

S41784: FAST INTER-GPU COMMUNICATION WITH NCCL FOR DEEP LEARNING TRAINING, AND MORE

SYLVAIN JEAUGEY, DAVID ADDISON

MULTI-GPU COMPUTING

NCCL : Key communication library for multi-GPU computing.
Optimized for all platforms, from desktop to DGX Superpod.

PCI



NVLink



NVLink +
Infiniband



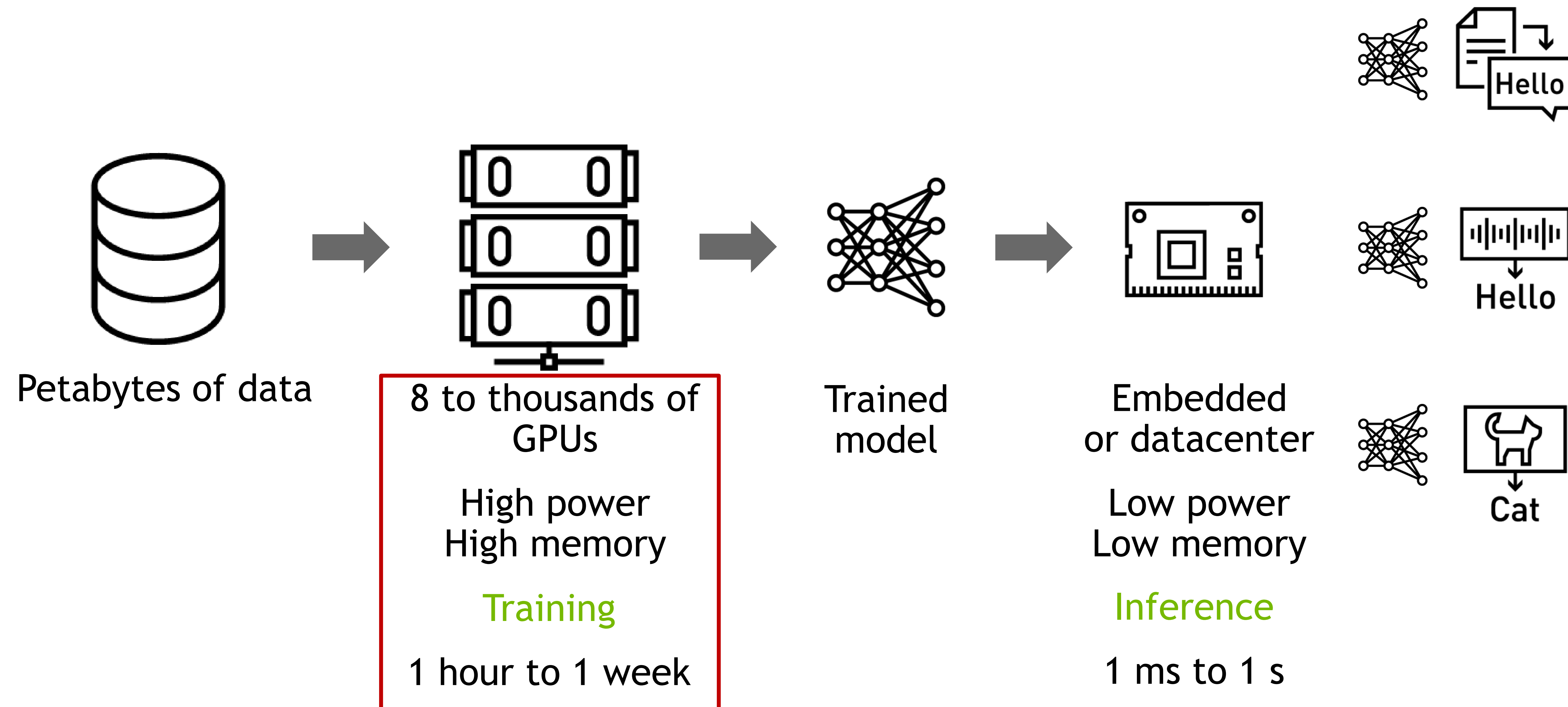
Download from <https://developer.nvidia.com/nccl> and in NGC containers.
Source code at <https://github.com/nvidia/nccl>



DEEP LEARNING TRAINING

DEEP LEARNING

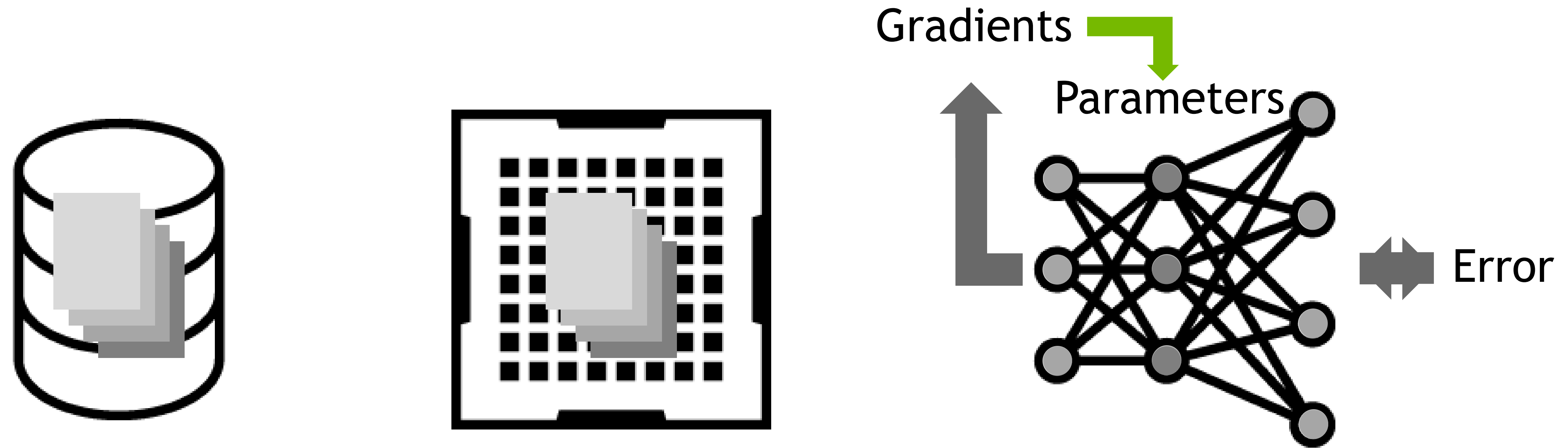
From data to AI



NCCL : make multi-GPU training as efficient as possible to reduce training time.

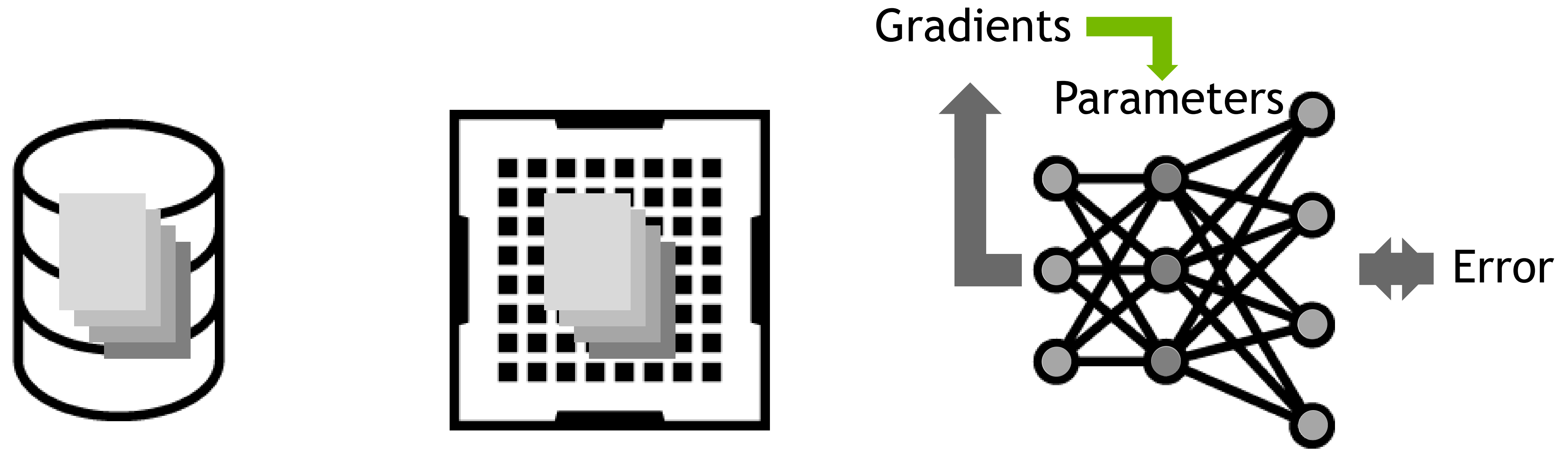
DEEP LEARNING TRAINING

On a single GPU



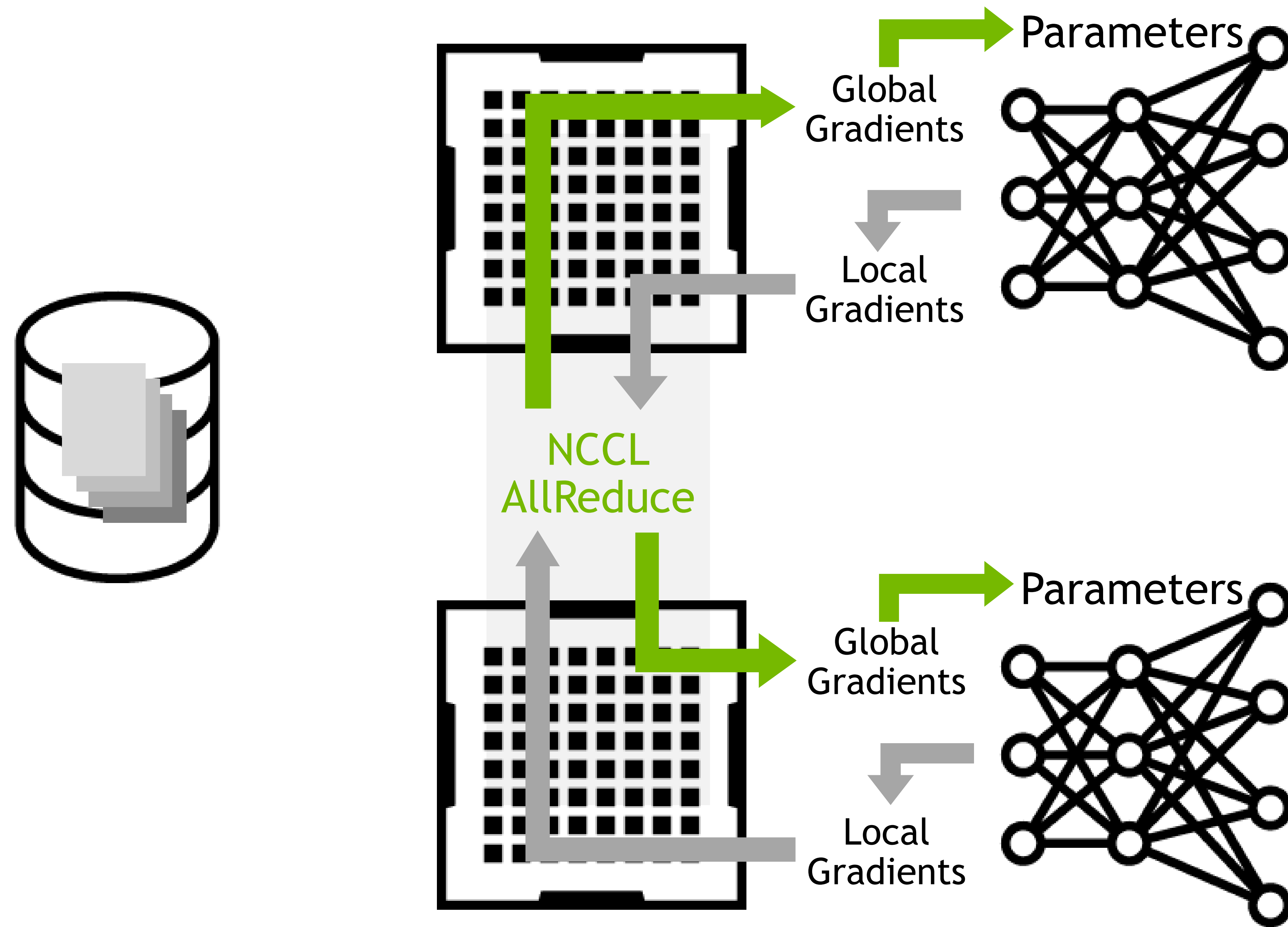
DEEP LEARNING TRAINING

On a single GPU



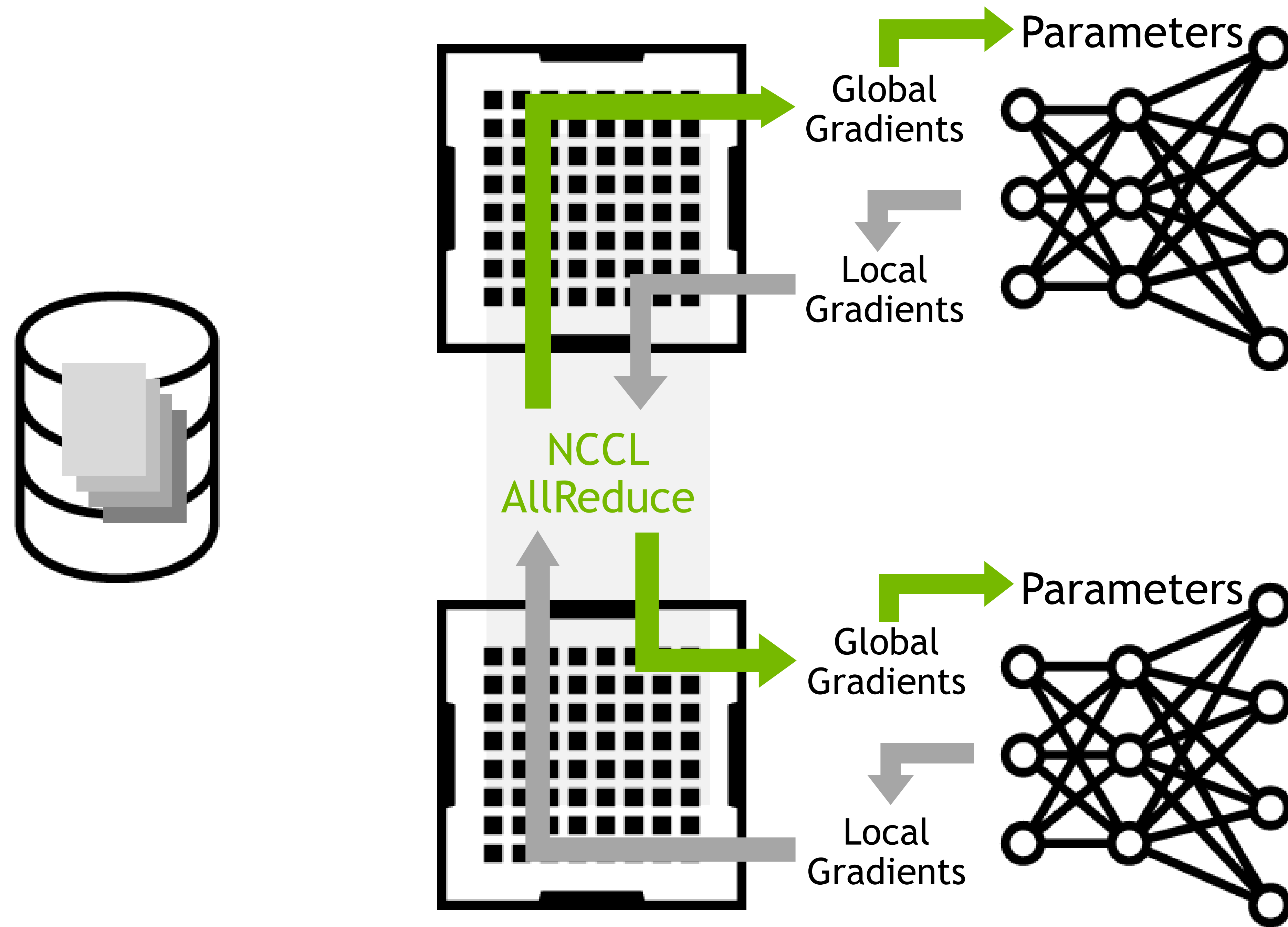
DEEP LEARNING TRAINING

On multiple GPUs

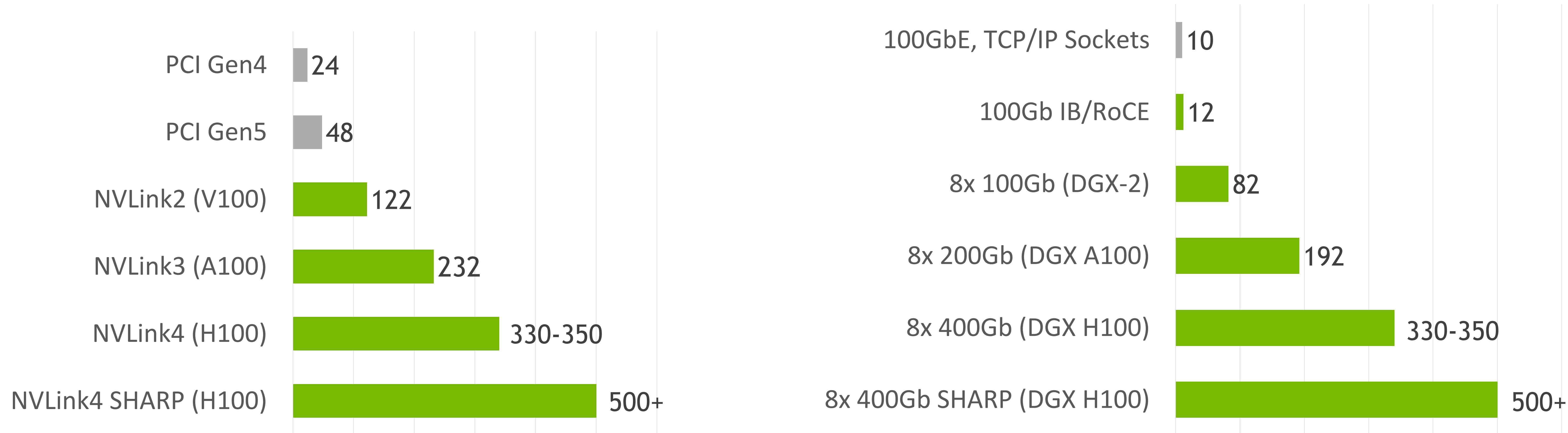
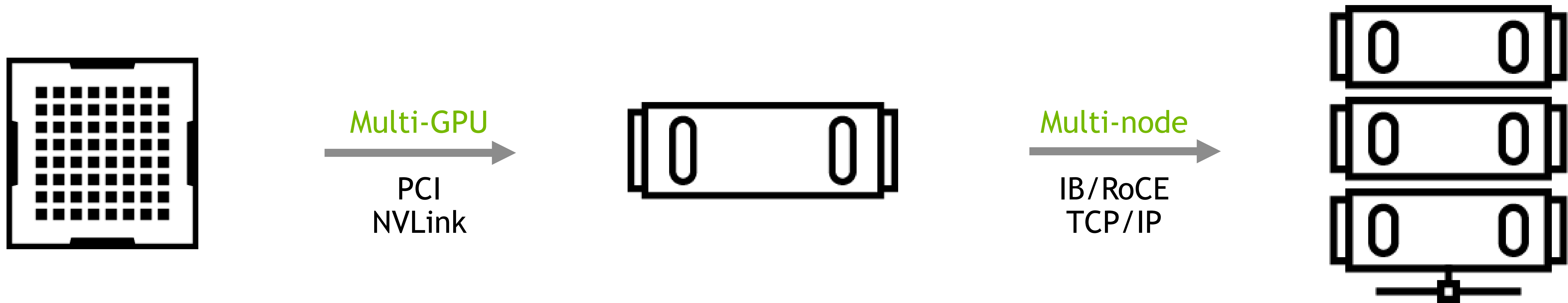


DEEP LEARNING TRAINING

On multiple GPUs



ALLREDUCE PERFORMANCE



NCCL tests Allreduce Bus Bandwidth in GB/s

DEEP LEARNING TRAINING

Why NCCL performance is critical

DL training at scale, PCI Gen 3 + 100G

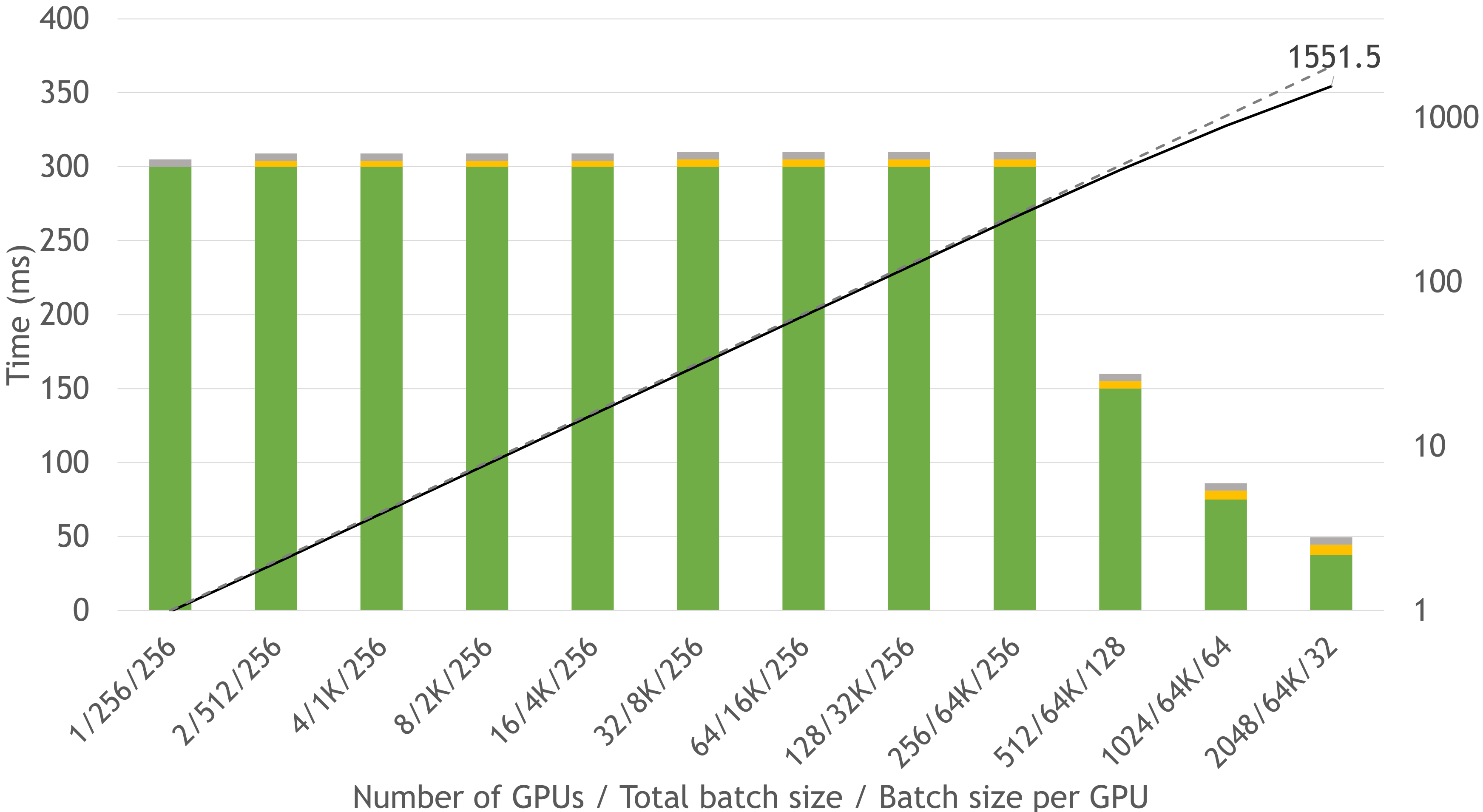


Efficiency: 77%

Max speedup:
640x

Theoretical model of size
500MB, compute time
300ms / 256 samples

DL training at scale, NVLink + 8x200G



Efficiency: 97%

Max speedup:
4800x

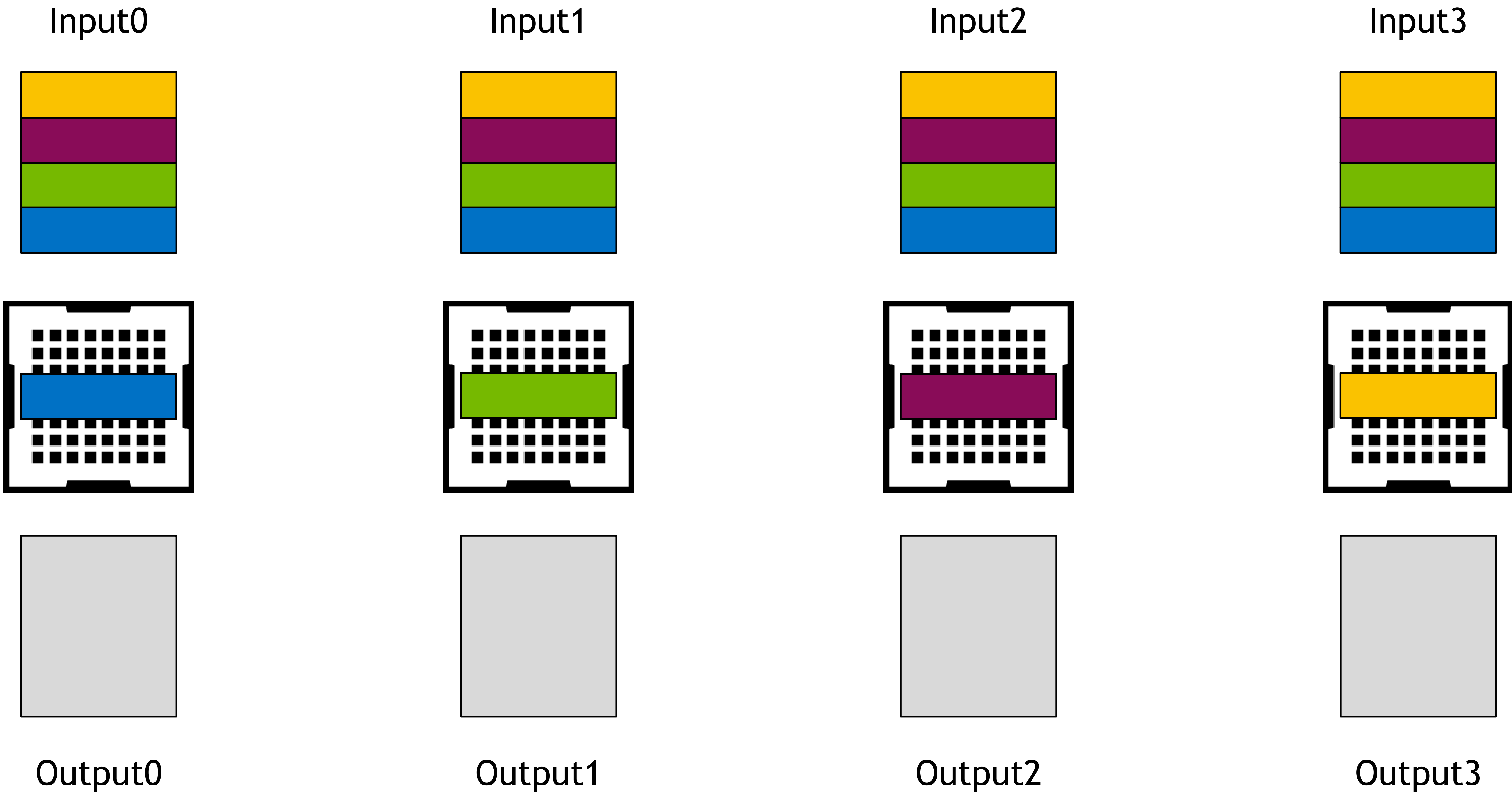




ALLREDUCE ALGORITHMS

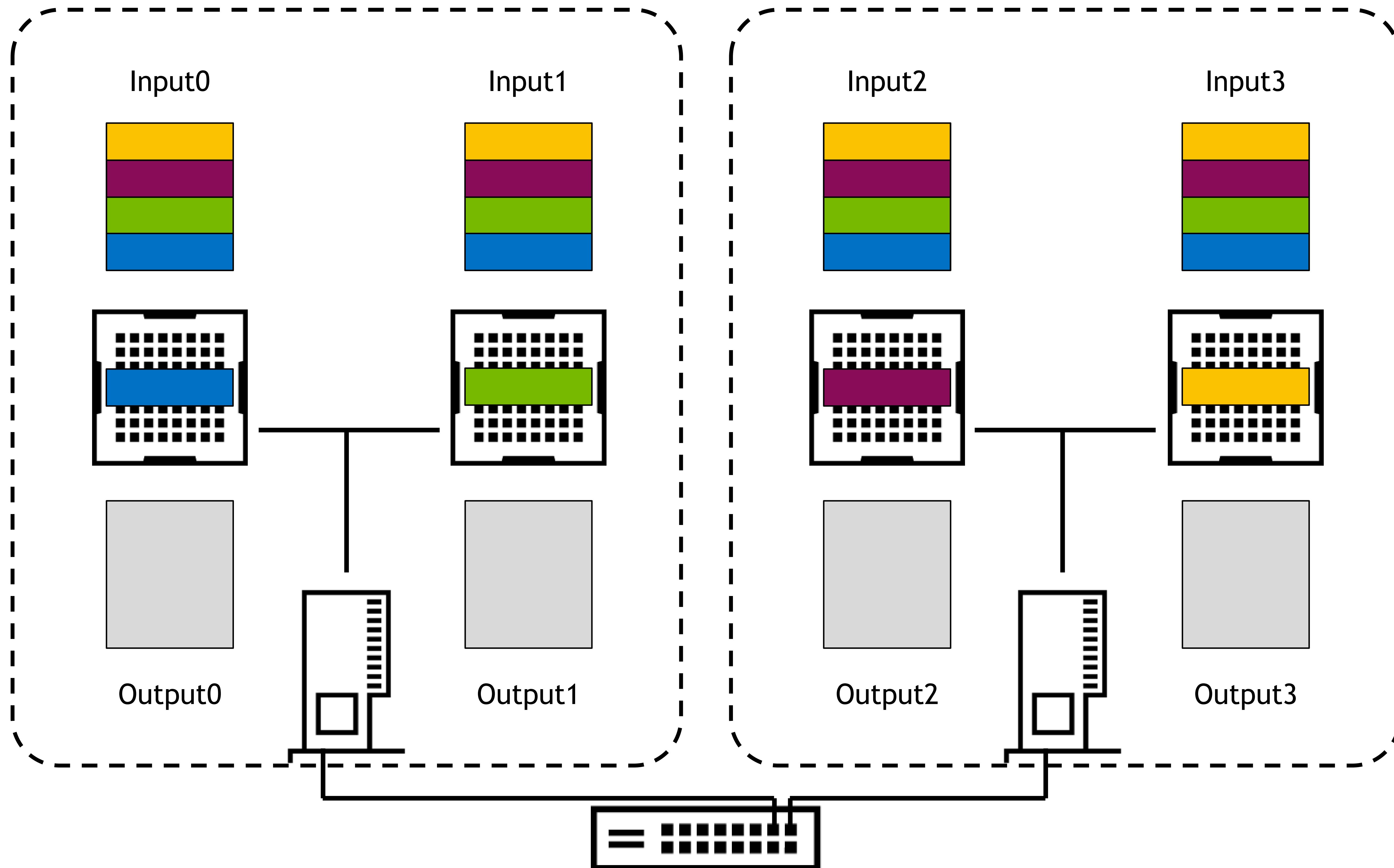
RING ALGORITHM

For allreduce



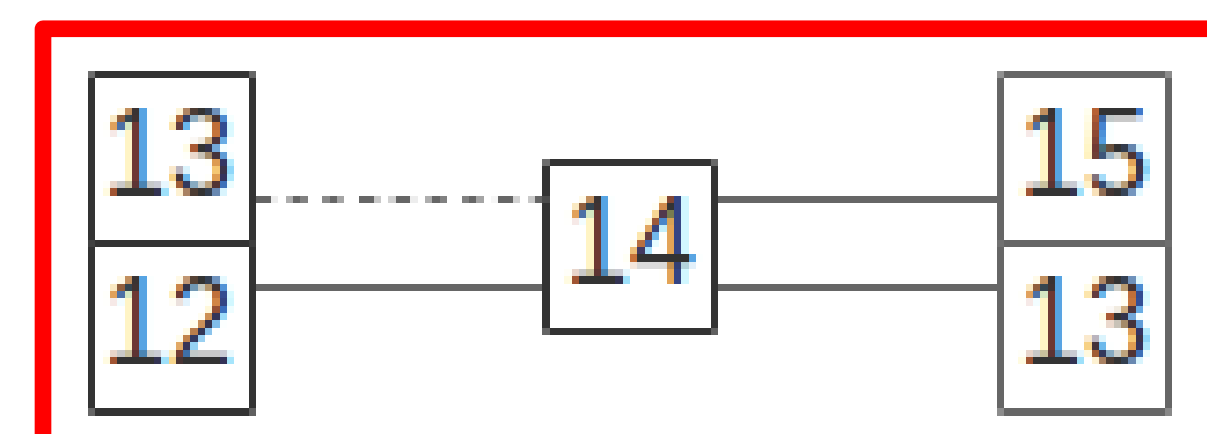
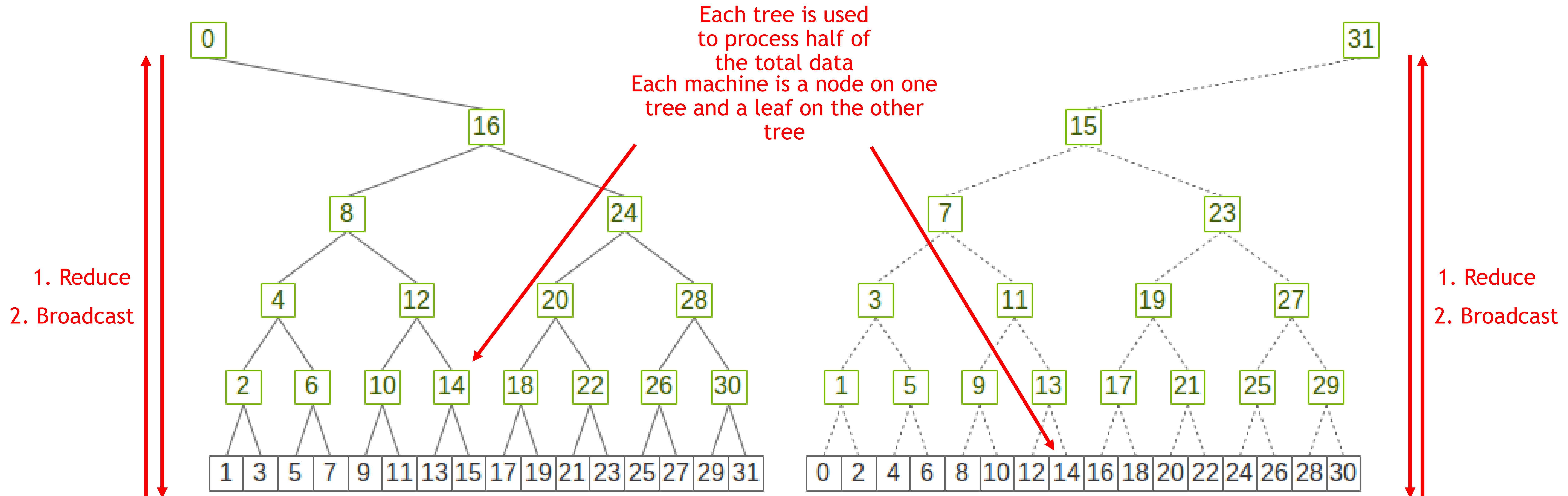
RING ALGORITHM

For allreduce

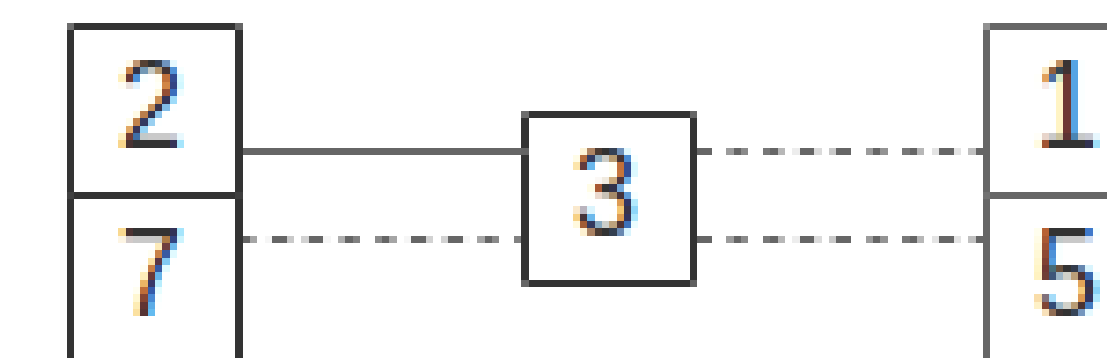


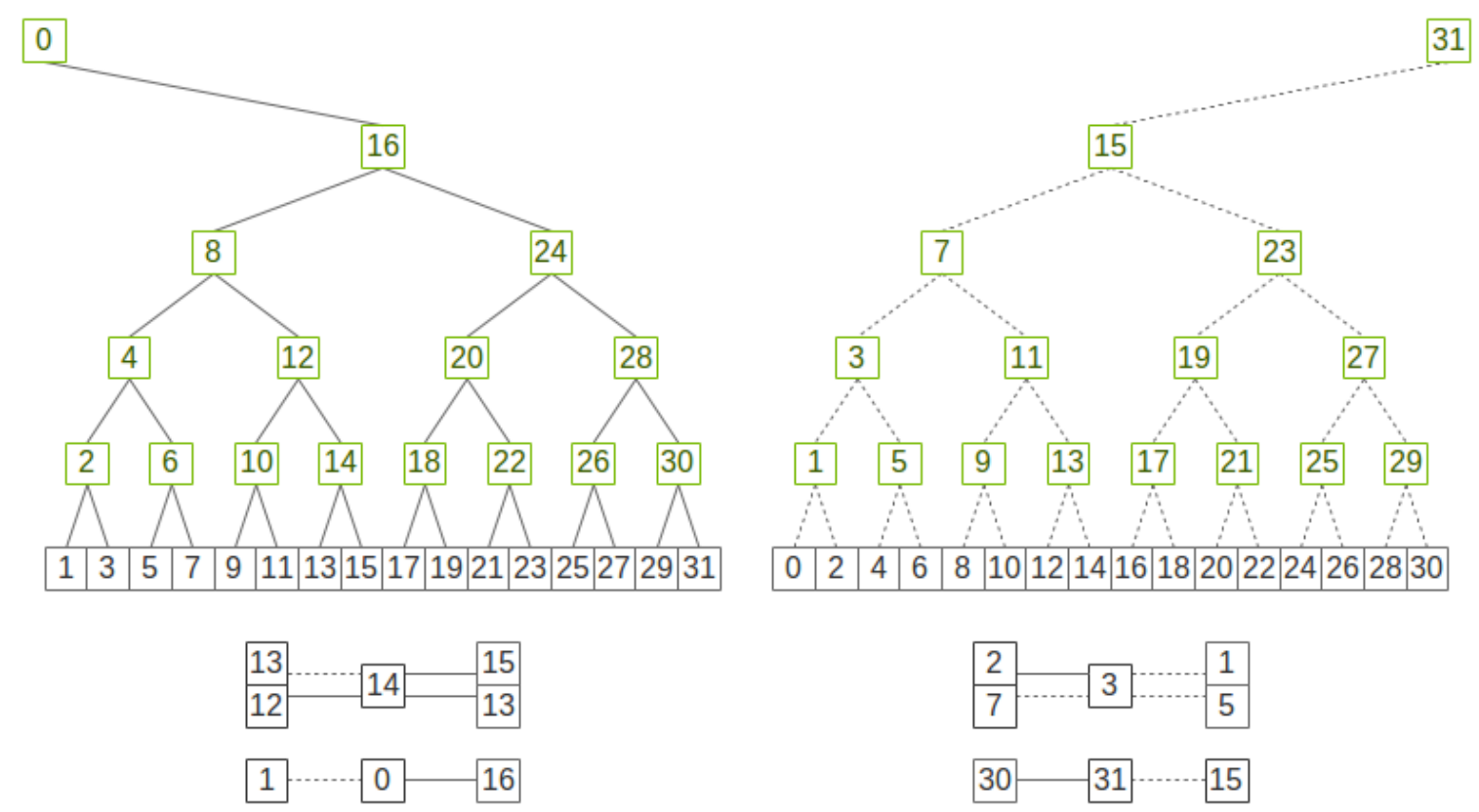
TREE ALGORITHM

For allreduce



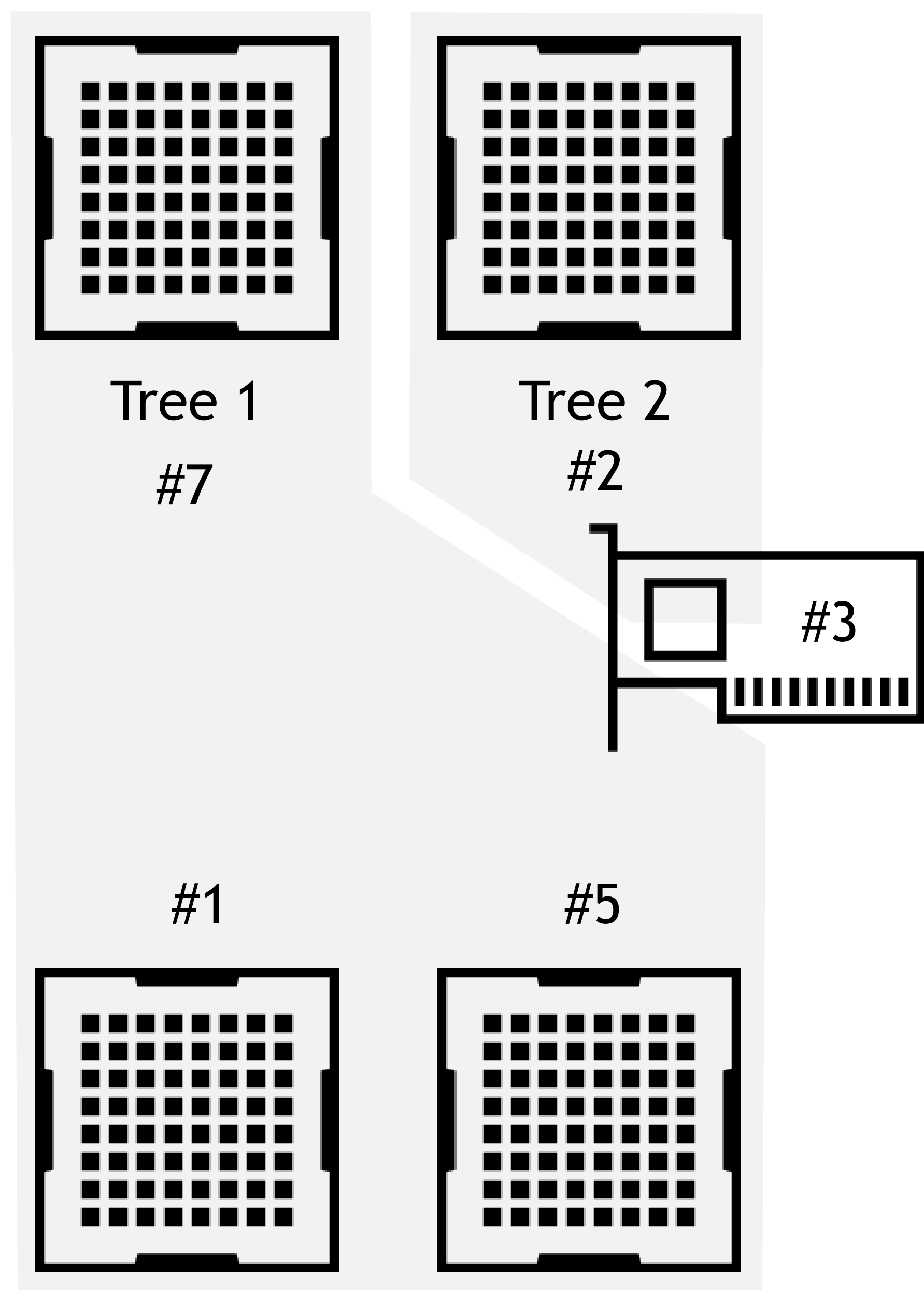
Each machine receives 2x half and sends 2x half



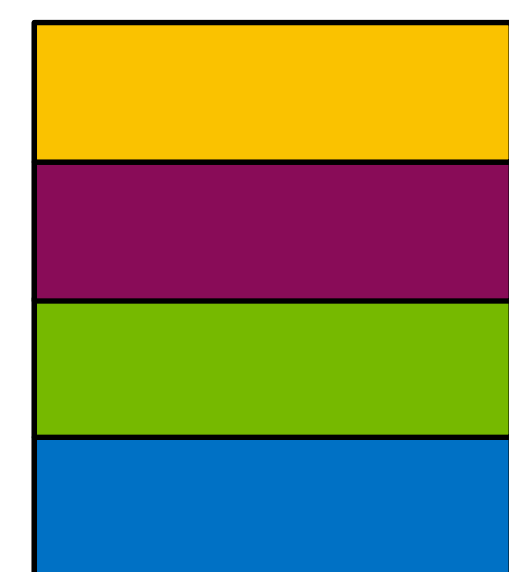


TREE ALGORITHM

For allreduce



Input0



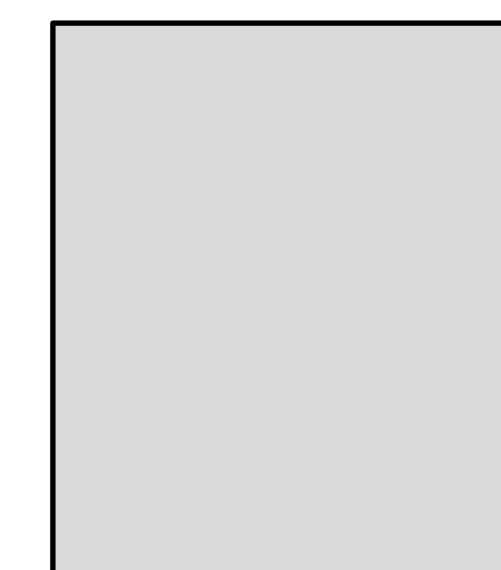
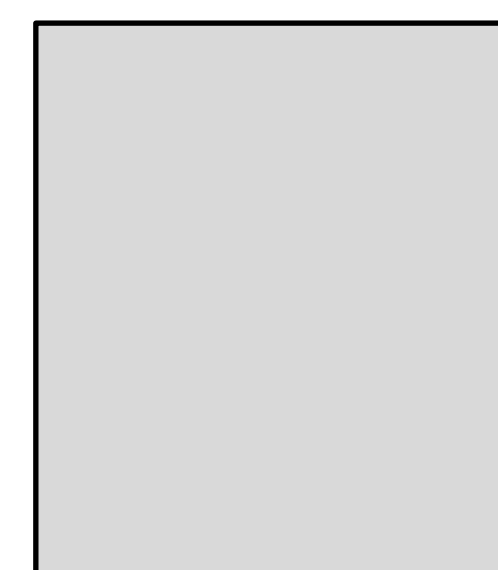
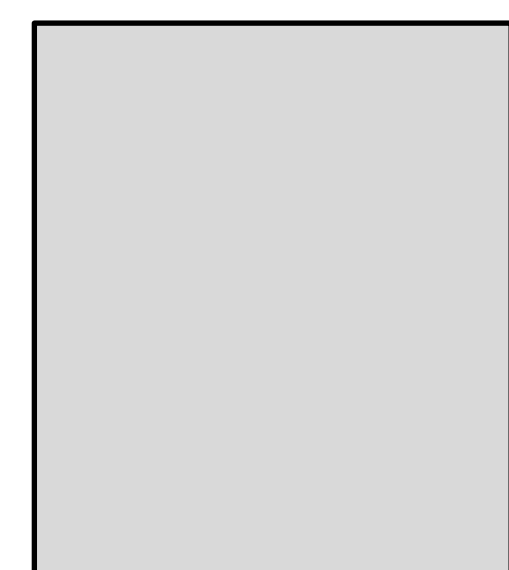
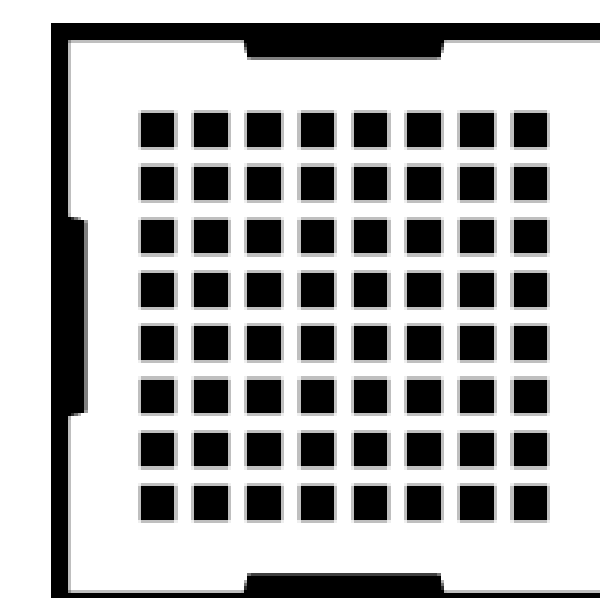
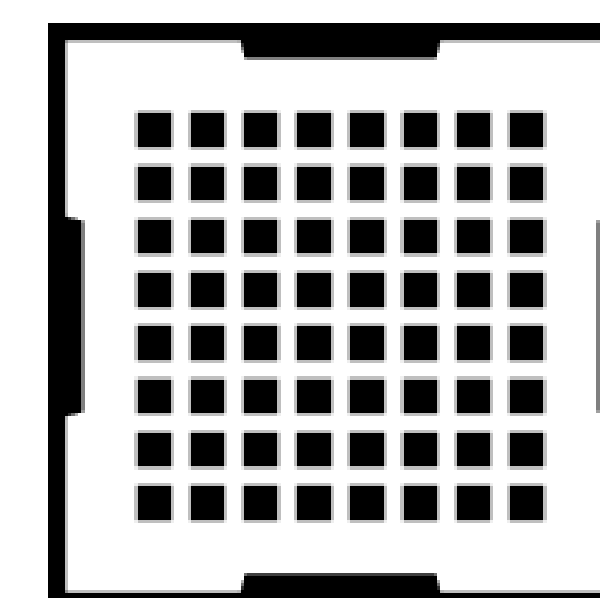
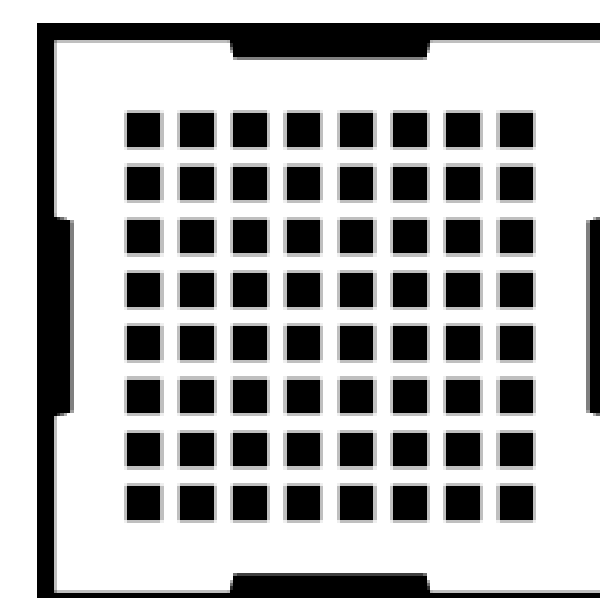
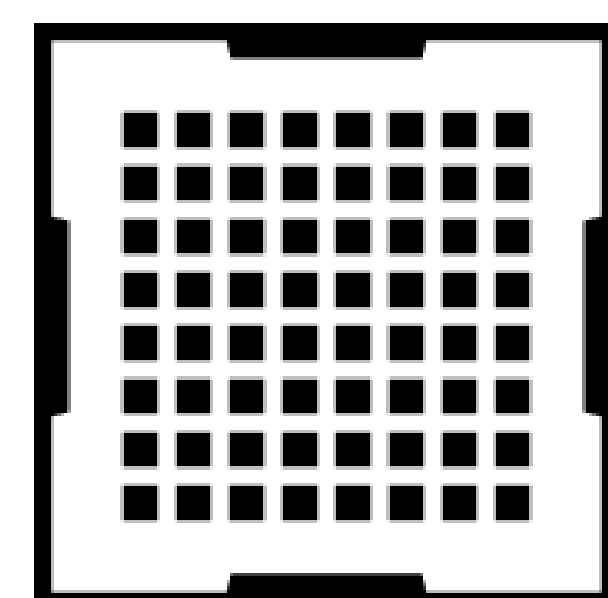
Input1



Input2



Input3



Output0

Output1

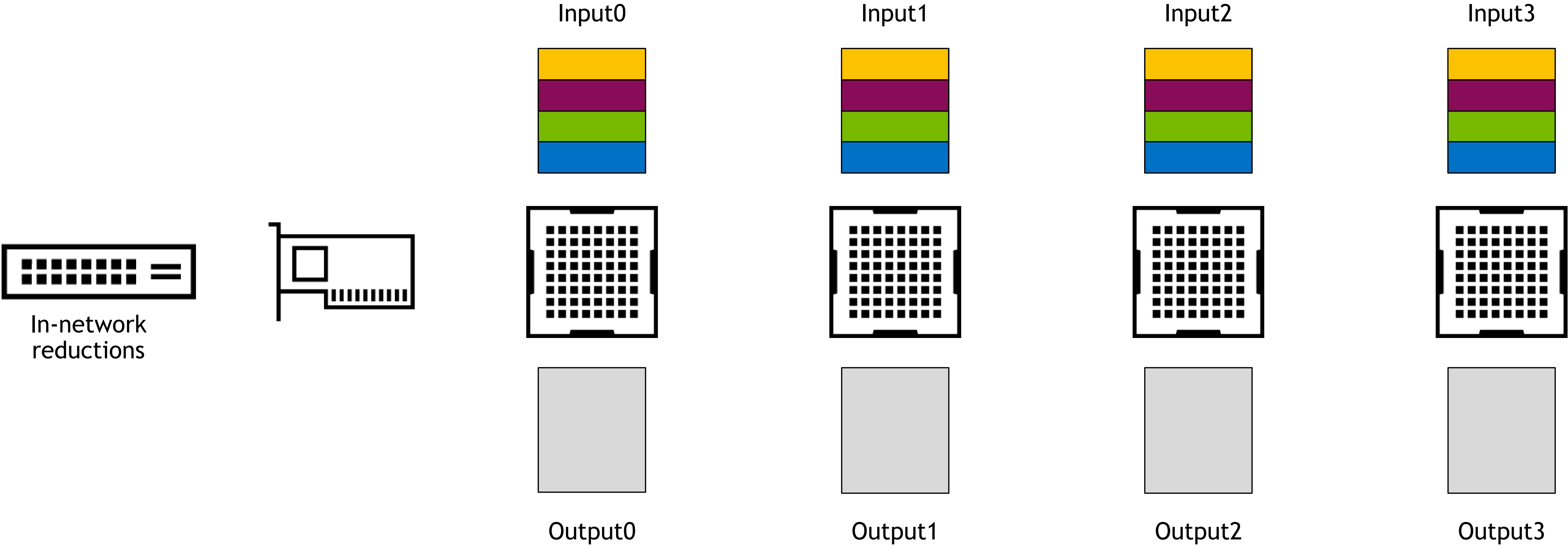
Output2

Output3



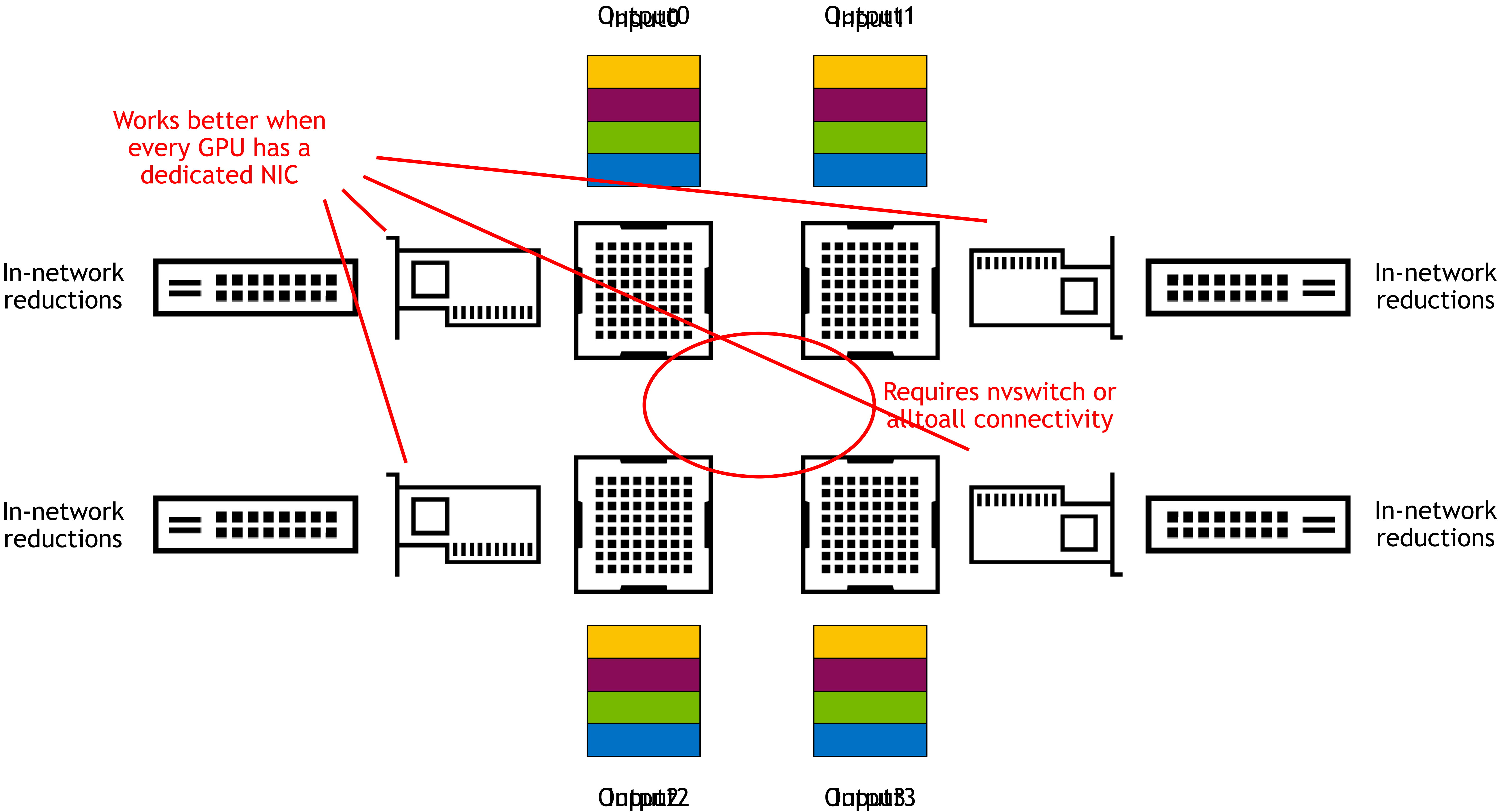
COLLNET ALGORITHM

Chain version



COLLNET ALGORITHM

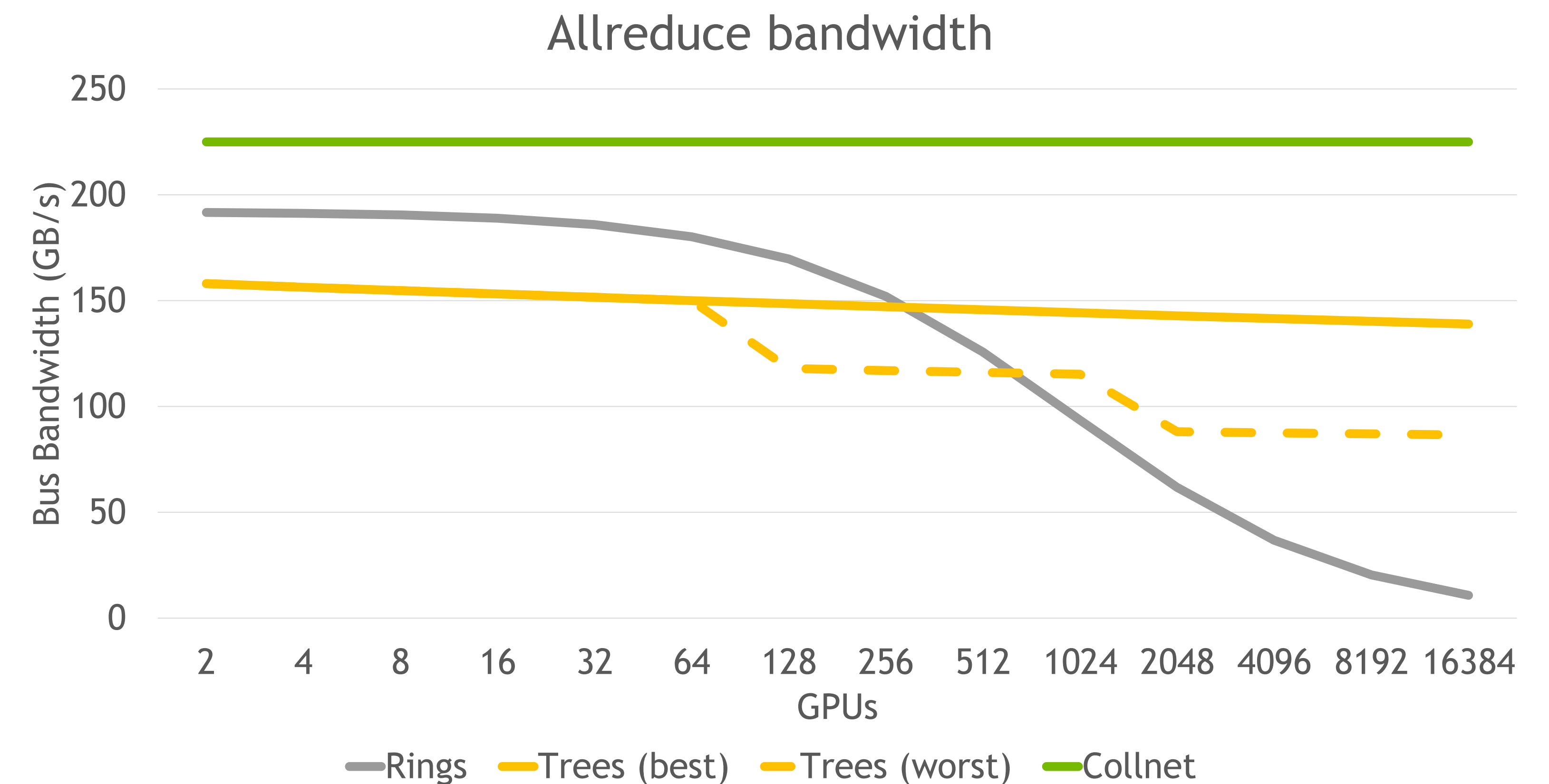
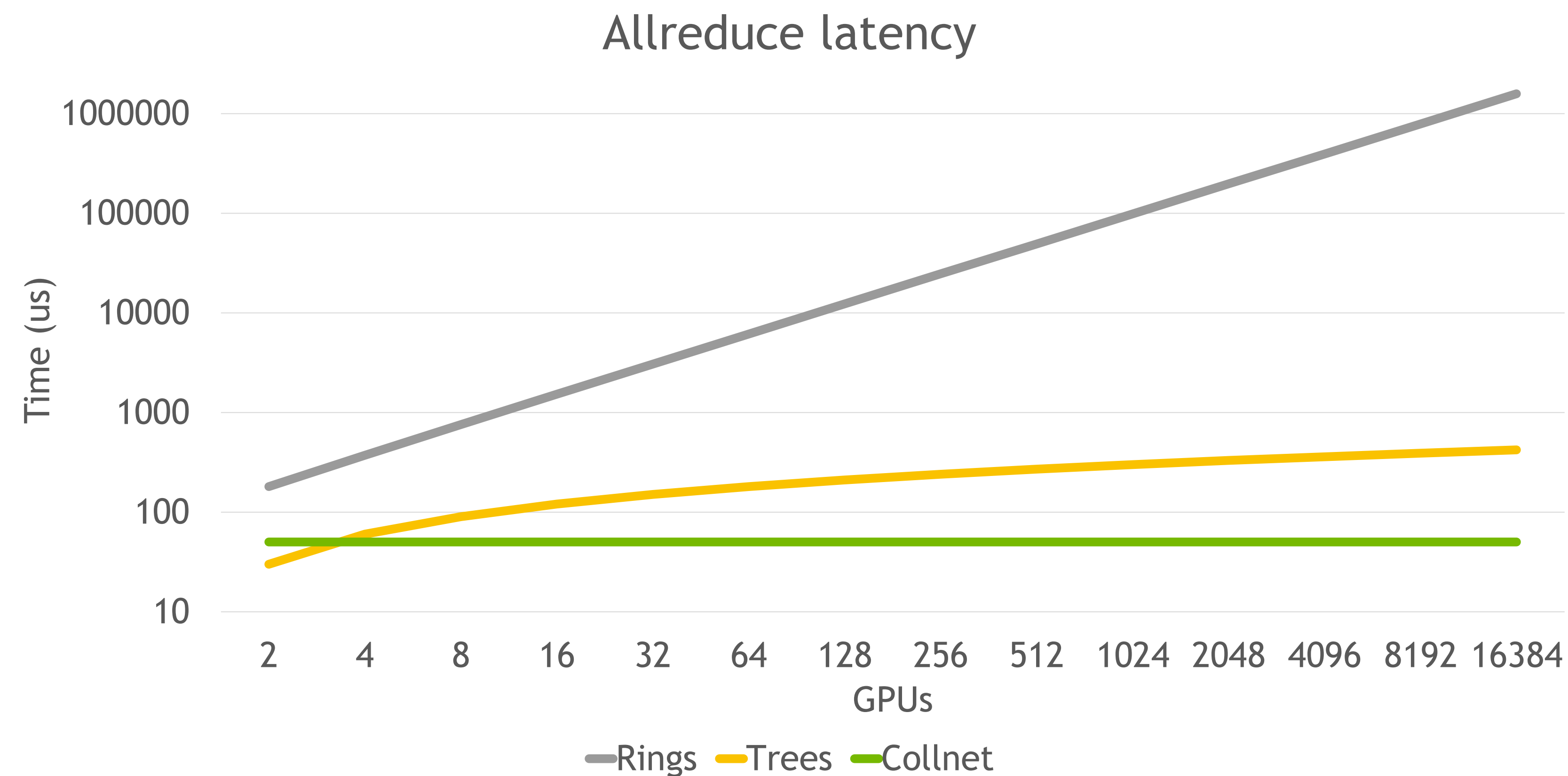
Direct version



ALGORITHMS SUMMARY

Pros and cons

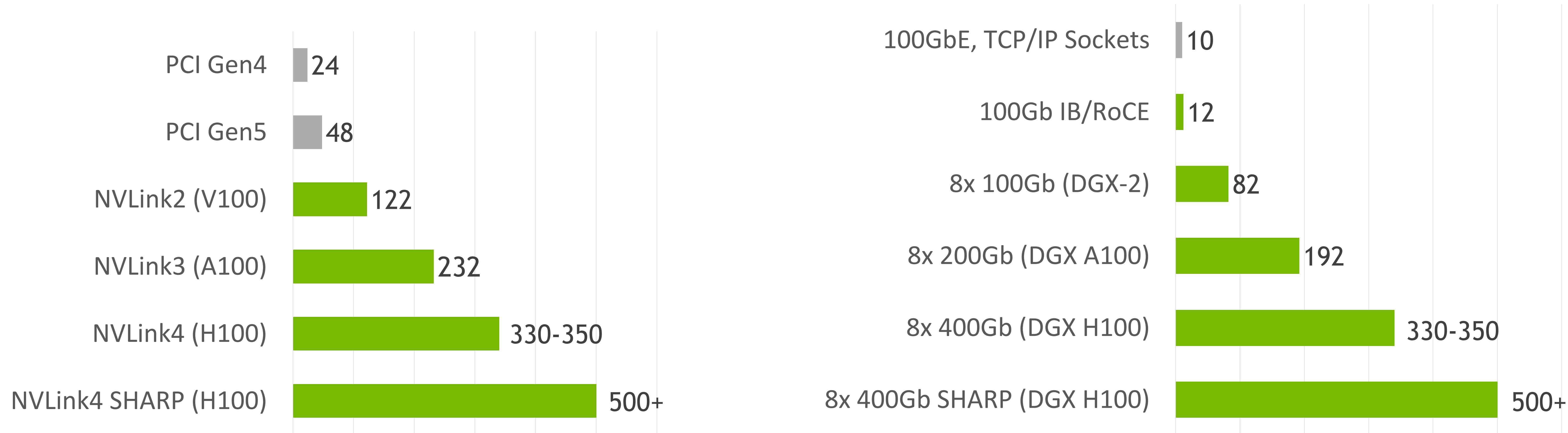
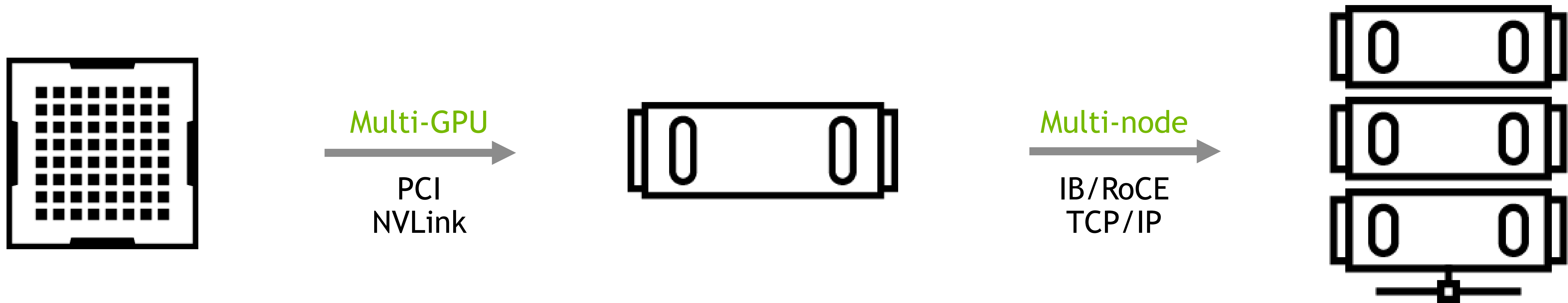
Algorithm	Latency	Bandwidth	Computing	Network Pattern
Rings	Linear	Perfect	Uniform	Few flows
Trees	Log	Close to perfect	Imbalanced	Many flows
Collnet	Constant	Close to 2x (may be limited by NVLink)	Almost uniform	Minimal flows





HARDWARE TOPOLOGY

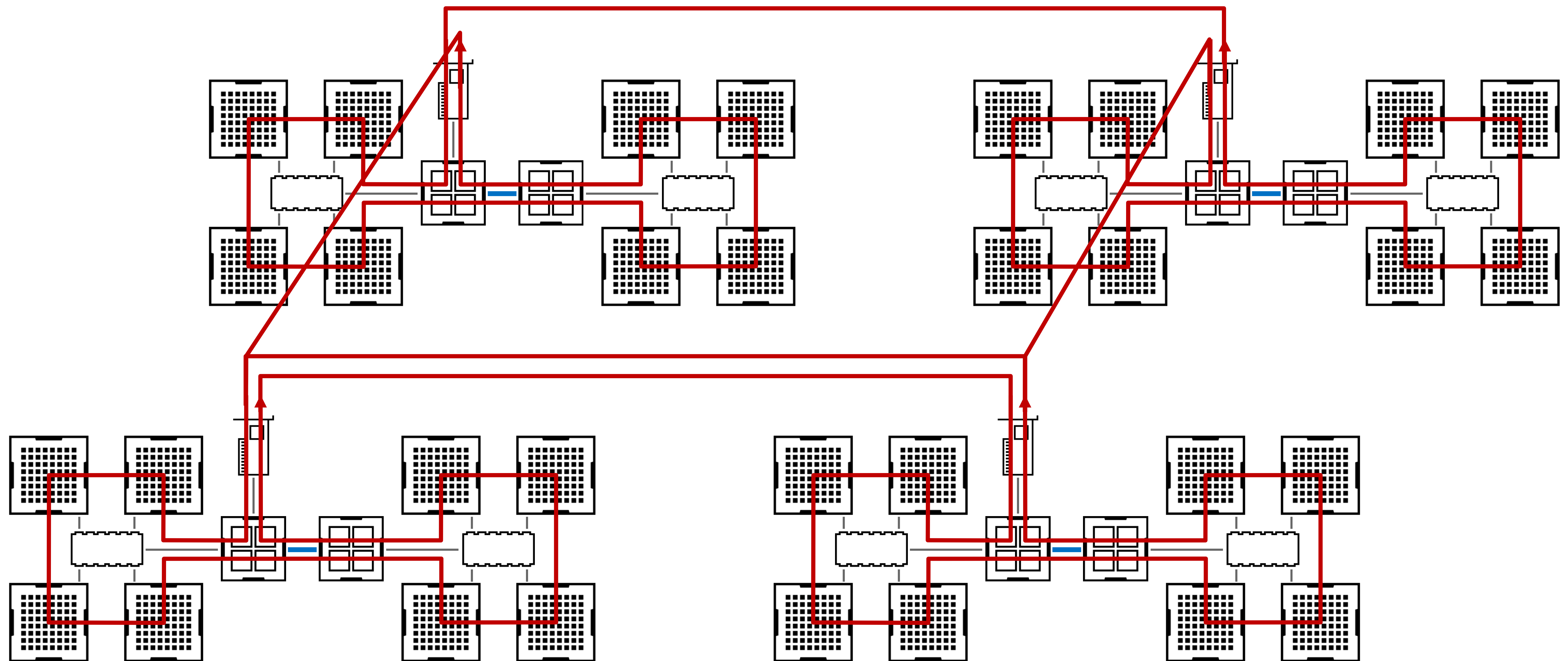
ALLREDUCE PERFORMANCE



NCCL tests Allreduce Bus Bandwidth in GB/s

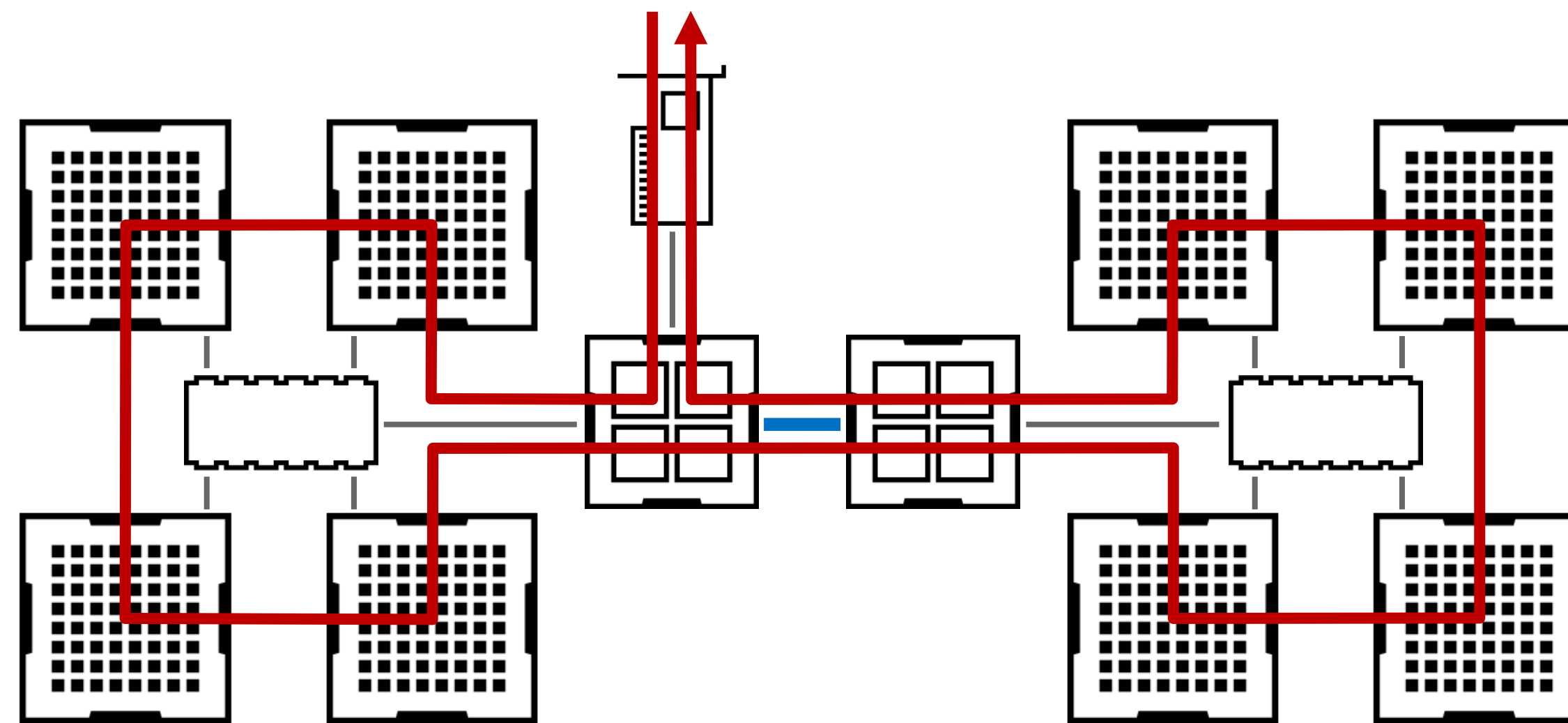
TOPOLOGY DETECTION

Mapping rings to the hardware

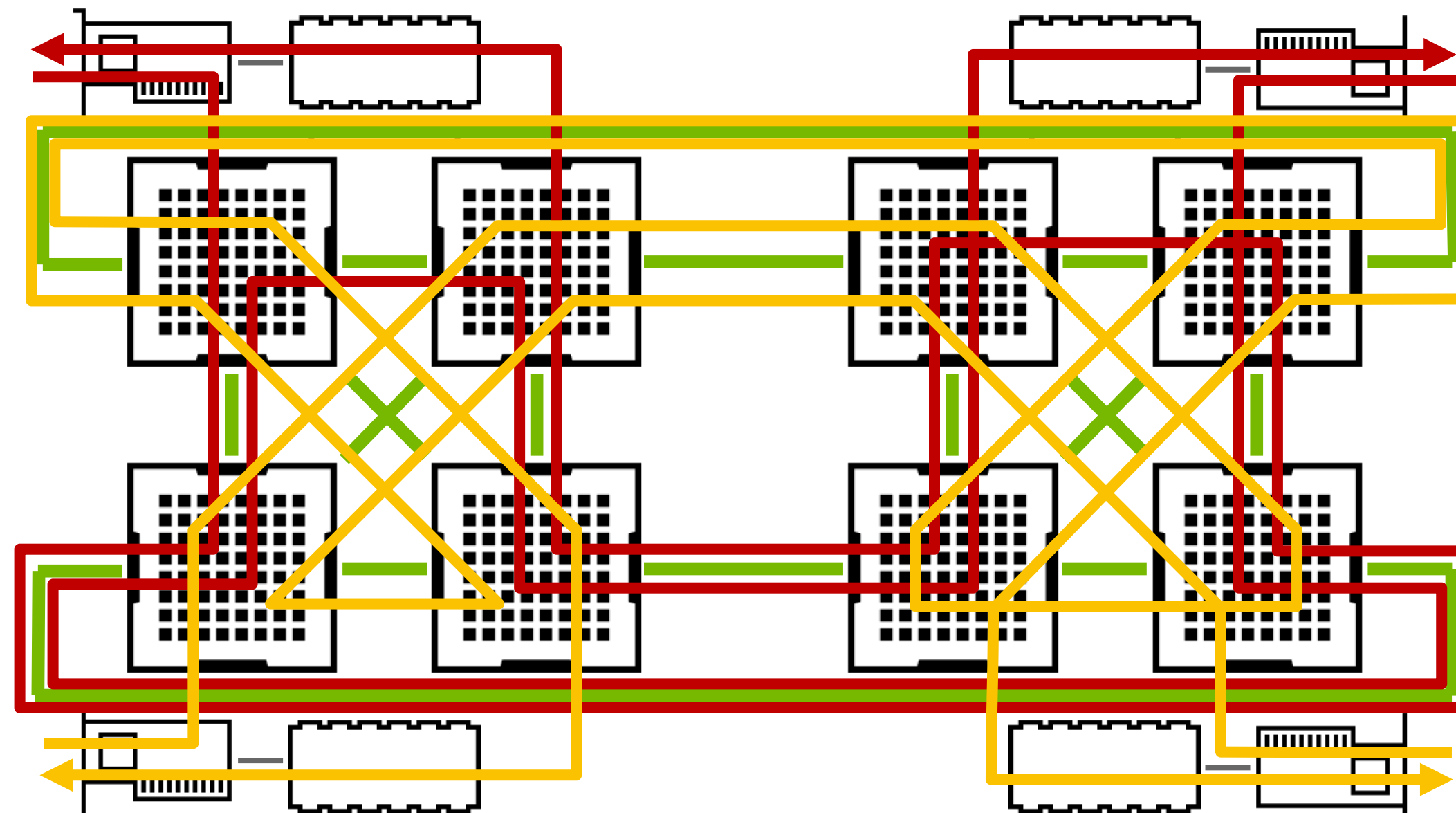


TOPOLOGY DETECTION

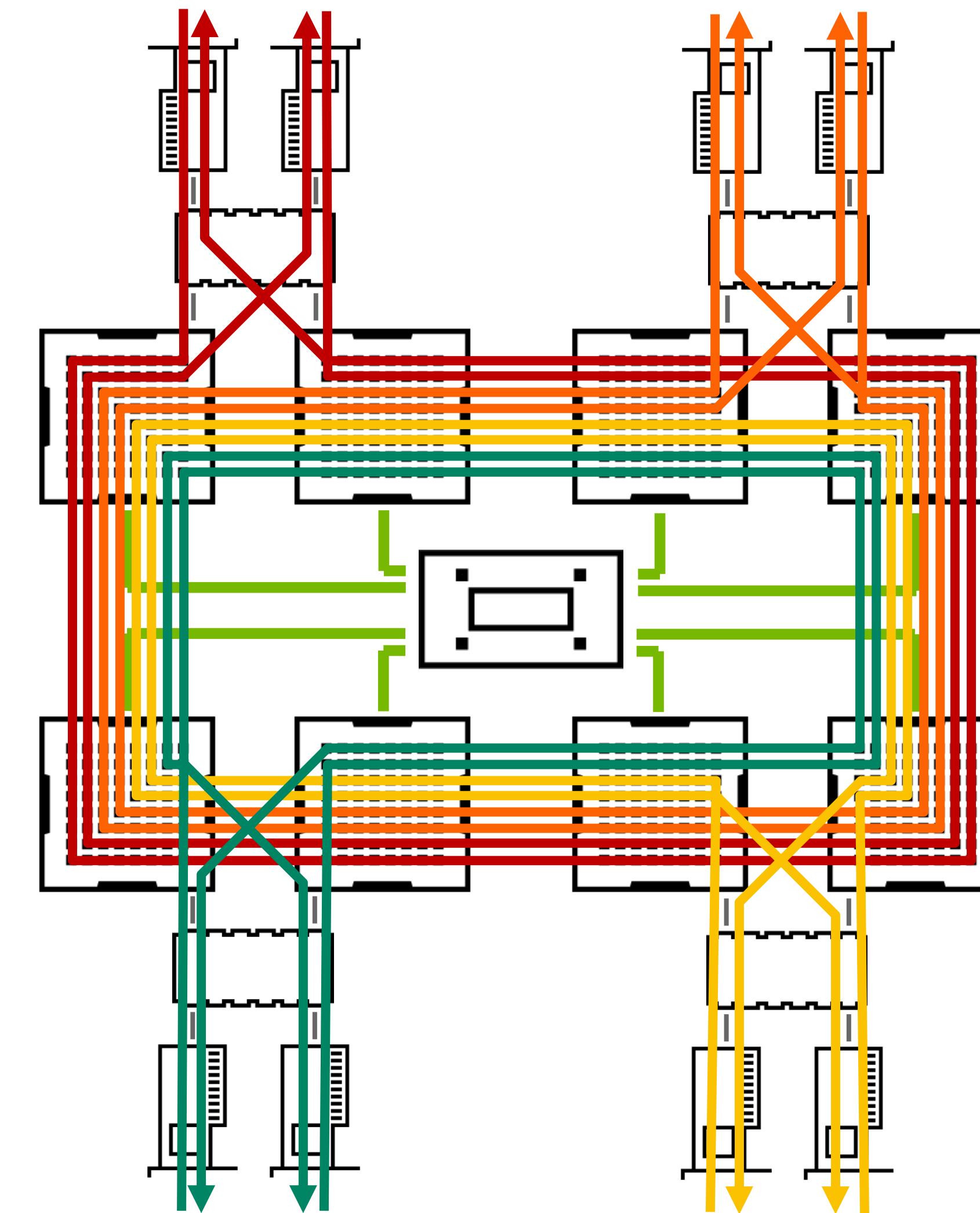
Mapping rings to the hardware



PCI platform



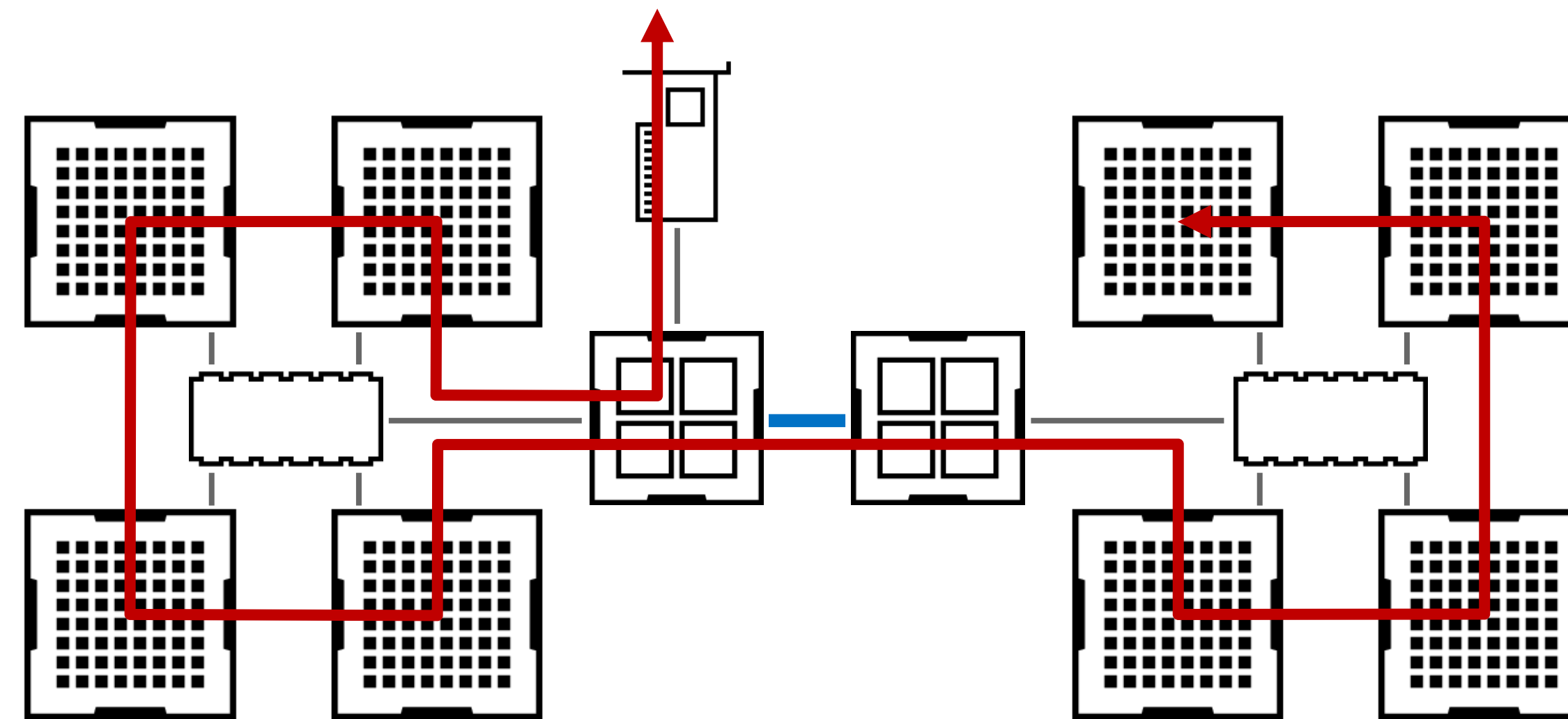
NVLink Cubemesh
(DGX-1)



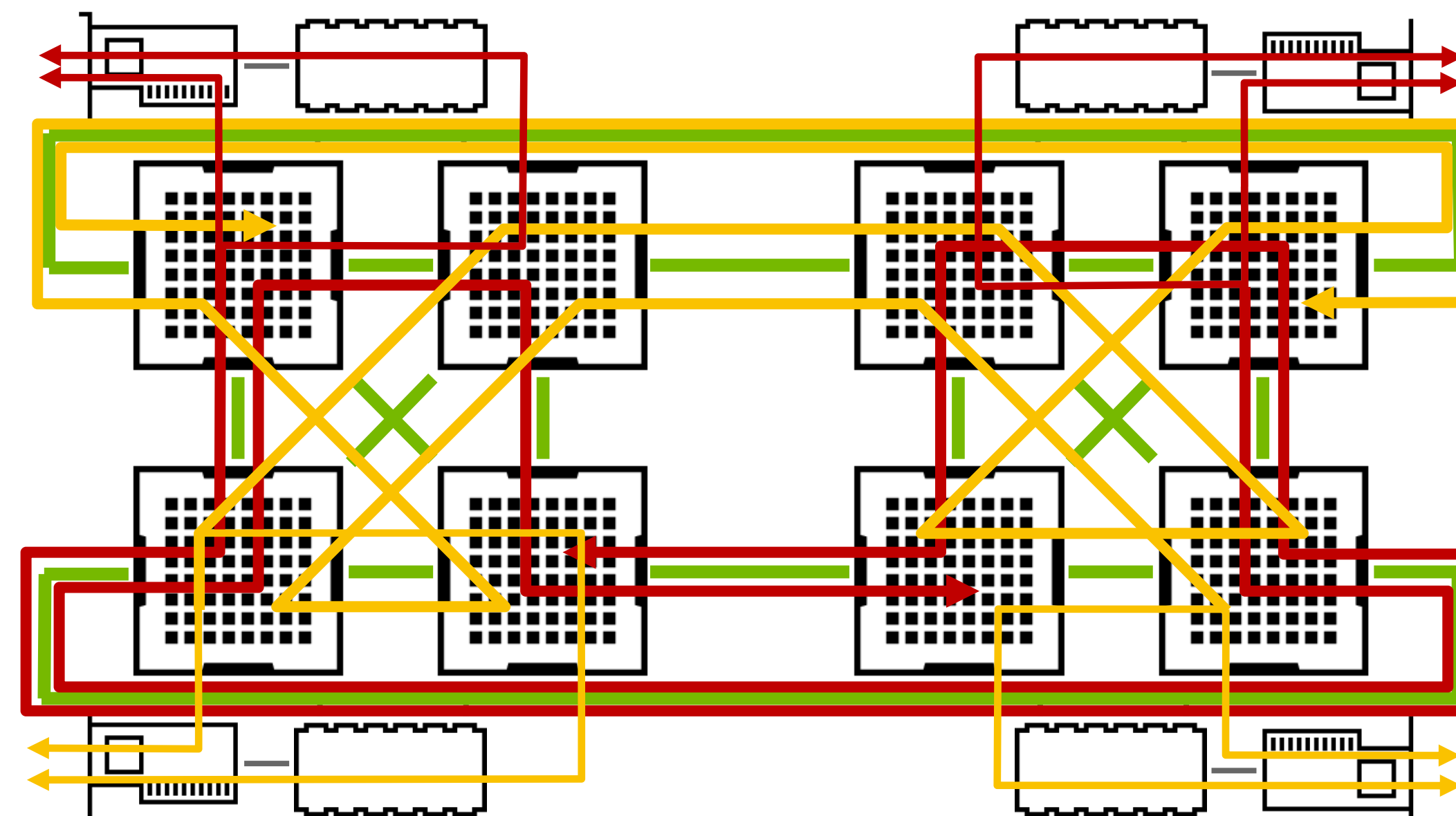
NVSwitch
(DGX A100)

TOPOLOGY DETECTION

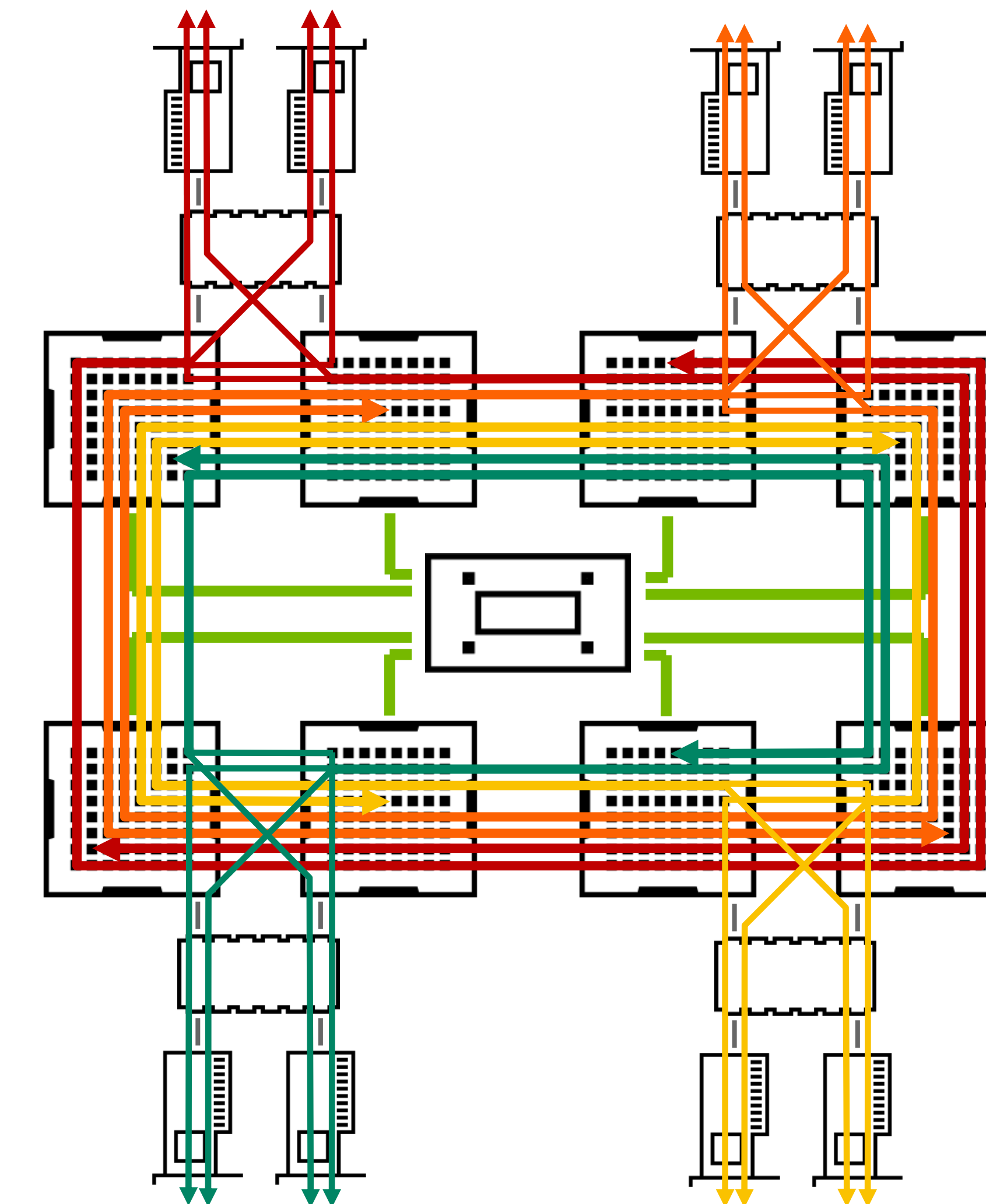
Mapping trees to the hardware



PCI platform



NVLink Cubemesh
(DGX-1)

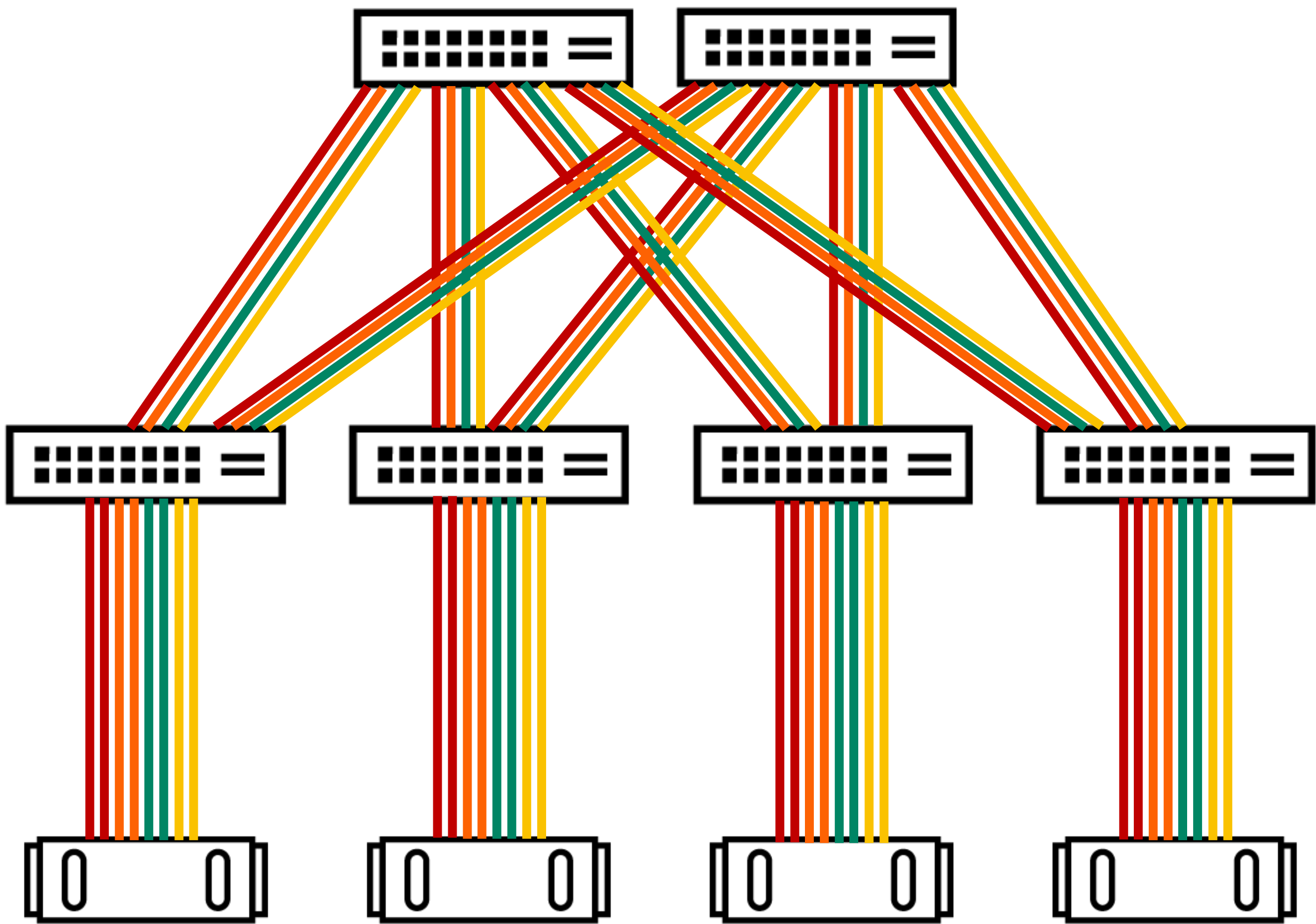


NVSwitch
(DGX A100)

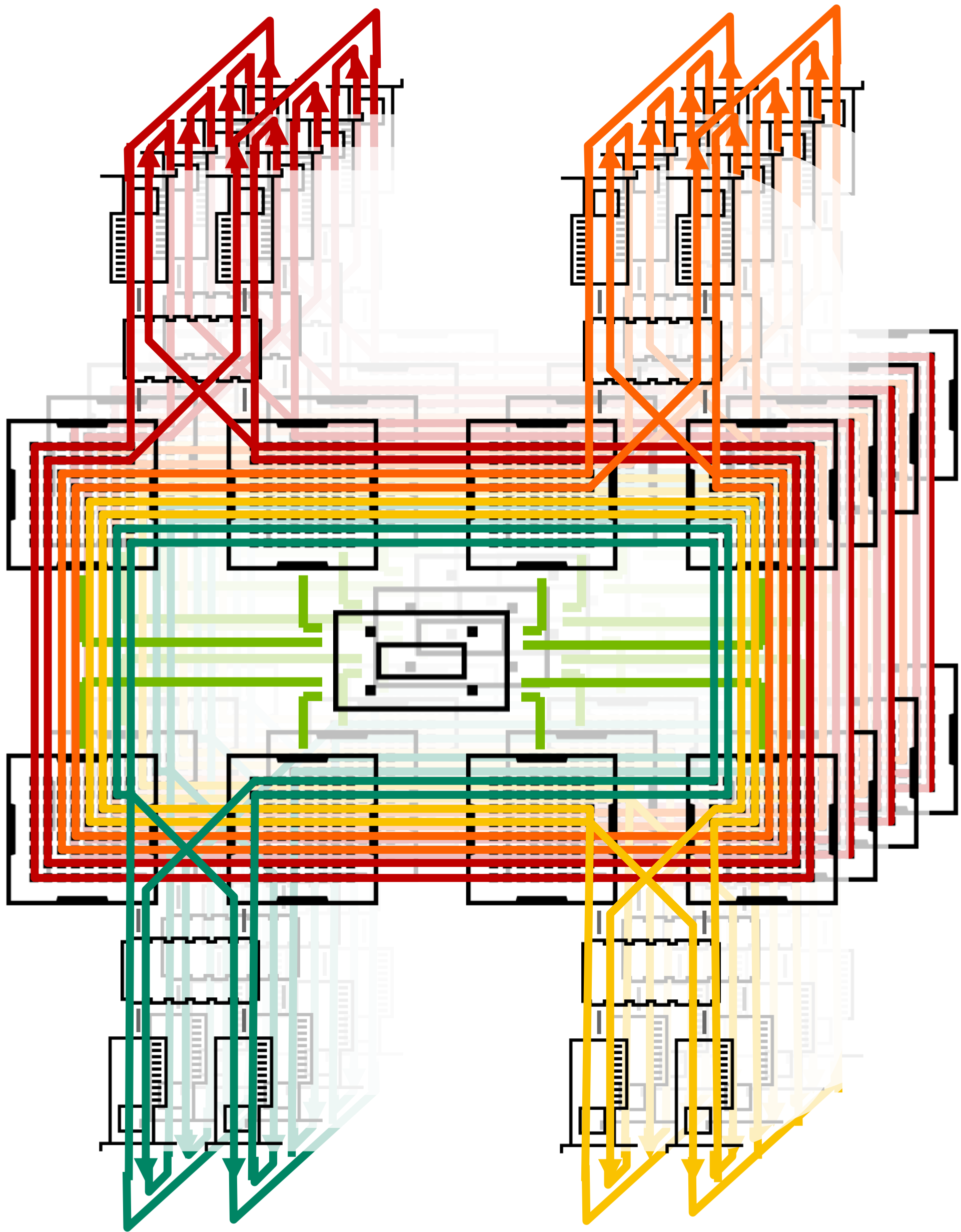
INTER-NODE COMMUNICATION

Rail-optimized design

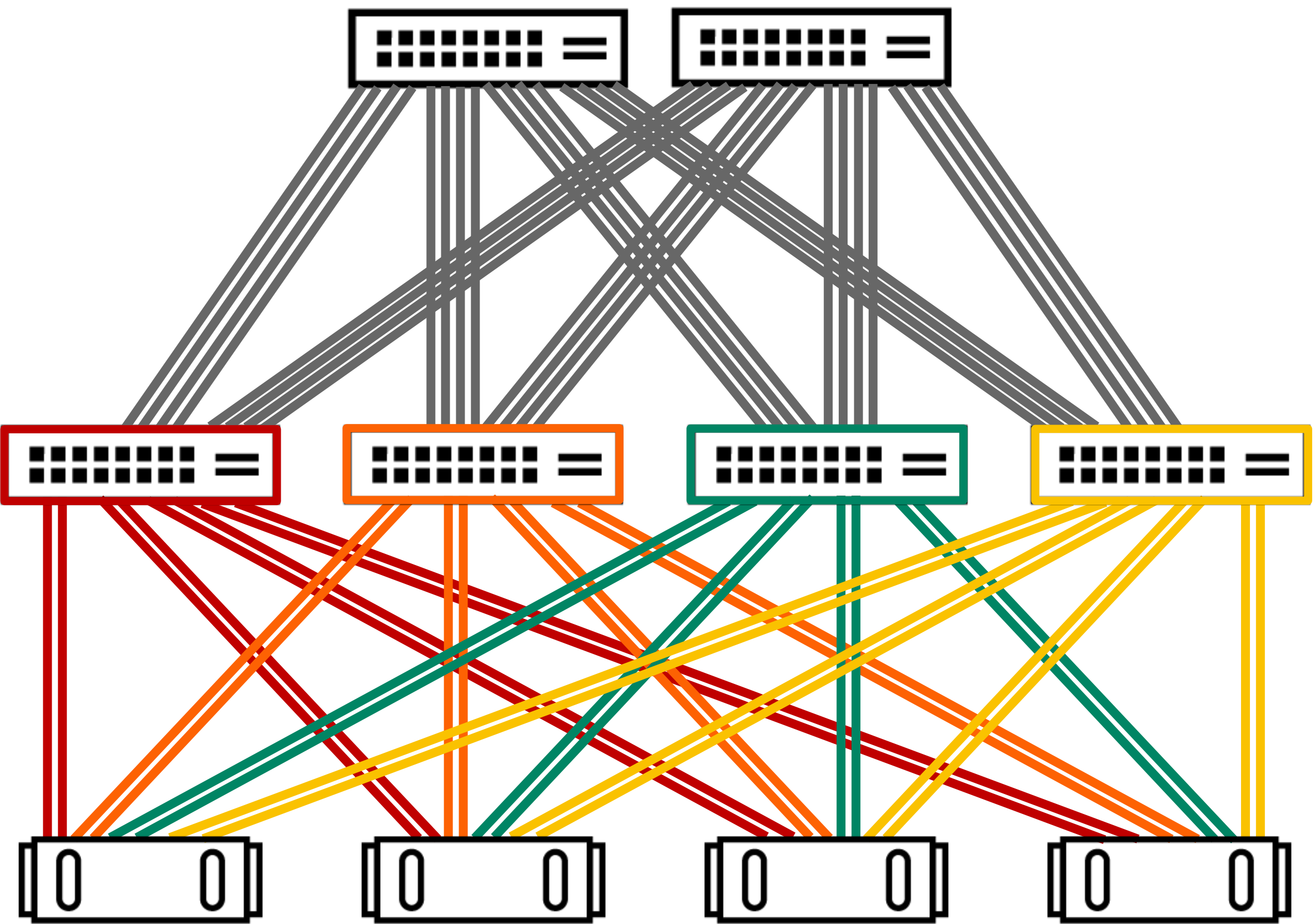
Routing must be perfect to ensure all flows use different links.



Classic fabric design



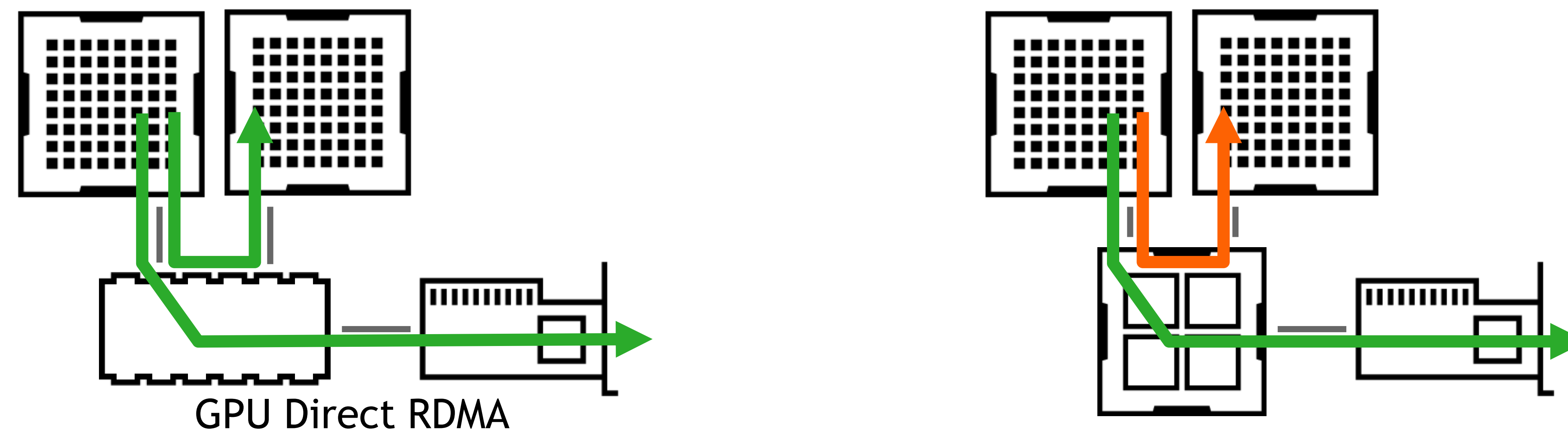
All traffic is local to leaf switches. Routing collisions are impossible.



Rail-optimized design

HARDWARE TOPOLOGY

Performance considerations



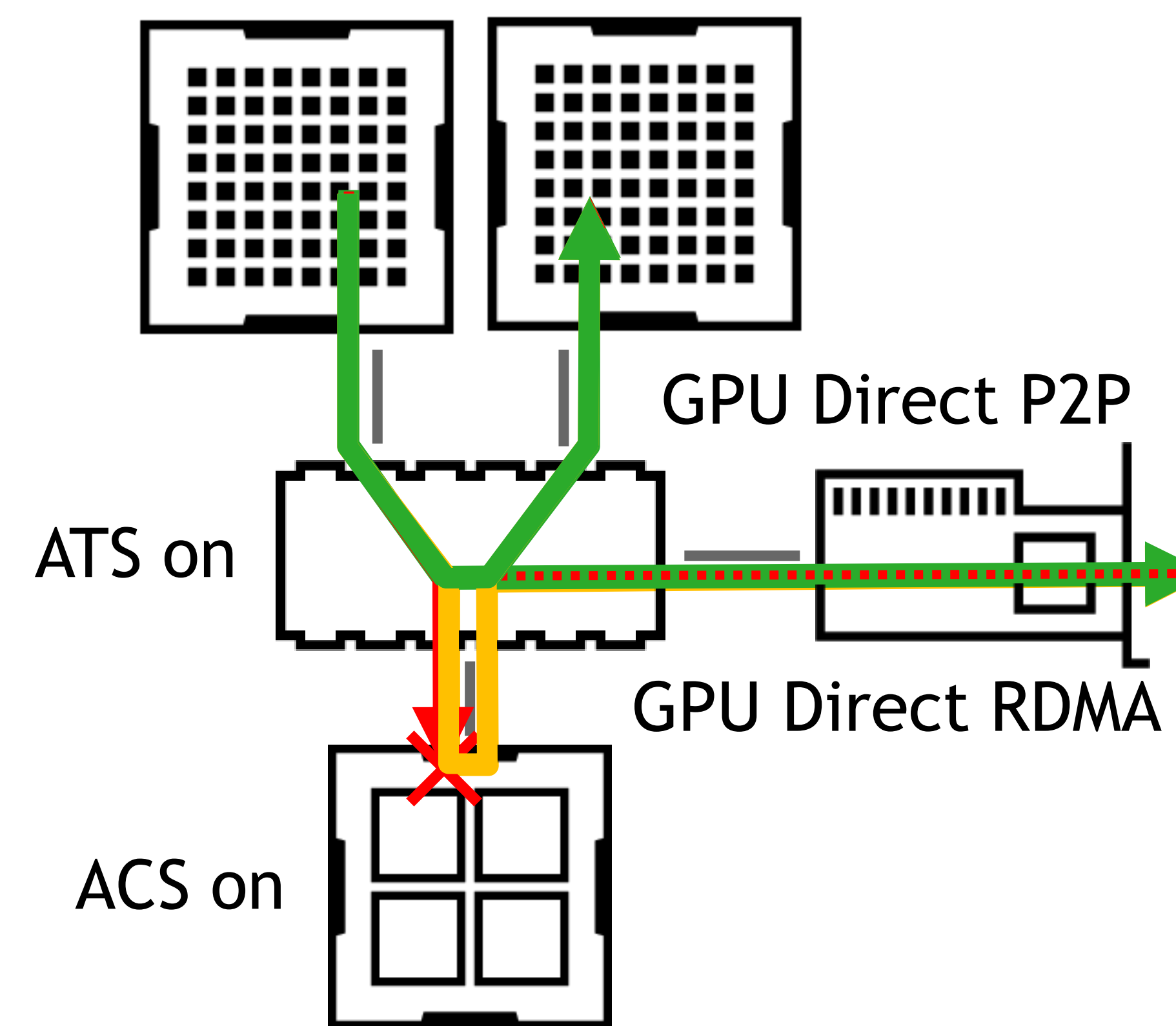
GPU to GPU: PCI switches get perfect performance, CPUs usually only get 50 to 80% of peak

GPU to NIC: GPU Direct RDMA is needed to go directly through the PCI switch

GPU DIRECT CONSIDERATIONS

ACS and ATS

ACS enabled without proper CPU-side configuration breaks GPU Direct. This is the default on most systems now due to BIOS enabling it and Linux not configuring ACS when no VM hypervisor is installed (e.g. KVM)



ACS forwards all PCI-to-PCI transactions to the CPU root complex for access control checks, halving (at least) the bidirectional bandwidth

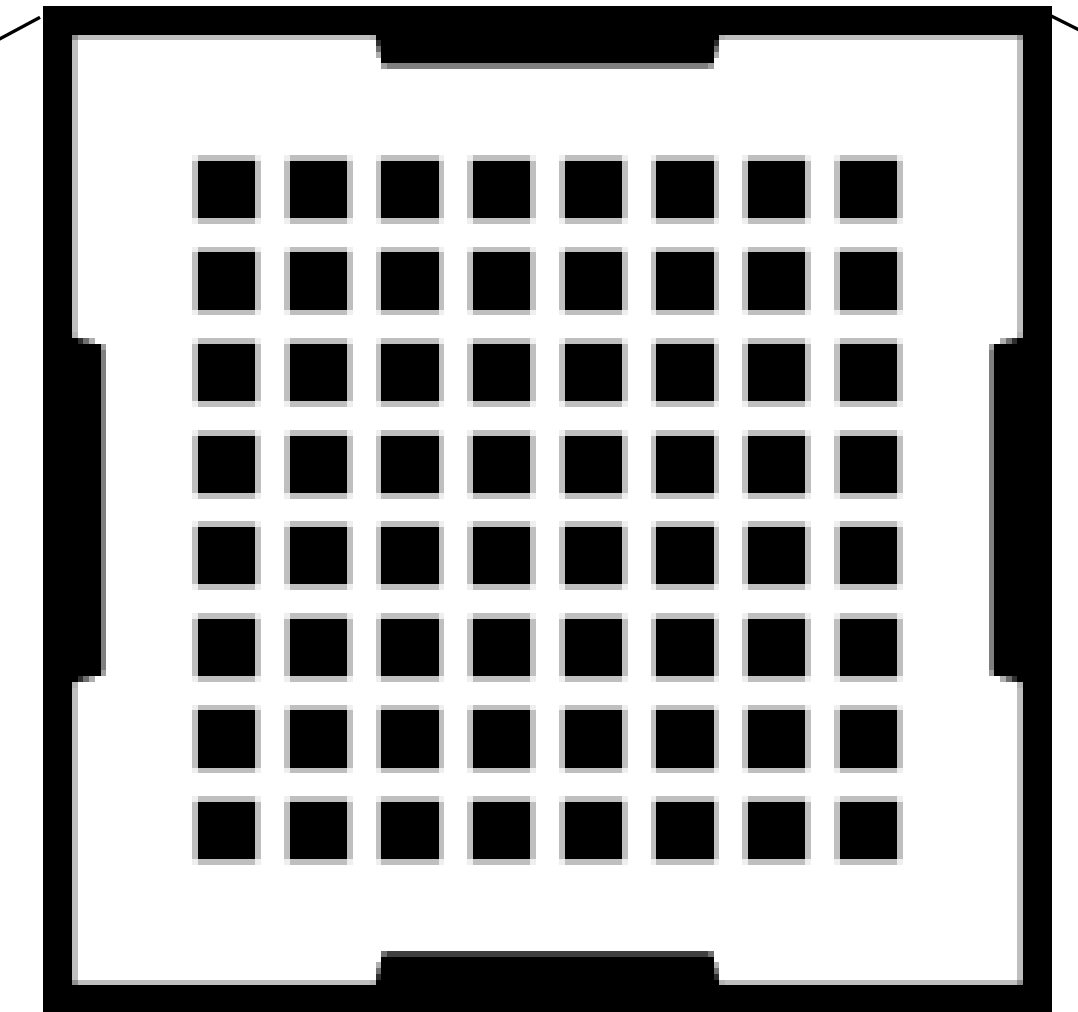
ATS caches ACS results to permit most traffic to go directly from GPU to NIC



POINT-TO-POINT COMMUNICATION

ALLTOALL COMMUNICATION

Example and implementation



NCCL usage:

```
ncclGroupStart();  
for (int i=0; i<n ranks; i++) {  
    ncclSend(sendbuffs[i], count, type, i, comm);  
    ncclRecv(recvbuffs[i], count, type, i, comm);  
}  
ncclGroupEnd();
```

With 8 p2p channels (=SMs/CTAs) 64 peers are handled in parallel.

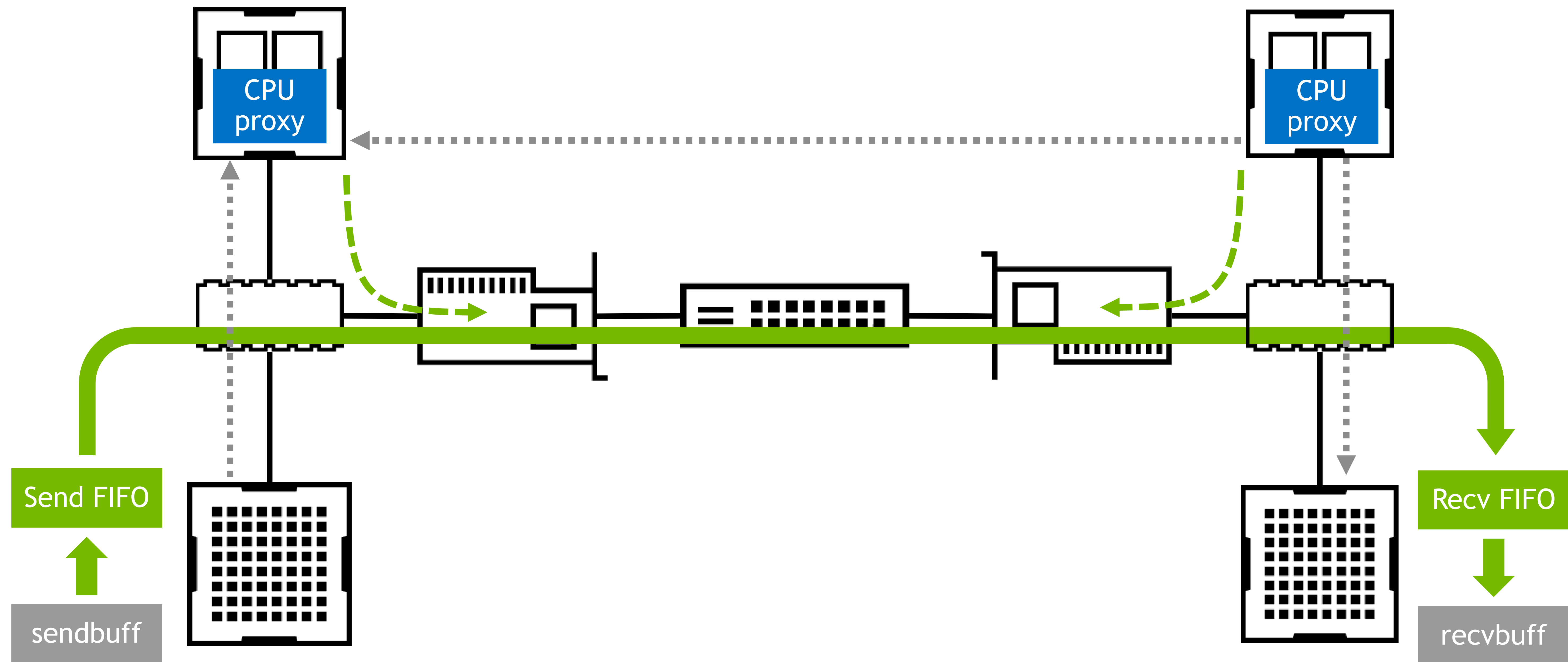
8 warps per SM/CTA send to 8 different peers

8 warps per SM/CTA recv from 8 different peers

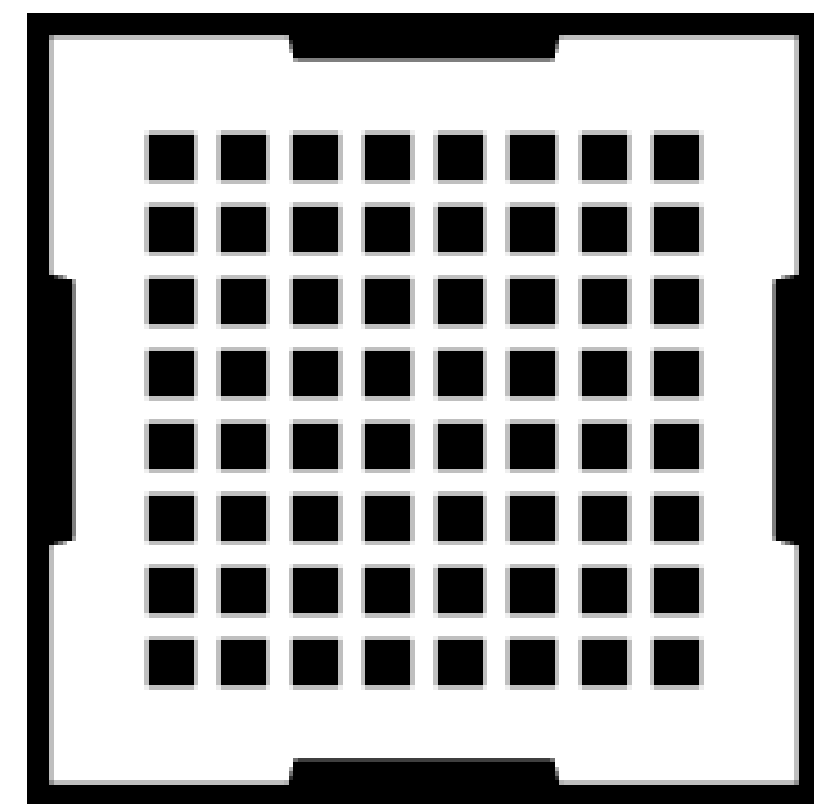


POINT-TO-POINT COMMUNICATION

Network proxy



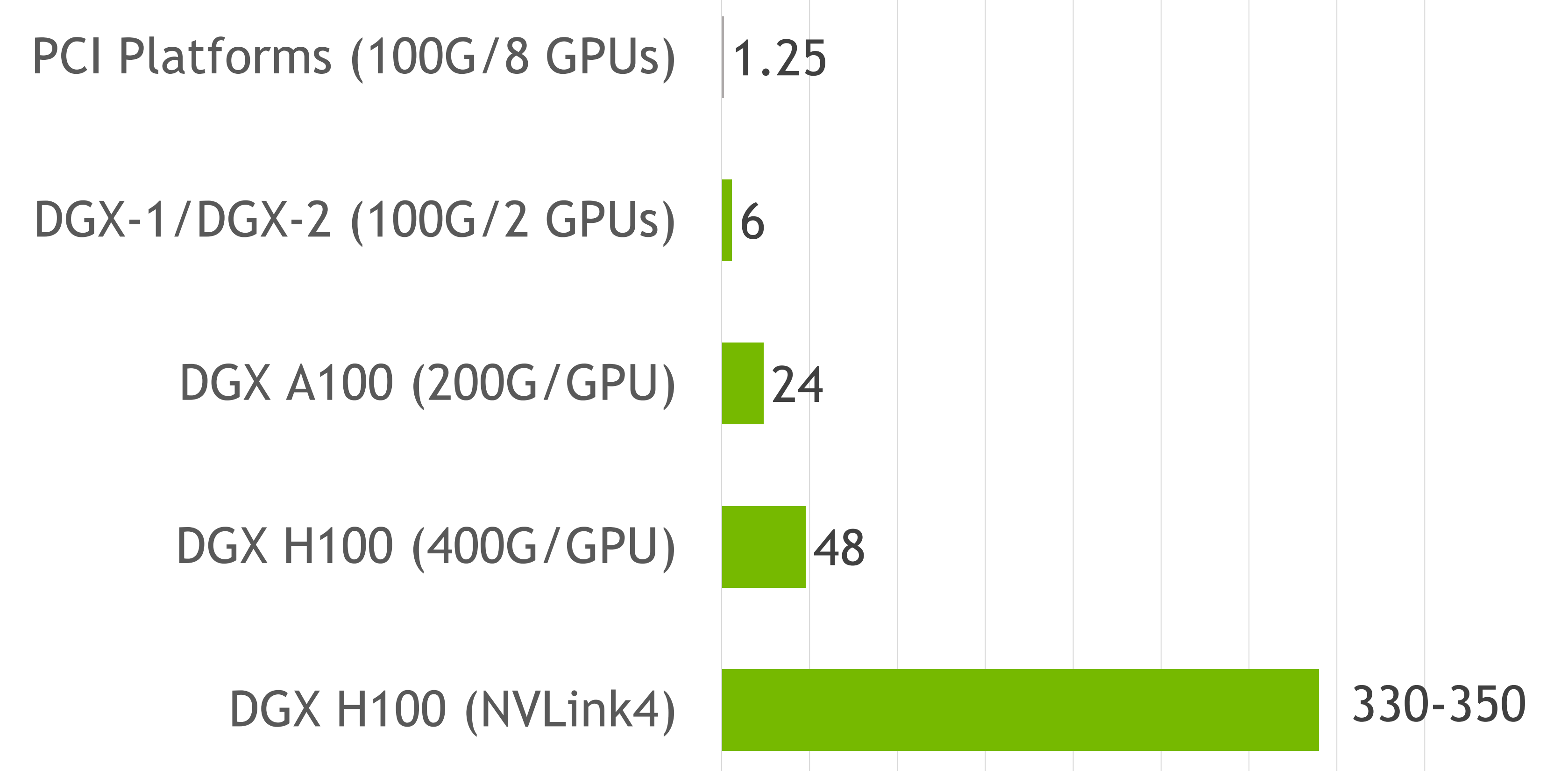
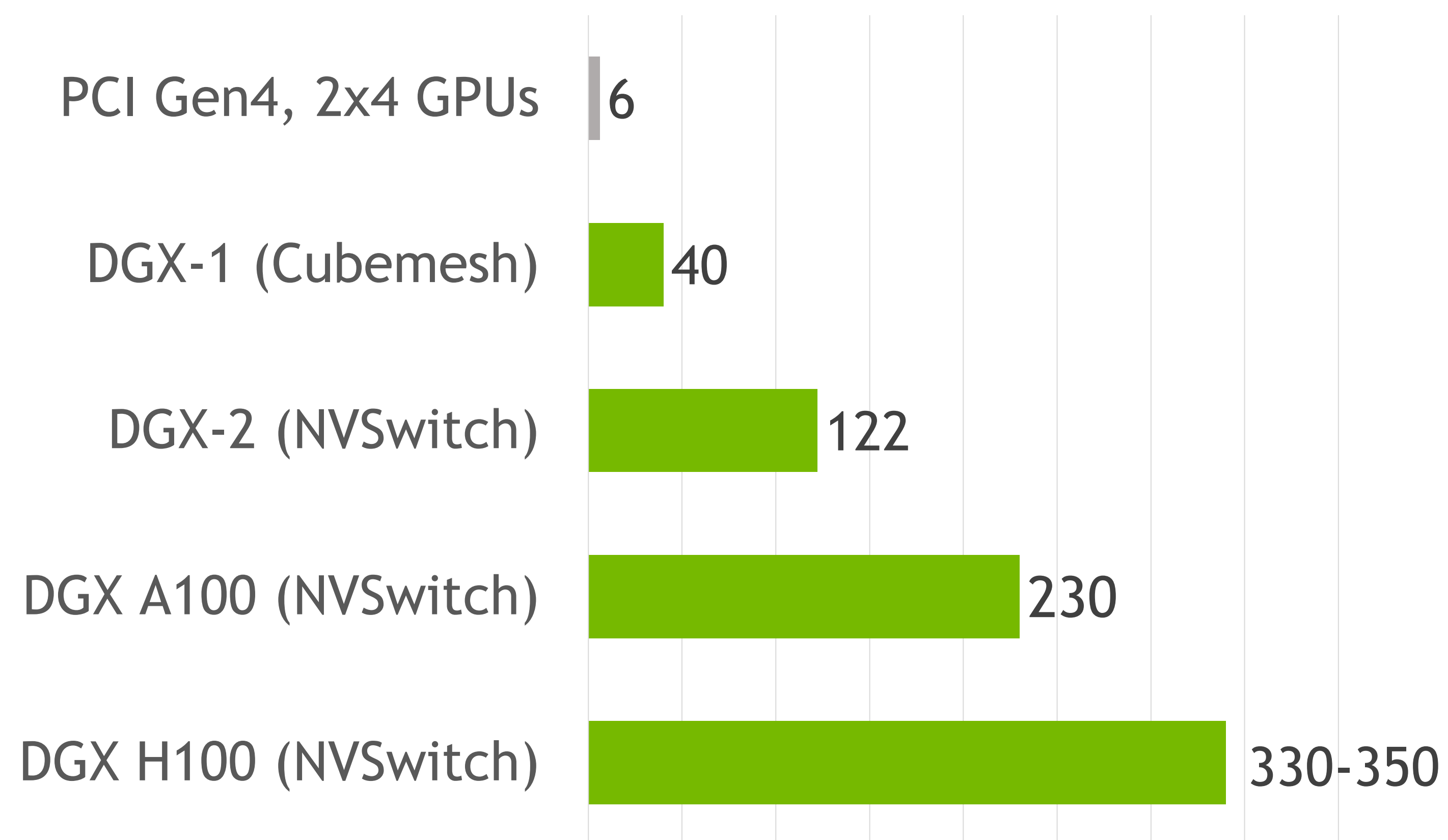
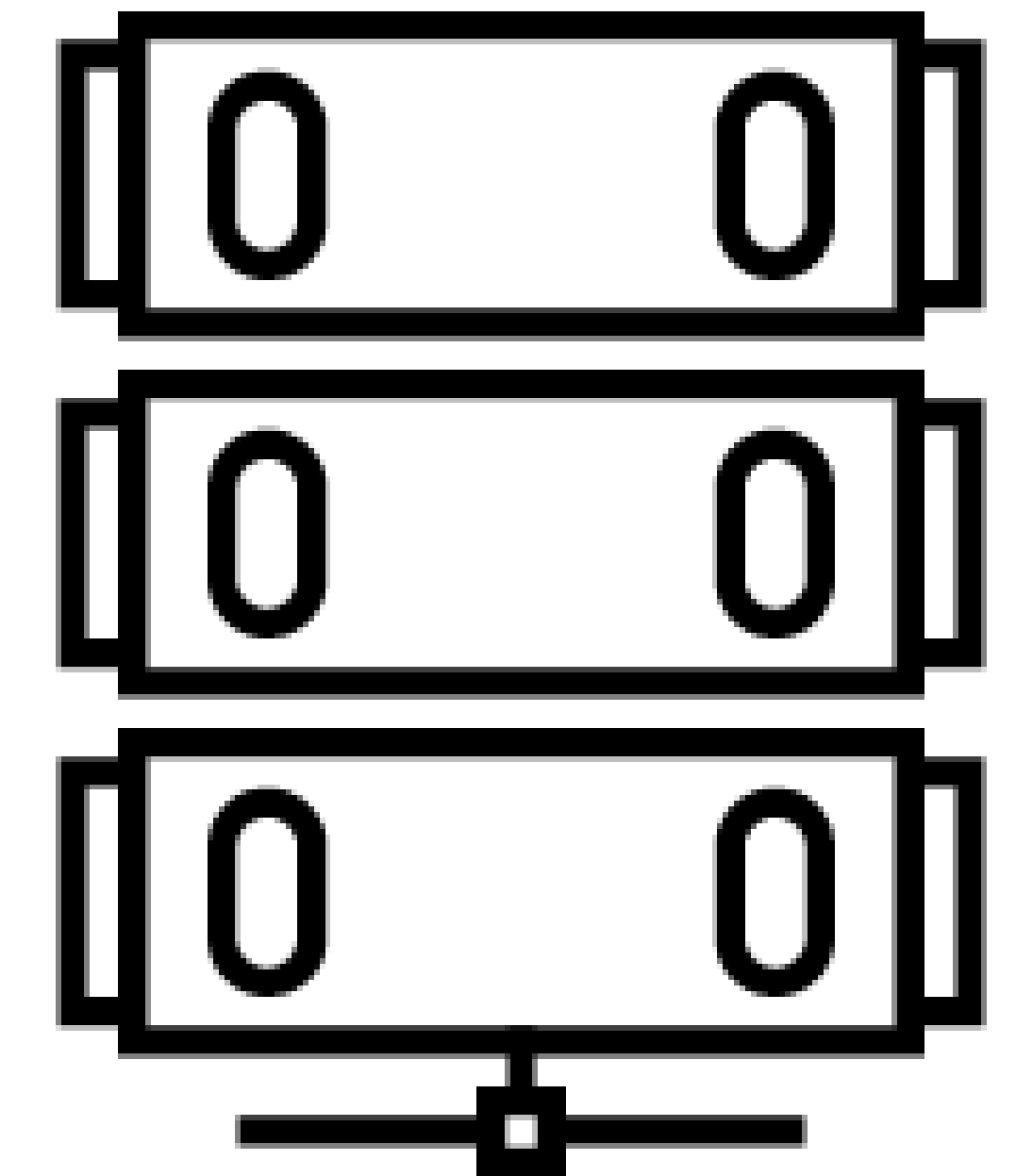
ALLTOALL BANDWIDTH



Multi-GPU
→
PCI
NVLink



Multi-node
→
TCP/IP
IB/RoCE
NVLink



Effective bandwidth in GB/s

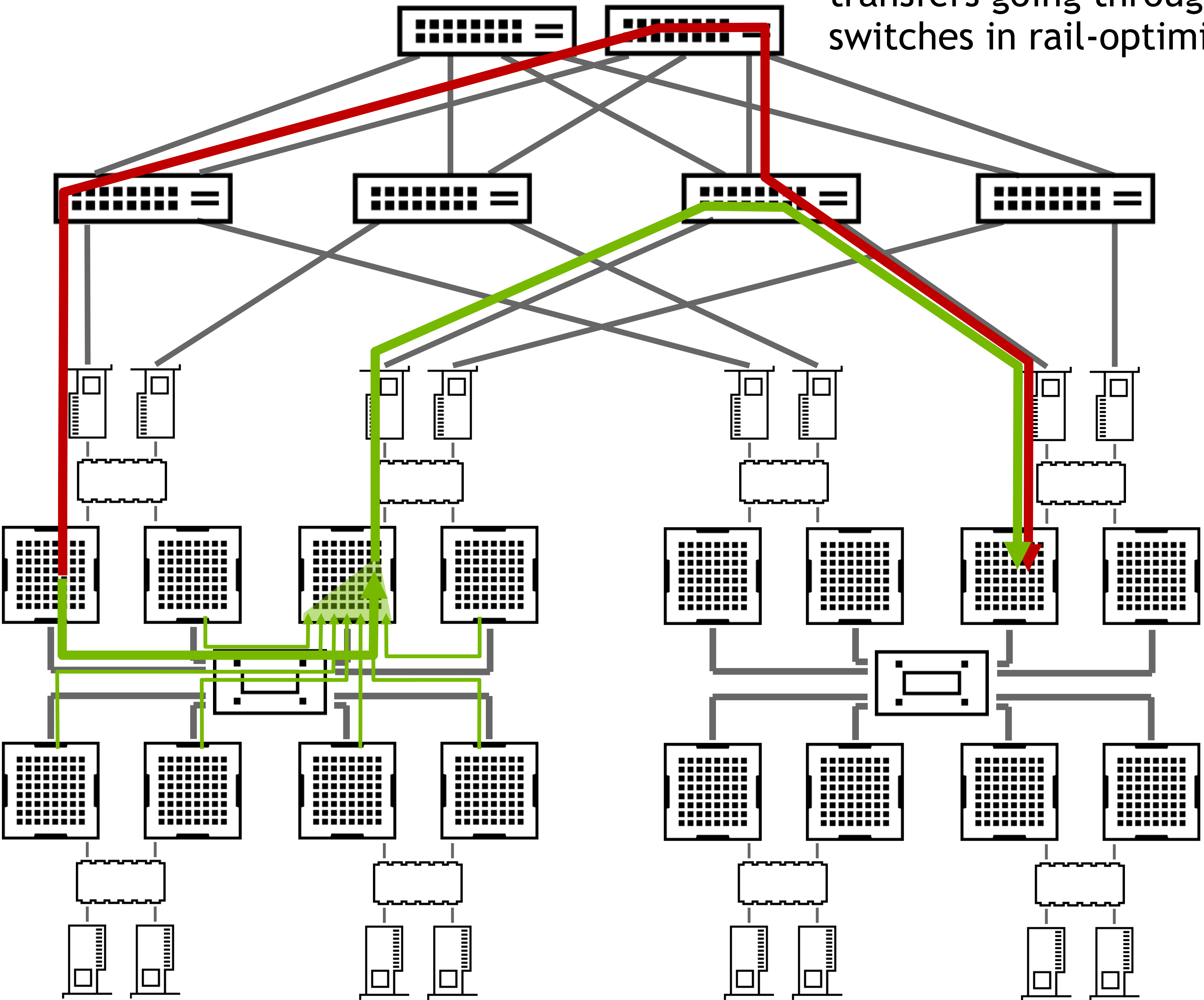
ALLTOALL COMMUNICATION

PXN optimization

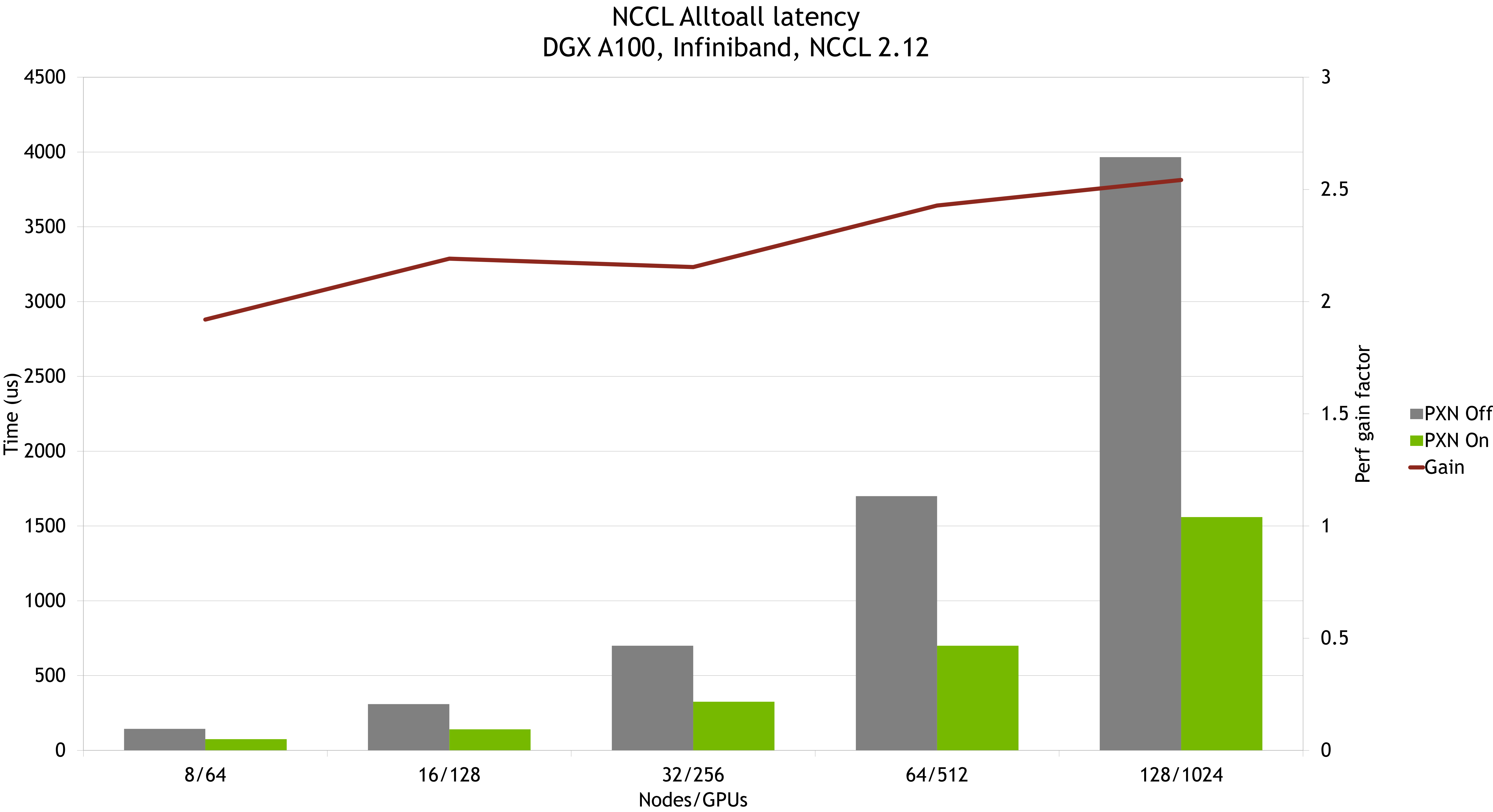
Alltoall communication has most transfers going through top-level switches in rail-optimized designs.

PXN also allows for the aggregation of all messages to the same destination.

PXN (PCI x NVLink) allows GPUs to send data using a distant NIC, using an intermediate GPU.



ALLTOALL LATENCY





INIT / BOOTSTRAP

NCCL INIT

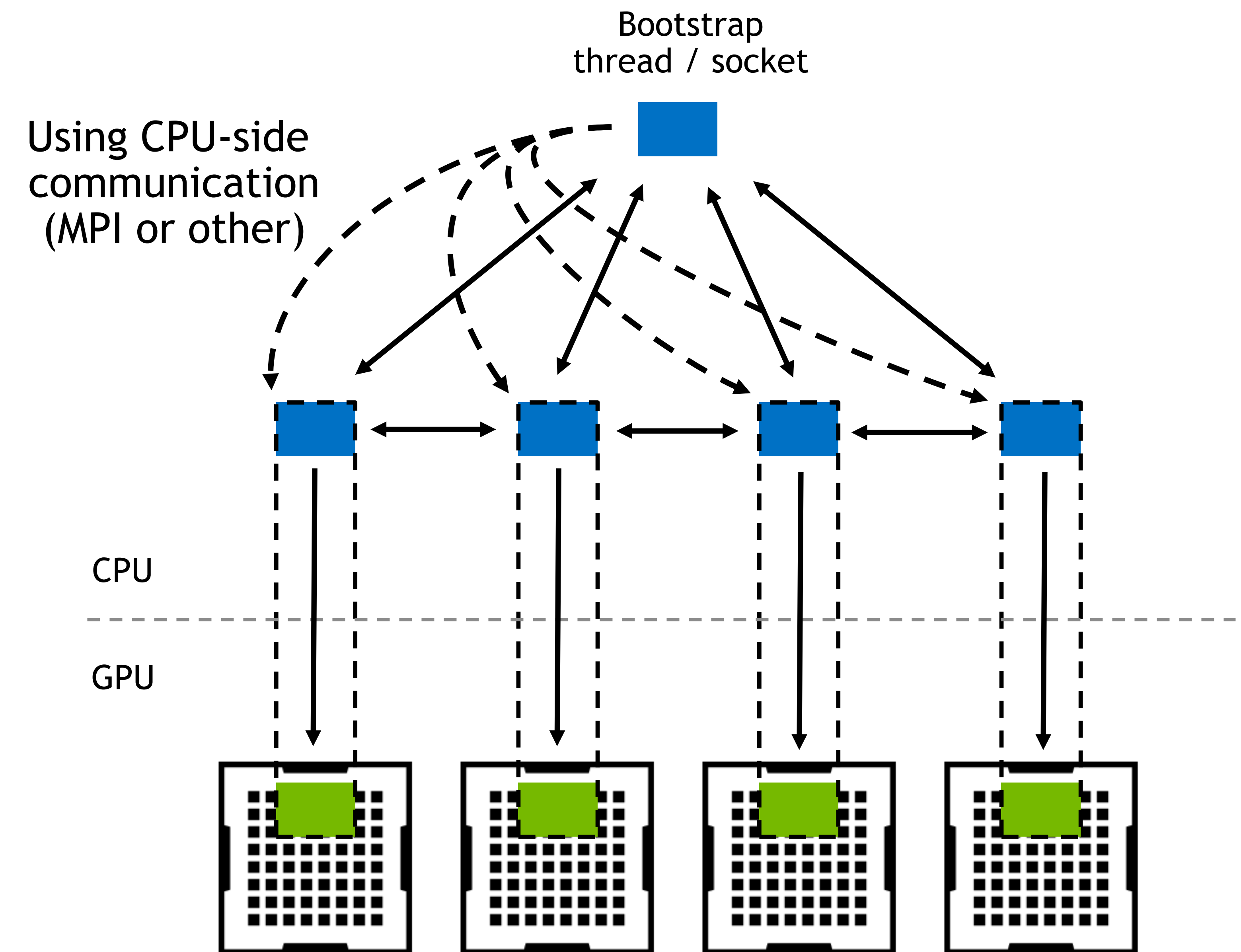
Bootstrap and operation

Once (typically on worker0):

```
ncclUniqueId id;  
ncclGetUniqueId(&id);  
broadcast_to_all_ranks(&id);
```

On all parallel workers:

```
// Initialization  
get_unique_id(&id);  
ncclCommInitRank(&comm, nranks, &id, rank);  
  
// Communication  
ncclAllReduce(..., comm);  
  
// Clean-up  
ncclCommDestroy(comm);
```



NCCL BOOTSTRAP

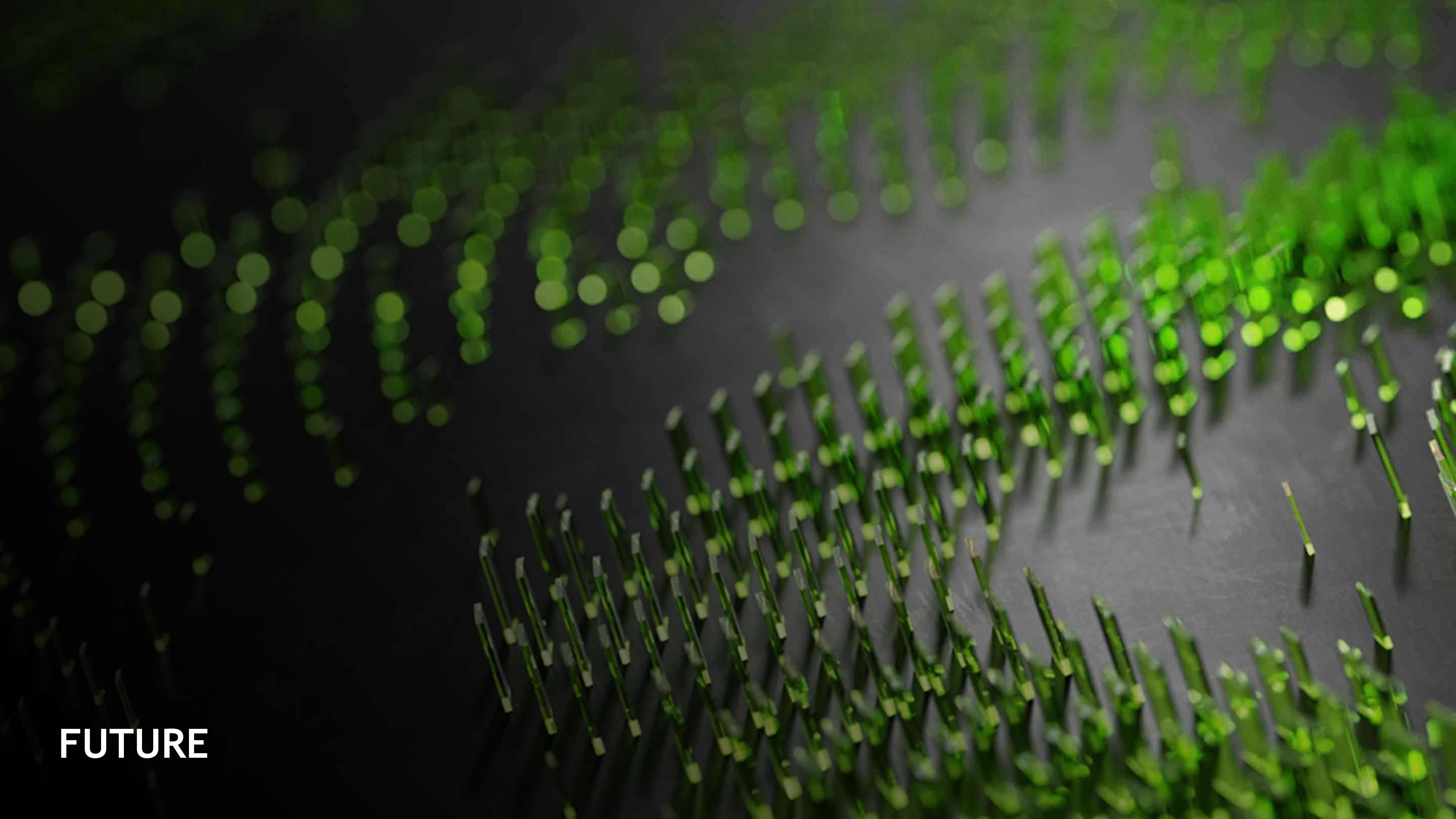
Network considerations

Bootstrap happens through simple, **unencrypted TCP/IP sockets**.

Each rank needs to be able to communicate through TCP/IP to each other rank. This may require **firewall configuration** or setting interfaces as “bridges”.

NCCL_SOCKET_IFNAME can be used to select which interface to use for bootstrap.

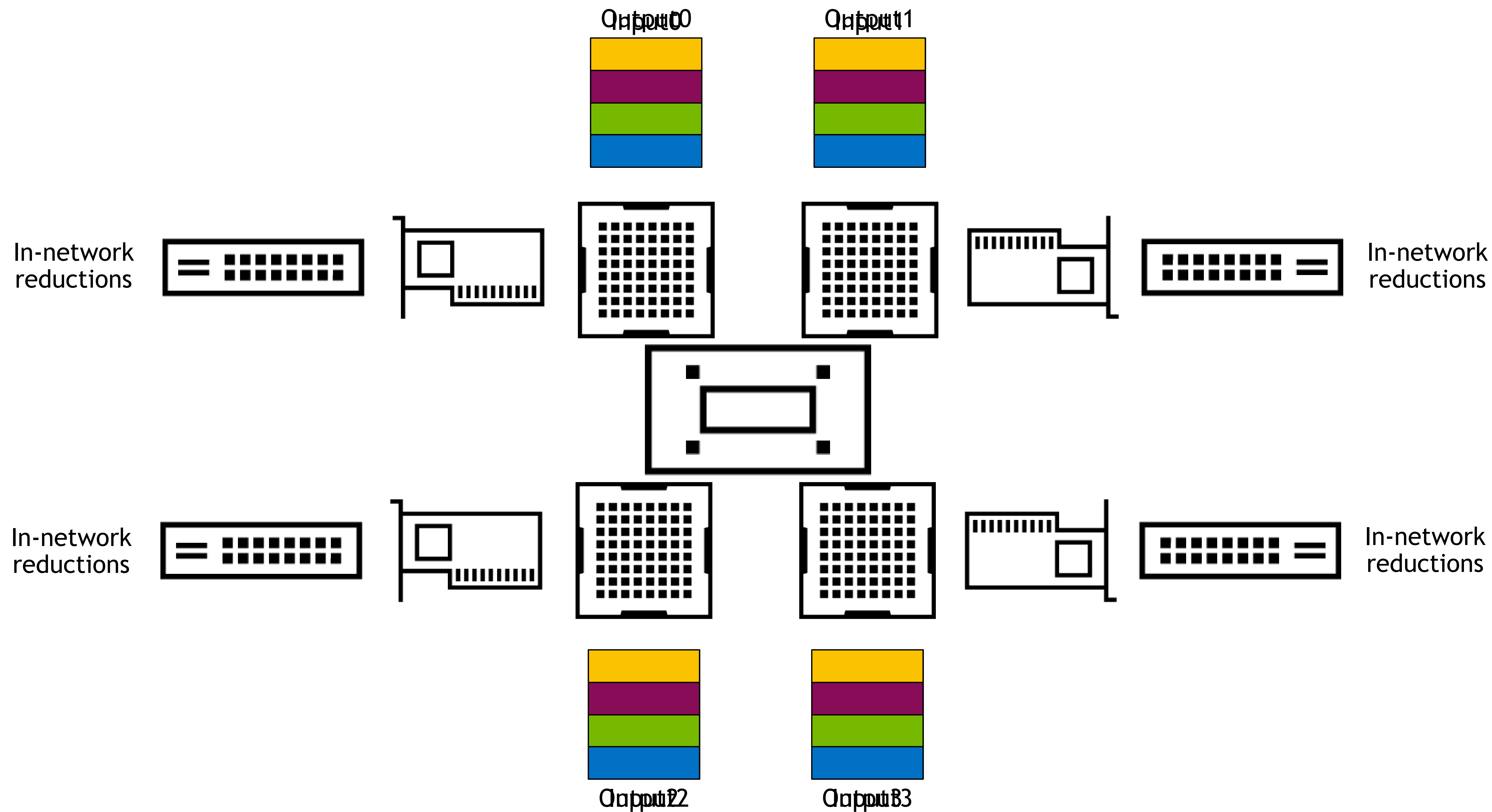
If security is a concern, NCCL_SOCKET_IFNAME should point to an interface which is on a **private network** or VLAN.



FUTURE

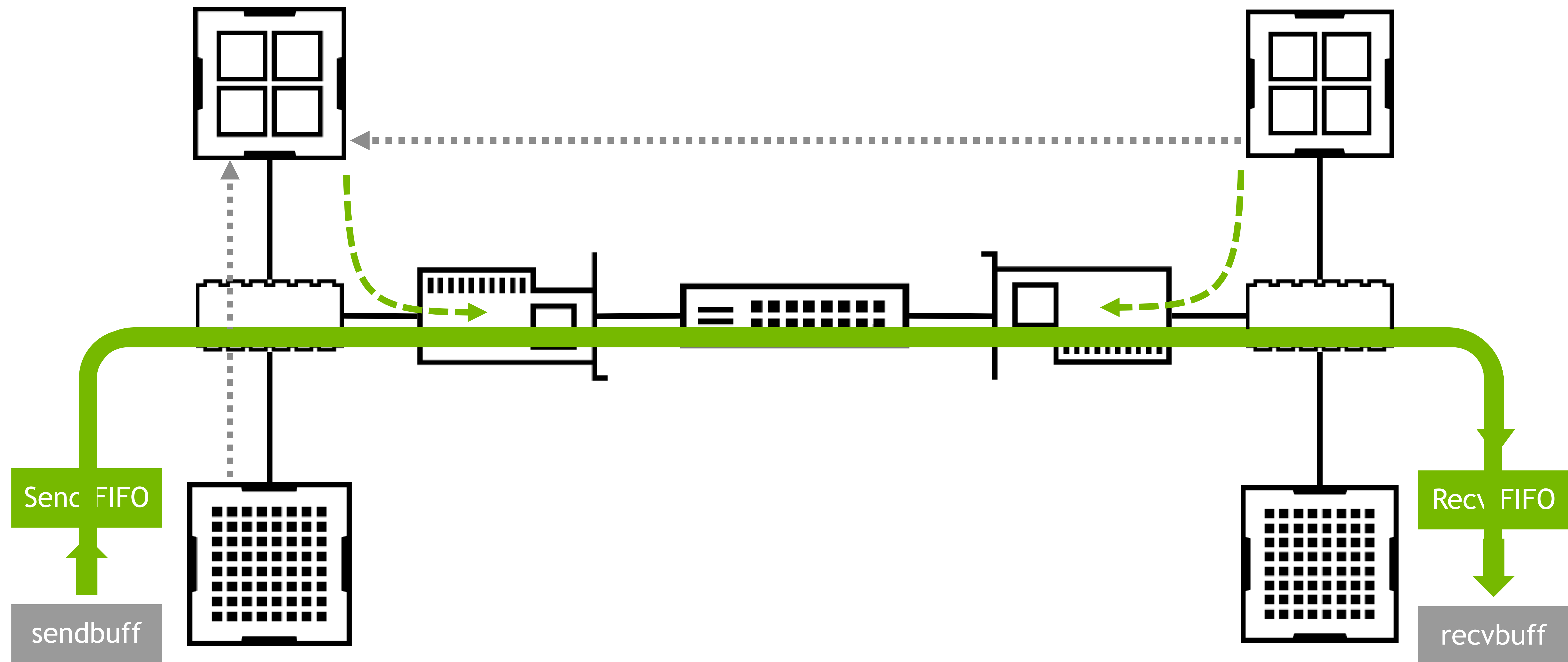
H100 SHARP

In-nvswitch and in-network



USER BUFFER REGISTRATION

Improved alltoall latency



FUTURE

Other improvements

Continued optimization of allreduce and alltoall

More allreduce algorithms (direct+tree/ring, collnet+chain, ...)

More efficient CUDA graph support

More efficient aggregation

Improved point-to-point performance on non-alltoall cases

Improved usability

Better integration with profilers

Better error reporting

Improved fault tolerance

Ability to abort init/destroy operations



SUMMARY

SUMMARY

NCCL : Key communication library for multi-GPU computing.

Optimized for all platforms, from desktop to DGX Superpod.

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



