The 30th IEEE International Symposium on High-Performance Computer Architecture (HPCA 2024)

Cepheus: Accelerating Datacenter Applications with High-Performance RoCE-Capable Multicast

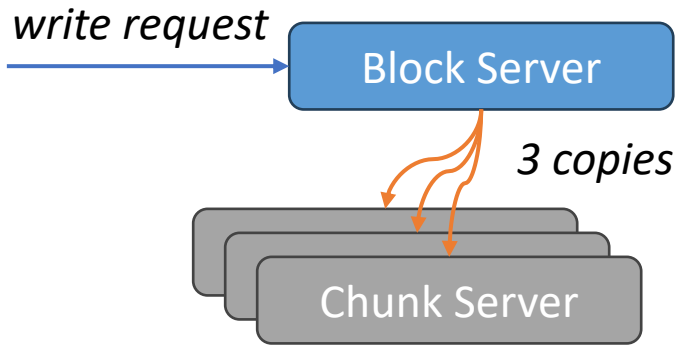
Wenxue Li[1*] , Junyi Zhang[2,3*] , Yufei Liu[2] , Gaoxiong Zeng[2] , Zilong Wang[1] , Chaoliang Zeng[1] , Pengpeng Zhou[2], Qiaoling Wang[2], Kai Chen[1,3]

[1]iSING Lab, Hong Kong University of Science and Technology,

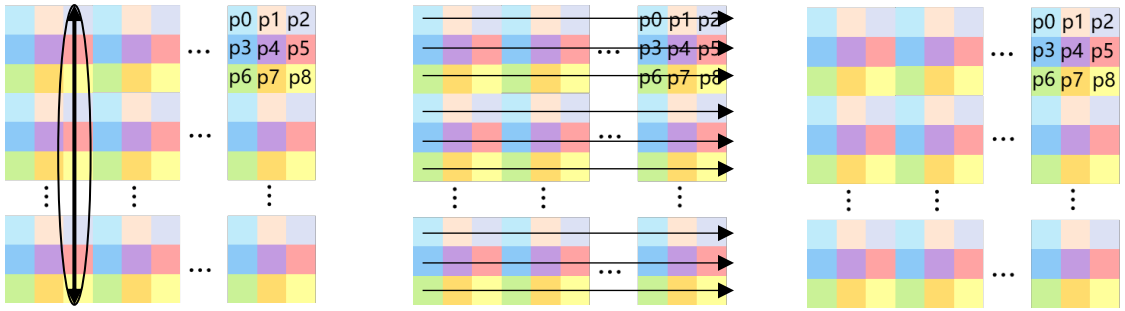
[2]Huawei, [3]USTC, [*]Equal contribution

One-to-many Communication is Prevalent

- Modern datacenter (DC) applications widely exhibit multicast communication patterns.
  - Replications distribution in distributed storage system
  - HPC applications, e.g., High-performance Linpack (HPL) benchmark



Block Storage system



PF, PB, RS phases during a HPL epoch

- An efficient multicast primitive substantially benefit DC applications.

RDMA: De-facto Networking Tech in DCs

- RDMA is emerging as the de-facto networking technology in DCs, to meet the stringent communication requirements from applications.
- RDMA over Converged Ethernet (RoCE)[1]: an RDMA transport protocol

RoCE vs. IB? This work focuses on RoCE.

- Reliable Connection (RC) mode of RoCE is mostly adopted.
- RoCE[2] semantics: one-to-one reliable connection.

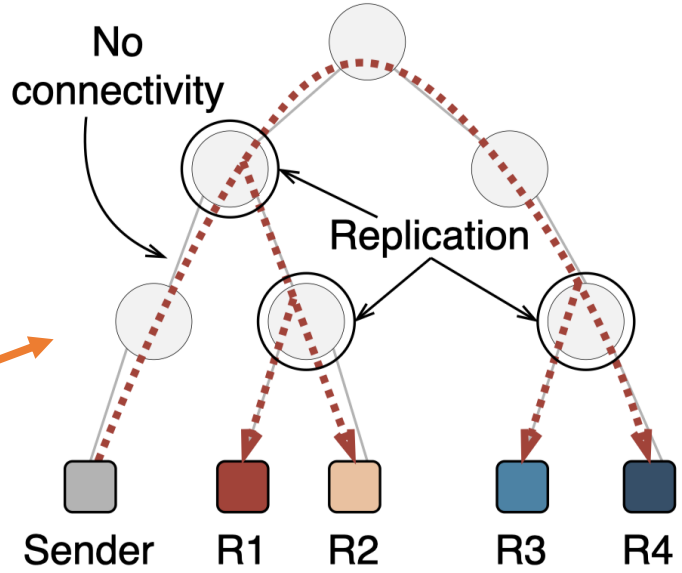
	SEND/RECV	WRITE	READ	Message Size
RC	✓	✓	✓	2 GB
UC	✓	✓	✗	2 GB
UD	✓	✗	✗	4 KB

Comparisons of RoCE transport modes.

[1]*RoCE has an extension version, RoCEv2, we actually focus on RoCEv2 and use RoCE for convenient notation.*
[2]*By default, RoCE refers to its RC mode*

Mismatch Between Multicast and RoCE

- Native Multicast
  - *Multicast sender*: only send out one single copy of data.
  - *Network*: replicates data at proper switches and forwards the data to multiple receivers.
  - *Distribution tree*: replication is made as late as possible to reduce traffic volume.



(a) Native multicast.

- Pros: efficient traffic transmission
- Cons: layer-4 transport unfeasibility
  - Due to the mismatch of native multicast data flow structure and transport's one-to-one semantics, causing limited usage among applications.

Insufficiency of Application-layer Multicast

- Distributed frameworks, MPI, NCCL, Spark, etc., develop their private application-layer multicast (AMcast) primitives.
- AMcast: a logical multicast interface to applications, where the traffic is delivered by multiple unicast (one-to-one) transmissions.

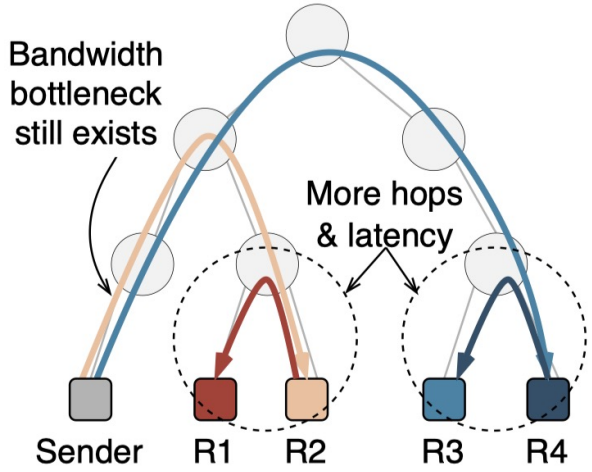
Prof: performant end-host transport from reusing RoCE.

Much more prevalent than Native Multicast in practice.

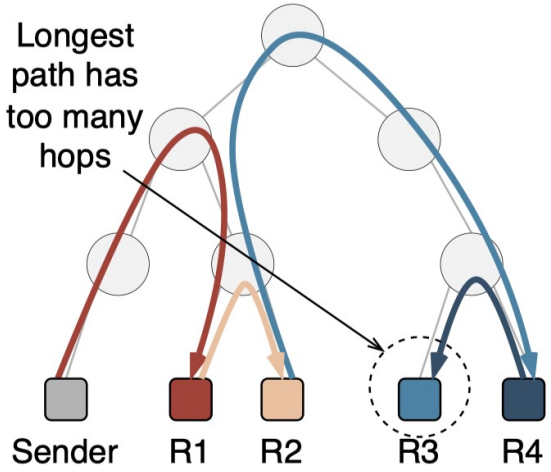
Cons: inefficient traffic transmission

Suffering from either redundant traffic (high throughput ❌) or increased transmission hops (low latency ❌).

Comparing Existing Schemes



(b) Binomial tree (AMcast).



(c) Chain (AMcast).

	# Hops of longest path	Bandwidth bottleneck released?	Reusing commodity RoCE?	# End-host stack experience
NMcast	6 (min)	Yes	No	Once
Binomial Tree	8 (mid)	Partially	Yes	Many
Chain	14 (max)	Yes	Yes	Many
Cepheus	**6 (min)**	**Yes**	**Yes**	**Once**

(d) Cepheus vs. existing schemes.

- *Native Multicast*. High throughput and low latency; Cannot reusing RoCE.
- *Binomial Tree*. Latency-friendly (logarithmic latency form); Poor performance with large messages
- *Chain*. Throughput-friendly (BW bottleneck fully release); Longer latency (linear to the number of nodes)

Our Goal

	# Hops of longest path	Bandwidth bottleneck released?	Reusing commodity RoCE?	# End-host stack experience
NMcast	6 (min)	Yes	No	Once

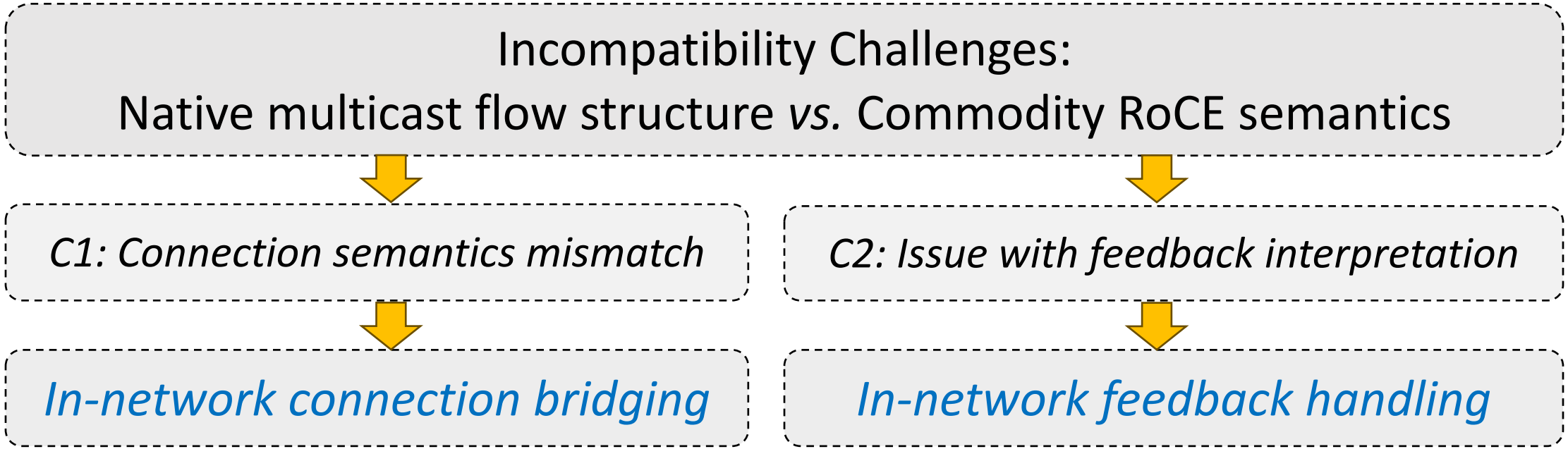
Can we design a multicast primitive that achieves performance gains from both multicast and commodity RoCE?

Cepheus

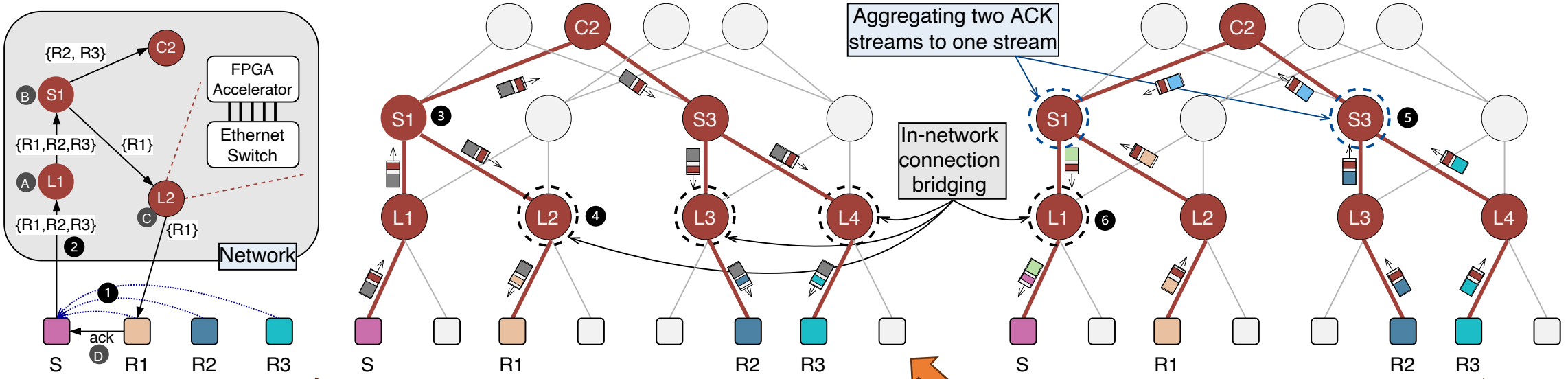
- Native Multicast. High throughput, low latency; Cannot reusing RoCE.
- Binomial Tree. Latency-friendly (logarithmic latency form); Poor performance with large messages
- Chain. Throughput-friendly (BW bottleneck fully release); Longer latency (linear to the number of nodes)

Intuition and Challenge

- *Basic Intuition:* *build on native multicast* (i.e., inherit its transmission-efficient multicast flow structure) and *exploit more switch functionalities to deliver a RoCE-capable multicast stream* that can be directly processed by commodity RNICs.



Cepheus Design Overview



(a) MFT Registration (b) Data Replication and Connection Bridging (c) Many-to-one ACK Aggregation

1. Hosts Establishing Connections

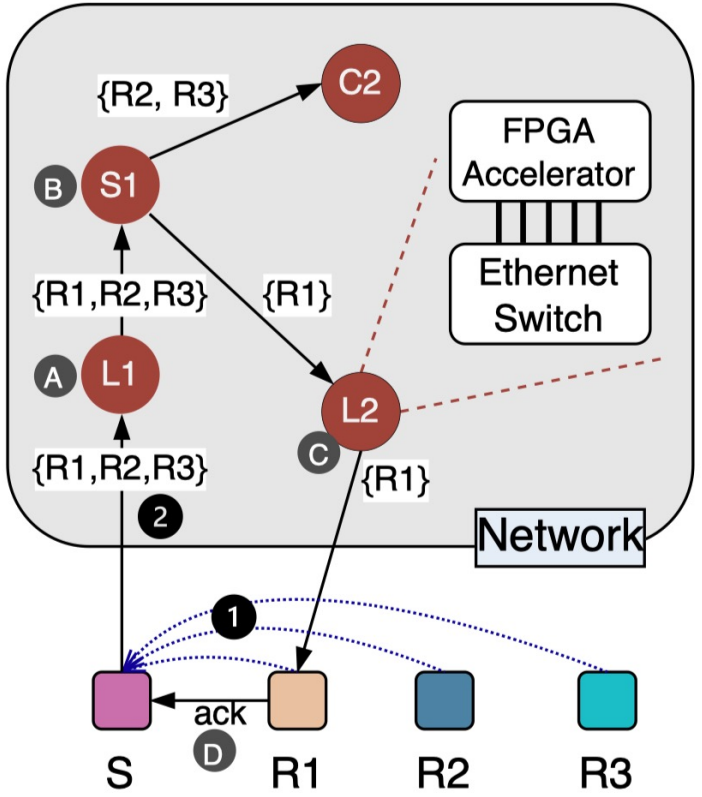
2. Multicast Forwarding Table Registration

3. Data Replication and Connection Bridging

4. RoCE-capable Feedback Handling

Connection Establish & Table Registration

- Hosts follow the existing unicast-like procedure to establish one RoCE connection for each multicast group.
  - Virtual remote connection: “dstIP = McstID”
- Table registration is performed in control-plane, comprising a controller and several agents.

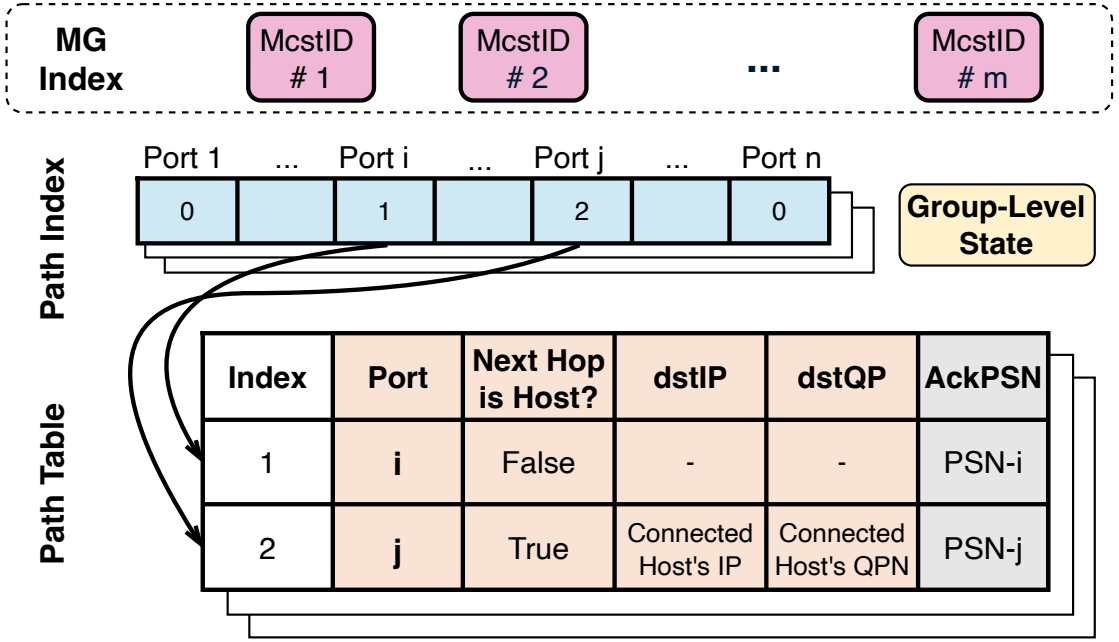


1. Controller collects the “IP” and “QPN” states of other hosts.

2. Controller fits these states into Table Registration Protocol packets and transmits them to switches for building multicast forwarding table (MFT)

Multicast Forwarding Table (MFT) Structure

- Every switch in the distribution tree has its local MFT, guiding its overall in-network processing logic.
- *Path Index*: an array that identifies whether a switch port is involved in the distributed tree.
- *Path Table*: each entry represents an outgoing path
  - Next hop is a switch
    → Two values are marked as invalid
  - Next hop is a host
    → Maintaining “dstIP” and “dstQP” in this entry

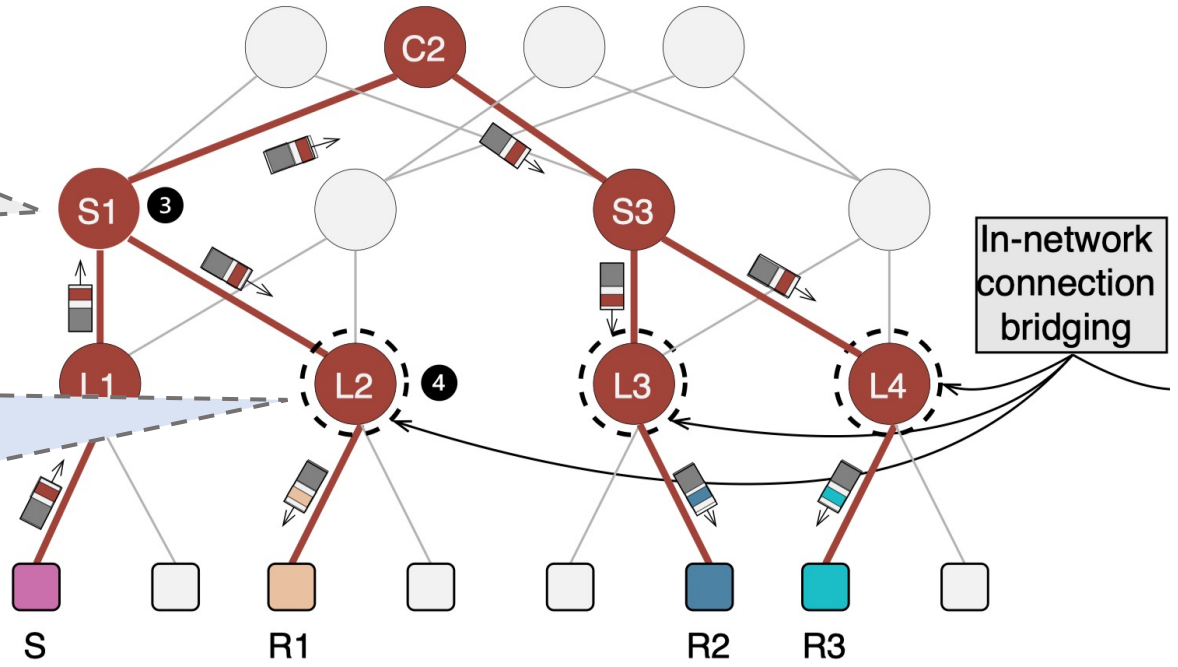


Data Replication and Connection Bridging

(1) Multicast sender transmits data via commodity RoCE logic.

(2) Non-leaf switch follows its local MFT to replicate and forward data to multiple output ports.

(3) Leaf switches are responsible for modifying the BTH header to bridge connections for different receivers

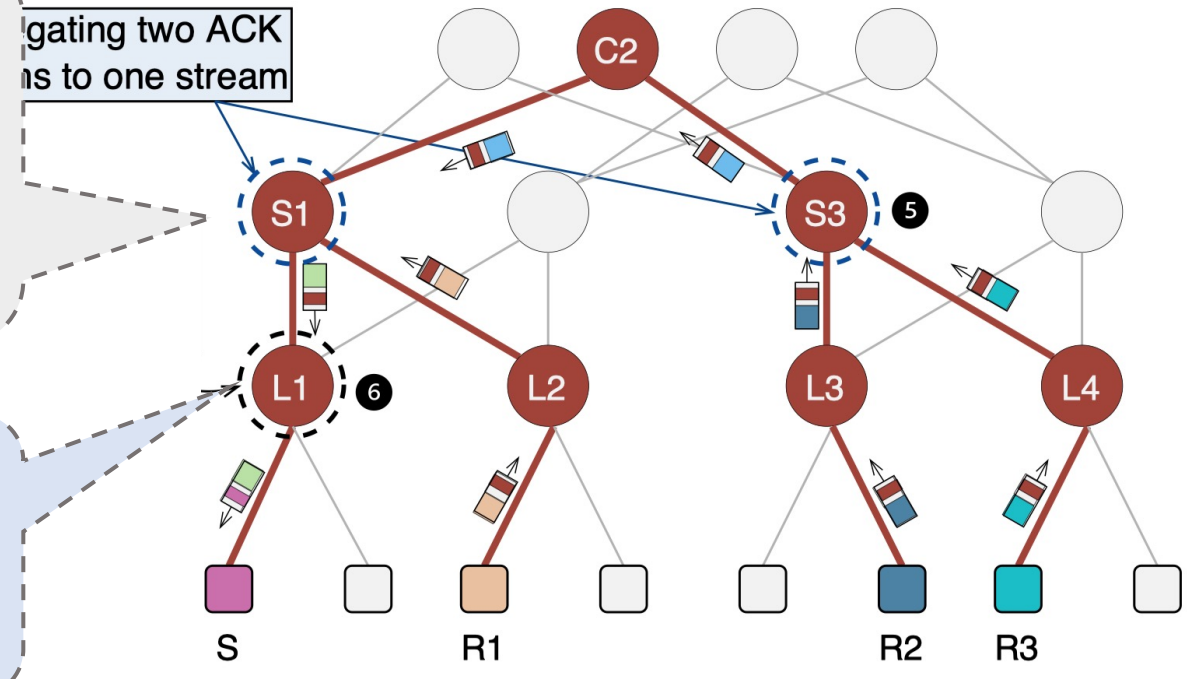


RoCE-capable Feedback Handling

(1) Receivers generate ACK/NACK/CNP packets following standard RoCE logic.

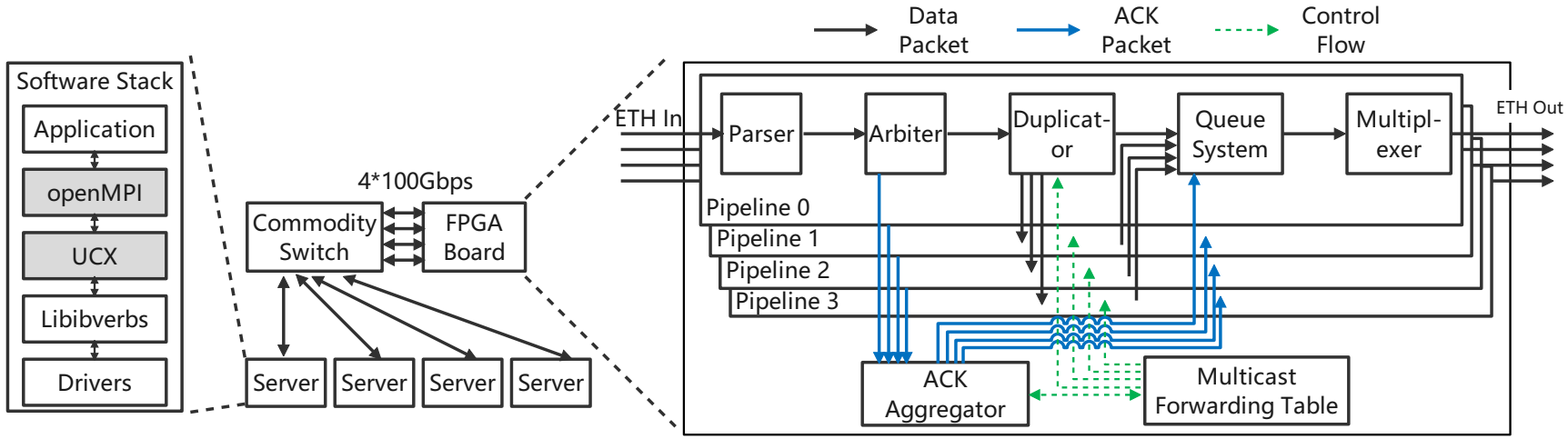
(2) Feedbacks traverse distribution tree inversely, and the switches aggregate ACK/NACK and filter CNP, when there are multiple input feedback streams.

(3) Leaf switch connected to the sender modifies the packet's BTH header before forwarding the final feedback.



Implementation and Testbed

- Cepheus Testbed: an Ethernet switch, an FPGA board, and four servers.

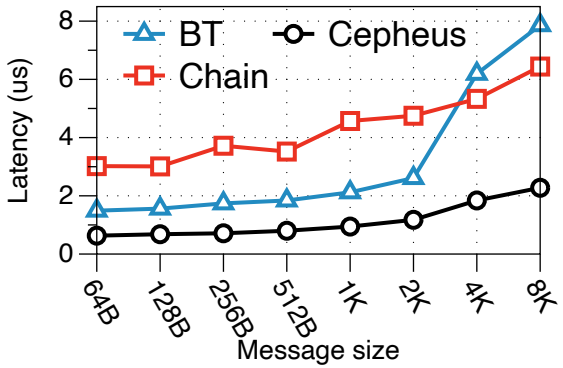


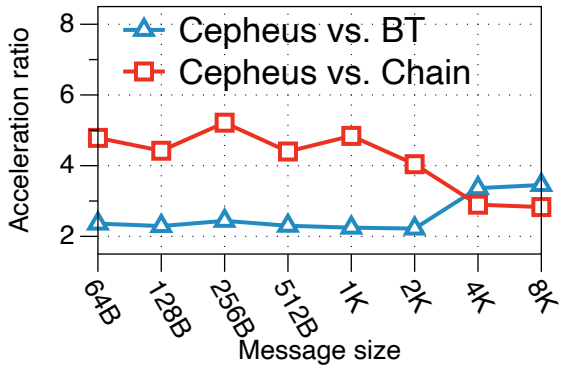
Resource	Usage	Proportion
LUT	53169	4.8%
Register	15391	0.7%
BRAM	188	4.9%

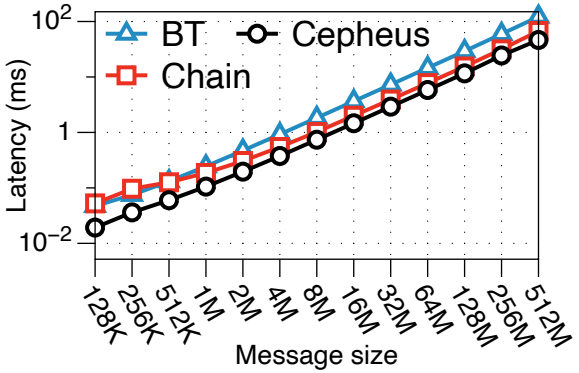
- FPGA Accelerator.
  - All in-network processing functions are implemented in an FPGA board, as a building block attached to the Ethernet switch.
- End-host APIs: integrated to MPI; transparent to applications; do not require any RNIC or driver modification.

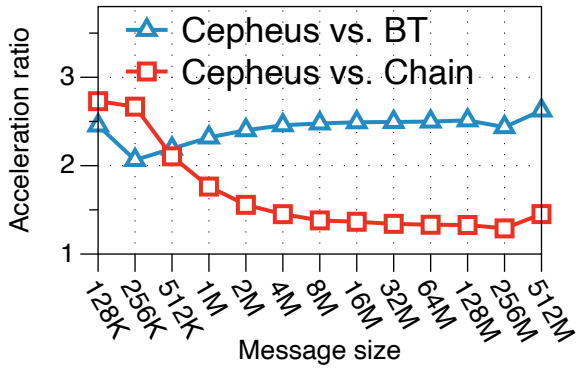
Evaluation: Micro Benchmark

- Integrating Cepheus into OpenMPI & evaluating *MPI Broadcast*.
- Comparing Cepheus with Binomial Tree (BT) and Chain, which are oriented for small and large messages, respectively.







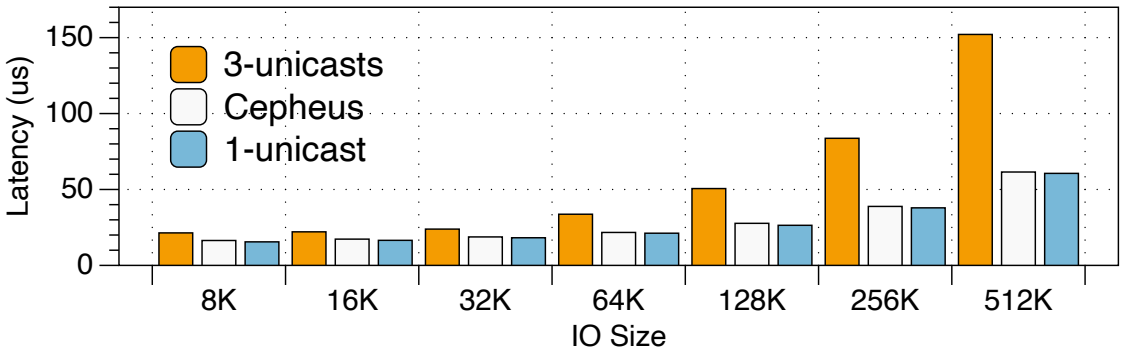


5.2X and 3.5X lower latency for small messages, compared to Chain and BT

1.3X and 2.8X higher throughput for large messages, compared to Chain and BT

Evaluation: Realistic Applications

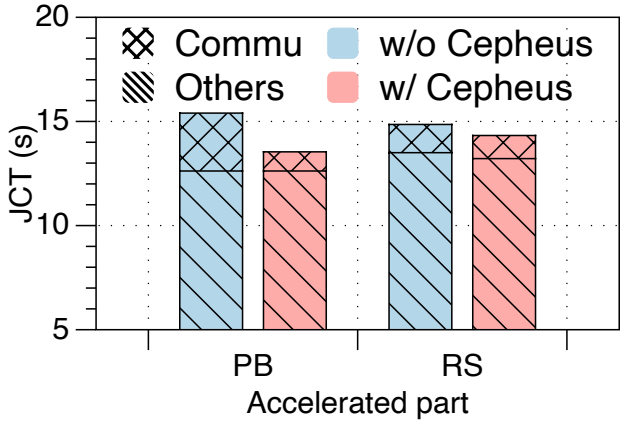
Distributed storage system



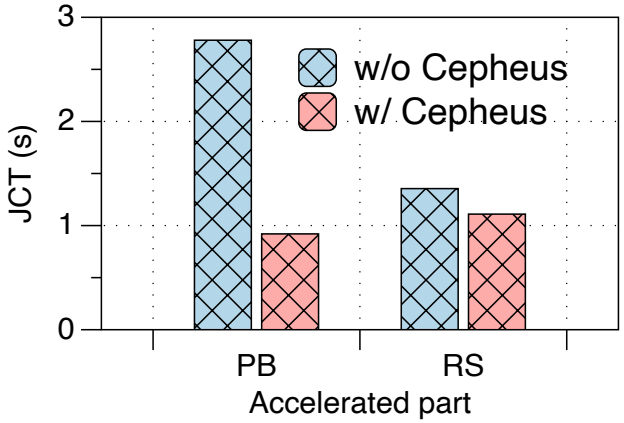
Scheme	1-unicast	3-unicasts	Cepheus
8KB IOPS(M)	1.188	0.413	1.167

Writing Throughput

HPL benchmark



End-to-end JCTs.



Communication time.

HPL is sensitive to throughput.

Cepheus enhances the performance of realistic applications.

Conclusion

- Cepheus is a high-performance RoCE-capable multicast solution that delivers *performance gains from both multicast and RDMA transport*.
- Cepheus opens the door for efficiently leveraging the widely adopted RDMA transport with *in-switch assistance* to accelerate collective communication patterns.
- For future works, we plan to extend Cepheus for more collective communication primitives, such as many-to-one (e.g., MPI-Reduce) and many-to-many (e.g., MPI-Alltoall).

Thank you!