deepspeed

Microsoft

ZeRO++: Extremely Efficient Collective Communication for Large Model Training

Guanhua Wang

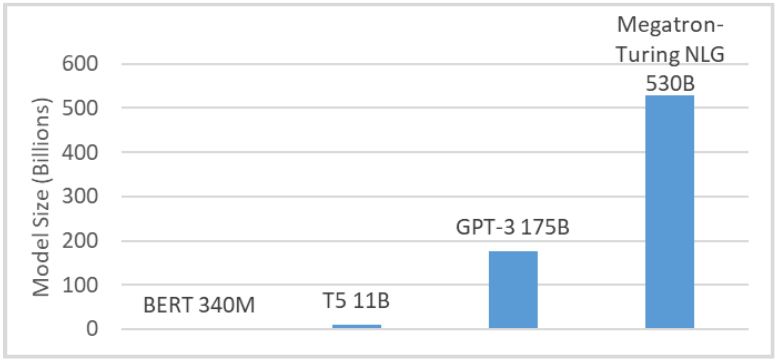
Heyang Qin, Sam Ade Jacobs, Xiaoxia Wu, Connor Holmes, Zhewei Yao, Samyam Rajbhandari, Olatunji Ruwase, Feng Yan, Lei Yang, Yuxiong He

DeepSpeed @ Microsoft

12/16/2023

Motivation

Model size grows exponentially

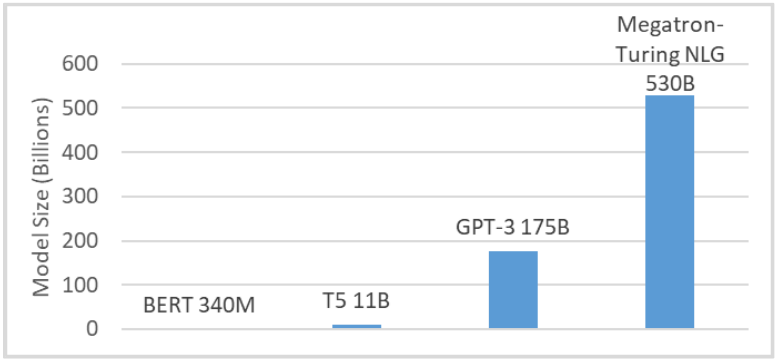


More GPU & More Parallelism

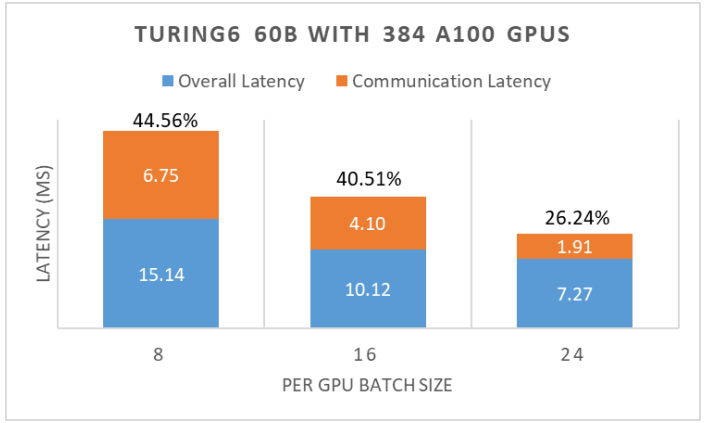
- Max global batch size is fixed
- More GPU => Smaller batch size per GPU

Motivation

Model size grows exponentially



Problem1: Communication is huge overhead in Small batch training

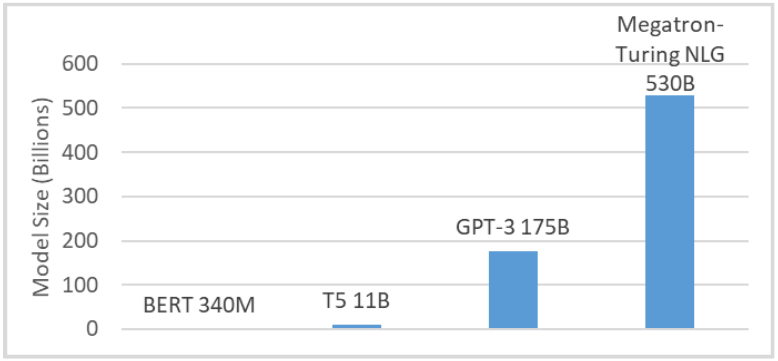


More GPU & More Parallelism

- Max global batch size is fixed
- More GPU => Smaller batch size per GPU

Motivation

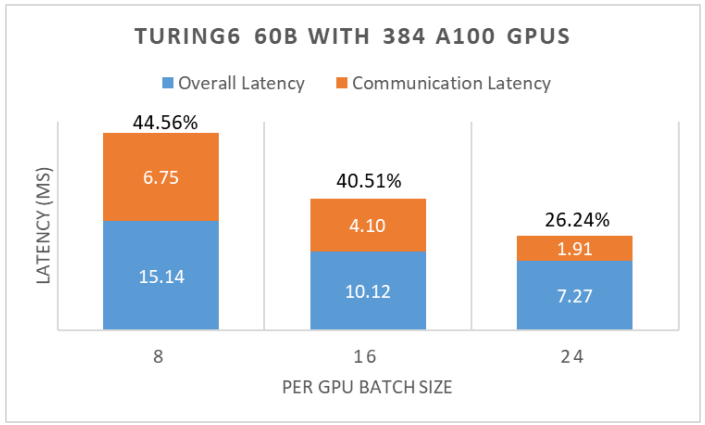
Model size grows exponentially



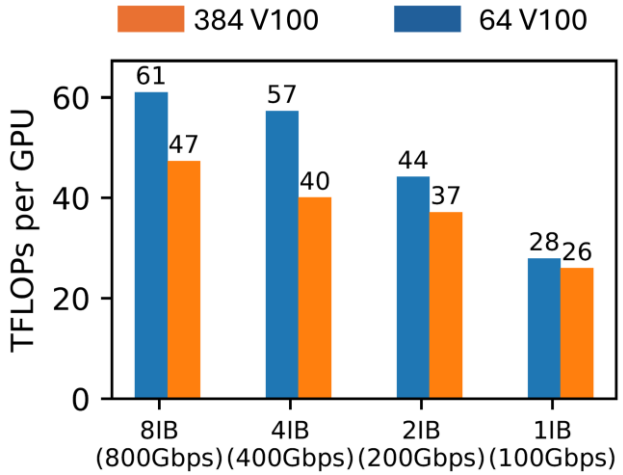
More GPU & More Parallelism

- Max global batch size is fixed
- More GPU => Smaller batch size per GPU

Problem1: Communication is huge overhead in Small batch training

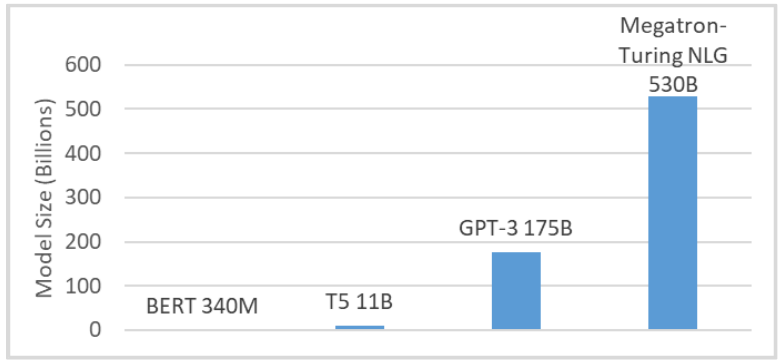


Problem2: Communication is huge overhead with limited inter-node bandwidth



Motivation

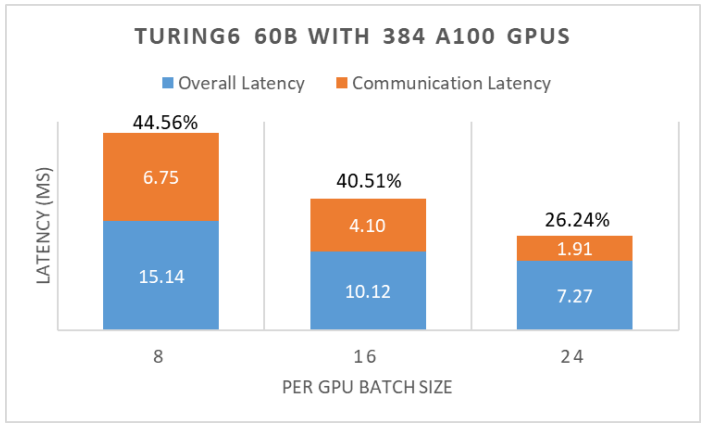
Model size grows exponentially



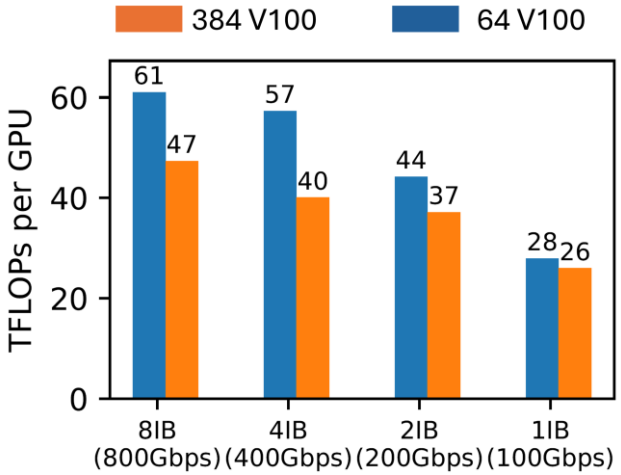
More GPU & More Parallelism

- Max global batch size is fixed
- More GPU => Smaller batch size per GPU

Problem1: Communication is huge overhead in Small batch training



Problem2: Communication is huge overhead with limited inter-node bandwidth



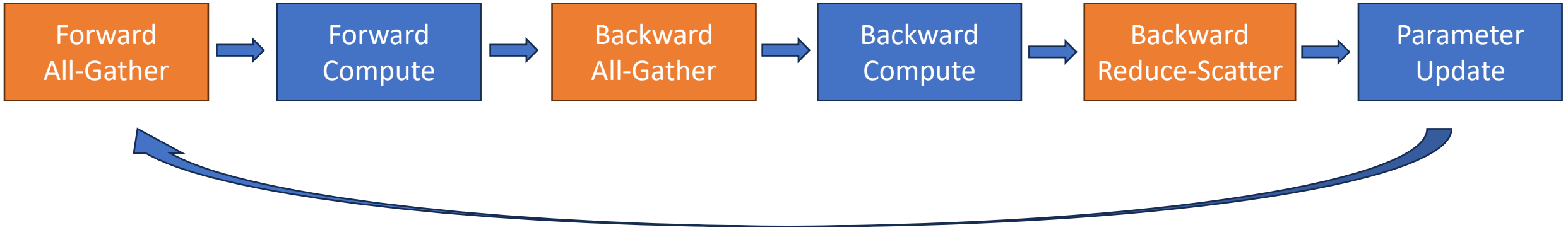
ZeRO++ reduces communication overhead in cases:

1. Small batch training
2. Limited inter-node bandwidth

ZeRO Optimizer

- Easy to Use
- Memory-Efficient Data-Parallel Training paradigm
- For Billion/Trillion parameters training: ZeRO-3

ZeRO-3 training workflow



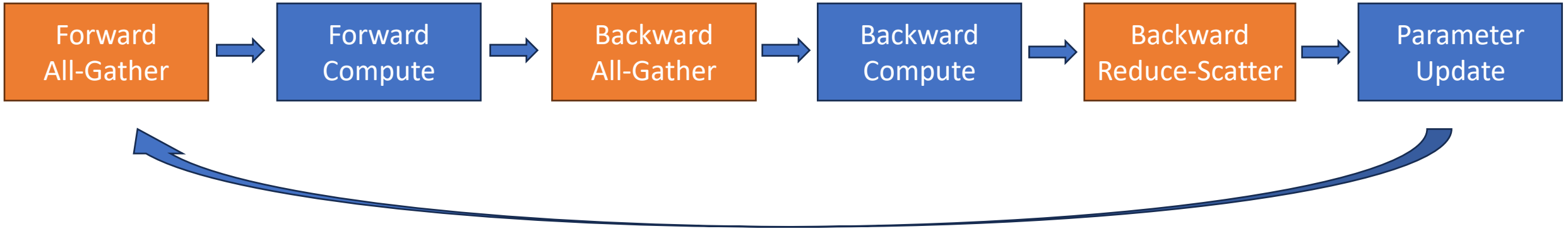
Communication Characterization for ZeRO

Communication Volume Breakdown (model size M):

1. Forward all-gather on weights: *M*
2. Backward all-gather on weights: *M*
3. Backward reduce-scatter on gradients: *M*

- ***Total Volume: 3M***

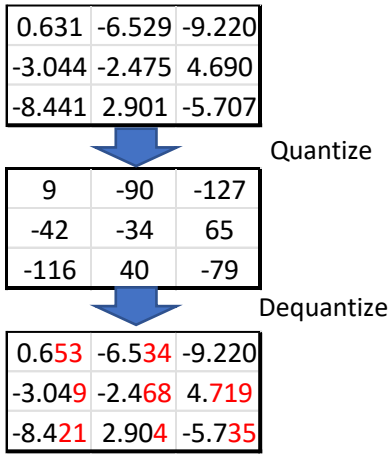
ZeRO-3 training workflow



How to reduce communication volume?

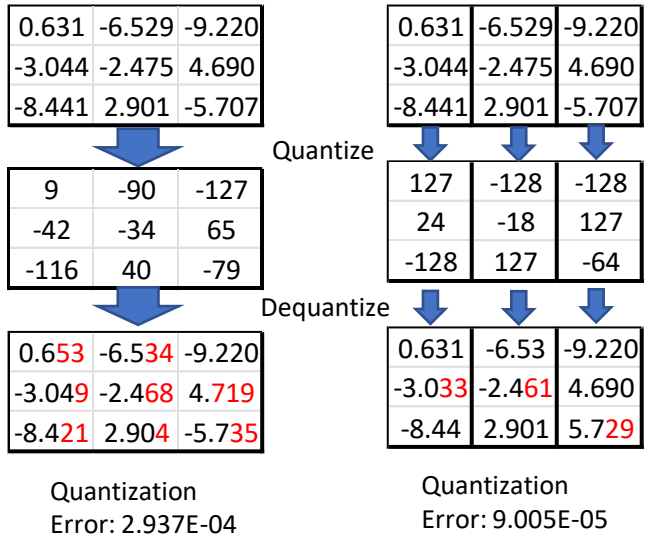
qwZ: Reduce *forward all-gather* Comm

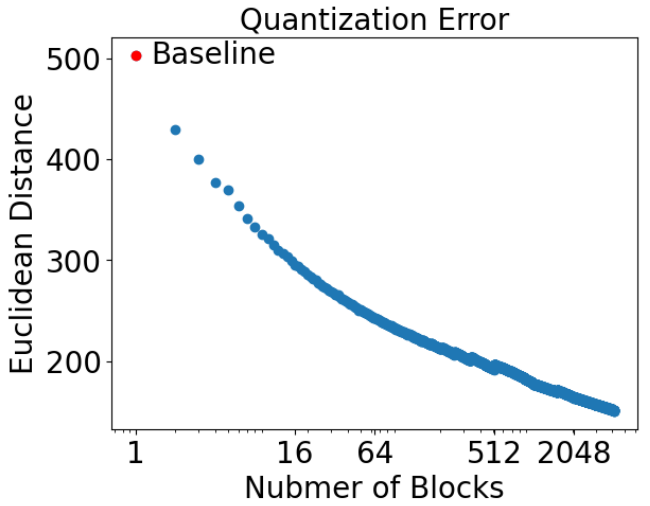
- Communicate 8-bit quantized weights
  - But naïve quantization causes model divergence



qwZ: Reduce *forward all-gather* Comm

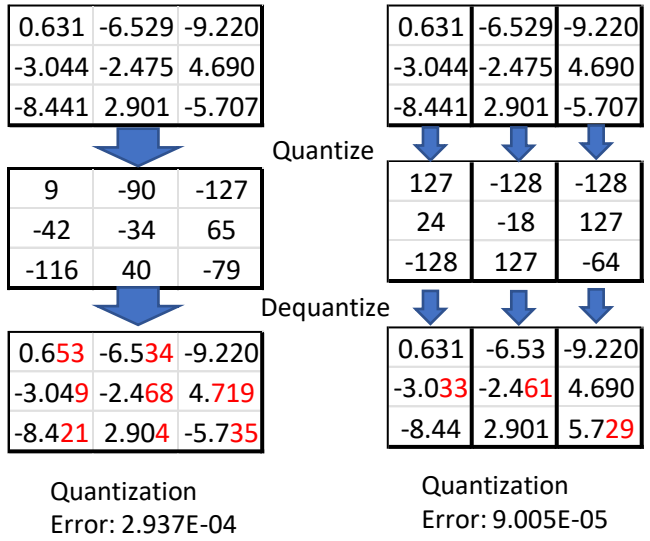
- Communicate 8-bit quantized weights
  - But naïve quantization causes model divergence
- Blocked quantization
  - 3.3x precision improvement in Euclidean distance
  - Optimized kernels for 2.5x faster performance over pytorch

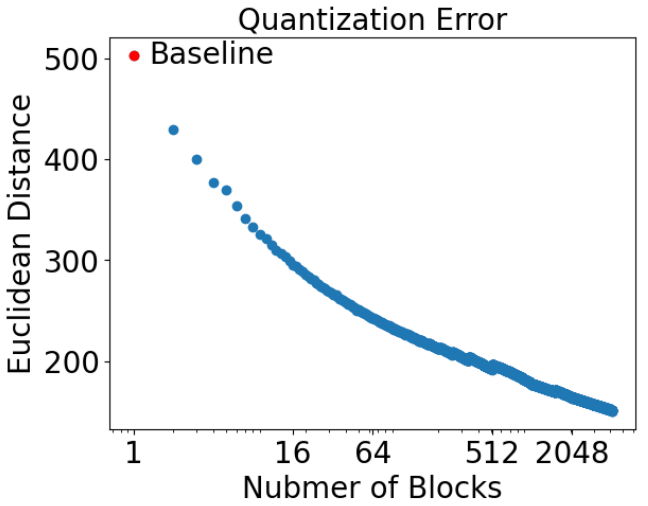




qwZ: Reduce *forward all-gather* Comm

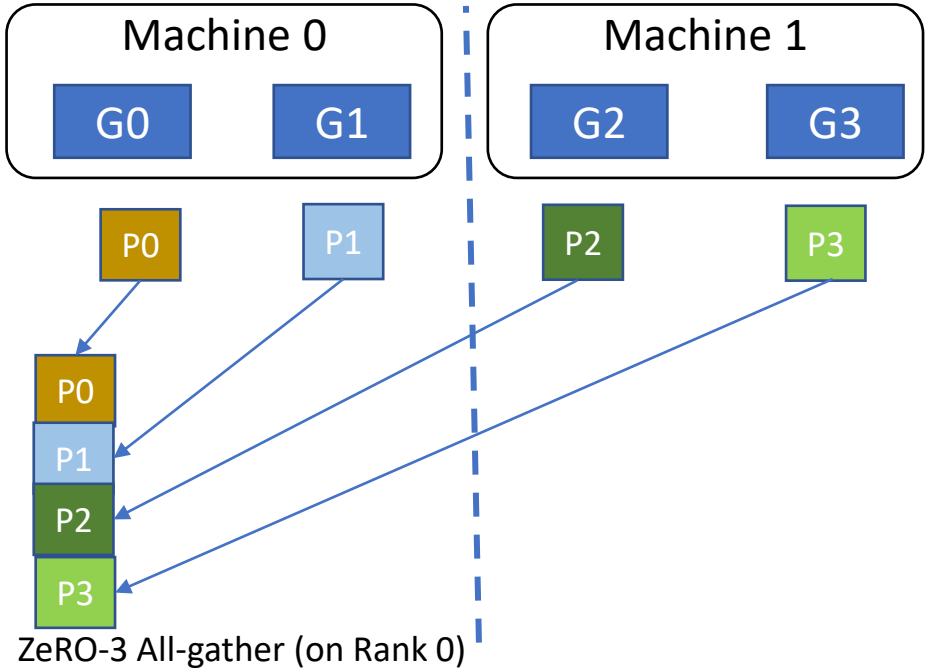
- Communicate 8-bit quantized weights
  - But naïve quantization causes model divergence
- Blocked quantization
  - 3.3x precision improvement in Euclidean distance
  - Optimized kernels for 2.5x faster performance over pytorch
- E2E comm reduction: ***3M → 2.5M***
  - Forward all-gather: *M → 0.5M*
  - Backward all-gather: M
  - Backward reduce-scatter: M





hpZ: Reduce *backward all-gather* Comm

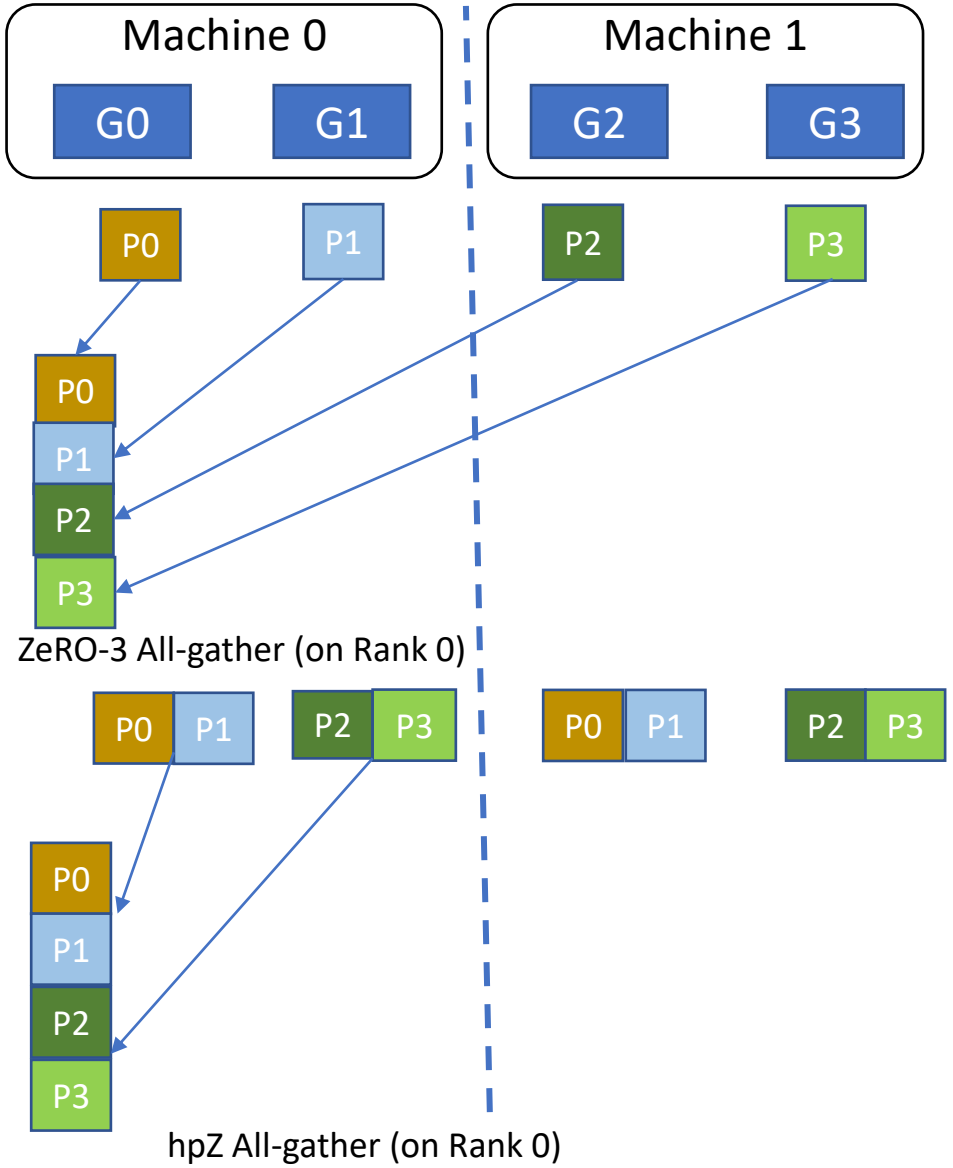
- Heterogeneous partitioning (hpZ)



hpZ: Reduce *backward all-gather* Comm

- Heterogeneous partitioning (hpZ)
  - Model weights within node, rest across all nodes
  - All-gather happens within node only
  - Trade off between memory and communication

- E2E comm reduction: ***3M → 1.5M***
  - Forward all-gather: M → **0.5*M***
  - Backward all-gather: *M → 0*
  - Backward reduce-scatter: M



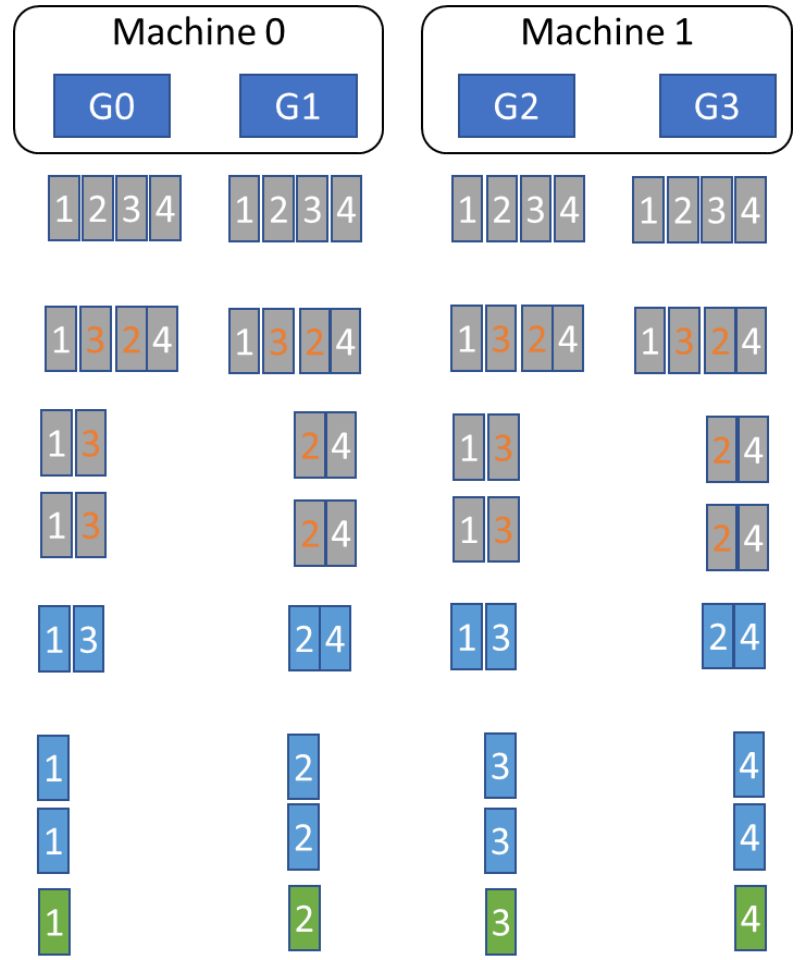
qgZ: Reduce *backward reduce-scatter Comm*

- Can we quantize gradient communication ?
  - Significant precision loss due to reduction operations

- Novel hierarchical All-to-All replacing reduce-scatter
  - Communicate in 4 or 8-bits
  - Reduce in full-precision

qgZ: Reduce *backward reduce-scatter Comm*

- Can we quantize gradient communication ?
  - Significant precision loss due to reduction operations
- Novel hierarchical All-to-All replacing reduce-scatter
  - Communicate in 4 or 8-bits
  - Reduce in full-precision
- E2E comm reduction: ***3M → 0.75M***
  - Forward all-gather: M → 0.5*M*
  - Backward all-gather: M → 0
  - Backward reduce-scatter: *M → 0.25M*



Methodology Summary

Breakdown of ZeRO communication cost (consider a model of size *M*):

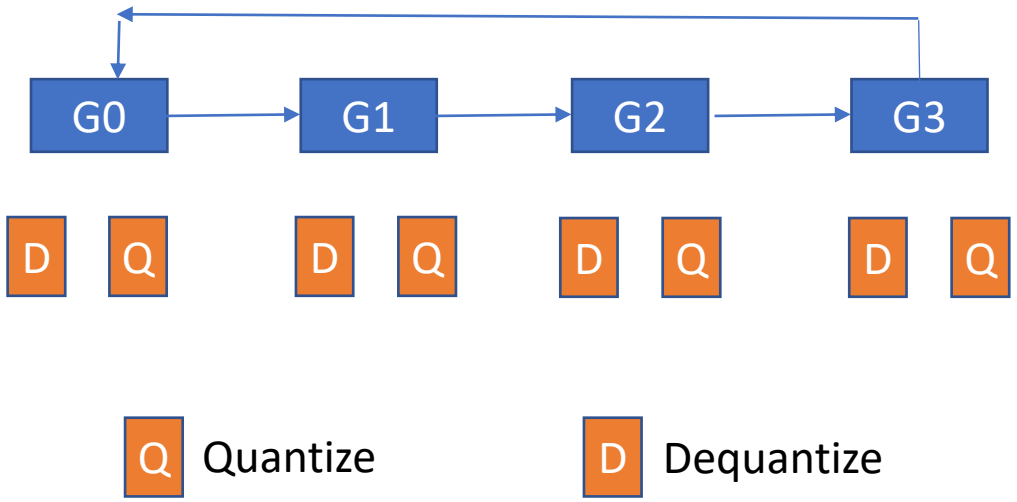
1. Forward all-gather (size *M*) —Accurate & Efficient Quantization→ (size 0.5*M*)
2. Backward all-gather (size *M*) —Heterogeneous Partitioning→ (size 0)
3. Backward reduce-scatter (size *M*) —Novel Quantized Collective→ (size 0.25*M*)

Overall communication reduction: 3M -> 0.75M

System Design for Gradients Communication(qgZ)

Initial Challenges for Quantization on Gradients:

- No existing collectives for quantized gradient communication
- 1-bit Adam optimizer cannot be applied at ZeRO-3.
- Directly apply quantization on reduce_scatter has longer latency & lower precision



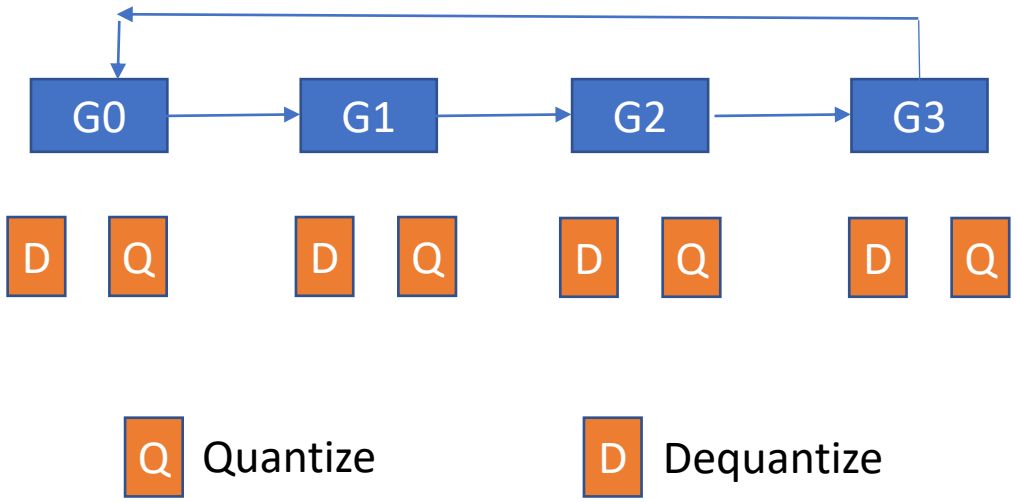
NCCL Ring based reduce_scatter

# of sequential Q+D == # of GPUs

System Design for Gradients Communication(qgZ)

Initial Challenges for Quantization on Gradients:

- No existing collectives for quantized gradient communication
- 1-bit Adam optimizer cannot be applied at ZeRO-3.
- Directly apply quantization on reduce_scatter has longer latency & lower precision



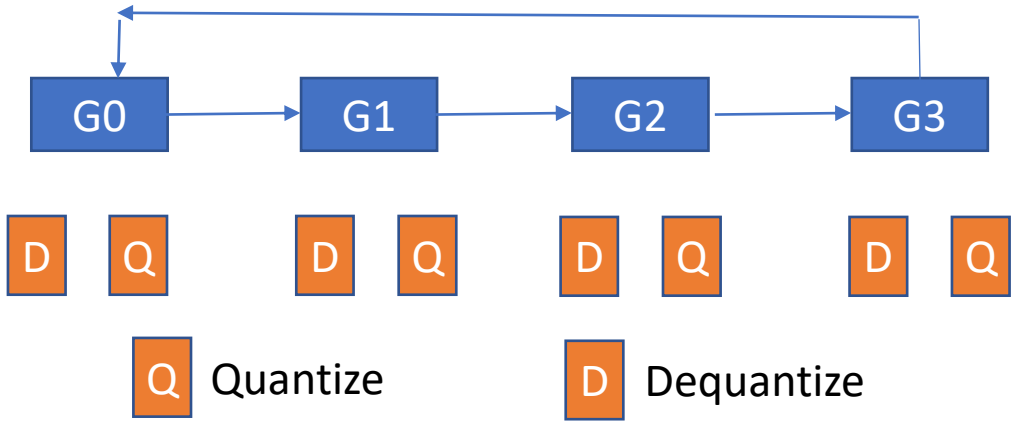
NCCL Ring based reduce_scatter

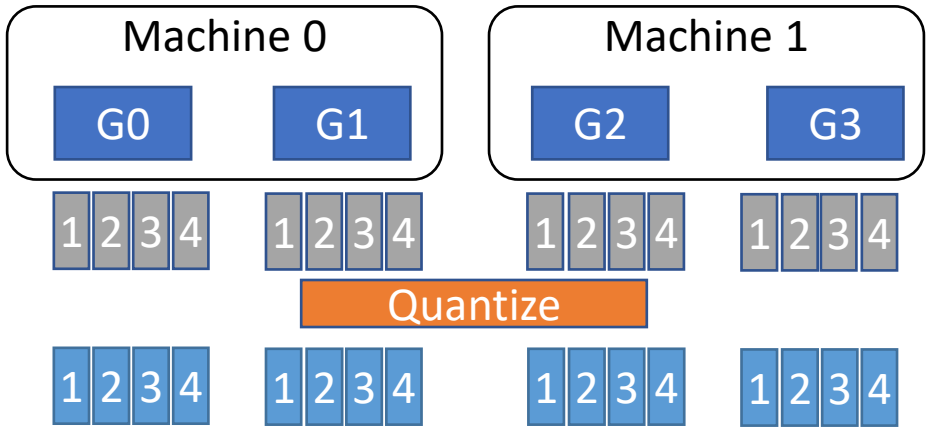
# of sequential Q+D == # of GPUs

Too many Quantization

System Design for Gradients Communication(qgZ)

- Challenge 1: Too many Quantization
- Solution 1: Replacing ring-based reduce_scatter with 1-hop all_to_all

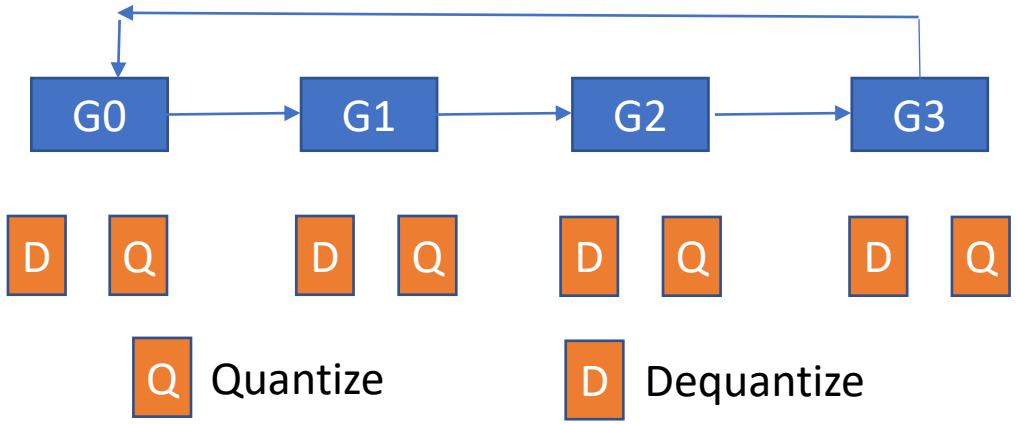


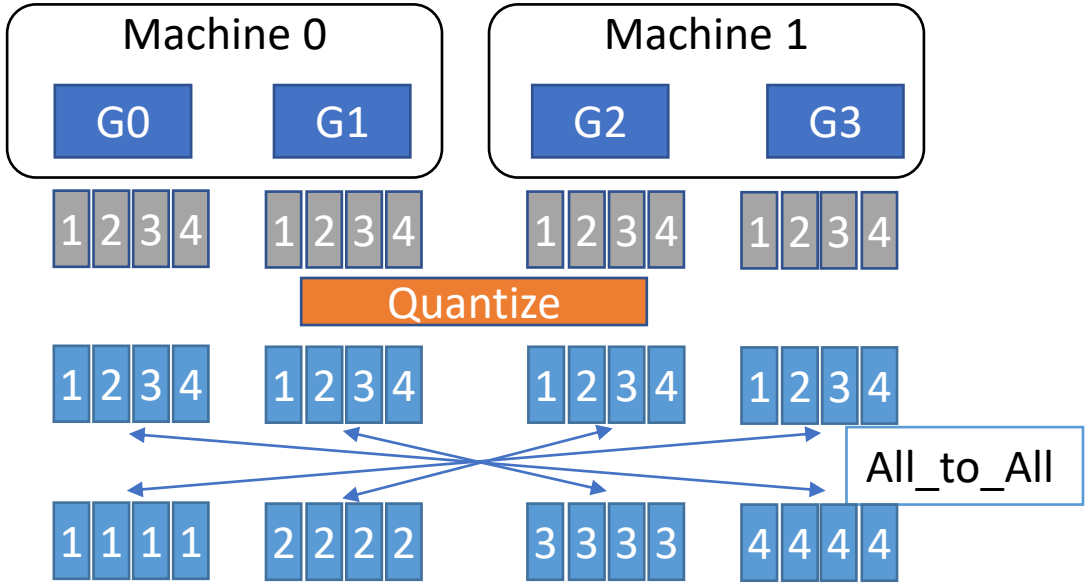


NCCL Ring-based reduce_scatter
# of sequential Q+D == # of GPUs

System Design for Gradients Communication(qgZ)

- Challenge 1: Too many Quantization
- Solution 1: Replacing ring-based reduce_scatter with 1-hop all_to_all

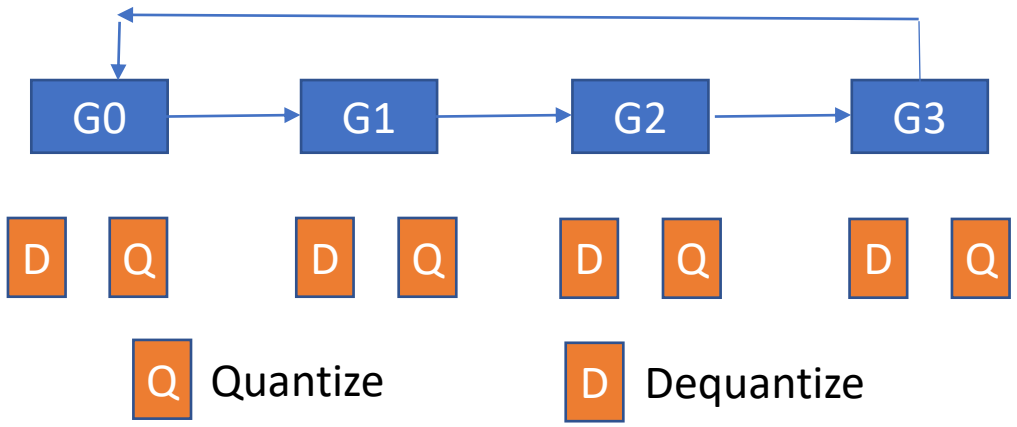




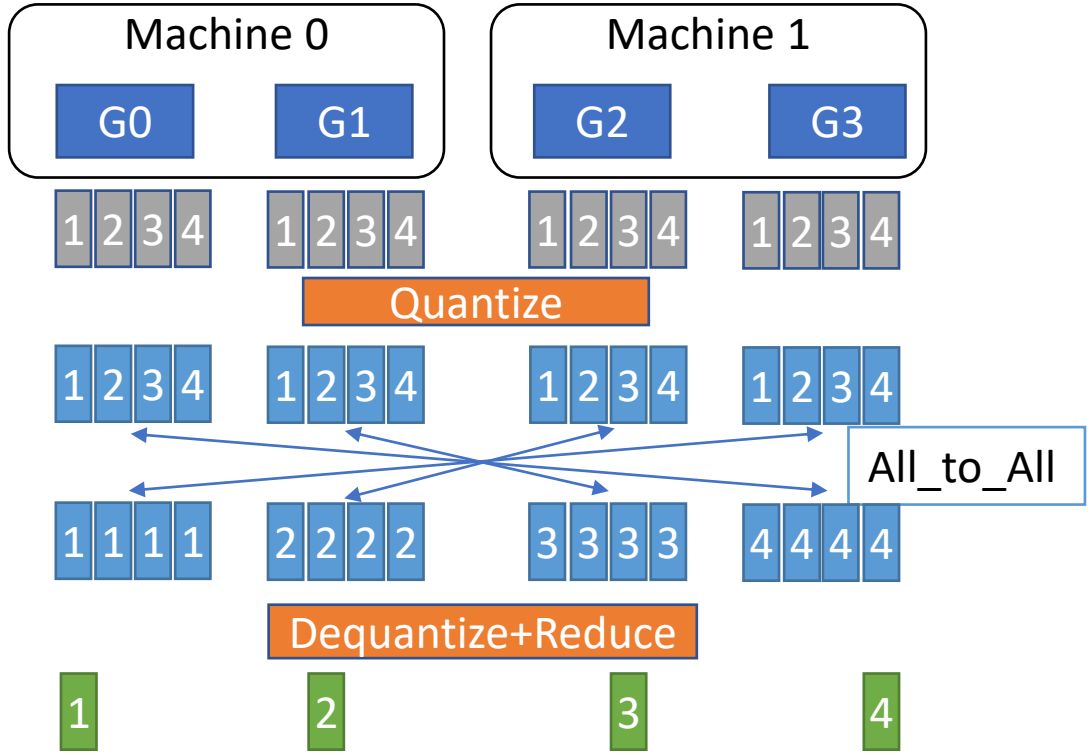
NCCL Ring-based reduce_scatter
# of sequential Q+D == # of GPUs

System Design for Gradients Communication(qgZ)

- Challenge 1: Too many Quantization
- Solution 1: Replacing ring-based reduce_scatter with 1-hop all_to_all



NCCL Ring-based reduce_scatter
# of sequential Q+D == # of GPUs

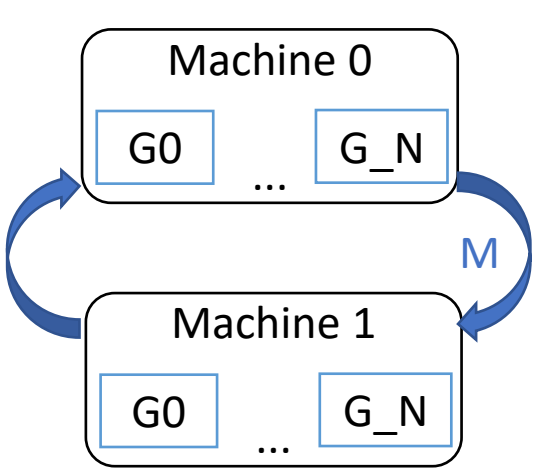


Our 1-hop all_to_all
# of sequential Q+D == 1

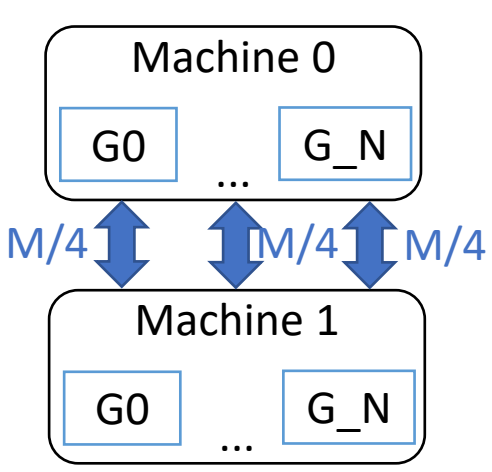
System Design for Gradients Communication(qgZ)

- Challenge 2: Issue with 1-hop all_to_all -> communication volumes blow-up

N gpu per node, model size M



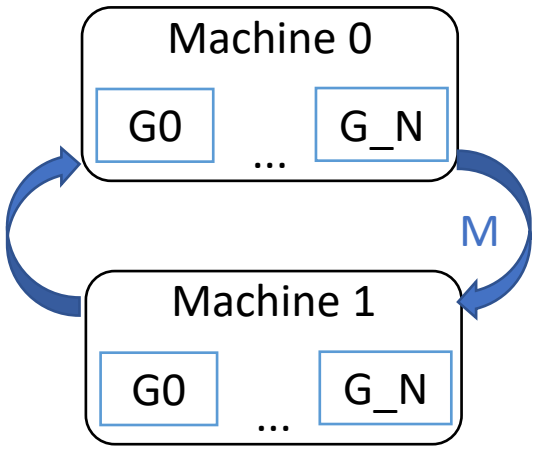
Reduce-scatter M



1-hop all-to-all N*M/4

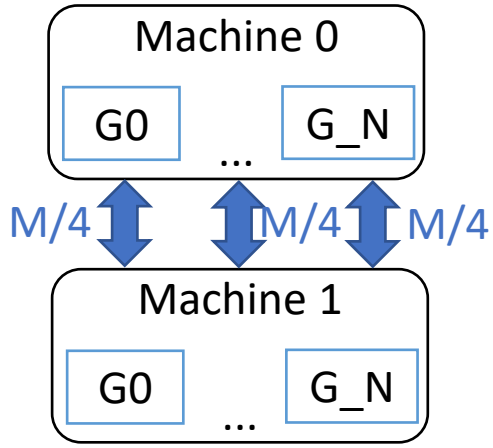
System Design for Gradients Communication(qgZ)

- Challenge 2: Issue with 1-hop all_to_all -> communication volumes blow-up

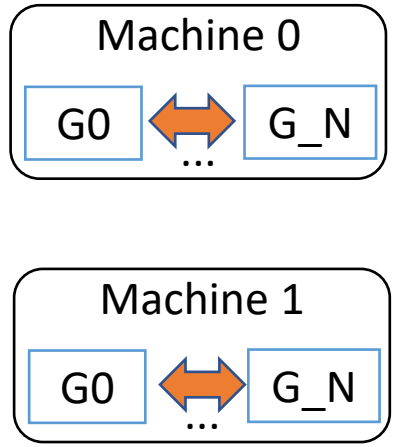


Reduce-scatter M

N gpu per node, model size M



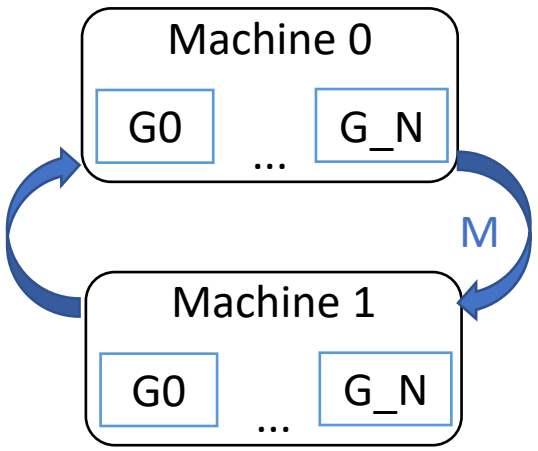
1-hop all-to-all N*M/4



Step 1: intra node all-to-all and reduction (M/(4N))

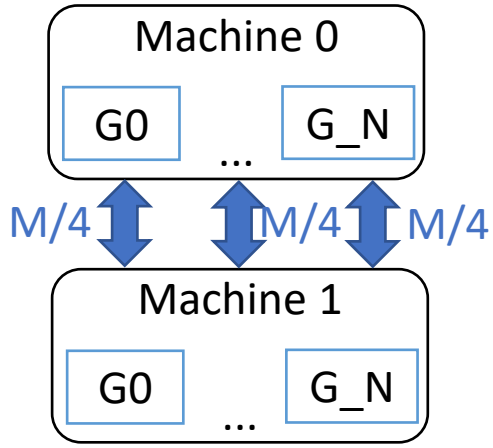
System Design for Gradients Communication(qgZ)

- Challenge 2: Issue with 1-hop all_to_all -> communication volumes blow-up

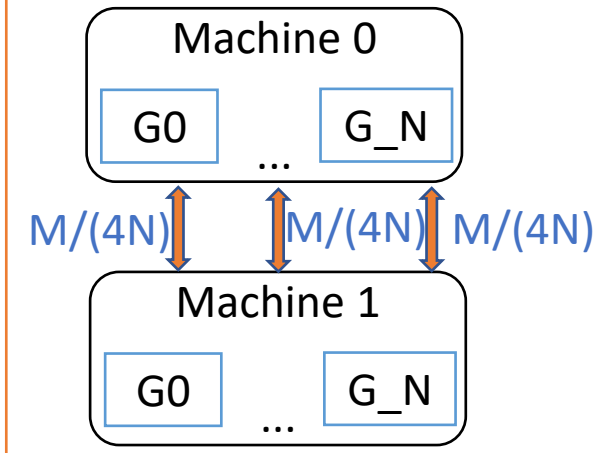


Reduce-scatter M

N gpu per node, model size M



1-hop all-to-all N*M/4

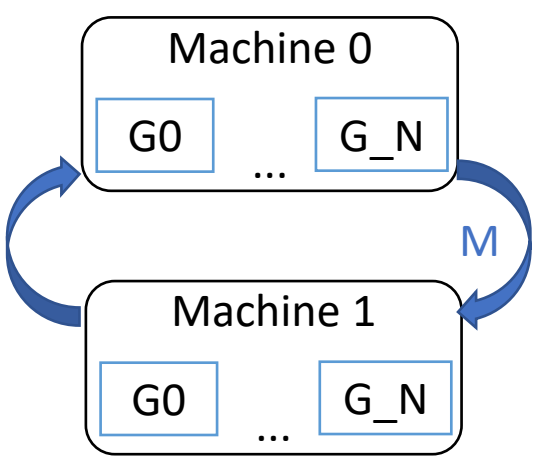


Step 2: inter node all-to-all
(M/(4N))*N = M/4

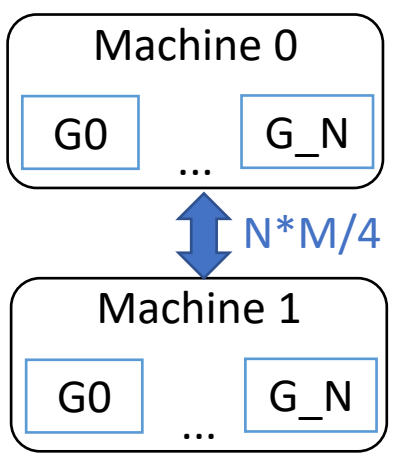
System Design for Gradients Communication(qgZ)

- Challenge 2: Issue with 1-hop all_to_all -> communication volumes blow-up

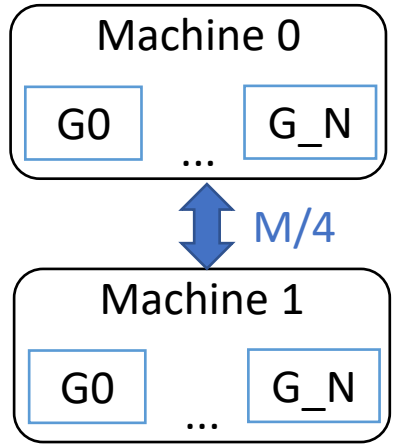
N gpu per node, model size M



Reduce-scatter M



1-hop all-to-all N*M/4

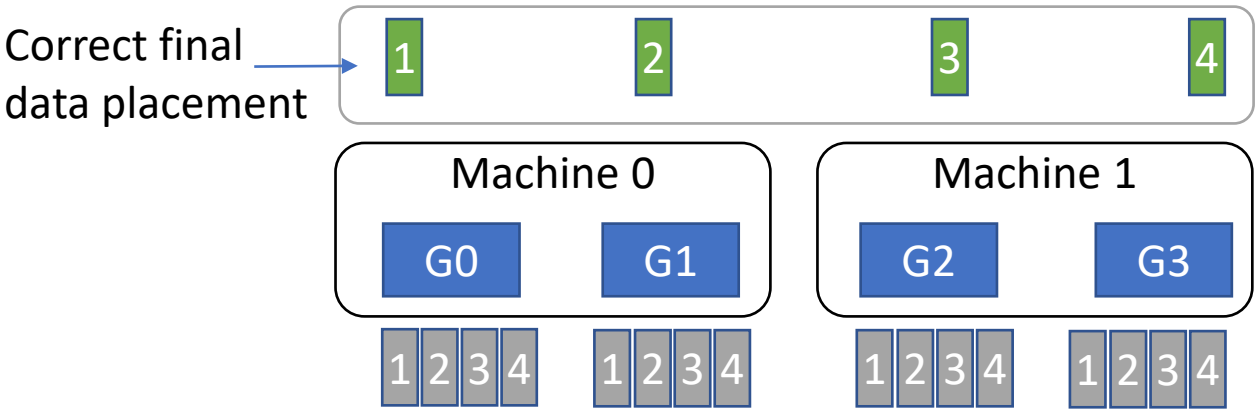


Hierarchical all-to-all M/4

- Solution 2: Hierarchical all-to-all

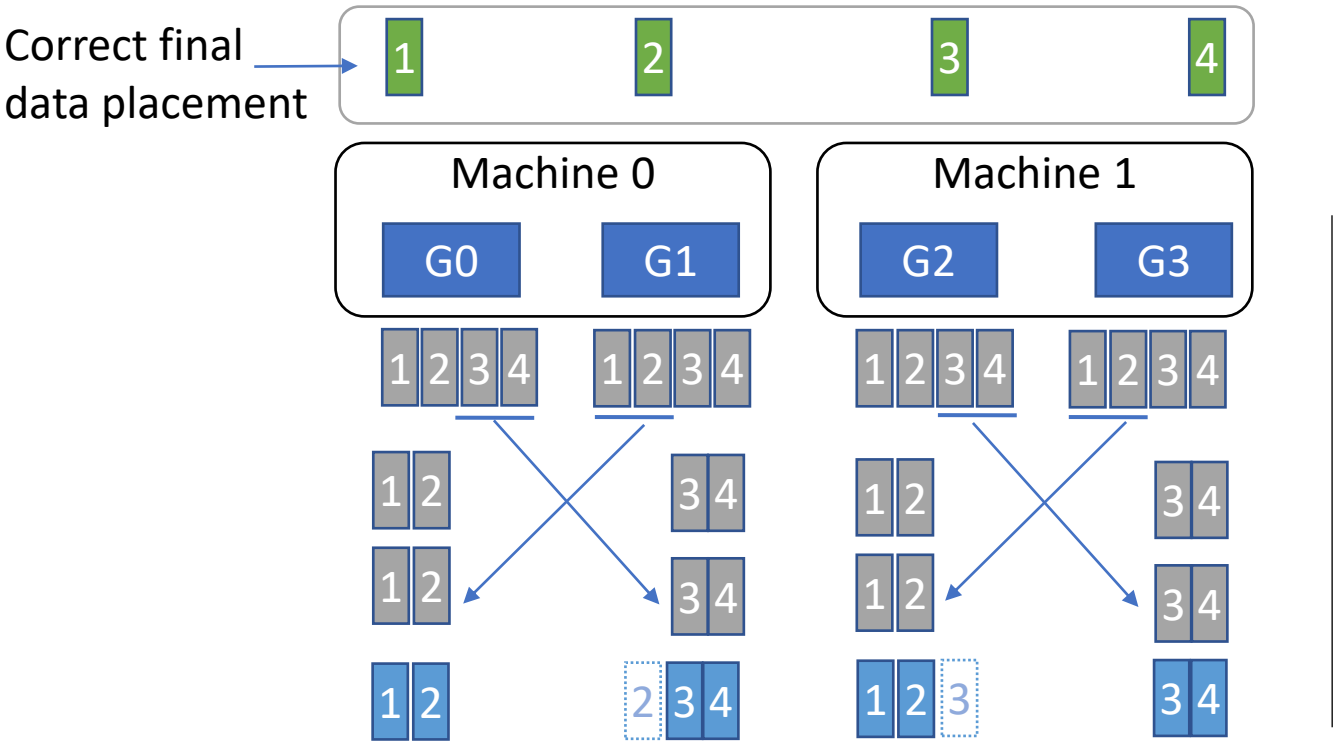
System Design for Gradients Communication(qgZ)

Challenge 3: Hierarchical all-to-all (Data-misplacement)



System Design for Gradients Communication(qgZ)

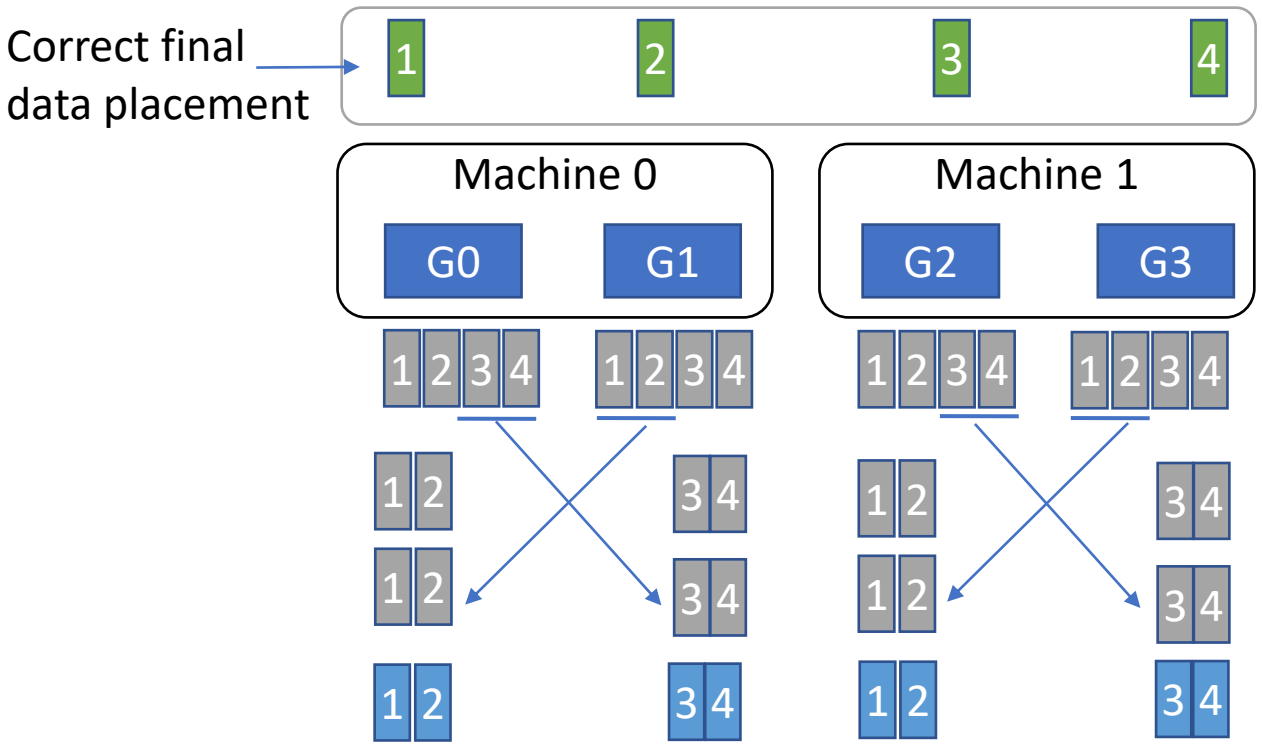
Challenge 3: Hierarchical all-to-all (Data-misplacement)



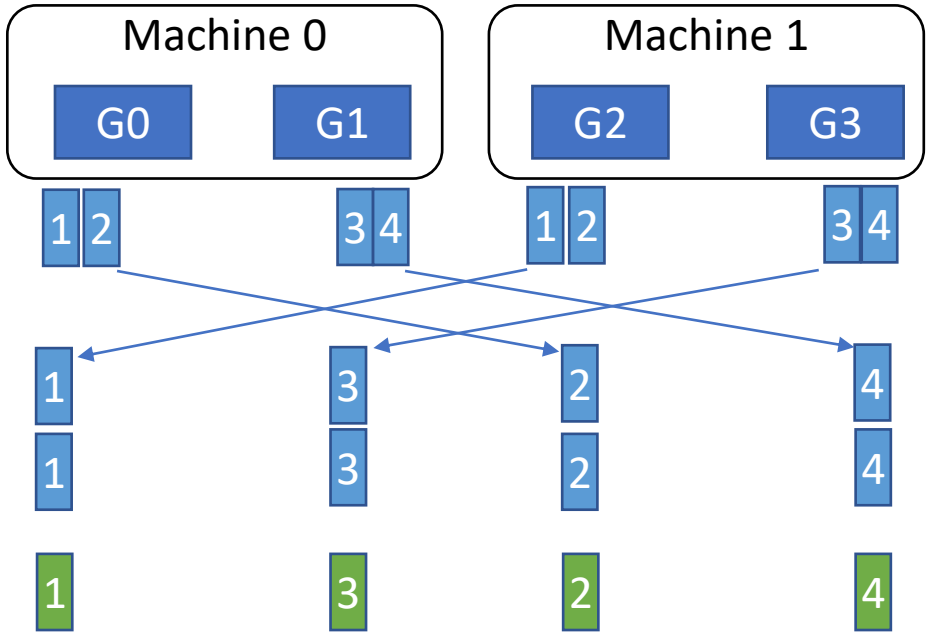
Step 1: intra-node all_to_all

System Design for Gradients Communication(qgZ)

Challenge 3: Hierarchical all-to-all (Data-misplacement)



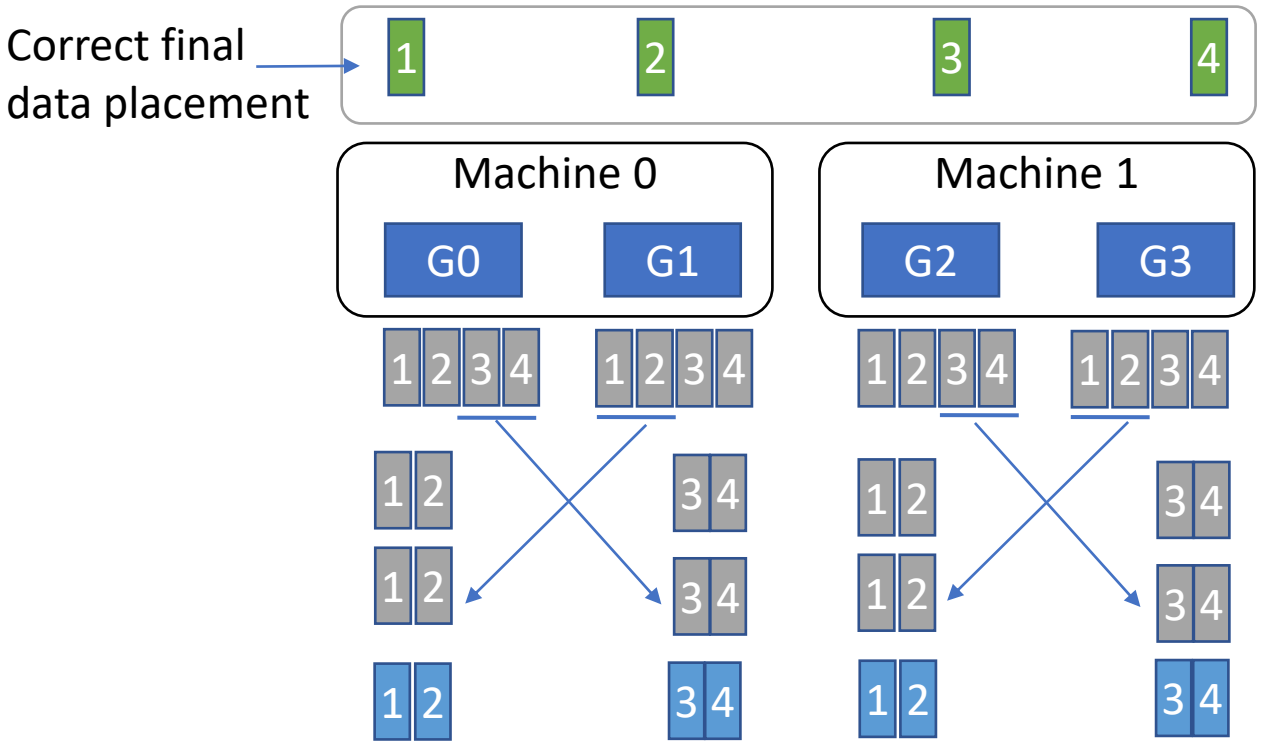
Step 1: intra-node all_to_all



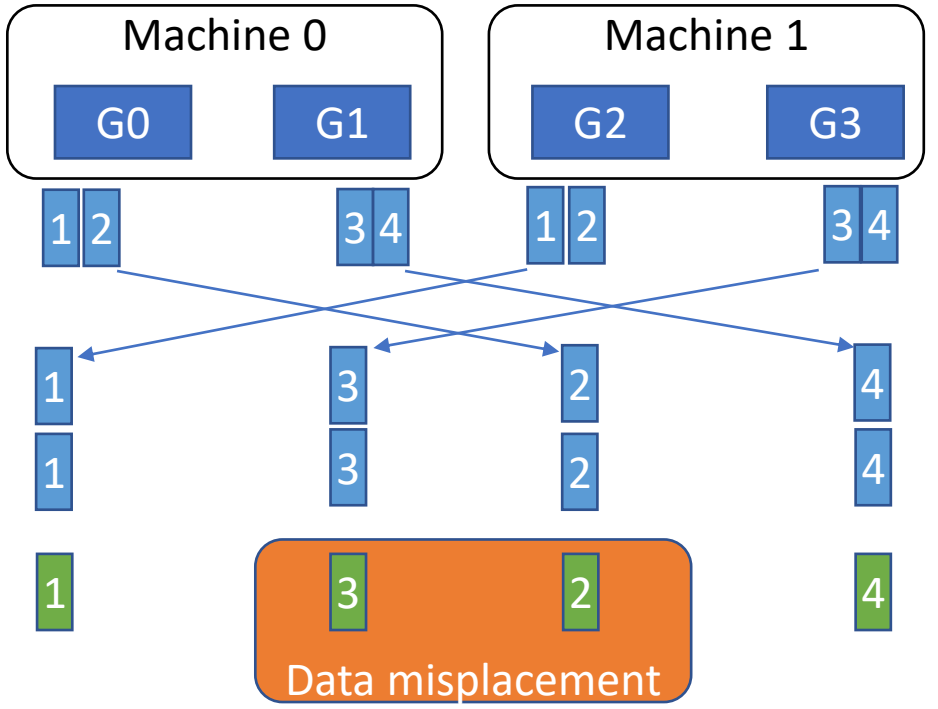
Step 2: Inter-node all_to_all

System Design for Gradients Communication(qgZ)

Challenge 3: Hierarchical all-to-all (Data-misplacement)



Step 1: intra-node all_to_all

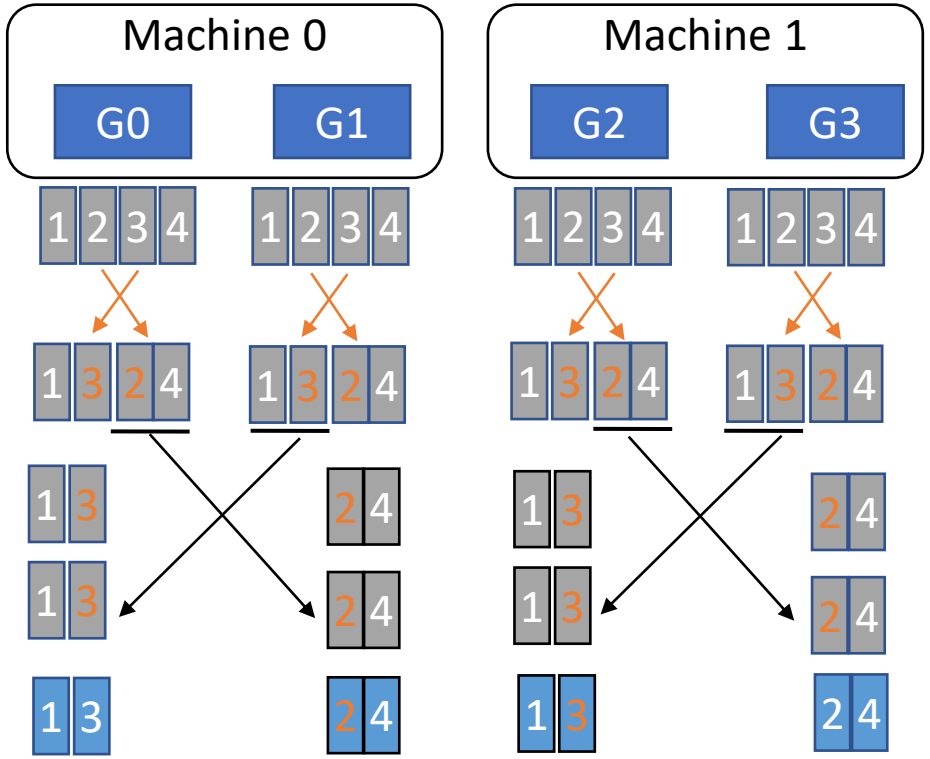


Step 2: Inter-node all_to_all

System Design for Gradients Communication(qgZ)

Challenge 3: Hierarchical all-to-all (Data-misplacement)

Solution 3: Tensor slices reordering

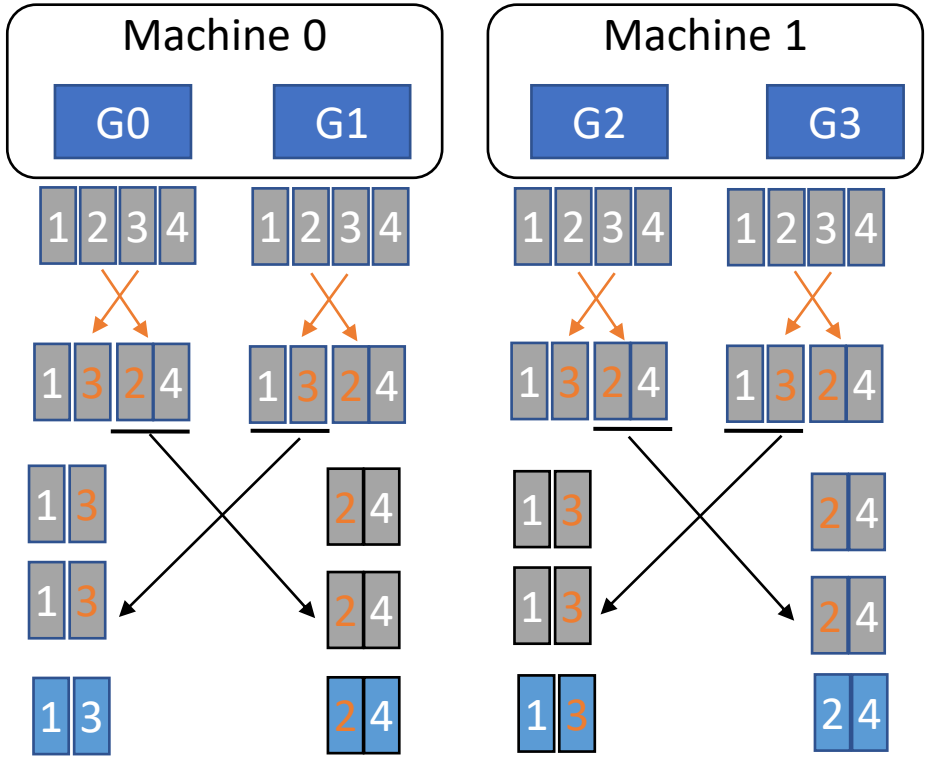


Step 1: intra-node all_to_all

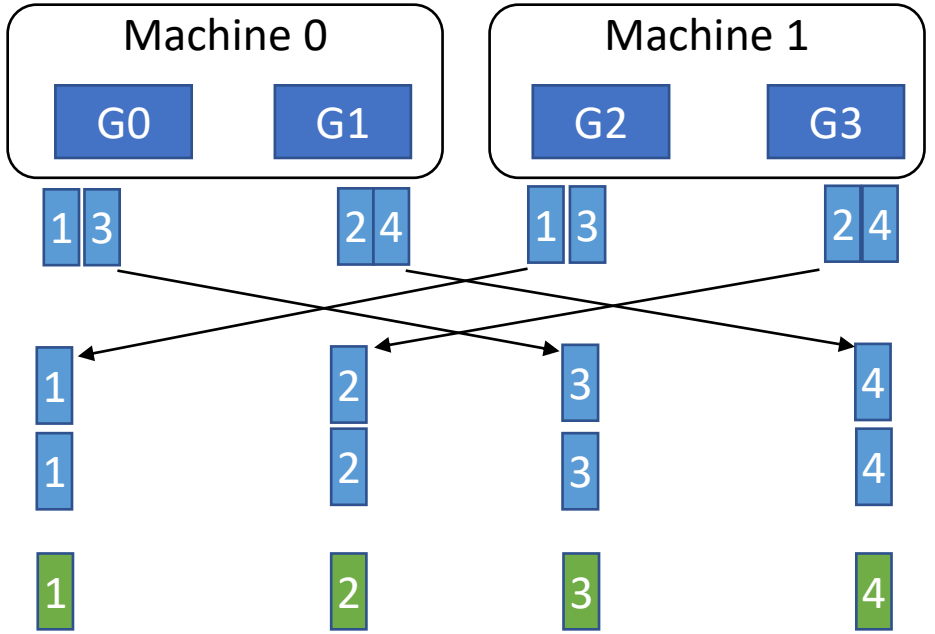
System Design for Gradients Communication(qgZ)

Challenge 3: Hierarchical all-to-all (Data-misplacement)

Solution 3: Tensor slices reordering

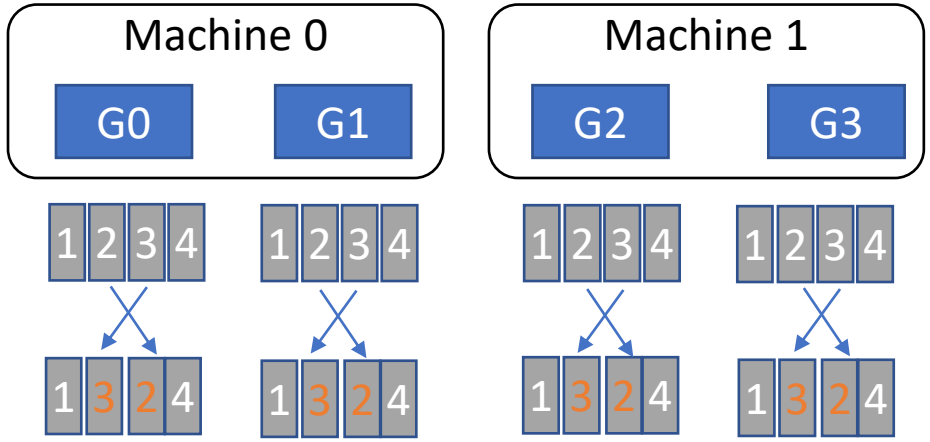


Step 1: intra-node all_to_all



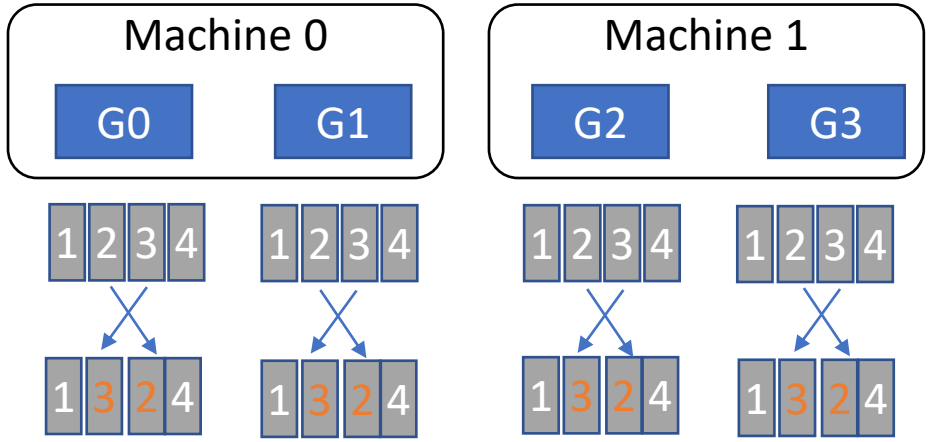
Step 2: inter-node all_to_all

qgZ Overall Workflow



Tensor slice reordering

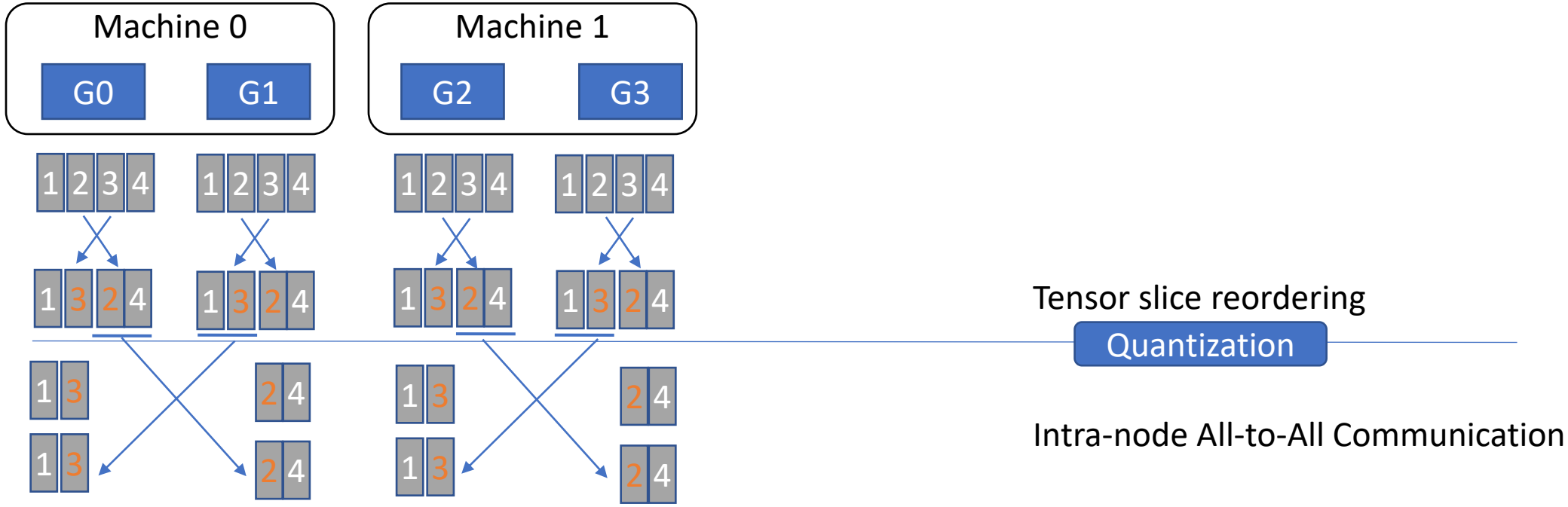
qgZ Overall Workflow



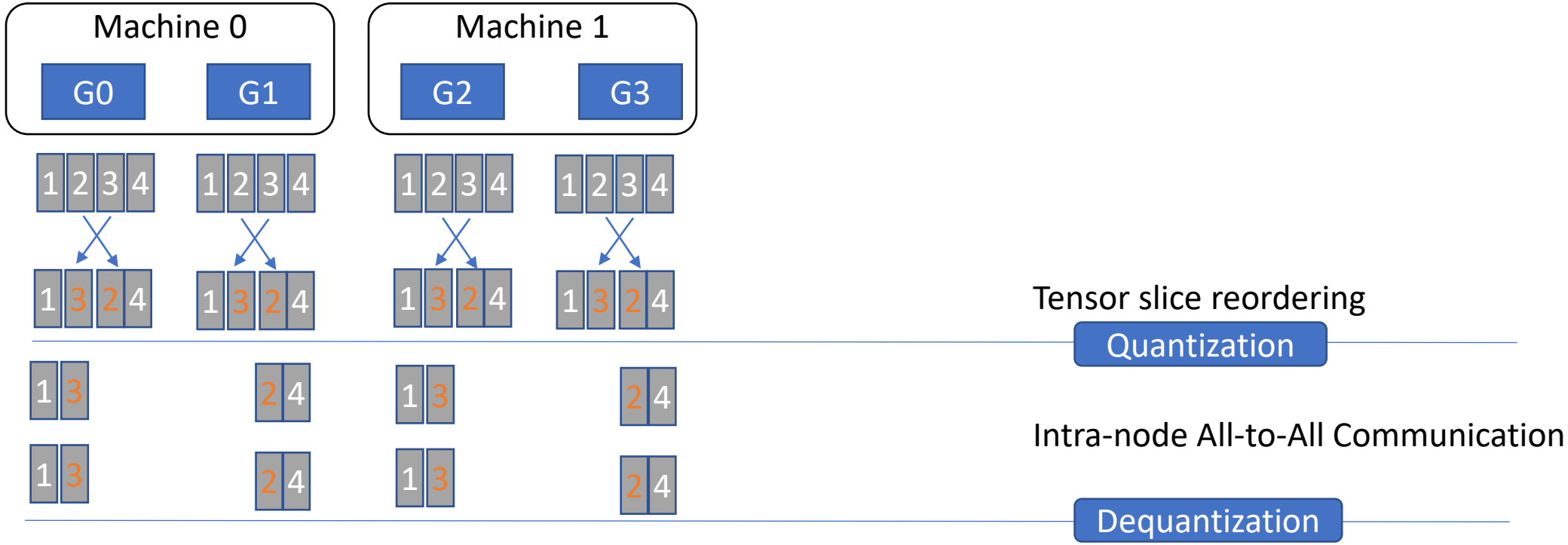
Tensor slice reordering

Quantization

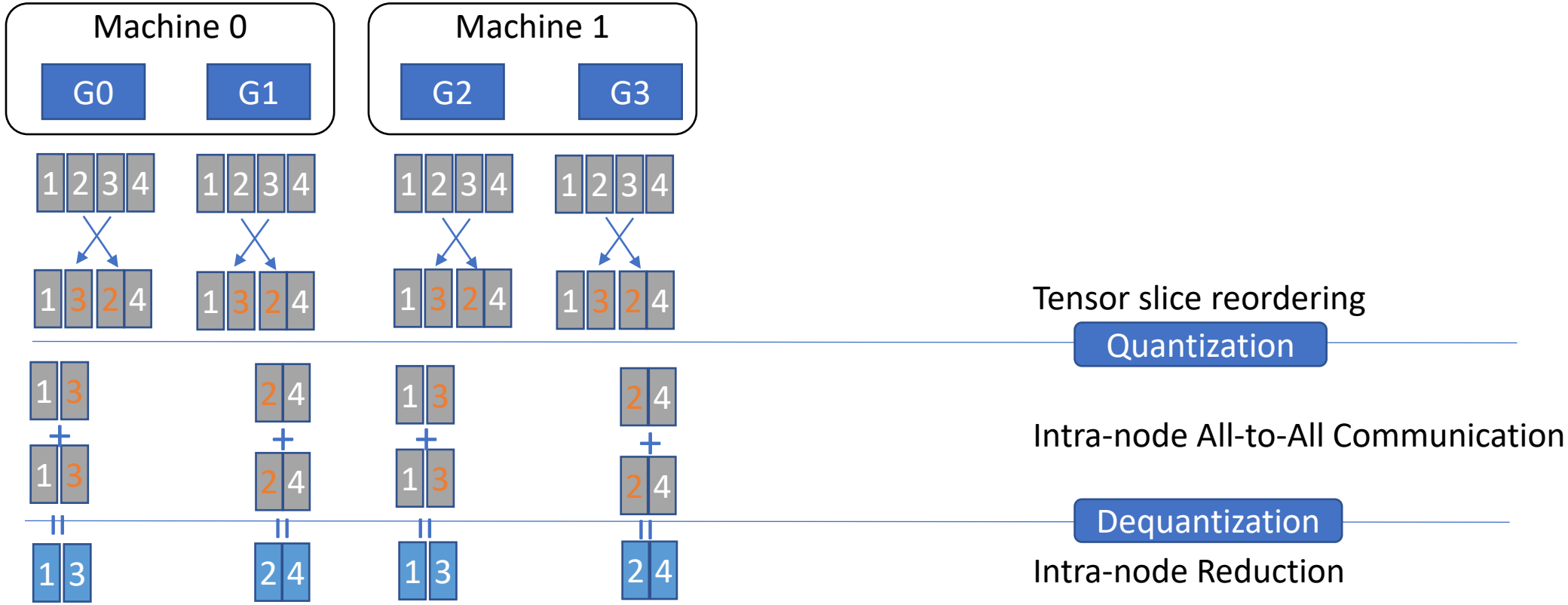
qgZ Overall Workflow



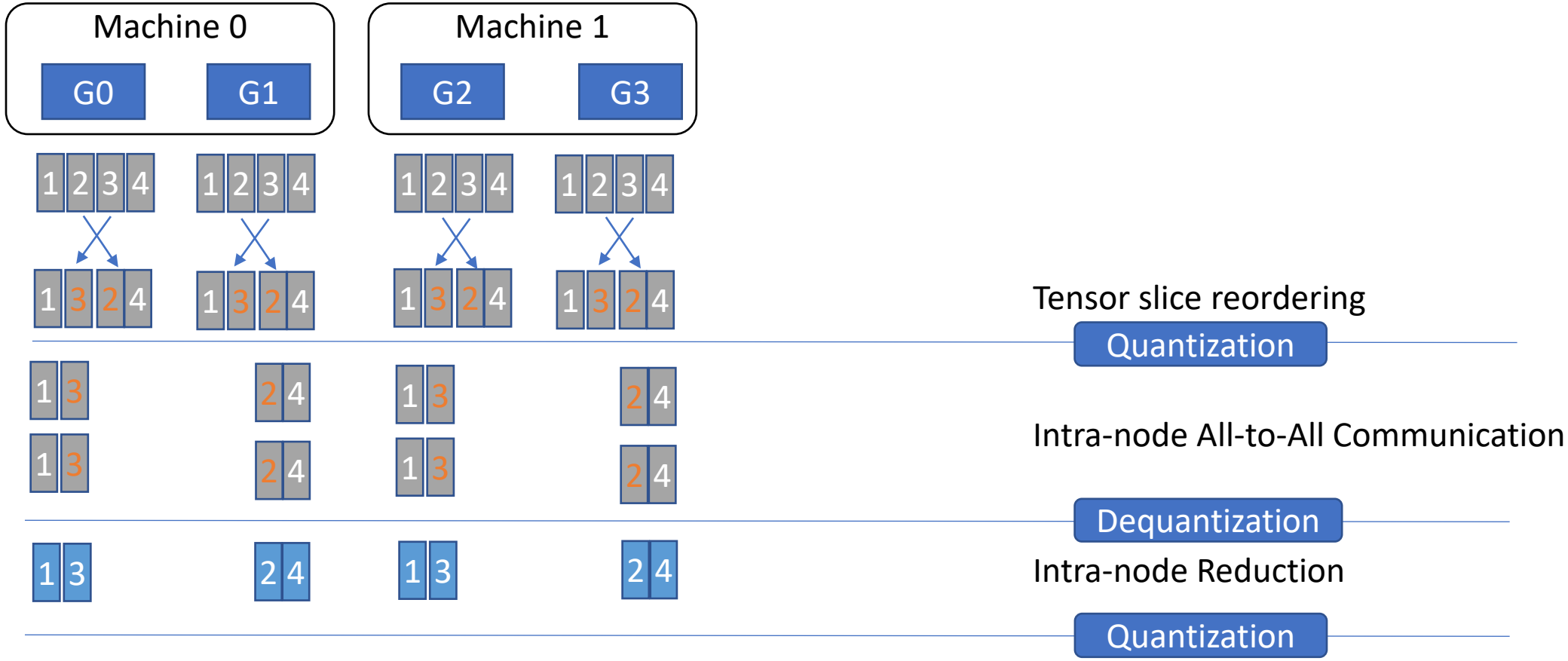
qgZ Overall Workflow



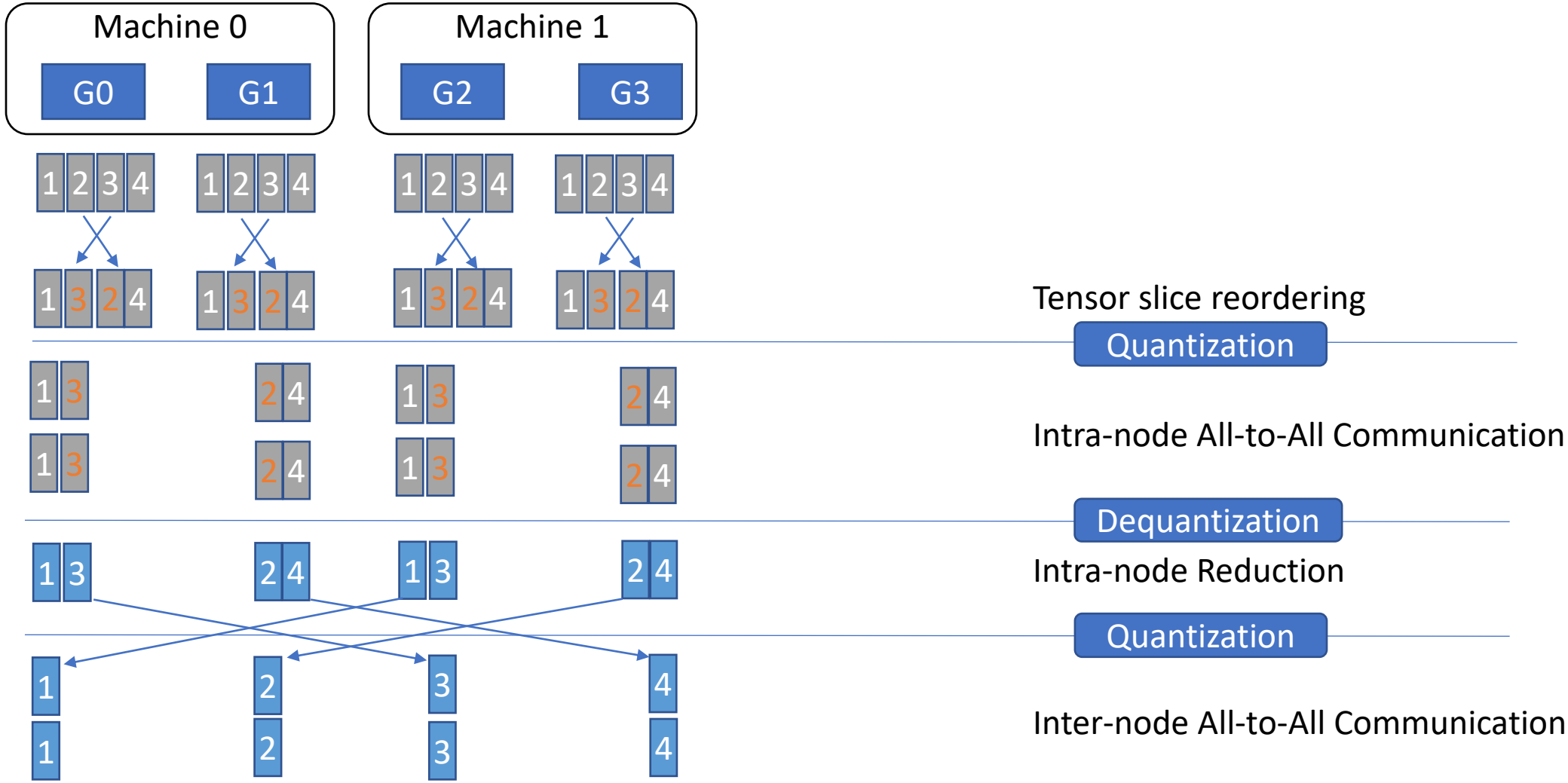
qgZ Overall Workflow



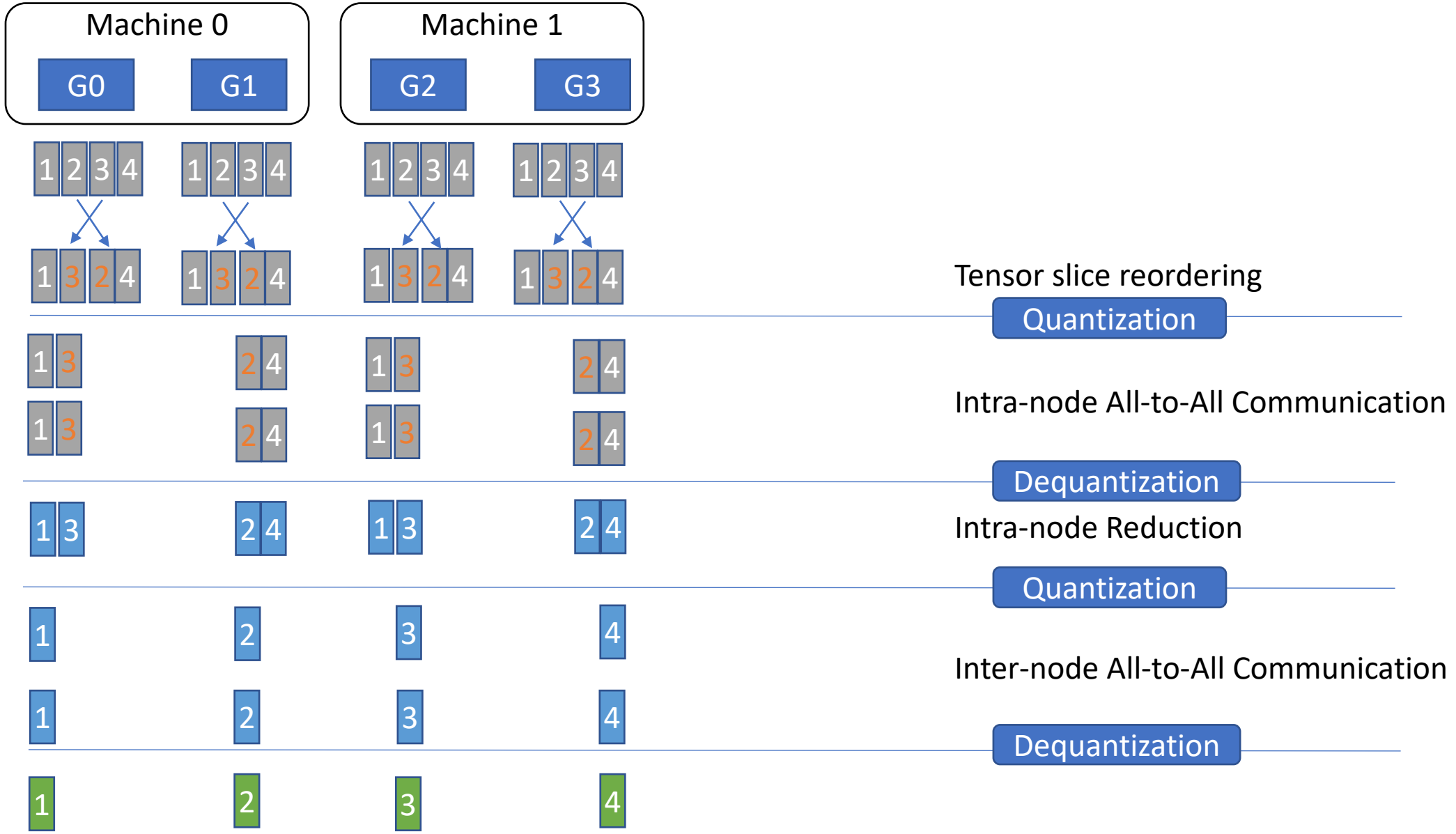
qgZ Overall Workflow



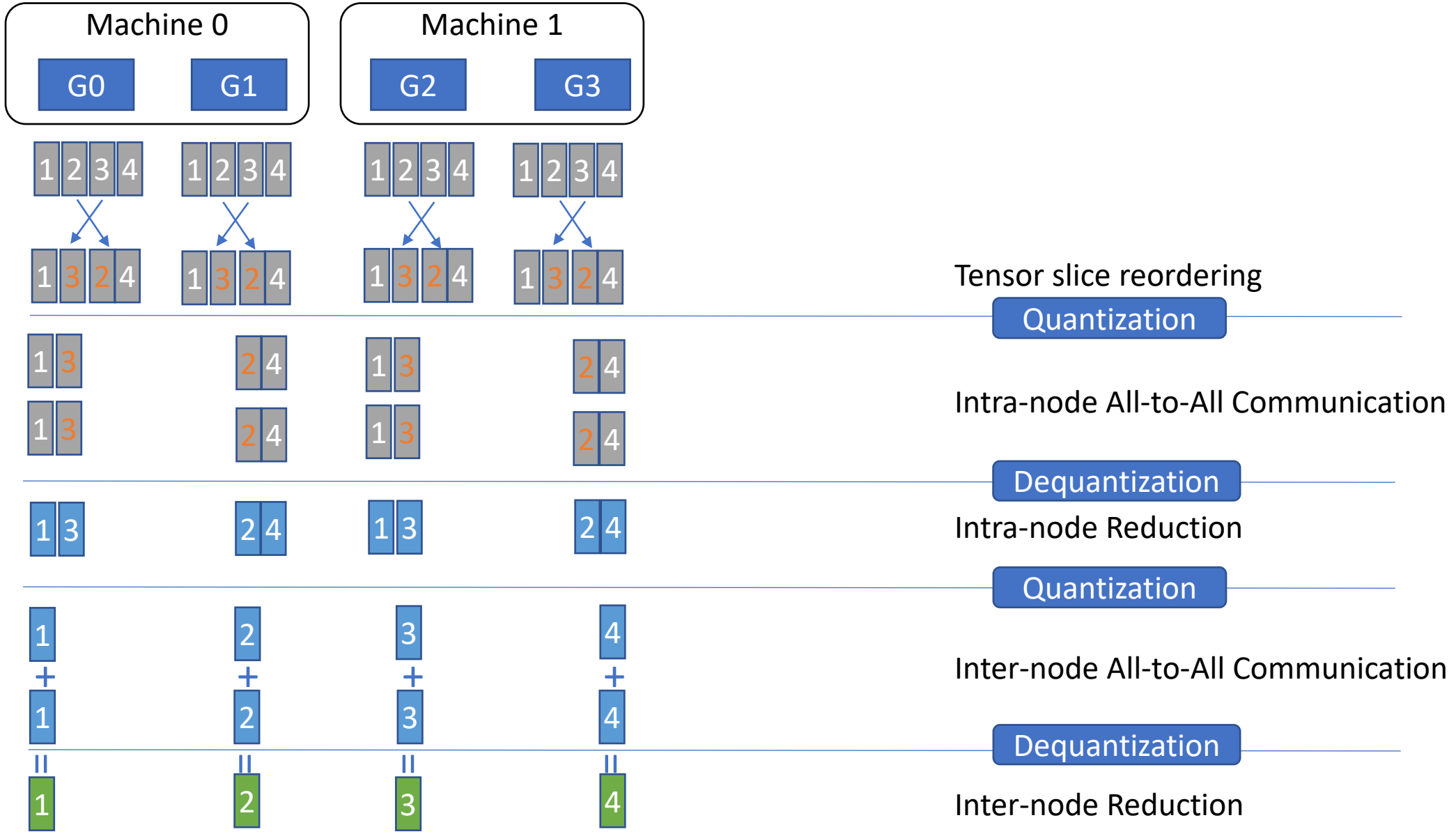
qgZ Overall Workflow



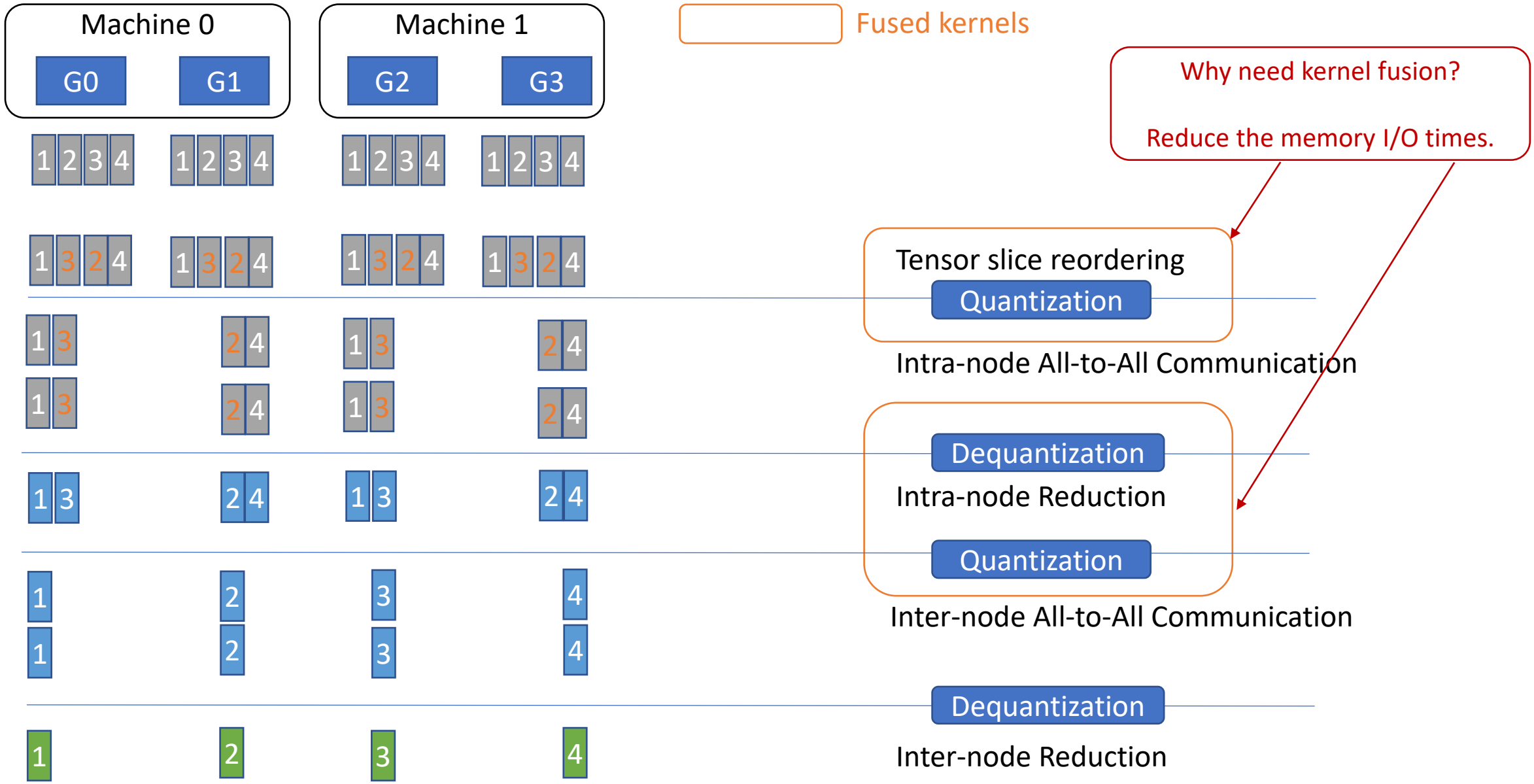
qgZ Overall Workflow



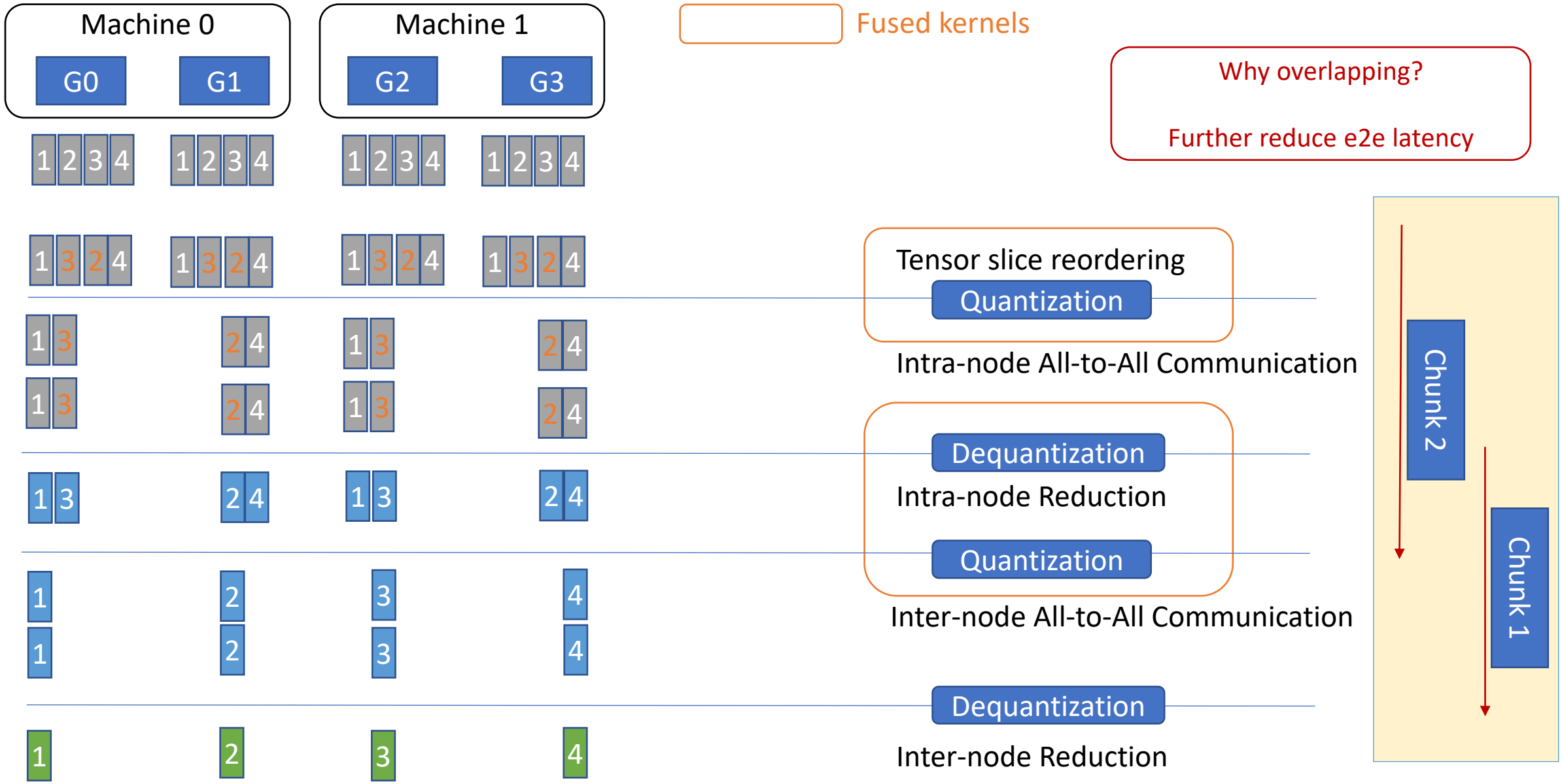
qgZ Overall Workflow



Further Optimization: kernel fusion



Further Optimization: overlapping



Evaluation: GPT model

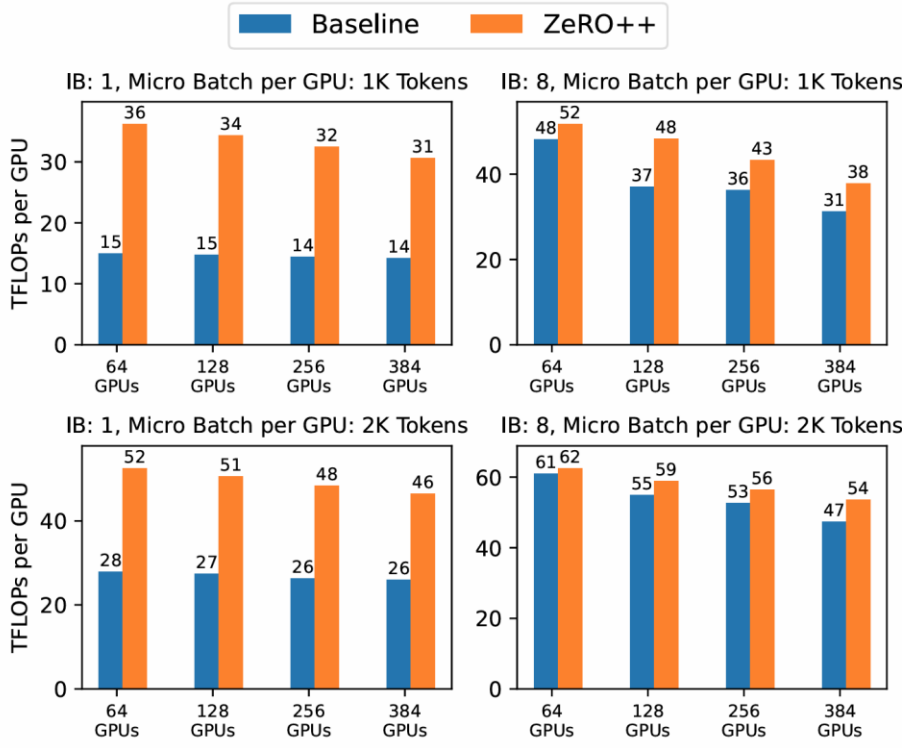
Small batch size

Low cross-node bandwidth

384 V100 GPUs

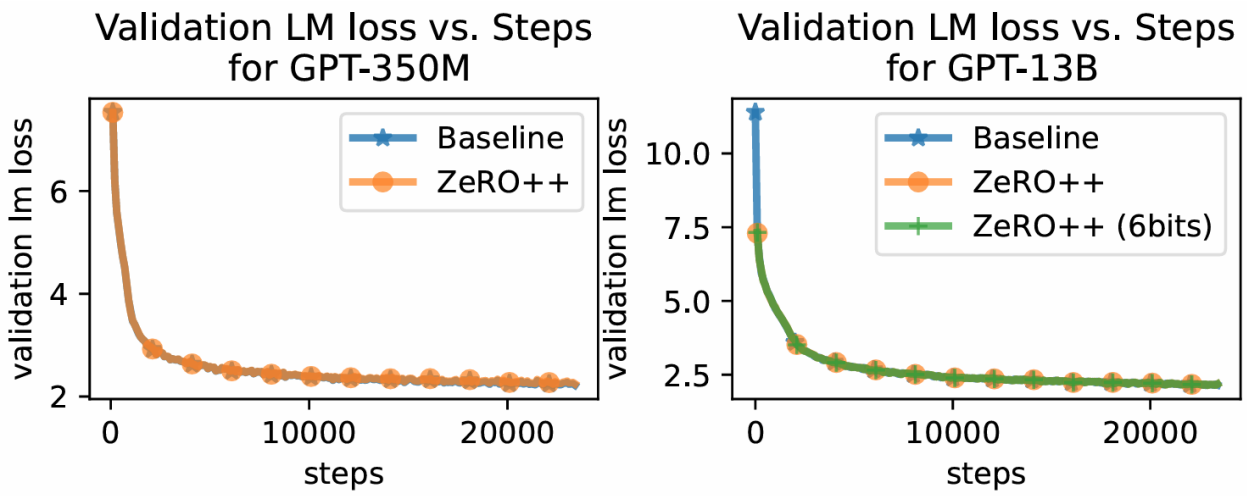
		1 IB Connection			8 IB Connections		
Model Size	Tokens per GPU	Baseline TFLOPs	ZeRO++ TFLOPs	Speedup	Baseline TFLOPs	ZeRO++ TFLOPs	Speedup
138B	2K	19.96	37.90	90%	47.55	55.30	16%
138B	1K	11.25	21.81	94%	34.19	44.38	30%
91B	2K	19.99	38.06	90%	47.74	56.26	18%
91B	1K	11.27	21.93	95%	34.49	44.36	29%
49B	2K	20.06	38.08	90%	48.05	56.24	17%
49B	1K	11.27	21.95	95%	34.54	44.46	29%
18B	2K	25.98	46.40	79%	47.31	53.65	13%
18B	1K	14.15	30.57	116%	31.27	37.87	21%

2.16x speedup

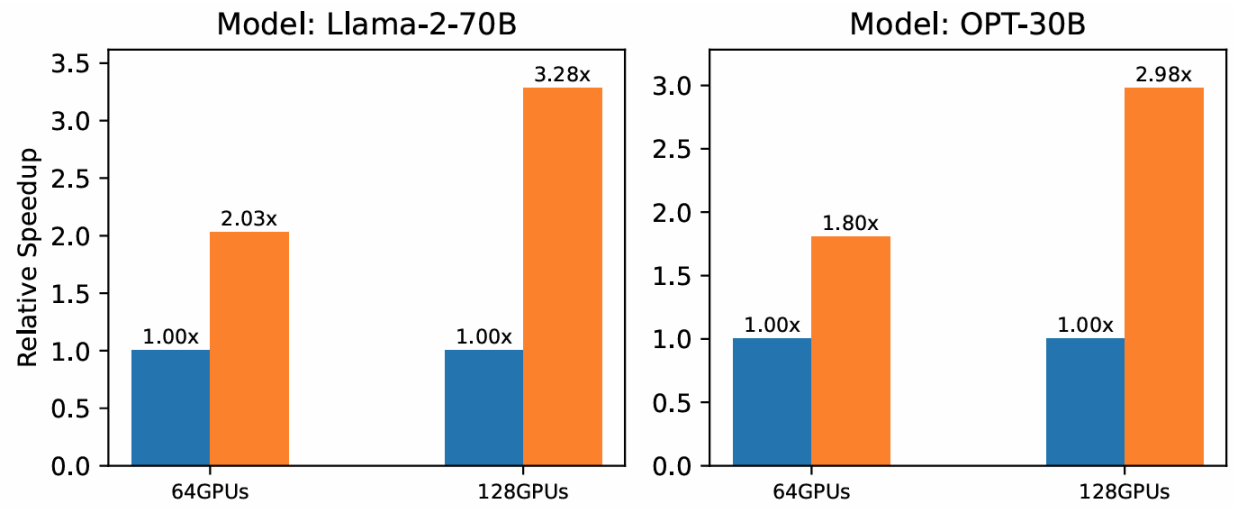


Scalability from 64 to 384 GPUs

Evaluation: GPT model validation loss



Evaluation: RLHF on LLAMA-2-70B & OPT-30B



3.3x for Llama-2-70B

3x for OPT 30B

Summary

- ZeRO++ incorporates three techniques qwZ, hpZ, qgZ to optimize e2e communication during ZeRO training, and reduce 4x communication volumes.

- Comparing with ZeRO, ZeRO++ achieves up to 2.16x speedup for 384 GPU training, and 3.3x speed for RLHF training.

- ZeRO++ has been Integrated into https://github.com/microsoft/DeepSpeed as Next-Gen ZeRO training engine.

guanhuawang@microsoft.com