Network-Offloaded Bandwidth-Optimal Broadcast and Allgather for Distributed AI

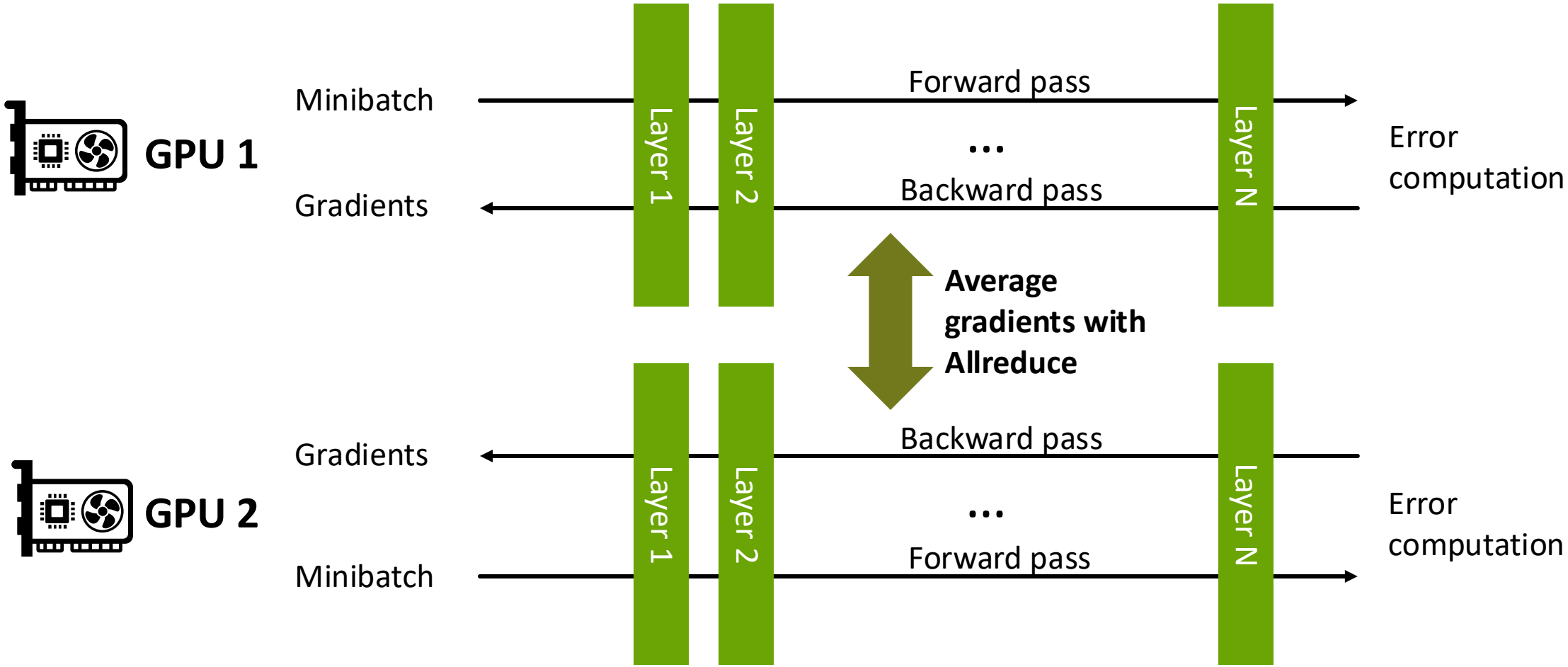
Mikhail Khalilov[1], Salvatore Di Girolamo[2], Marcin Chrapek[1], Rami Nudelman[2], Gil Bloch[2], Torsten Hoefler[1]

[1] ETH Zurich

[2] Nvidia Corporation

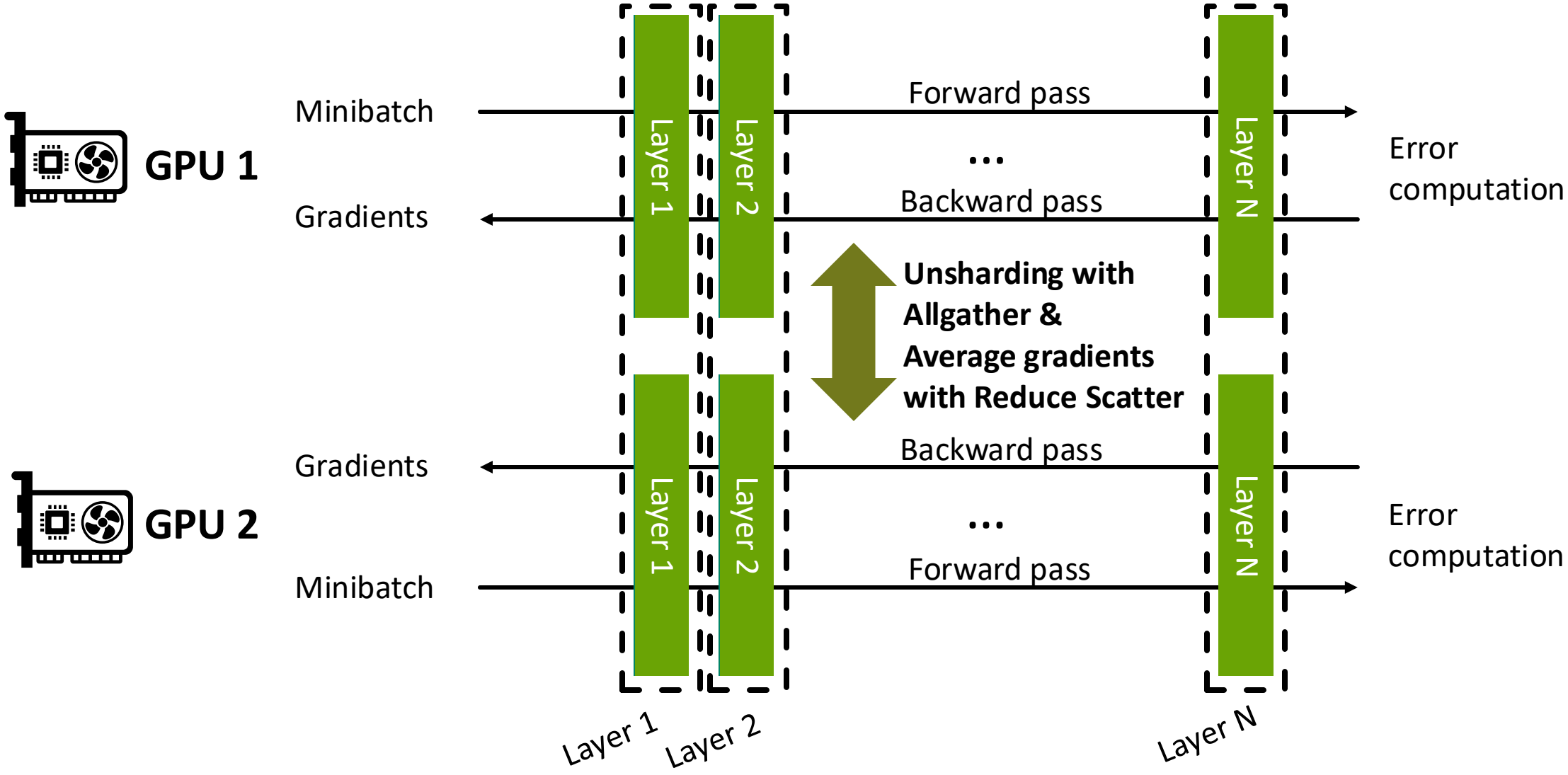
SC24 Atlanta, GA | hpc creates.

Collectives in data parallel training

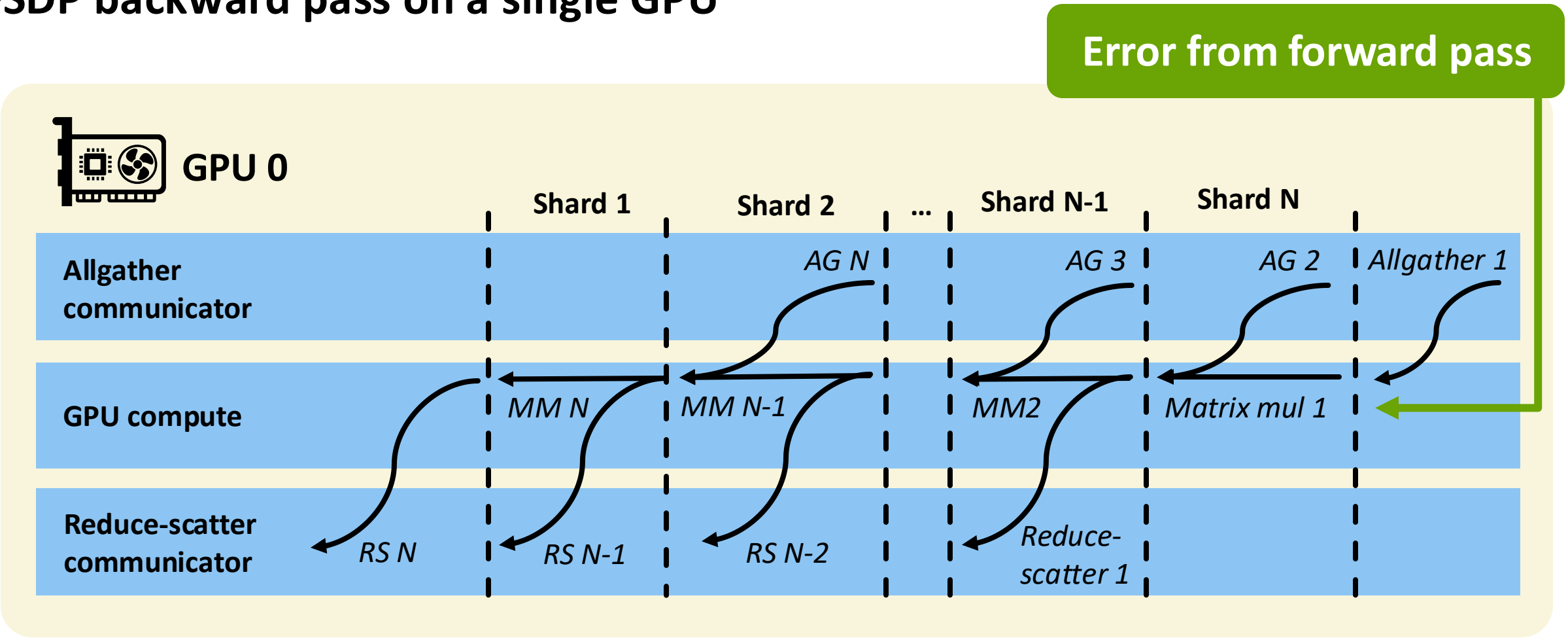


? But GPT4 or Llama 400B are considerably larger than a single GPU?

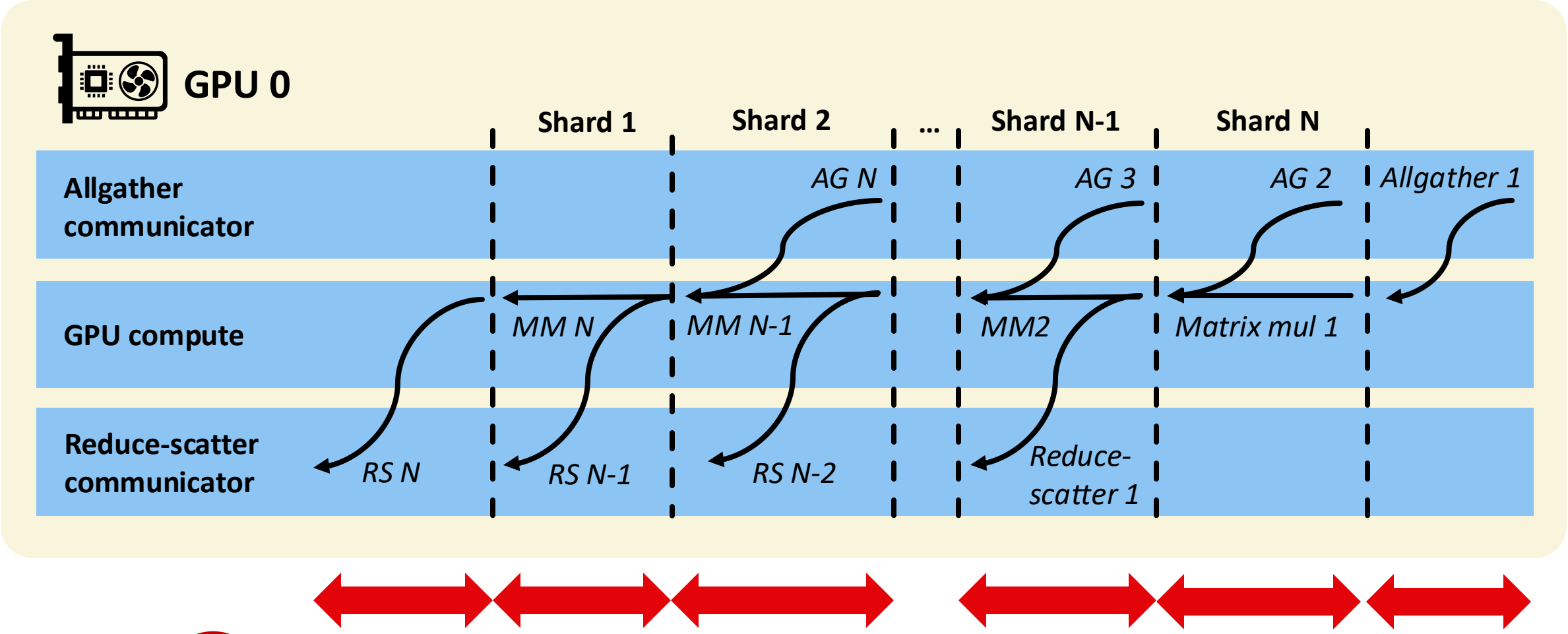
Collectives in Fully Sharded Data Parallel (FSDP) training



FSDP backward pass on a single GPU

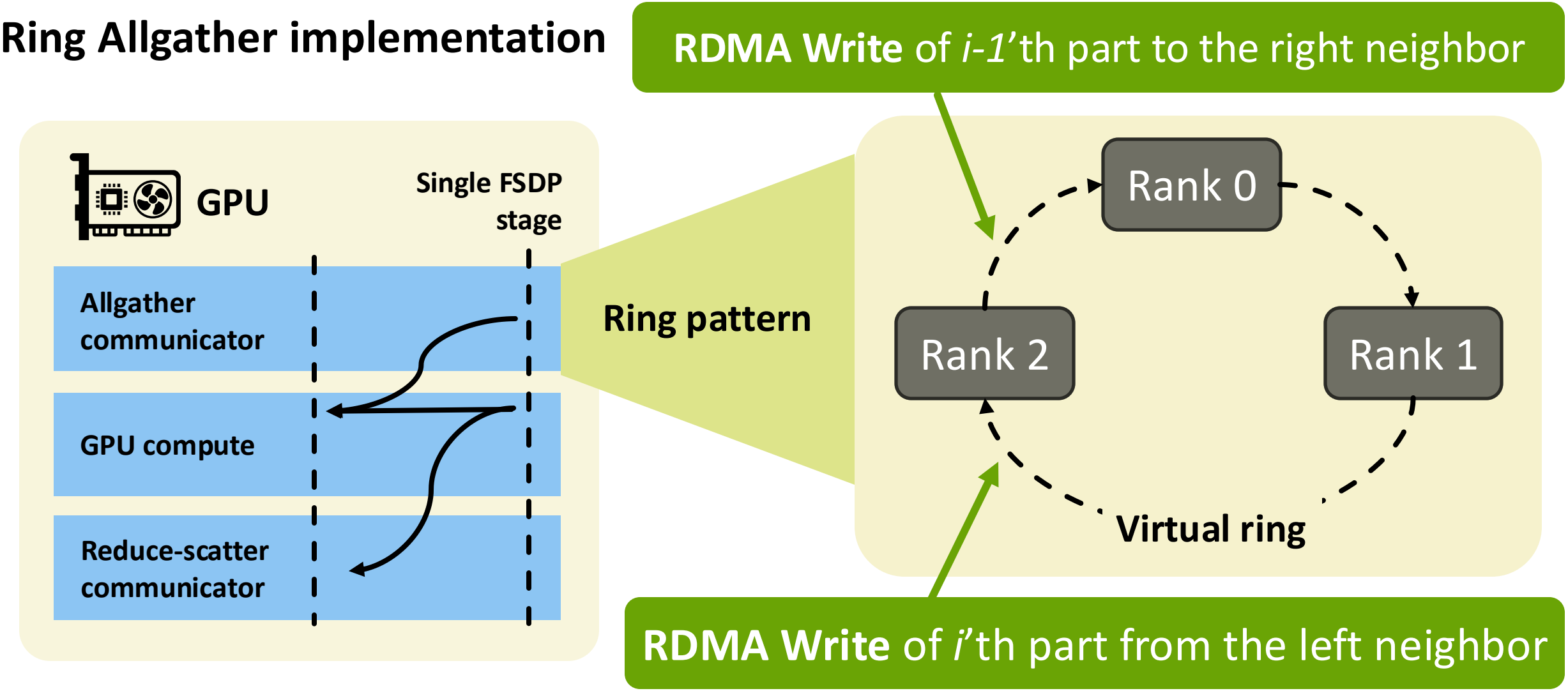


FSDP backward pass on a single GPU



Inter-job traffic and slow collectives can create pipeline bubbles!

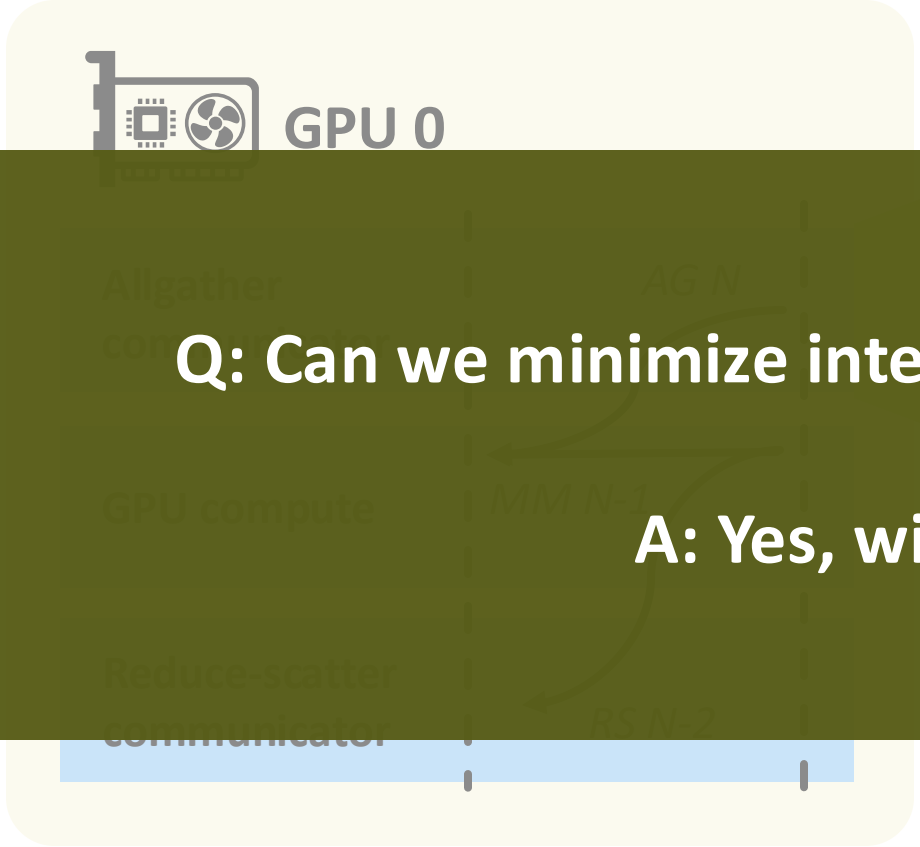
Ring Allgather implementation

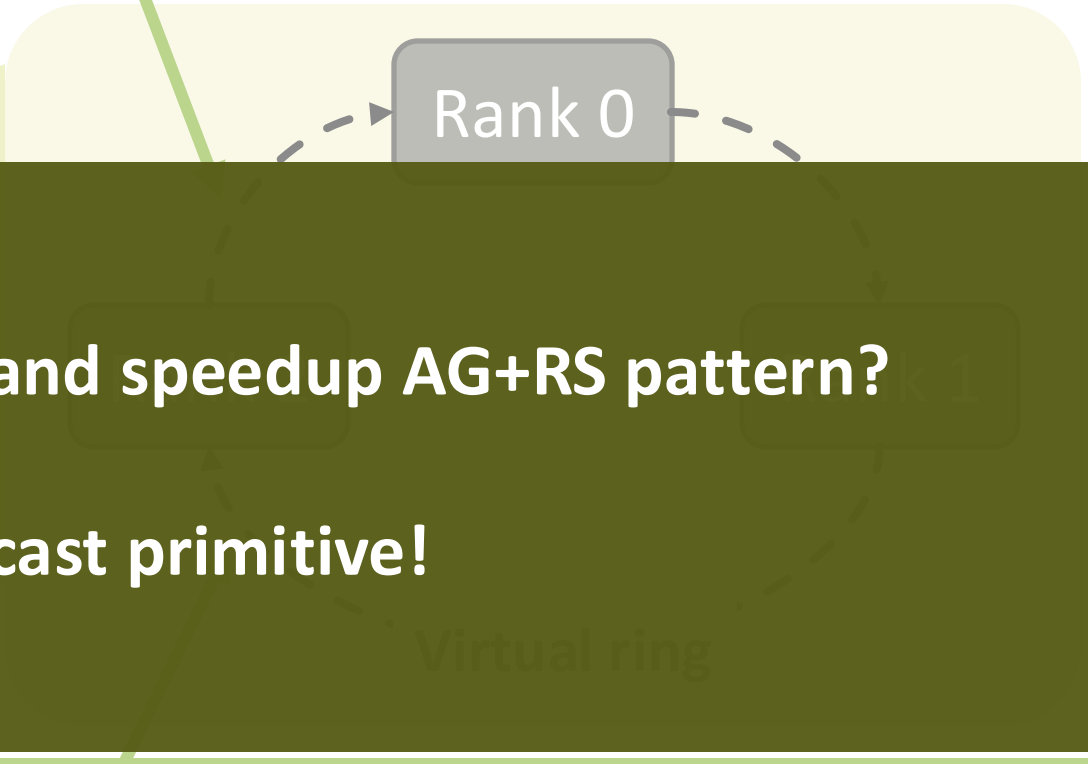


With N bytes in the send buffer and P ranks, N(P-1) bytes are sent *and* received, contending for resources with Reduce Scatter

Ring allgather implementation

RDMA Write of *i-1*'th part to the right neighbor





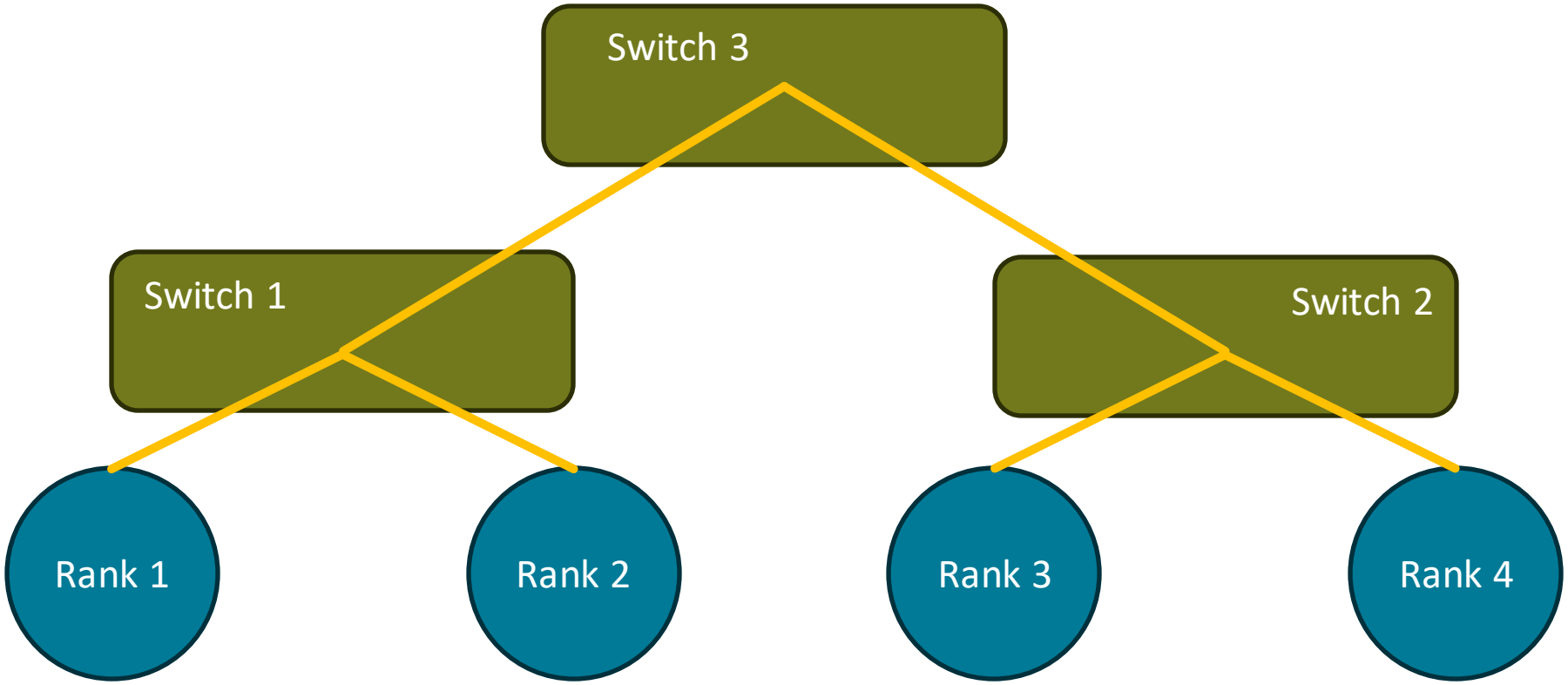
Q: Can we minimize inter-job traffic and speedup AG+RS pattern?

A: Yes, with the multicast primitive!

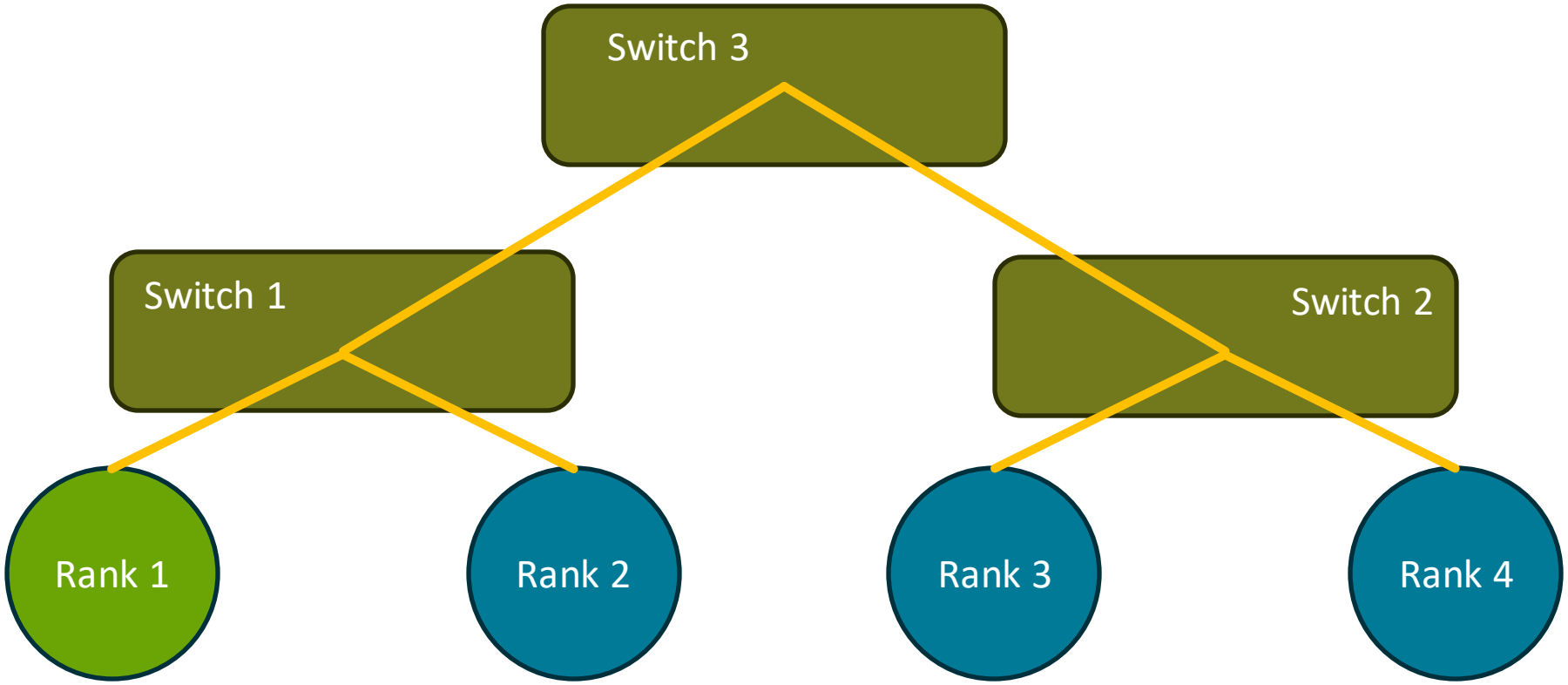
RDMA Write of *i*'th part from the left neighbor

With N bytes in the send buffer and P ranks, N(P-1) bytes are sent *and* received, contending for resources with Reduce Scatter

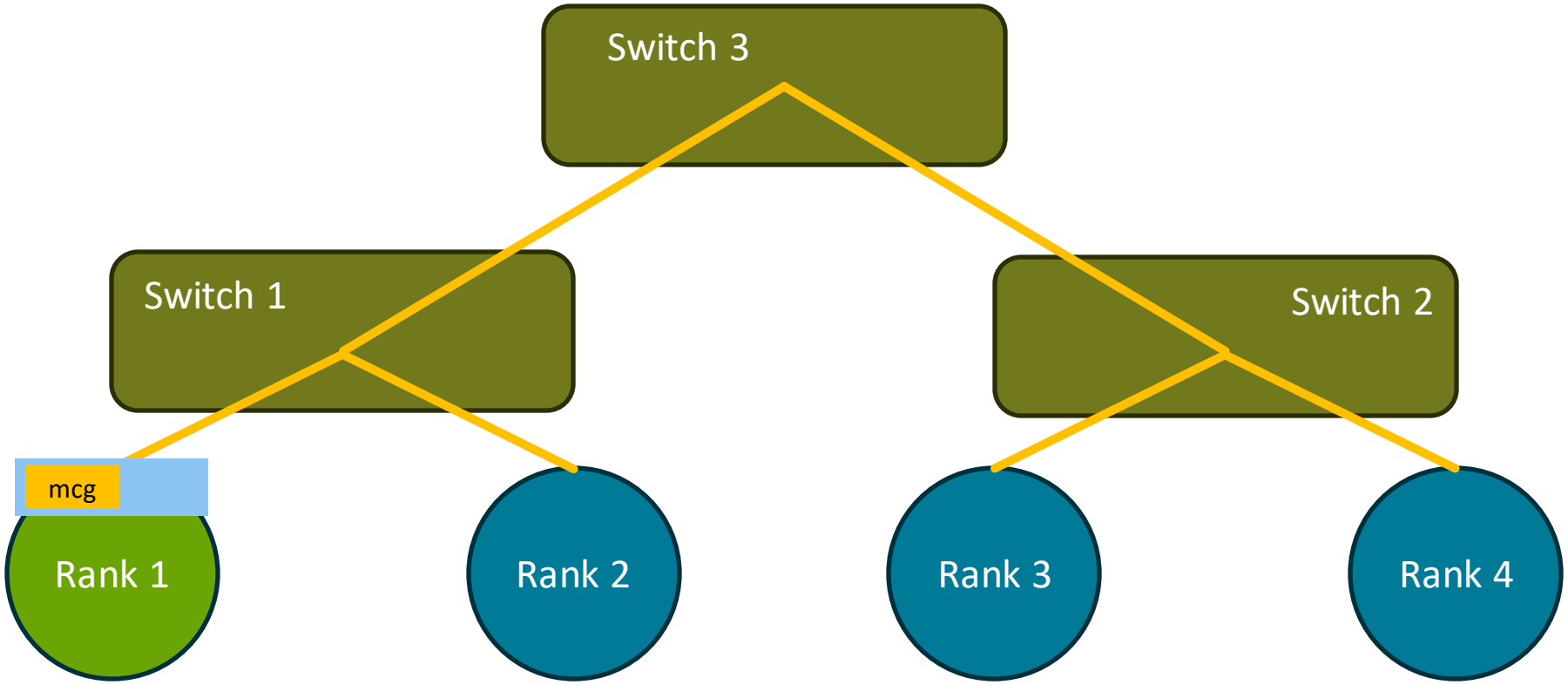
Switch-based multicast



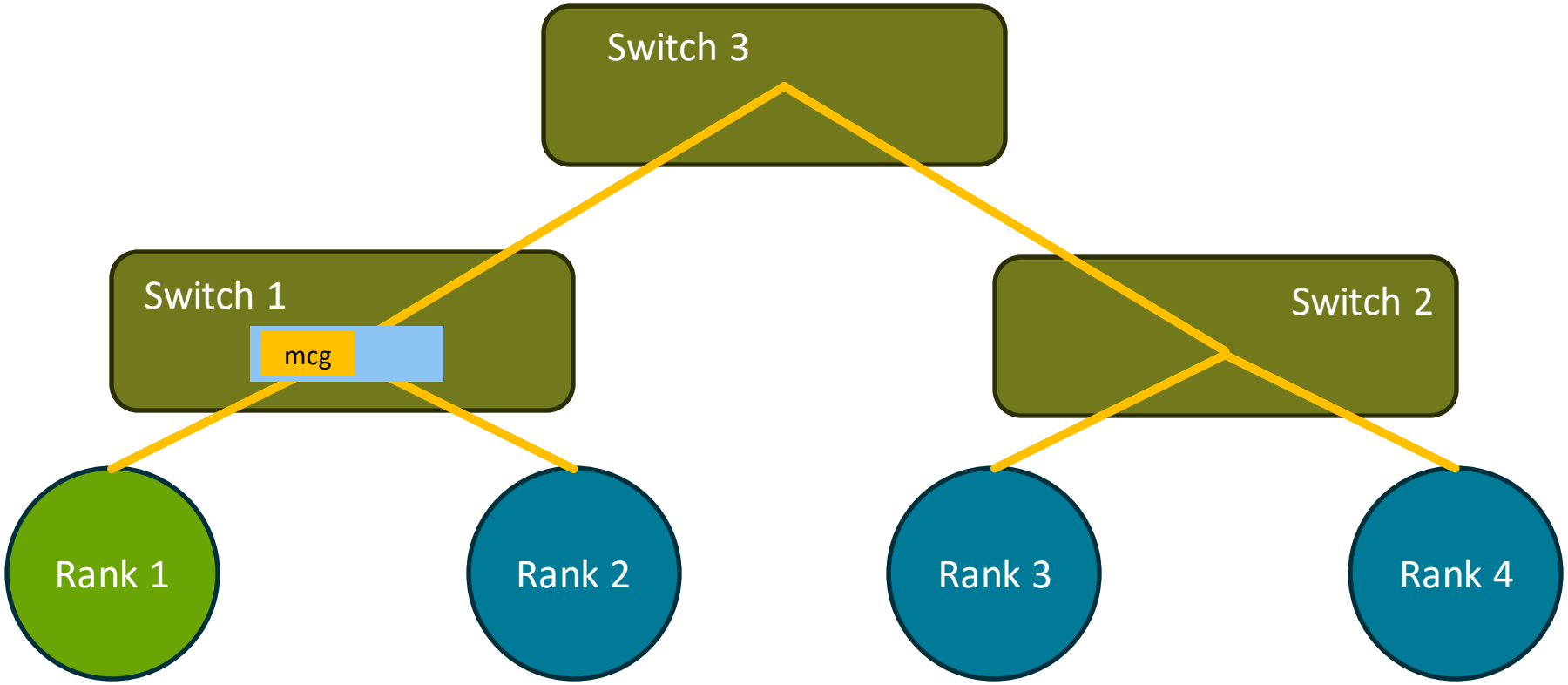
Switch-based multicast



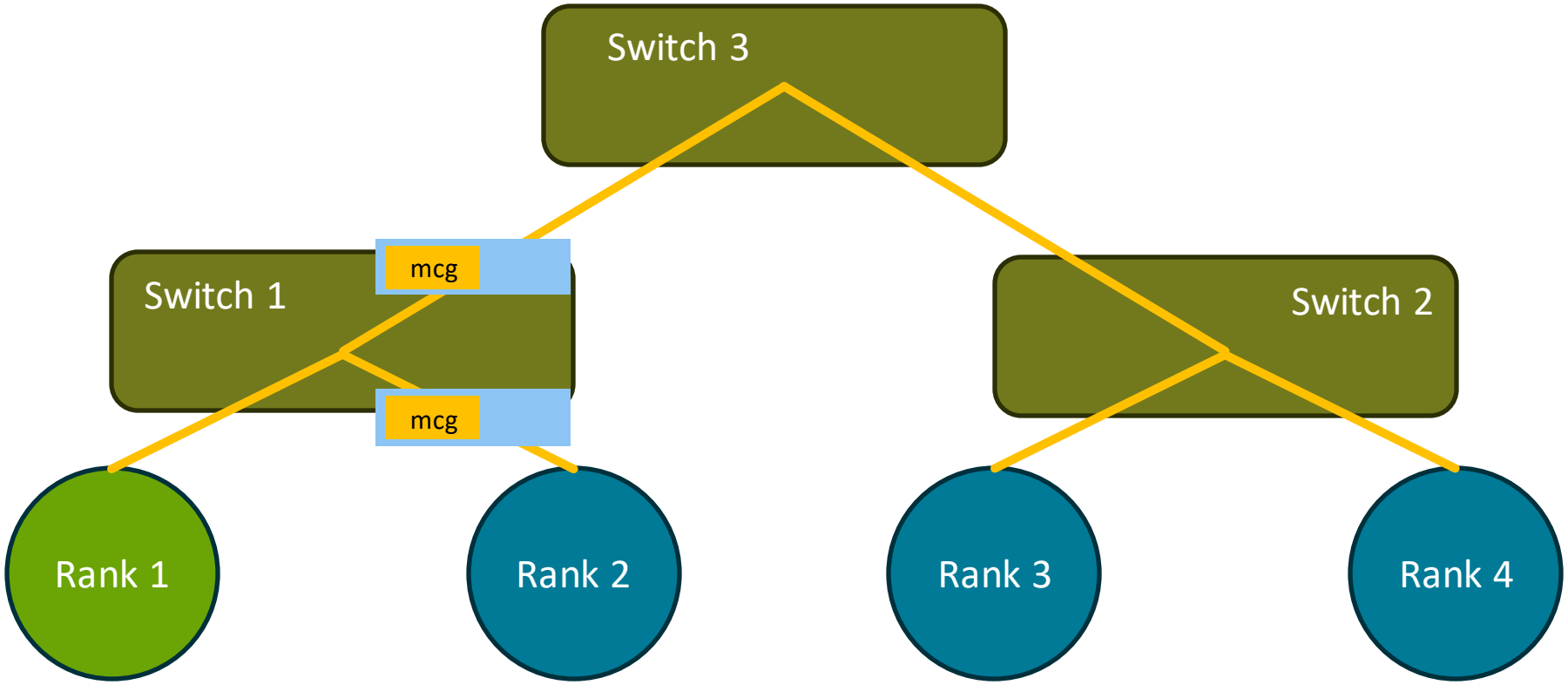
Switch-based multicast



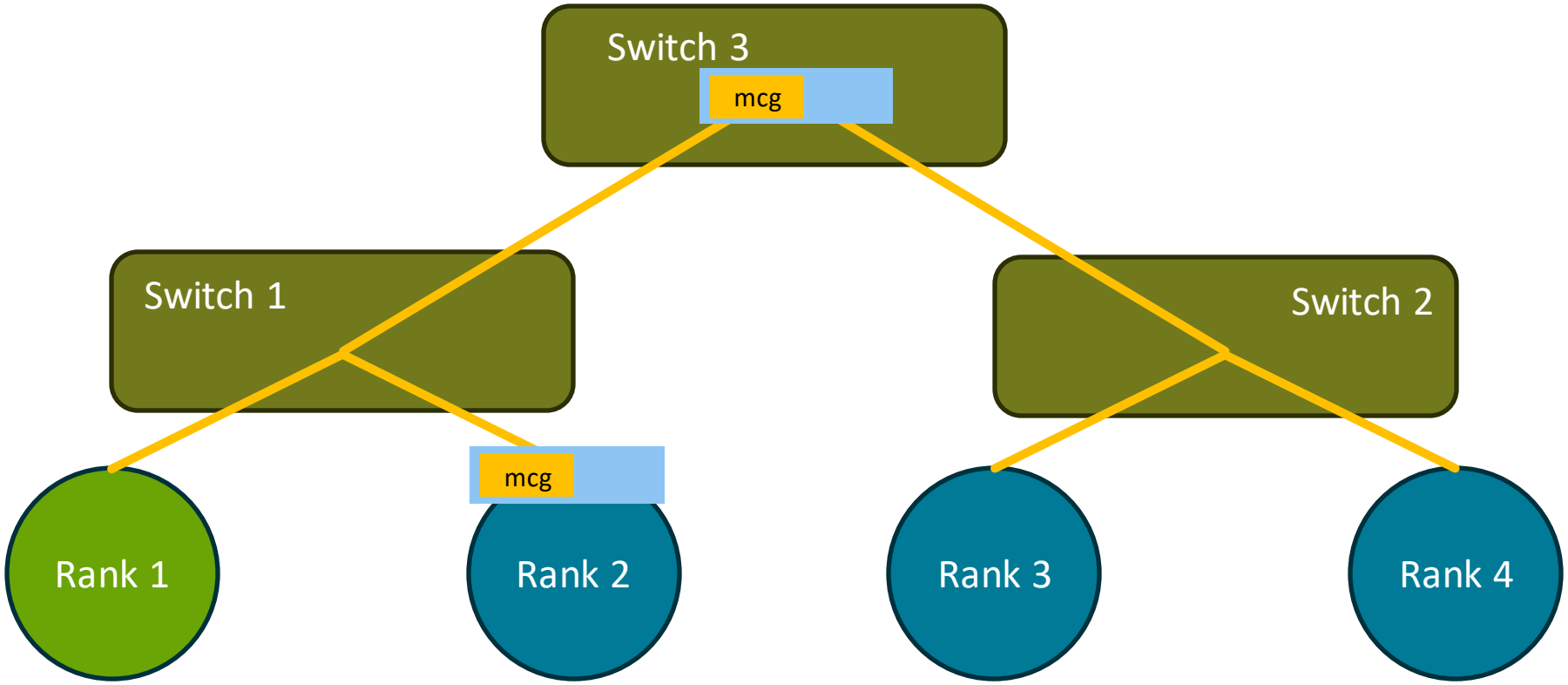
Switch-based multicast



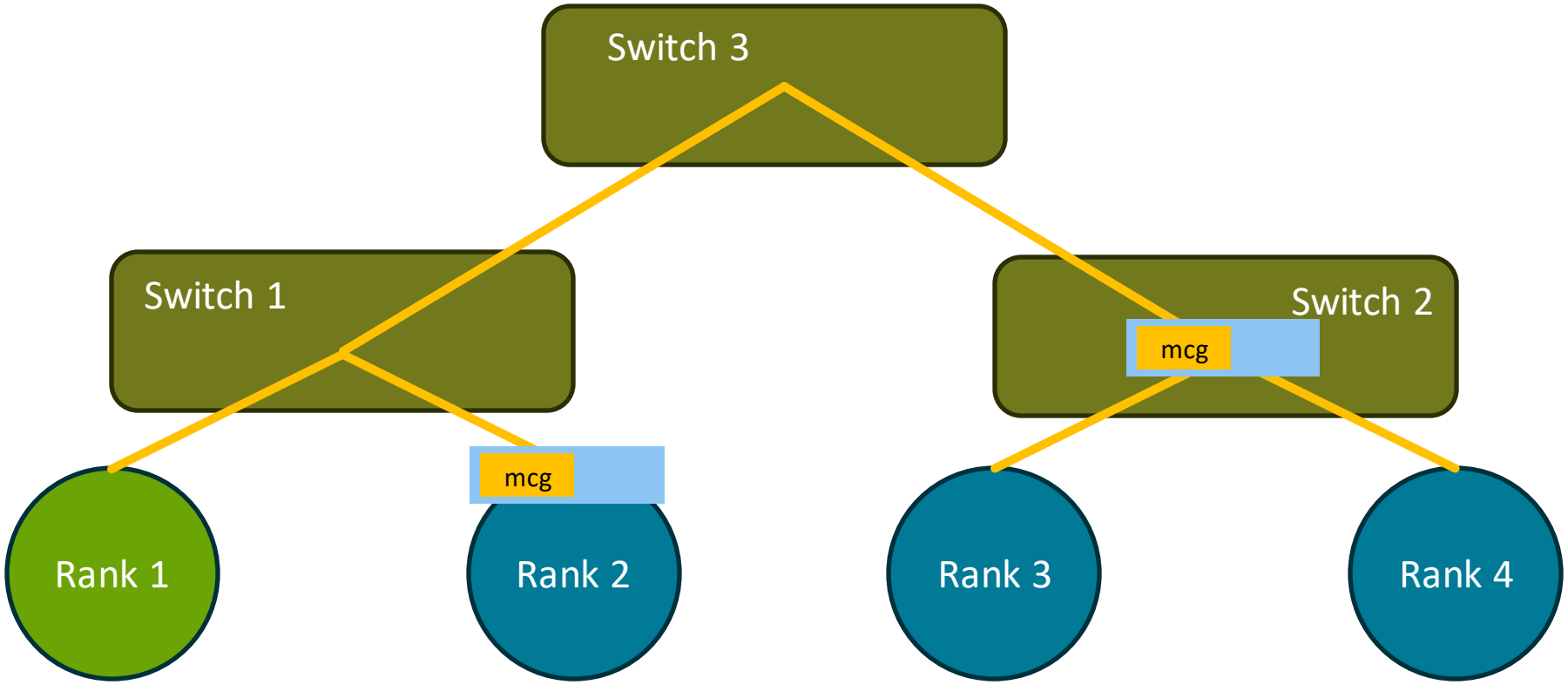
Switch-based multicast



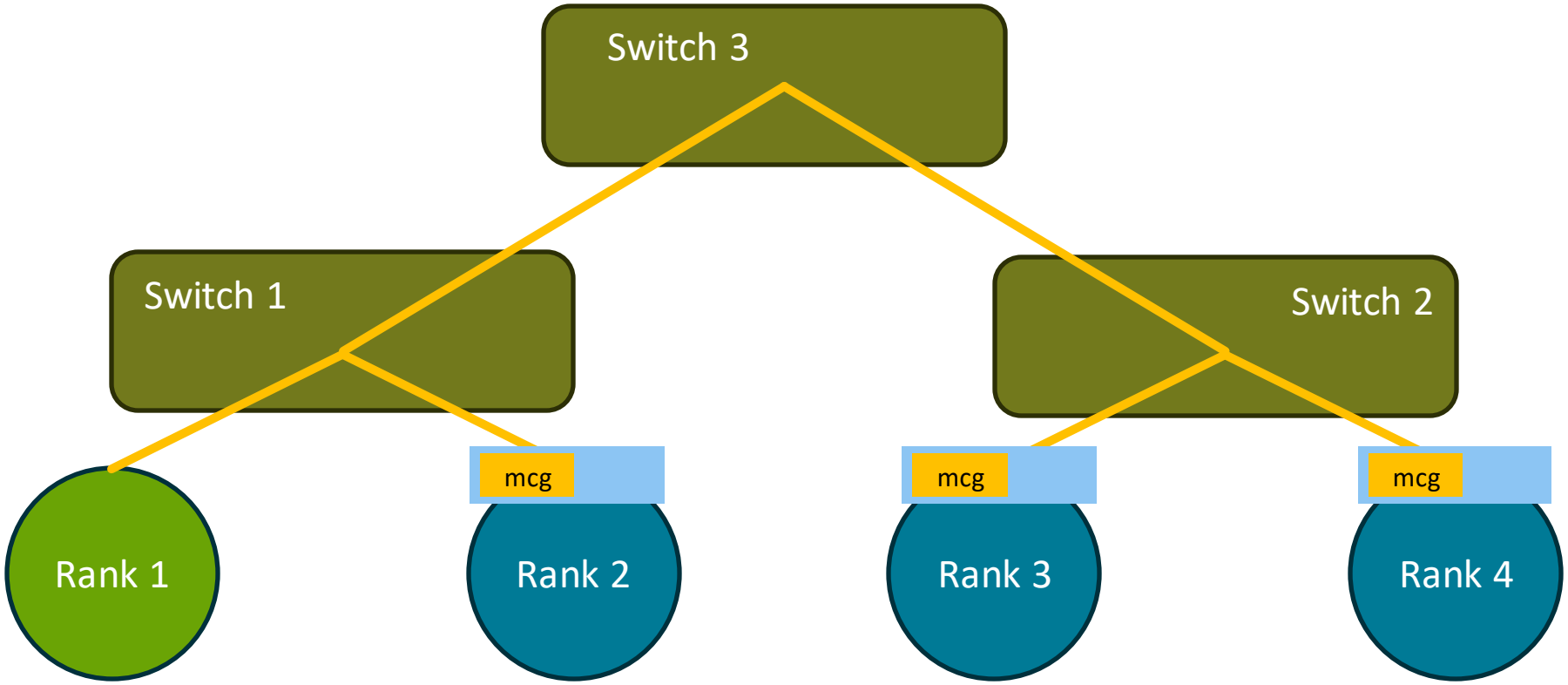
Switch-based multicast



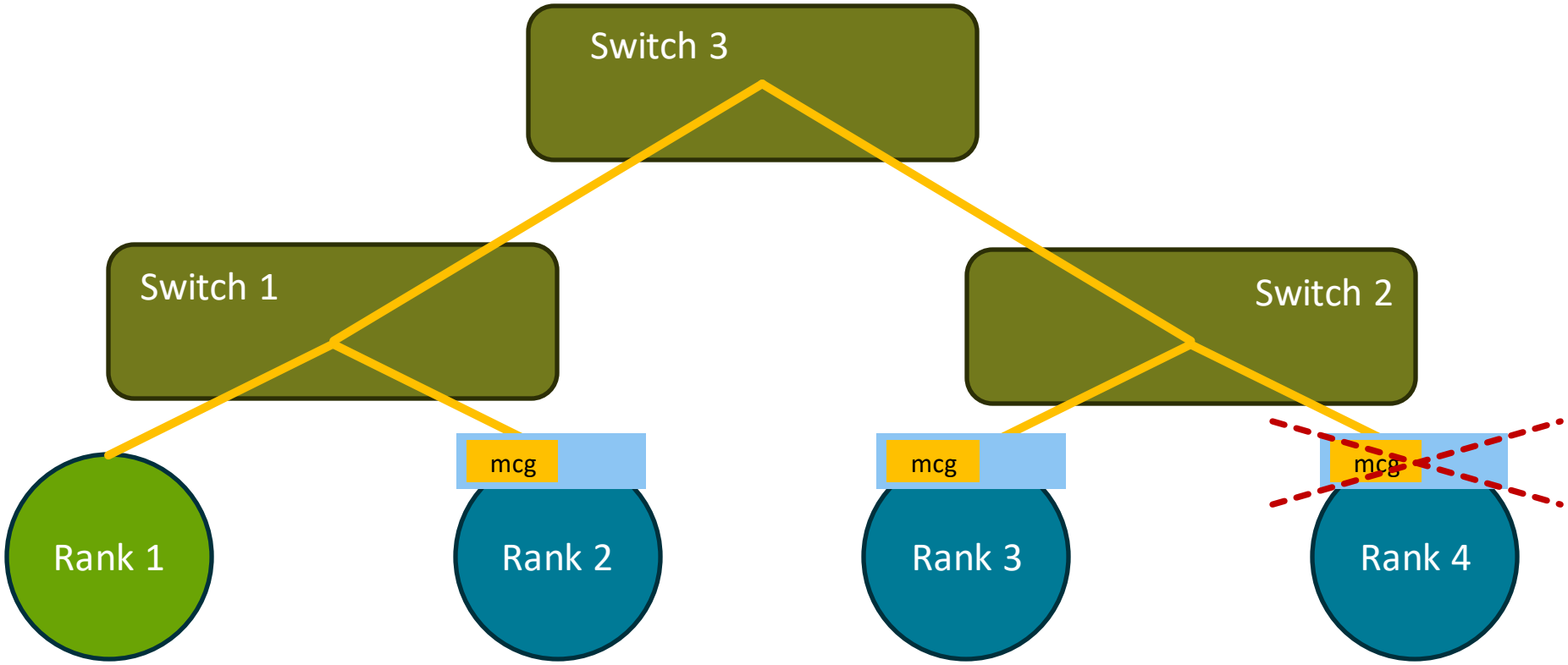
Switch-based multicast



Switch-based multicast



Switch-based multicast



? How to deal with reliable data delivery?

Switch-based multicast

Fast MPI Broadcasts through Reliable Multicasting

Paul Sack[1]* and An

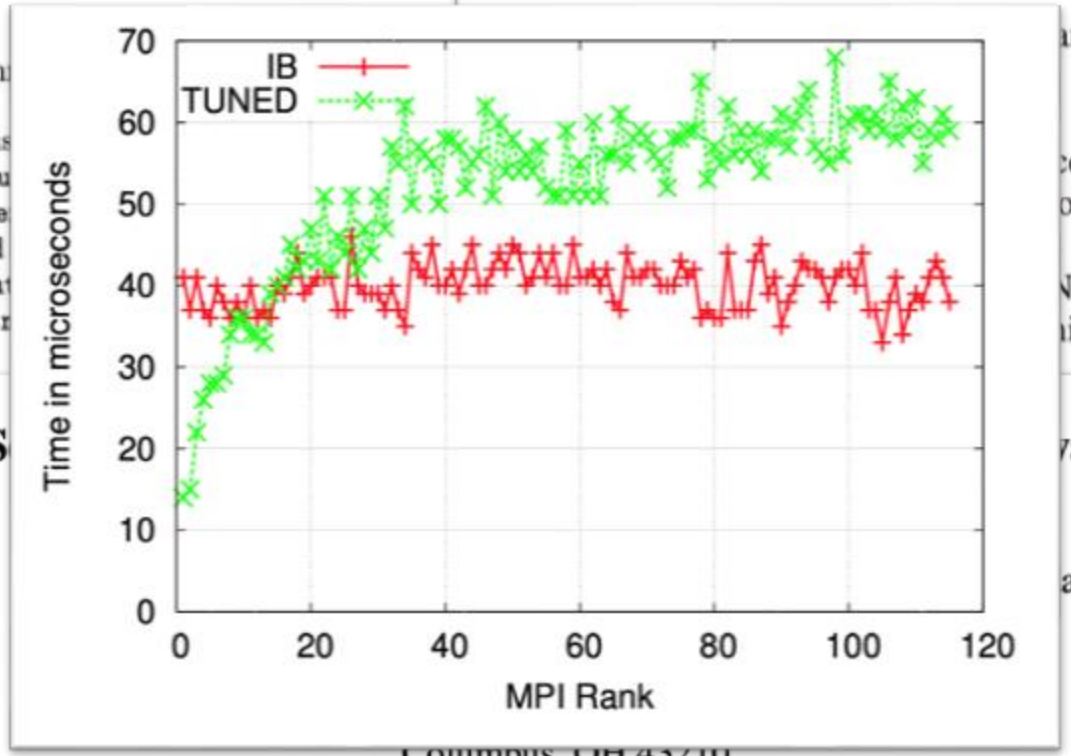
[1] University of Texas at Aus
paulsack@mail.u
http://www.ece.ute
[2] Norwegian University of Science and
elster@comput
http://www.idi.ntr

A practically constant-time MPI Broadcast Algorithm for large-scale InfiniBand Clusters with Multicast

an Siebert,[1] and Wolfgang Rehm[1]

ce
ology
NY
itz.de

[2]Open Systems Laboratory
Indiana University
501 N. Morton Street
Bloomington, IN 47404 USA
htor@cs.indiana.edu



Fast and S

are Multicast

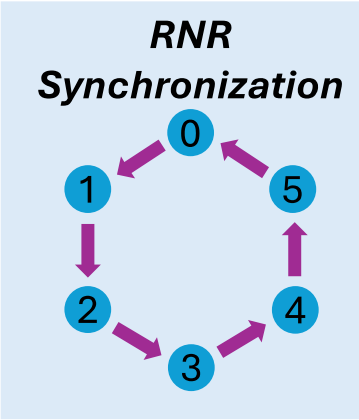
a

Columbus, OH 43210
{liuj, mamidala, panda}@cis.ohio-state.edu

Plot source: *A practically constant-time MPI Broadcast Algorithm for large-scale InfiniBand Clusters with Multicast Torsten Hoefler, Christian Siebert, and Wolfgang Rehm*

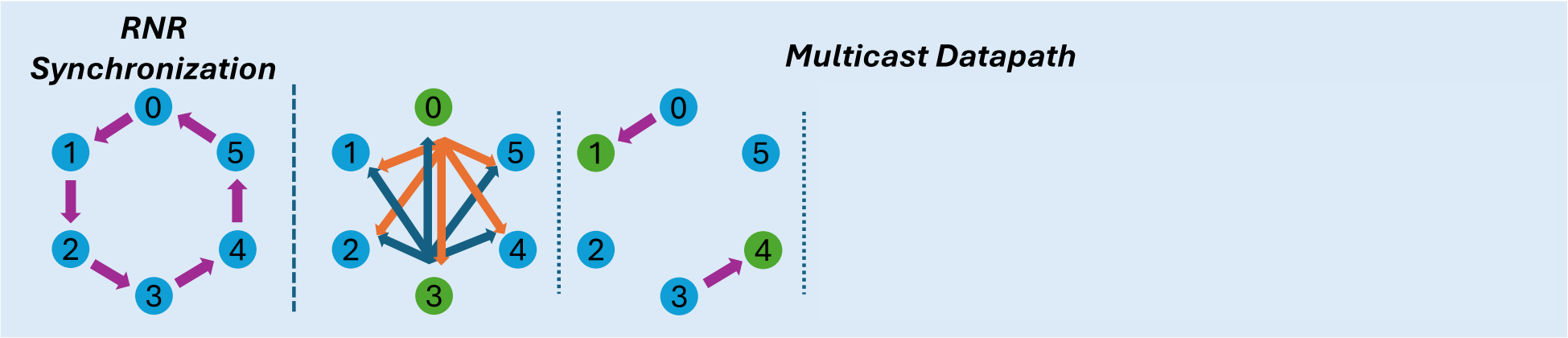
Allgather as a composition of Broadcasts

Stage 1
Ranks post buffers and perform barrier



Allgather as a composition of Broadcasts

Stage 1
Ranks post buffers and perform barrier

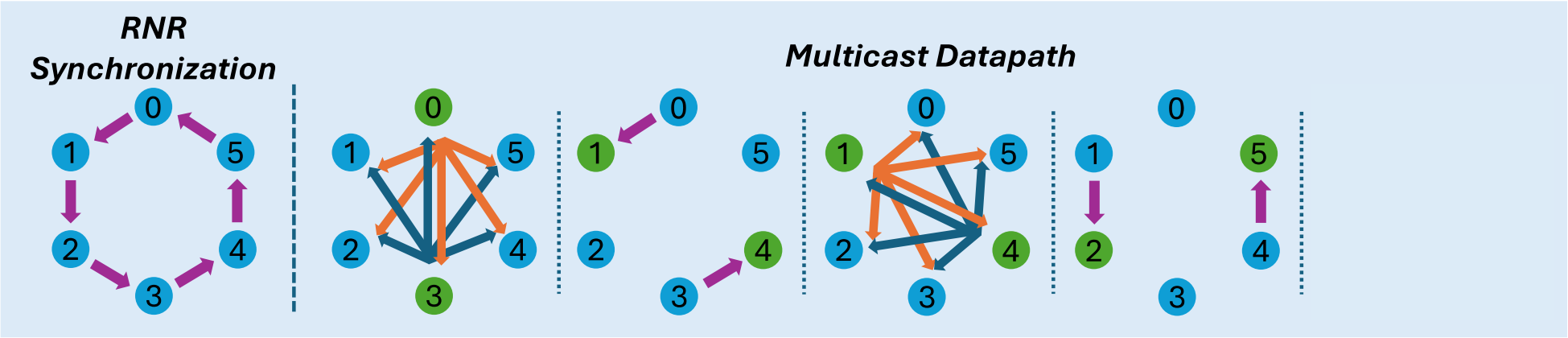


Stage 2.1
Groups of simultaneously multicasting ranks distribute data

Stage 2.2
Completed groups pass the activation signals to the next groups

Allgather as a composition of Broadcasts

Stage 1
Ranks post buffers and perform barrier

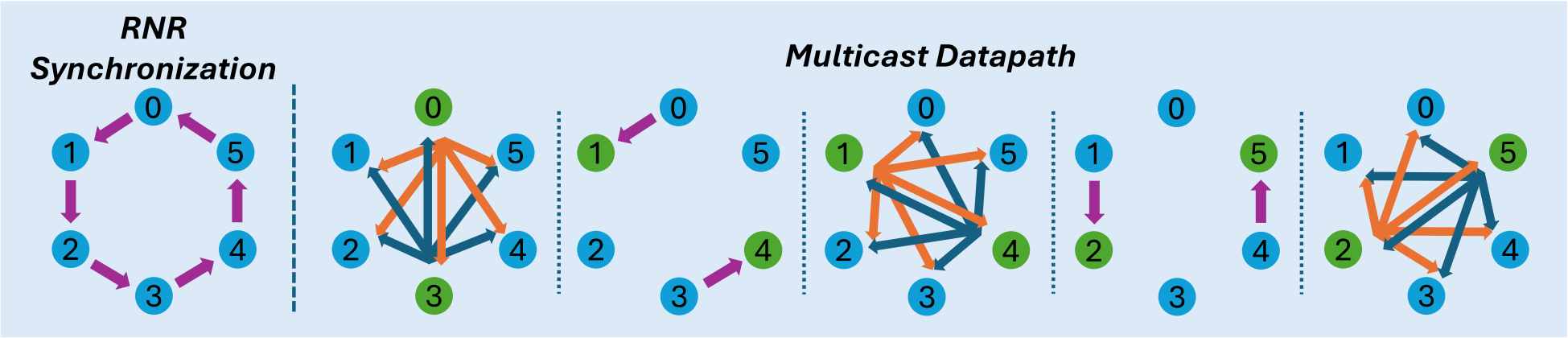


Stage 2.3
The second multicast group distributes data

Stage 2.4
And then passes activation signals to the next groups

Allgather as a composition of Broadcasts

Stage 1
Ranks post buffers and perform barrier

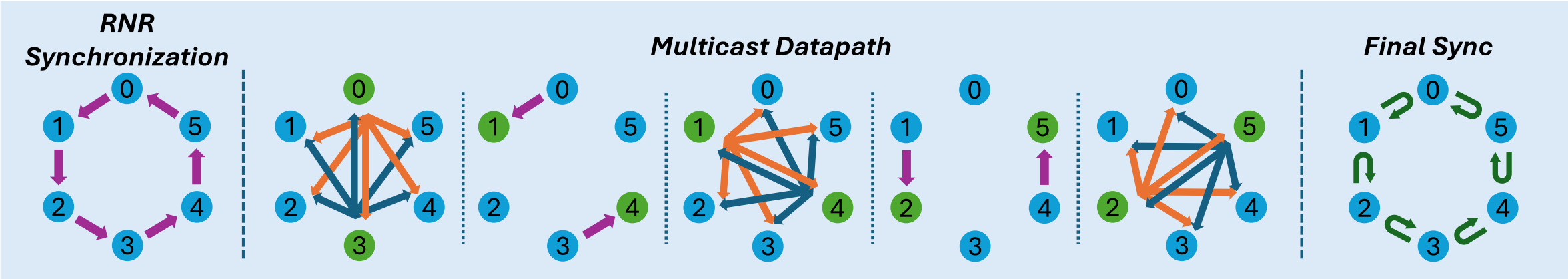


Stage 2.5
The last group multicasts chunks

Allgather as a composition of Broadcasts

Stage 1
Ranks post buffers and perform barrier

Stage 3
Missing chunks are fetched with RDMA Reads



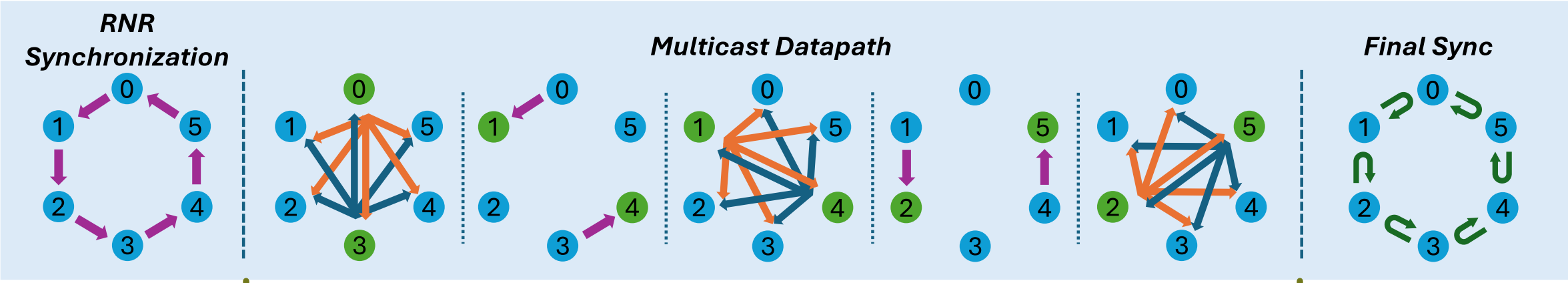
Stage 2.5
The last group multicasts chunks

Each rank maintains a bitmap of successfully received chunks

Allgather as a composition of Broadcasts

Stage 1
Ranks post buffers and perform barrier

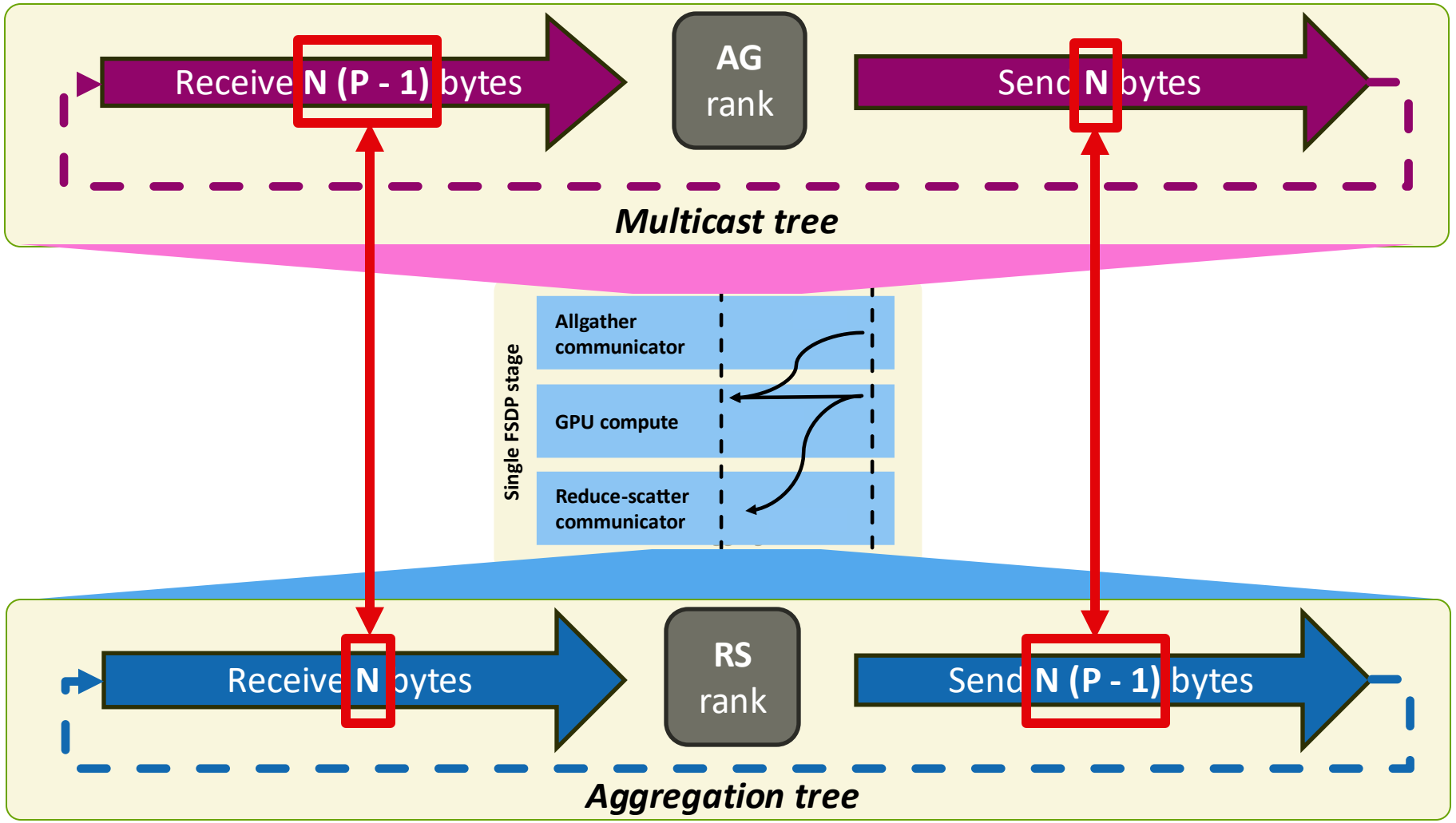
Stage 3
Missing chunks are fetched with RDMA Reads



$$T = N(P - 1)/BW$$

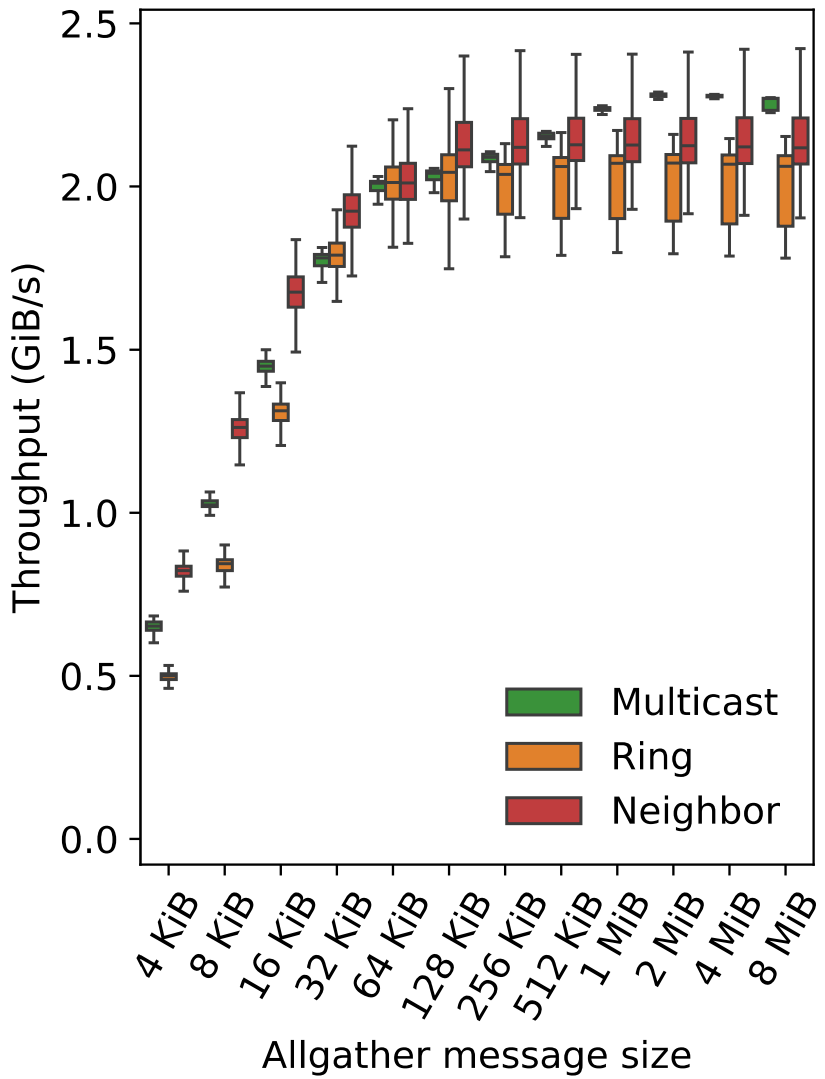
As good as ring but brings traffic reductions on the send NIC path!

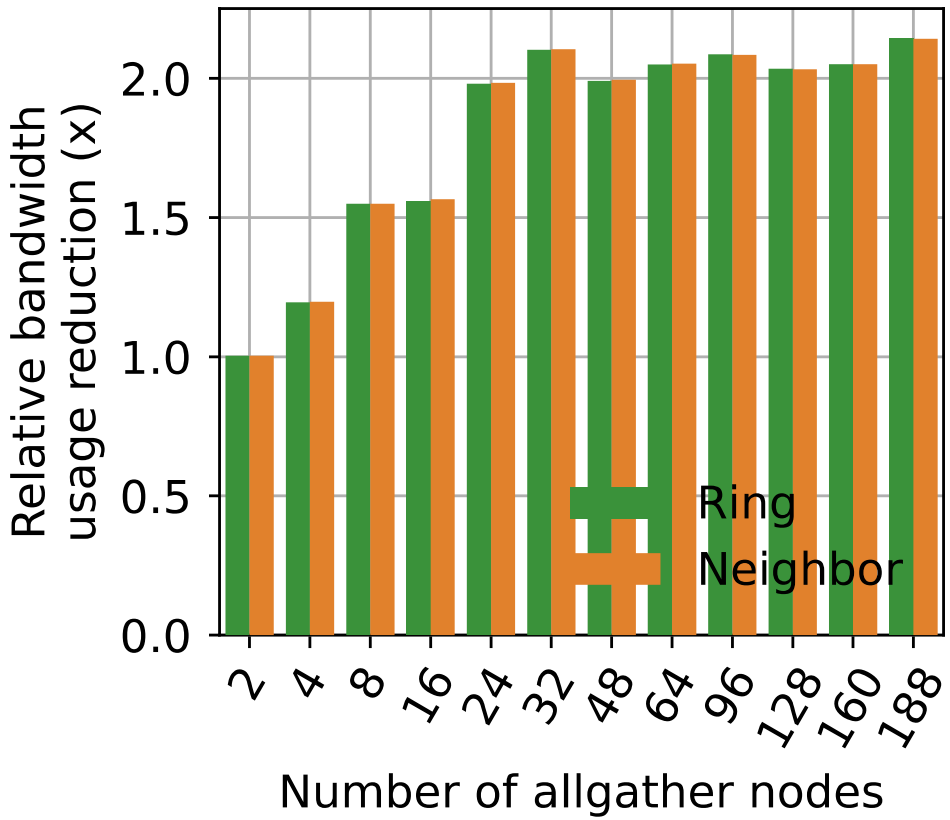
Further improvements for the FSDP pipeline



Up to 2X theoretical AG+RS speedup when compared to rings!

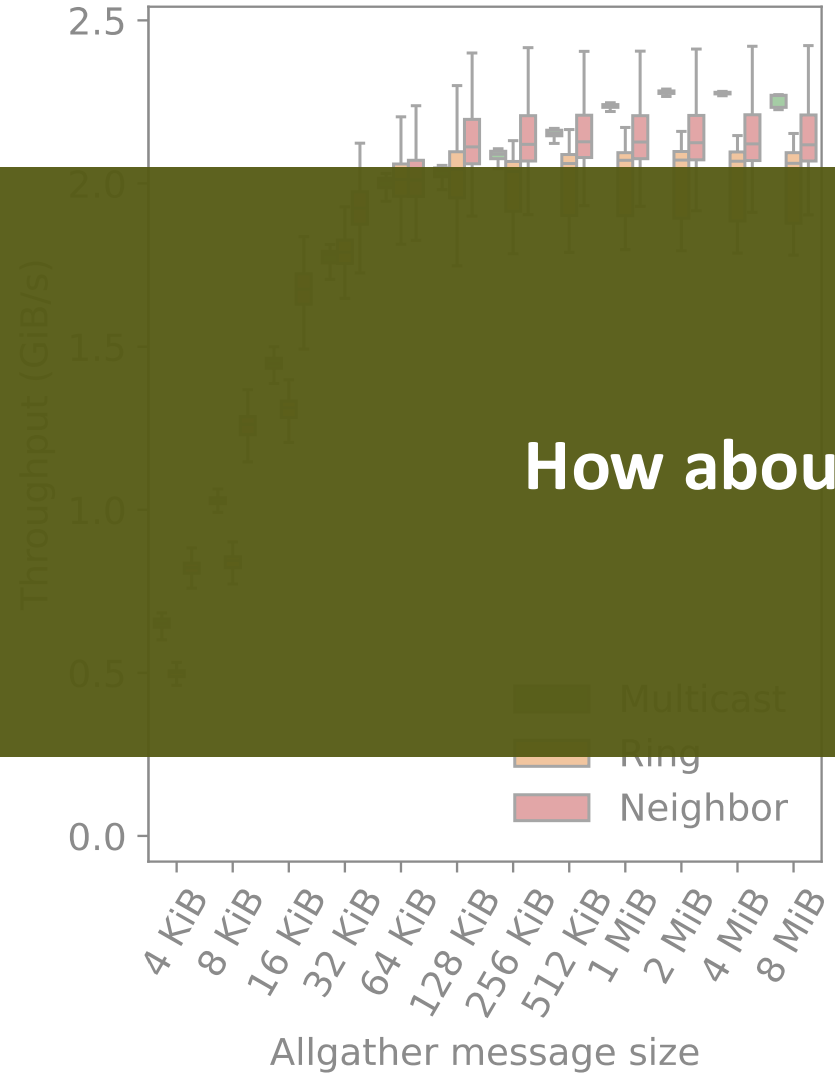
Allgather at 188 nodes with ConnectX-3 and 18 switches

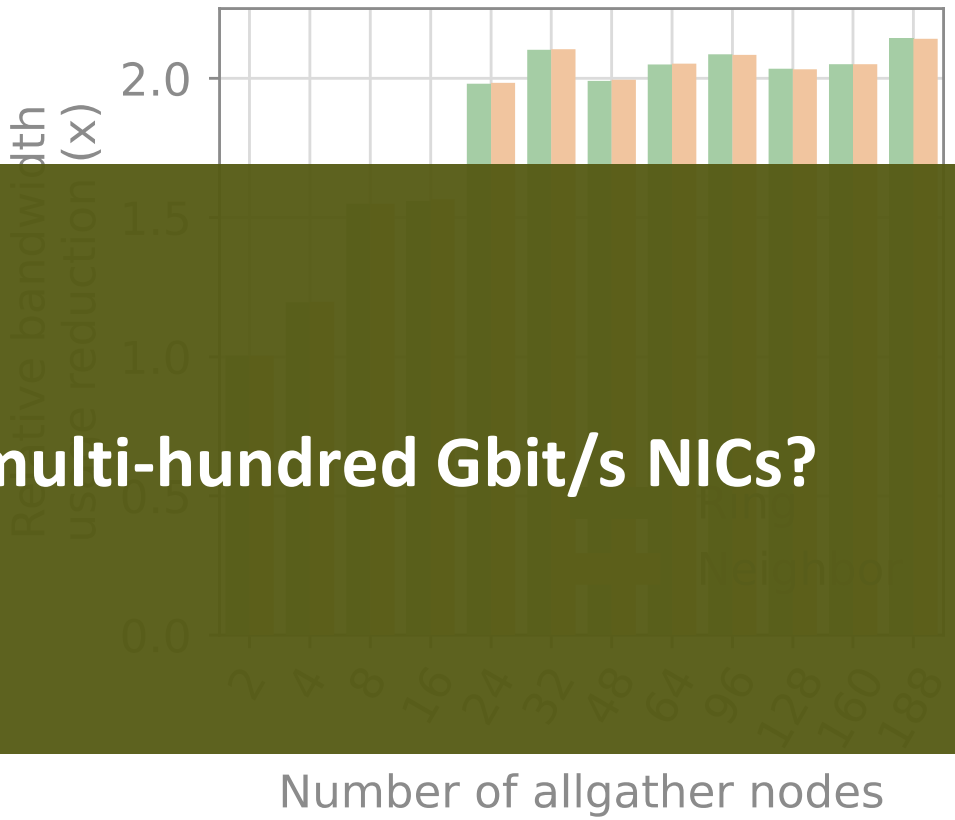




Same performance with up to 2x traffic reduction with multicast-based Allgather

Allgather at 188 nodes with ConnectX-3 and 18 switches

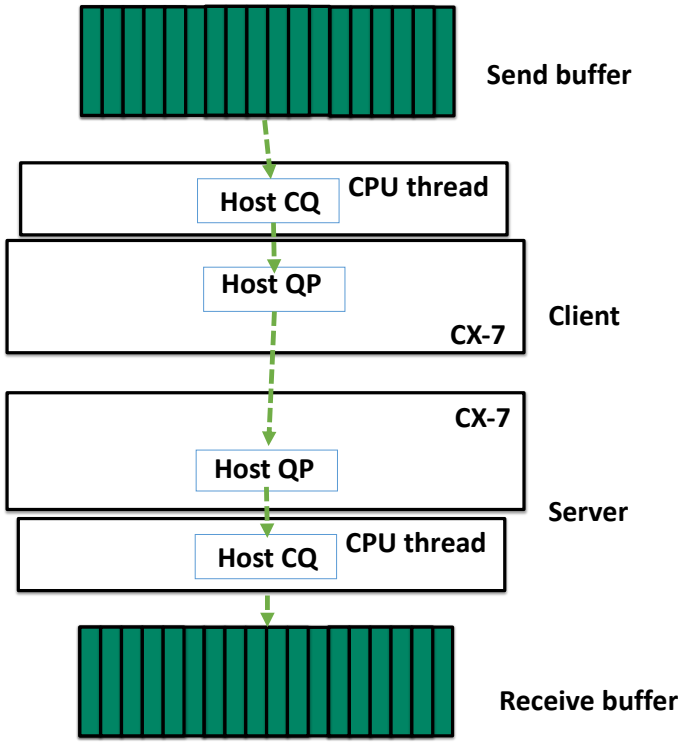


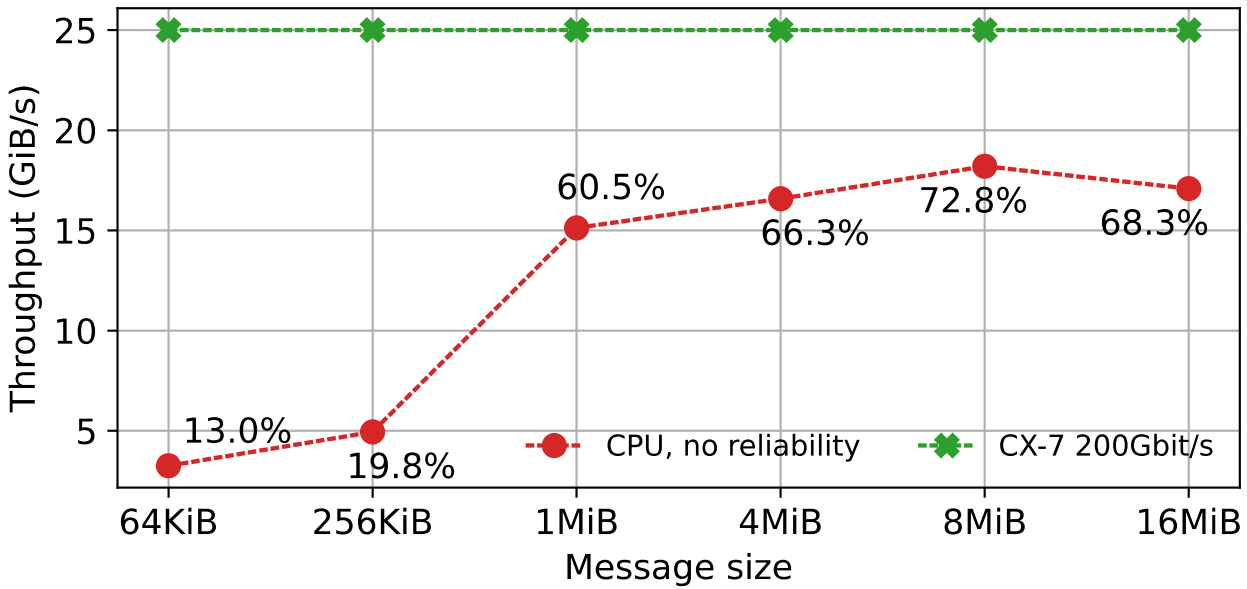


How about multi-hundred Gbit/s NICs?

Same performance with up to 2x traffic reduction with multicast-based Allgather

Why offloading?

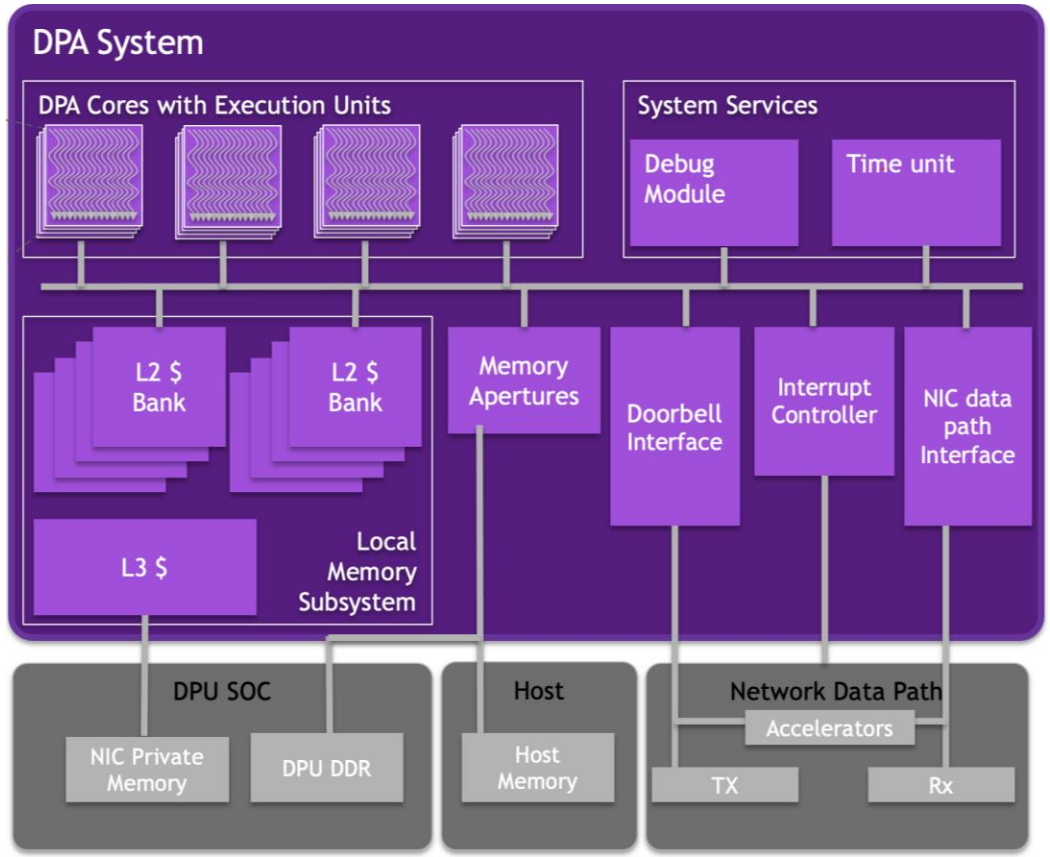




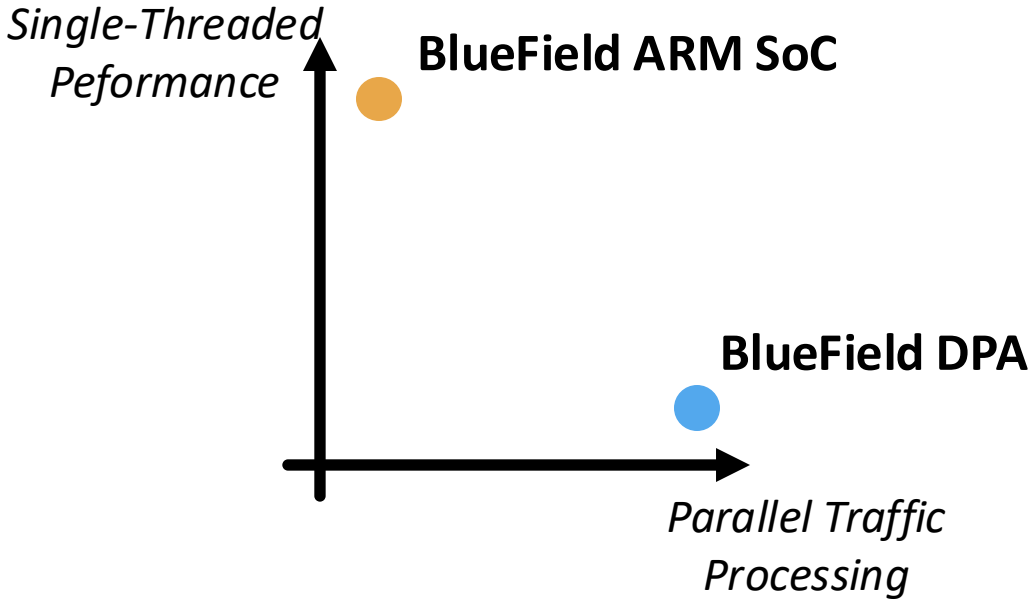
Single-threaded CPU-based collective progress engine is **infeasible**

How about using offloading?

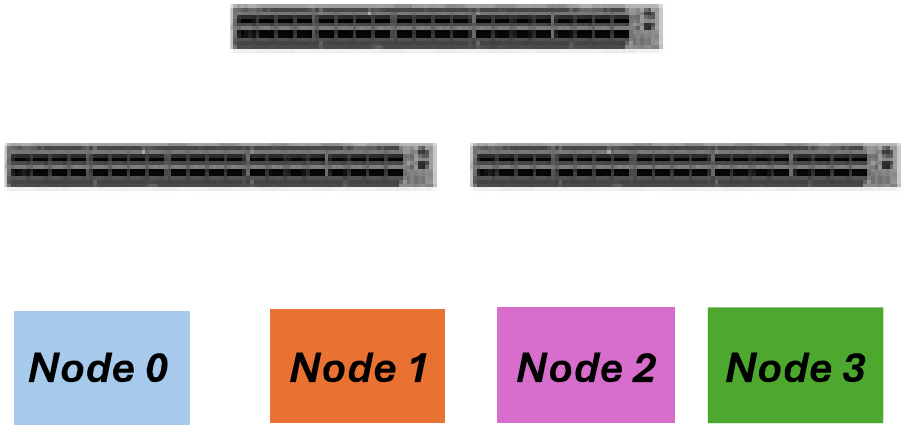
NVIDIA Datapath Accelerator (DPA)



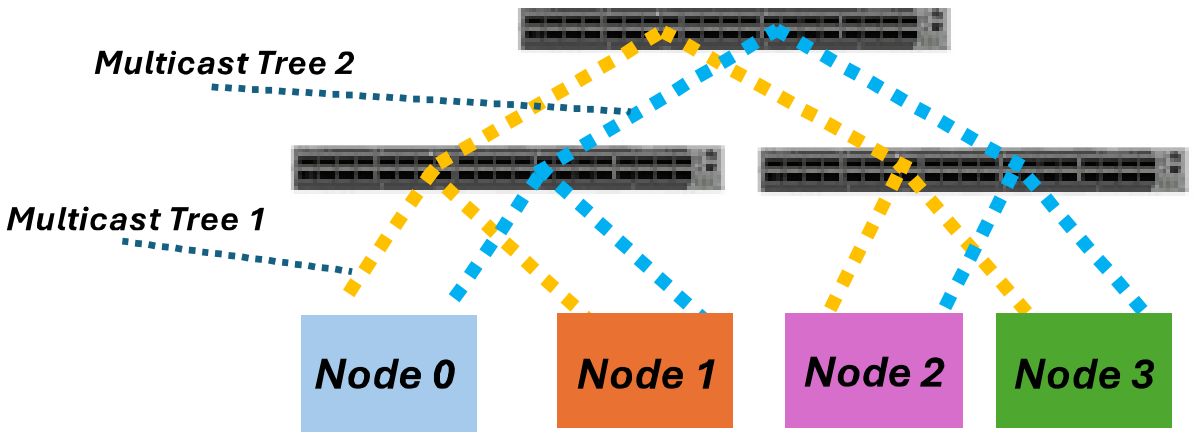
- Designed for data movement offloading
- **256** hardware threads
- Programmed with user-space C API



SmartNIC-offloaded system design

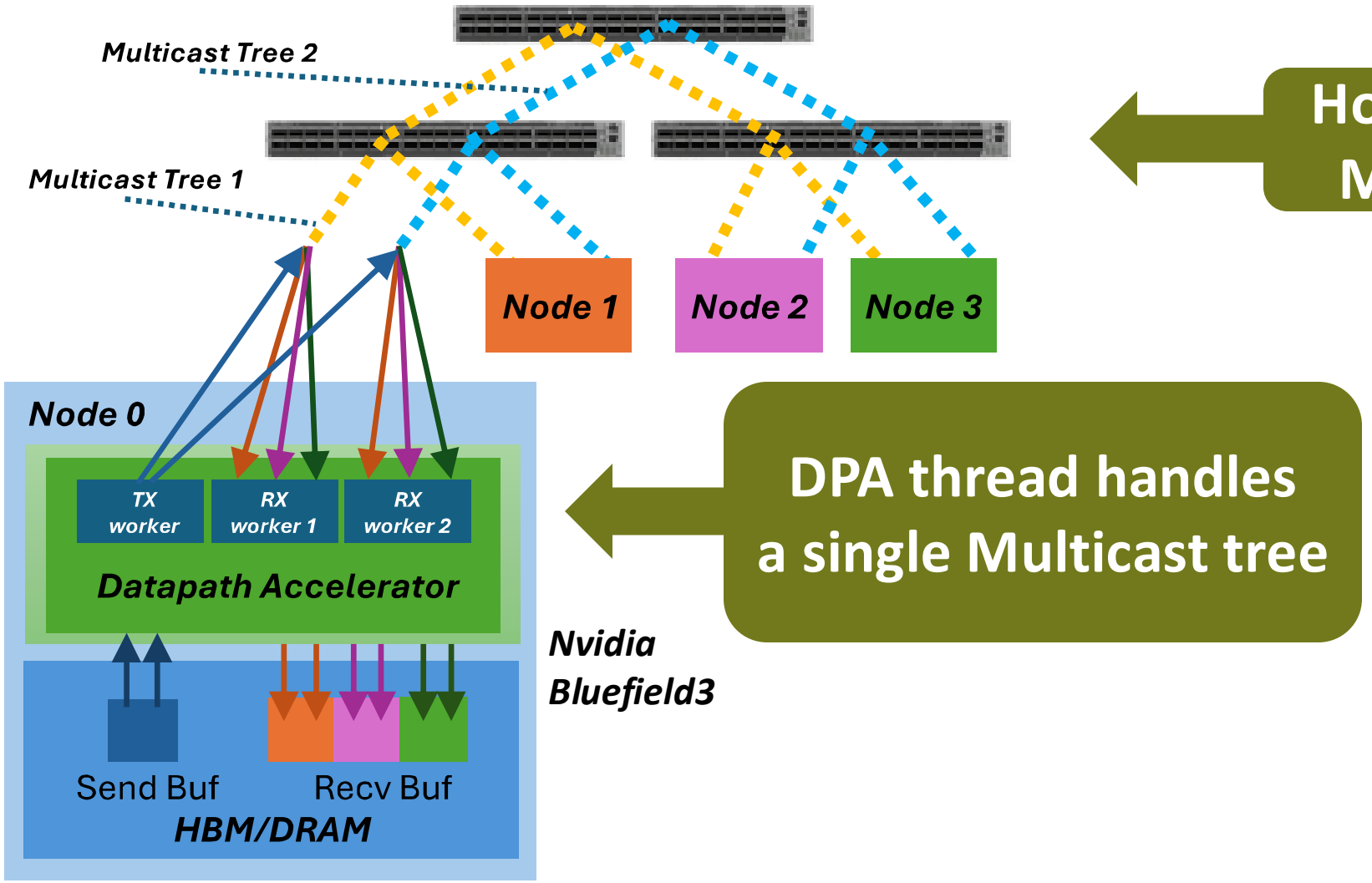


SmartNIC-offloaded system design



Host-side creates Multicast trees

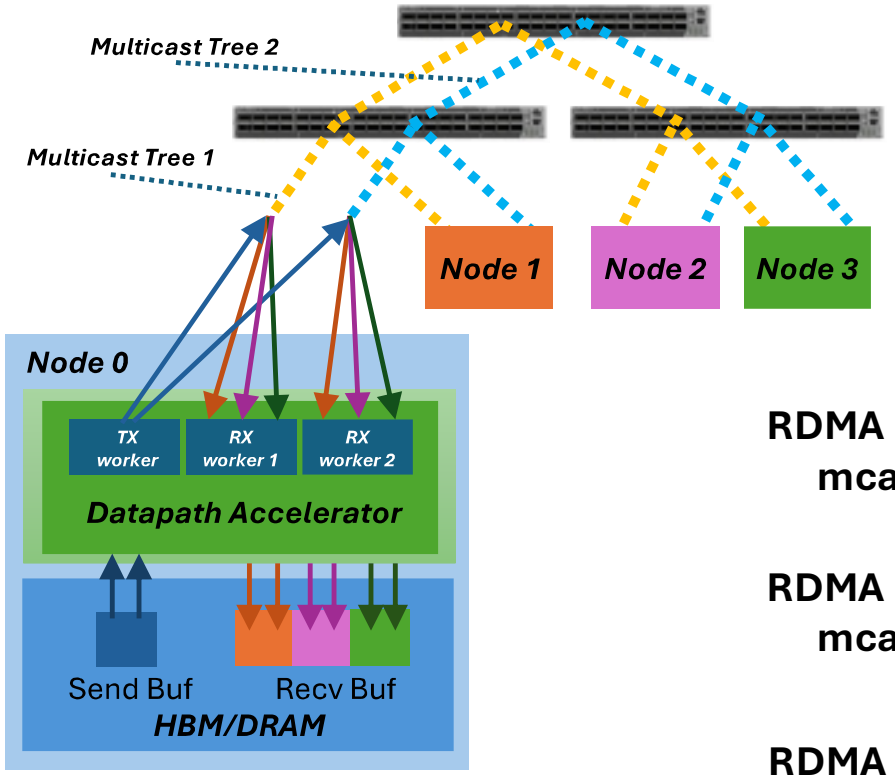
SmartNIC-offloaded system design

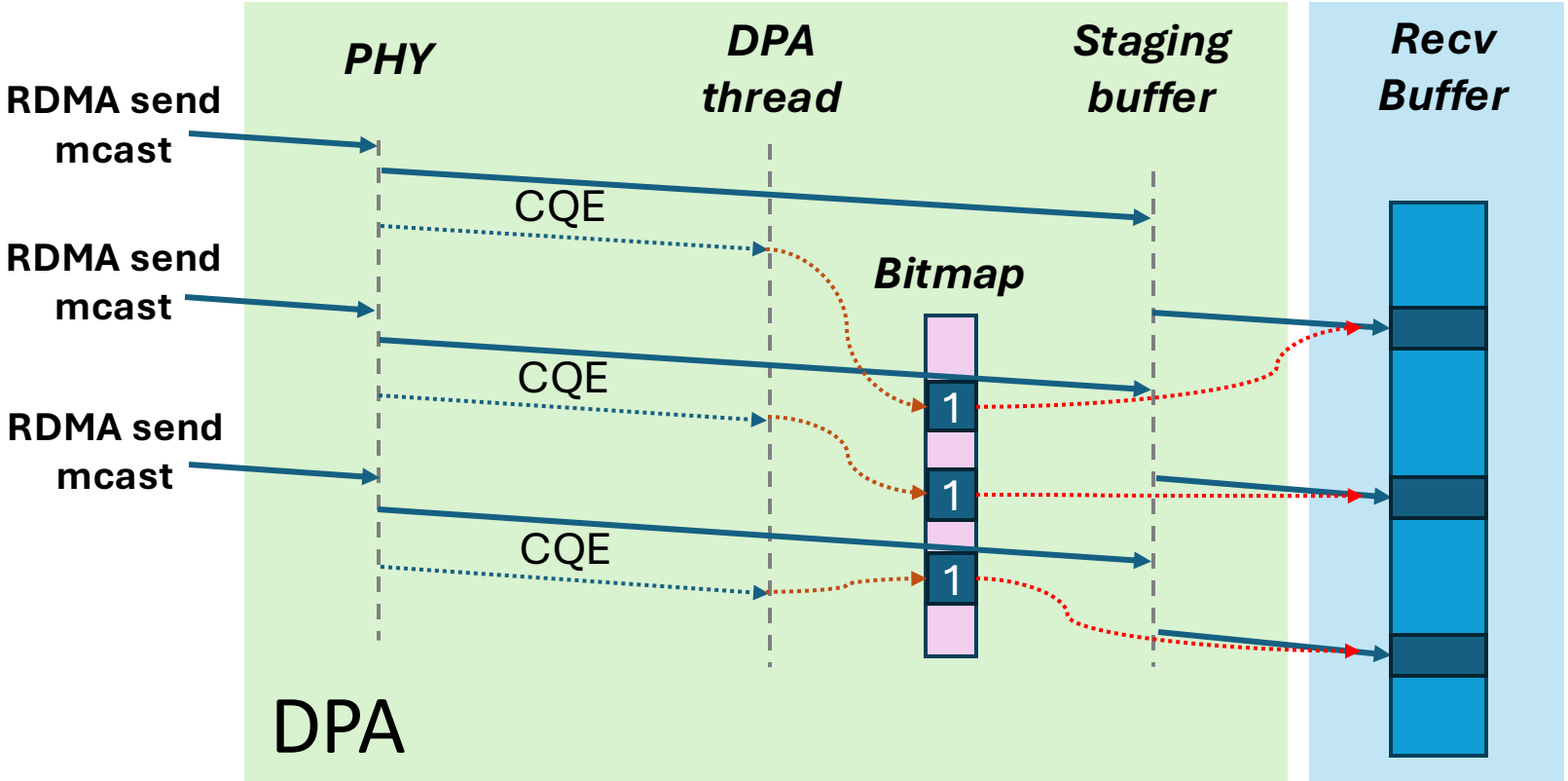


Host-side creates Multicast trees

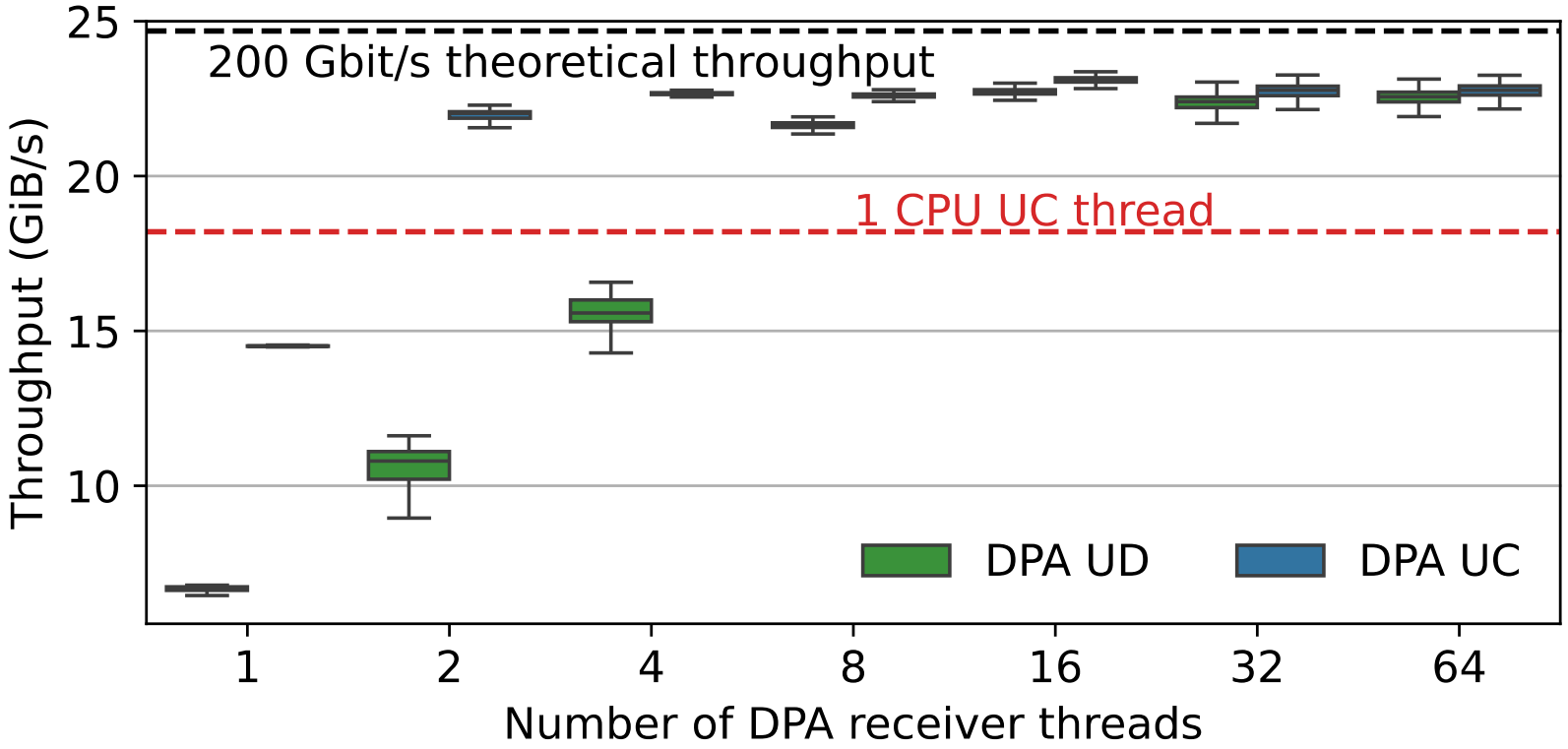
DPA thread handles a single Multicast tree

DPA-based receive progress engine





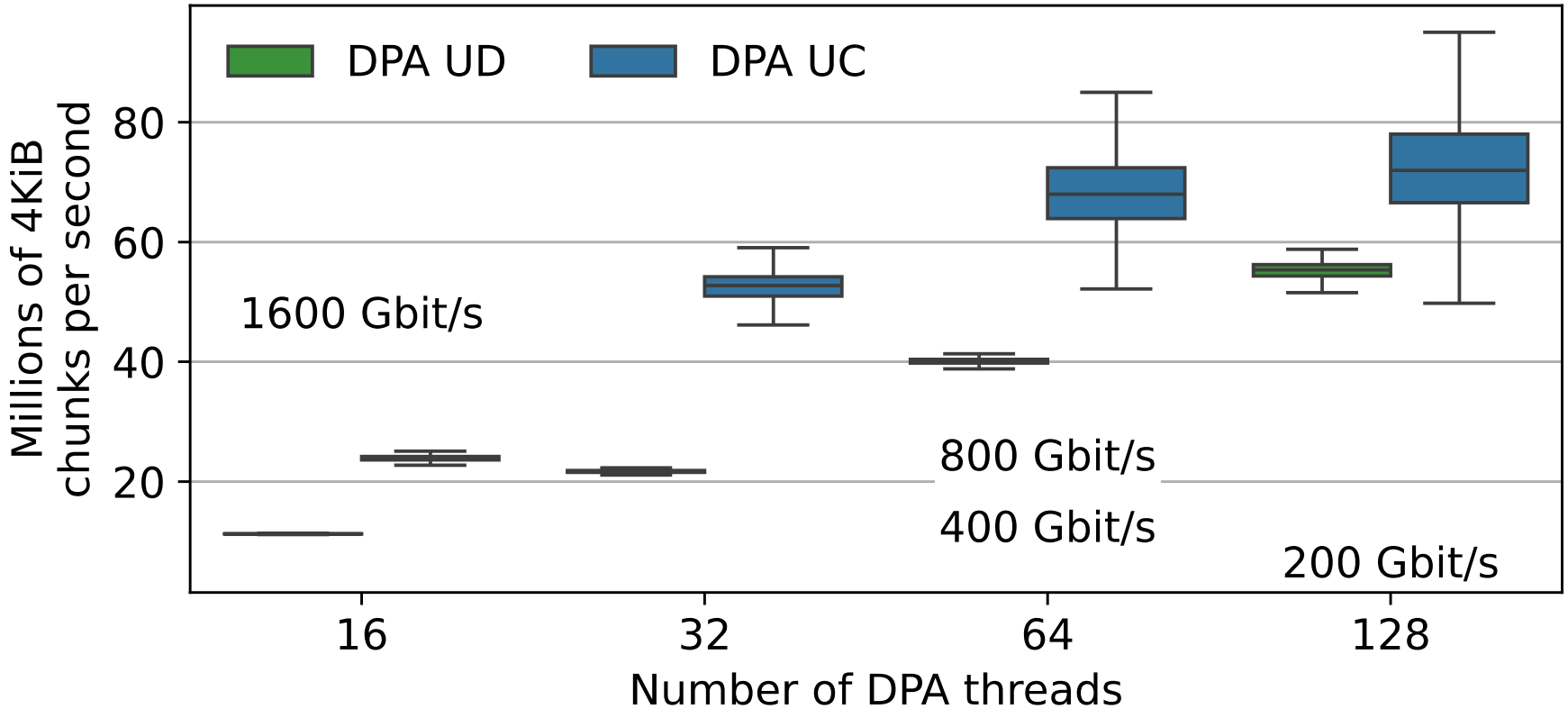
DPA performance



1/16'th of DPA capacity is enough to sustain line rate

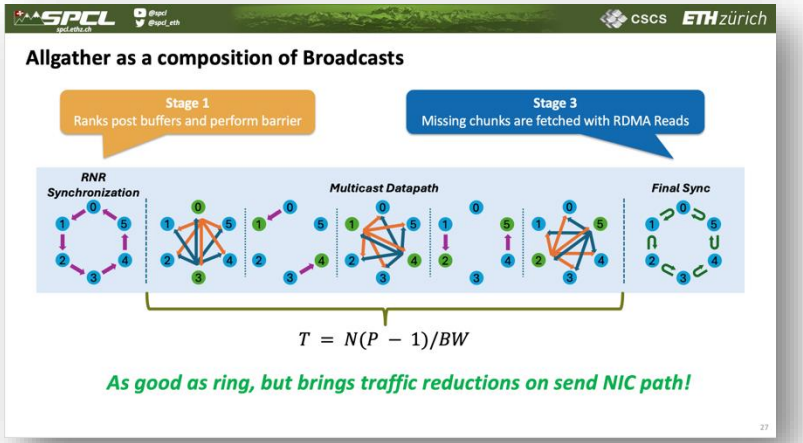
Write-based zero-copy solution further reduces DPA footprint by 4x

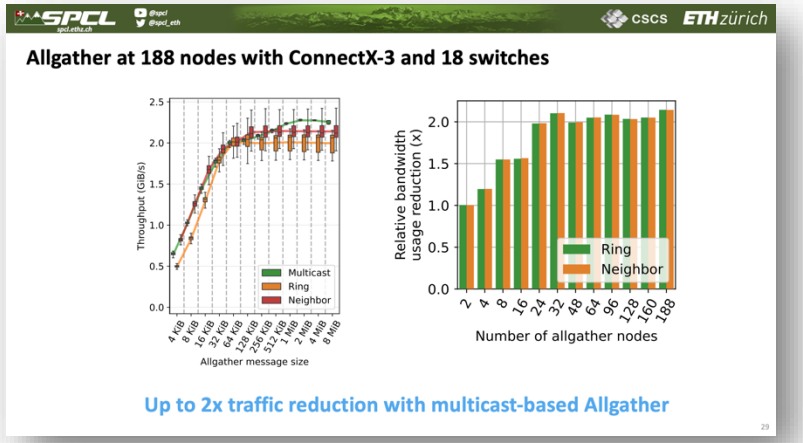
How about Tbit/s links?

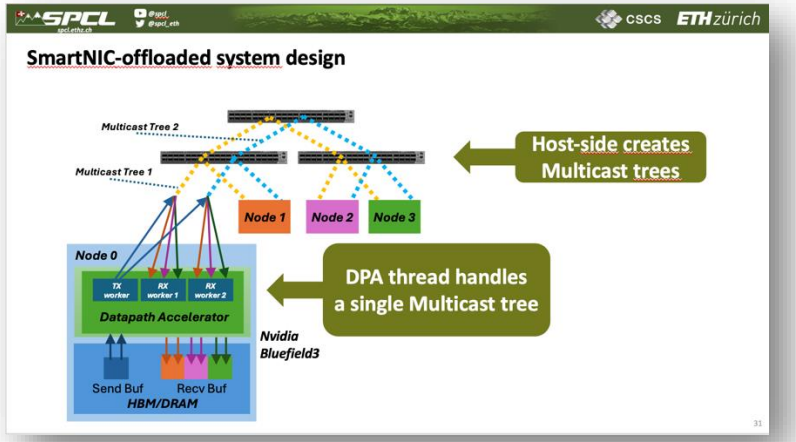


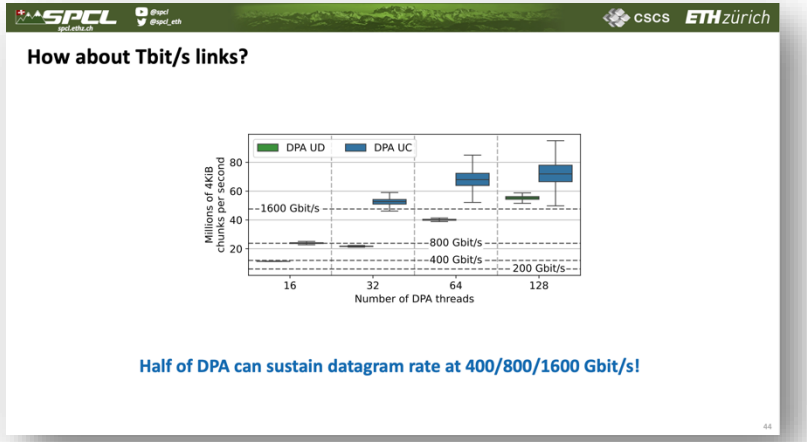
Half of DPA can sustain datagram rate at 400/800/1600 Gbit/s!

Conclusions









More of SPCL's research:

youtube.com/@spcl — 180+ Talks

twitter.com/spcl_eth — 1.4K+ Followers

github.com/spcl — 3.8K+ Stars

... or spcl.ethz.ch

arXiv paper
https://arxiv.org/abs/2408.13356

Open-source protocol
https://github.com/spcl/muliticast-based-allgather