Generalized Hierarchical Communication

Mert Hidayetoglu

Postdoctoral Scholar

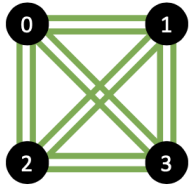
merth@stanford.edu

November 1, 2023

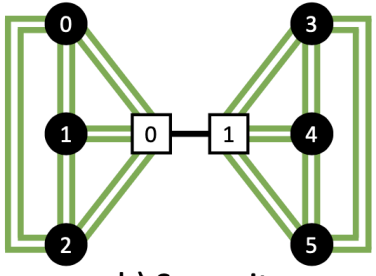
Stanford University

Networks are diverse!

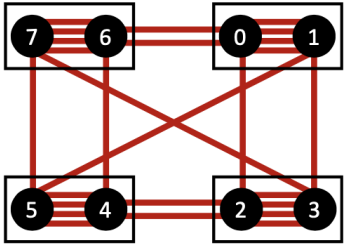
Intra-Node Subnetwork



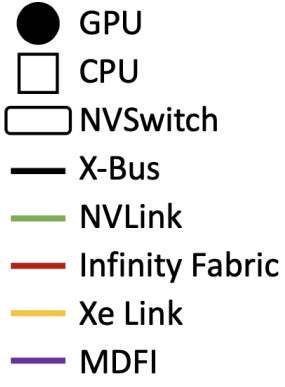
a) Delta, Perlmutter

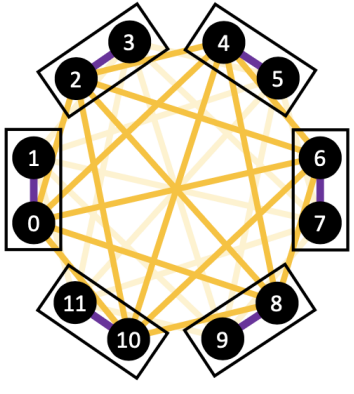


b) Summit

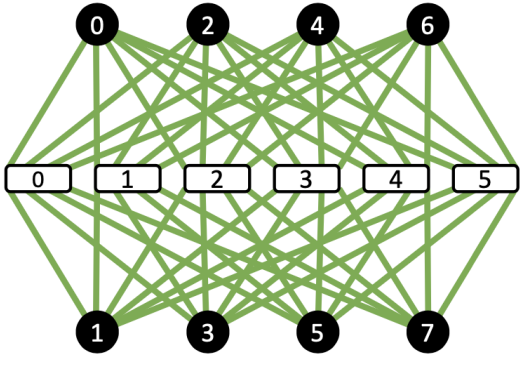


c) Frontier





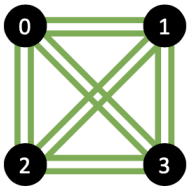
d) Sunspot



e) ThetaGPU

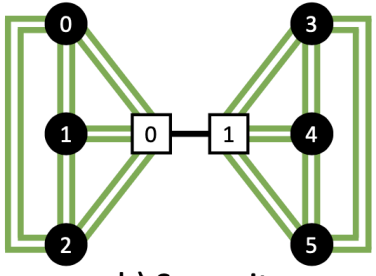
Networks and communication libraries are diverse!

MPI, NCCL



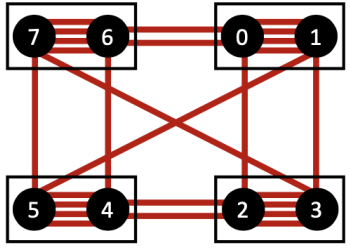
a) Delta, Perlmutter

MPI, NCCL

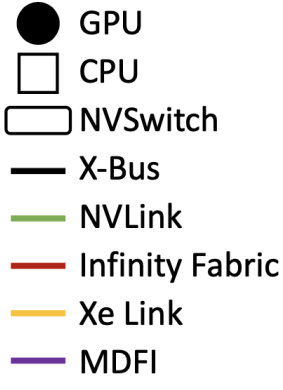


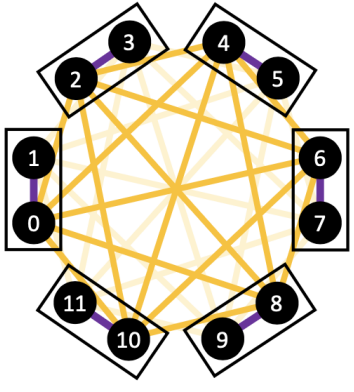
b) Summit

MPI, RCCL



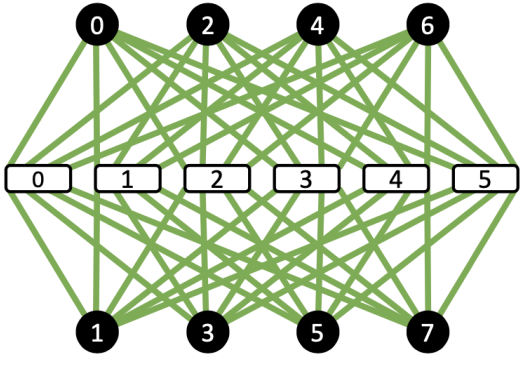
c) Frontier





d) Sunspot

MPI



e) ThetaGPU

MPI, NCCL

Communication Libraries

1) **MPI**
2) **NCCL**
3) **IPC**
4) GASNet
5) SHMEM

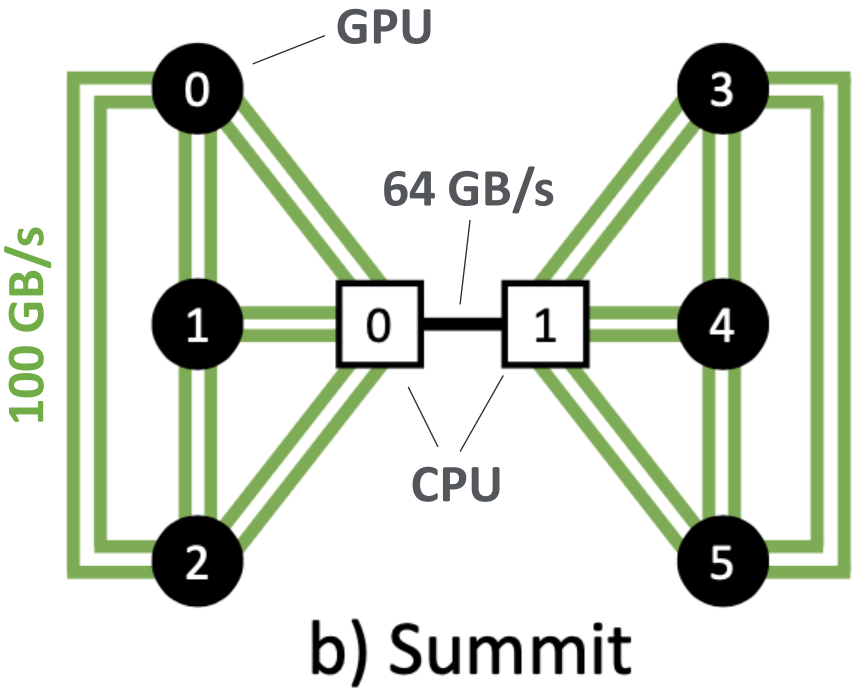
Runtime Tools for Communications

1) CommBench: Configurable Benchmarking

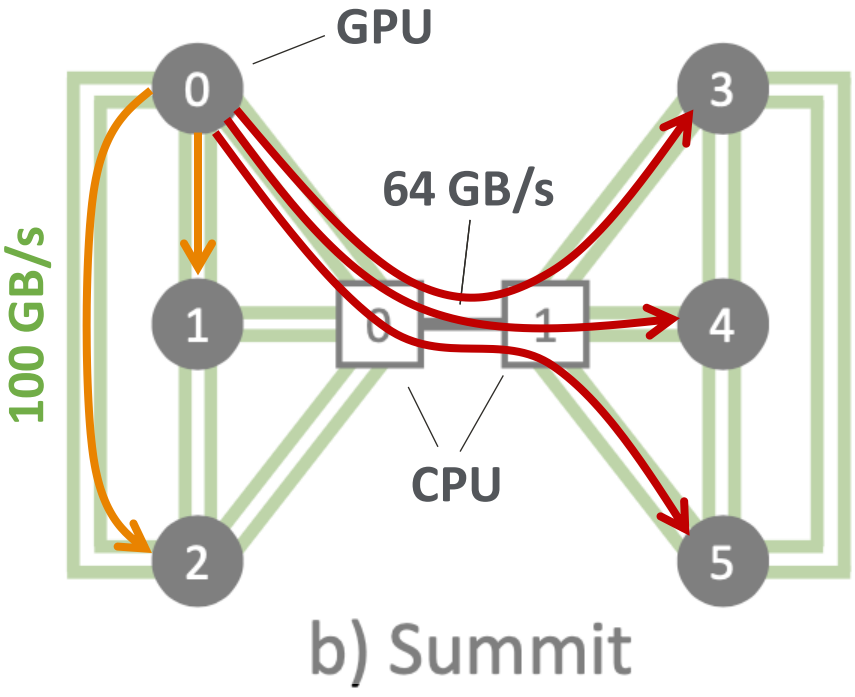
2) HiCCL: Hierarchical Collective Communications 

3) Application Highlight: 3D Image Reconstruction

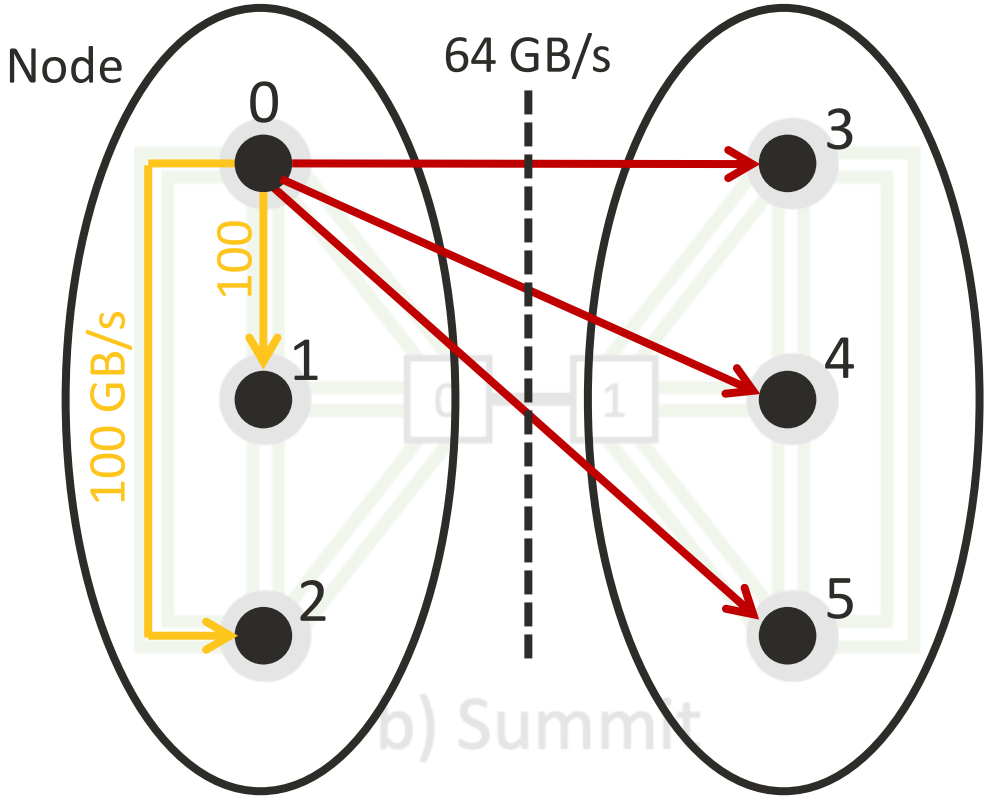
Networks are hierarchical.



Networks are hierarchical.



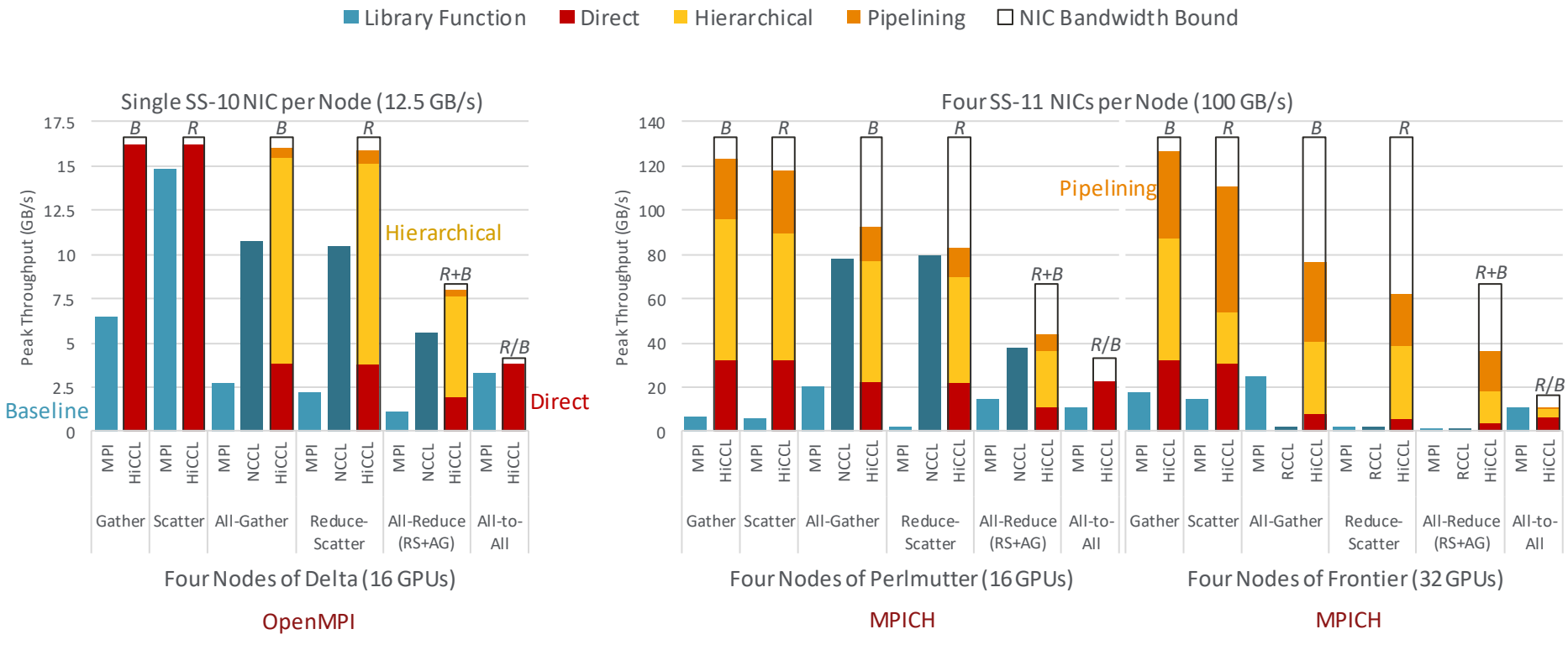
Two-Level Hierarchy



HiCCL: A Hierarchical Collective Communication Library

Stanford University

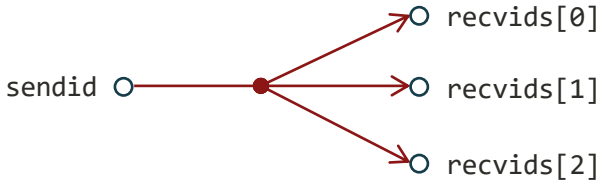
Collective Throughput



Composition Primitives: Broadcast (*B*), Reduce (*R*)

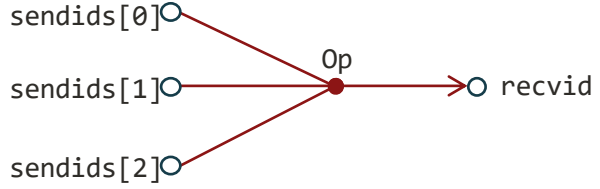
Pattern Composition: Primitives

Broadcast



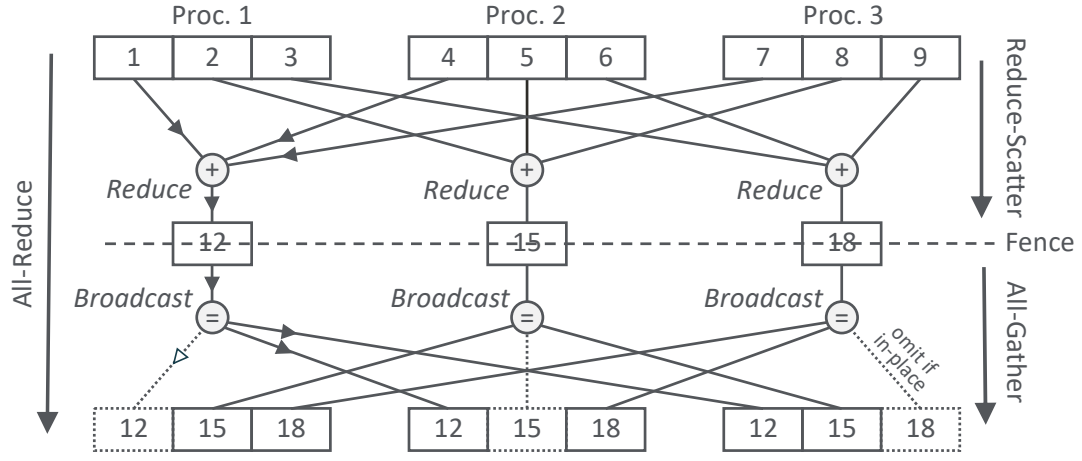

```
void Comm::add_bcast(T *sendbuf, size_t sendoffset,
                     T *recvbuf, size_t recvoffset,
                     size_t count,
                     int sendid,
                     std::vector<int> recvids);
```

Reduce




```
void Comm::add_reduce(T *sendbuf, size_t sendoffset,
                      T *recvbuf, size_t recvoffset,
                      size_t count,
                      std::vector<int> sendids,
                      int recvid);
```

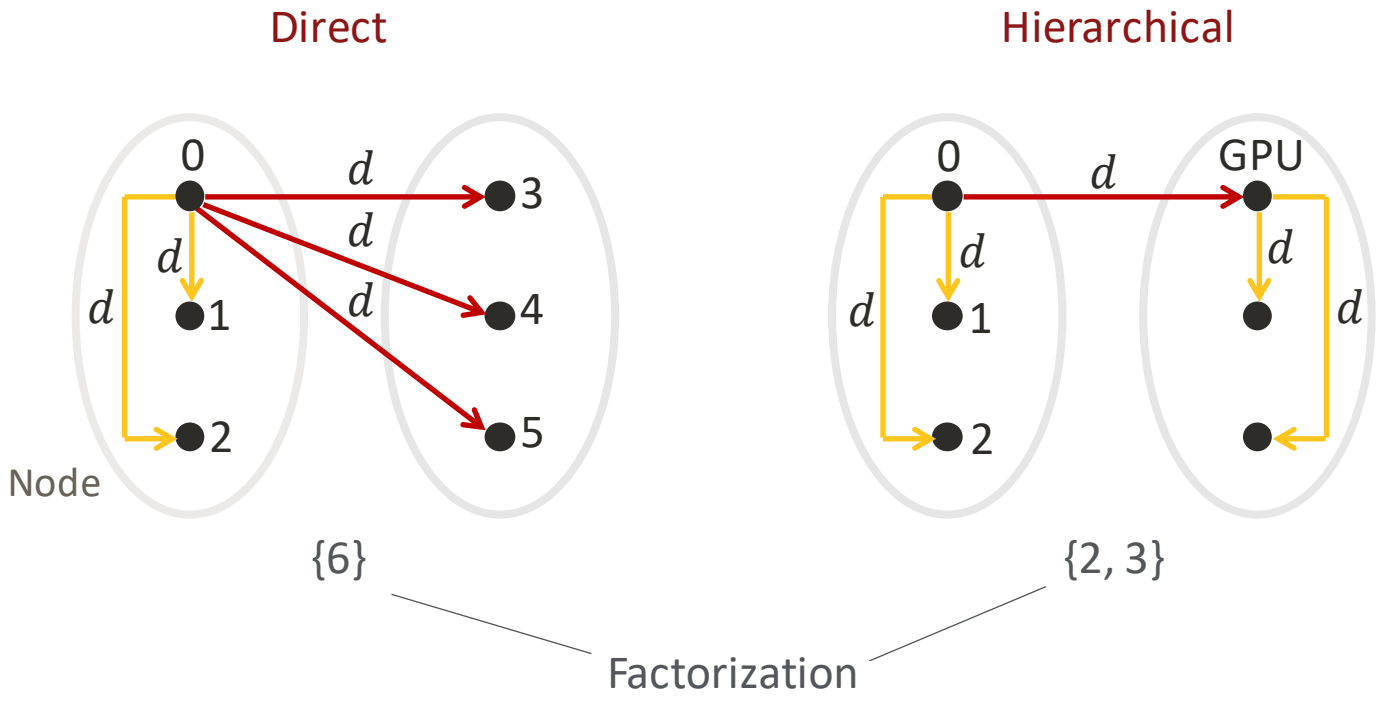
Pattern Composition: All-Reduce




```
HiCCL::Comm<Type> allreduce;

// REDUCE-SCATTER
for(int recvid = 0; recvid < numrank; recvid++)
  allreduce.add_reduce(sendbuf, recvid * count,
                       recvbuf, recvid * count,
                       count, proclist, recvid);
// DEPENDENCY
allreduce.fence();
// ALL-GATHER
for(int sendid = 0; sendid < numrank; sendid++)
  allreduce.add_broadcast(recvbuf, sendid * count,
                          recvbuf, sendid * count,
                          count, sendid, otherlist);
```

```
// PARAMETERS FOR FRONTIER
std::vector<int> hierarchy = {numrank / 8, 4, 2};
std::vector<Library> library = {MPI, MPI, MPI, IPC};
int ring(1);
int stripe(1);
int pipeline(16);

// INITIALIZATION
allreduce.init(hierarchy, library, ring, stripe, pipeline);
```

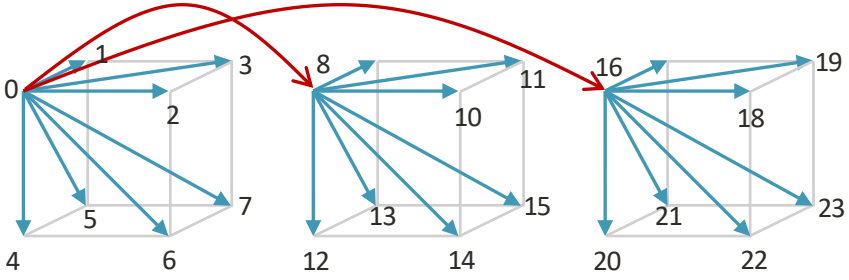
Hierarchical Broadcast



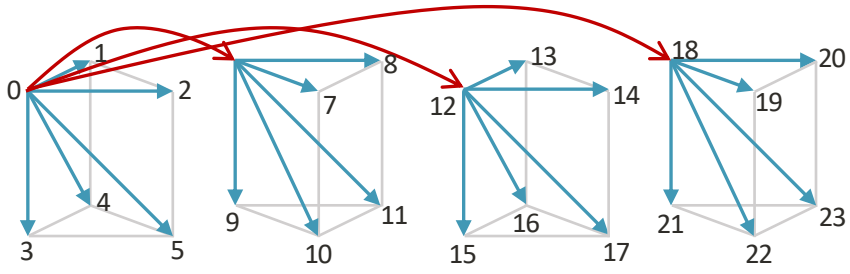
Tree Factorization

24 GPUs

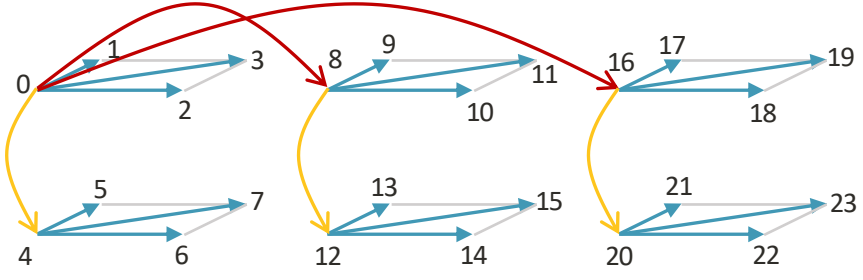
(a) {3, 8}



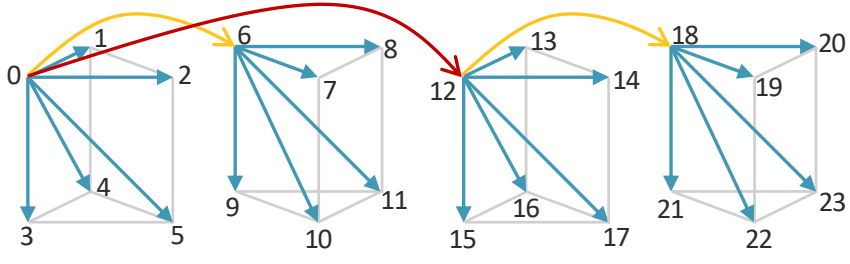
(b) {4, 6}



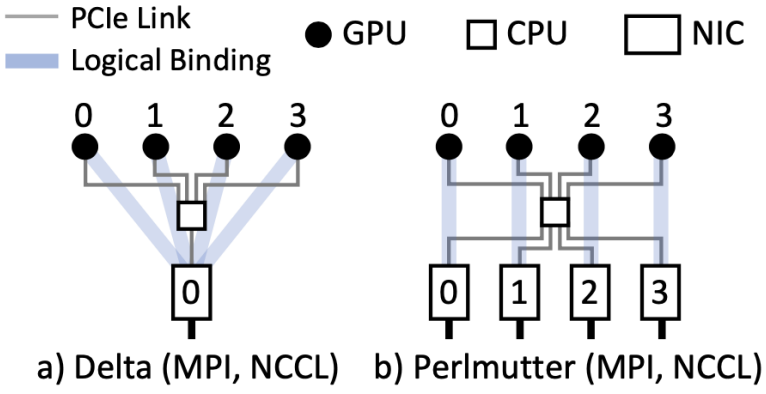
(c) {3, 2, 4}



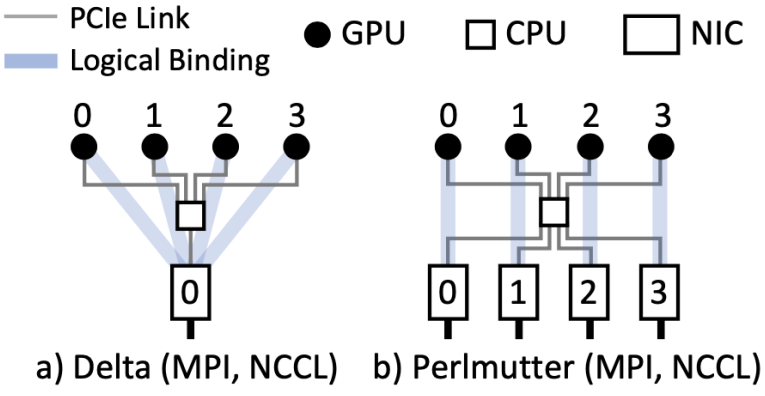
(d) {2, 2, 6}



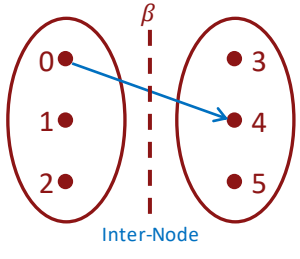
Optimization: Multi-Rail Striping



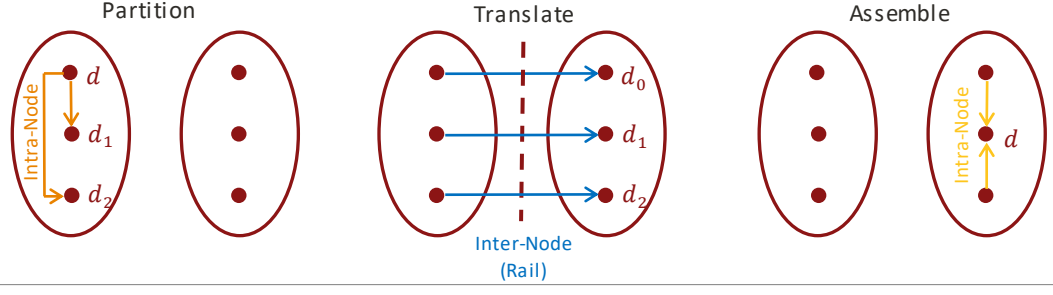
Optimization: Multi-Rail Striping



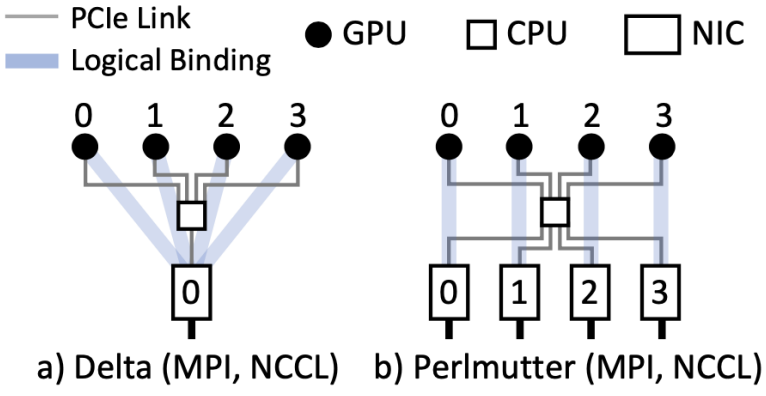
Point-to-Point Comm.



Striping Algorithm

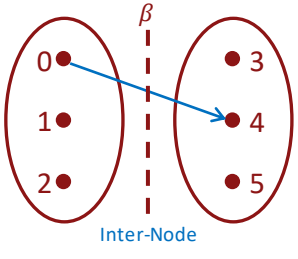


Optimization: Multi-Rail Striping

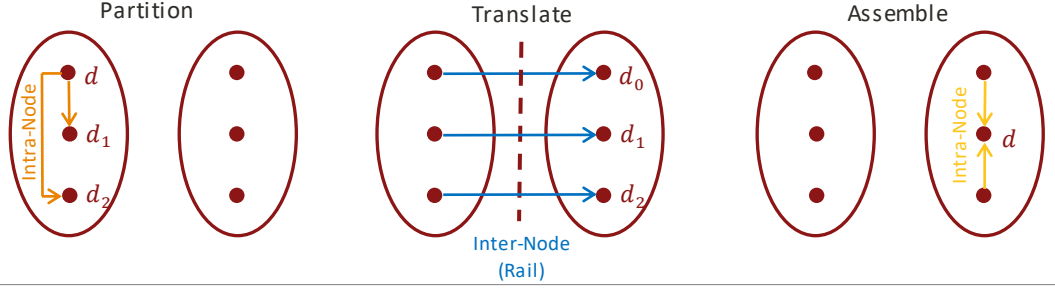



```
using namespace CommBench;

Comm<Type> partition(Library::IPC);
Comm<Type> translate(Library::NCCL);
Comm<Type> assemble(Library::IPC);

Type *send_temp;
Type *recv_temp;
allocate(send_temp, count, 1);
allocate(send_temp, count, 2);
allocate(recv_temp, count, 3);
allocate(recv_temp, count, 5);

partition.add(sendbuf, count, send_temp, 0, count, 0, 1);
partition.add(sendbuf, 2 * count, send_temp, 0, count, 0, 2);
translate.add(sendbuf, 0, recv_temp, 0, count, 0, 3);
translate.add(send_temp, 0, recvbuf, count, count, 1, 4);
translate.add(send_temp, 0, recv_temp, 0, count, 2, 5);
assemble.add(recv_temp, 0, recvbuf, 0, count, 3, 4);
assemble.add(recv_temp, 0, recvbuf, 2 * count, count, 5, 4);

partition.start();
partition.wait();
translate.start();
translate.wait();
assemble.start();
assemble.wait();
```

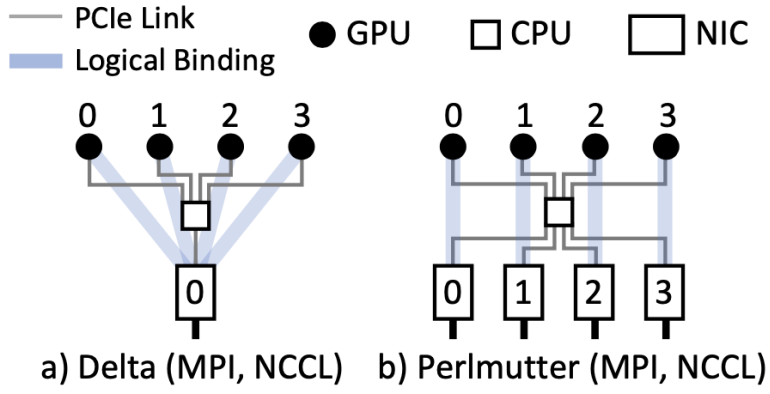
Point-to-Point Comm.

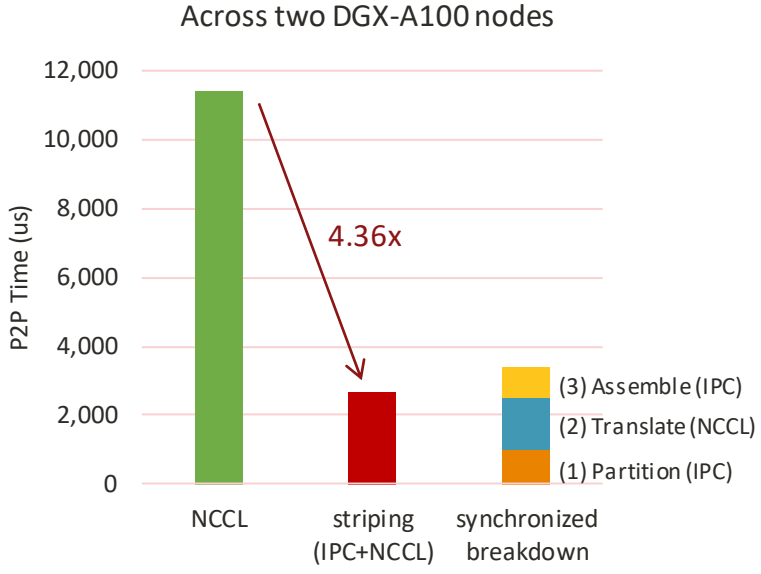


Striping Algorithm

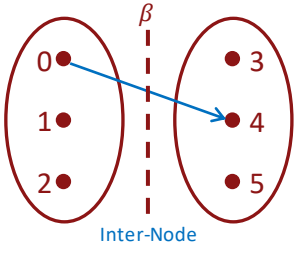


Optimization: Multi-Rail Striping

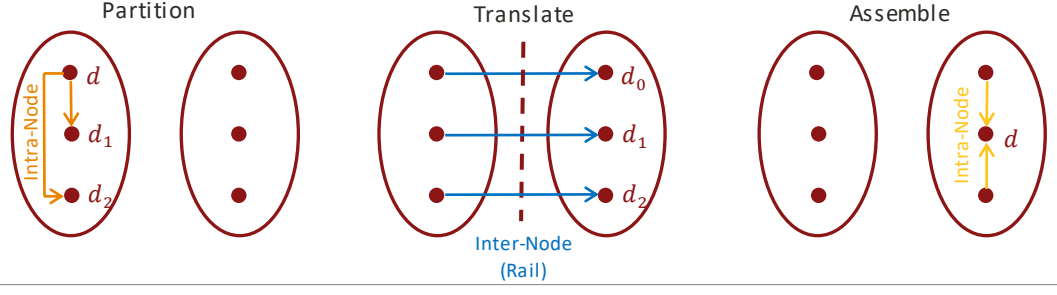




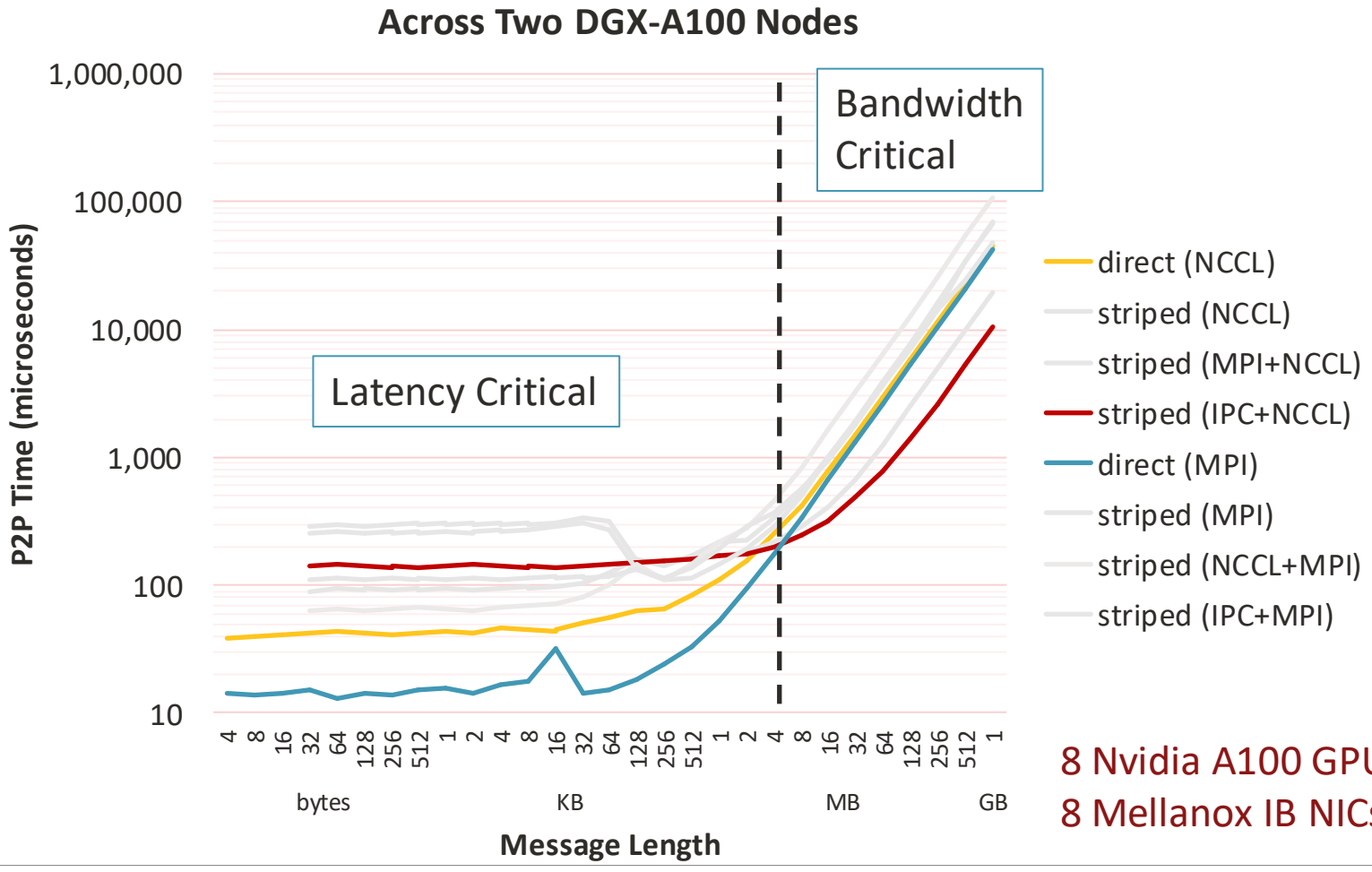
Point-to-Point Comm.



Striping Algorithm

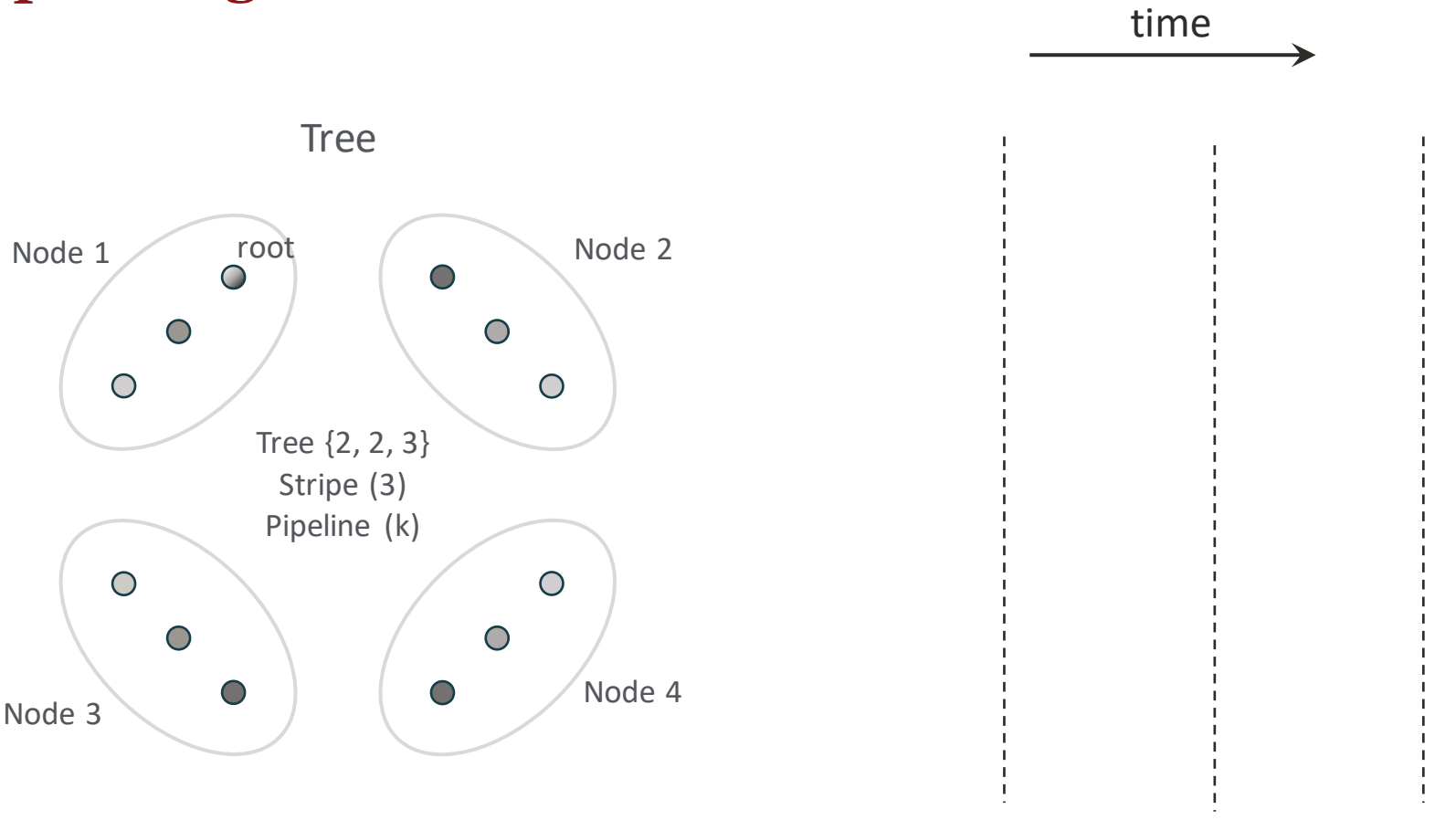


Bandwidth or Latency?

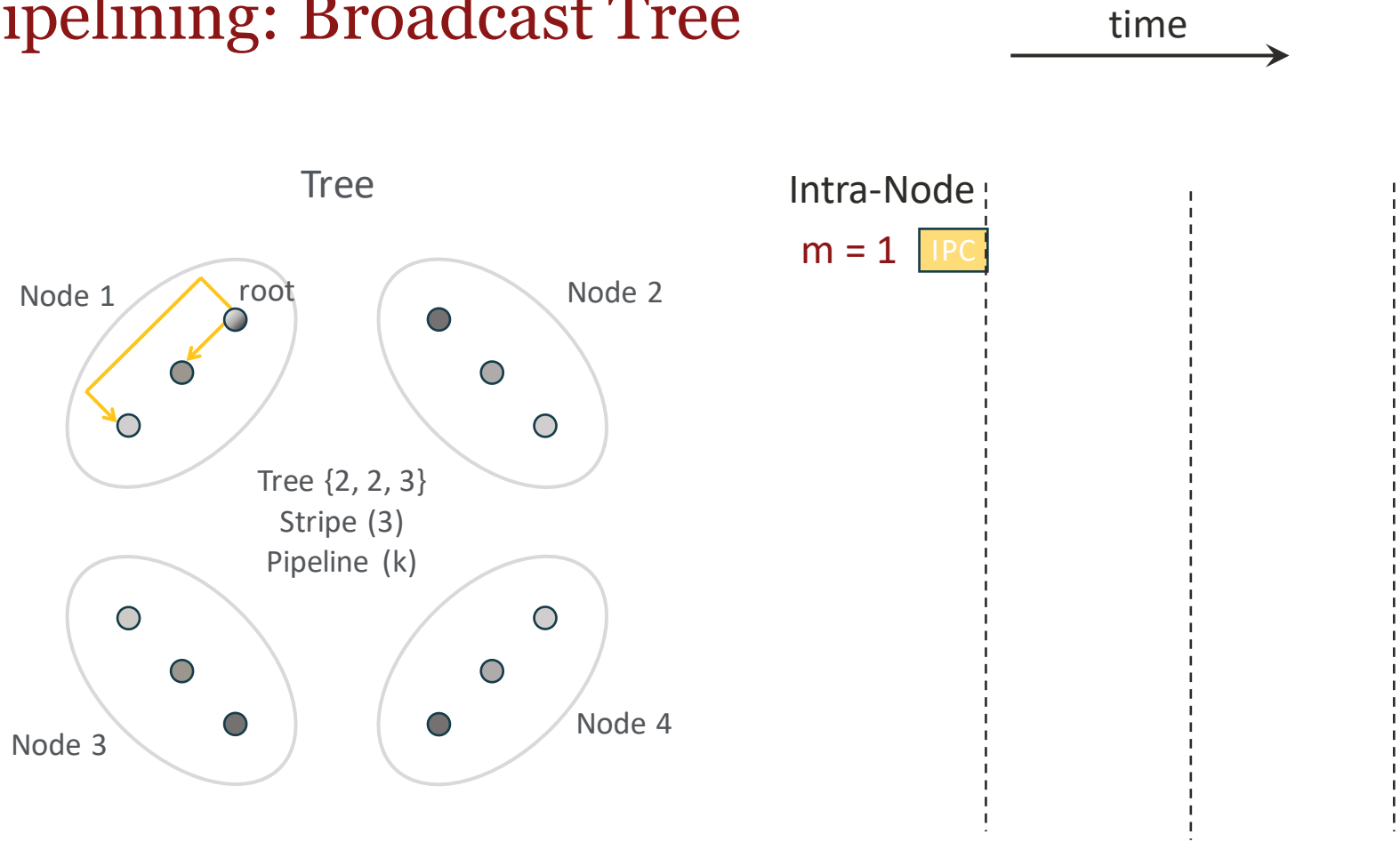


8 Nvidia A100 GPUs
8 Mellanox IB NICs

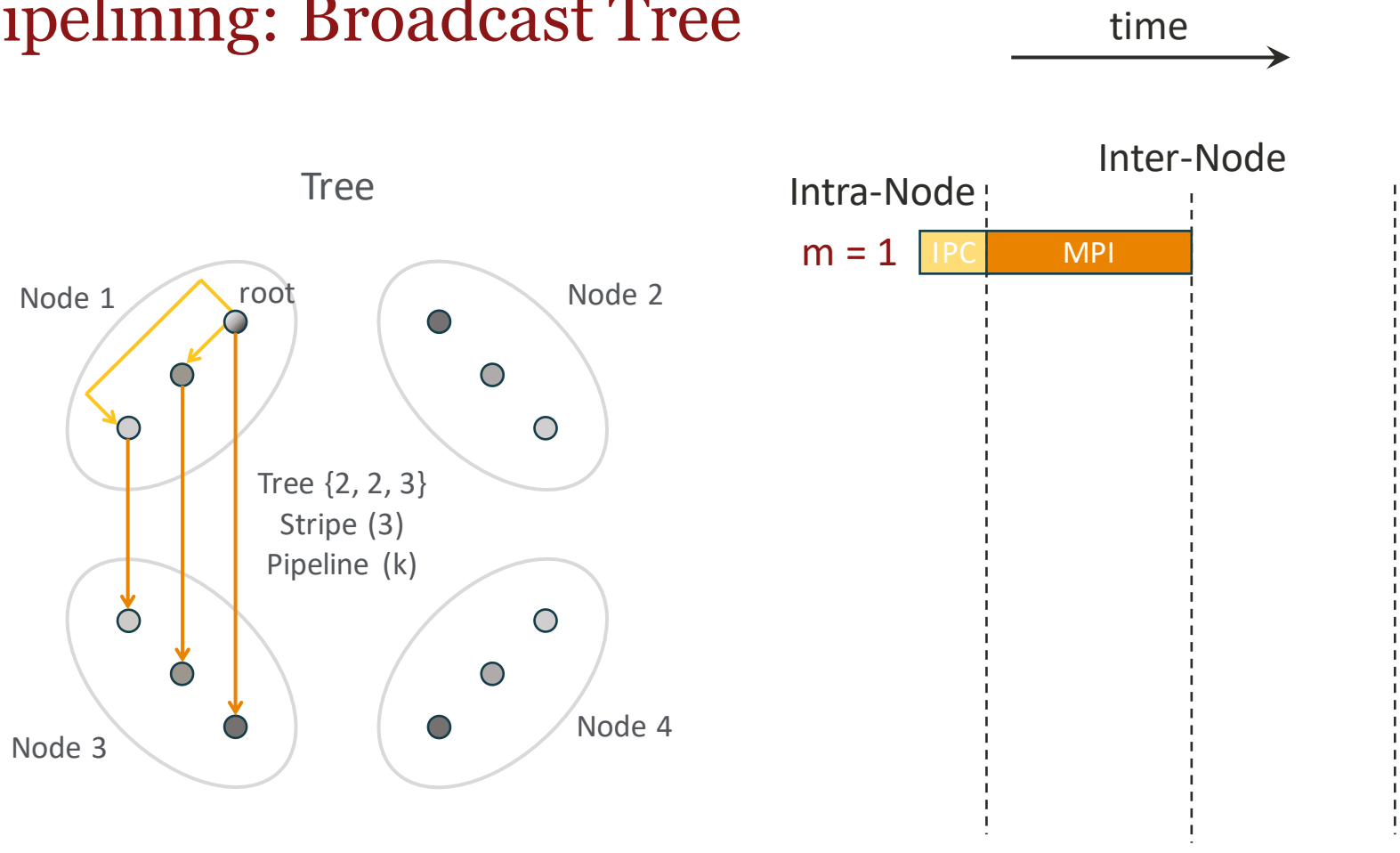
Pipelining: Broadcast Tree



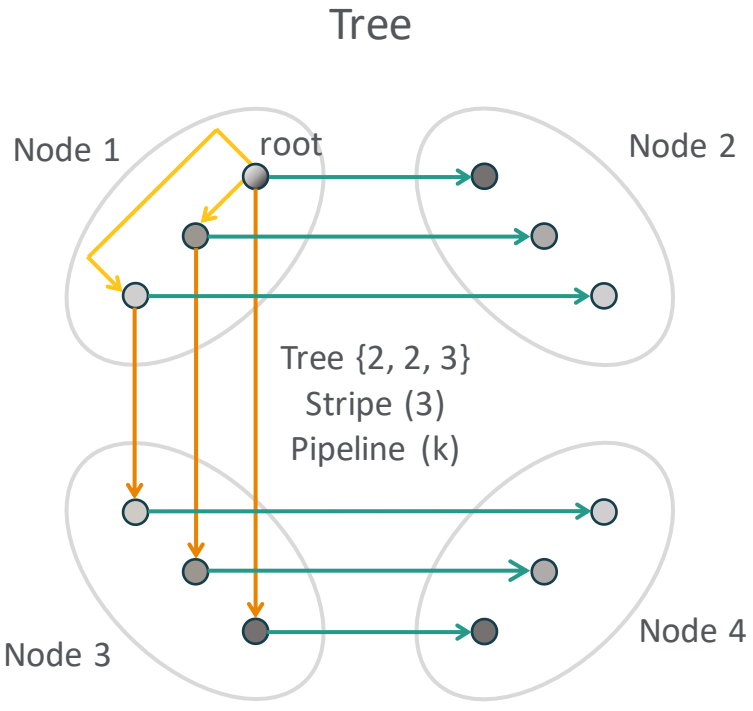
Pipelining: Broadcast Tree

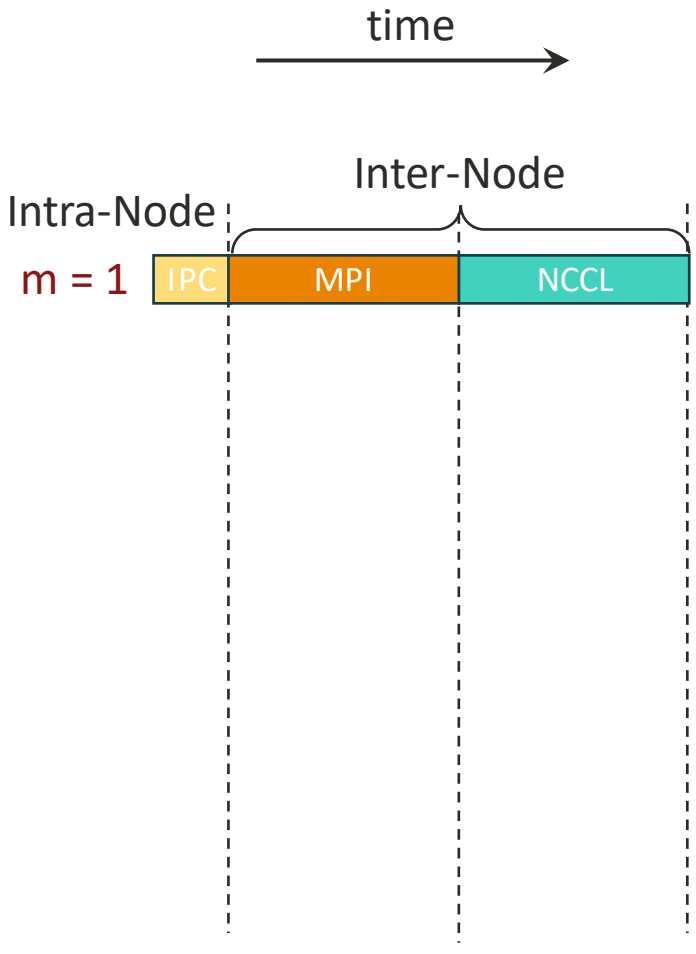


Pipelining: Broadcast Tree

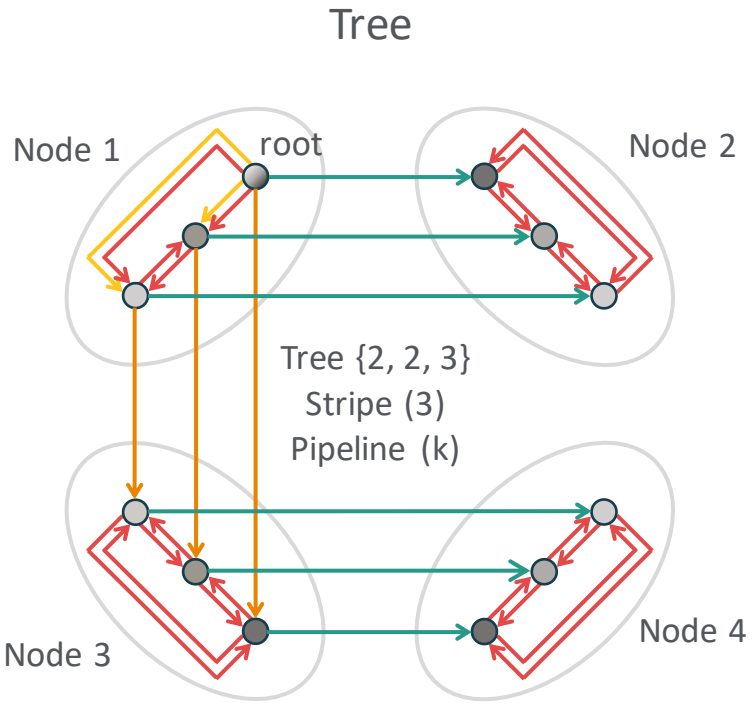


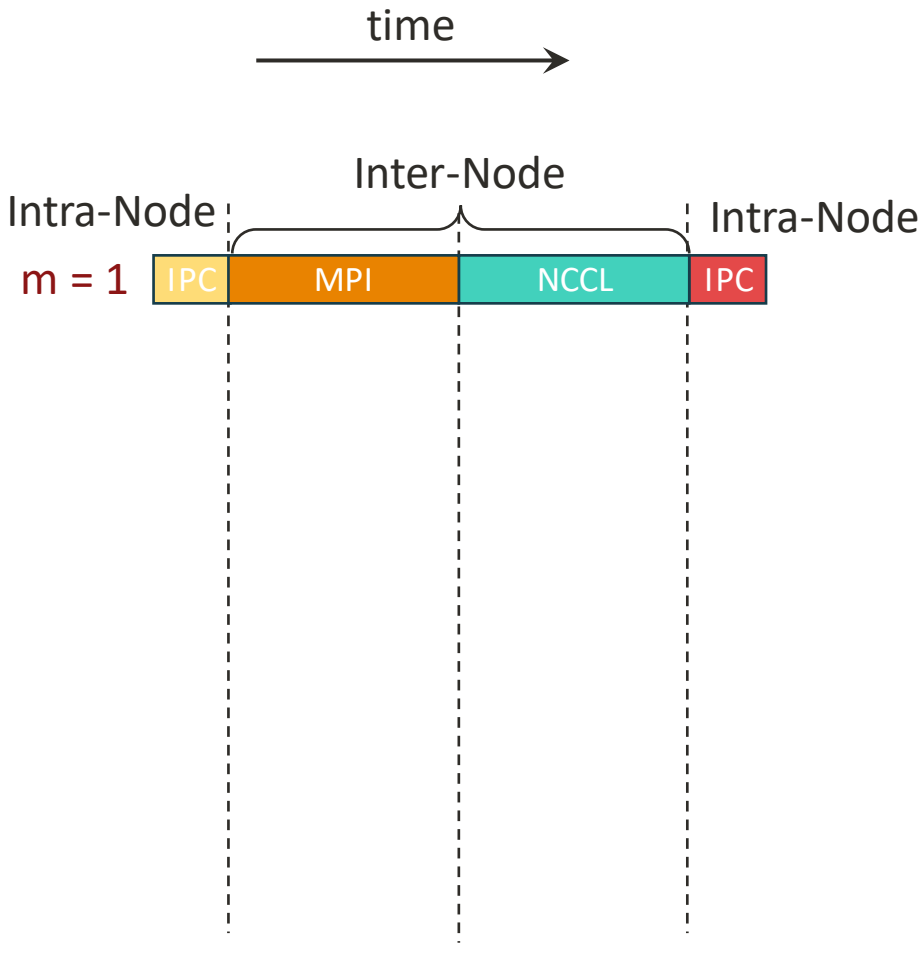
Pipelining: Broadcast Tree



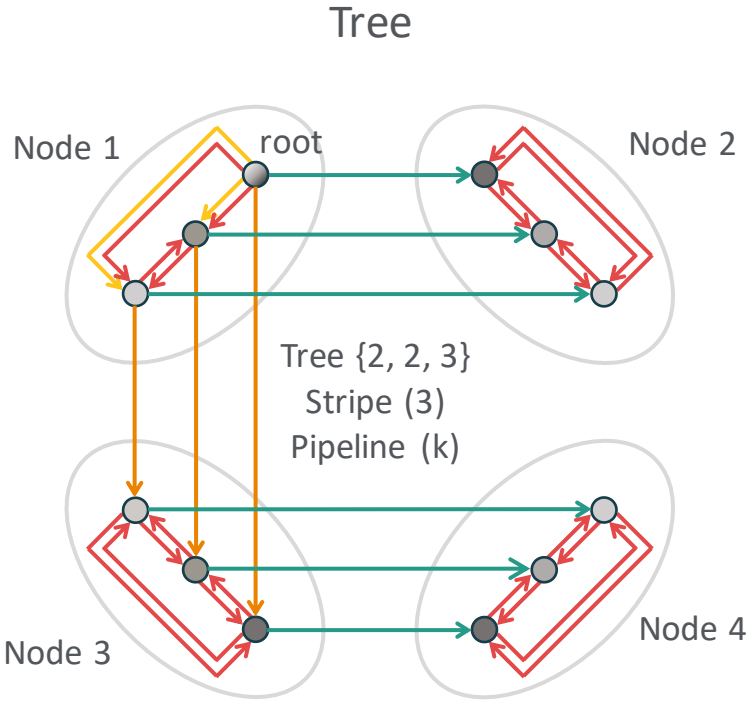


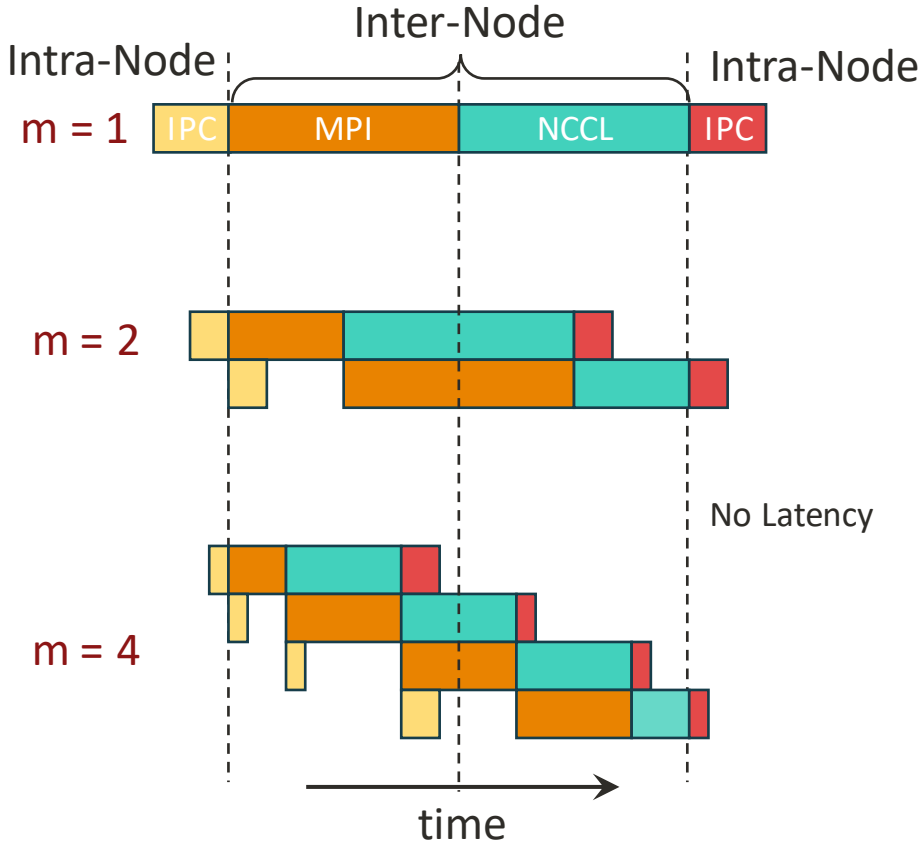
Pipelining: Broadcast Tree



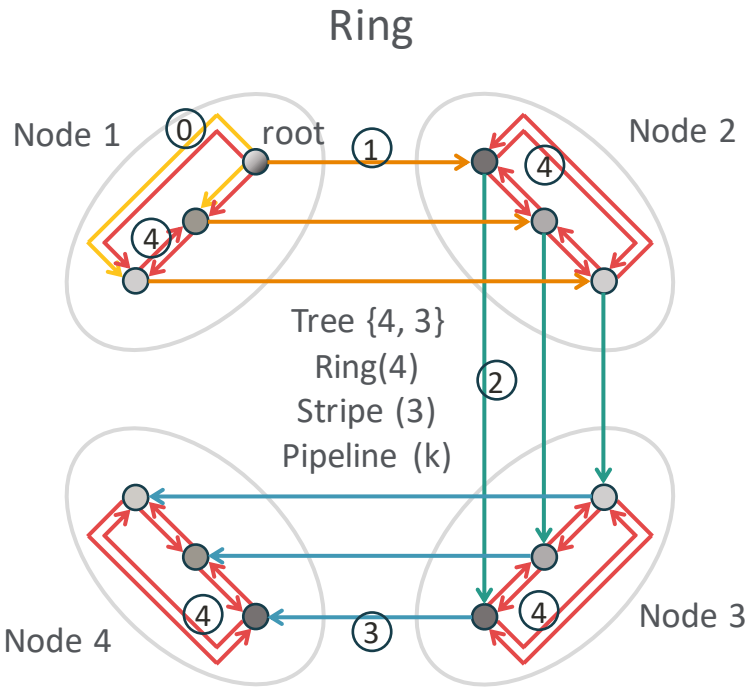


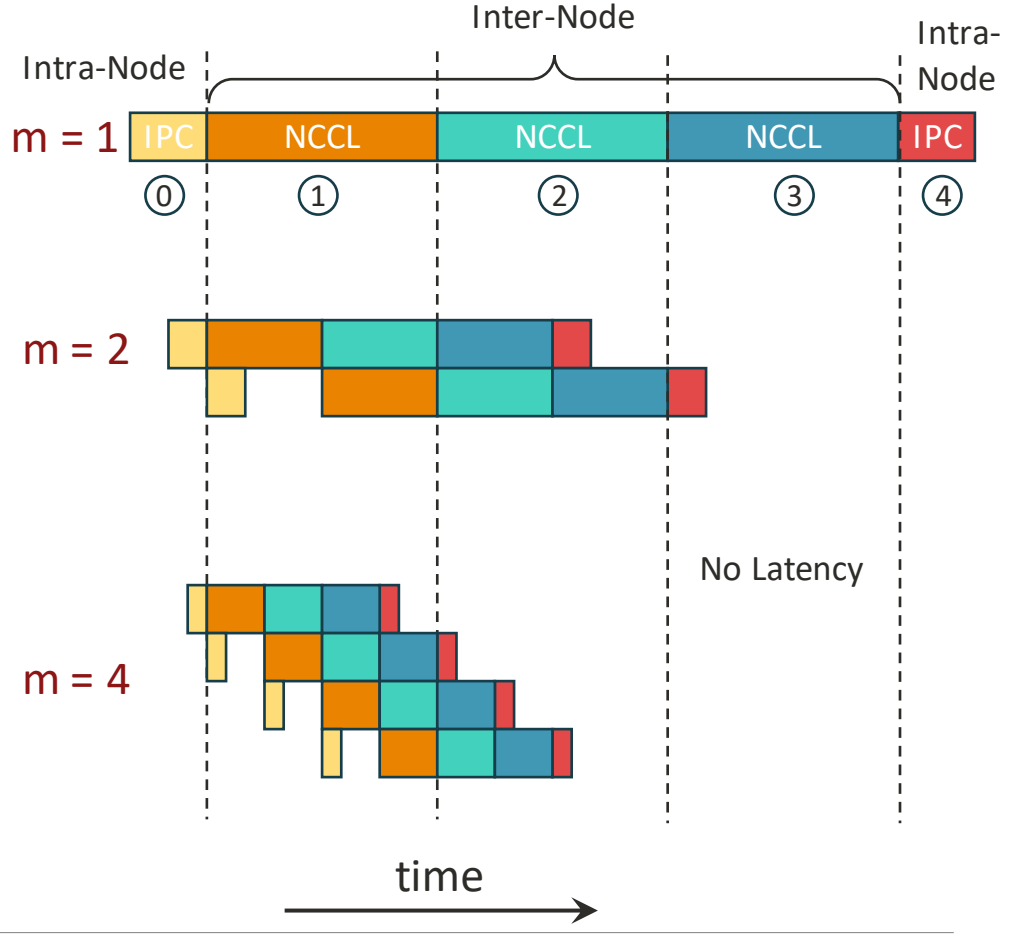
Pipelining: Broadcast Tree

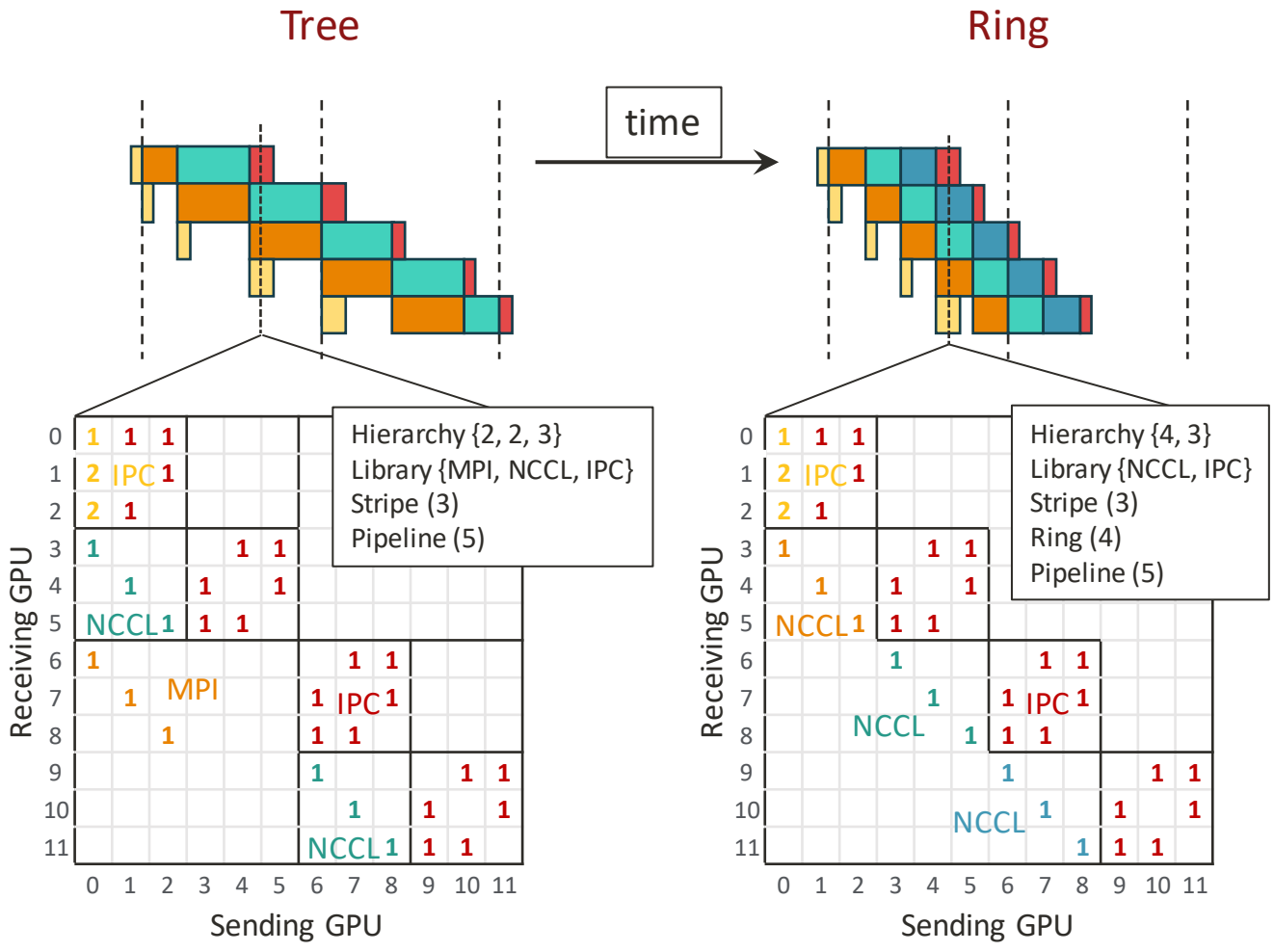




Pipelining: Broadcast Ring

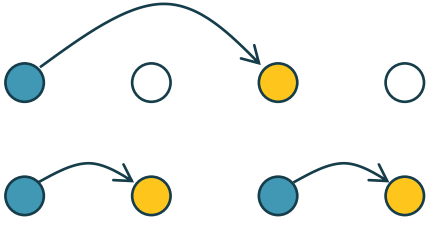





Tree
Ring
time
Hierarchy {2, 2, 3}
Library {MPI, NCCL, IPC}
Stripe (3)
Pipeline (5)
Hierarchy {4, 3}
Library {NCCL, IPC}
Stripe (3)
Ring (4)
Pipeline (5)
Receiving GPU
Sending GPU
IPC
NCCL
MPI
0 1 2 3 4 5 6 7 8 9 10 11

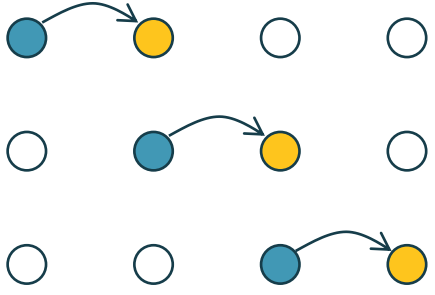
t total time
n number of nodes
d message length
α latency per message
f bandwidth per node
k pipeline depth

Tree



$$t = \left(\alpha + \frac{d}{f}\right)\log_2 n$$

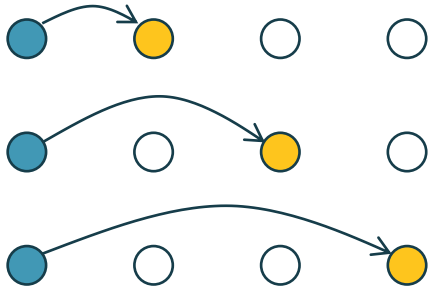
Ring



$$t = \left(\alpha + \frac{d}{fk}\right)(k + n - 2)$$

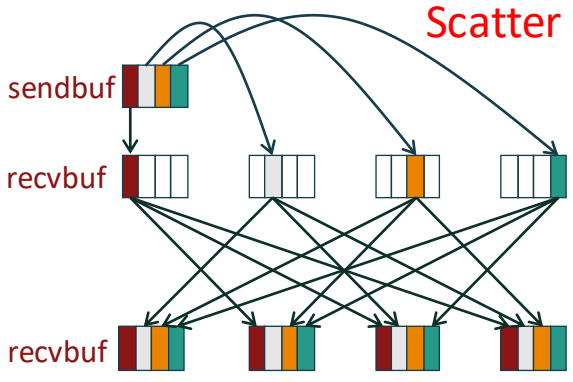
t total time
n number of nodes
d message length
α latency per message
f bandwidth per node

Direct



$$t = \left(\alpha + \frac{d}{f}\right)(n-1)$$

Scatter

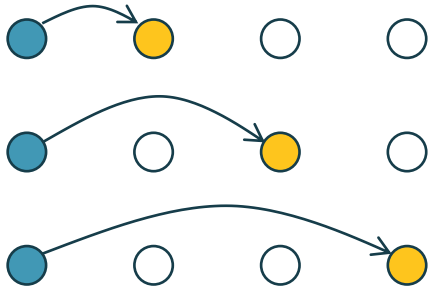


+All-Gather

$$t = 2\left(\alpha + \frac{d}{fn}\right)(n-1)$$

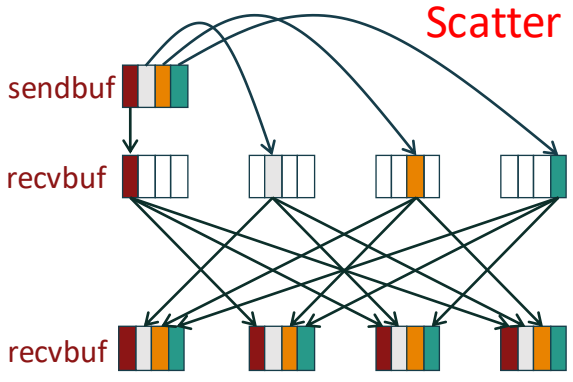
t total time
n number of nodes
d message length
α latency per message
f bandwidth per node

Direct



$$t = \left(\alpha + \frac{d}{f}\right)(n-1)$$

Scatter



+All-Gather

$$t = 2\left(\alpha + \frac{d}{fn}\right)(n-1)$$

Tree

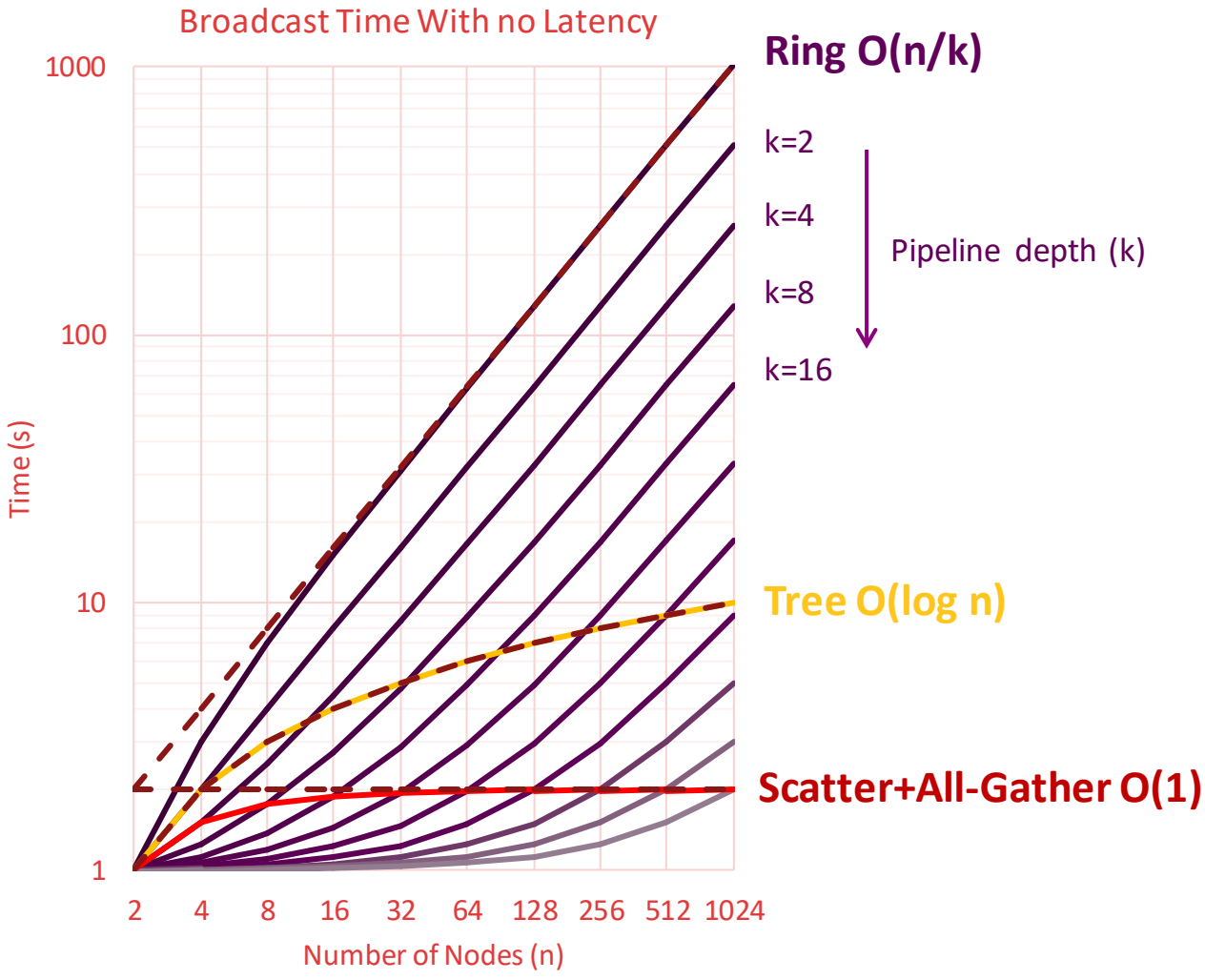
$$t = \left(\alpha + \frac{d}{f}\right)\log_2 n$$

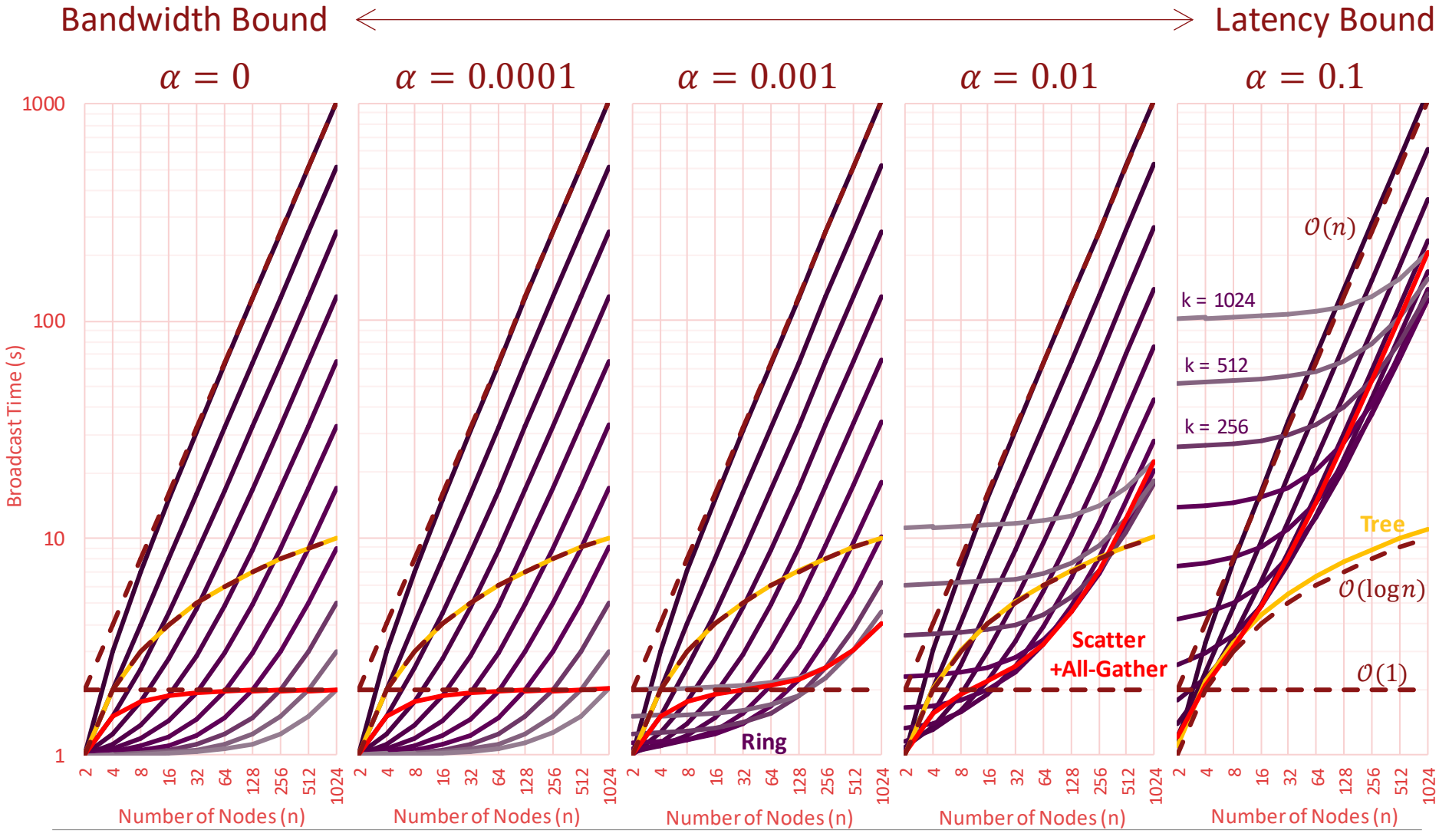
Ring

$$t = \left(\alpha + \frac{d}{fk}\right)(k + n - 2)$$

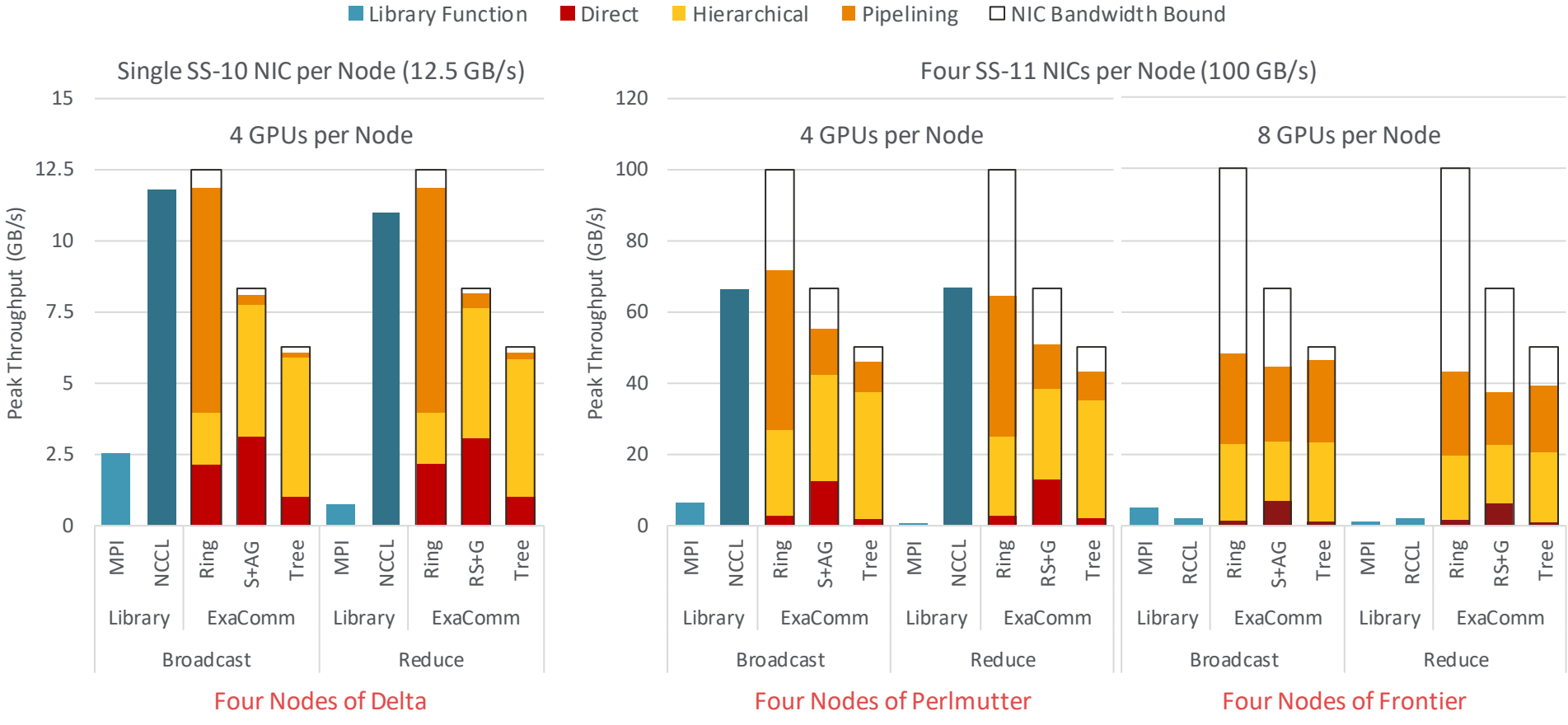
Scatter+All-Gather

$$t = 2\left(\alpha + \frac{d}{fn}\right)(n - 1)$$

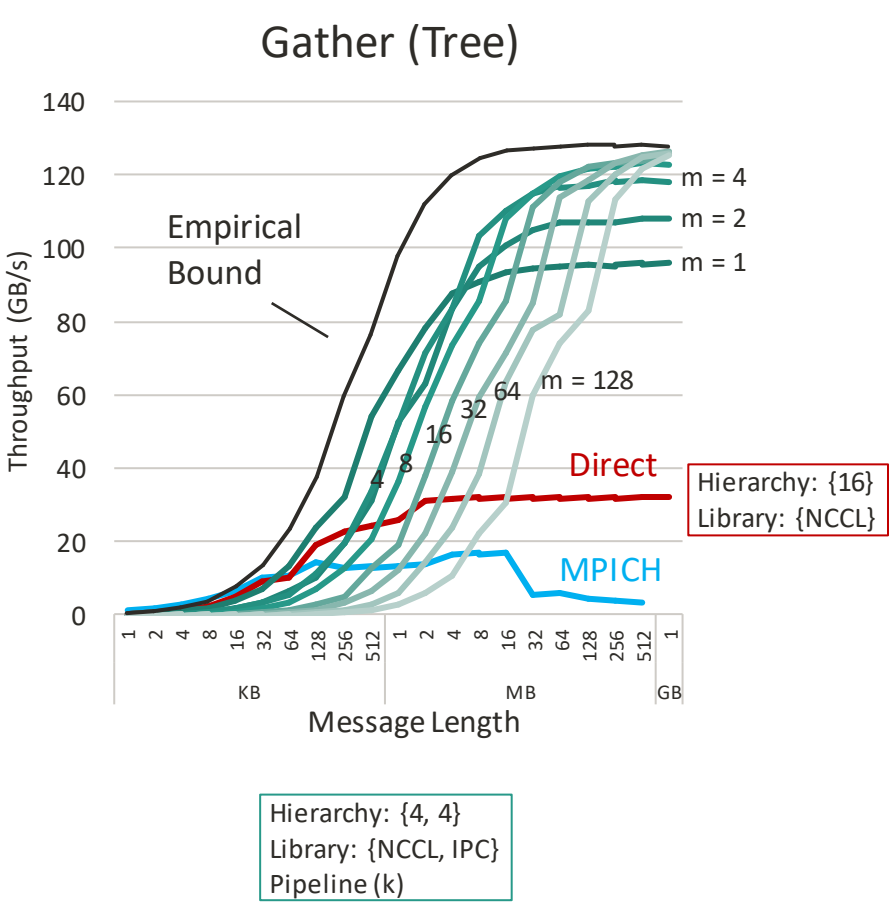


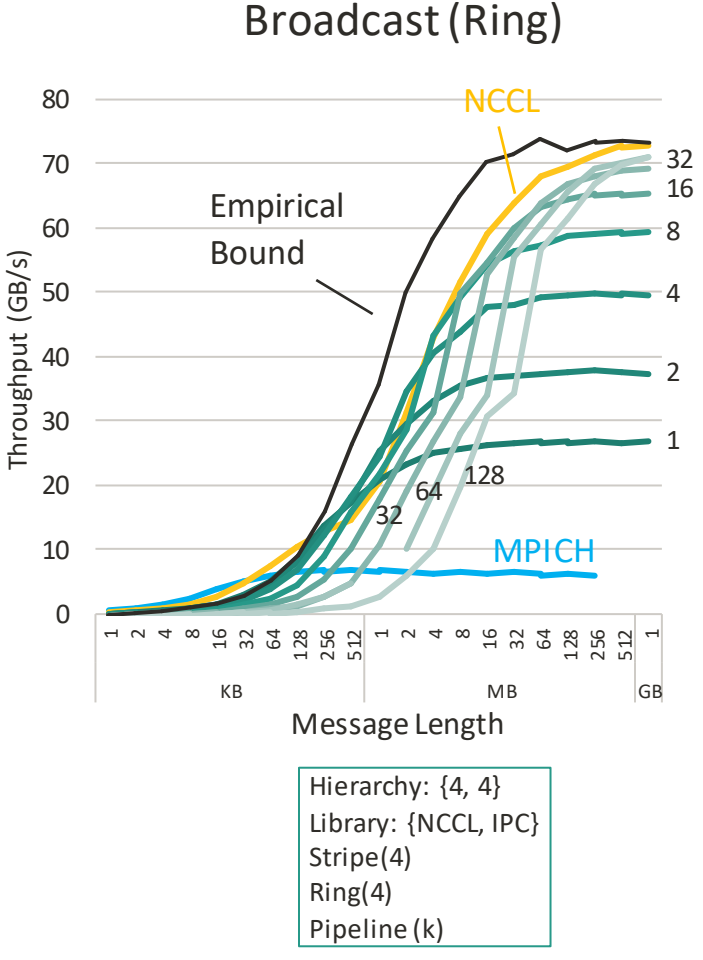

Bandwidth Bound
Latency Bound
$\alpha = 0$
$\alpha = 0.0001$
$\alpha = 0.001$
$\alpha = 0.01$
$\alpha = 0.1$
1000
100
10
1
Broadcast Time (s)
2 4 8 16 32 64 128 256 512 1024
Number of Nodes (n)
Ring
Scatter
+All-Gather
$\mathcal{O}(n)$
k = 1024
k = 512
k = 256
Tree
$\mathcal{O}(\log n)$
$\mathcal{O}(1)$

Primitive Performance: Broadcast and Reducecc

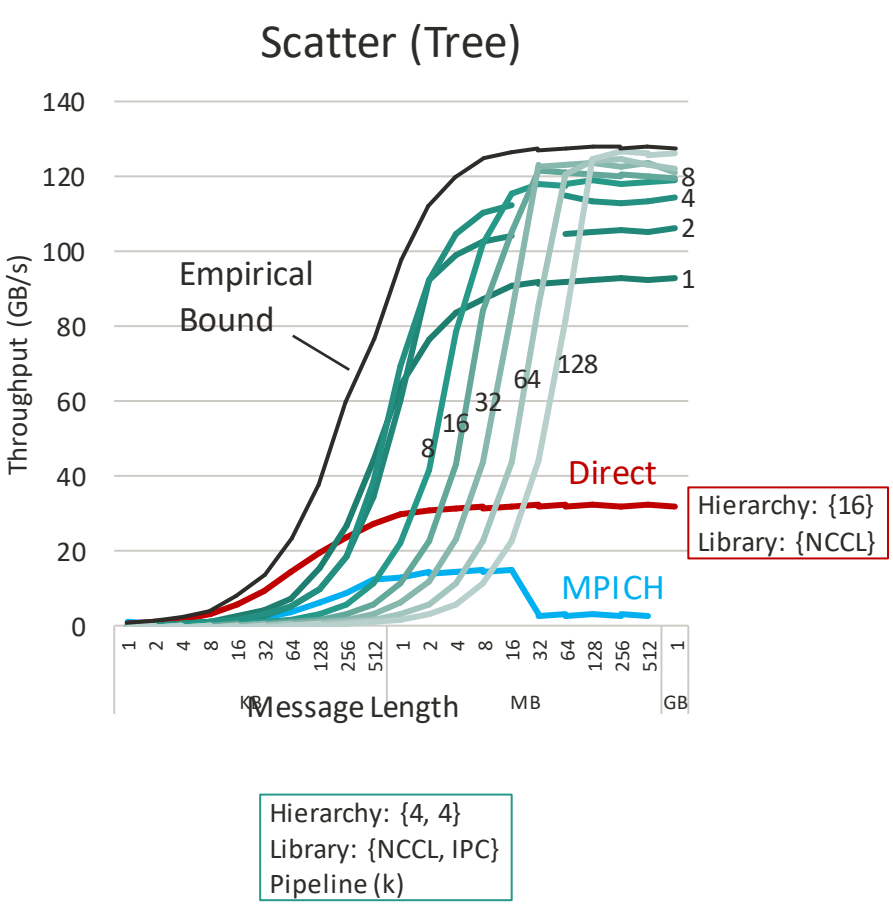


Pipelining: Results on Perlmutter

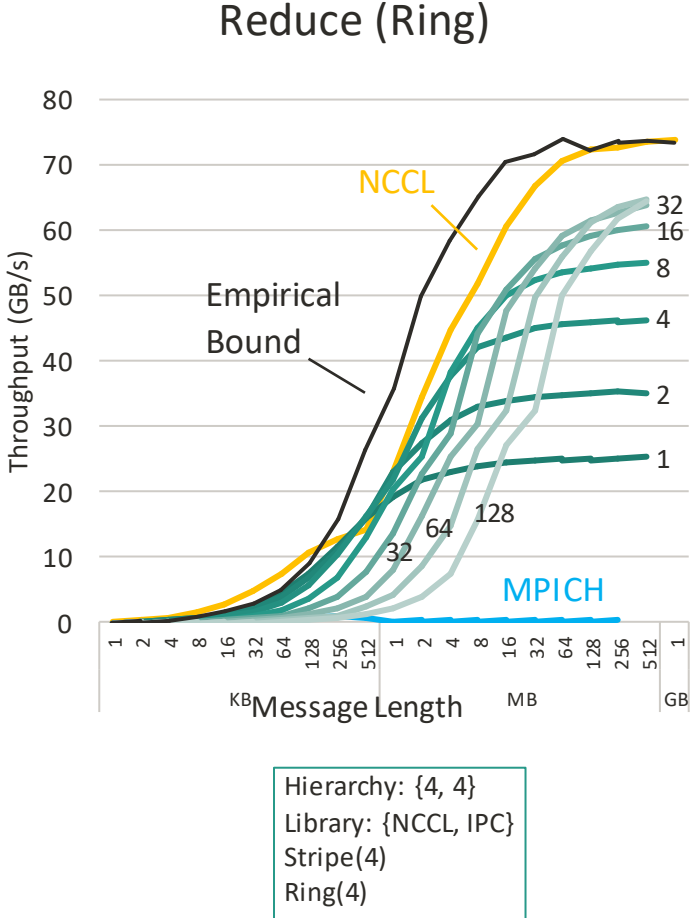




Pipelining: Results on Perlmutter



Hierarchy: {4, 4}
Library: {NCCL, IPC}
Pipeline (k)



Hierarchy: {4, 4}
Library: {NCCL, IPC}
Stripe(4)
Ring(4)
Pipeline (k)

Runtime Tools for Communications

1) CommBench: Configurable Benchmarking

2) HiCCL: Hierarchical Collective Communications 

3) Application Highlight: 3D Image Reconstruction

Runtime Tools for Communications

1) CommBench: Configurable Benchmarking

2) HiCCL: Hierarchical Collective Communications

3) Application Highlight: 3D Image Reconstruction

Application Highlight:

3D Image Reconstruction

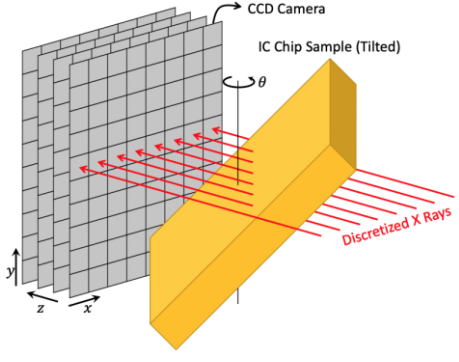
Application Highlight: Image Reconstruction

Synchrotron Light Source

APS, Argonne National Lab

summit

Summit, Oak Ridge National Lab



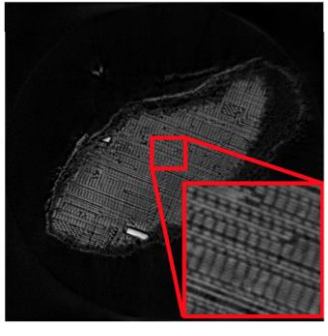
$$\boldsymbol{A} \times \boldsymbol{B} = \boldsymbol{C}$$

Model (Sparse)

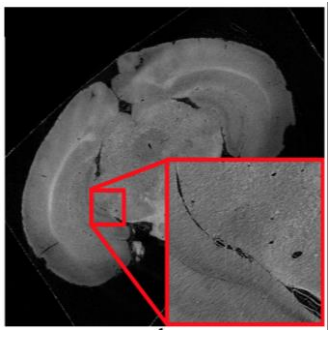
3D Image (Dense)

Measurement (Dense)

Integrated Circuit



Mouse Brain



M. Hidayetoglu, T. Bicer, S. Garcia de Gonzalo, B. Ren, V. De Andrade, D. Gursoy, R. Kettimuthu, I. T. Foster, and W.-M. W. Hwu, "Petascale XCT: 3D image reconstruction with hierarchical communications on multi-GPU nodes," SC20, Best Paper.

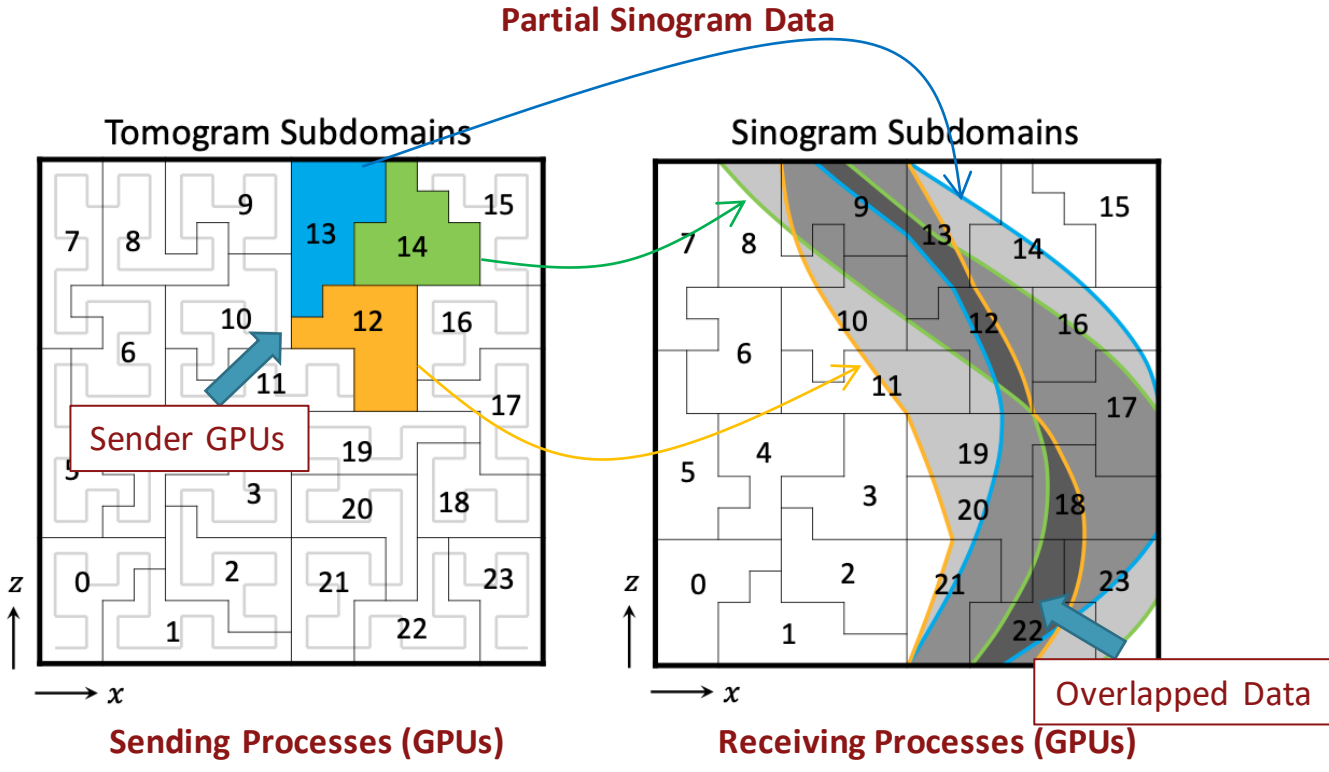
Application Highlight: Image Reconstruction

$$A \times B = C$$

Model (Sparse)

3D Image (Dense)

Measurement (Dense)



M. Hidayetoglu, T. Bicer, S. Garcia de Gonzalo, B. Ren, V. De Andrade, D. Gursoy, R. Kettimuthu, I. T. Foster, and W.-M. W. Hwu, "Petascale XCT: 3D image reconstruction with hierarchical communications on multi-GPU nodes," SC20, Best Paper.

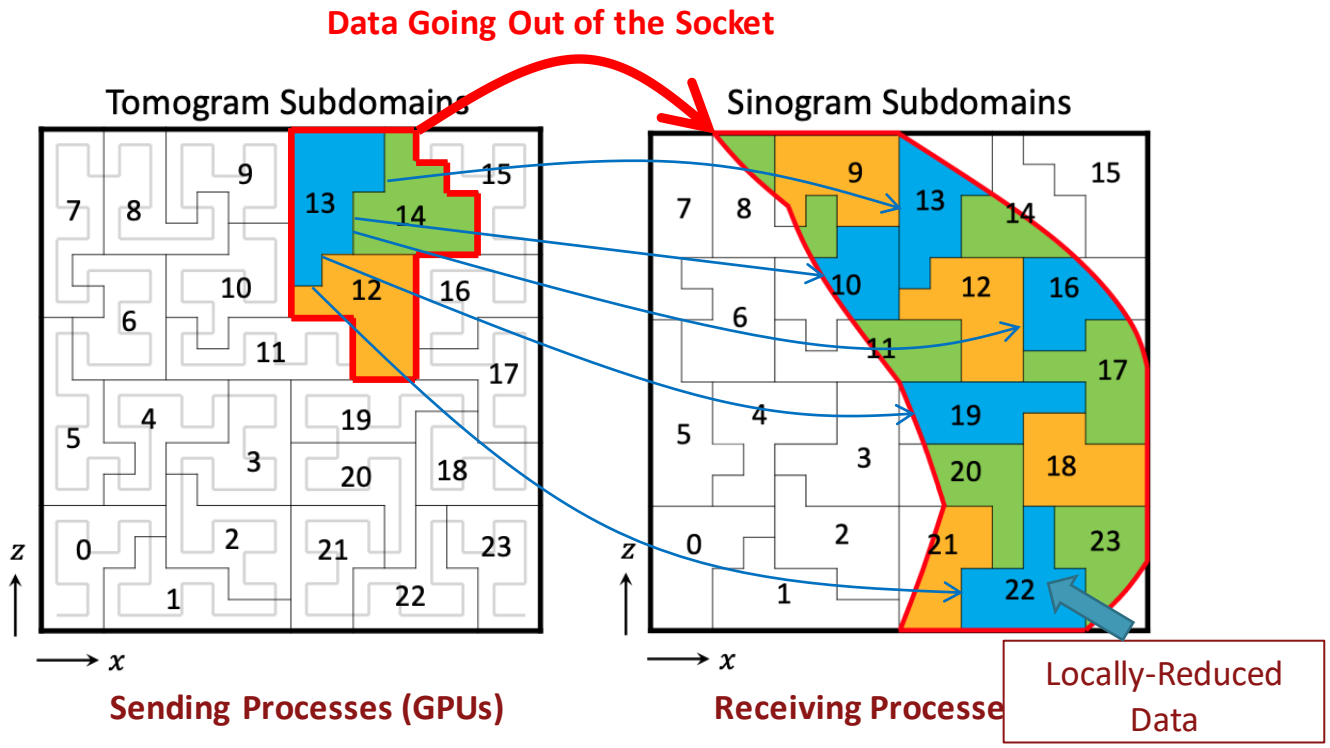
Application Highlight: Image Reconstruction

$$A \times B = C$$

Model (Sparse)

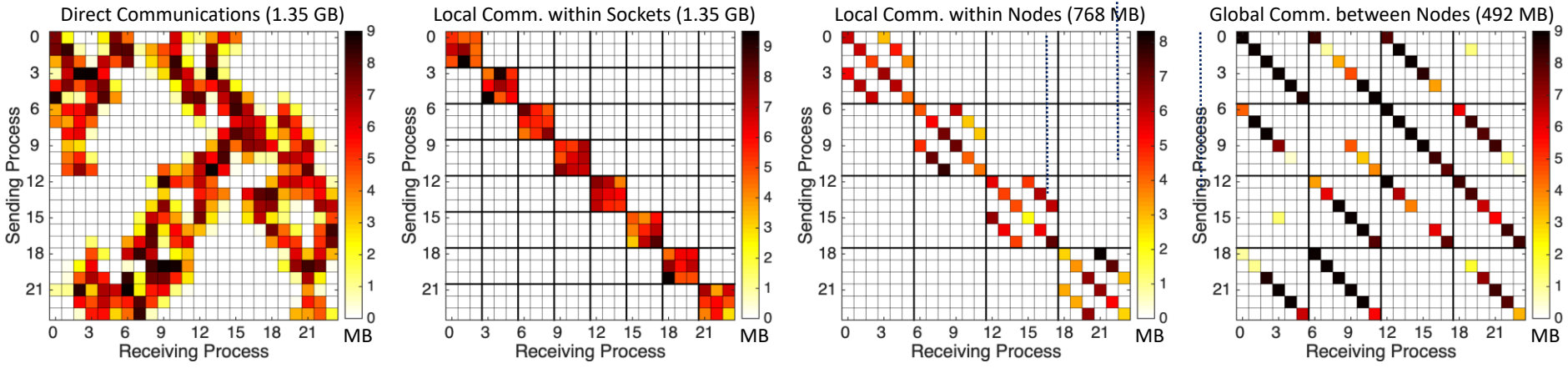
3D Image (Dense)

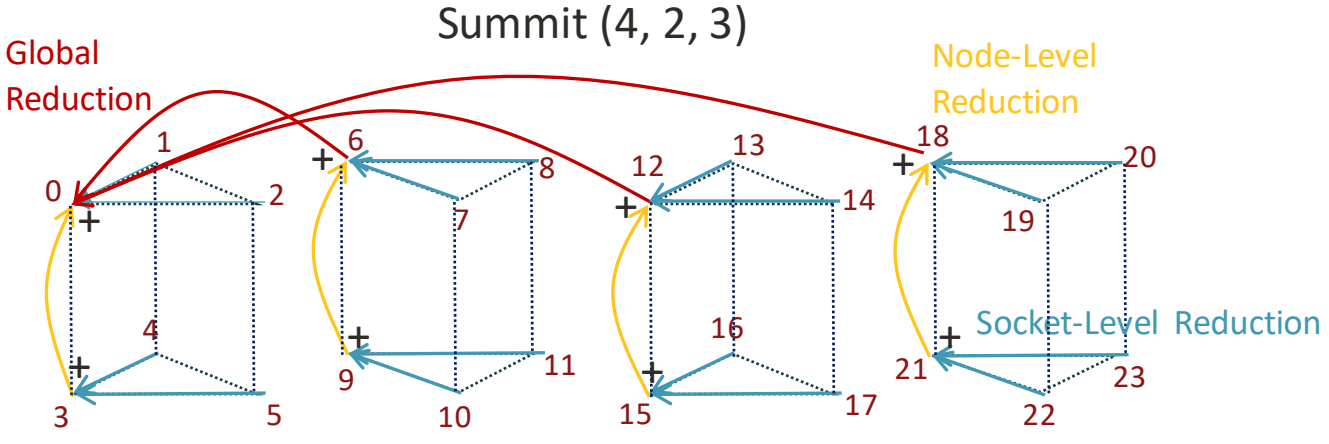
Measurement (Dense)



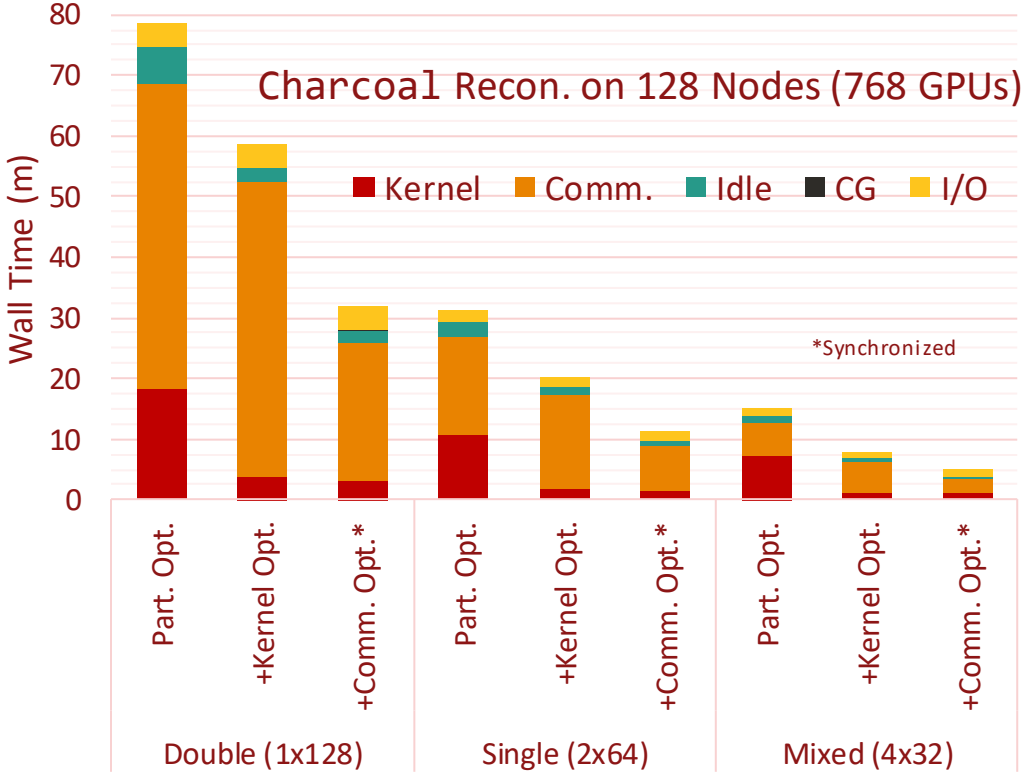
M. Hidayetoglu, T. Bicer, S. Garcia de Gonzalo, B. Ren, V. De Andrade, D. Gursoy, R. Kettimuthu, I. T. Foster, and W.-M. W. Hwu, "Petascale XCT: 3D image reconstruction with hierarchical communications on multi-GPU nodes," SC20, Best Paper.

Application Highlight: Image Reconstruction

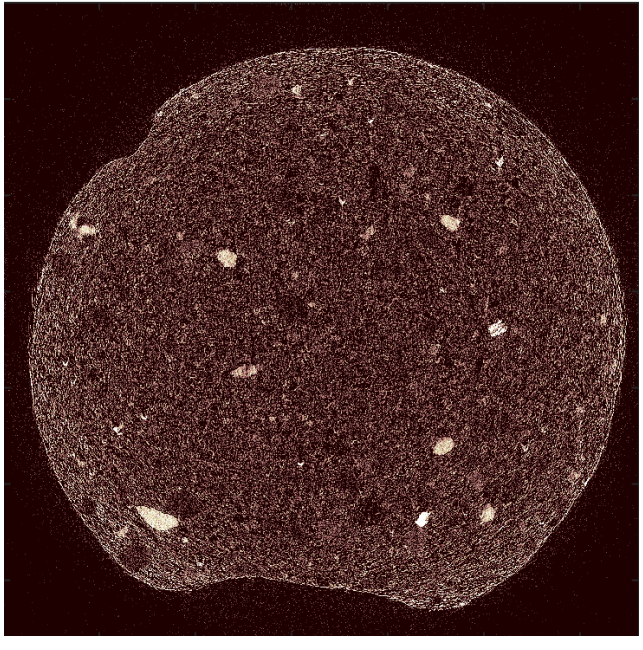




Application Highlight: Image Reconstruction

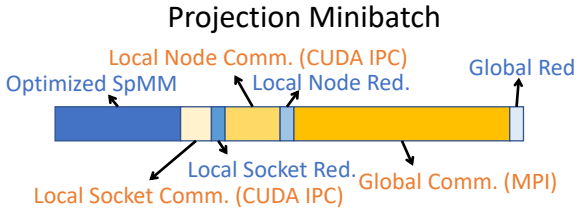


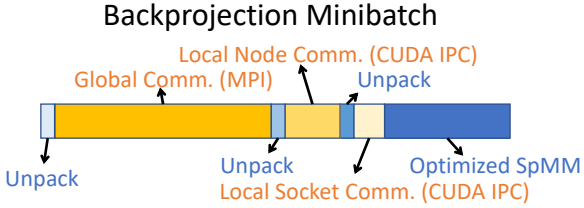
4198 x 6613 x 6613

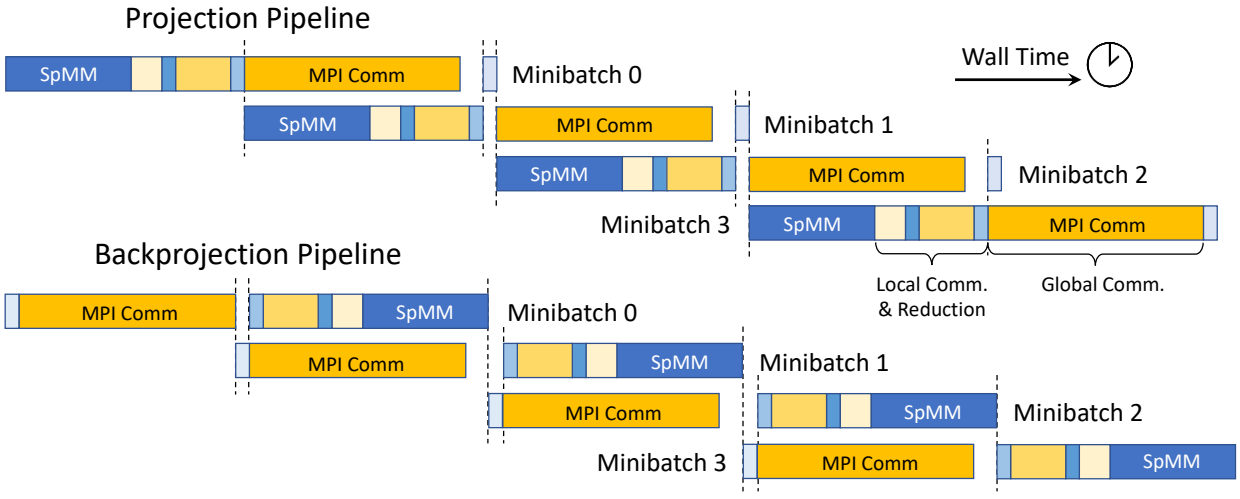


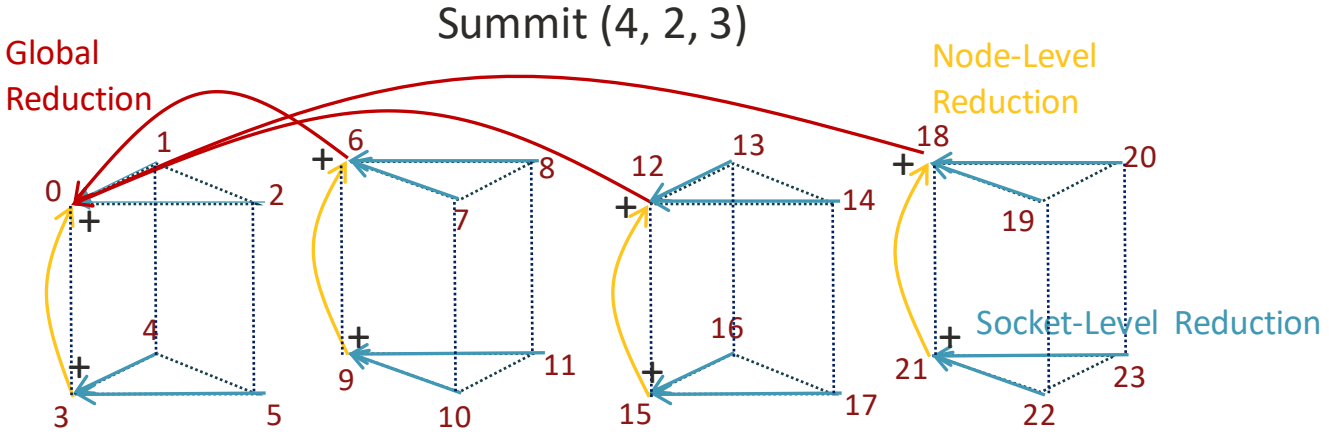
M. Hidayetoglu, T. Bicer, S. Garcia de Gonzalo, B. Ren, V. De Andrade, D. Gursoy, R. Kettimuthu, I. T. Foster, and W.-M. W. Hwu, "Petascale XCT: 3D image reconstruction with hierarchical communications on multi-GPU nodes," SC20, Best Paper.

Application Highlight: Image Reconstruction



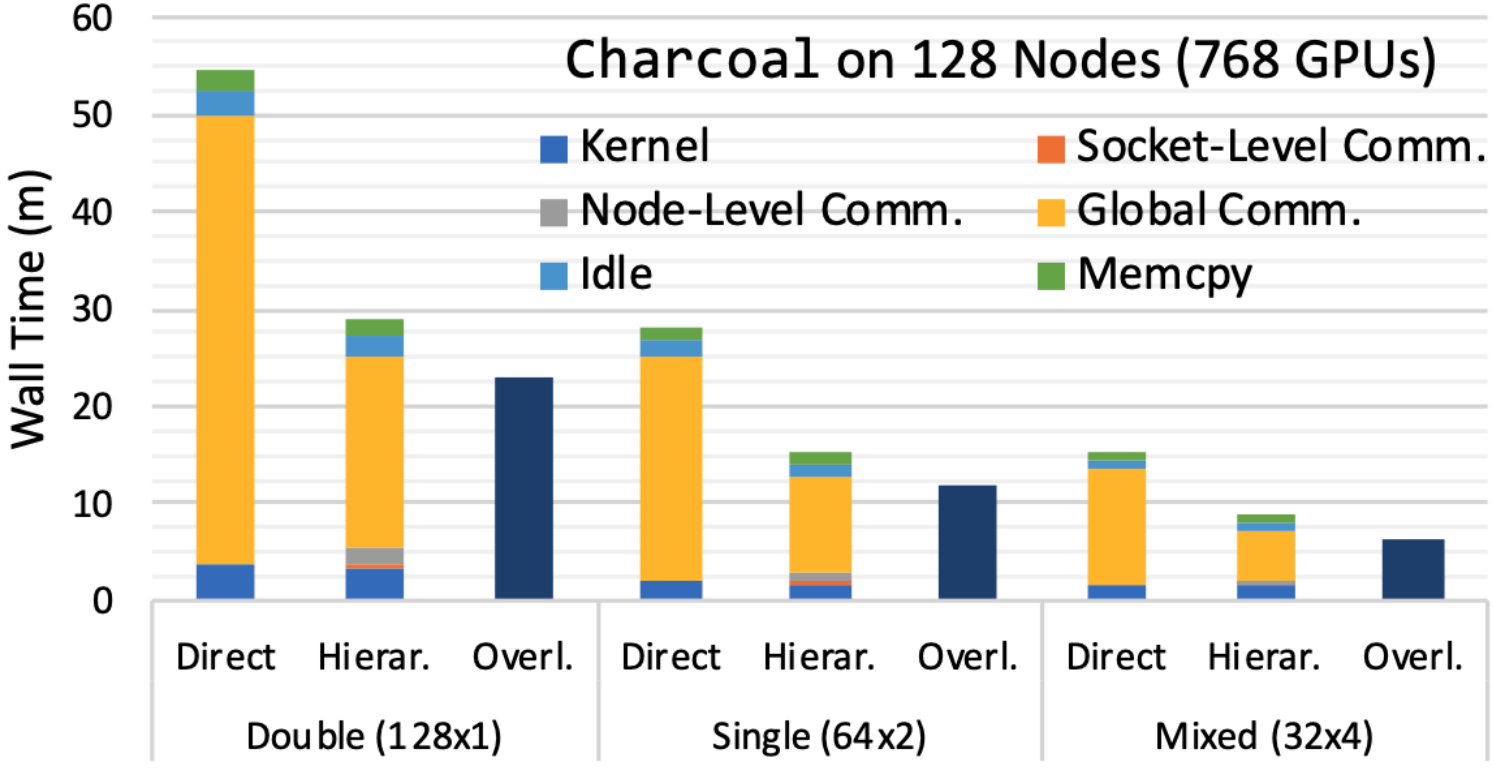






Application Highlight: Image Reconstruction

Hierarchical Communications Summary



Conclusion

Runtime Tools for Communications

1) CommBench: Configurable Benchmarking

2) HiCCL: Hierarchical Collective Communications

3) Application Highlight: 3D Image Reconstruction

Thank You