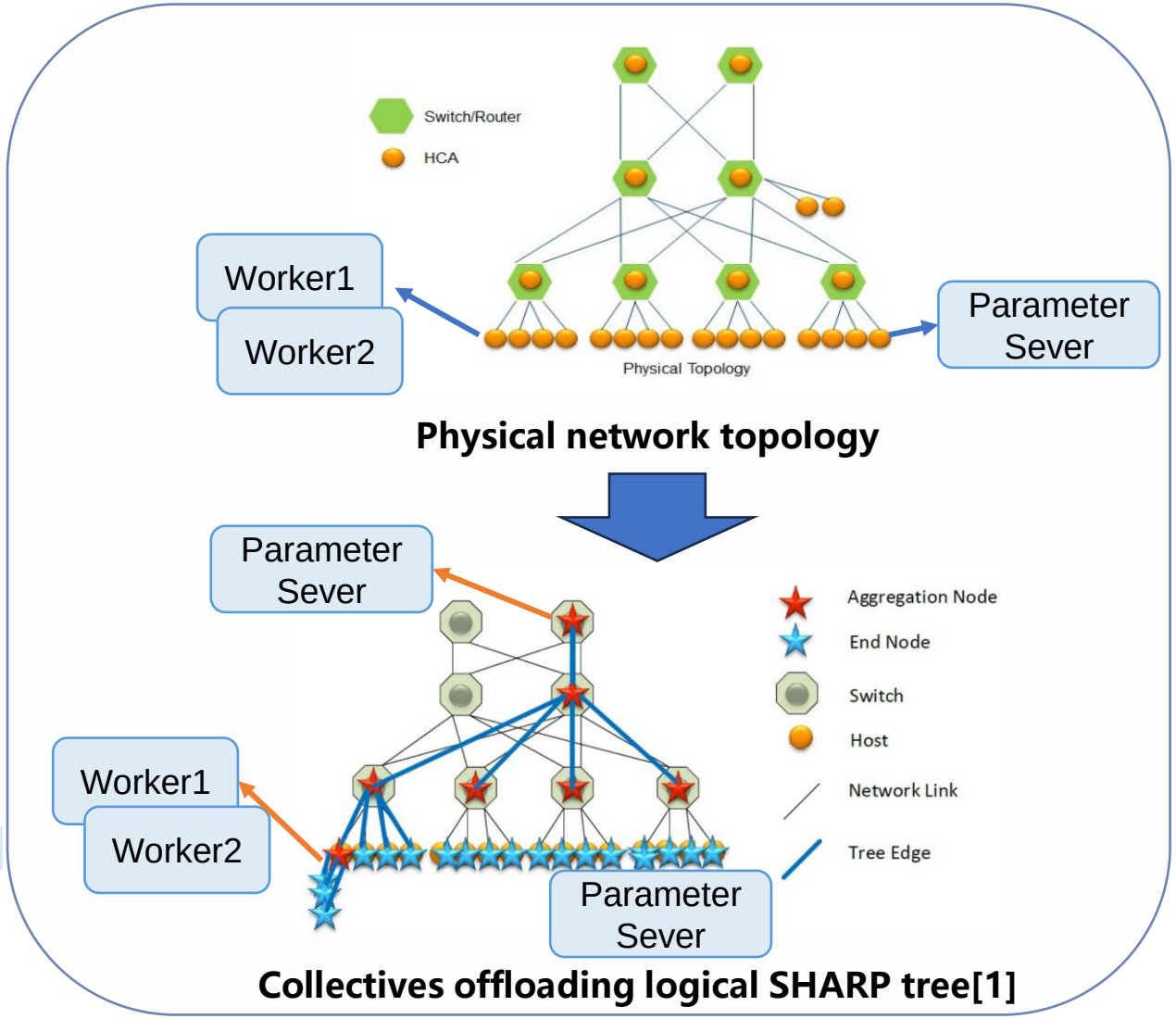
[1] DeepSpeed-MoE: Advancing Mixture-of-Experts Inference and Training to Power Next-Generation AI Scale

[2]Data Movement Is All You Need: A Case Study on Optimizing Transformers

Collective Operations Offloading Illustration



Typical distributed AI model training architecture: Parameter Server in data-parallel mode



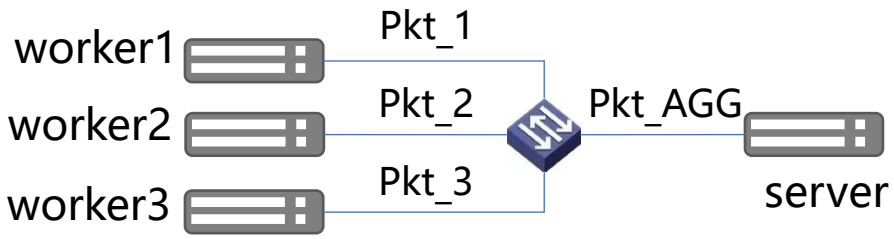
Offloading collective operations to the network can **save bandwidth**, **reduce data movement** and **save time spent on endpoint data copies**.

[1]Scalable Hierarchical Aggregation Protocol (SHArP): A Hardware Architecture for Efficient Data Reduction, by Graham et al. (COM-HPC'16)

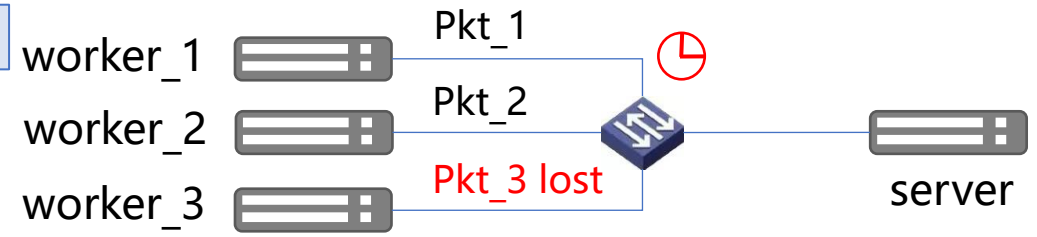
Transport Issues In Collectives Offloading:

- **Reliability**
  - P2P reliability doesn't work well when collective operations are offloaded to intermediate network nodes.

All-to-one communication scenario

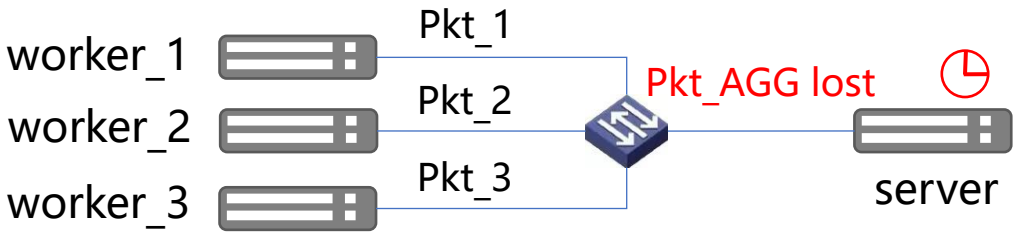


Worker side



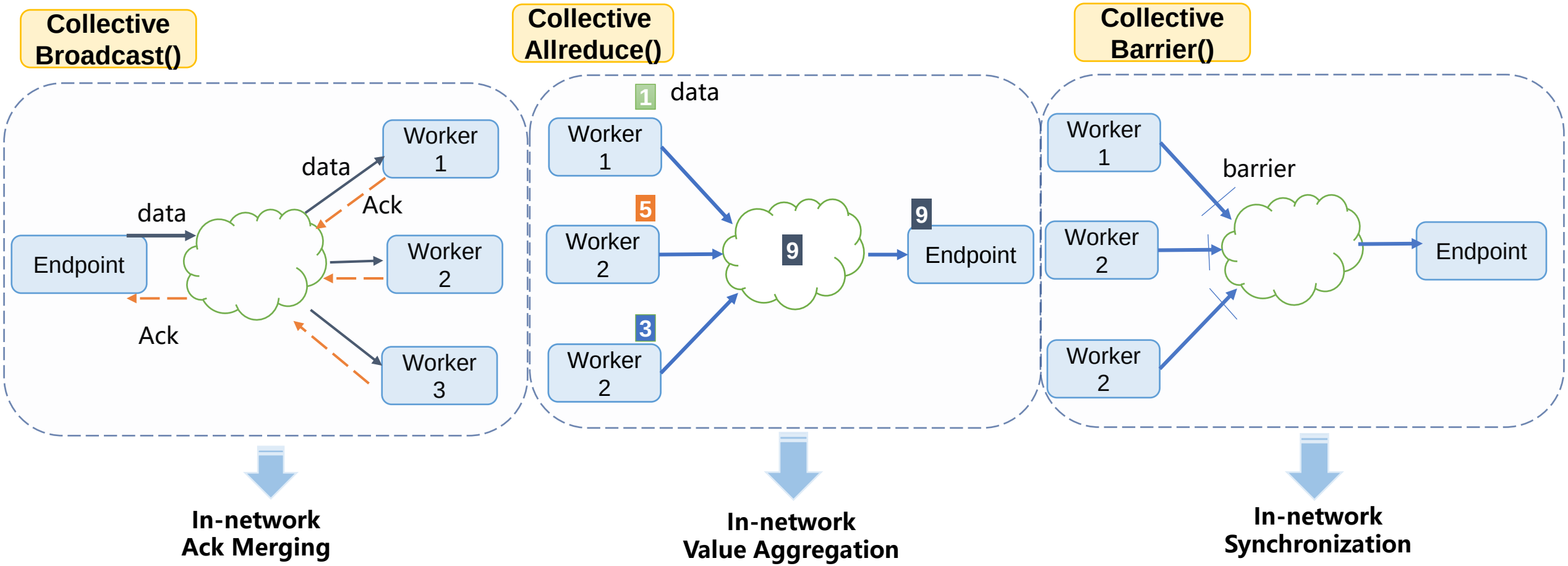
- How to determine if all data has been collected?
- How should the switch handle the loss of original data? Set a timer and drop when timeout? Notify to resend?

Server side

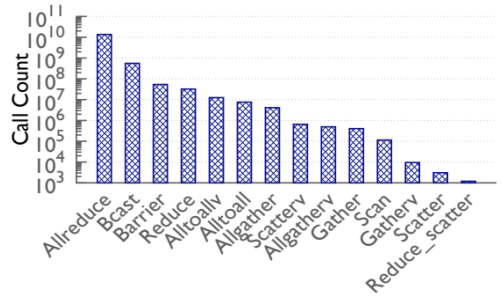


- How does the server determine packet loss? The aggregated packet loss or original packet loss?
- How does the switch confirm the aggregated packet is received?
- Does the switch save aggregation packets that have been sent but not confirmed to be received? How much resources will be consumed?

Transport Reliability Matters in Major Collective Operations



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



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

Transport Issues:

- **Semantic Gap**
  - Collective Communication is inter-process mode, it carries **messages.**
    - **Messages** have no limitation for size**.**
  - Underlying network delivers **packets.**
    - **Packets** have upper limitation**, MTU.**
  - There needs a **mapping function or mechanism** for intermediate node to **recognize, combinate and compute** on the incoming packets, to form messages.
  - Some affecting issues:
    - Message and packet sending rate.
    - Switch buffer management.
    - In-order or out-of-order delivery.

Will impact transport performance as well as correctness.

- **Blocking and Non-blocking transmission**
  - Collective communication supports two types of communication methods simultaneously
  - Collective operations offloading should adjust to different modes.

One-to-all transmission:

- What we need is **a low latency multi-destination delivery mechanism** (with reliability or at least failure detection).
- Collective operations like Broadcast() and AlltoAll() need augmented multicasting mechanisms, as well as other composite operations like reduce-scatter() and all-gather().
- IP multicast has been designed to support broadcast related applications like live streaming and video conferencing. However, collective communication has gaps with IP multicast right now, primarily reflected in requirements on reliability.

Data, Control & Management:

- **In-network primitives**
  - Collective operations: Allreduce, Bcast, AlltoAll…
    - can be called as functions to implement collective communication.
  - In-network primitives:
    - Switch/NIC implementations of collective operations.

- Current in-network primitives is designed for different applications case-by-case, in “chimney-mode”.
- Should have a standard definition on these in-network primitives, to avoid waste in network configurations:
  - Data structure
  - Date type…

- **Topology awareness based on collective operations offloading.**
  - Existing topology awareness algorithms are not co-designed with In-network computing capabilities.
  - Should adjust the algorithms to be suitable for collective operations offloading
    - For example, in clos-based topology, find the most suitable in-network tree node for offloading
    - Switch memory, distance, availability…

Problem Space and Relative Areas:

- **Collective API Enhancement:**
  - Should design collective offloading specific functions
  - xCCL/standardized CCL libraries should support these extensions
- **Transport Issues:**
  - Reliability

  Underlying network lacks collective communication reliability
  - Semantic Gap

  Message passing vs packet delivering
  - Blocking & Non-blocking

  Collectives offloading should adjust to different communication modes
- **One-to-Group Transmission:**
  - Message Bcast/AlltoAll/...

  Need better multi-destination delivery mechanism
- **Data & Control & Management:**
  - In-network Primitives

  Collective operations based on unified In-network primitives
  - Topology Awareness

  Improve existing topology-aware algorithms to support collectives offloading

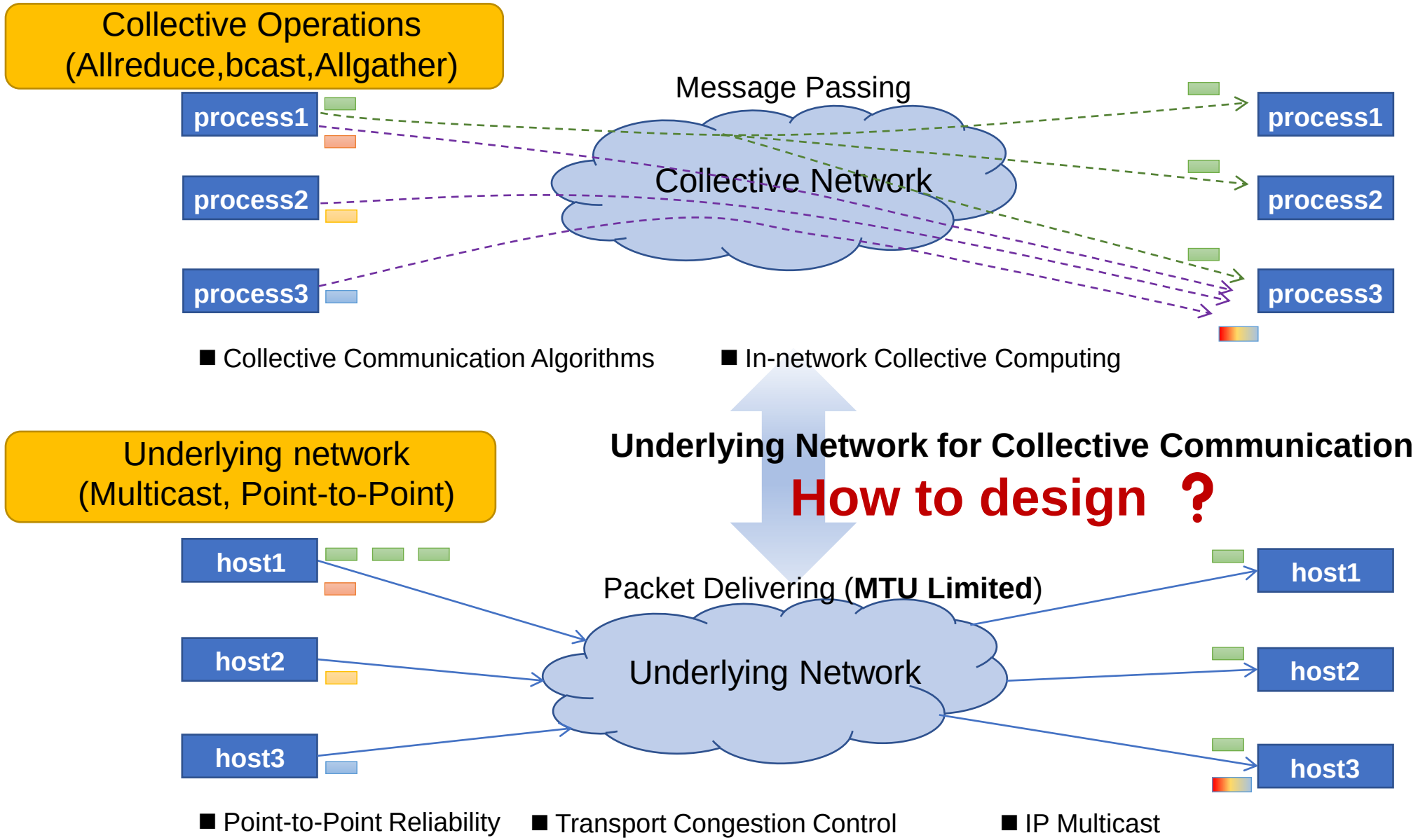
ART Area

Transport Area

RTG Area

OPS Area

Goal: Better Communication Pattern



Requirements:

- R1. Transport layer **MUST** support RMA function.
- R2. Memory sharing is **RECOMMENDED** to support collective operations.
- R3. The implementation of blocking or non-blocking communication of applications **MUST** be adjusted according to different communication modes.
- R4. The transport layer **MUST** provide appropriate reliability mechanisms to adapt to different communication modes.
- R5. The transport layer **MUST** carry messages that network devices can recognize to complete offloaded collective operations processing.
- R6. The transport layer **MUST** support fallback mechanism, in case network devices are not sufficient for collective operations offloading.
- R7. **MUST** support unified definition and management of data types, data structures supported by network devices considering collectives offloading, and the unified management of resources such as memory of network devices.
- R8. It is **RECOMMENDED** to achieve topology awareness, task scheduling, and allocation in collaboration between the end and network.
- R9. Multicast protocols **SHOULD** be extended to support collective communication. However, whether to design new multicast algorithms and protocols that are dedicated for collective communication is out of the scope.
- R10. The mechanism of choosing alternative node for implementing collective operations **MUST** be designed, to ensure system robustness and reliability.

Analysis:

- **Other on-going work which may be of interest to the topic:**

- **COINRG(Computing in the Network Research Group):**
  - COIN investigate how network data plane programmability can improve Internet architecture.
  - Broad applications(network functions offloading, machine learning acceleration, in-network caching and in-network control, etc.)
  - Collective communication optimization not necessarily designed with network programmability.

- **SHARP(Scalable Hierarchical Aggregation and Reduction Protocol):**
  - Collective operations offloading based on Infiniband network architecture.
  - Currently, not interoperable with the Internet architecture.

- **UCX(Unified Communication Framework):**
  - Design a framework for collective communication implementation and address the cross-platform functionality and performance portability challenges.

Security and Operational Considerations:

- Collective communication optimization may introduce some security and privacy concerns:

- On one hand, it may impact data confidentiality, integrity, and authentication.
  - Both security-enabled and security-less deployments should be considered.
  - It's suggested to deploy in **limited domains**[RFC8799] at first, since it does not have to pay the penalty of expensive crypto or authority operations. Applications can choose to trust the network within limited domains or they can trust mutually if both of them belong to the same administrator.
  - Extending the technology to the Internet should be designed together with some intrinsic protective actions.

- On the other hand, decrypting and encrypting data on network devices is not only inefficient, but also involves issues such as key management and authorization.

Welcome for more discussions and contributions.