Impact of RoCE Congestion Control Policies on Distributed Training of DNNs

Tarannum Khan*, Saeed Rashidi [†], Srinivas Sridharan [‡], Pallavi Shurpali [‡], Aditya Akella*, and Tushar Krishna [†]

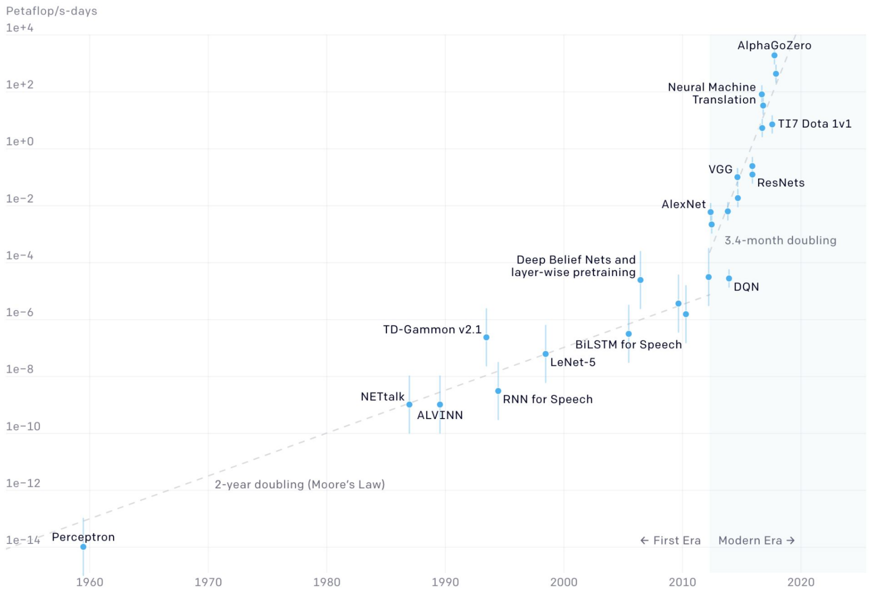
* The University of Texas at Austin, Austin, USA
[†] Georgia Institute of Technology, Atlanta, USA
[‡] Meta, Menlo Park, USA

Agenda

- **Introduction** - Distributed training
- **Introduction** - Specialised machine learning platform
- **Problem**
- **Introduction** - RDMA and congestion control
- **Background** - collective communication algorithms, DLRM
- **Simulator** - Astra-Sim and NS3
- **Experiments** - Incast study for congestion control algorithms
- **Experiments** - Collectives study for single switch
- **Experiments** - Collectives study for Clos topology
- **Experiments** - DLRM workload
- **Conclusion**

Distributed Training

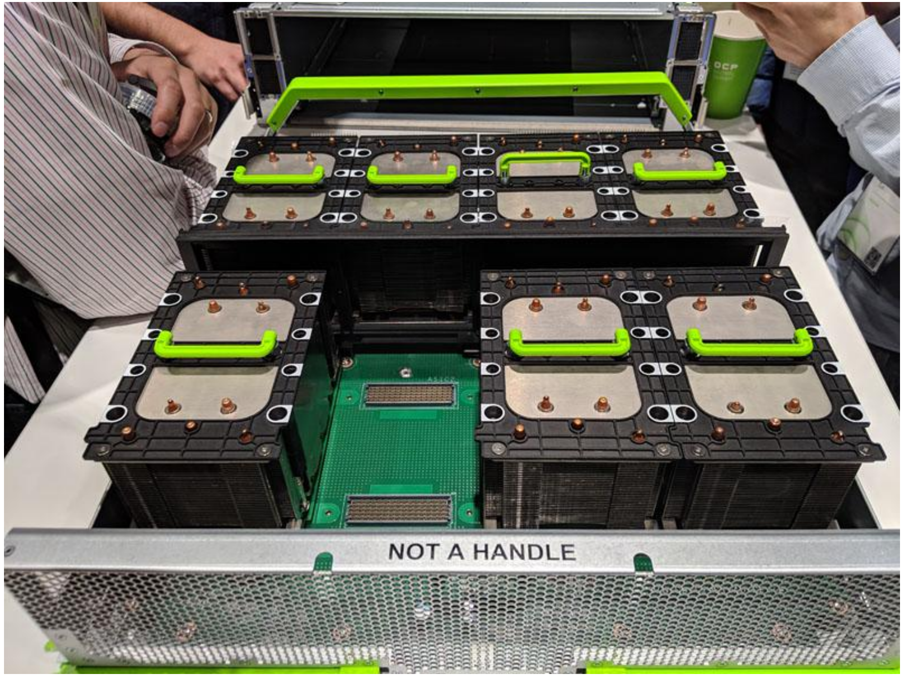
- Training time is increasing.
- Distributed training is used to decrease the training time by distributing the training tasks across multiple accelerators aka neural processing units (NPU).
- Distributed training comes at the expense of communication overhead between NPUs to synchronize model gradients and/or activation.



Source:https://openai.com/blog/ai-and-compute/

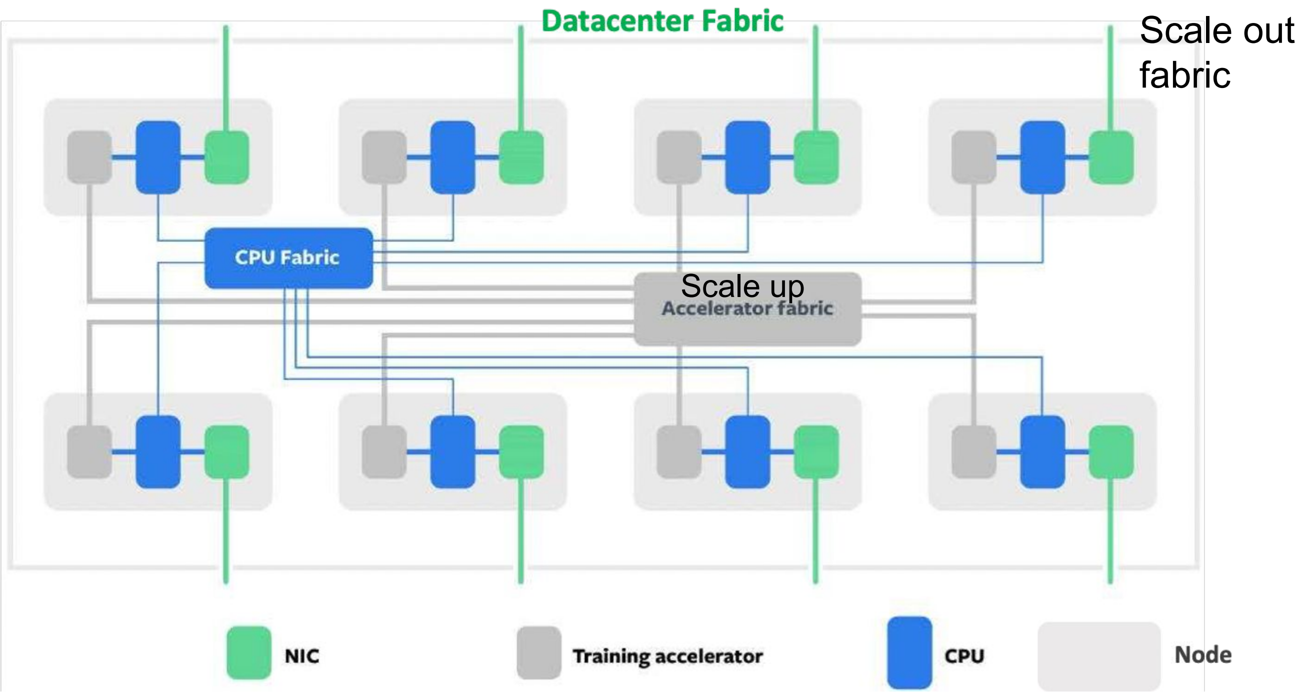
Specialized distributed training platforms

- Distributed training platforms are built using high-end compute and network components.
- Distributed training platforms employ dedicated networks that separate training traffic from the rest of the datacenter traffic.
- Due to the growing size of DNN models and of training datasets, training platforms are often scheduled to perform only one training job at a time for the critical DNN workloads (e.g., recommendation models).



Meta Zion Server

Components of DL training platform



Source: “Zion: Facebook Next- Generation Large Memory Training Platform”, Misha Smelyanskiy, Hot Chips 31"

The problem

- Because of the unique characteristic of DL platform and the usage of networks in distributed deep learning workload, it is crucial to revisit the networking stack and identify whether the current state-of-the-art networking protocols are optimal for such platforms.
- We focus on RDMA over Converged Ethernet (RoCE) protocol due to its compatibility with current Ethernet-based fabric and widespread usage on distributed training platforms

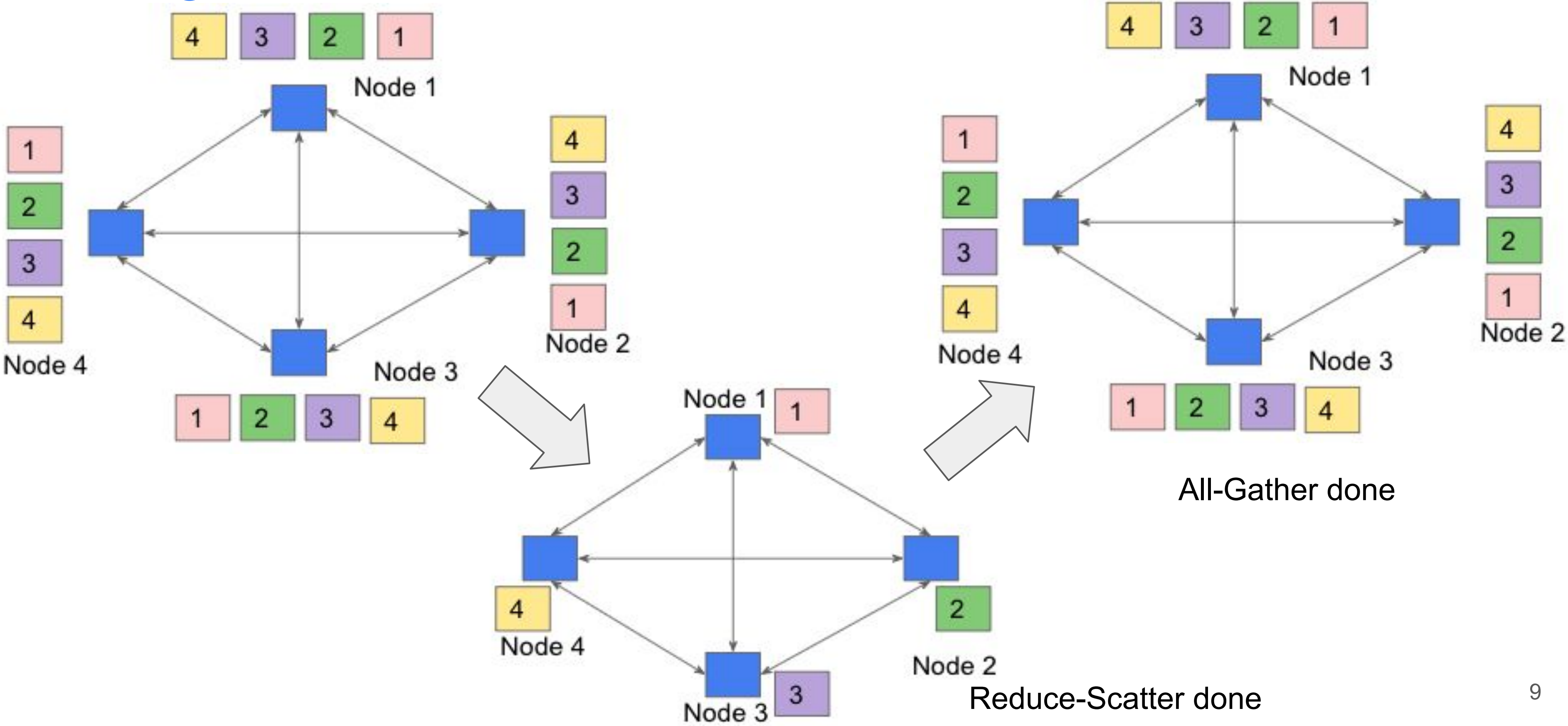
RDMA over Converged Ethernet (RoCE) and Congestion Control

- RDMA protocol is more efficient on lossless networks, which is not natively supported on Ethernet-based fabrics.
- Baseline RoCE enforces congestion control at the link layer through the Priority Flow Control (PFC) mechanism.
- PFC mechanism suffers from many drawbacks in conventional data-center environments, including unfairness, head-of-line-blocking, and deadlock.
- Recent works have shown the importance of congestion control on RoCE to achieve maximum performance with minimal PFC generation.
- In this paper, we study thoroughly performance of different congestion control mechanism for DL workloads on specialised DL platforms.

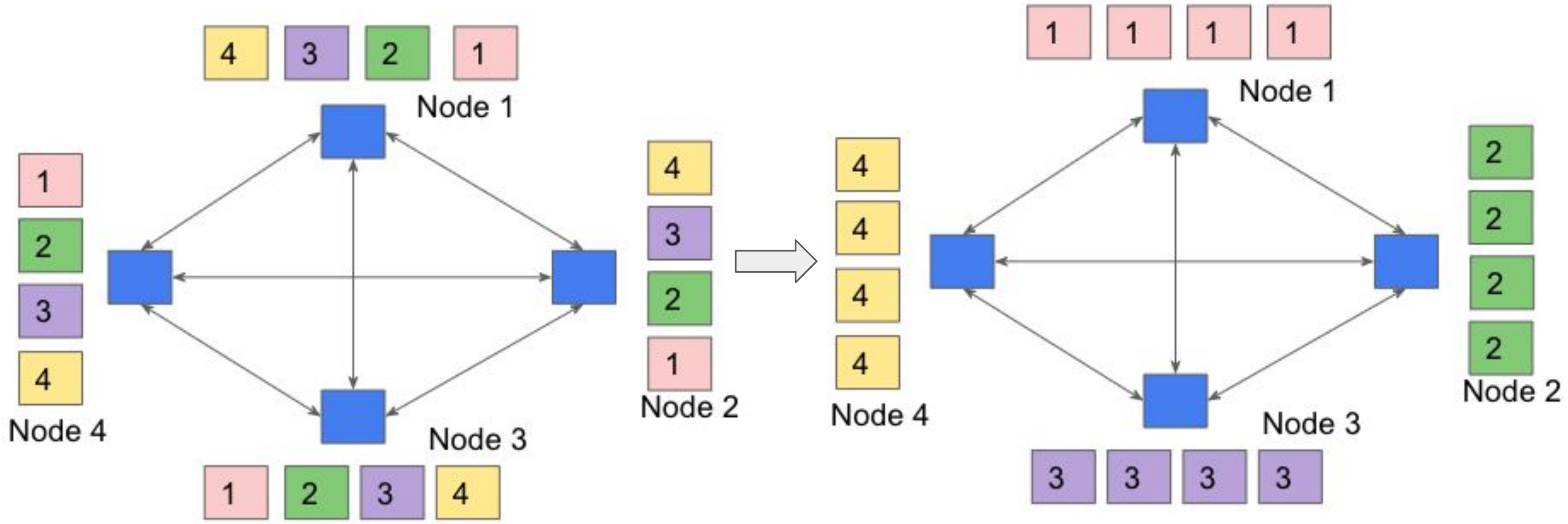
Our contributions

- This is the first work that evaluates the effect of different congestion control schemes on distributed training.
- We developed a simulator using ASTRA-Sim and NS3.
- We provide a detailed analysis of the effect of each state-of-the-art congestion control scheme (i.e., Baseline PFC, DCQCN, DCTCP, Timely, and HPCC) for both single collective communication micro-benchmarks and end-to-end training time of the DLRM workload.
- We found out that different state-of-the-art RoCE congestion control schemes have little impact on the end-to-end training performance.
- Based on our analysis, we provide directions for designing an optimized yet low-overhead congestion control scheme tuned for distributed training.

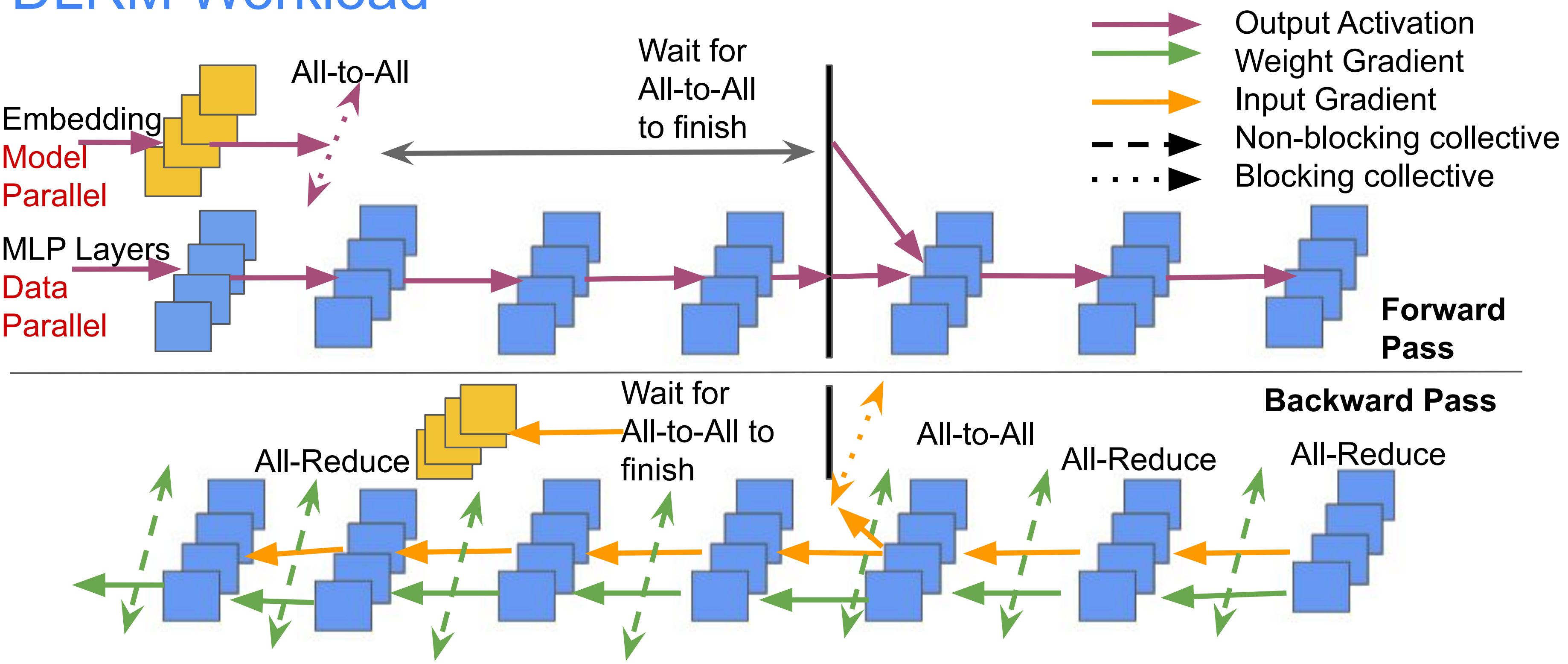
Background: All Reduce Collective Communication



Background: All to All Collective Communication

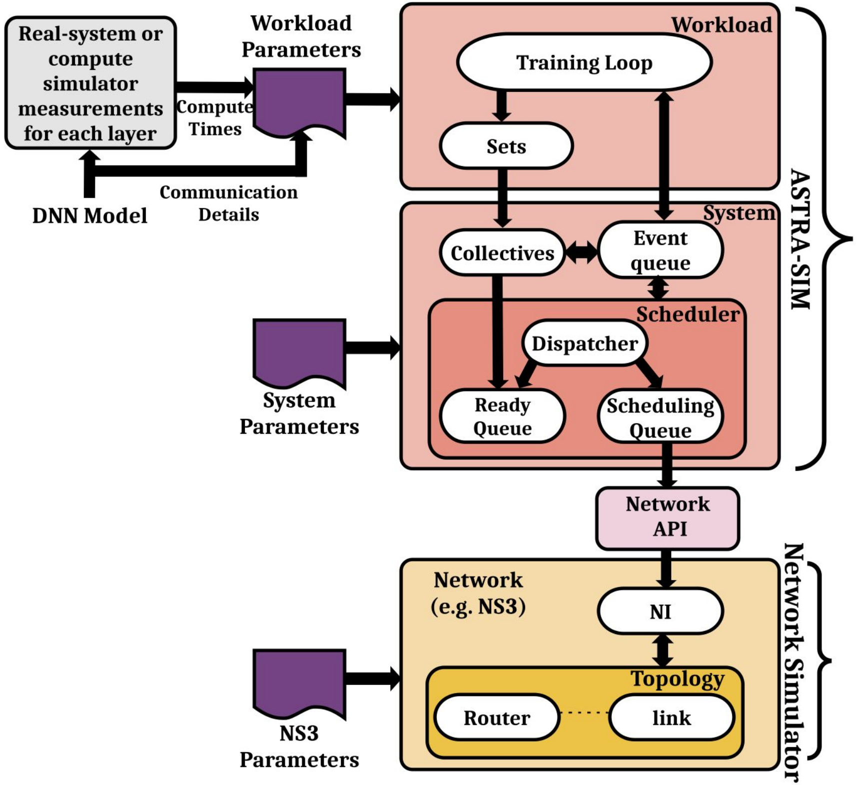


DLRM Workload



ASTRA-Sim

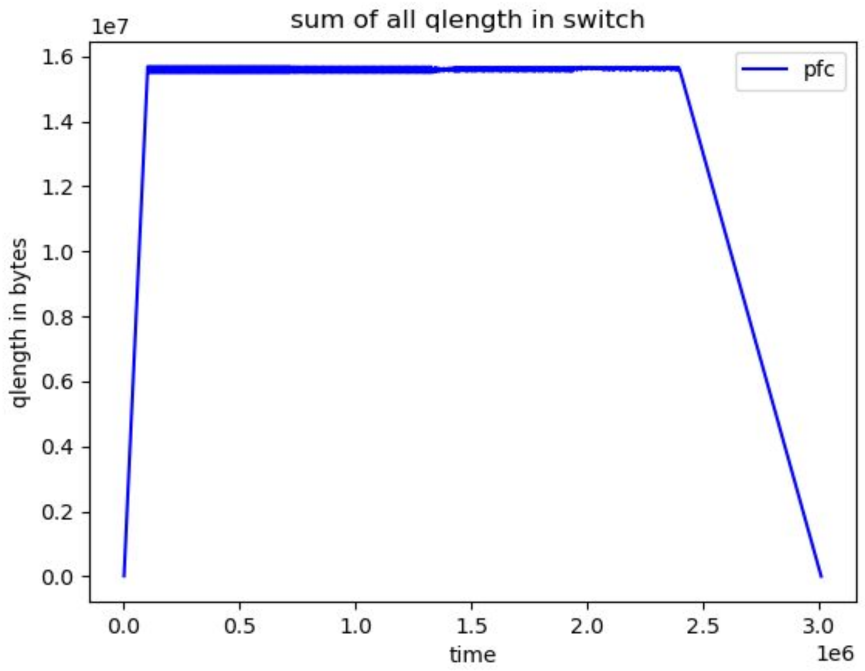
- ASTRA-Sim provides a high level interface to the user to define new DNN models and simulate distributed training on different network topologies and configurations.
- The Simulator generates a detailed analysis regarding the communication behaviour of the workload and the effect of communication overhead over training.



Experimental study: Incast and congestion control

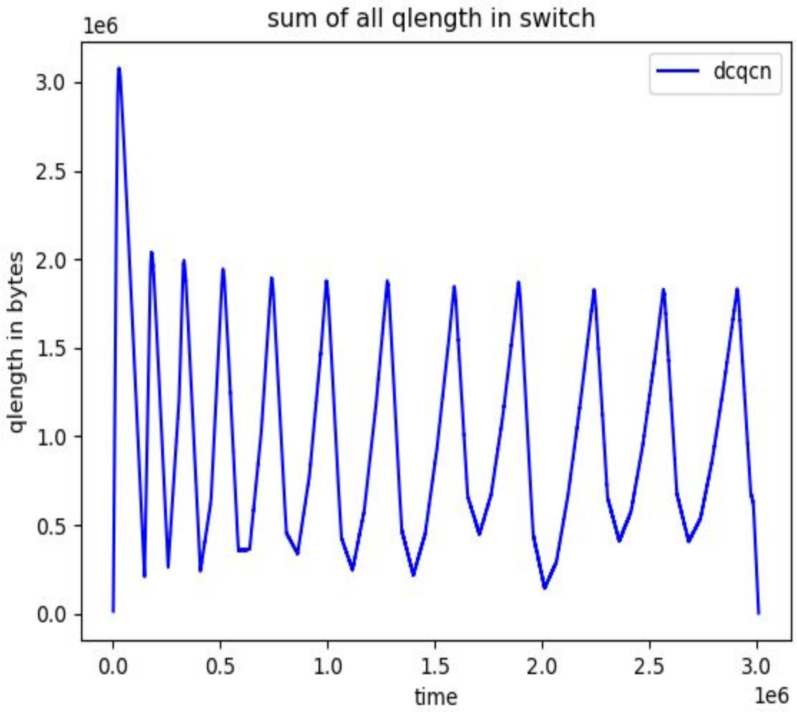
PFC only: Single-switch Incast

- Once the switch queue's threshold is reached for a switch, PFCs are produced.
- Bandwidth is utilised efficiently.
- No compute on the GPU like other CC policies.
- PFCs suffers from several issues - head of line blocking, unfairness, etc.



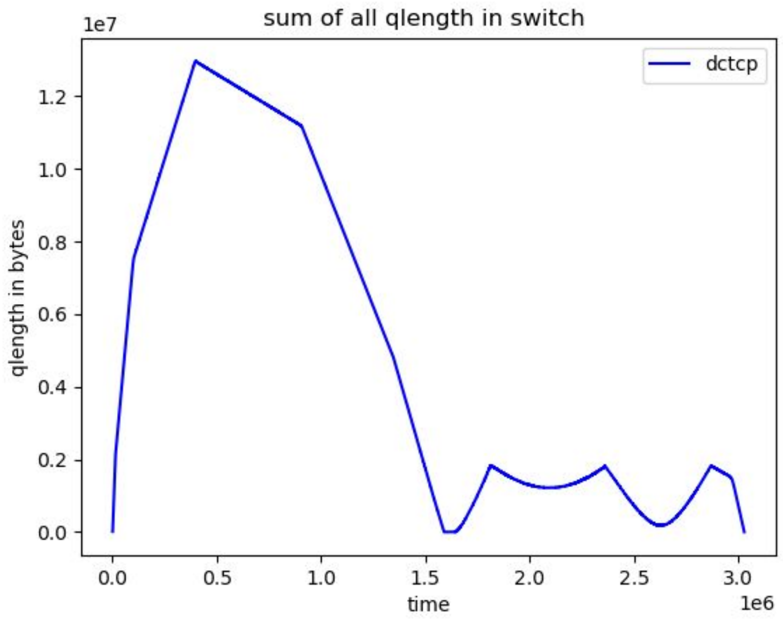
DCQCN: Single-switch Incast

- DCQCN works on CNP packets and accordingly reduces or increases rate.
- No PFCs is produced here.
- Cons:
  - Lot of parameters need to be tuned.
  - Extra time for computation.



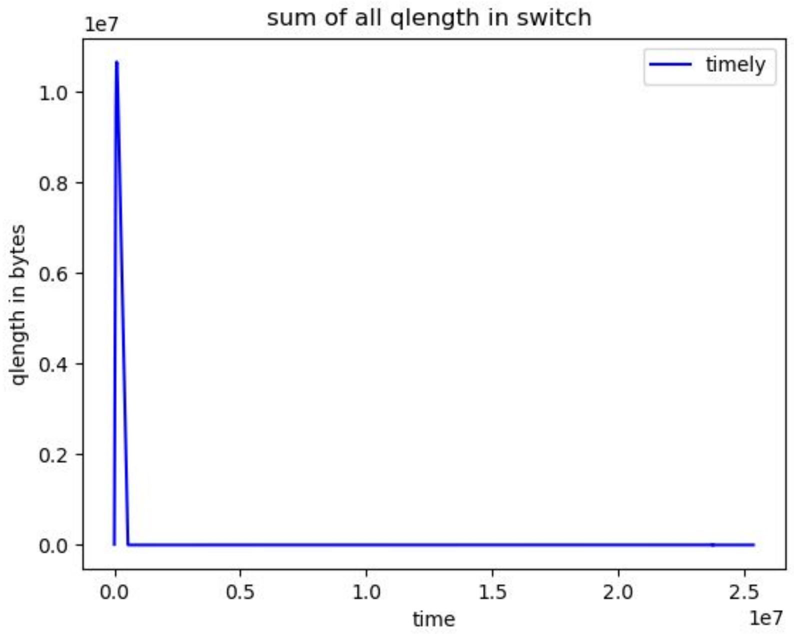
DCTCP: Single-switch Incast

- DCTCP, but at line rate
- DCTCP uses a simple marking scheme at switches that is it marked packets by setting the ECN flag at switches.
- After receiving ECN packets, the window size is reduced.
- No PFCs are being produced.



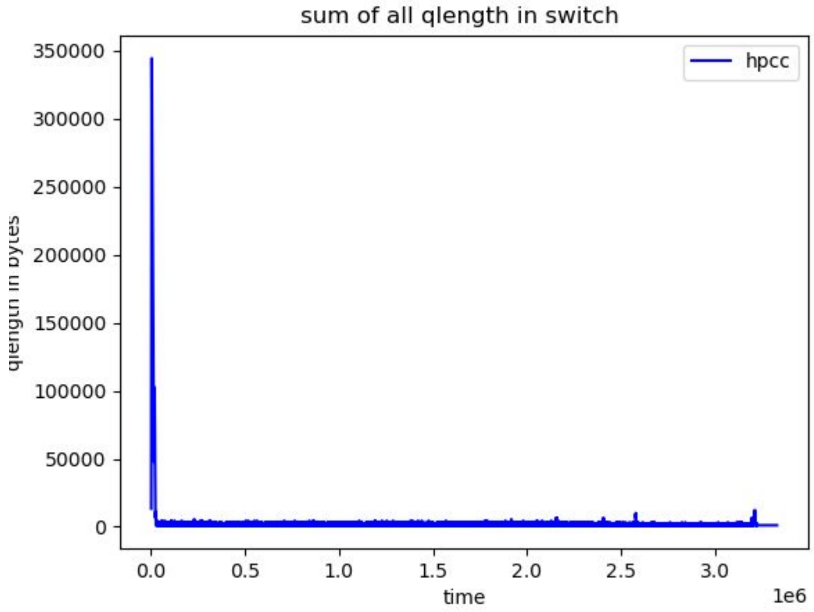
TIMELY: Single-switch Incast

- Based on RTT delay, the rate is increased or decreased.
- Initial rate reduction is sharp as rate is reduced multiplicatively in TIMELY.
- As in all the other gpus, the rate is quickly reduced, the overall switch queue usage is reduced heavily.
- Underutilisation of bandwidth.
- Maximum latency compared to all CC.



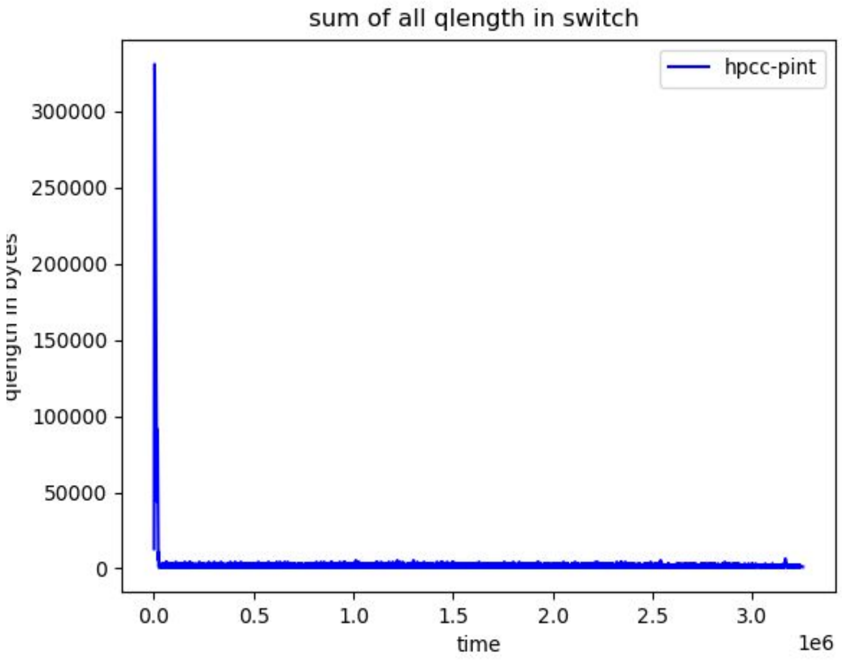
HPCC: Single-switch Incast

- HPCC uses in-network Telemetry (INT) for congestion control.
- It starts reducing the window size, once it starts getting the ACKs and aims to use the minimum queue size.
- At every packet there is an INT overhead as each switch adds this information into the packet.
- Hence, we are actually transferring more data than in other congestion control schemes and this may increase flow completion time.

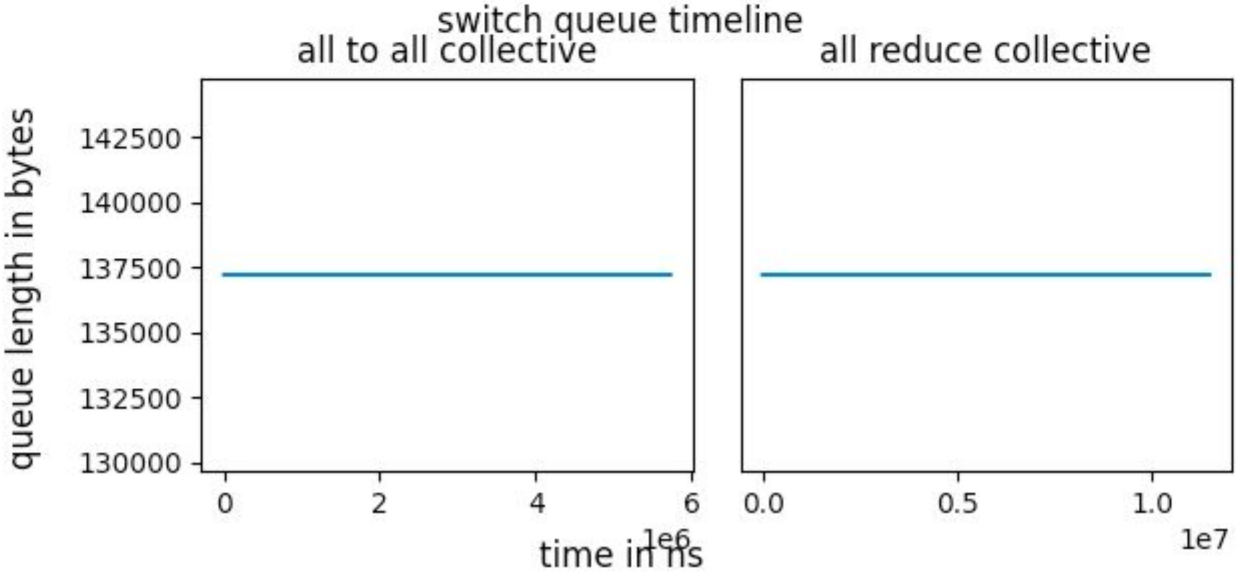


HPCC-PINT: Single-switch Incast

- Solves the HPCC problem of sending extra bytes
- HPCC-PINT does not send per packet feedback, hence feedback can be delayed.
- As focusing on larger flows, HPCC-PINT finishes early.

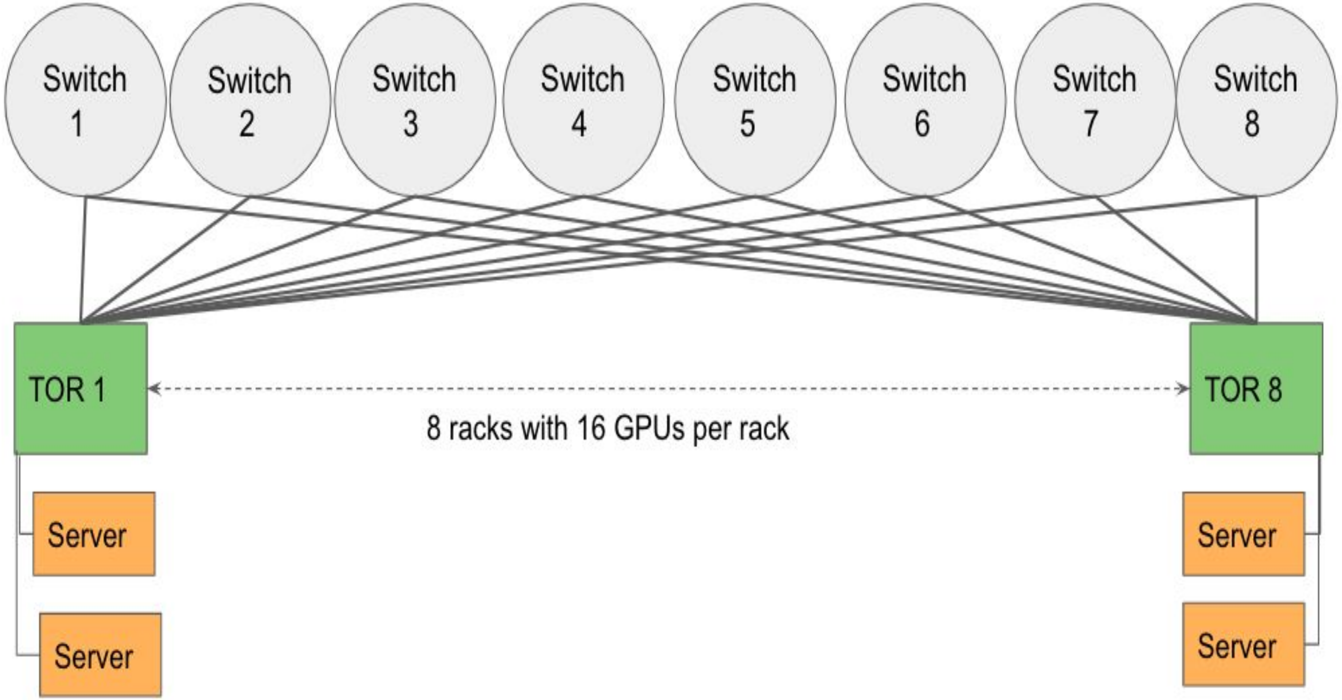


Result: Single-switch Collectives Micro-benchmark

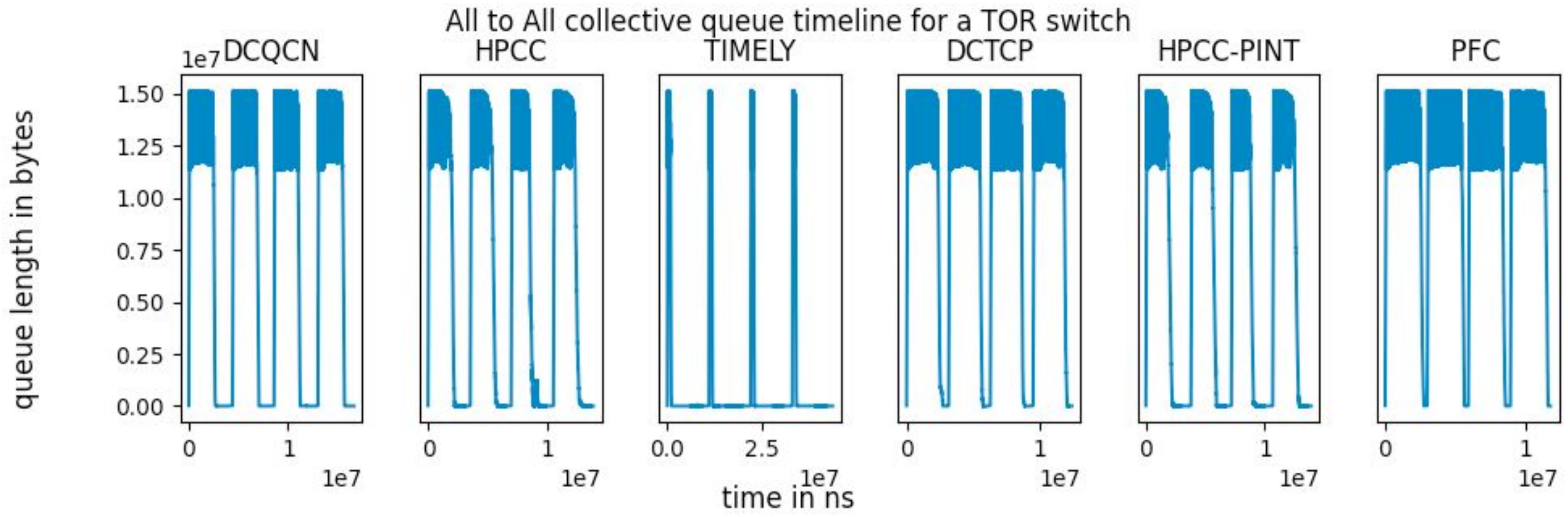


We did not observe any congestion with NPUs connected to a single switch

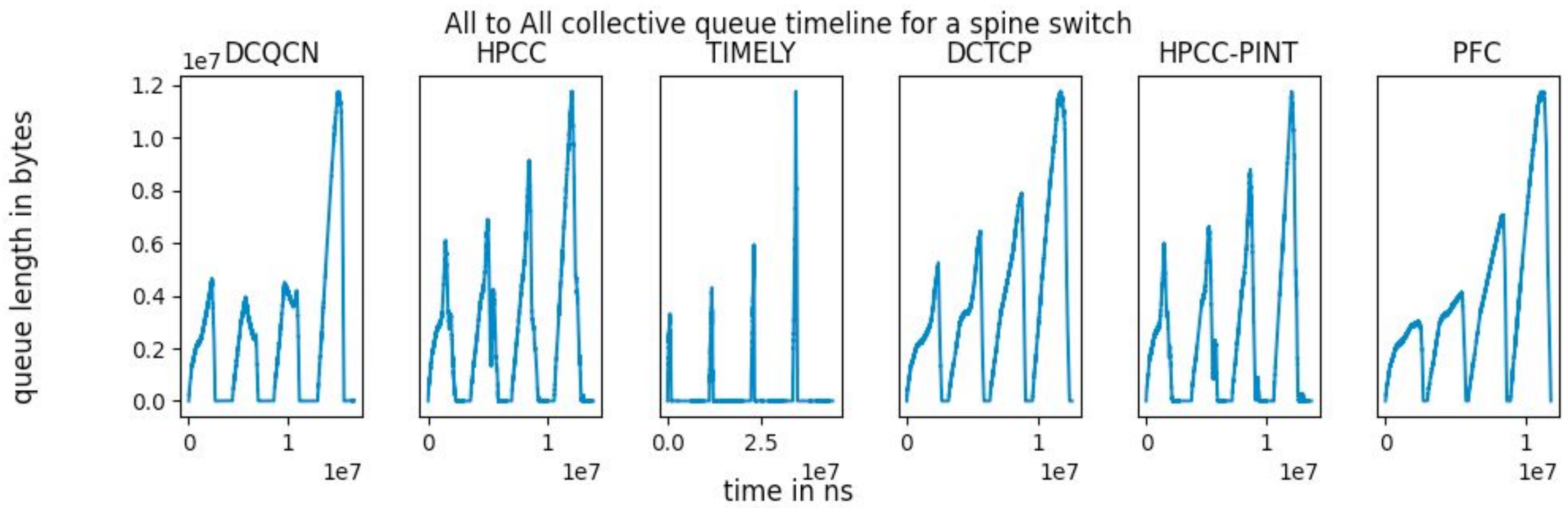
Topology


Switch 1
Switch 2
Switch 3
Switch 4
Switch 5
Switch 6
Switch 7
Switch 8
TOR 1
TOR 8
8 racks with 16 GPUs per rack
Server
Server
Server
Server

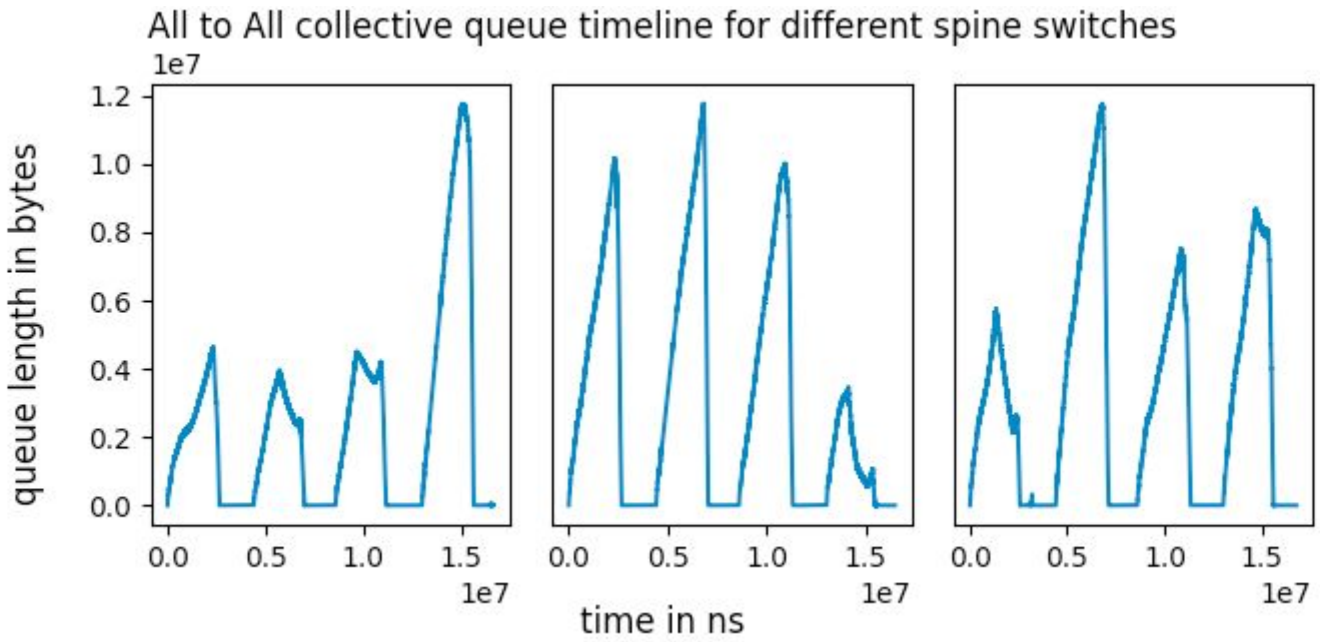
Result: Two-level CLOS topology (TOR switch)



Result: Two-level CLOS topology (Spine switch)

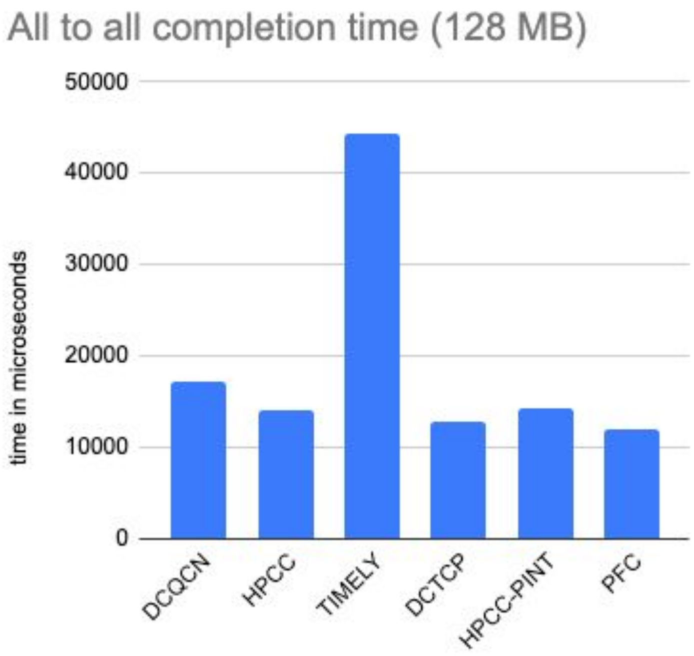


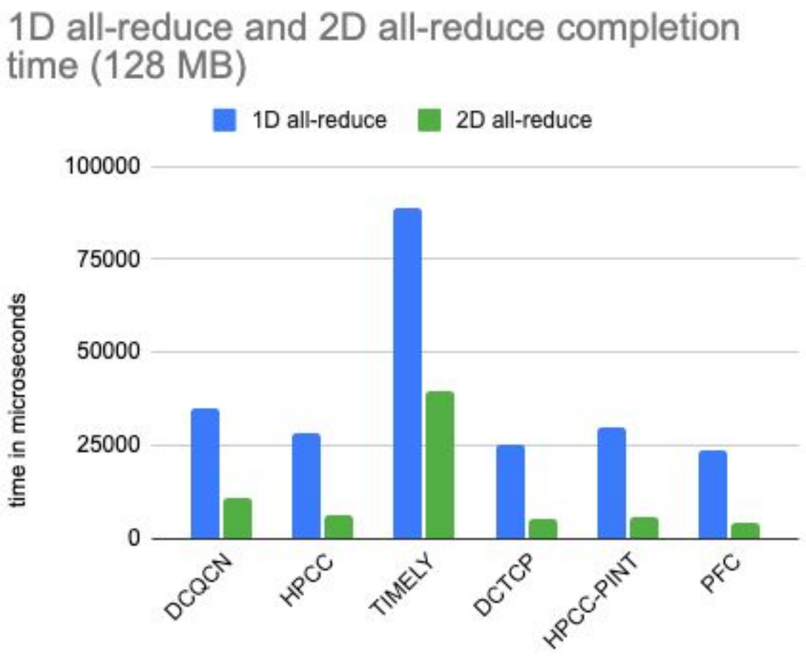
Result: Two-level CLOS topology (Spine switches)



Spine switches have different queue build-ups simultaneously for the same AllTo-All collective flow

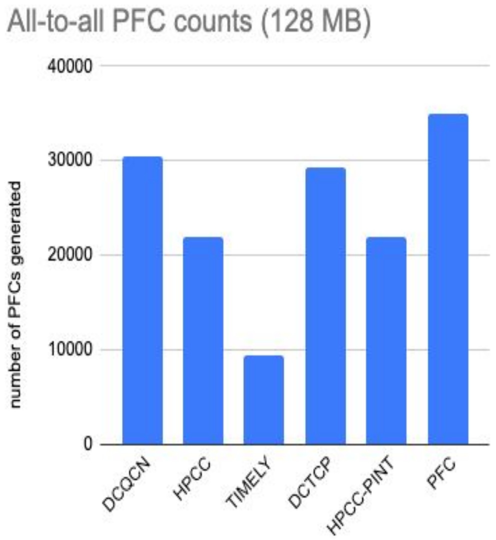
Completion time for collectives

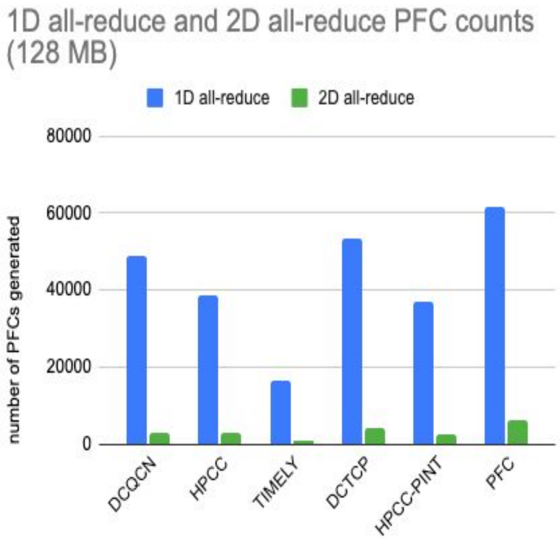


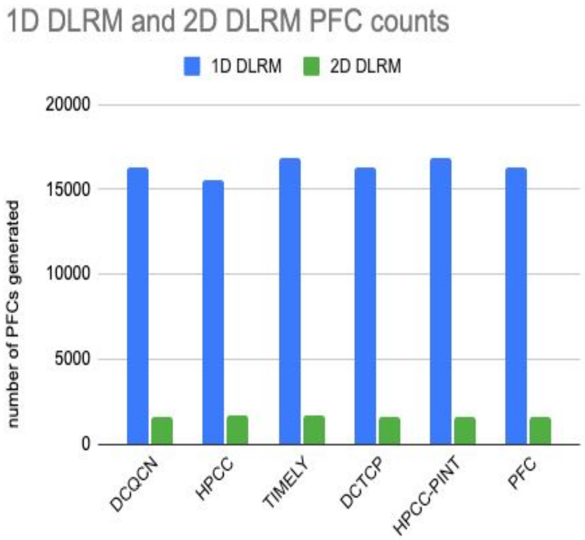


The completion time for Timely is more than any other congestion control algorithm

PFC count for workloads

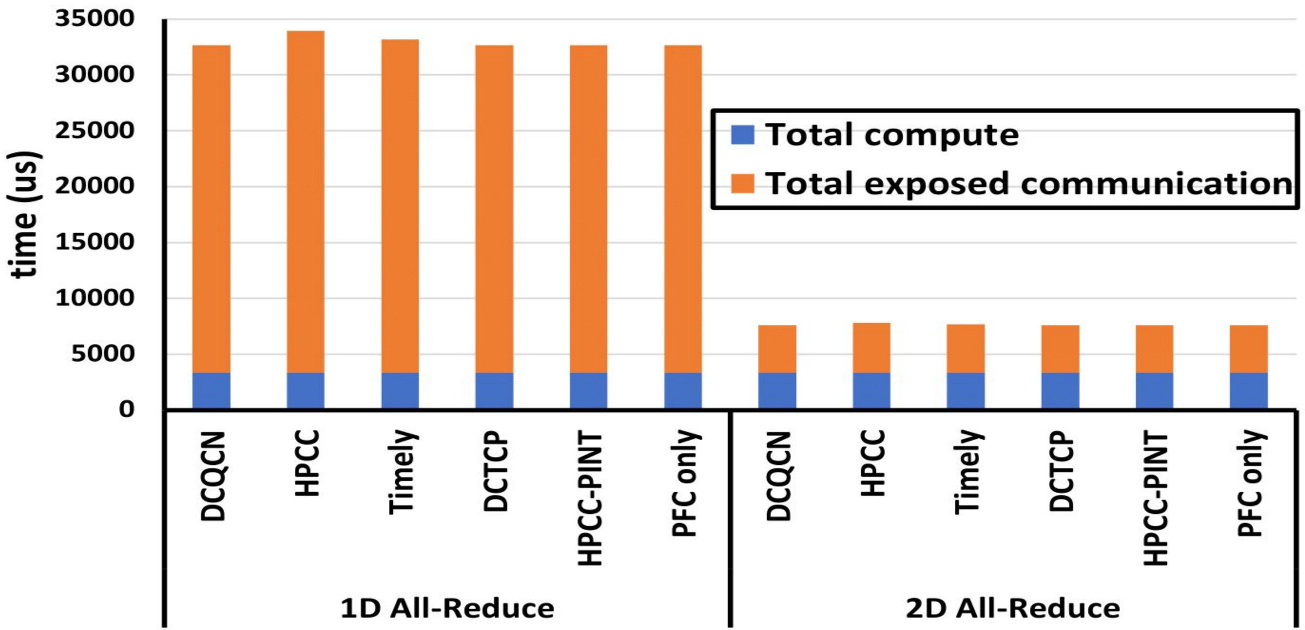






Topology aware collective communication causes less congestion.

Real Workload: DLRM Results



Total exposed communication is comparatively lesser in topology aware collective

Conclusion

- The congestion control algorithm has not much effect in distributed training in specialised distributed training platform.
- The only benefit of proposed CCs over the baseline PFC is that reducing the number of PAUSE frames minimizes the chance of PFC deadlocks that can rarely happen and halt the network.
- The communication patterns of distributed training are deterministic and repeated for each training iteration.
- An optimized CC can be designed which can be very low overhead by leveraging this deterministic communication behavior and setting the congestion window to minimize PFCs.

THANK YOU