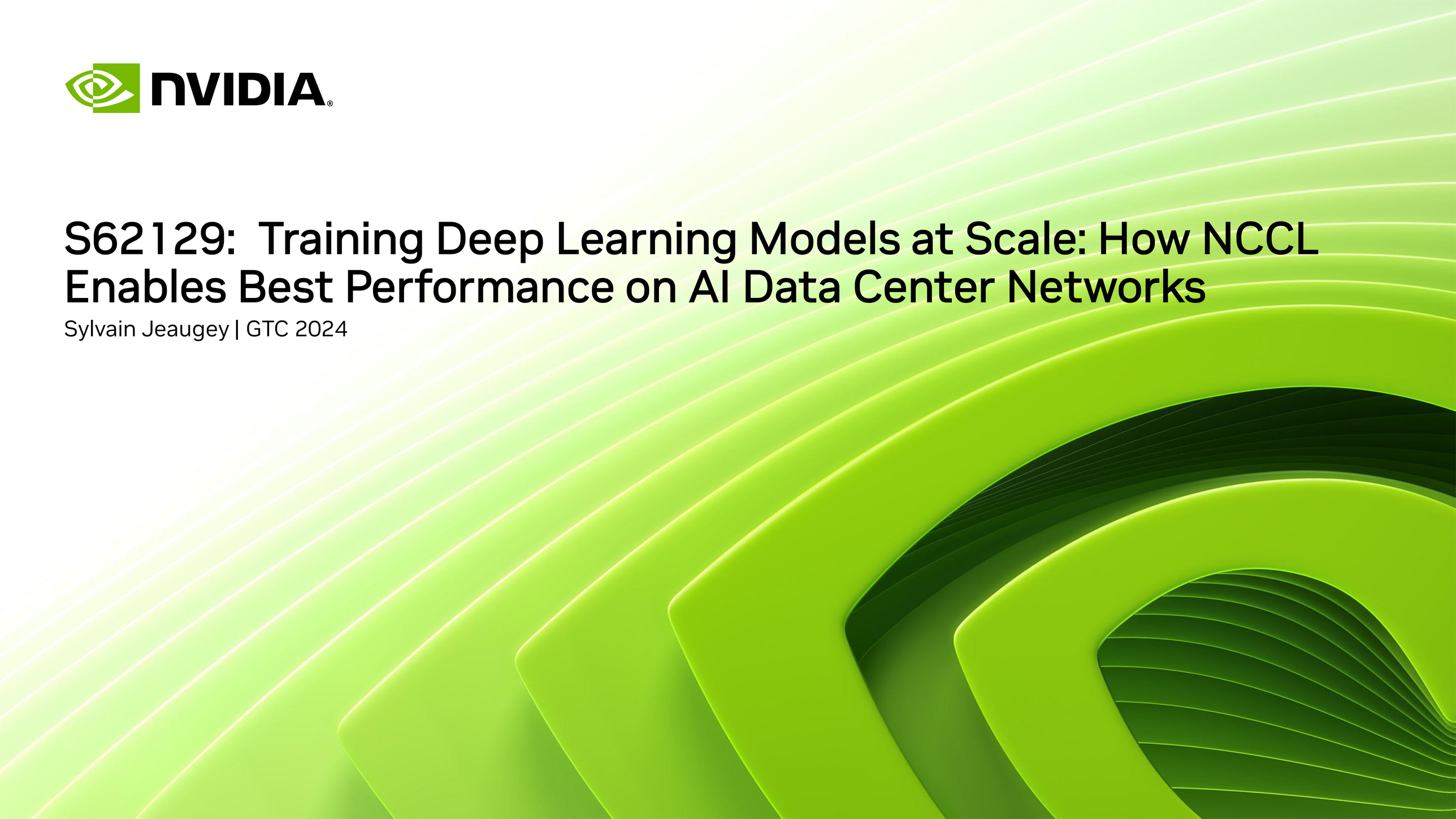


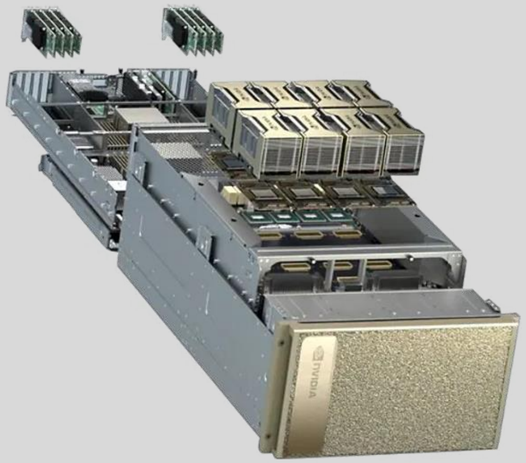
S62129: Training Deep Learning Models at Scale: How NCCL Enables Best Performance on AI Data Center Networks

Sylvain Jeaugey | GTC 2024

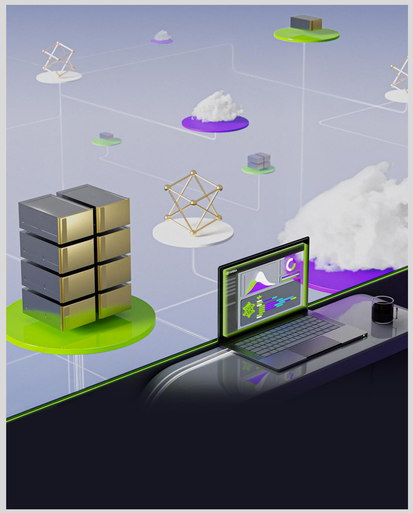
Deep Learning Training On Multiple GPUs

NCCL : Central piece of software for multi-GPU DL training, handles inter-GPU communication on PCI, NVLink, networking.

PCI Server



DGX/HGX



Cloud

Who should know about NCCL?

System designers, to understand how design choices will affect the performance of DL training:
- PCI topology
- NVLink/NVLink switch support (not only for intra-node communication!)
- Network technology and topology

Users, to know what performance and scalability to expect from a given platform.

Developers, if they need inter-GPU communication for their application.

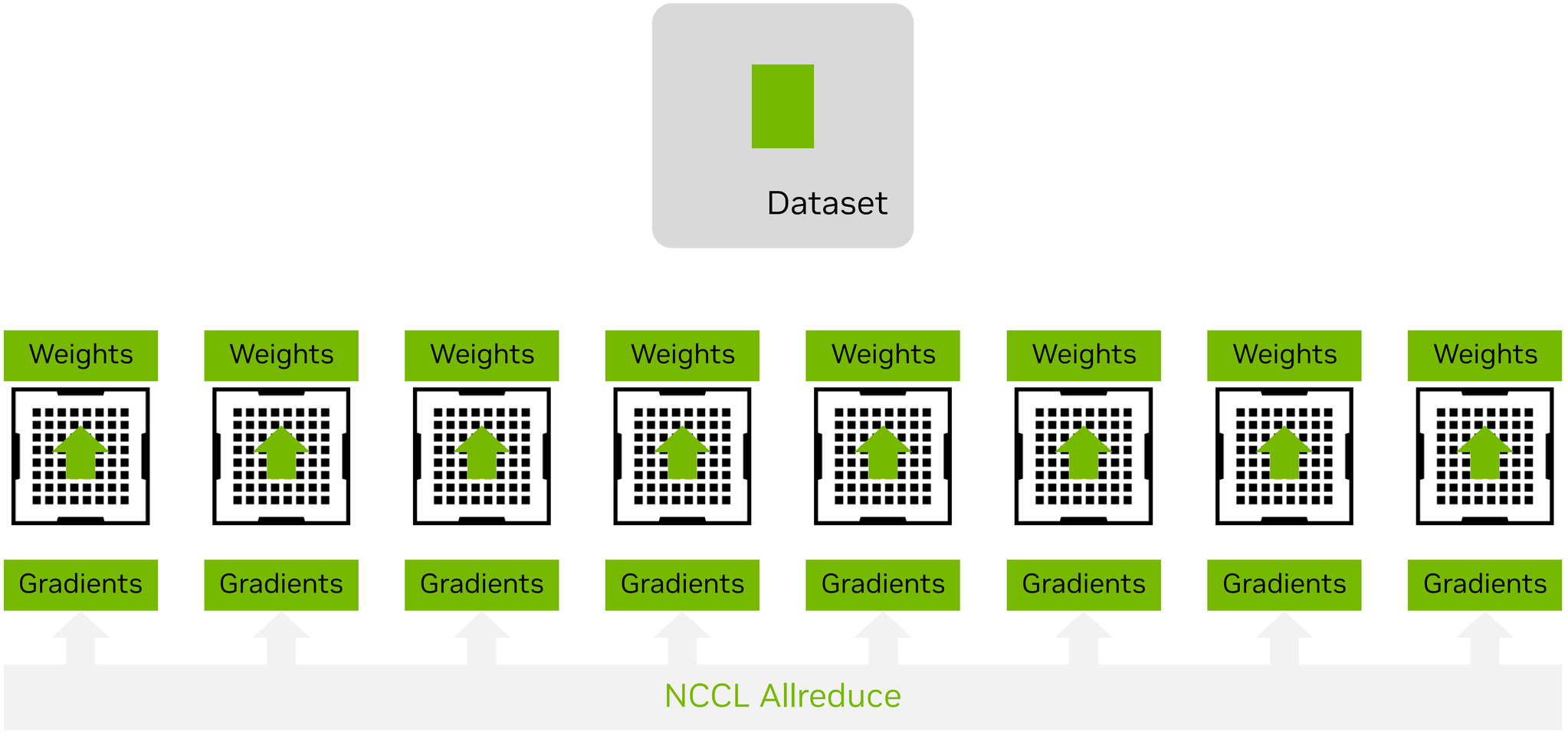
Agenda

- Deep Learning Training
- NCCL Overview
- Protocols
- Algorithms
- New and Future

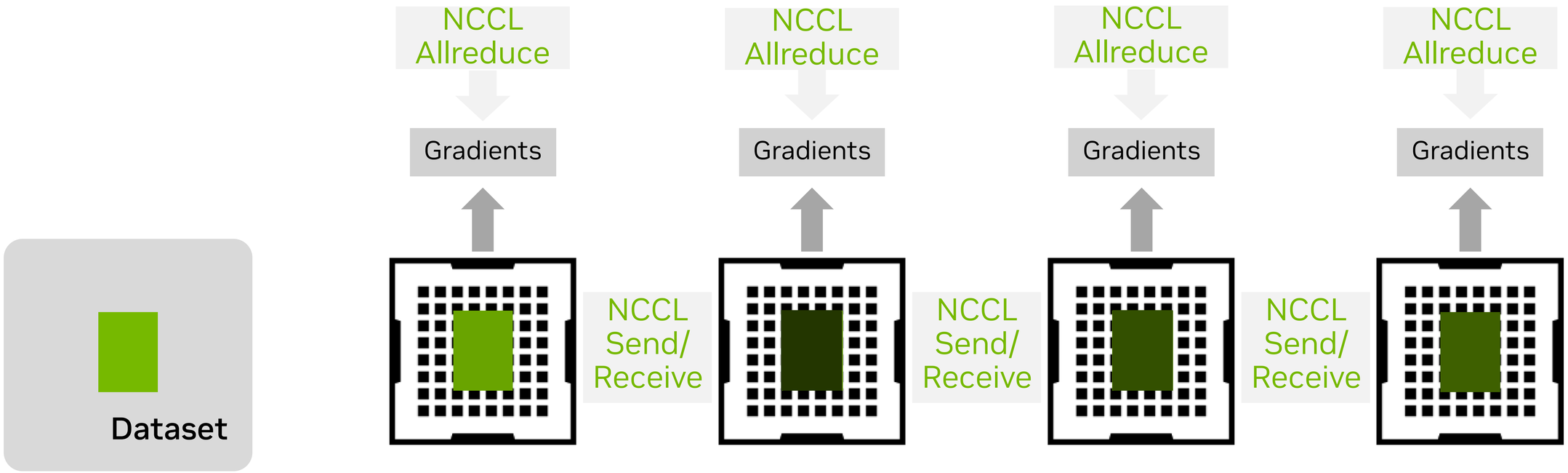
Agenda

- Deep Learning Training
- NCCL Overview
- Protocols
- Algorithms
- New and Future

Data Parallelism

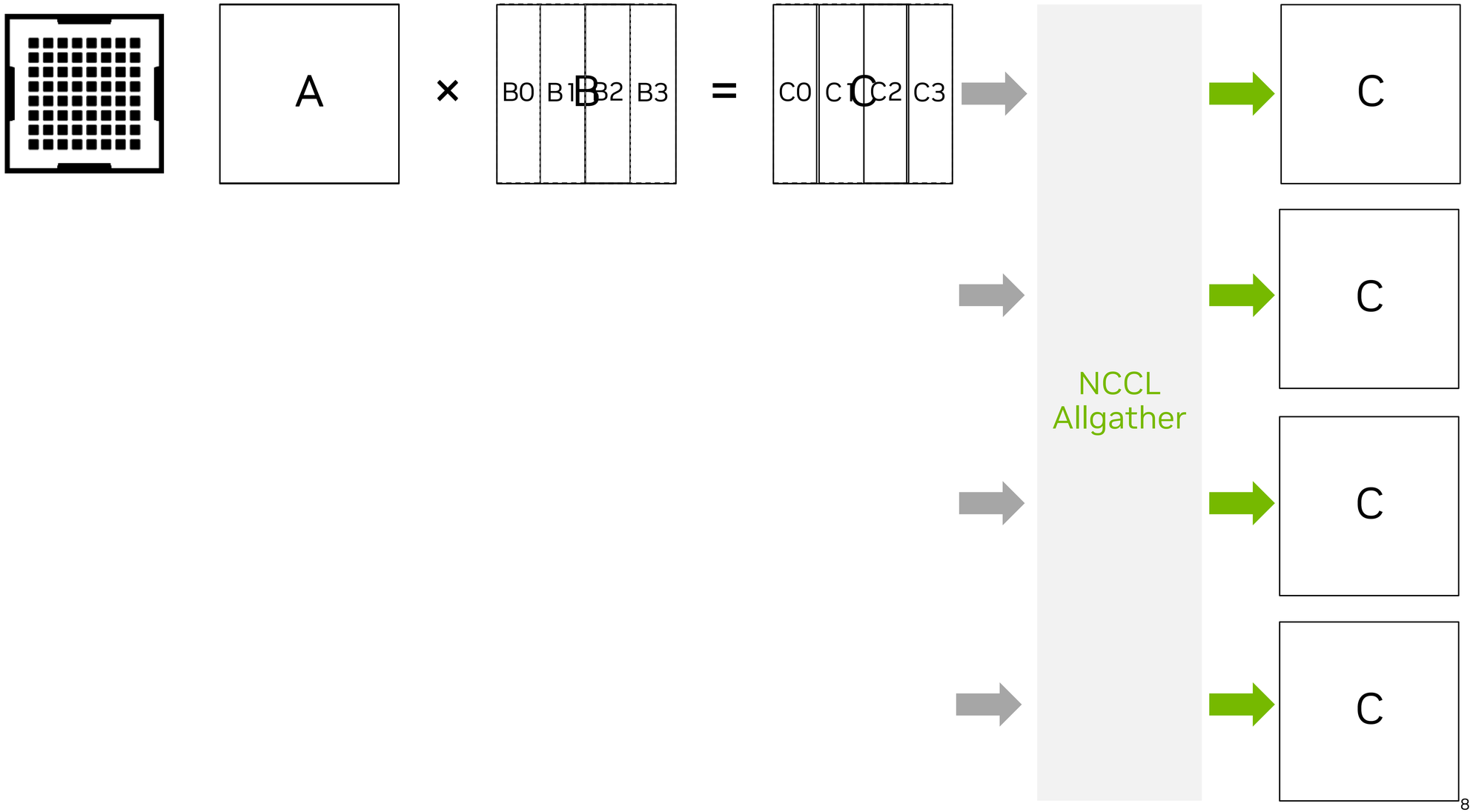


Pipeline Parallelism



Deep Learning Training

Tensor parallelism



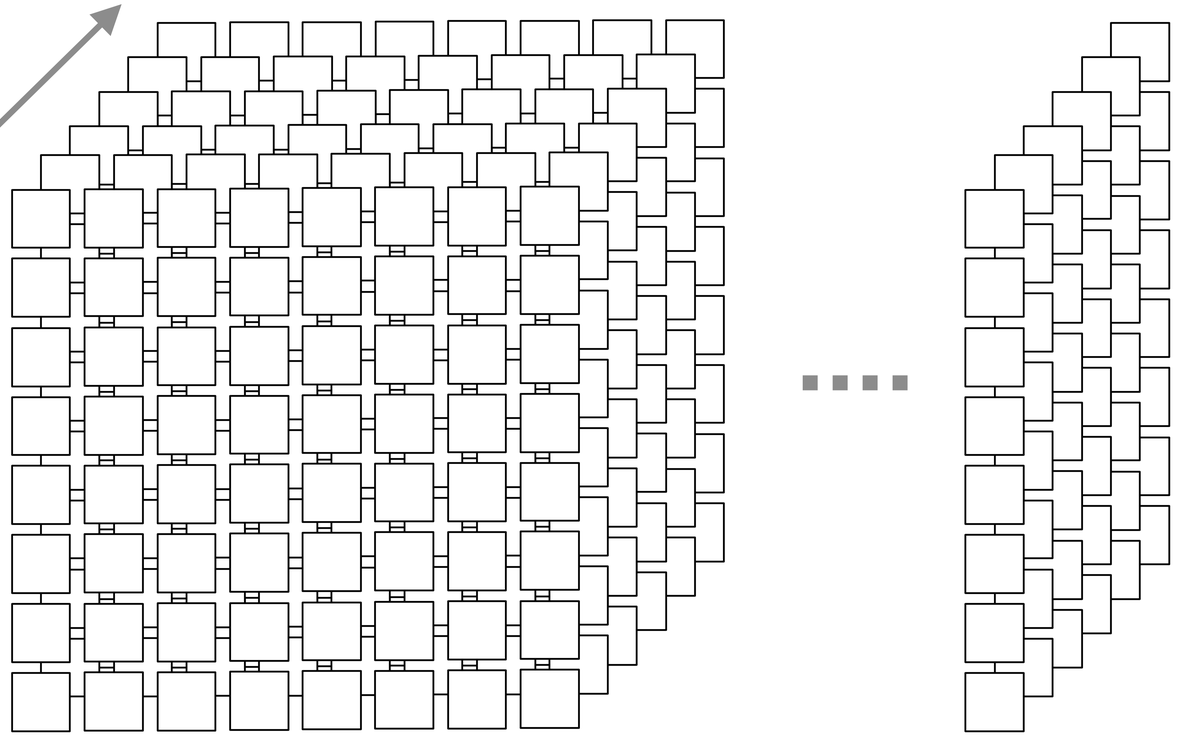
Large scale LLM Training

Pipeline parallelism

ncclSend/
ncclRecv

Tensor parallelism

ncclAllGather



Data parallelism
ncclAllReduce

And also:

MoE (mixture of experts)
ncclSend/ncclRecv (alltoall)

FSDP (fully sharded data parallelism)
ncclAllGather

And other variations …

Agenda

- Deep Learning Training
- NCCL Overview
- Protocols
- Algorithms
- New and Future

NCCL Overview

Where to get NCCL?

NCCL is provided as binary packages like CUDA, and integrated into containers. It is also open-source, and available on Github.

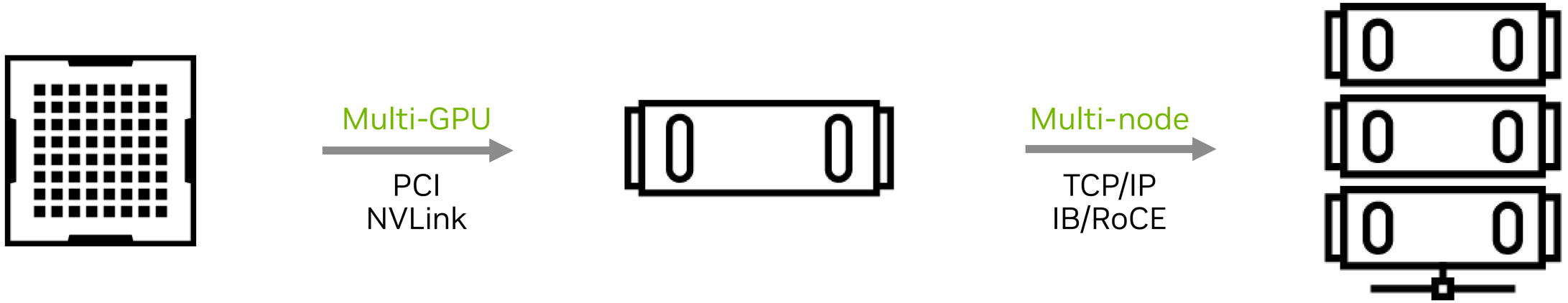
How does NCCL work?

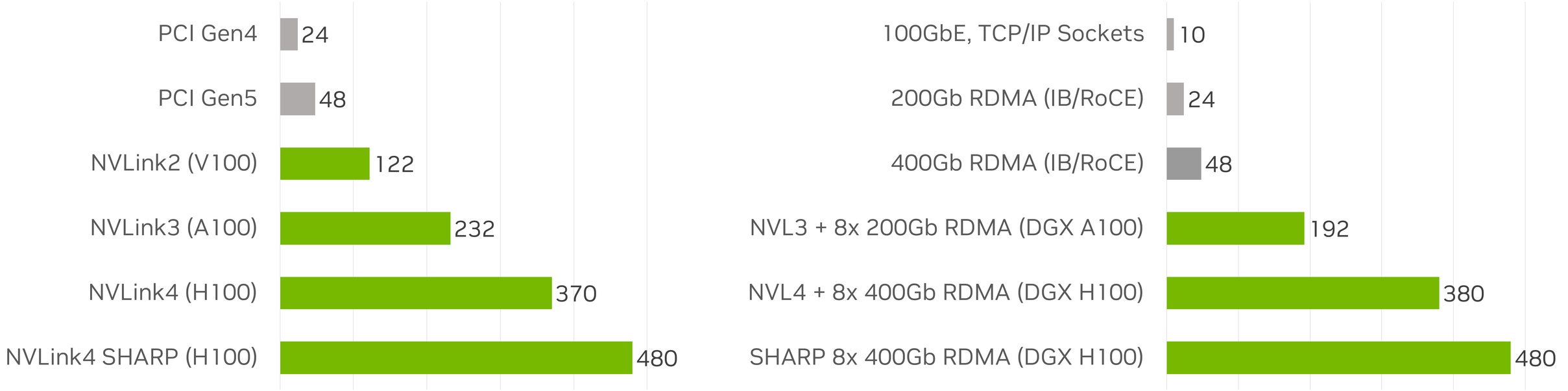
NCCL implements communication primitives as CUDA operations (kernels) which can be inserted into the computing workflow. For network communication, a CPU proxy thread will assist the GPU.

What is the goal of NCCL?

Allow users to not have to worry about inter-GPU communication, and run the same code on a desktop with 2 GPUs, a DGX server, or a GPU instance in the cloud.

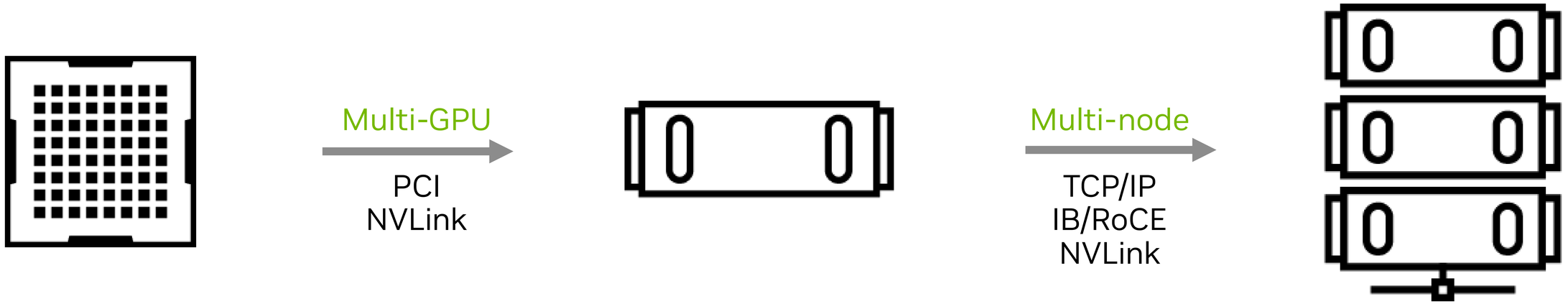
Collective Communication Bandwidth

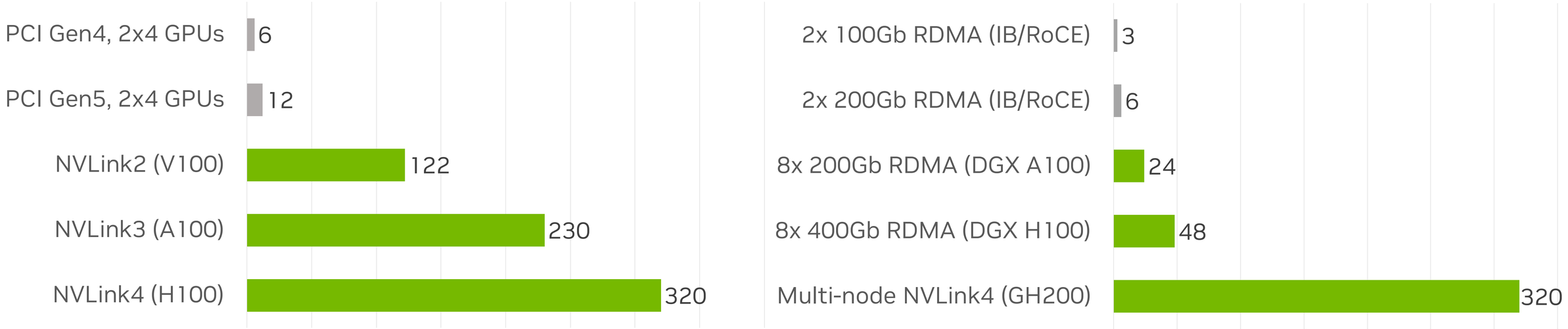




NCCL Tests Allreduce Bus Bandwidth in GB/s

Point-to-point Communication Bandwidth





NCCL Tests Alltoall Bus Bandwidth in GB/s

Agenda

- Deep Learning Training
- NCCL Overview
- **Protocols**
- Algorithms
- New and Future

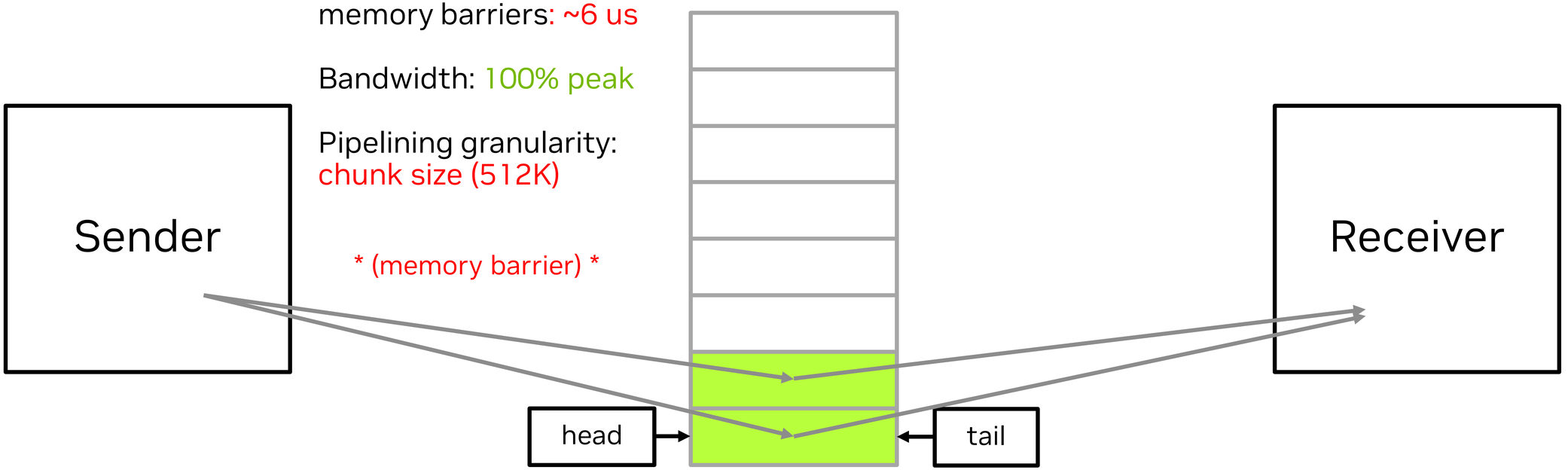
Communication protocols: Simple

High latency due to memory barriers: ~6 us

Bandwidth: 100% peak

Pipelining granularity: chunk size (512K)

* (memory barrier) *



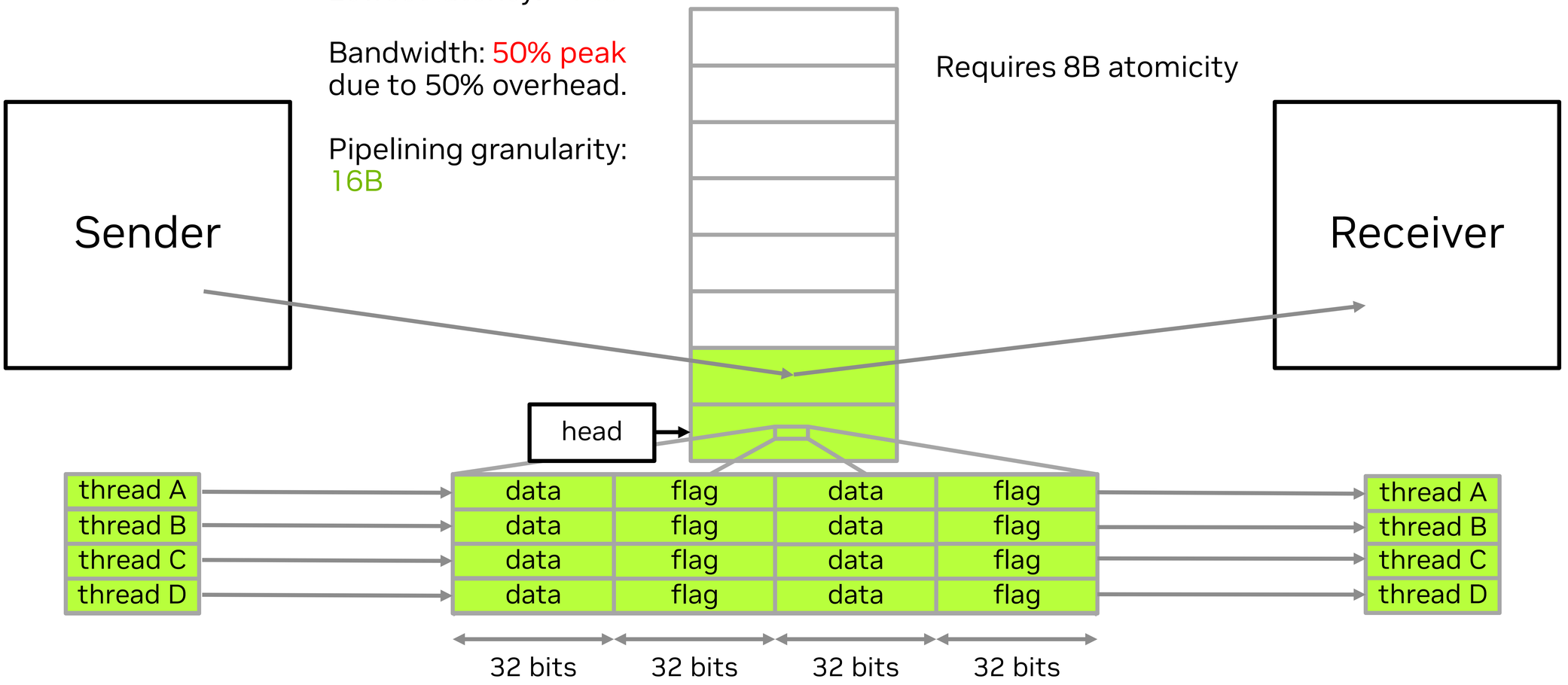
Communication protocols: LL

Lowest latency: ~1us

Bandwidth: 50% peak due to 50% overhead.

Pipelining granularity: 16B

Requires 8B atomicity



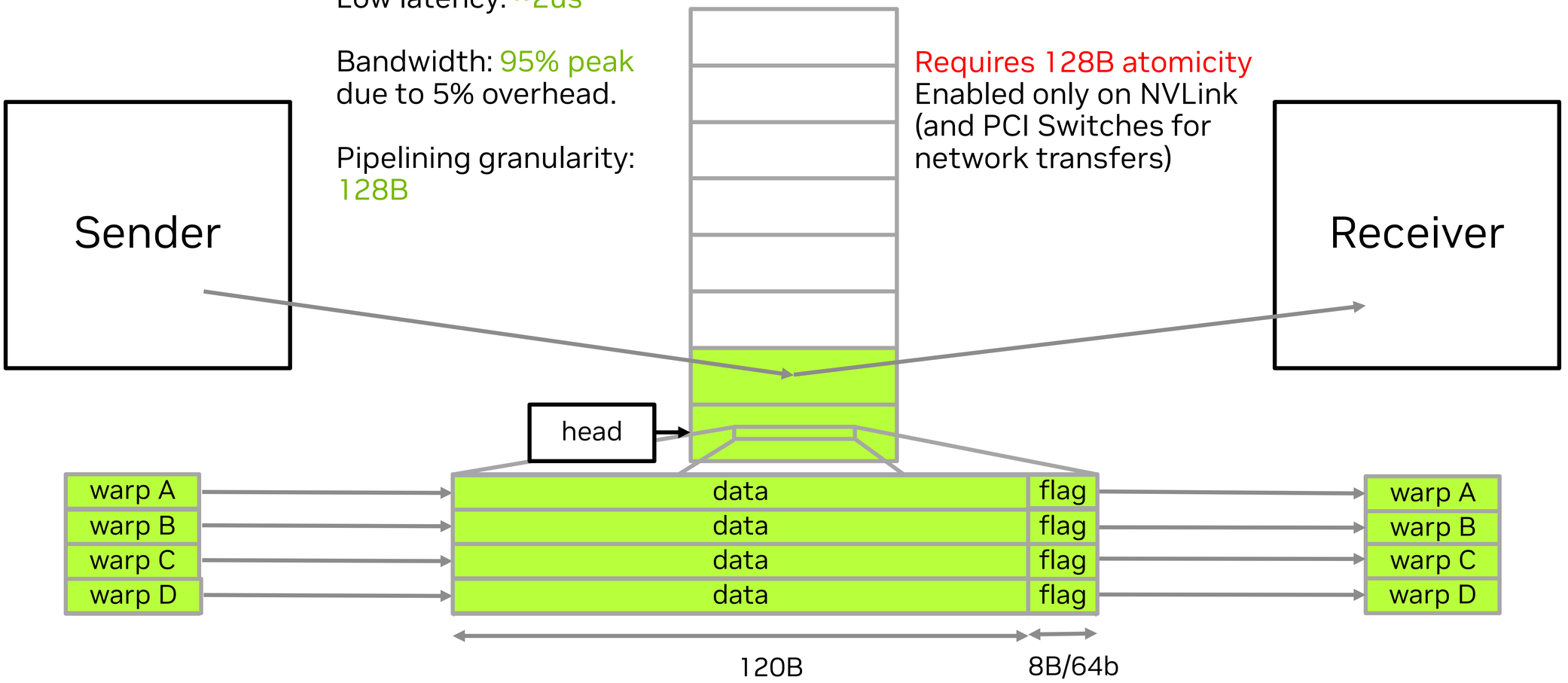
Communication protocols: LL128

Low latency: ~2us

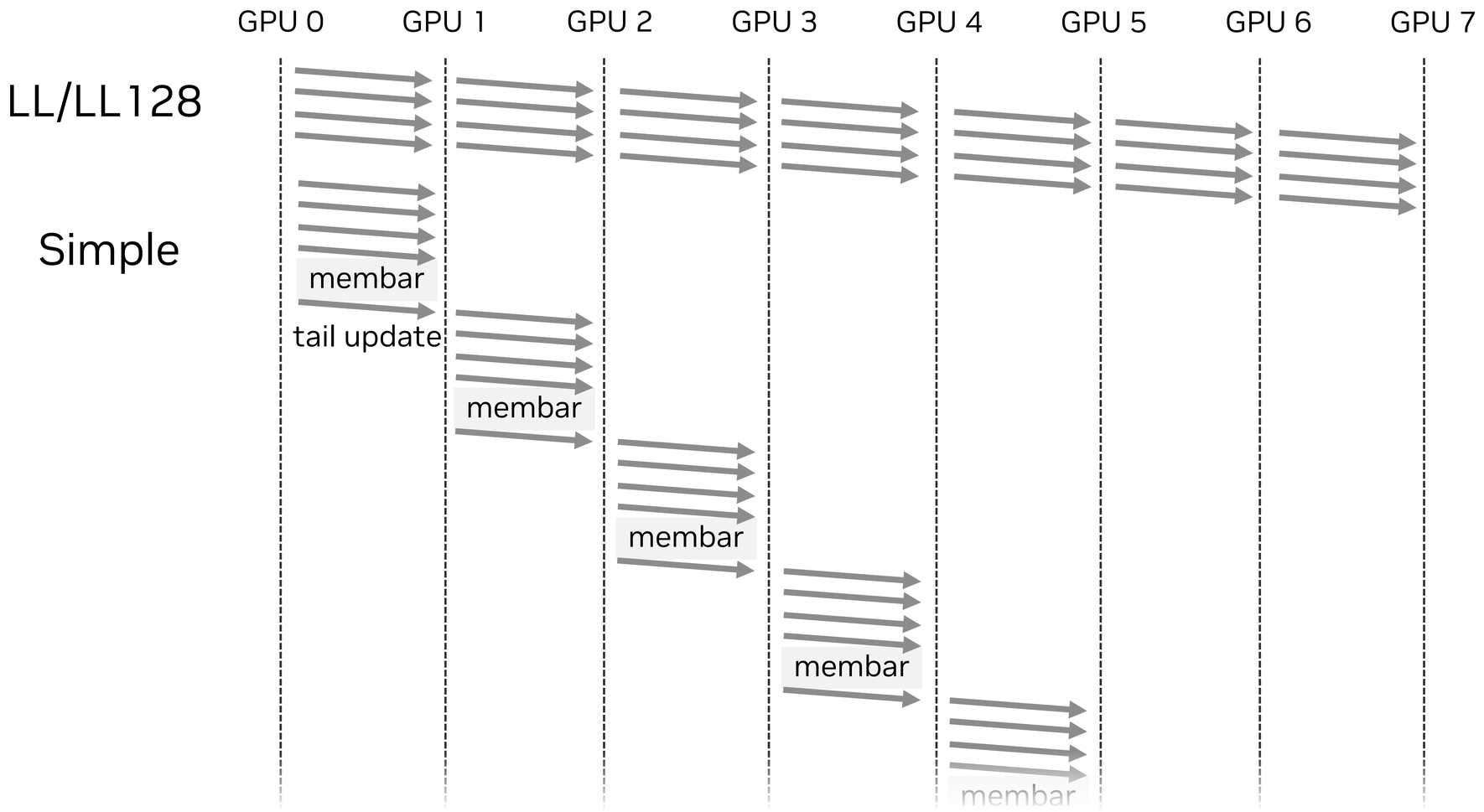
Bandwidth: 95% peak due to 5% overhead.

Pipelining granularity: 128B

Requires 128B atomicity
Enabled only on NVLink (and PCI Switches for network transfers)



Pipelining


GPU 0
GPU 1
GPU 2
GPU 3
GPU 4
GPU 5
GPU 6
GPU 7
LL/LL128
Simple
membar
tail update
membar
membar
membar
membar

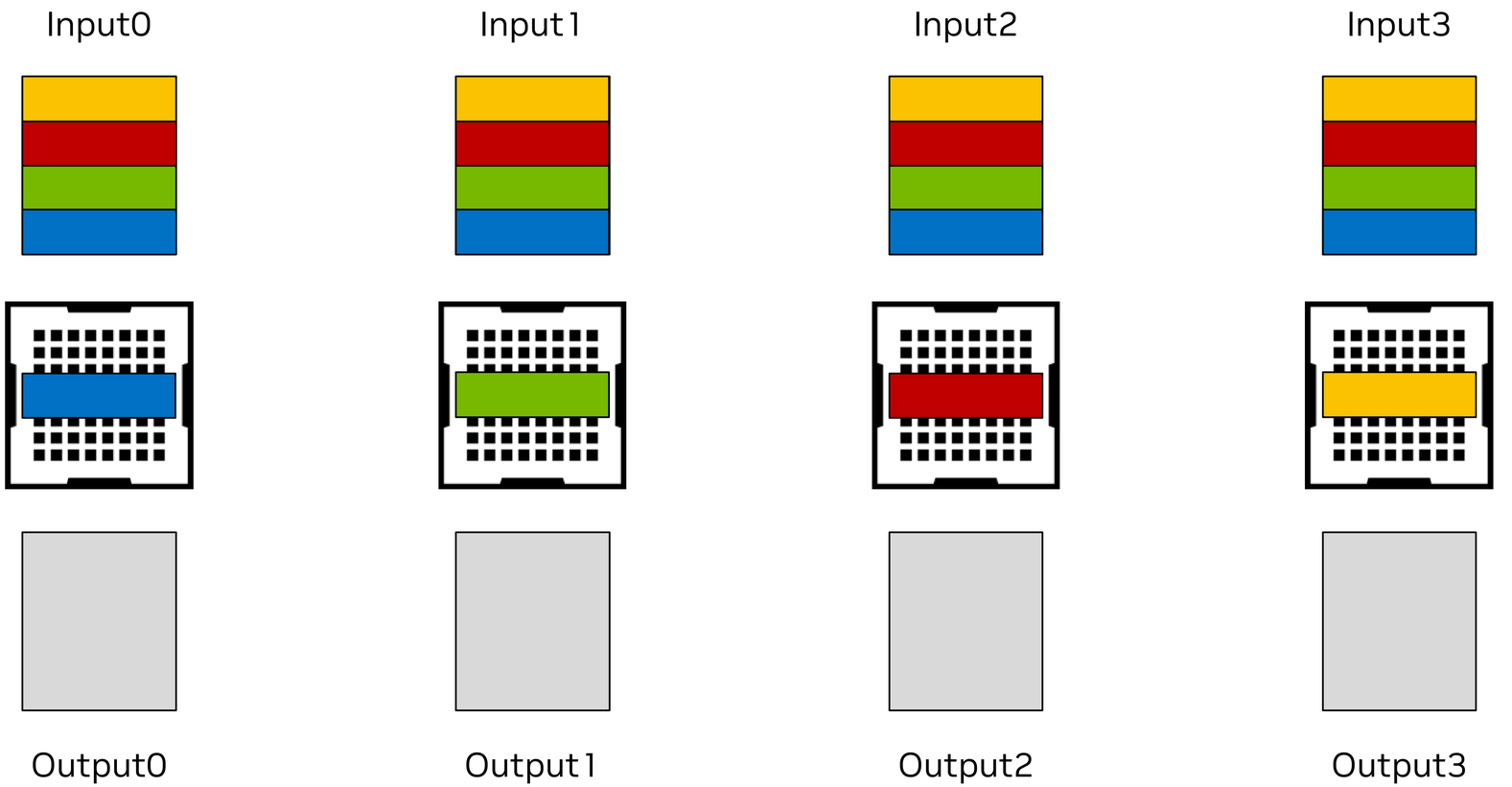
Communication protocols summary

	Simple (NCCL 1.0)	LL (NCCL 2.1)	LL128 (NCCL 2.5)
Latency	6 us	1 us	2 us
Bandwidth	100%	50%	95%
Pipelining	Chunk level (128-512KB)	8B	128B
Requirements	None	8B atomic and unique stores	Intra-node: NVLink Inter-node: PCI switches

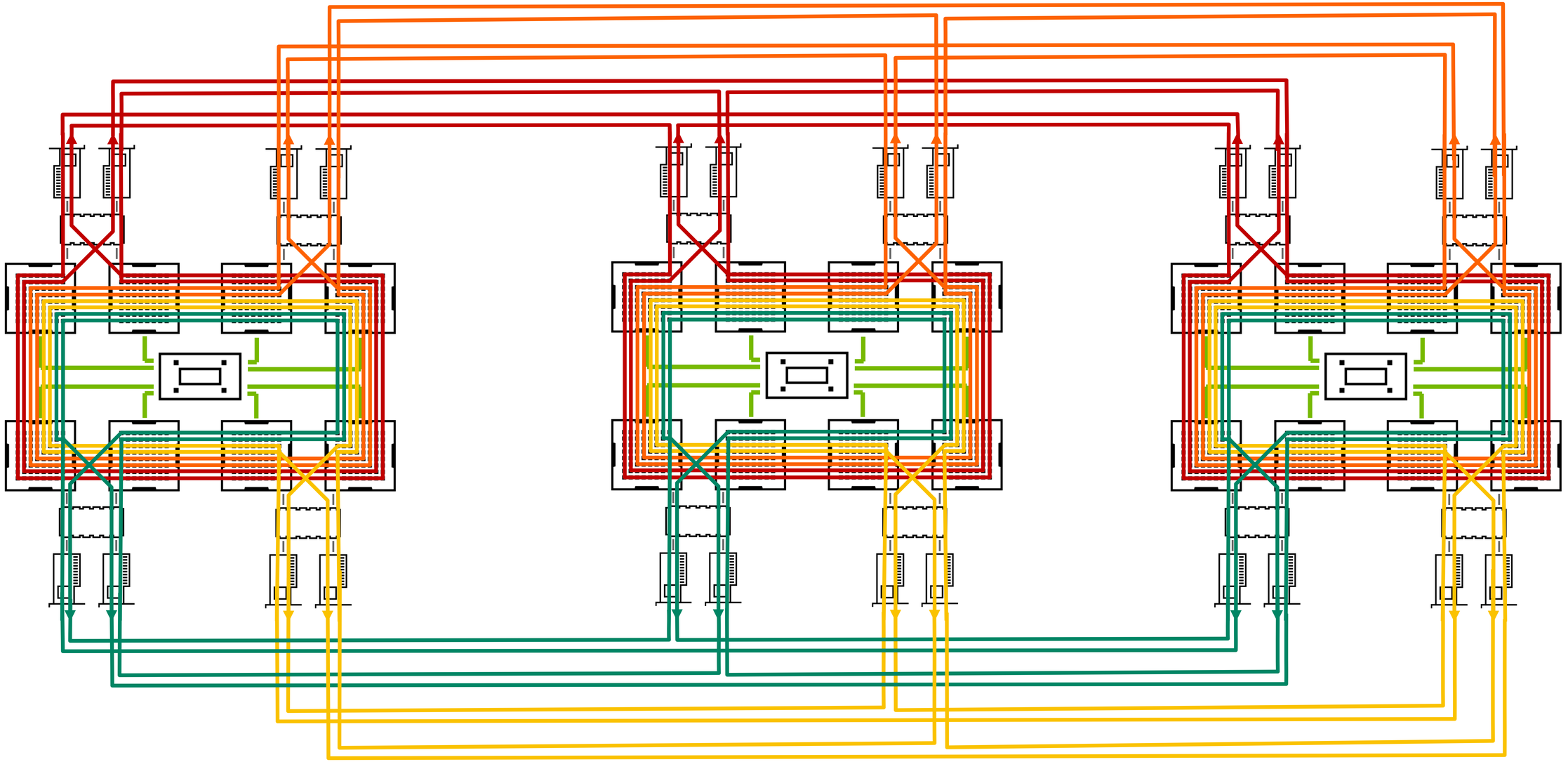
Agenda

- Deep Learning Training
- NCCL Overview
- Protocols
- **Algorithms**
- New and Future

Ring Algorithm



Rings on DGX A100

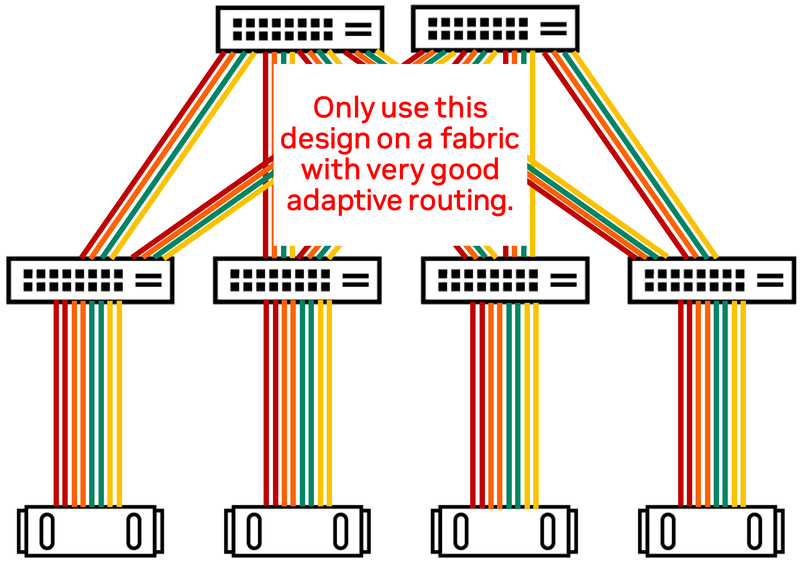


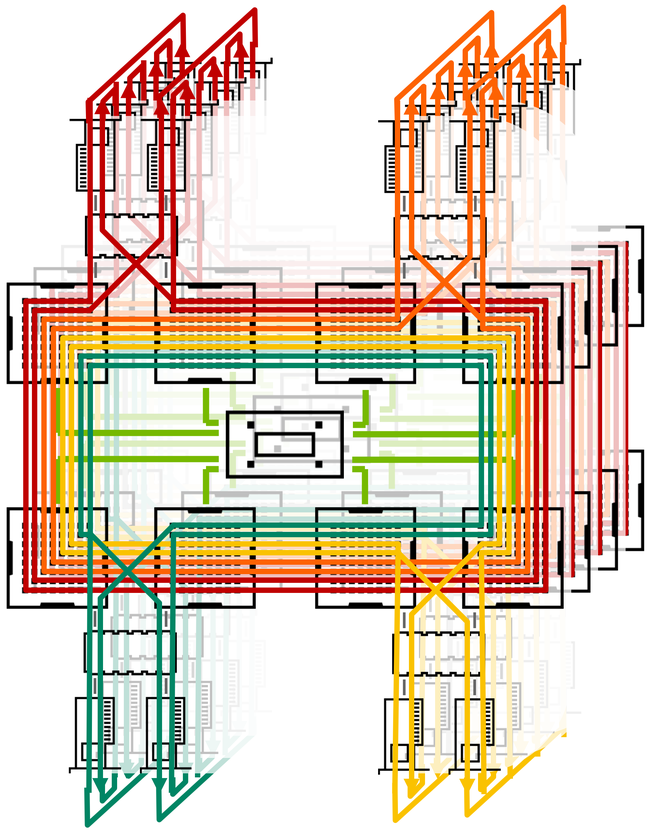
Inter-node Communication

Rail-optimized design

Classic fabric design

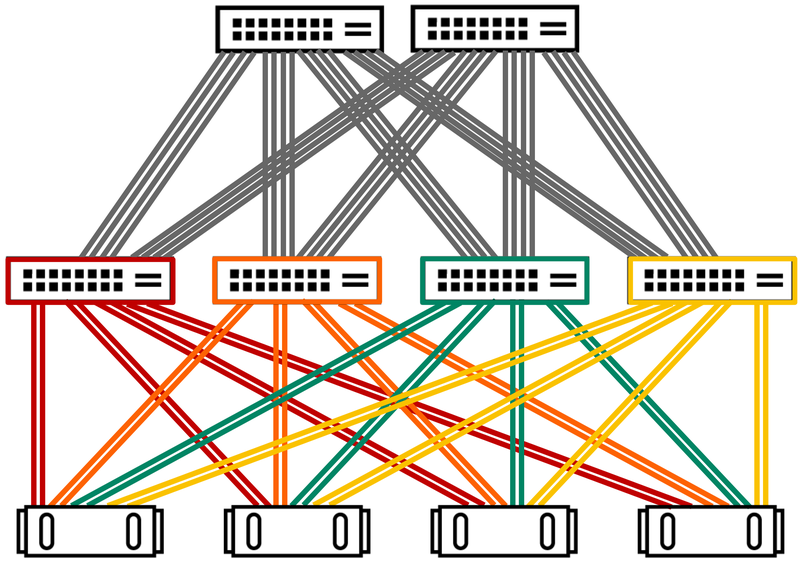
8 flows go across multiple level of switches.
Each flow needs to use a different path.



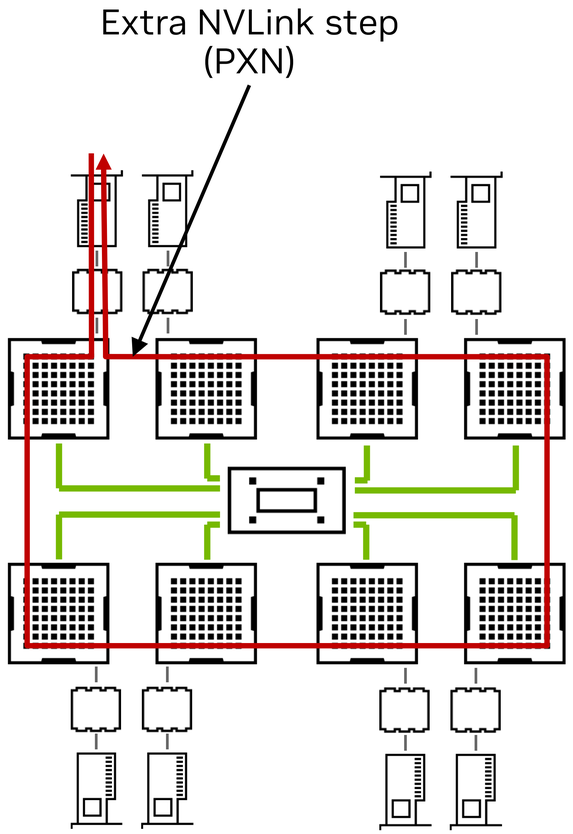


Rail-optimized design

Flows are physically separated.
Routing collisions cannot occur.



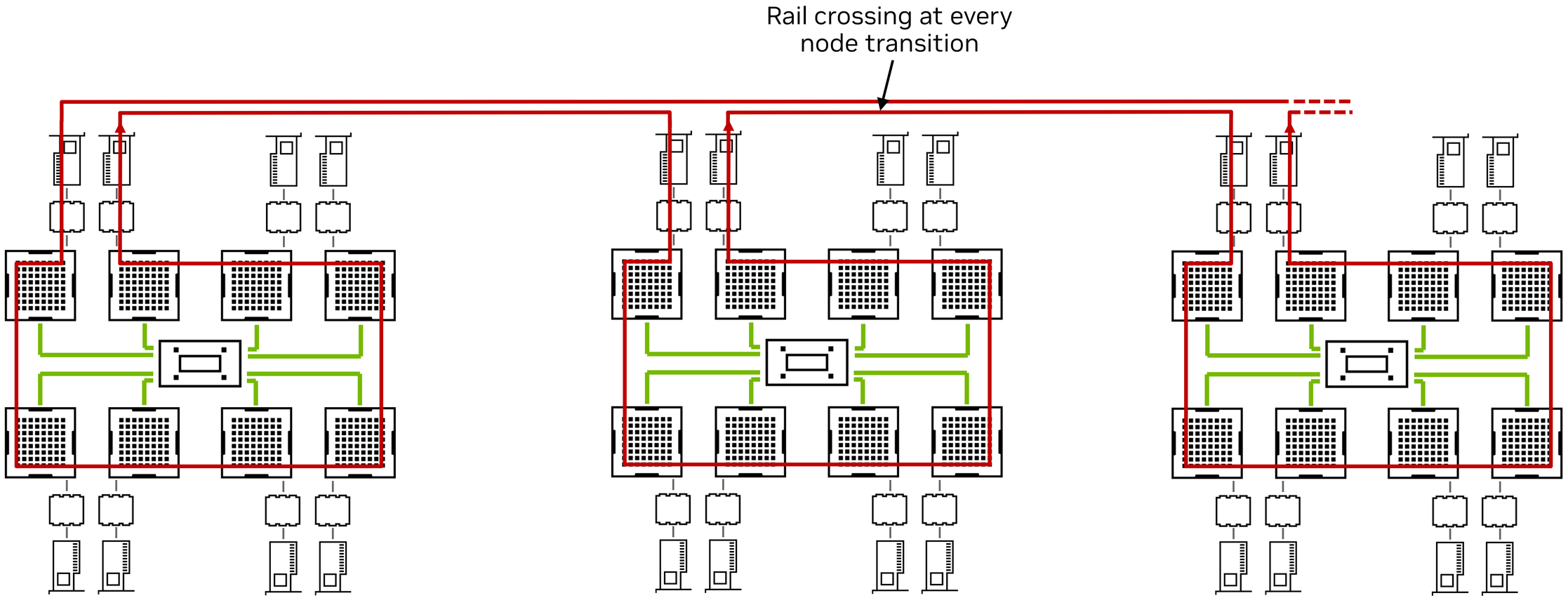
Rings on DGX H100



NVLink bandwidth: 370 GB/s
Total network bandwidth: 390 GB/s

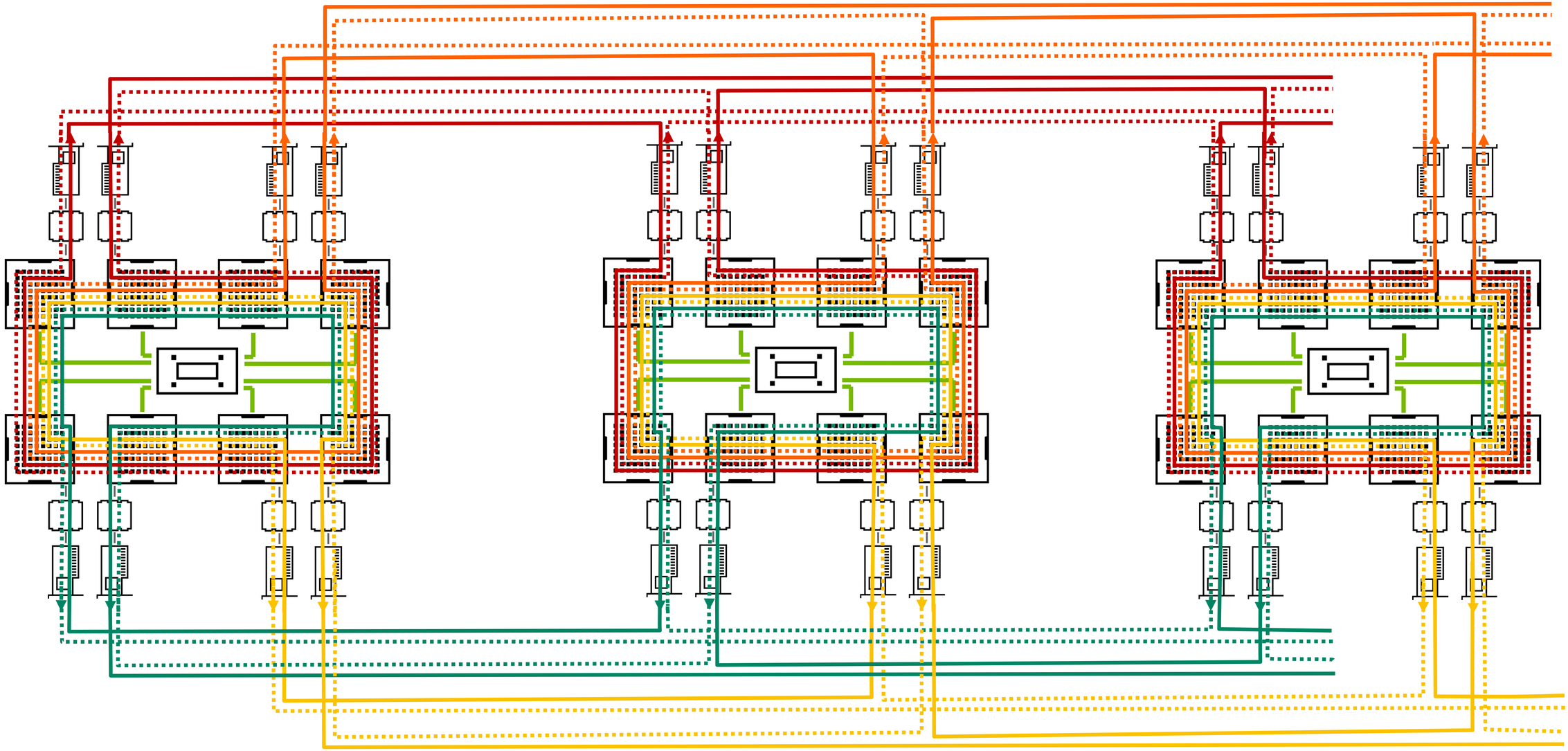
Ring max BW : 370 GB/s

Rings on DGX H100



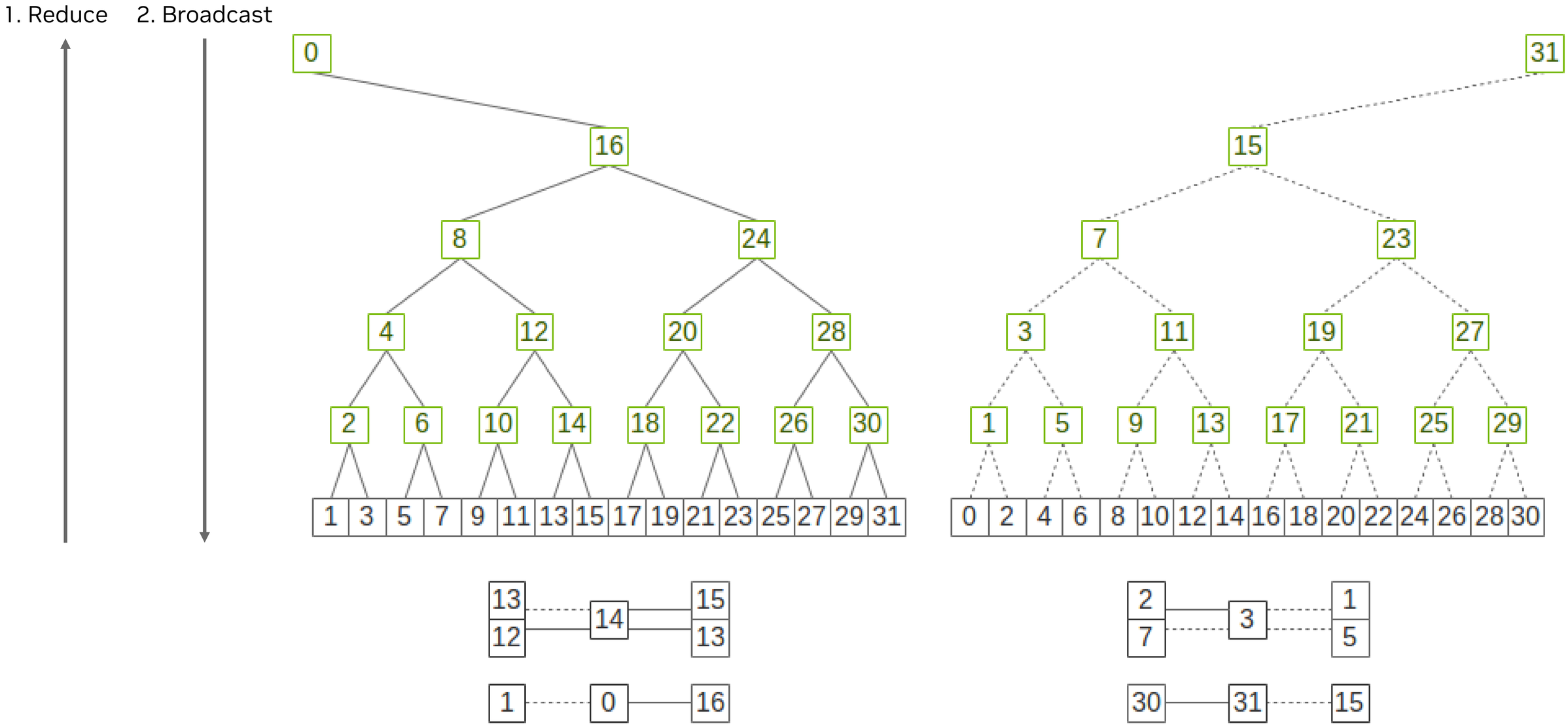
Ring max BW: 390 GB/s with inter-rail traffic

Alternating Rings

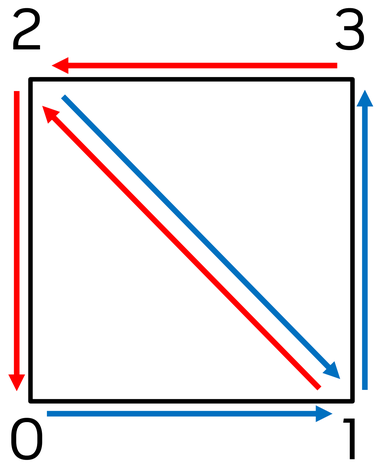


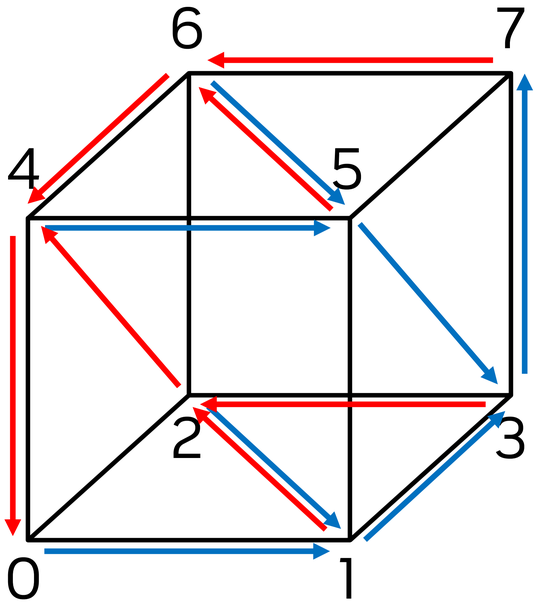
Ring max BW: 390 GB/s without inter-rail traffic

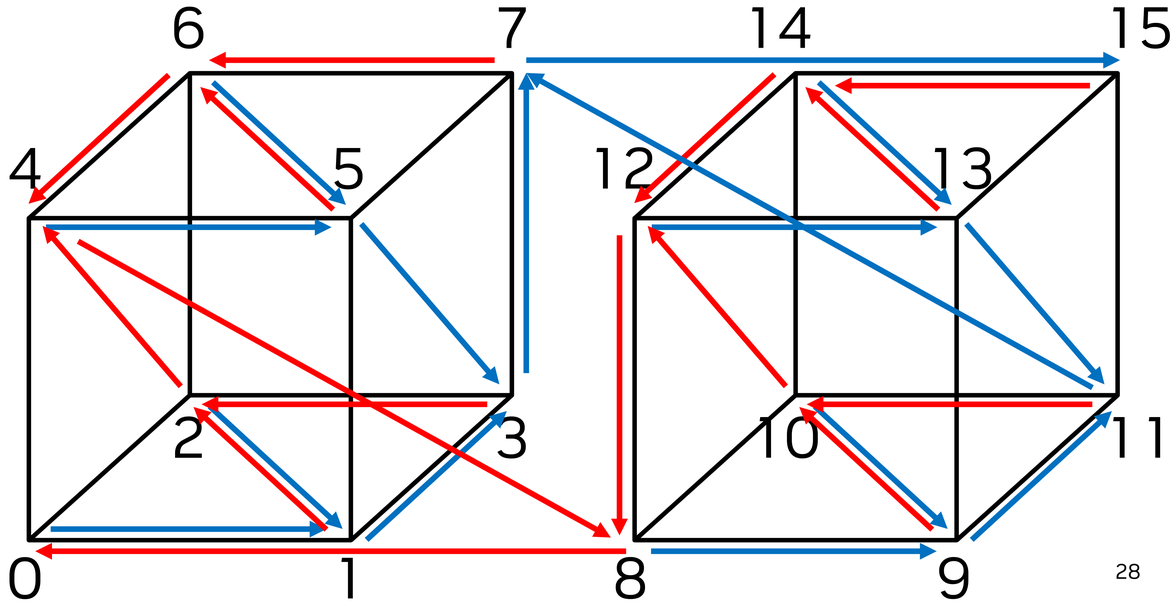
Tree Algorithm



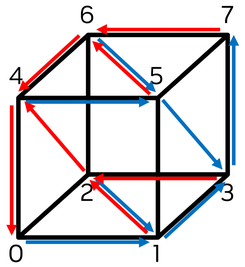
Tree Algorithm







Tree Algorithm



Tree #1

Tree #2

GPU A
Node 7

GPU A
Node 2

GPU A
Node 1

GPU A
Node 5

NIC A
Node 3

Input A

Input B

Input C

Input D

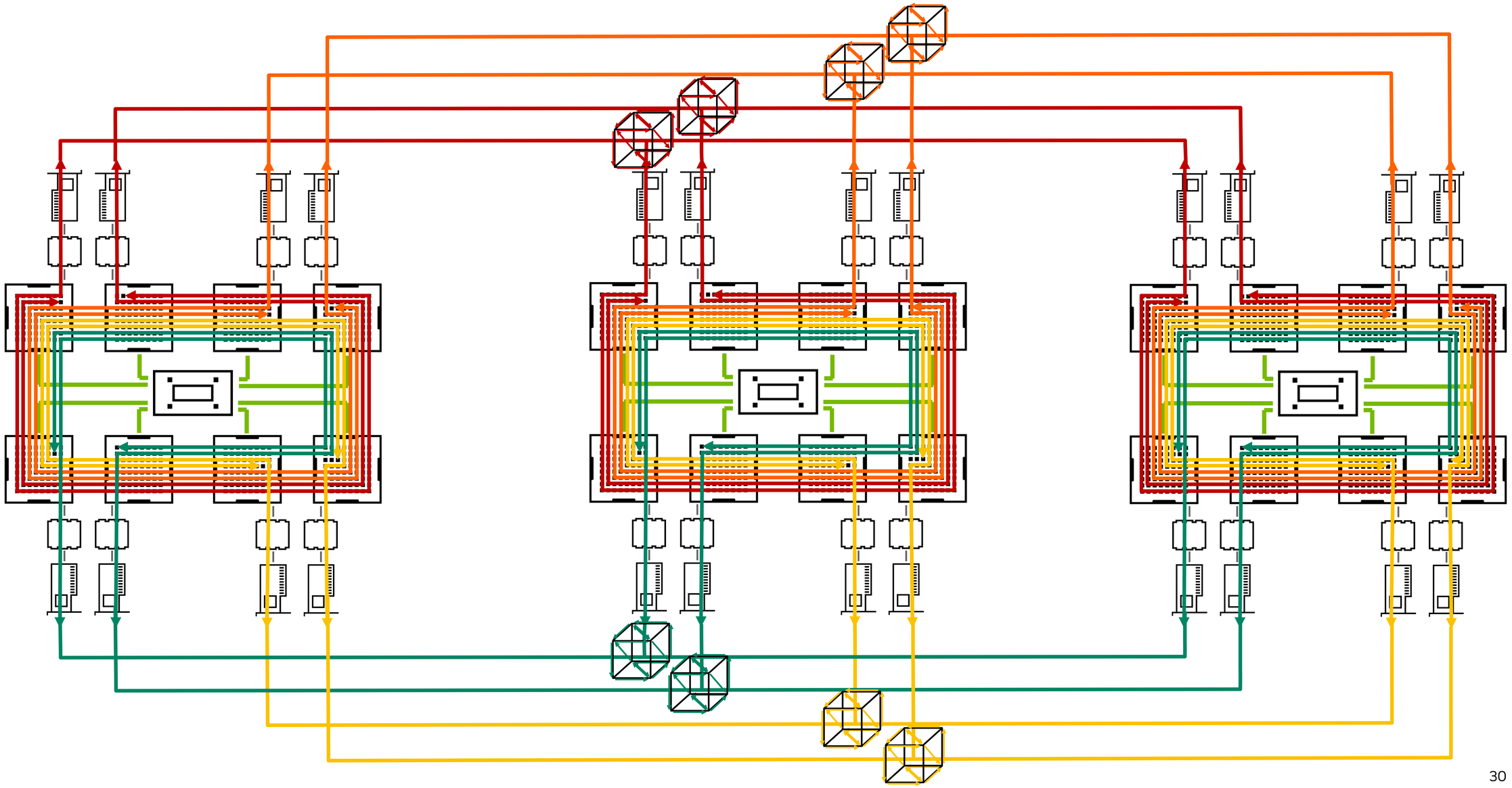
Output A

Output B

Output C

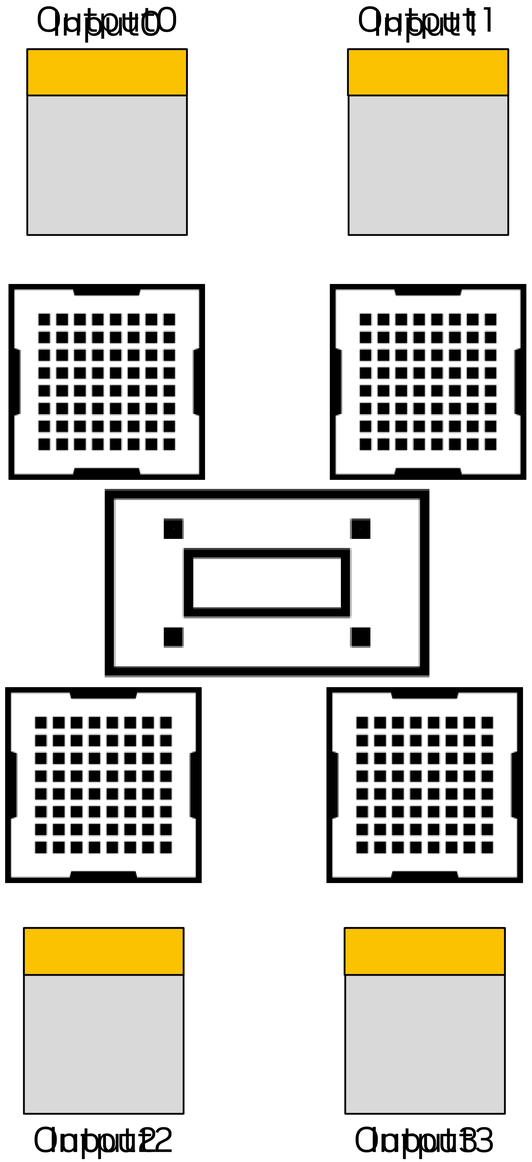
Output D

Trees on DGX H100



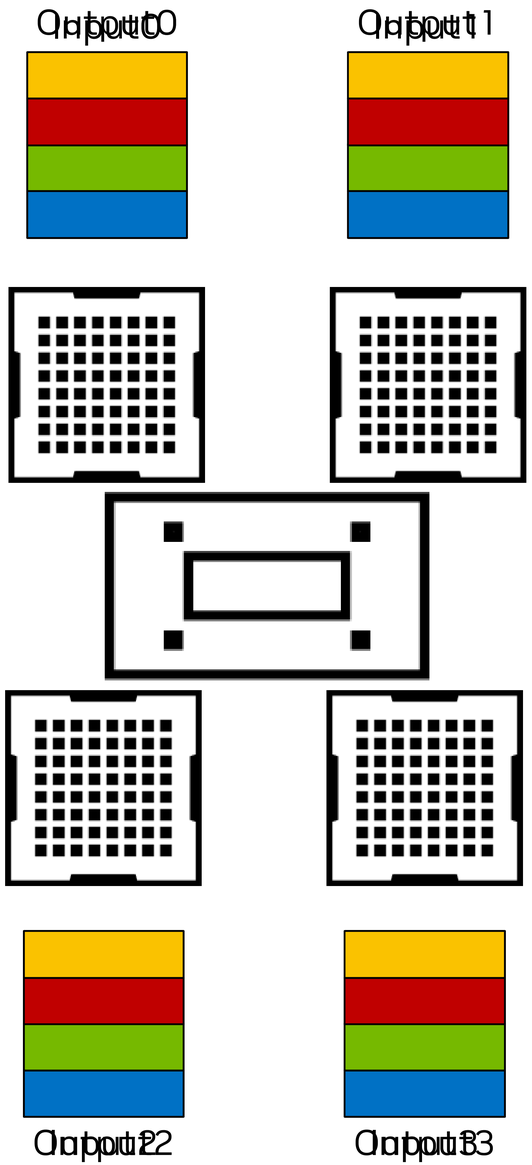
H100 SHARP

NVLink SHARP



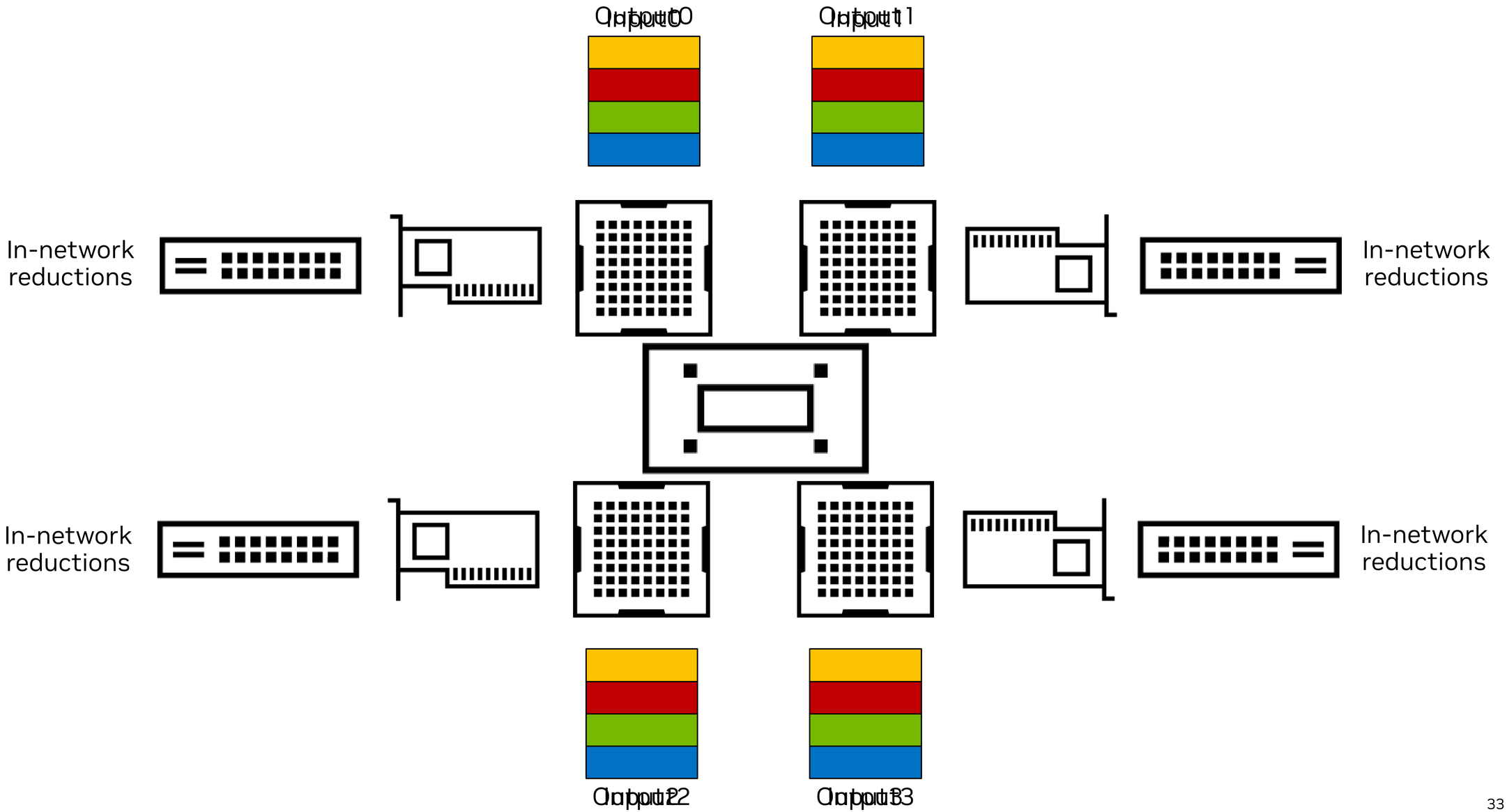
H100 SHARP

NVLink SHARP



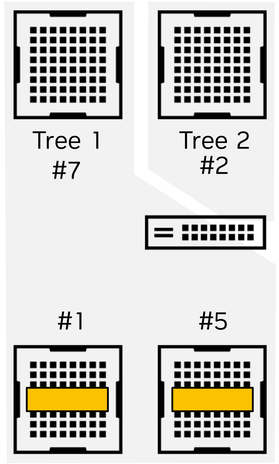
H100 SHARP

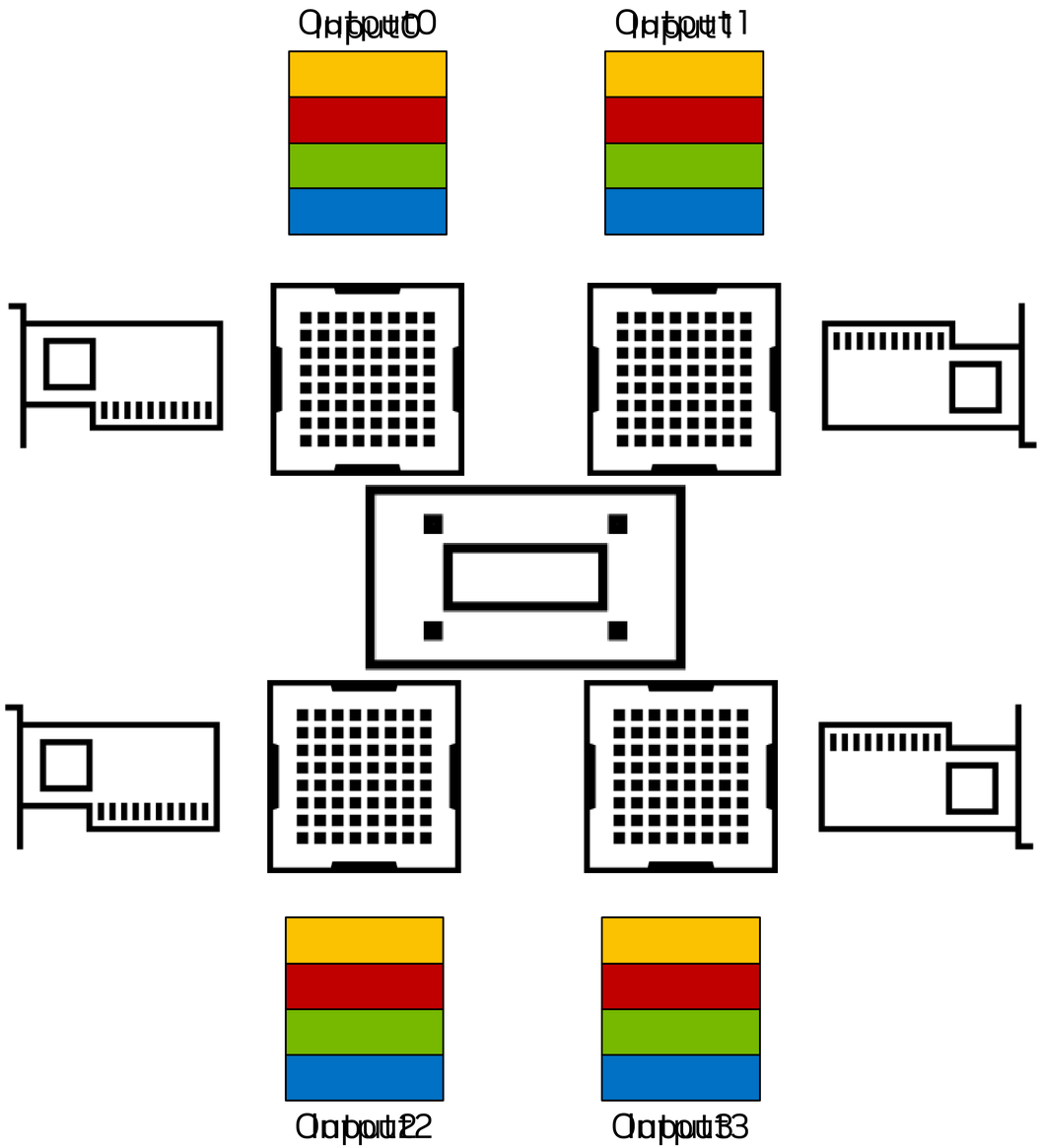
NVLink and IB



NVLSTree Algorithm

Double binary tree



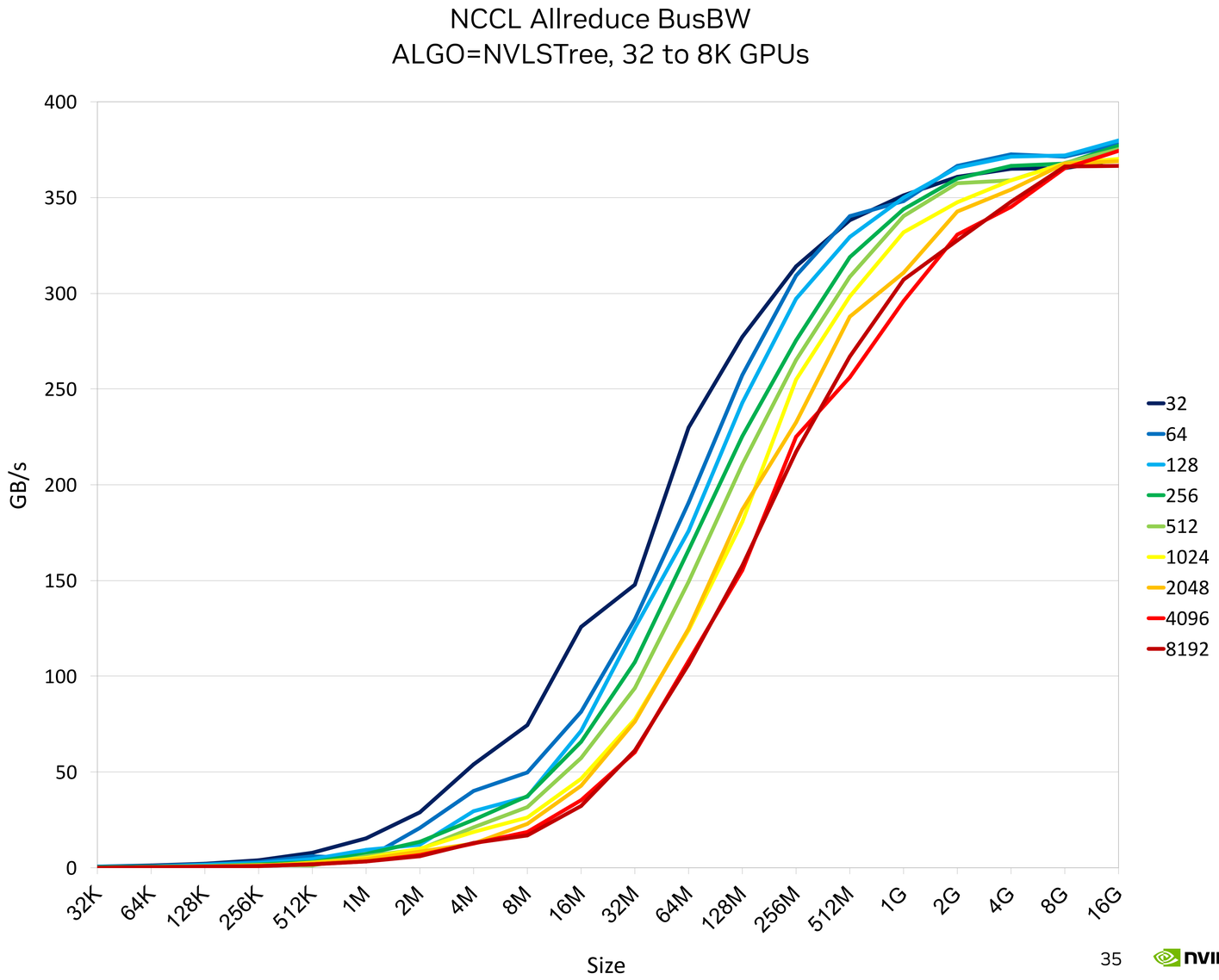


NVLSTree Performance

Logarithmic latency causes very little shift to the right as we scale.

Still able to reach peak bandwidth at up to 8K GPUs.

Offload of intra-node reductions allows to reach that bandwidth with only 16 SMs, and only 6 SMs with registered buffers.



Agenda

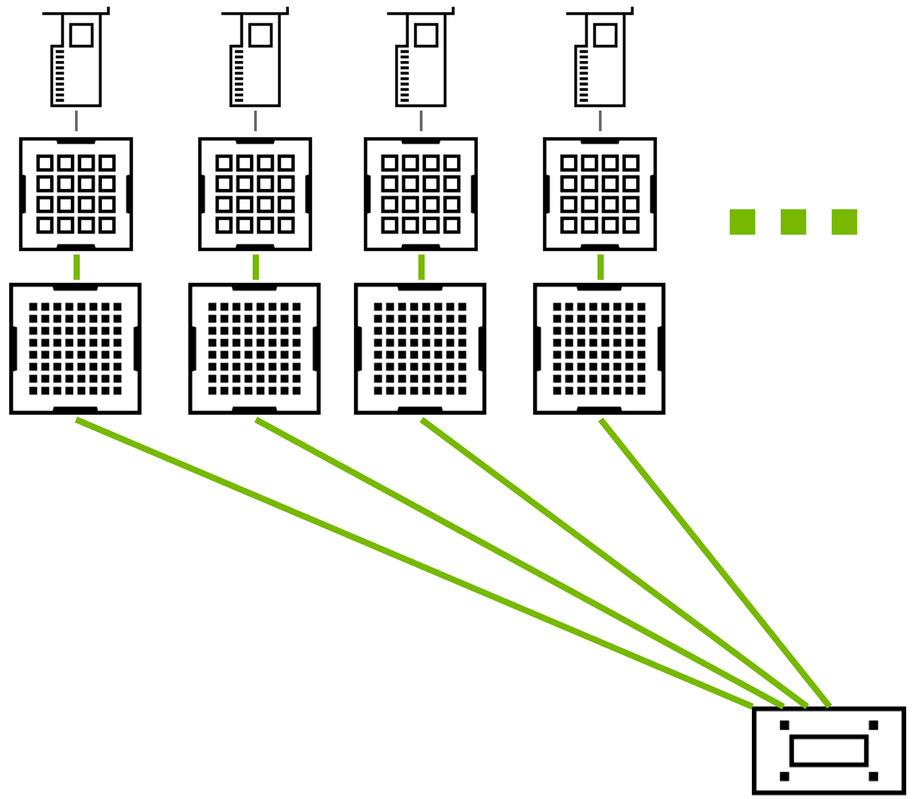
- Deep Learning Training
- NCCL Overview
- Protocols
- Algorithms
- New and Future

Grace Hopper Support

Use new NVML APIs to:

- Detect high-bandwidth C2C links between Grace CPUs and Hopper GPUs

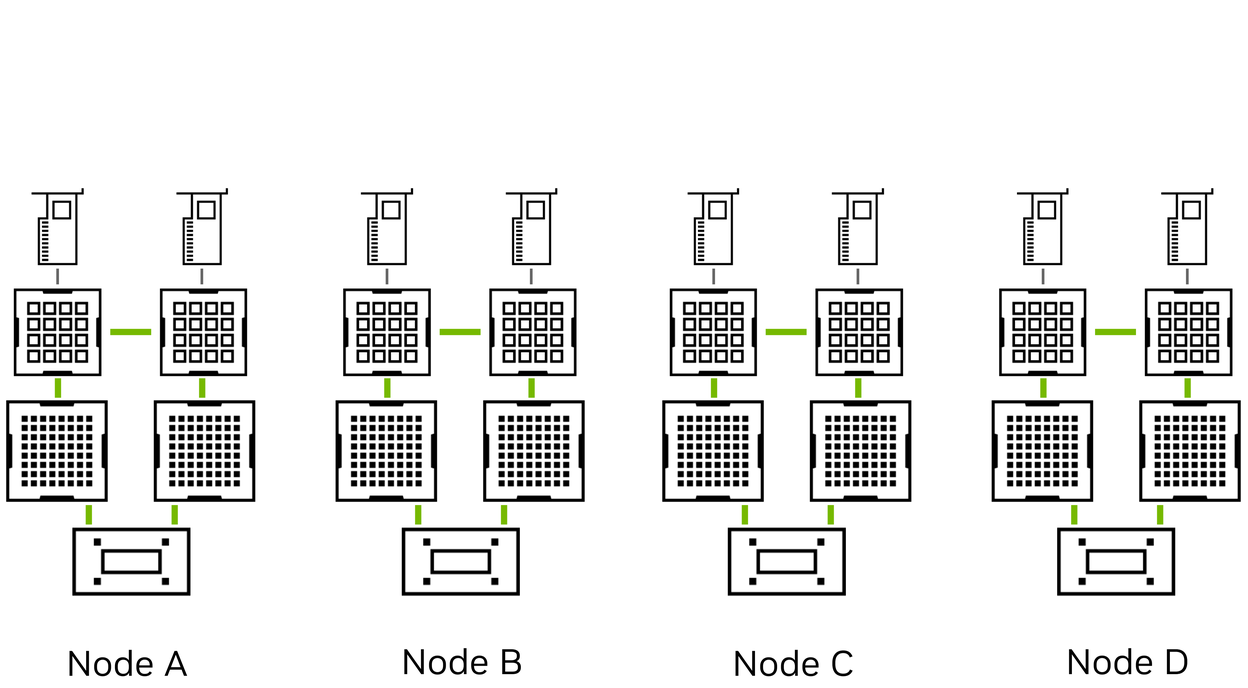
- Detect multi-node NVLink domain (clique ID)



Multi-node NVLink Topology detection

Topology graph with all GPUs within the NVLink domain:

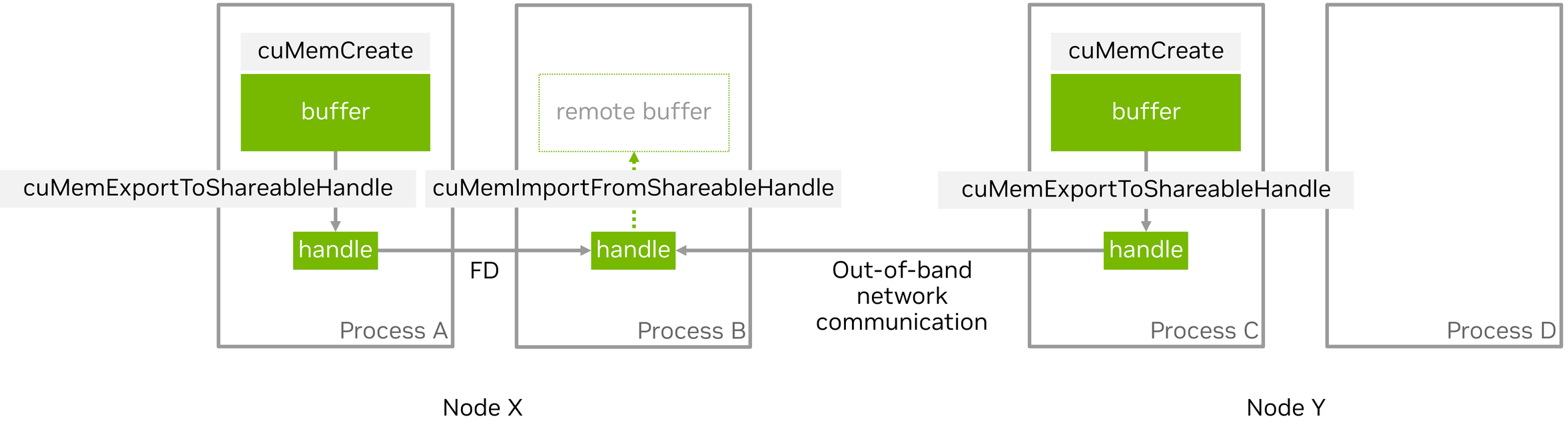
- Merge intra-node topology with other ranks in the same NVLink domain
- All topology detection in NCCL works as if the NVLink domain was a single node



Multi-node NVLink Communication

Multi-node NVLink is mostly the same as intra-node NVLink communication:

- uses existing CUDA buffer mapping mechanism, only with a fabric handle (CUDA 12.4).
- communicates via load/stores through NVLink
- requires an imex channel to be created (see CUDA documentation)



Other features

User buffer registration

New ncclCommRegister function to allow NCCL to access buffers directly, avoiding a copy from user buffer to NCCL's internal buffers (zero-copy).

Can help a lot when NCCL operations are overlapped with compute kernels.

Currently used by NVLink SHARP communication, network send/receive operations, and pure IB SHARP network operations.

Fault tolerance and elastic training

Allow applications to tear-down a NCCL communicator (e.g. in case of an error) and recreate a new one with new/more/less GPUs.

GPU-initiated network communication (IB GDA-KI)

[S61368] Magnum IO GPUDirect, NCCL, NVSHMEM, and GDA-KI on Grace Hopper and Hopper systems

Summary

NCCL : Key communication library for DL training and multi-GPU computing.

Optimized for all platforms, from desktop, to DGX Superpod, to cloud.

Download from https://developer.nvidia.com/nccl and in NGC containers.
Source code at https://github.com/nvidia/nccl

Feel free to join our connect with the NVIDIA experts session:
[CWE61229] Inter-GPU Communication Techniques and Libraries for HPC and AI
Wednesday, Mar 20 10:00 AM - 10:50 AM PDT - CWE Pod D (LL)