TACCL: Guiding Collective Algorithm Synthesis using Communication Sketches

Aashaka Shah[1], *Vijay Chidambaram*[1,2], *Meghan Cowan*[3], *Saeed Maleki*[3], *Madan Musuvathi*[3], *Todd Mytkowicz*[3], *Jacob Nelson*[3], *Olli Saarikivi*[3], *Rachee Singh*[4,5]

[1]UT Austin [2]VMWare Research [3]Microsoft Research [4]Microsoft [5]Cornell University

Email: aashaka@utexas.edu

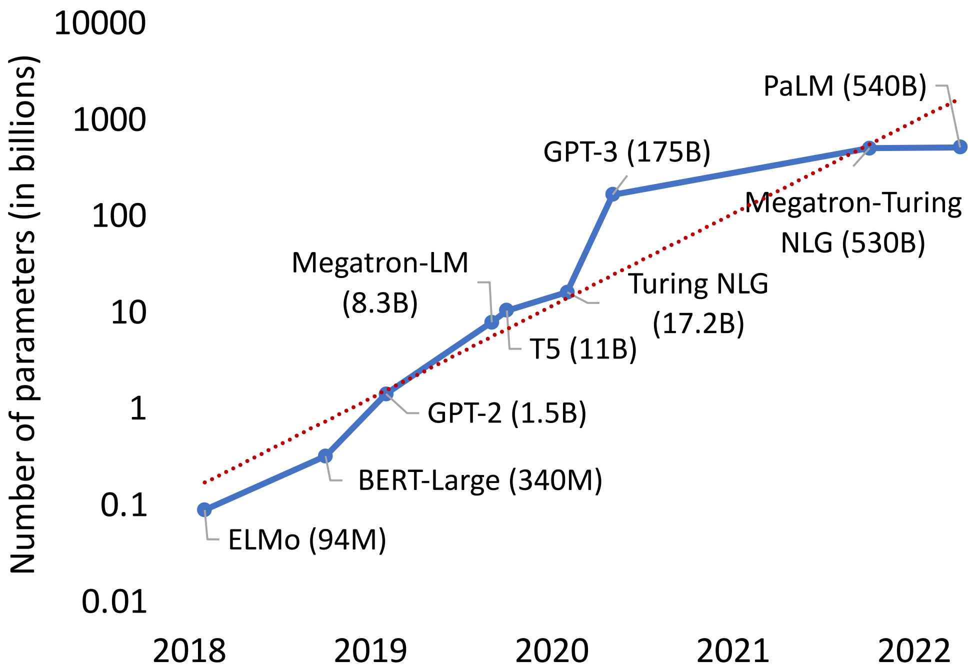
Distributed Deep Learning

Deep learning models are getting larger

- Distributed across various nodes/servers, each with multiple GPUs

Incurs network communication overhead

- GPUs can spend as much as 20% - 65% of time idle waiting on network communication

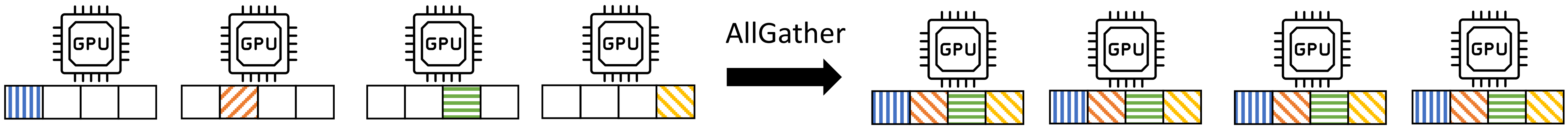


1. https://huggingface.co/blog/large-language-models
2. Sapio, Amedeo, et al. "Scaling distributed machine learning with in-network aggregation." NSDI'21

Communication in Distributed ML

- MPI-style collective communication used as abstractions for communication
  - Gather, shuffle, accumulate data
  - AllGather, AlltoAll, ReduceScatter, AllReduce

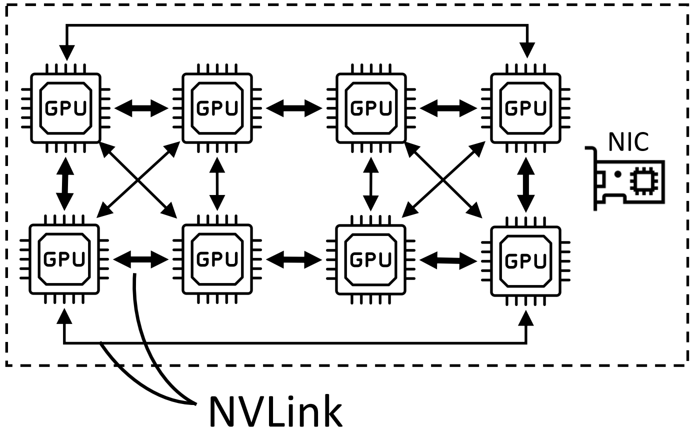
- Collective algorithm determines network utilization and speed of communication



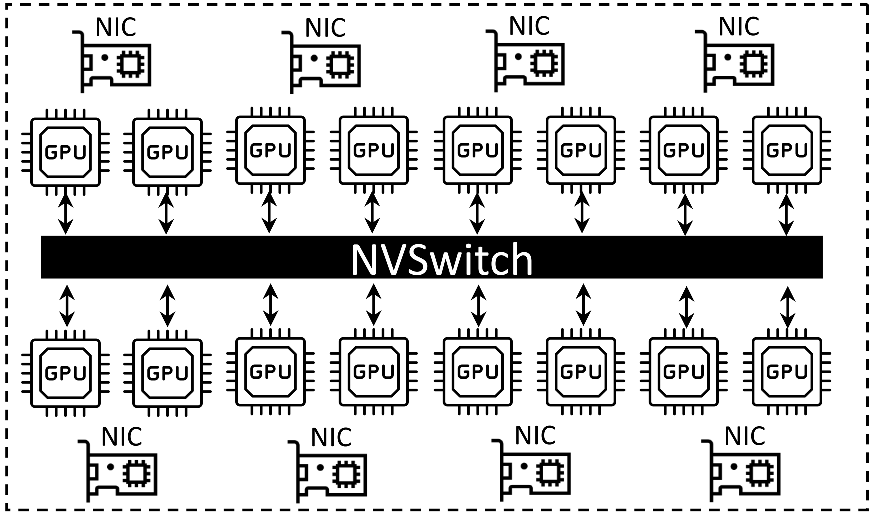
Challenges in building algorithms for collectives?

Wide variety of node topologies

Azure NDv2 node



NVIDIA DGX-2 node

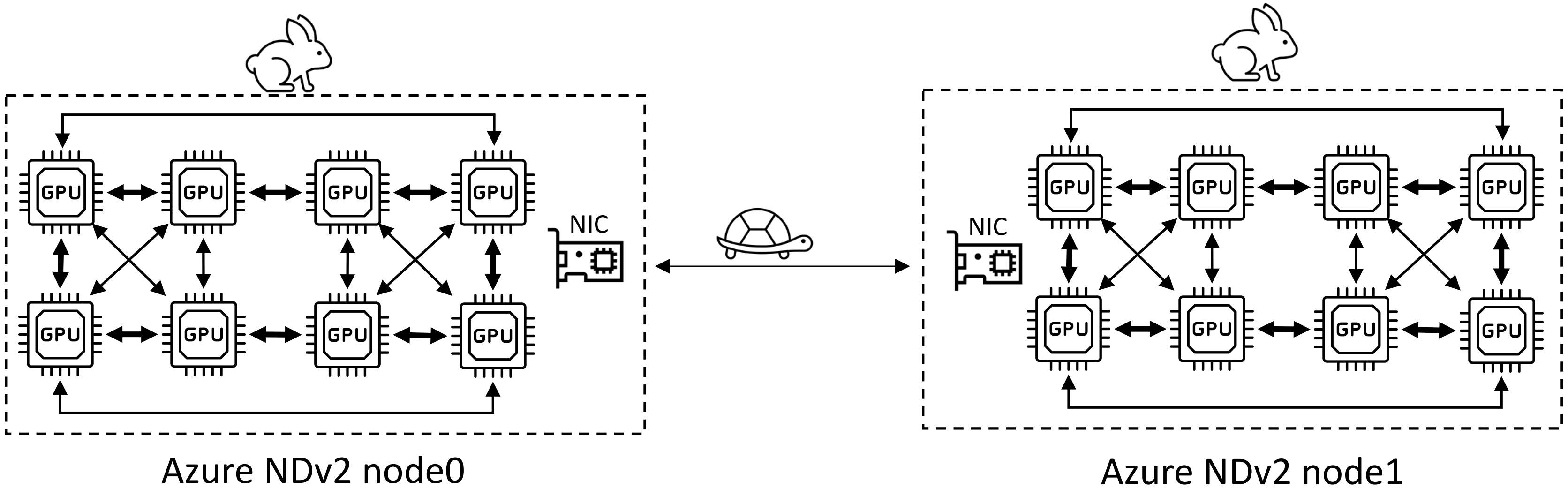


Number of GPUs (8, 16)	Number of NICs (shared/dedicated)	GPU interconnects (NVLink, NVSwitches)

⇒ Best collective algorithm could be different for different topologies

Challenges in building algorithms for collectives?

Hardware heterogeneity

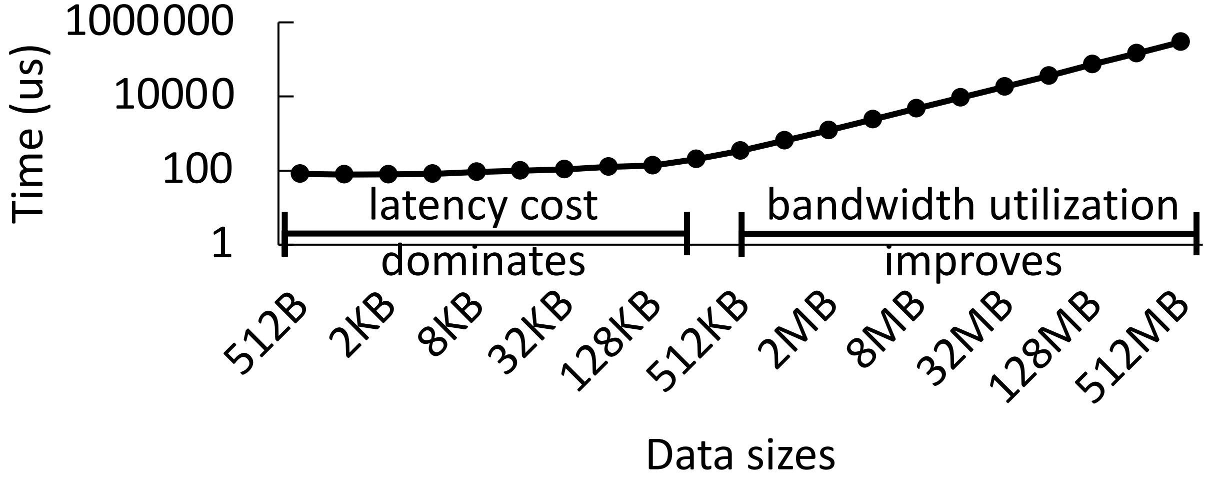


- Link connections are heterogeneous
  - Inter-node bandwidth < intra-node bandwidth
  - Inter-node latency > intra-node latency

⇒ Efficient collective algorithms need to be built keeping in mind link heterogeneity

Challenges in building algorithms for collectives?

Data size awareness



- Collective algorithms with many data transfer steps (like a state-of-the-art Ring algorithm) perform poorly for small data sizes

⇒ Efficient collective algorithms depends on size of data chunks to be transferred

Current state-of-the-art

NCCL (NVIDIA Collective Communication Library)

[Topology awareness] Generic algorithms like Ring or Tree mapped onto target topology

- − Not custom-built for a particular heterogeneous topology

[Data size awareness] Tuning to select algorithms (Ring, Tree) based on input size

- − Complicated
- − Not present for all collectives
- − Done using experiments that may not match reality

[Availability at scale]

- + Scales to multi-node topologies

1. https://github.com/NVIDIA/nccl/issues/457#issuecomment-771863198

Current state-of-the-art

Synthesis-based approaches (Blink, SCCL)

[Topology and data size awareness] Synthesize collective algorithm targeted to a particular topology

+ Maximize link utilization in heterogeneous topology (Blink)
+ Synthesize pareto-optimality in terms of latency and bandwidth (SCCL)

[Availability at scale] Synthesis is NP-hard

− Cannot scale synthesis to a multi-node topology

1. Wang, Guanhua, et al. "Blink: A fast NVLink-based collective communication library." *Proc. Conf. Syst. Mach. Learn*. 2018.
2. Cai, Zixian, et al. "Synthesizing optimal collective algorithms." *Proceedings of the 26th ACM SIGPLAN* 2021.

TACCL

Collective Communication Library that solves a set of mixed integer linear programming problems to synthesize collective algorithms

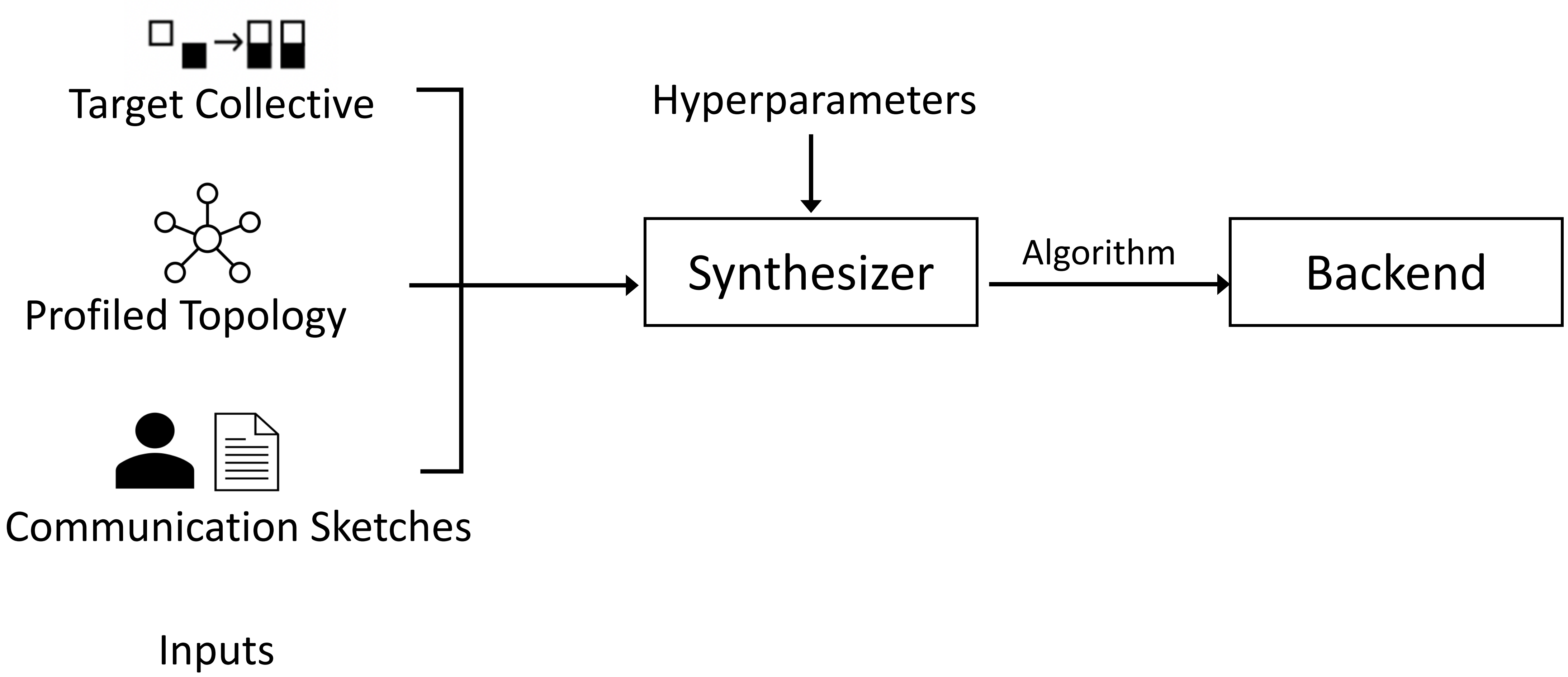
- Topology-aware
- Input-size aware

Scales to multi-node topologies

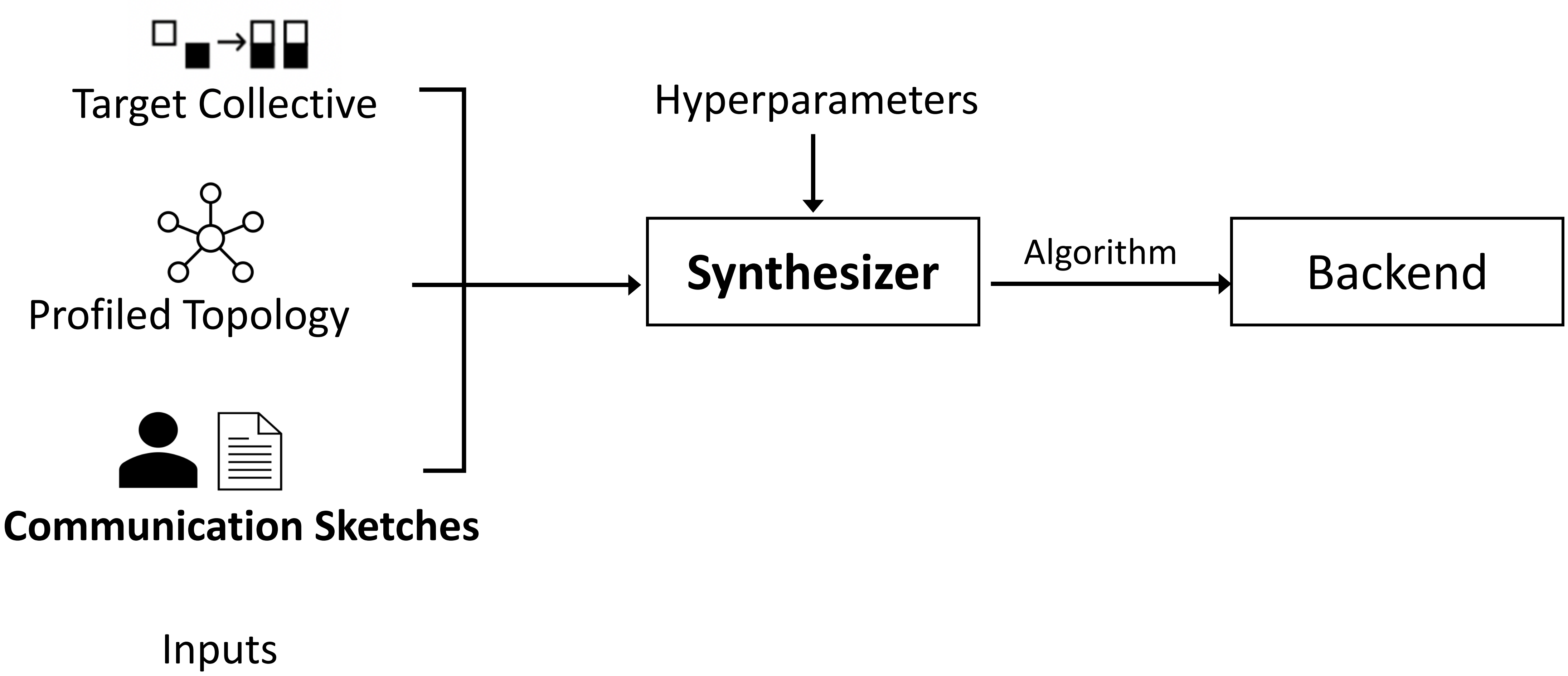
Drop-in replacement for NCCL

TACCL algorithms outperform NCCL by up-to 6.7x for evaluated topologies and provide up-to 2.4x end-to-end speedup for evaluated ML models

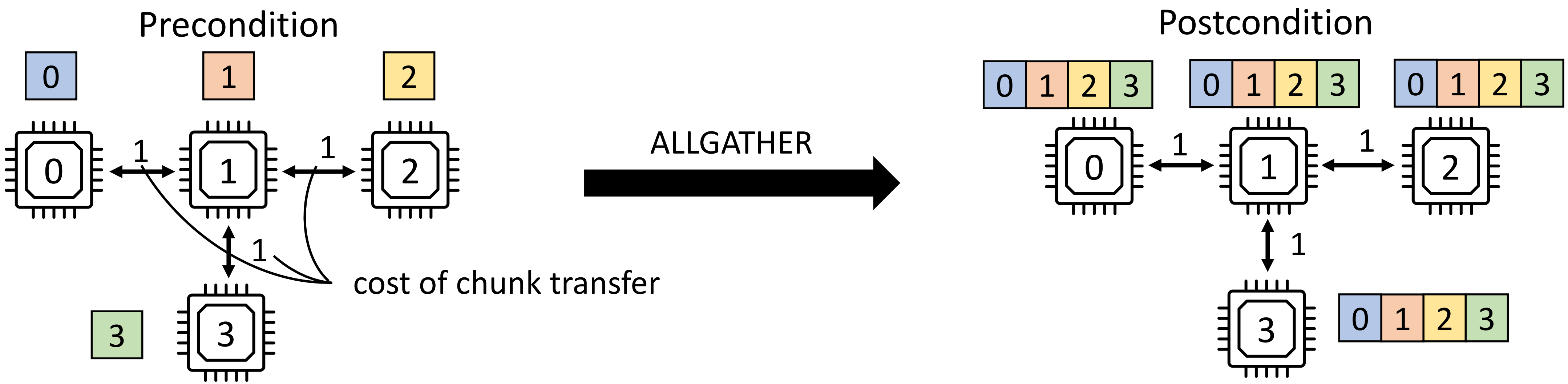
TACCL: Guiding Collective Algorithm Synthesis using Communication Sketches



TACCL: Guiding Collective Algorithm Synthesis using Communication Sketches



What does a synthesized collective algorithm look like?



Link	Chunk sent?	Send order	Send time
(0,1)	0	0	[0]
(1,0)	1 2 3	1 > 2 = 3	[0, 1, 1]
(1,2)	0 1 3	1 > 0 > 3	[1, 0, 2]
(2,1)	2	2	[0]
(1,3)	0 1 2	1 > 0 > 2	[1,0,2]
(3,1)	3	3	[0]

L links, C chunks (NP-hard) ☹

For each link:

1) Will a data chunk be sent across it? $O(2^{(C \times L)})$
2) How will chunks be ordered wrt each other? $O(2^{(C \times C \times L)})$

Communication Sketches

- ML engineer provides *Communication Sketches*
- Specify intuitive parts of the algorithm
- Do not require a lot of domain knowledge to write
- Guide algorithm synthesis

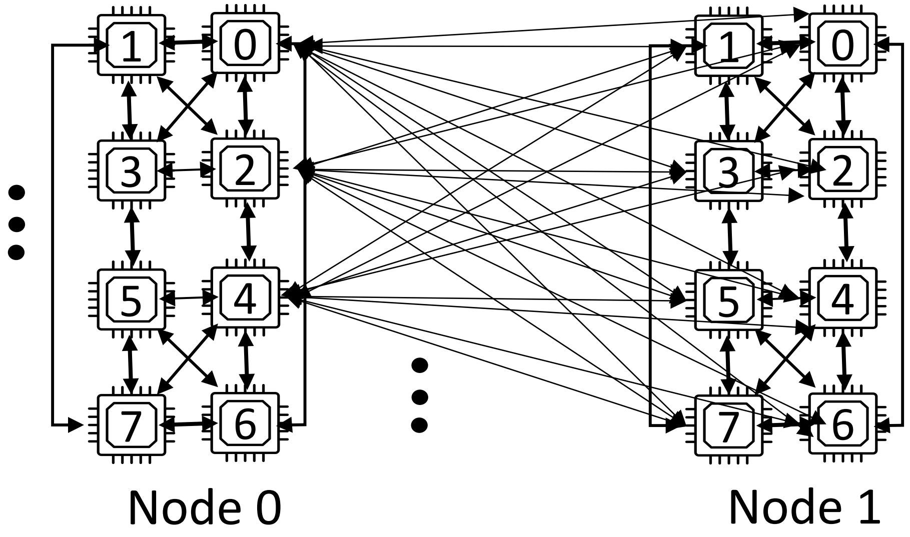
Chunks

Links	0	1	2	3
(0,1)	Y/N?	Y/N?	Y/N?	Y/N?
(1,0)	Y/N?	Y/N?	Y/N?	Y/N?
(1,2)	Y/N?	Y/N?	Y/N?	Y/N?
(2,1)	Y/N?	Y/N?	Y/N?	Y/N?
(1,3)	Y/N?	Y/N?	Y/N?	Y/N?
(3,1)	Y/N?	Y/N?	Y/N?	Y/N?

Sketch →

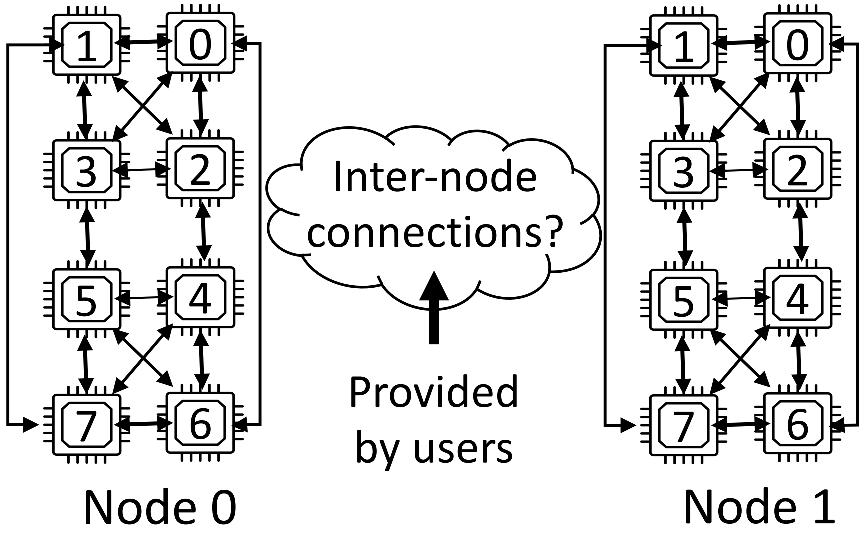
	0	1	2	3
(0,1)	-	-	Y/N?	Y/N?
(1,0)	-	Y/N?	Y/N?	Y/N?
(1,2)	Y/N?	Y/N?	-	-
(2,1)	-	-	Y/N?	Y/N?
(1,3)	Y/N?	Y/N?	-	-
(3,1)	-	-	Y/N?	Y/N?

1) Sketching Logical Topology



1) Sketching Logical Topology

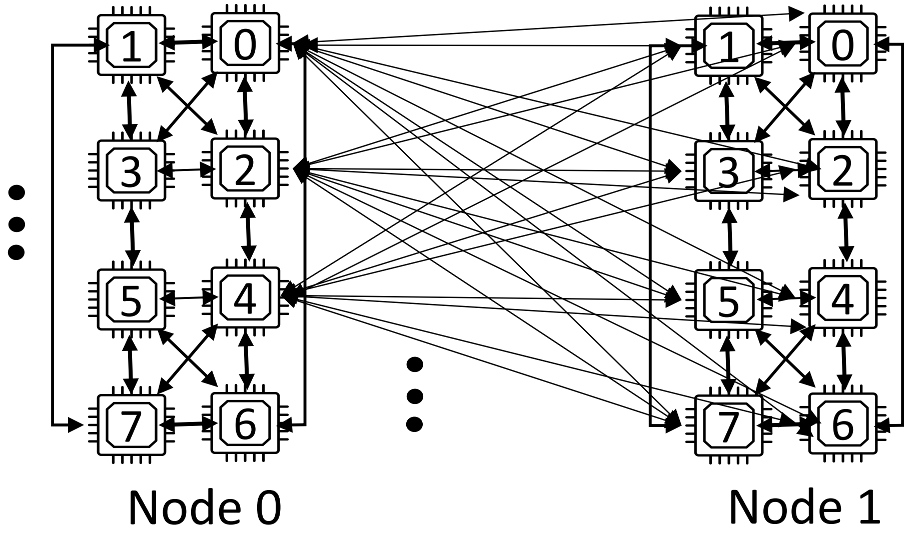
Allow users to select/deselect links between nodes

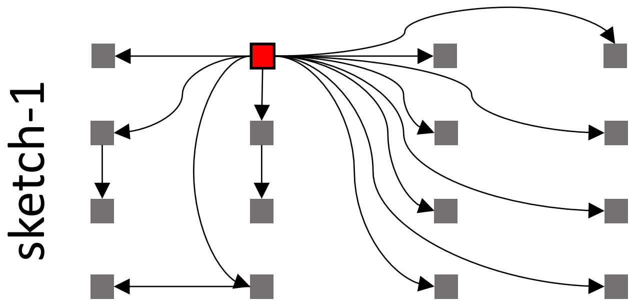


1) Sketching Logical Topology

Allow users to select/deselect links between nodes

Sketch-1: [0,1,2,3,4,5,6,7] → [0,1,2,3,4,5,6,7]





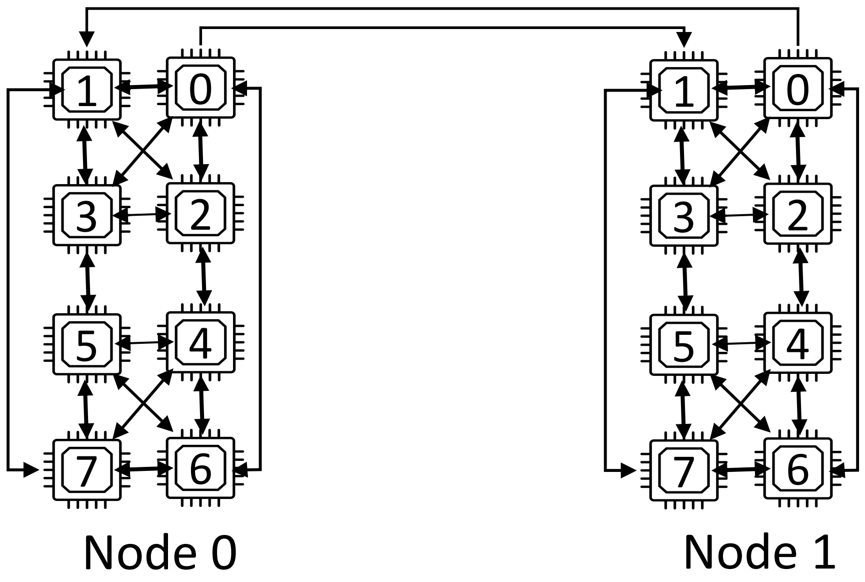
2-node Azure NDv2, AlltoAll

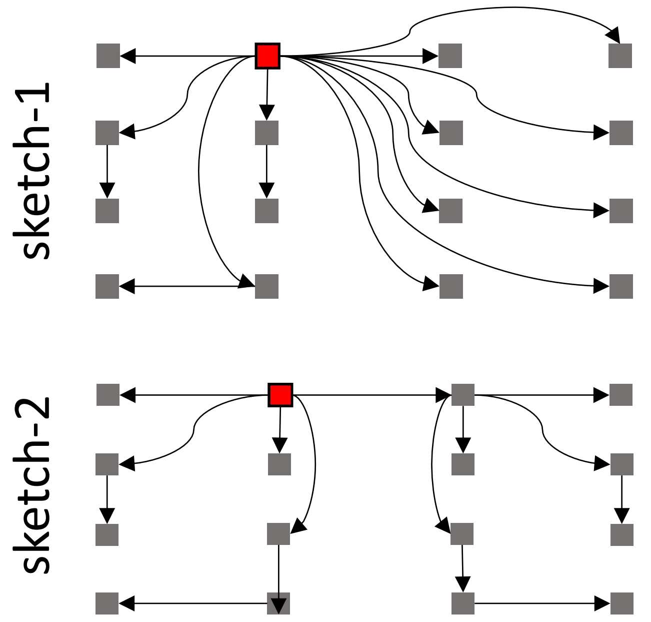
1) Sketching Logical Topology

Allow users to select/deselect links between nodes

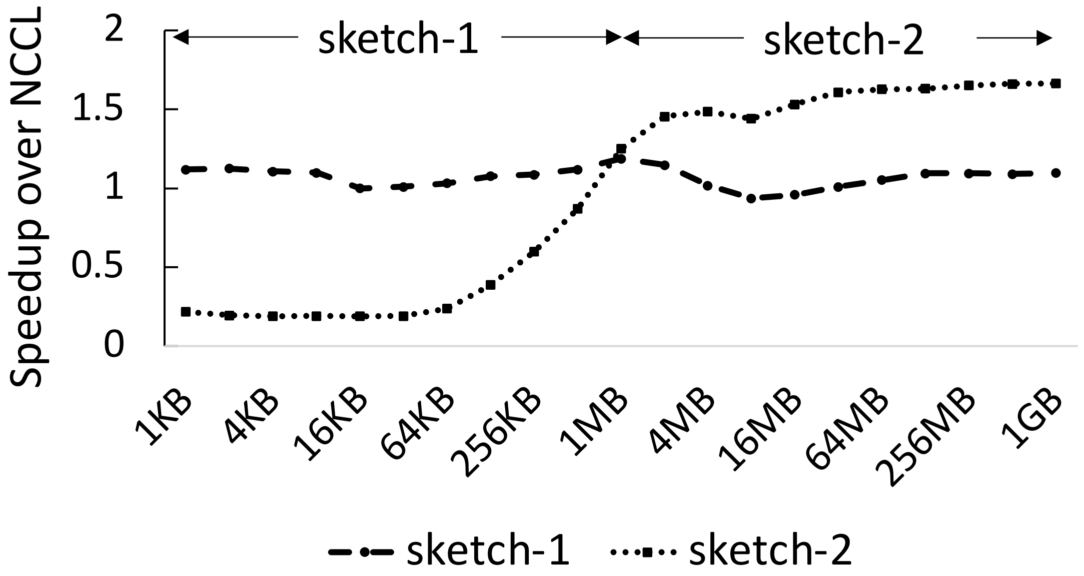
Sketch-1: [0,1,2,3,4,5,6,7] → [0,1,2,3,4,5,6,7]

Sketch-2: 0 → 1





2-node Azure NDv2, AlltoAll



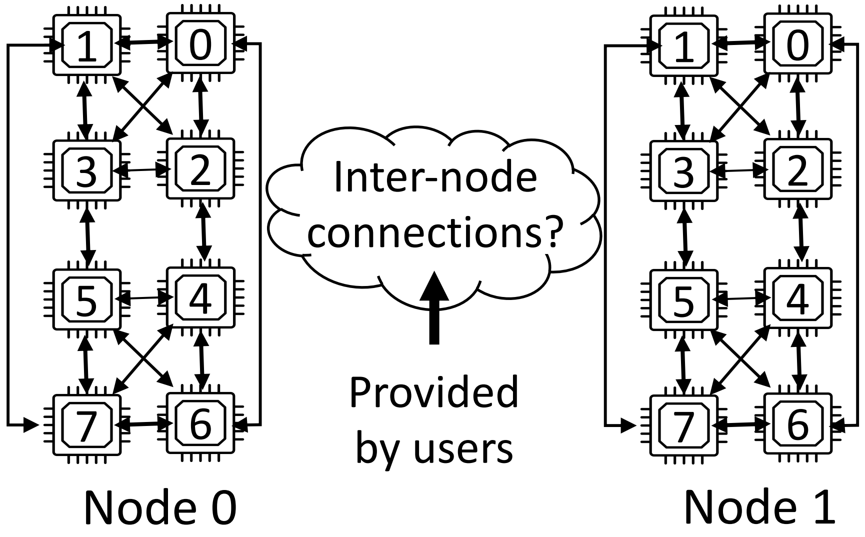
Higher is better

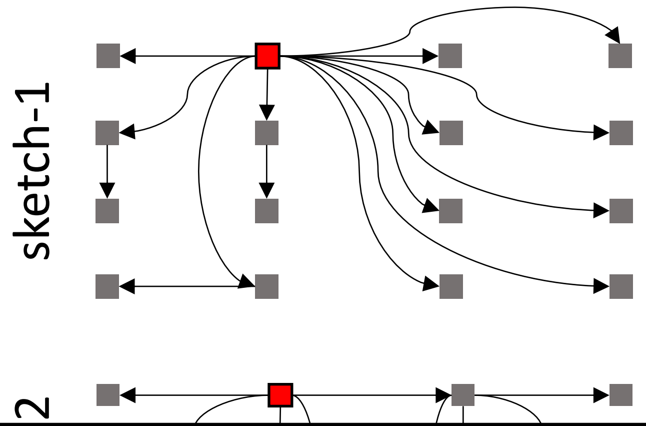
1) Sketching Logical Topology

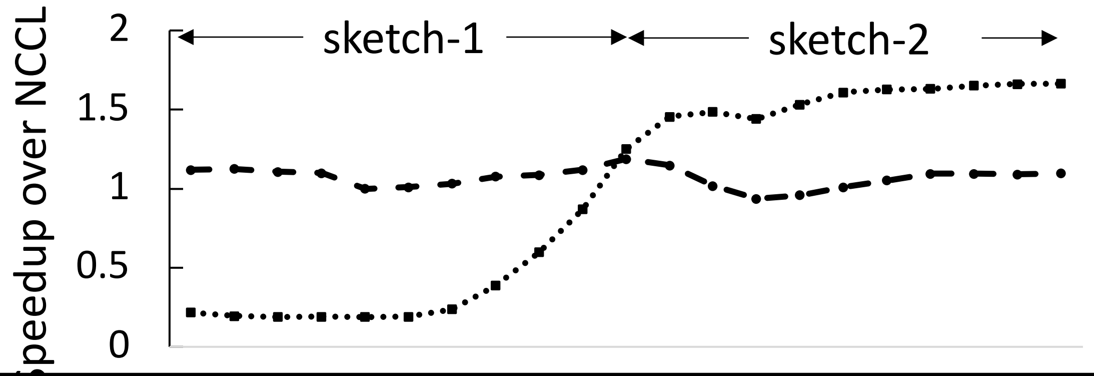
Allow users to select/deselect links between nodes

Sketch-1: [0,1,2,3,4,5,6,7] → [0,1,2,3,4,5,6,7]

Sketch-2: 0 → 1





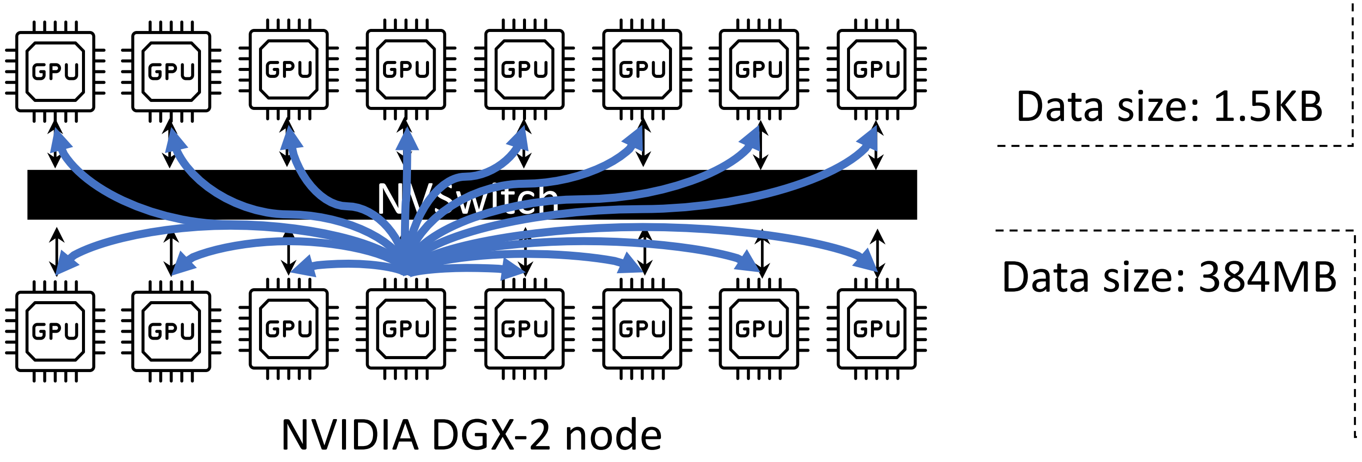


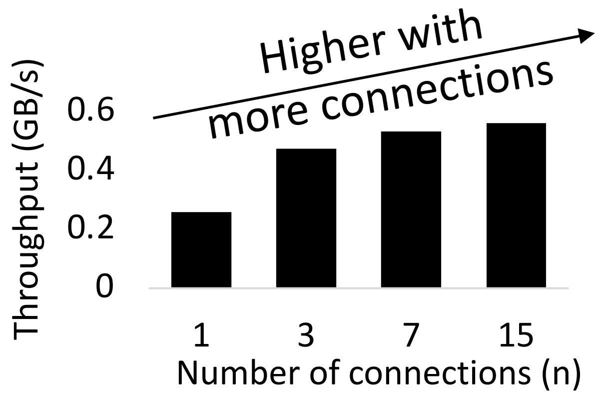
Reduces number of links to make decisions about
Can extract high performance over a range of input sizes

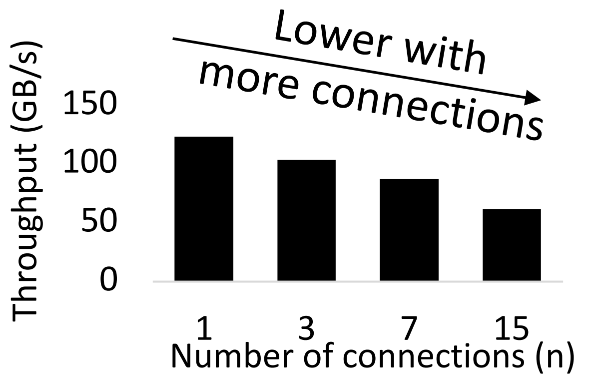
2-node Azure NDv2, AlltoAll

2) Sketching NVSwitch connections

Allow users to maximize or minimize unique connections over NVSwitches

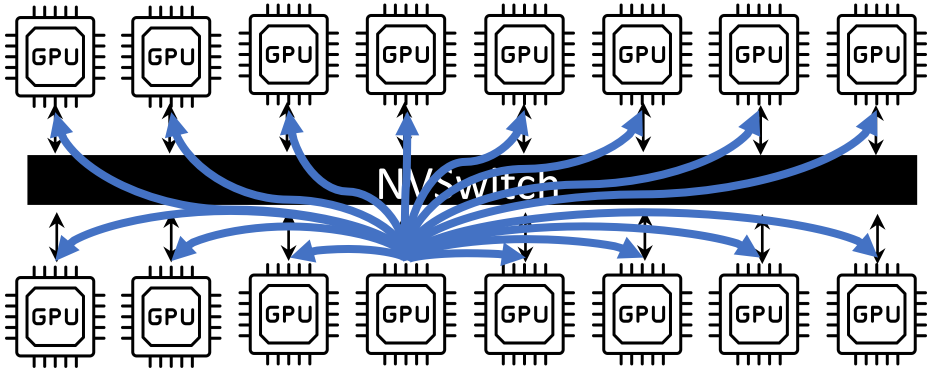




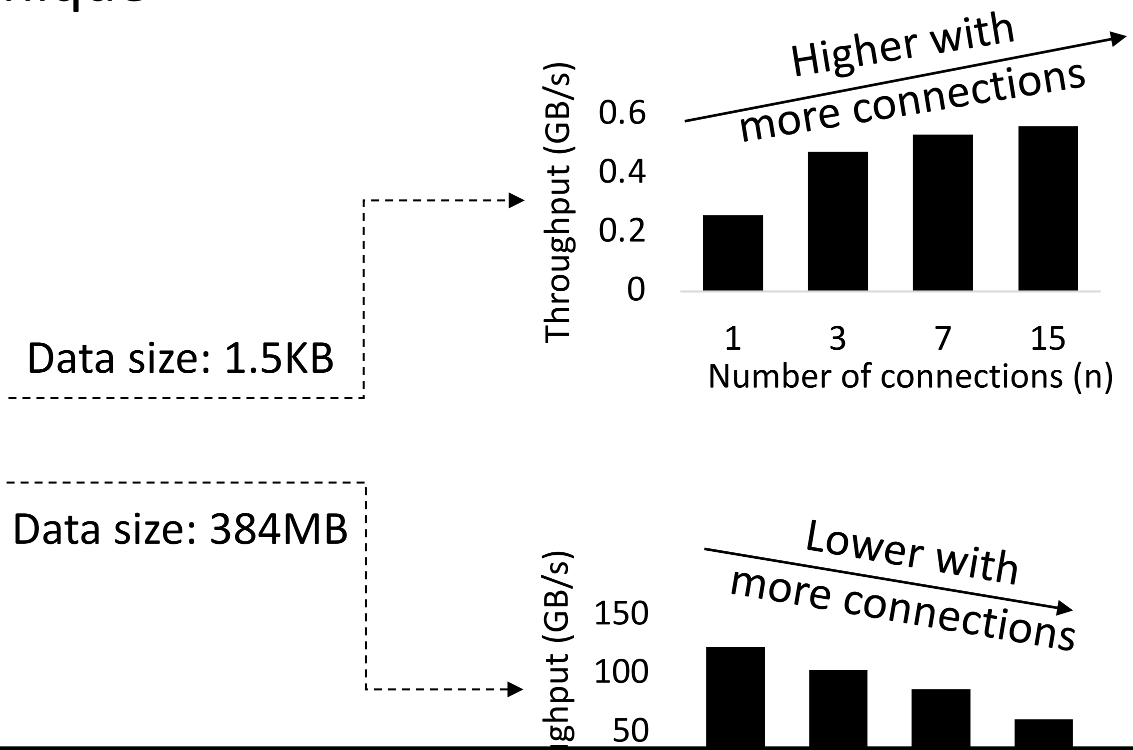


2) Sketching NVSwitch connections

Allow users to maximize or minimize unique connections over NVSwitches

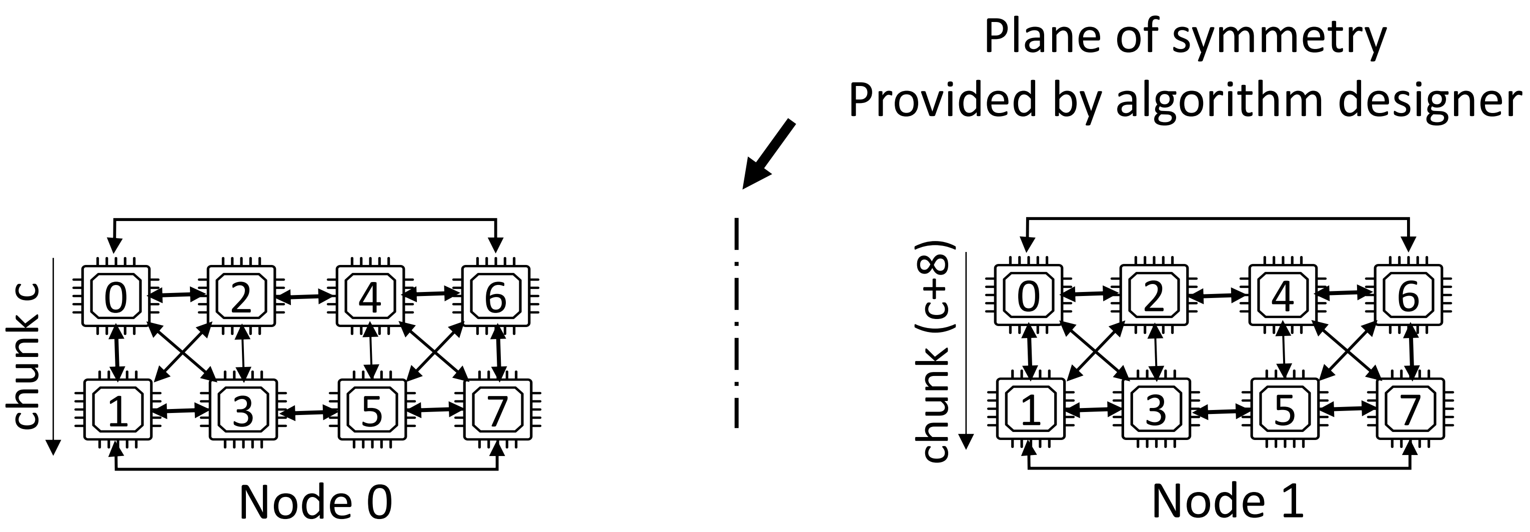


NVIDIA DGX-2 node



Used to guide algorithm synthesis to be performant for a particular range of input sizes

3) Sketching for symmetry

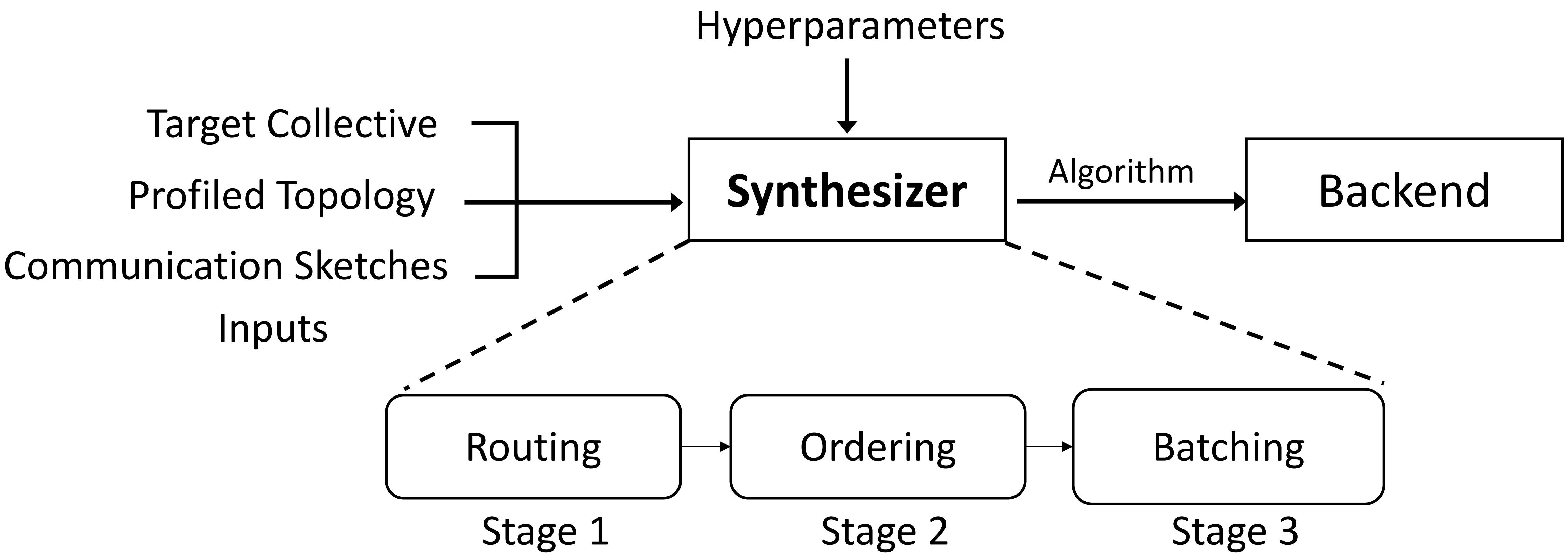


- Designer can annotate symmetry planes around which data transfers will be fixed to be rotationally symmetric

Reduces number of transfers to make decisions about

Generating the algorithm

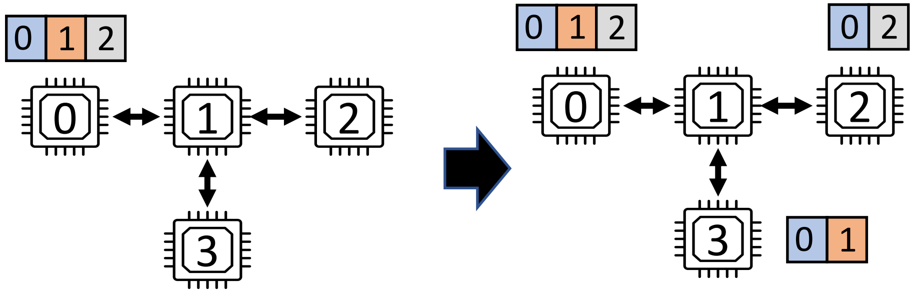
Stage-wise synthesis simplifies the problem!



TACCL Synthesizer

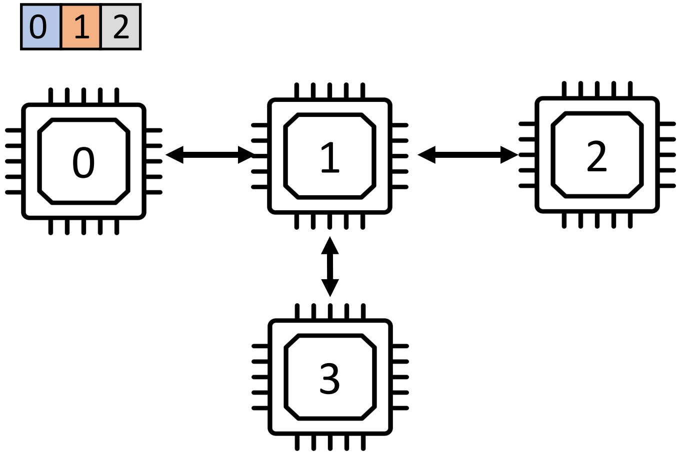
Routing → Ordering → Batching

Stage 1 | Stage 2 | Stage 3

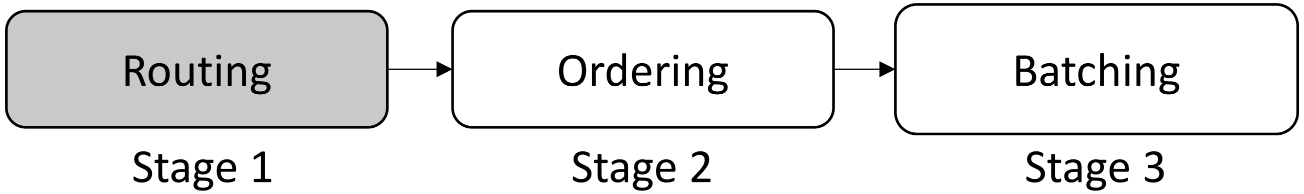


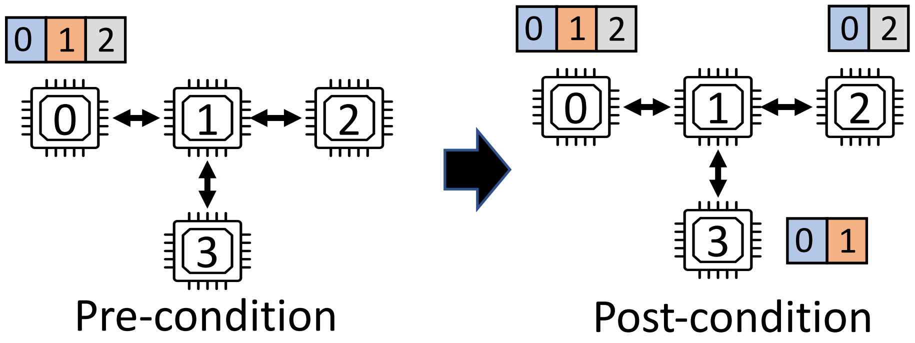
Pre-condition | Post-condition

Example collective

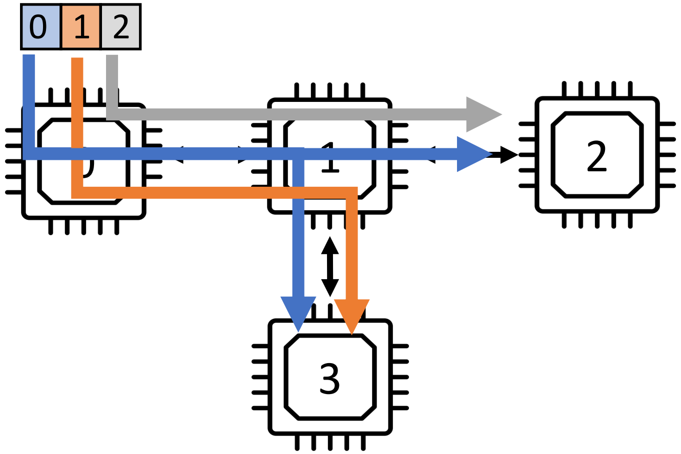


TACCL Synthesizer



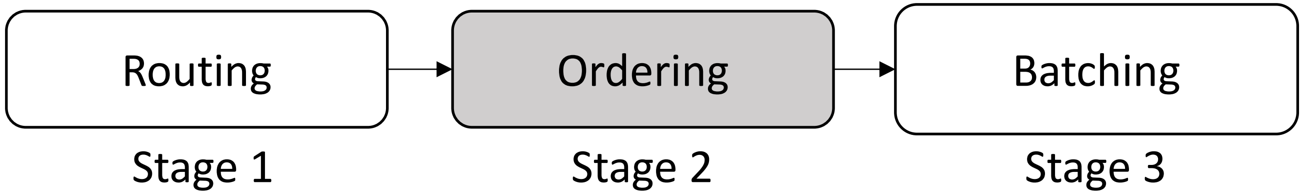


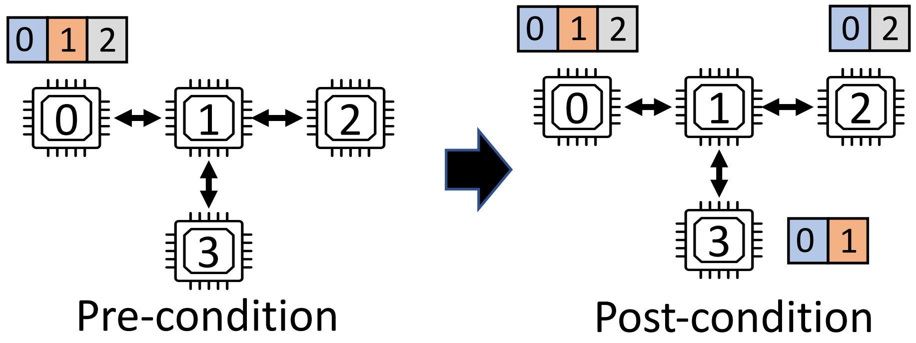
Example collective



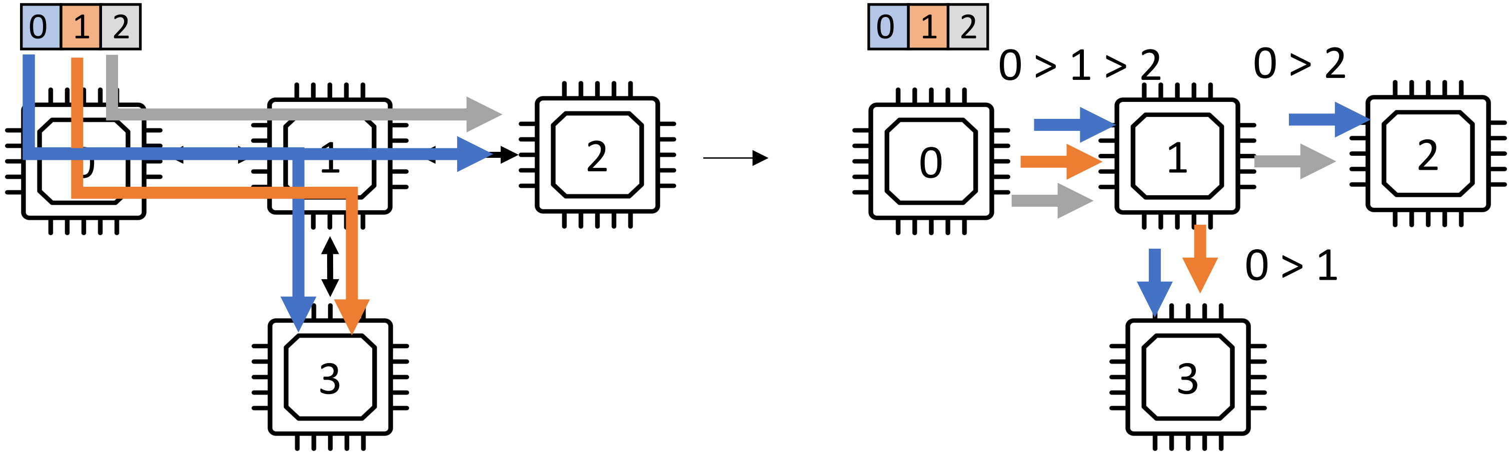
Determine the path that each data chunk will take
(Ordering between data chunks not decided yet)
Solve an ILP to obtain optimal paths (based on congestion and dilation metrics)

TACCL Synthesizer





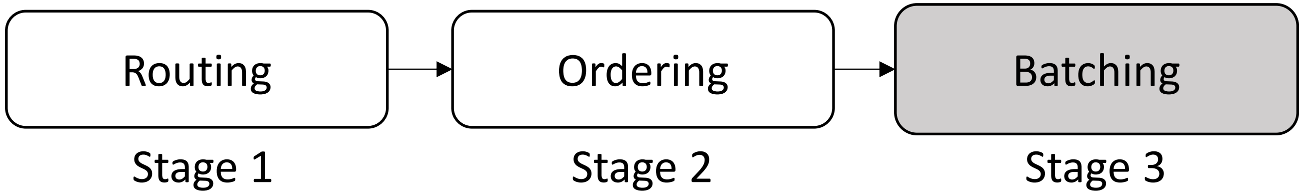
Example collective

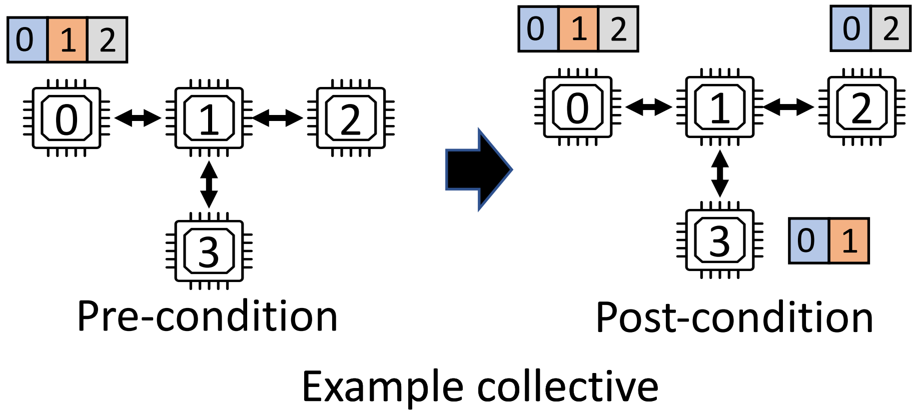


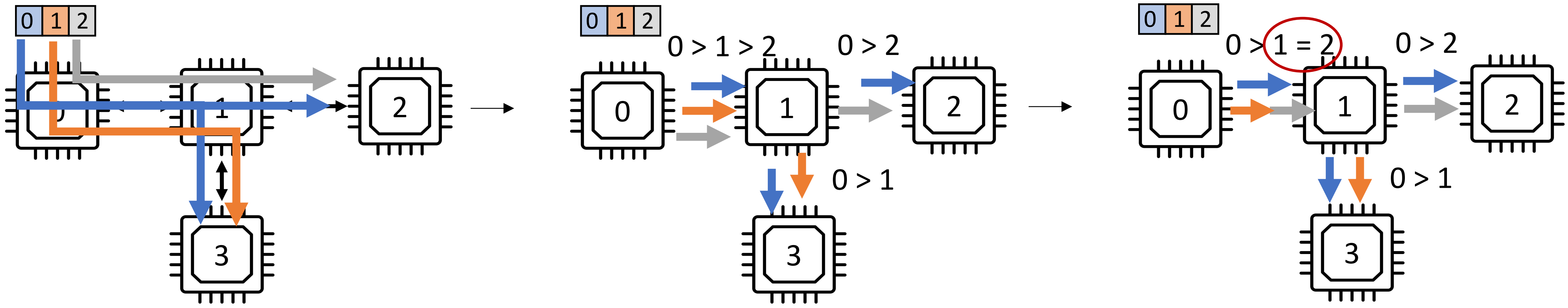
Order chunks sent over the same link

Ordering is done using heuristics, e.g., by giving more preference to chunks that need to travel longer distance

TACCL Synthesizer

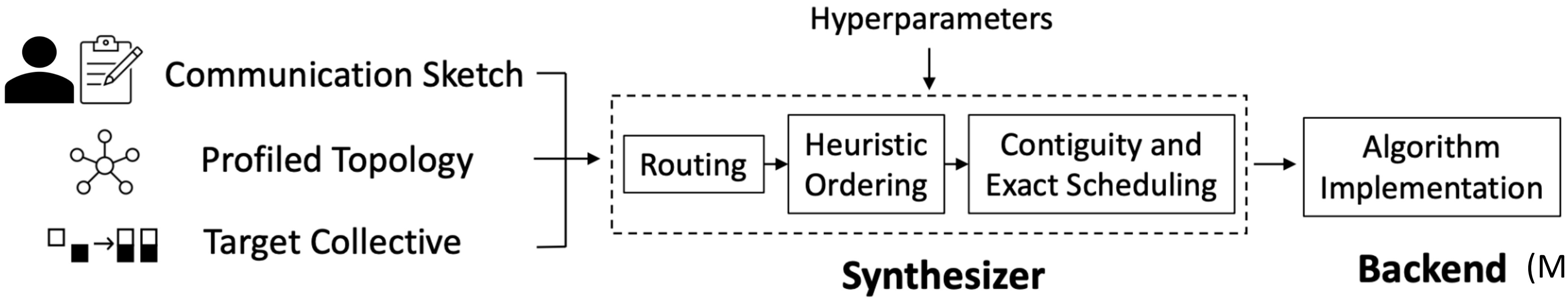






Batches data chunks to send them together over links in order to reduce link latency costs
Solves an ILP to optimize between reduced latency cost v/s possible pipelining gaps

TACCL: Guiding Collective Algorithm Synthesis using Communication Sketches



(MSCCL)

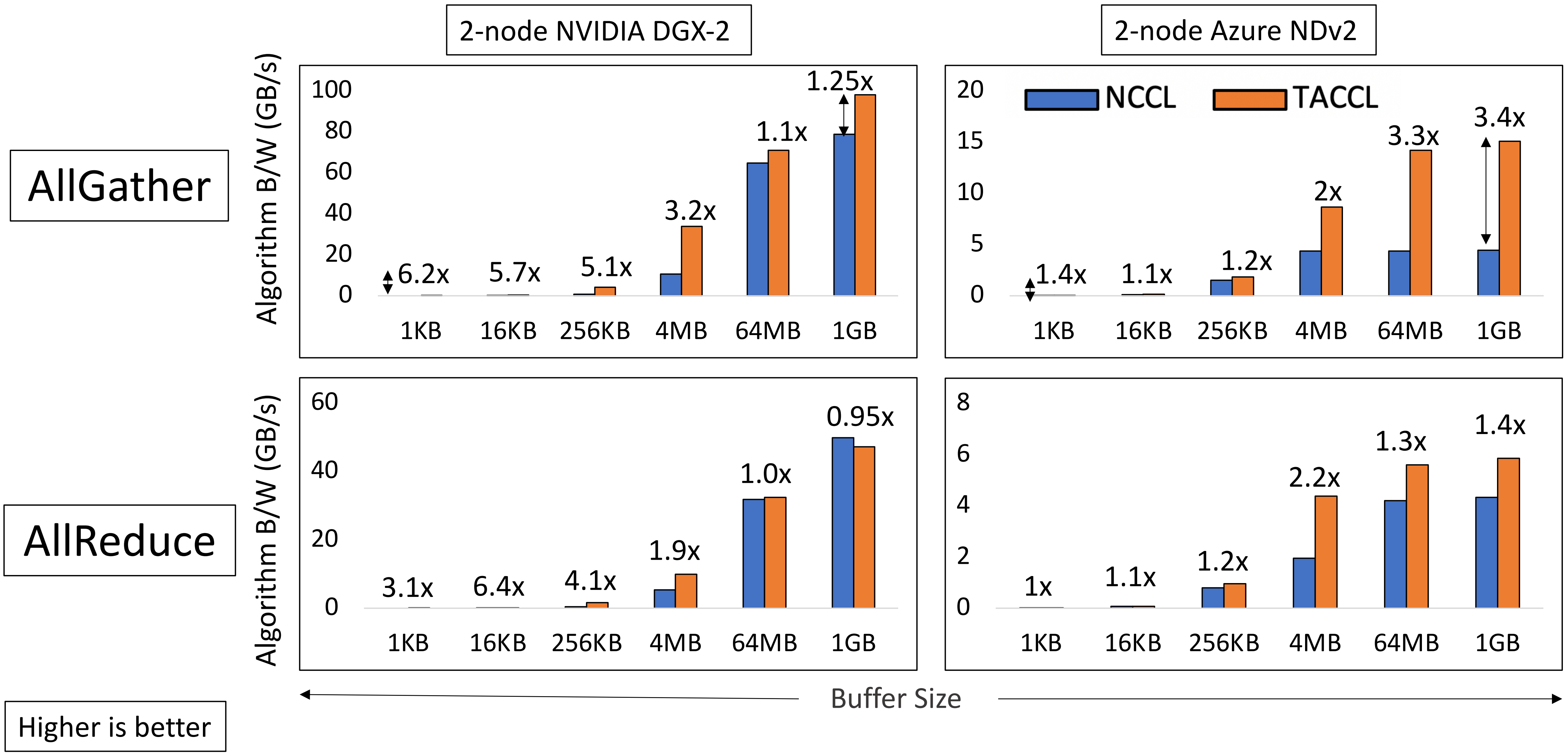
(Extends NCCL to implement synthesized collectives)

https://github.com/microsoft/msccl/

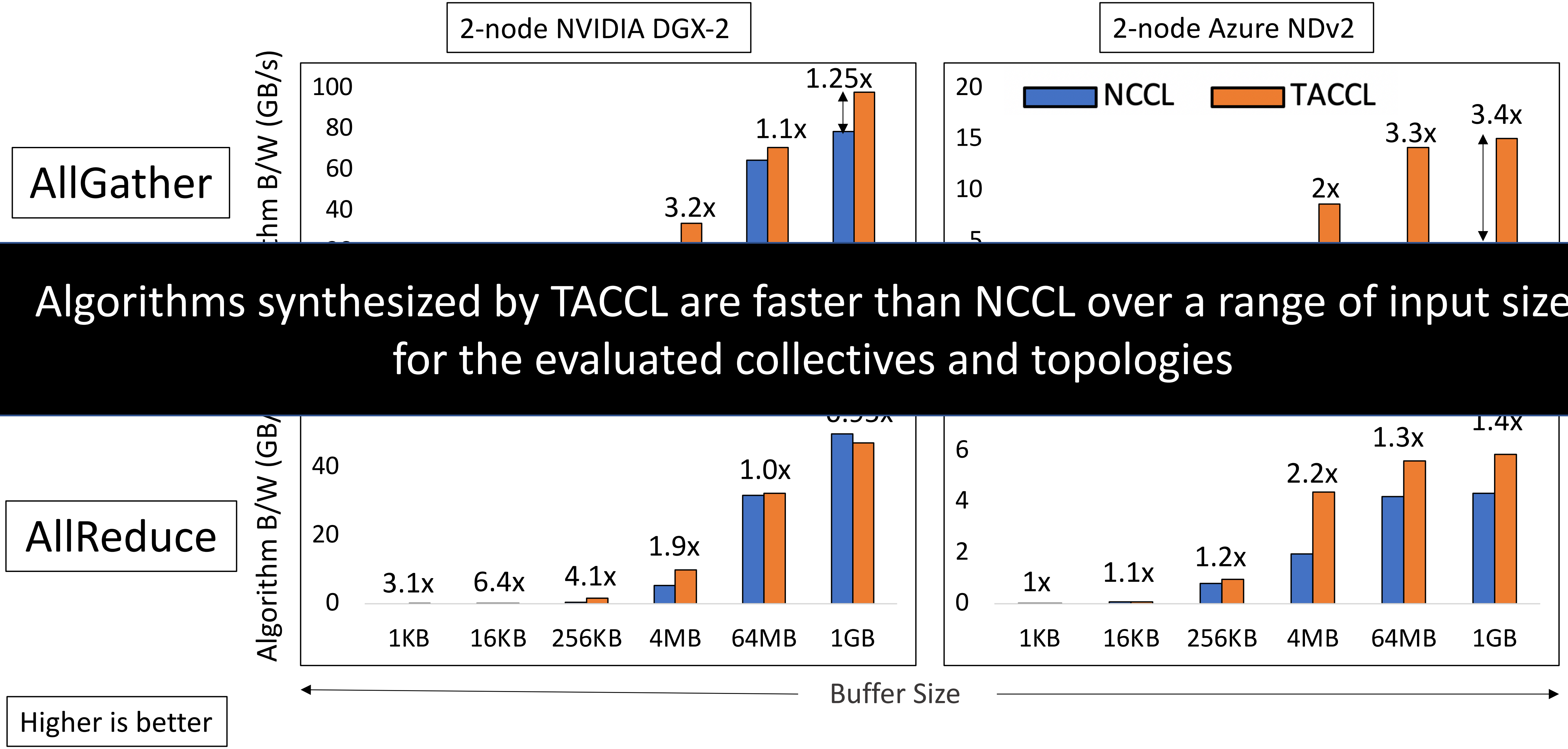
Evaluation

- Compare performance against NCCL (v2.8.4)
- Collectives: AllGather, AllReduce, AlltoAll
- Topologies:
  - 2-node NVIDIA DGX-2 (32 GPUs)
  - 2-node & 4-node Azure NDv2 (16 GPUs & 32 GPUs)
- Distributed ML models:
  - Transformer-XL, BERT (PyTorch implementation)

How do TACCL algorithms perform against NCCL?

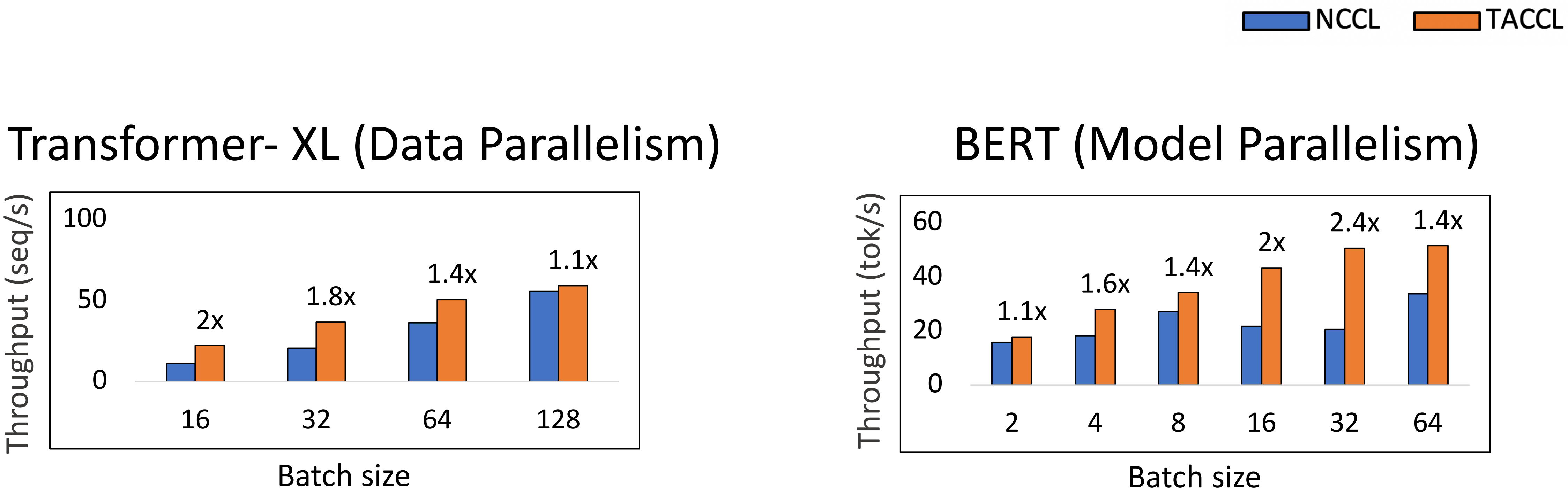


How do TACCL algorithms perform against NCCL?



Algorithms synthesized by TACCL are faster than NCCL over a range of input sizes for the evaluated collectives and topologies

Do we see speedups in end-to-end model training?

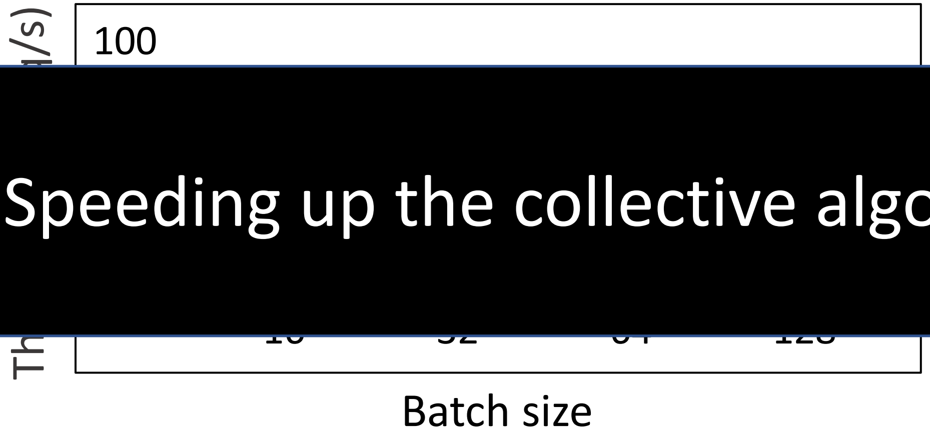


Microsoft-internal Mixture-of-Experts workload: 17% speedup

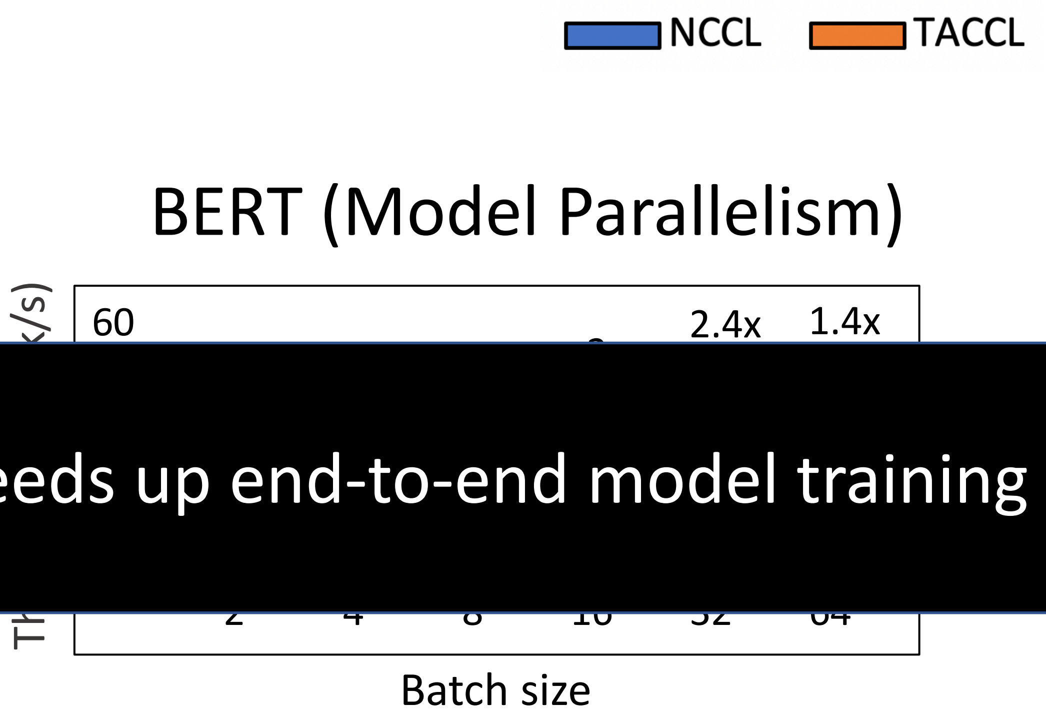
Higher is better

Do we see speedups in end-to-end model training?

Transformer- XL (Data Parallelism)



BERT (Model Parallelism)



Speeding up the collective algorithm speeds up end-to-end model training

Microsoft-internal Mixture-of-Experts workload: 17% speedup

Higher is better

Conclusion

TACCL is a tool to synthesize efficient algorithms for collectives

- Guided using intuitive communication sketches
- Solved using novel 3-stage synthesizer

Will soon be available at https://github.com/microsoft/taccl

Email: aashaka@utexas.edu