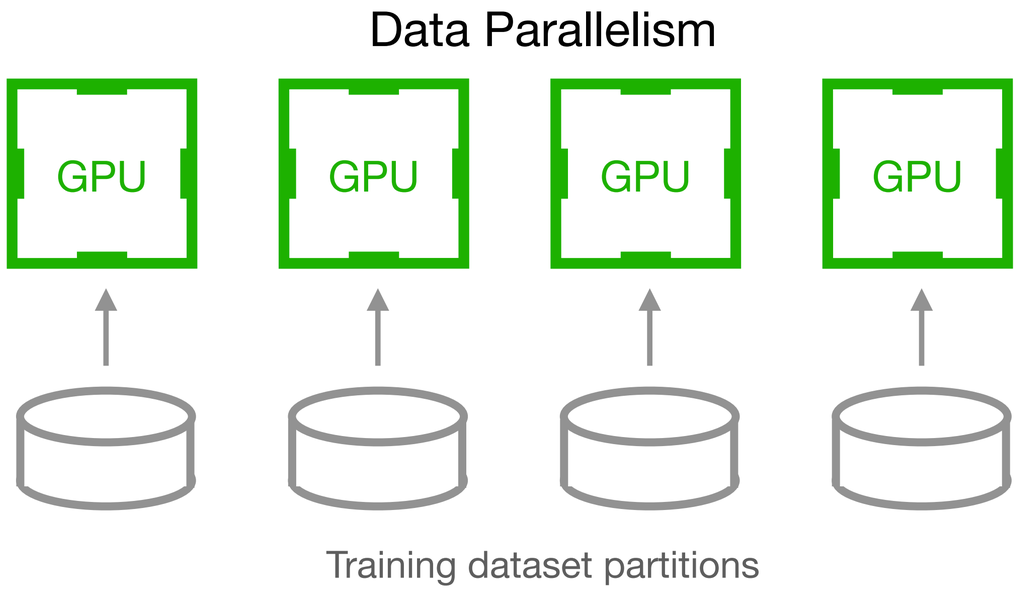
ZEN: Empowering Distributed Training with Sparsity-driven Data Synchronization

Zhuang Wang, Zhaozhuo Xu, Jingyi Xi, Yuke Wang
Anshumali Shrivastava, T. S. Eugene Ng

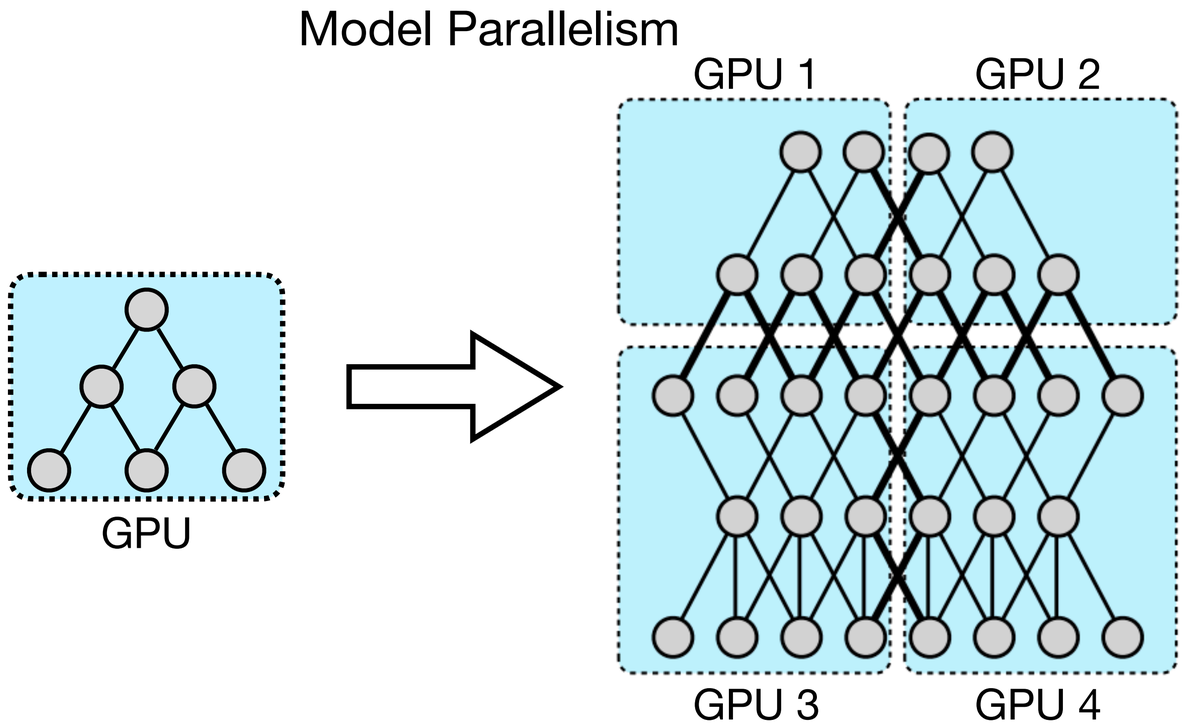
Trends in Deep Learning

- Data grows exponentially
  - Data Parallelism



Trends in Deep Learning

- Data grows exponentially
  - Data Parallelism
- Model size grows exponentially
  - Scale to trillion parameters
  - Model Parallelism: TP, PP, FSDP



Trends in Deep Learning

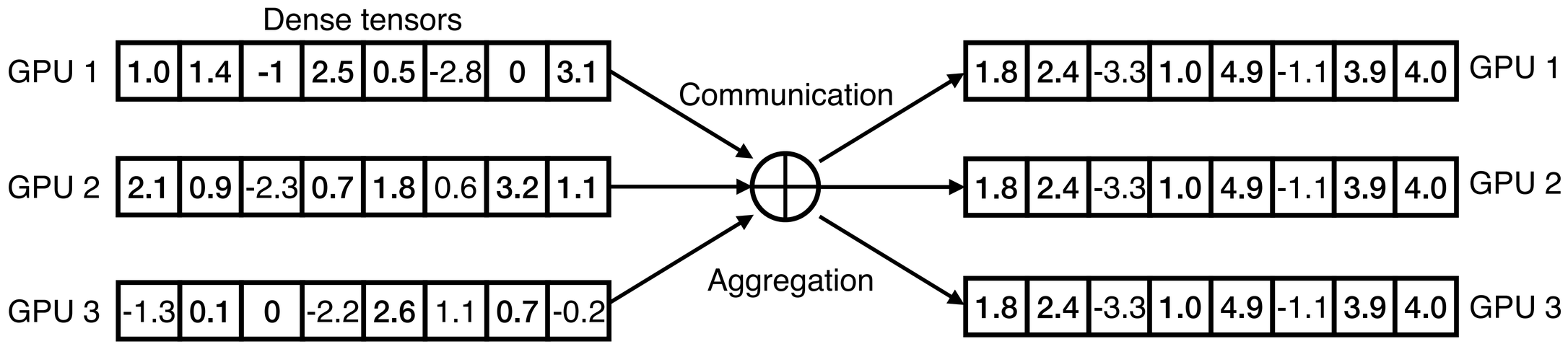
Distributed training

- Data grows exponentially
  - Data Parallelism
- Model size grows exponentially
  - Scale to trillion parameters
  - Model Parallelism: TP, PP, FSDP
- Hybrid parallelism for distributed training
  - Data Parallelism + Model Parallelism

Gradient synchronization

Dense tensor synchronization among GPUs

- Communication and aggregation



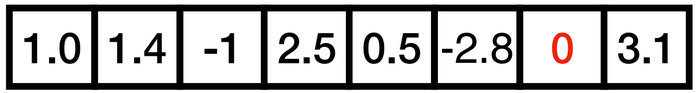
- Synchronization of dense tensors has been well studied
  - BytePS [1] and Ring-AllReduce [2]

[1] A unified architecture for accelerating distributed DNN training in heterogeneous GPU/CPU clusters, OSDI 2020
[2] Horovod: fast and easy distributed deep learning in TensorFlow, arXiv 2018

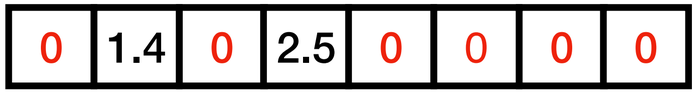
Sparsity in gradient tensors

Zero gradients in tensors

- Gradients can be zeros



Low sparsity



High sparsity

Sparsity in gradient tensors

Zero gradients in tensors

- Gradients can be zeros
- Natural tensor sparsity in popular models

Model	MLP Size	Embedding Size	Embedding Sparsity
LSTM	20M	406M	98.87%
DeepFM	68M	214M	97.20%
NMT	31M	112M	97.53%

Sparsity in gradient tensors

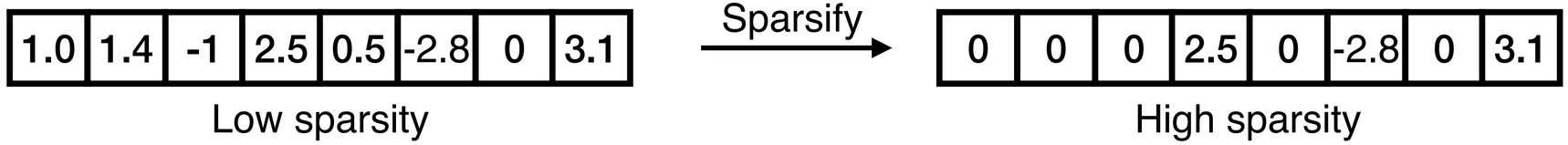
Zero gradients in tensors

- Gradients can be zeros
- Natural tensor sparsity in popular models

Model	MLP Size	Embedding Size	Embedding Sparsity
LSTM	20M	406M	98.87%
DeepFM	68M	214M	97.20%
NMT	31M	112M	97.53%

- Tensor sparsity from gradient sparsification algorithms [1,2,3]

Up to 99% sparsity

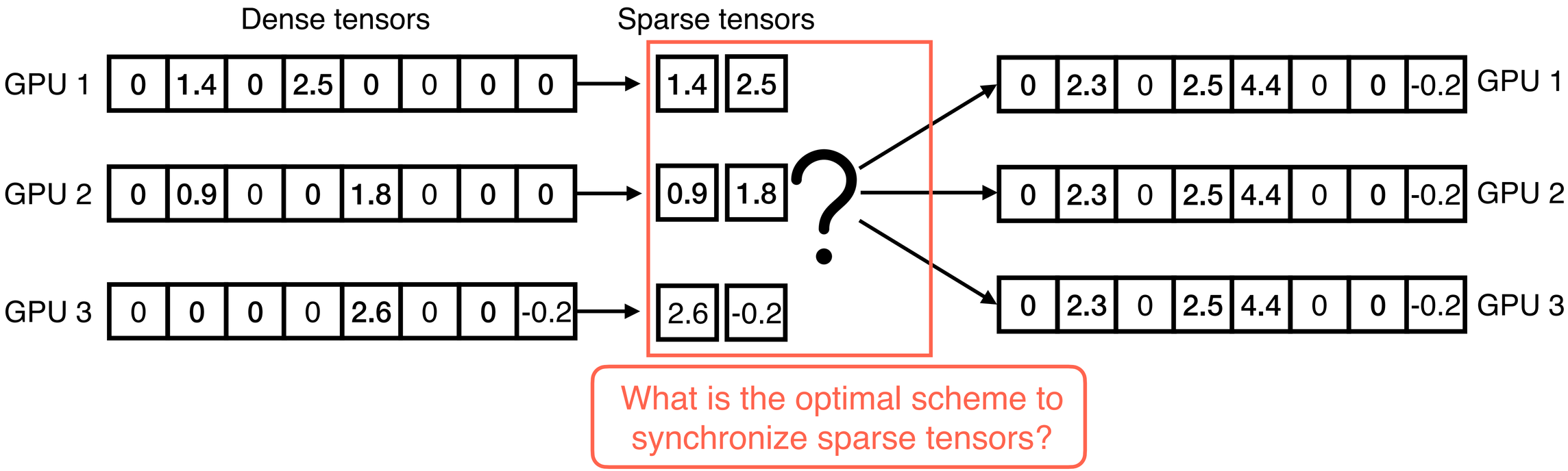


[1] Sparse communication for distributed gradient descent, ACL 2017
[2] Deep gradient compression: Reducing the communication bandwidth for distributed training, ICLR 2018
[3] Sparsified SGD with Memory, NeurPS 2018

Synchronization of sparse tensors

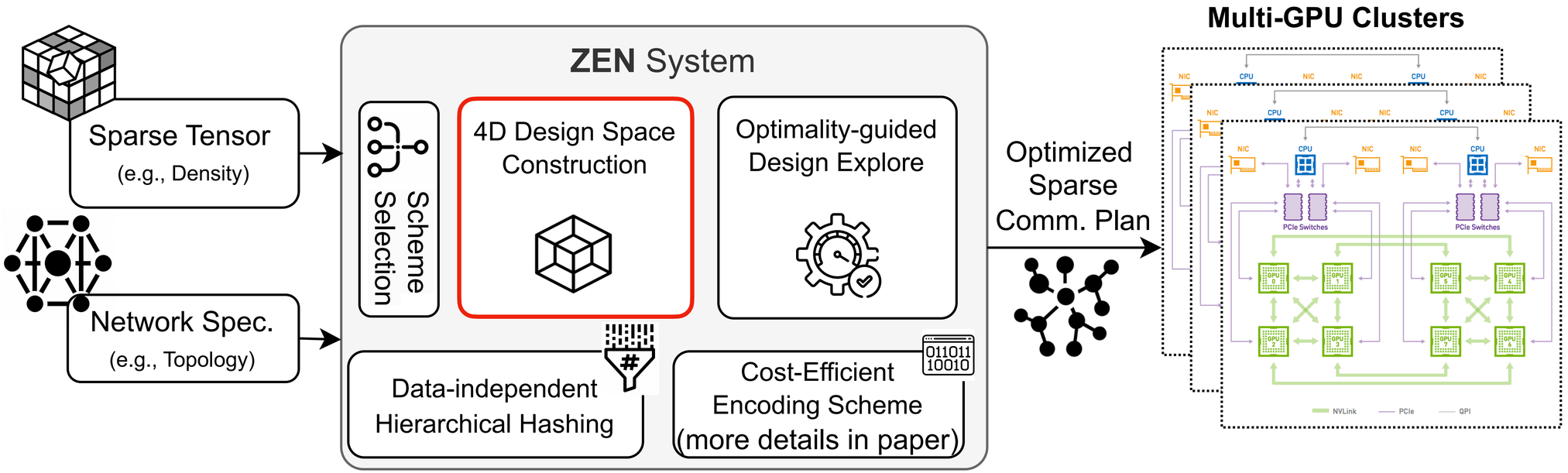
Opportunities and Research Problem

- Only communicate sparse tensors, i.e., non-zero gradients
  - Greatly reduces traffic volume and synchronization time



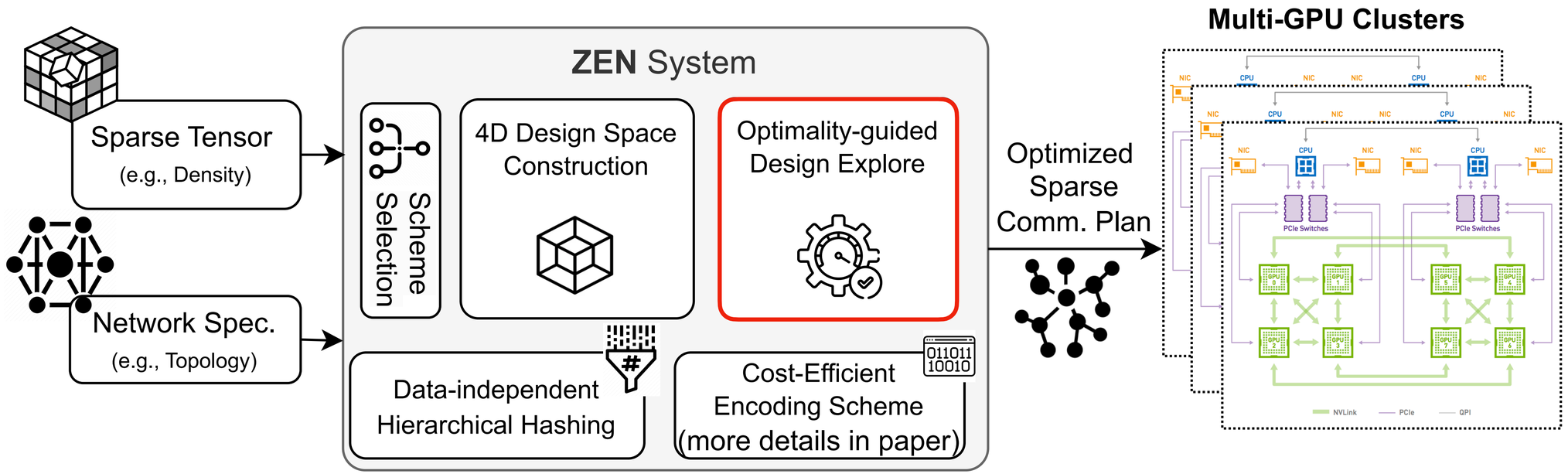
ZEN

A near-optimal synchronization system



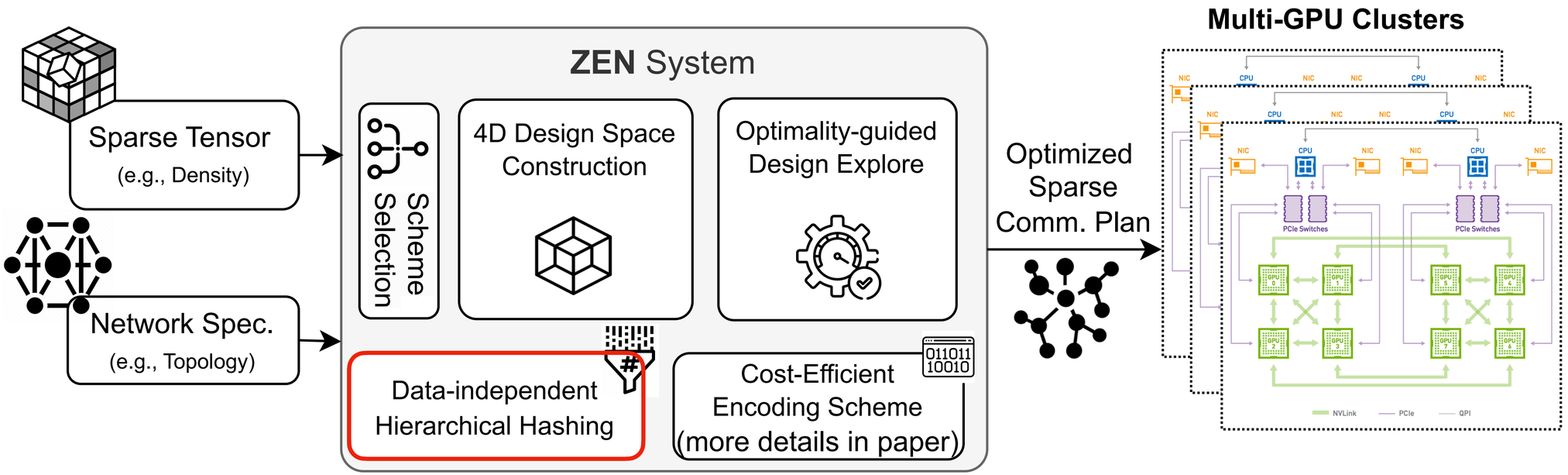
ZEN

A near-optimal synchronization system



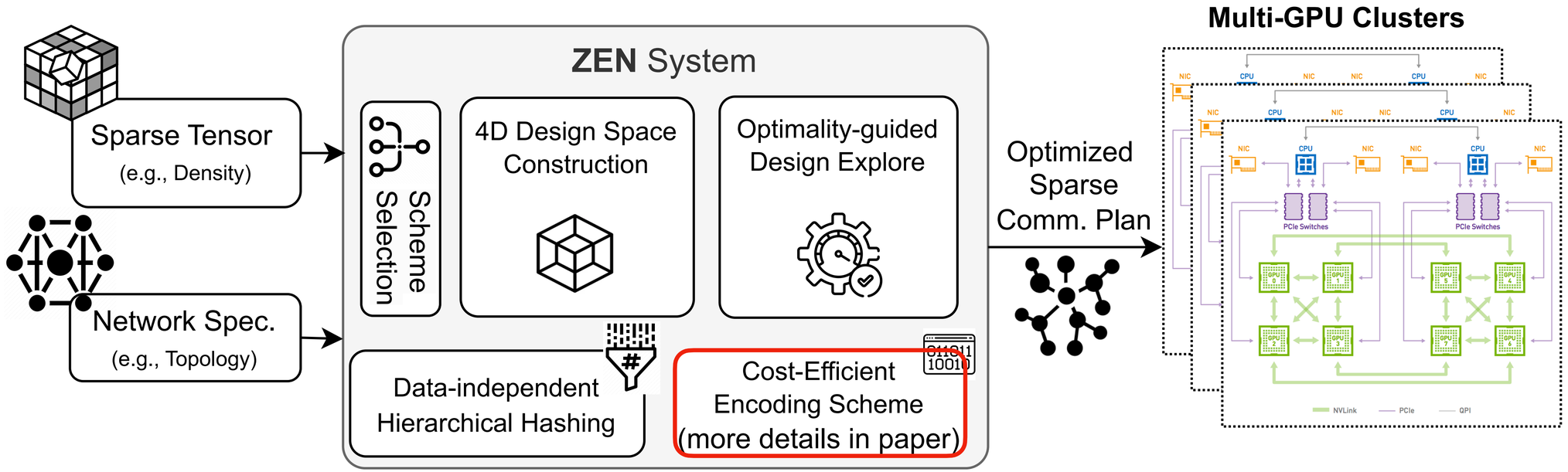
ZEN

A near-optimal synchronization system



ZEN

A near-optimal synchronization system



Design space construction

Four elemental dimensions

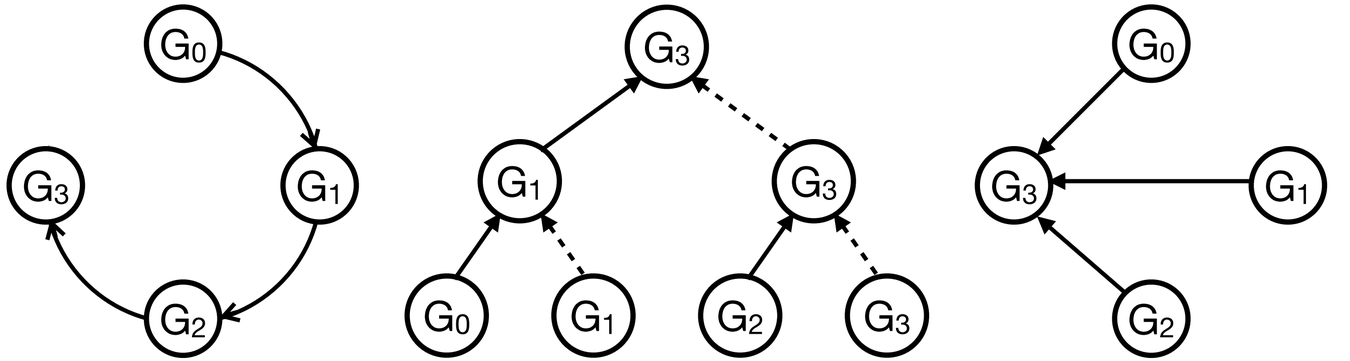
- Sparse tensor before synchronization

Design space construction

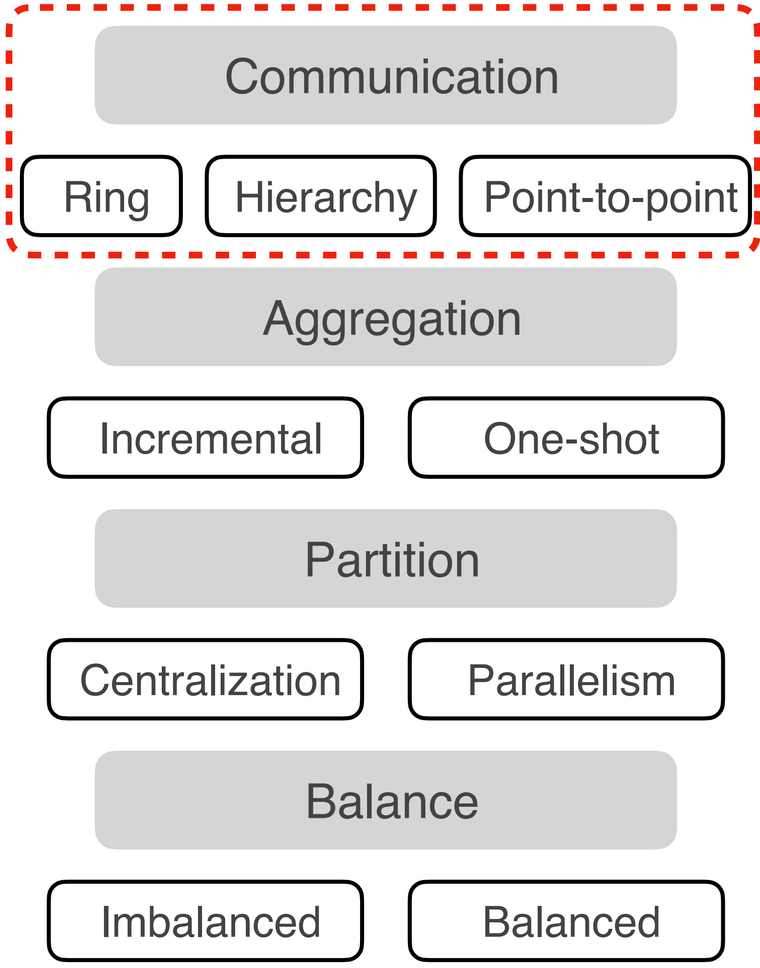
Four elemental dimensions

- Sparse tensor before synchronization

- Communication dimension



(a) Ring (b) Hierarchy (c) Point-to-point

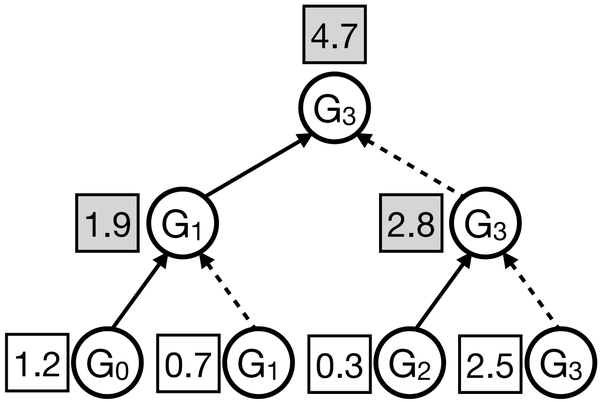


Design space construction

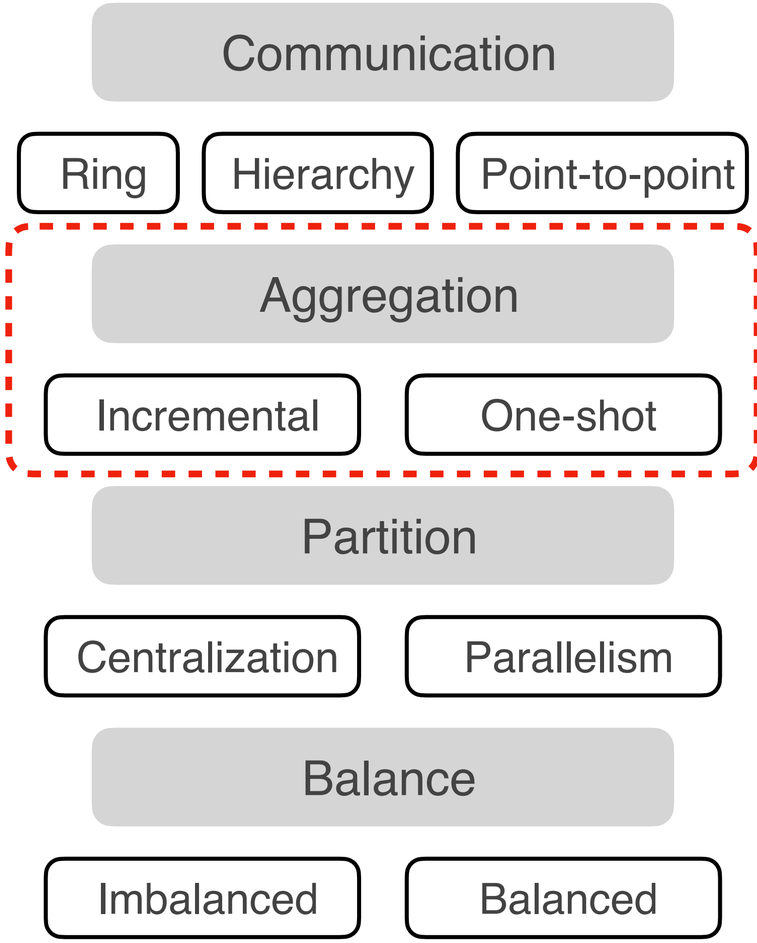
Four elemental dimensions

- Sparse tensor before synchronization

- Aggregation dimension



(a) Incremental (b) One-shot

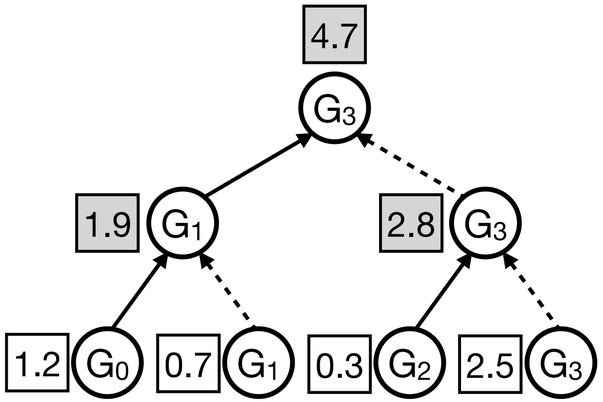


Design space construction

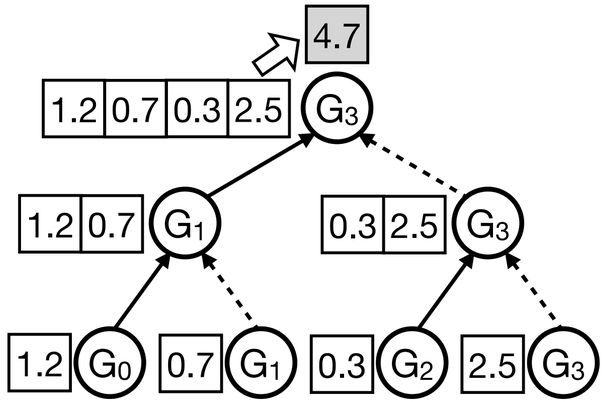
Four elemental dimensions

- Sparse tensor before synchronization

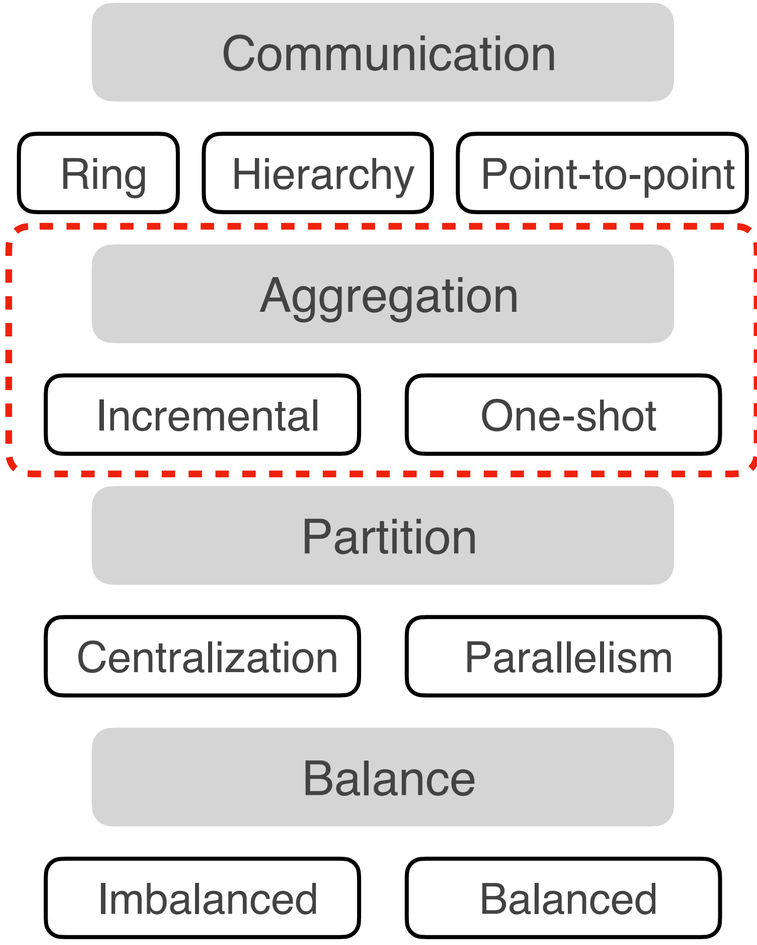
- Aggregation dimension



(a) Incremental



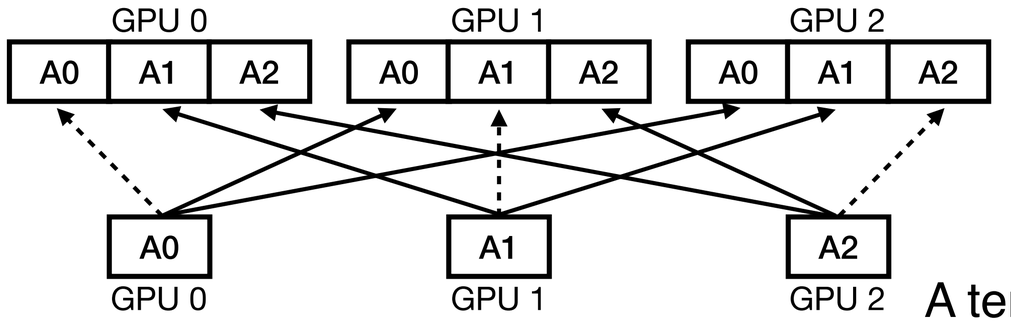
(b) One-shot



Design space construction

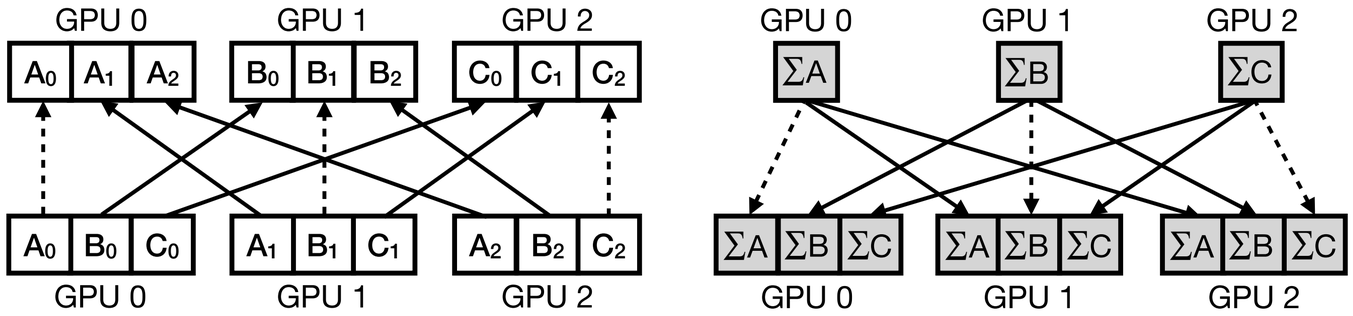
Four elemental dimensions

- Partition dimension



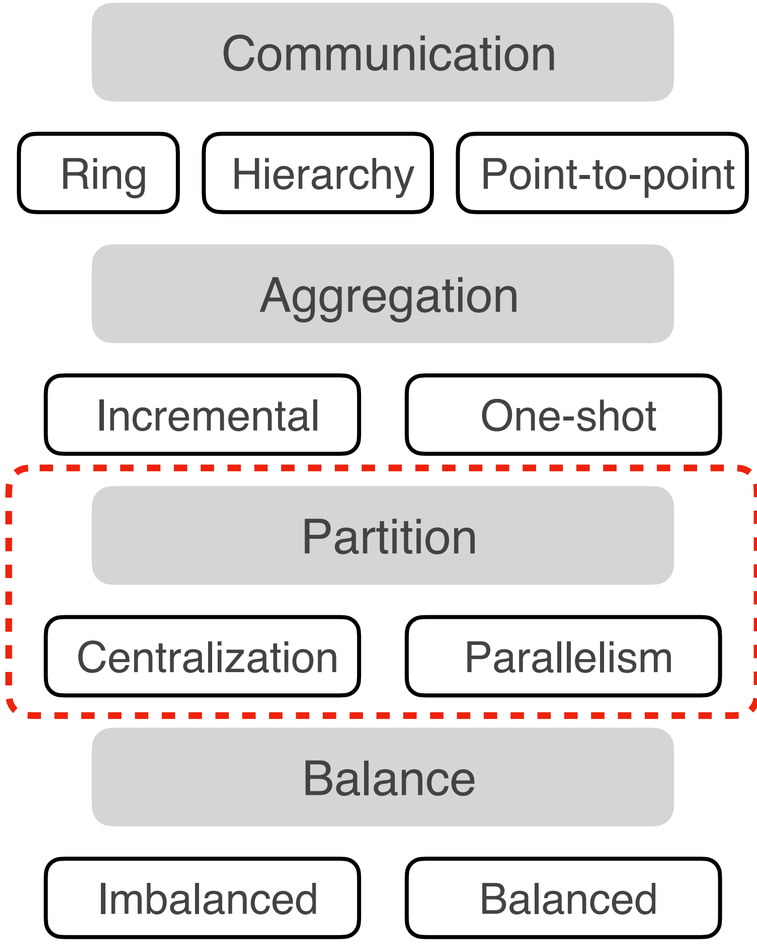
(a) Centralization

A tensor is communicated as a whole



(b) Parallelism

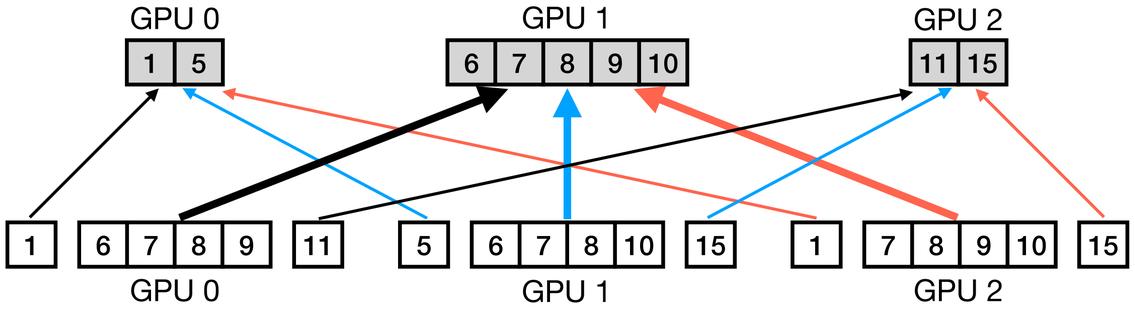
A tensor is partitioned before communication



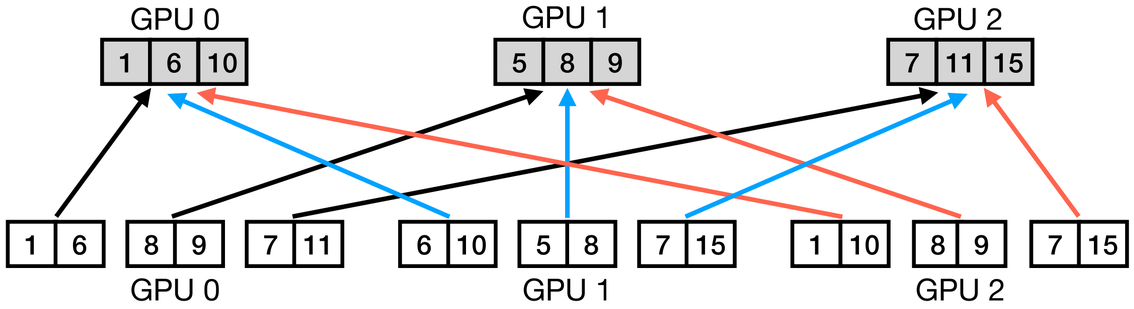
Design space construction

Four elemental dimensions

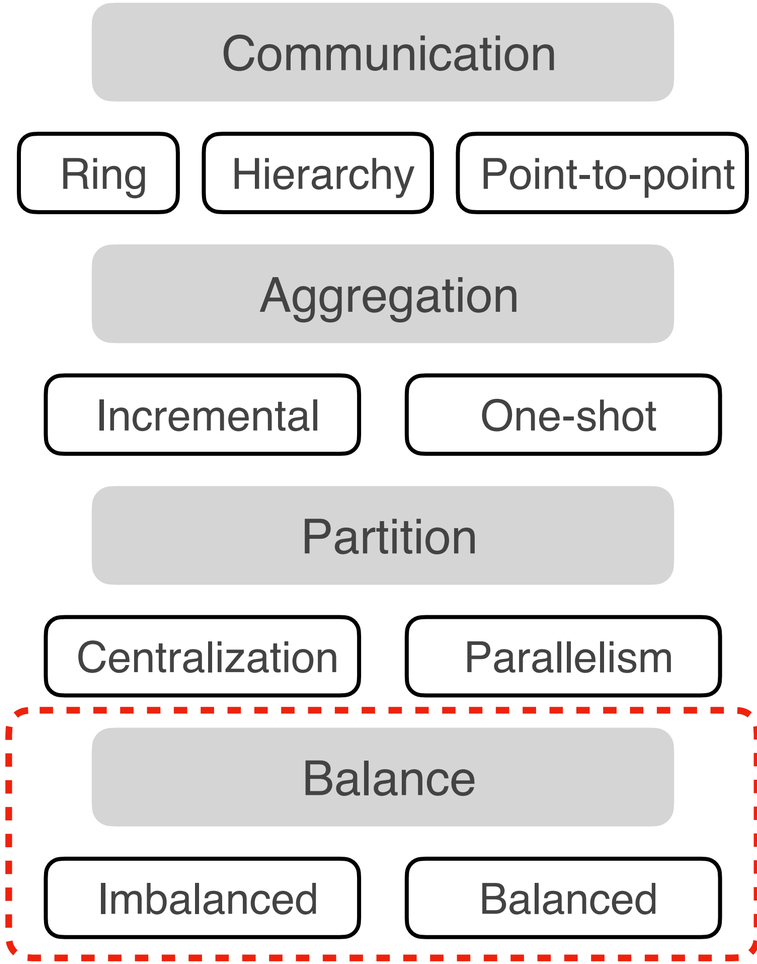
- Balance dimension



(a) Imbalanced communication



(b) Balanced communication



Design space construction

Expressiveness

- Existing schemes can be described by the four dimensions

Schemes	Communication	Aggregation	Partition	Balance
AGsparse [1]	Ring, Hierarchy, Point-to-point	One-shot	Centralization	N/A
SparCML [2]	Hierarchy	Incremental	Centralization	N/A
OmniReduce [3]	Point-to-point	One-shot	Parallelism	Imbalanced

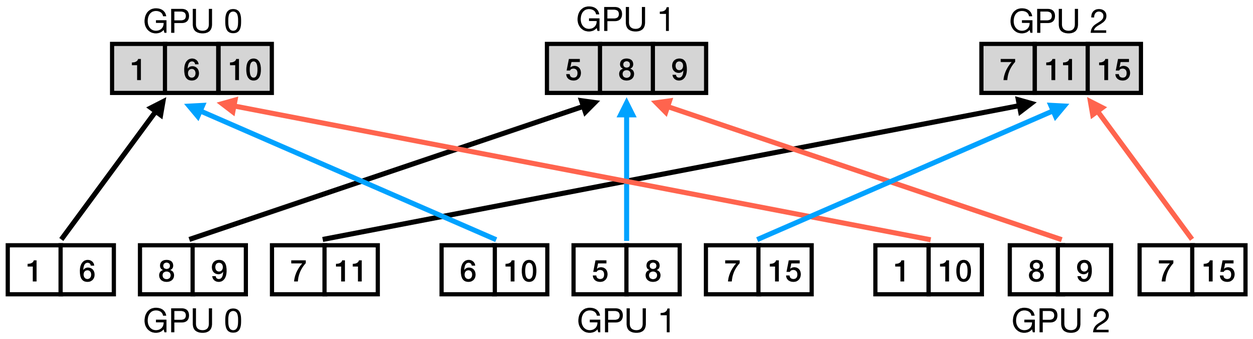
- The four dimensions can describe the whole design space

[1] Pytorch distributed: Experiences on accelerating data parallel, VLDB 2020
[2] SparCML: High-performance sparse communication for machine learning, SC 2019
[3] Efficient sparse collective communication and its application, SIGCOMM 2021

Find the optimal schemes

Based on the four dimensions

- “Balanced Parallelism” (our proposal)



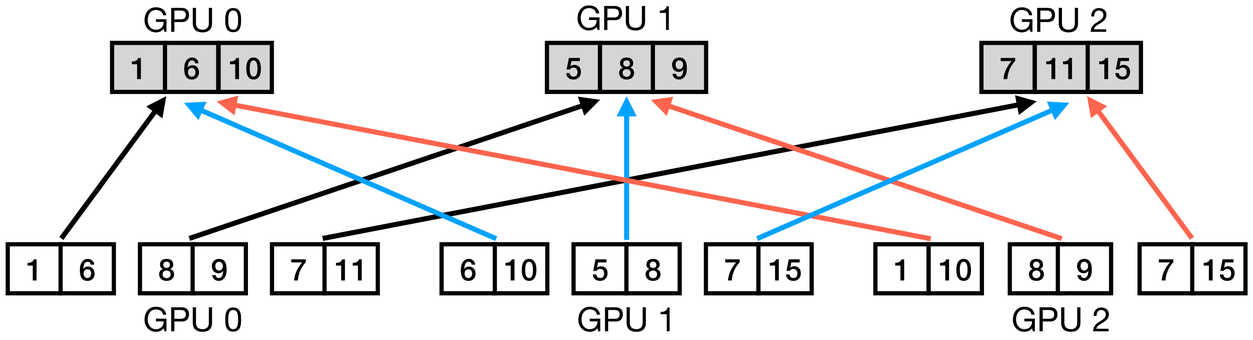
“Balanced Parallelism”

- Communication
  - Ring
  - Hierarchy
  - Point-to-point
- Aggregation
  - Incremental
  - One-shot
- Partition
  - Centralization
  - Parallelism
- Balance
  - Imbalanced
  - Balanced

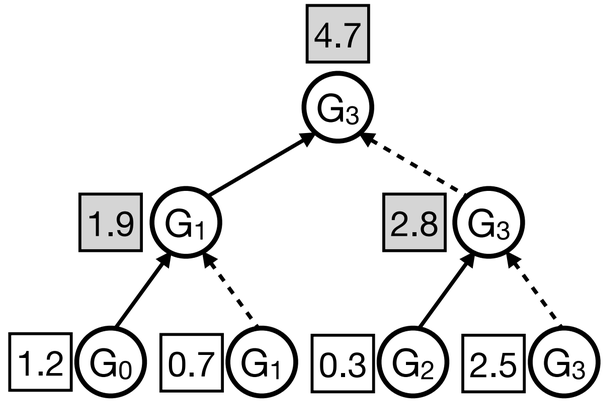
Find the optimal schemes

Based on the four dimensions

- “Balanced Parallelism” (our proposal)



- “Hierarchical Centralization” (SparCML [1])



Balanced Parallelism as the practical optimal scheme

“Hierarchical Centralization”

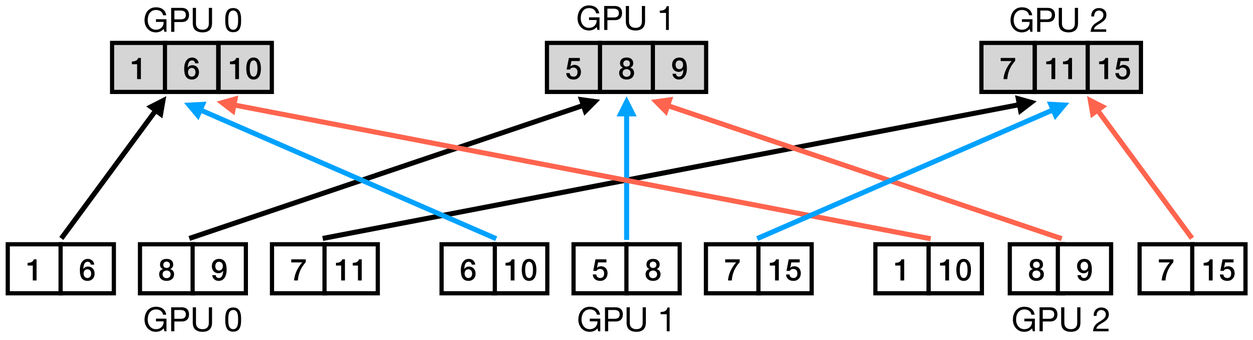
- Communication
  - Ring
  - Hierarchy
  - Point-to-point
- Aggregation
  - Incremental
  - One-shot
- Partition
  - Centralization
  - Parallelism
- Balance
  - Imbalanced
  - Balanced

[1] SparCML: High-performance sparse communication for machine learning, SC 2019

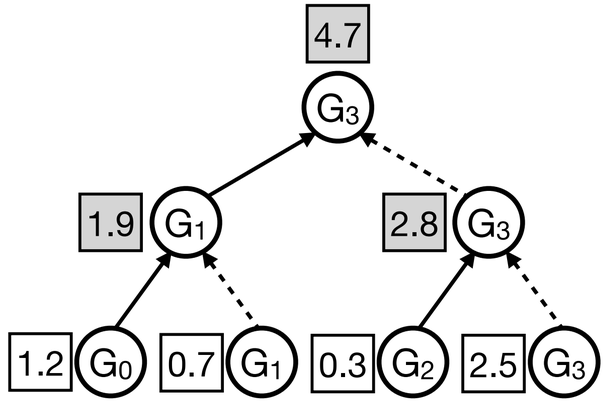
Find the optimal schemes

Based on the four dimensions

- “Balanced Parallelism” (our proposal)



- “Hierarchical Centralization” (SparCML [1])



No existing realization of Balanced Parallelism

“Hierarchical Centralization”

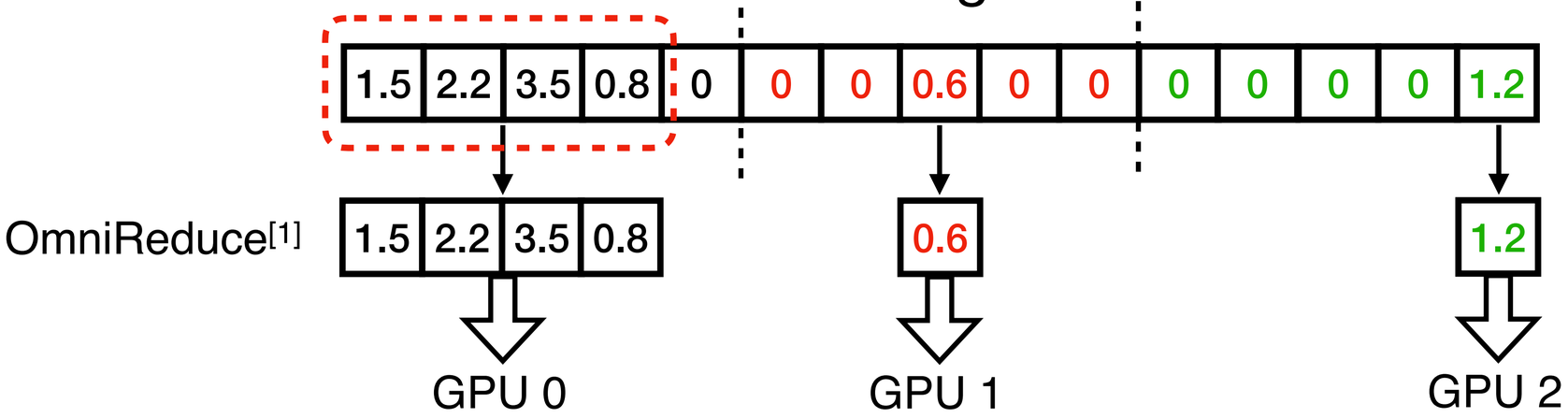
- Communication
  - Ring
  - Hierarchy
  - Point-to-point
- Aggregation
  - Incremental
  - One-shot
- Partition
  - Centralization
  - Parallelism
- Balance
  - Imbalanced
  - Balanced

[1] SparCML: High-performance sparse communication for machine learning, SC 2019

How to realize “Balanced Parallelism”?

Imbalanced communication

- Skewed distribution of non-zero gradients



imbalanced communications

Communication

Point-to-point

Aggregation

One-shot

Partition

Parallelism

Balance

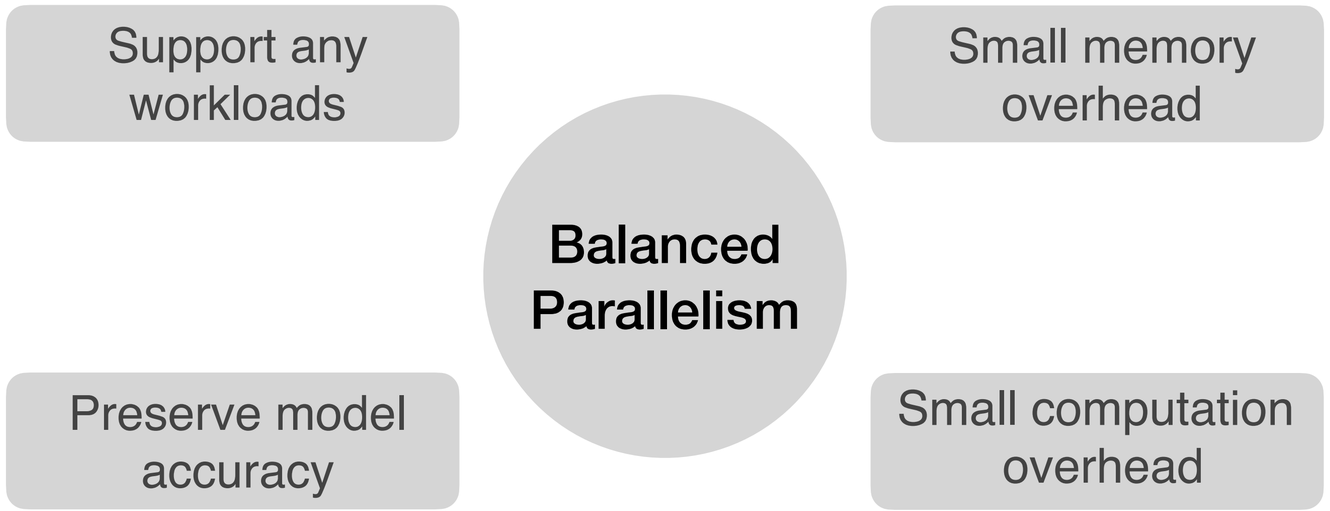
Balanced

[1] Efficient sparse collective communication and its application, SIGCOMM 2021

How to realize “Balanced Parallelism”?

Challenge and requirements

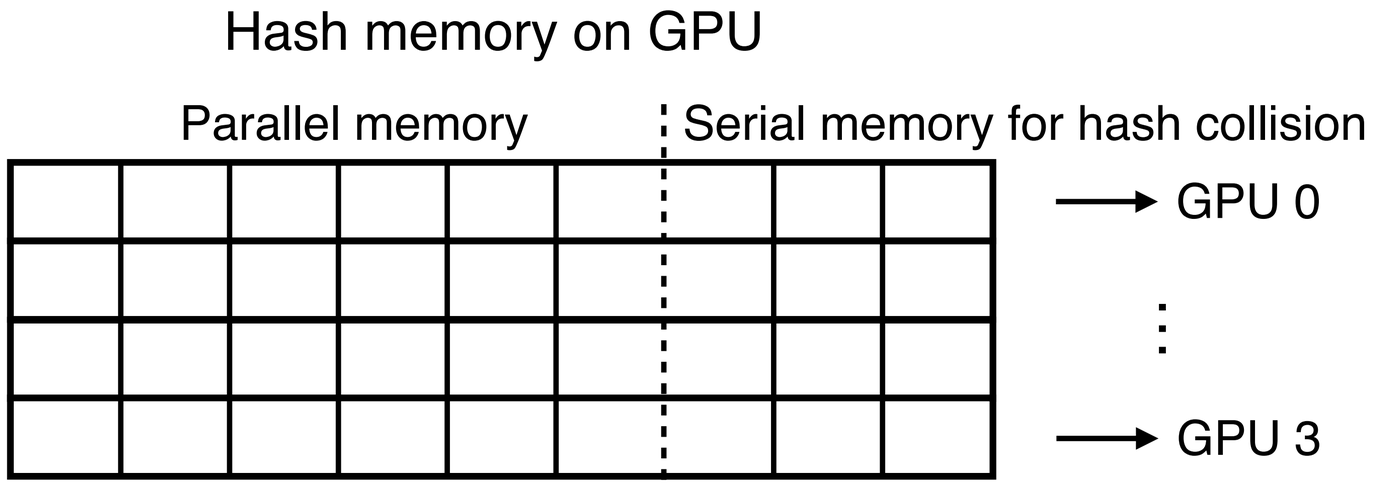
- Challenge: How to achieve balanced communications?
- Requirements



A hierarchical hashing algorithm

Balanced Parallelism on GPUs

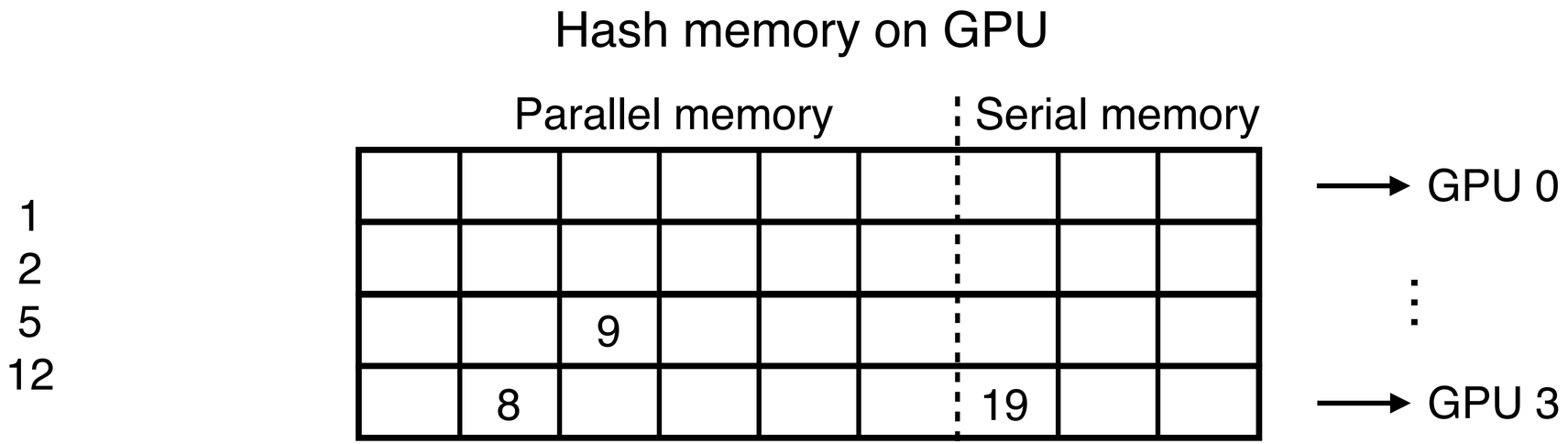
- Technique #1: Communication-oriented hash memory management



A hierarchical hashing algorithm

Balanced Parallelism on GPUs

- Technique #1: Communication-oriented hash memory management

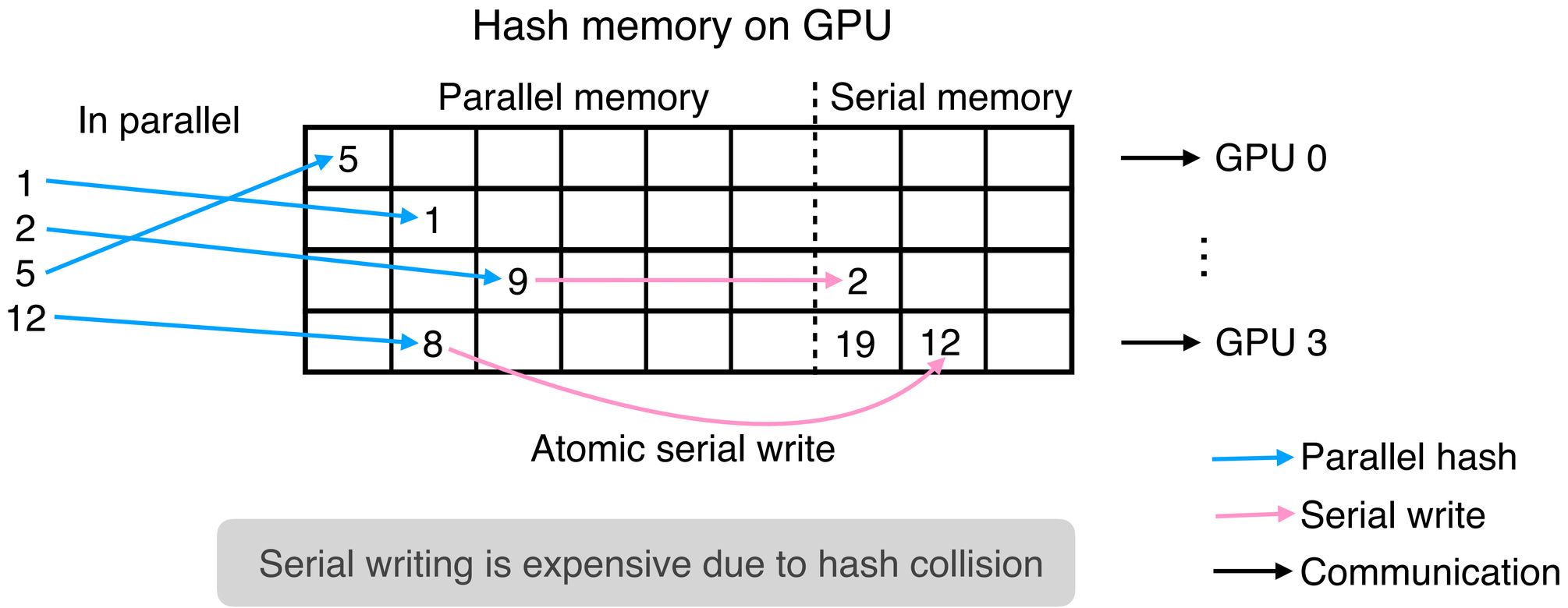


Indices of non-zero gradients

A hierarchical hashing algorithm

Balanced Parallelism on GPUs

- Technique #1: Communication-oriented hash memory management

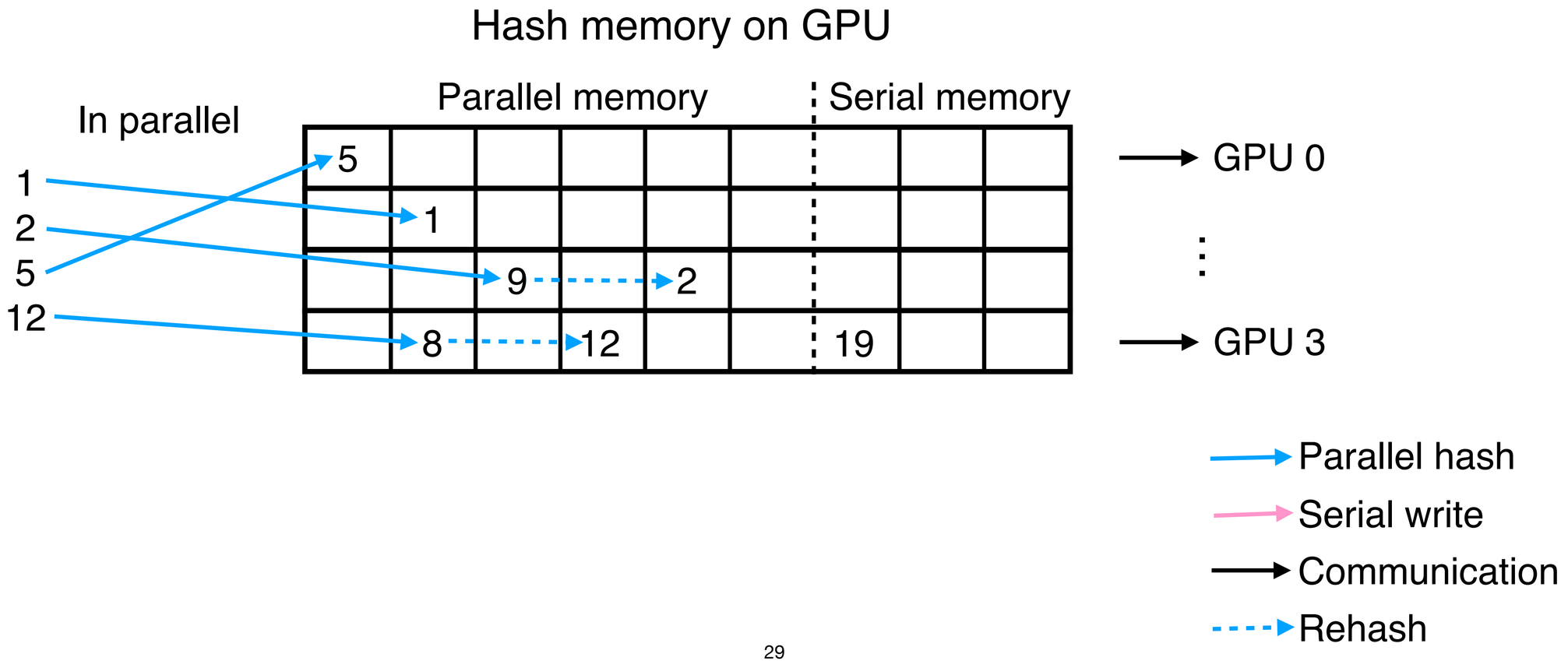


Serial writing is expensive due to hash collision

A hierarchical hashing algorithm

Balanced Parallelism on GPUs

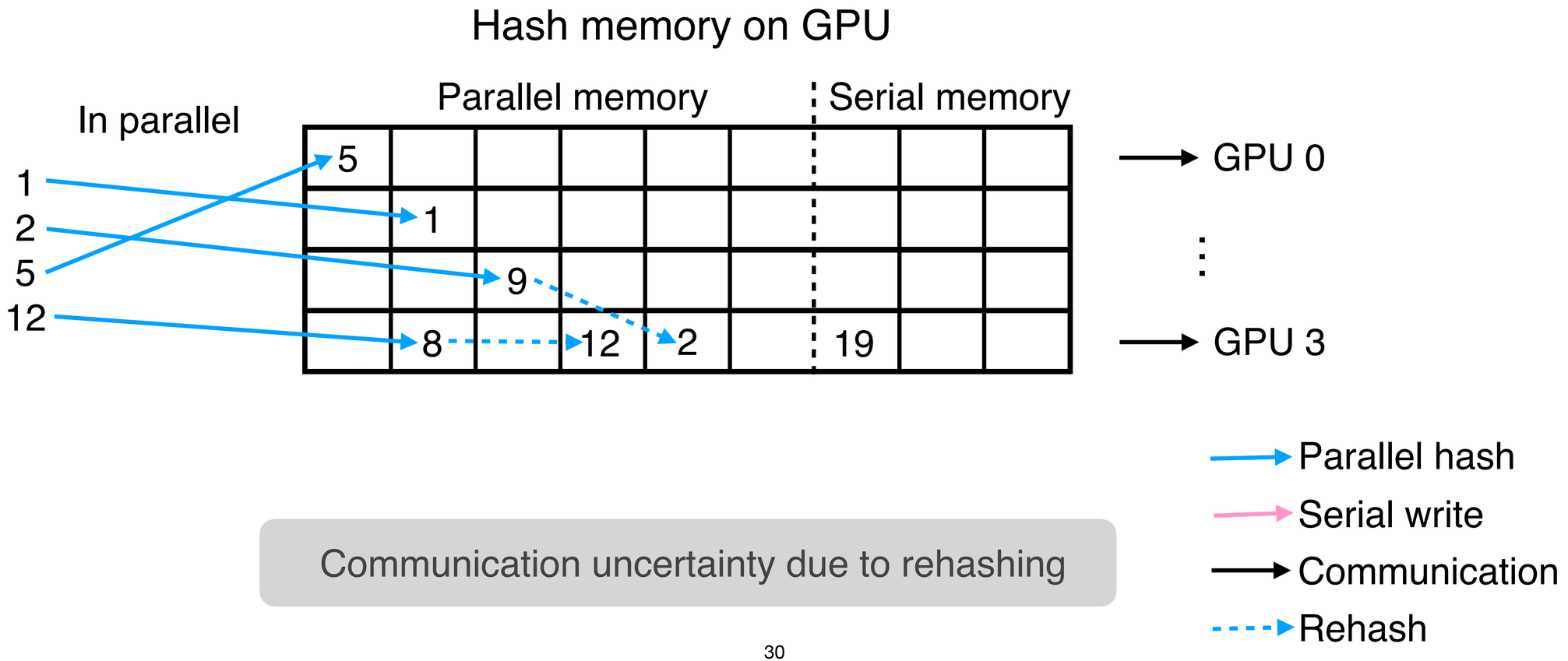
- Technique #2: Multiple hash functions in each GPU thread



A hierarchical hashing algorithm

Balanced Parallelism on GPUs

- Technique #2: Multiple hash functions in each GPU thread

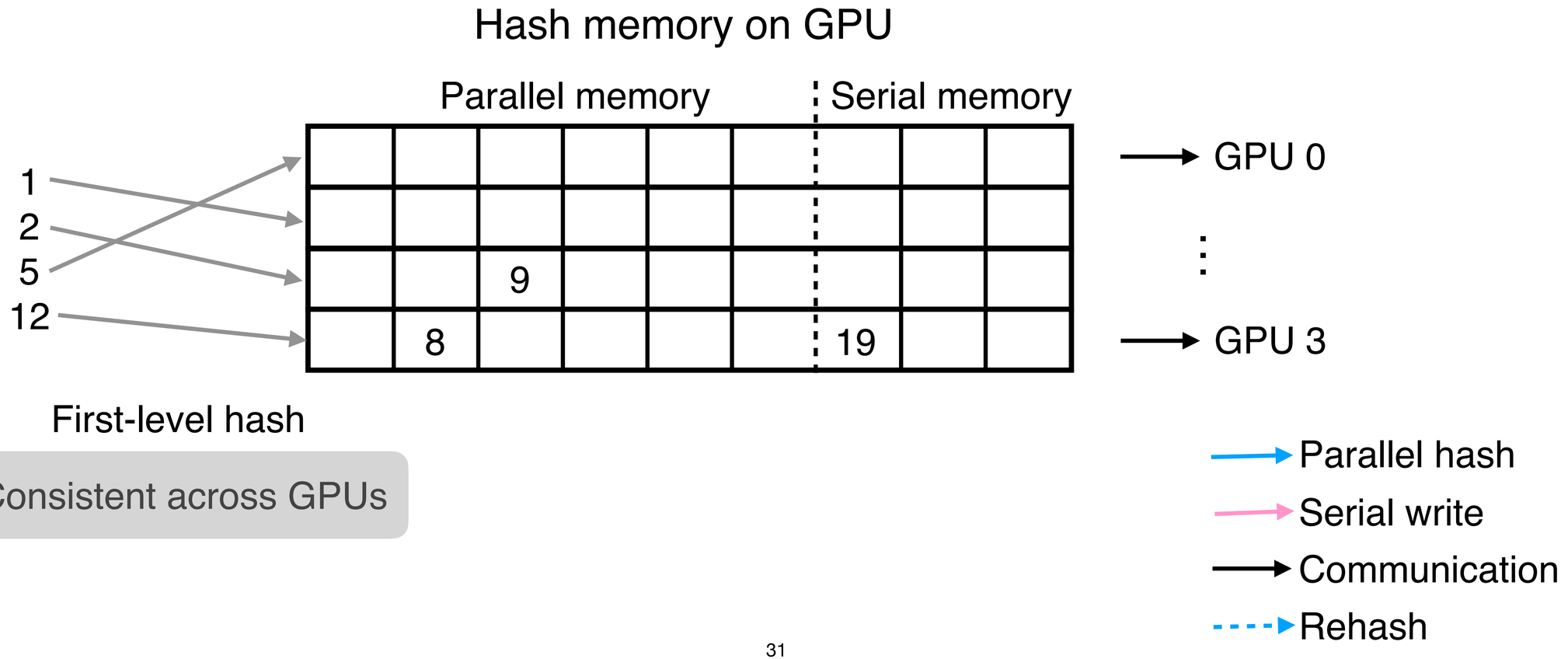


Communication uncertainty due to rehashing

A hierarchical hashing algorithm

Balanced Parallelism on GPUs

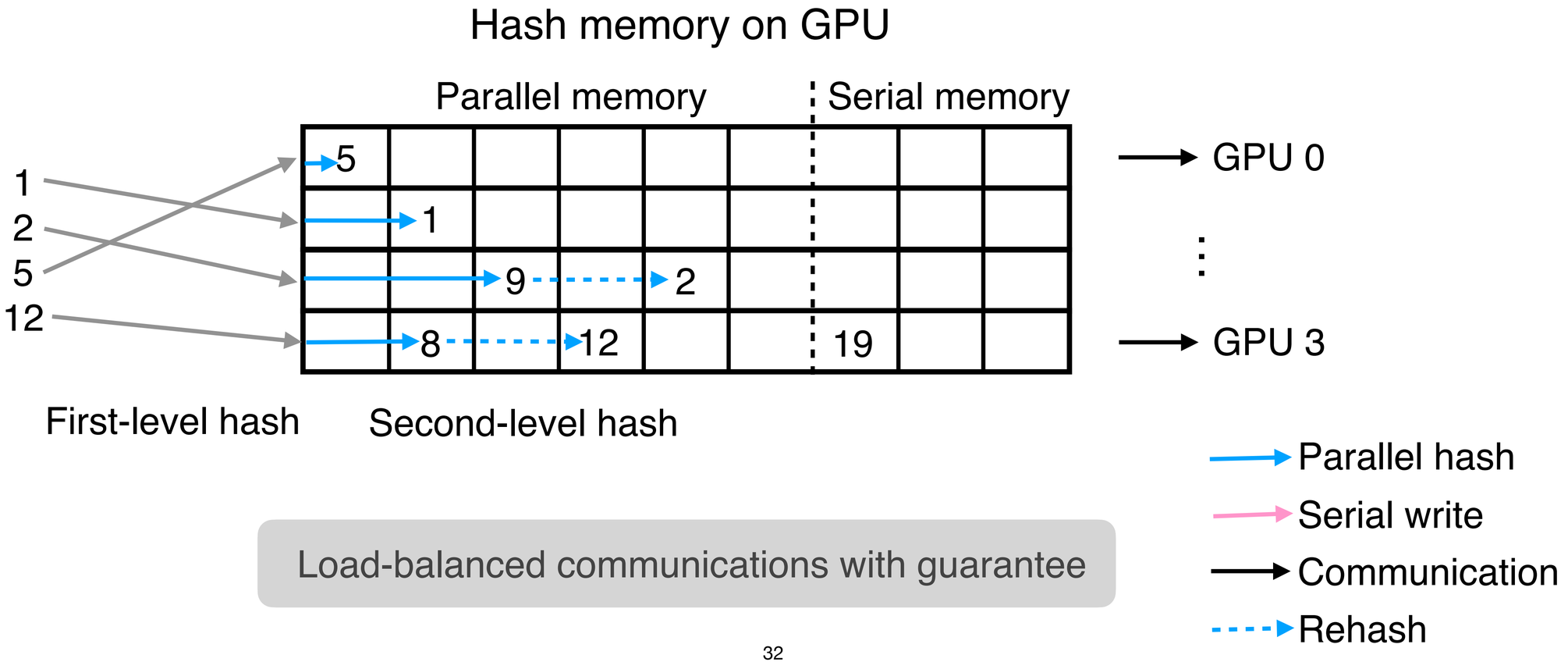
- Technique #3: Hierarchical consistent hashing across GPUs



A hierarchical hashing algorithm

Balanced Parallelism on GPUs

- Technique #3: Hierarchical consistent hashing across GPUs



Load-balanced communications with guarantee

Evaluations

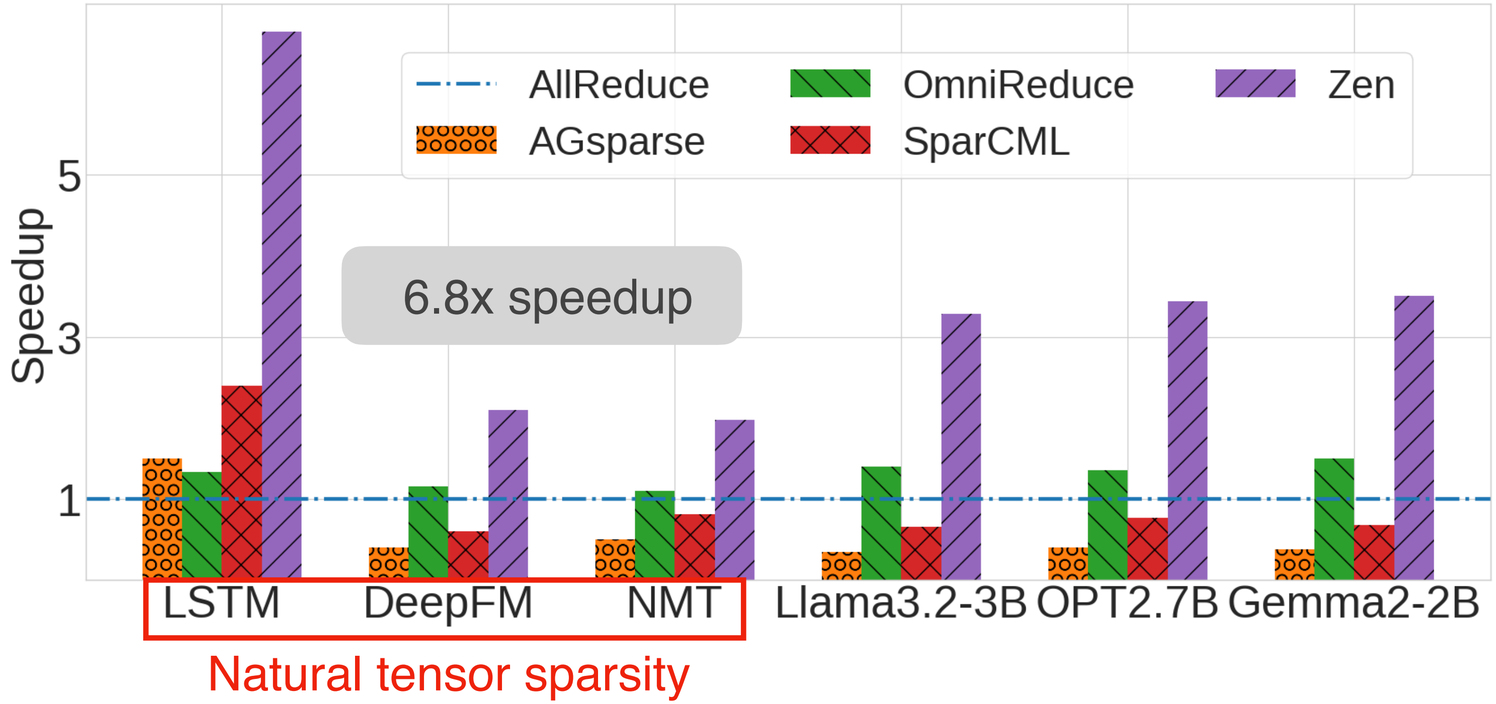
- Two testbeds
  - AWS EC2 p3dn.24xlarge with V100 GPUs, 100 Gbps RDMA networks
  - AWS EC2 p3.16xlarge with V100 GPUs, 25 Gbps TCP/IP networks
- Workloads
  - Models with natural tensor sparsity
  - Models with gradient compression (DGC [1])

[1] Deep gradient compression: Reducing the communication bandwidth for distributed training, ICLR 2018

Communication improvement

128 V100 GPUs

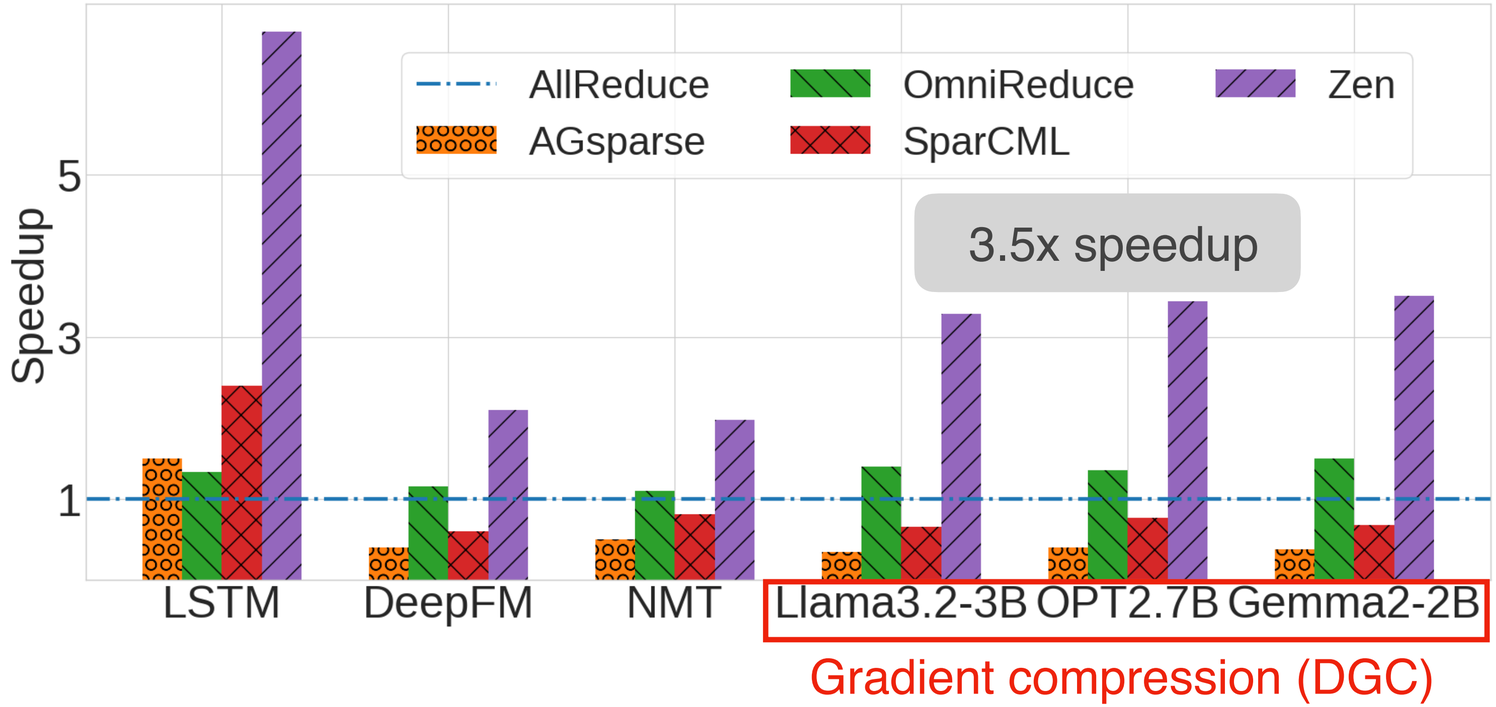
- Communication speedups are normalized to AllReduce



Communication improvement

128 V100 GPUs

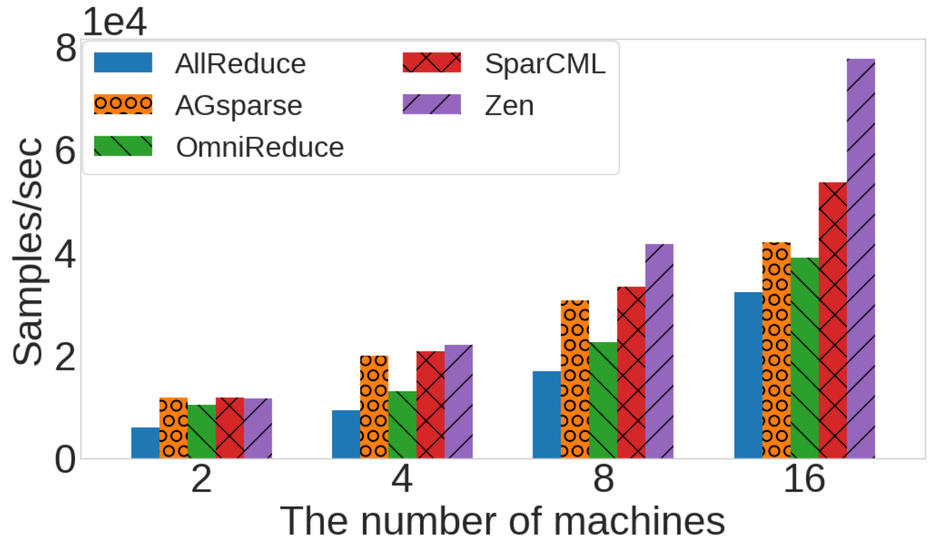
- Communication speedups are normalized to AllReduce



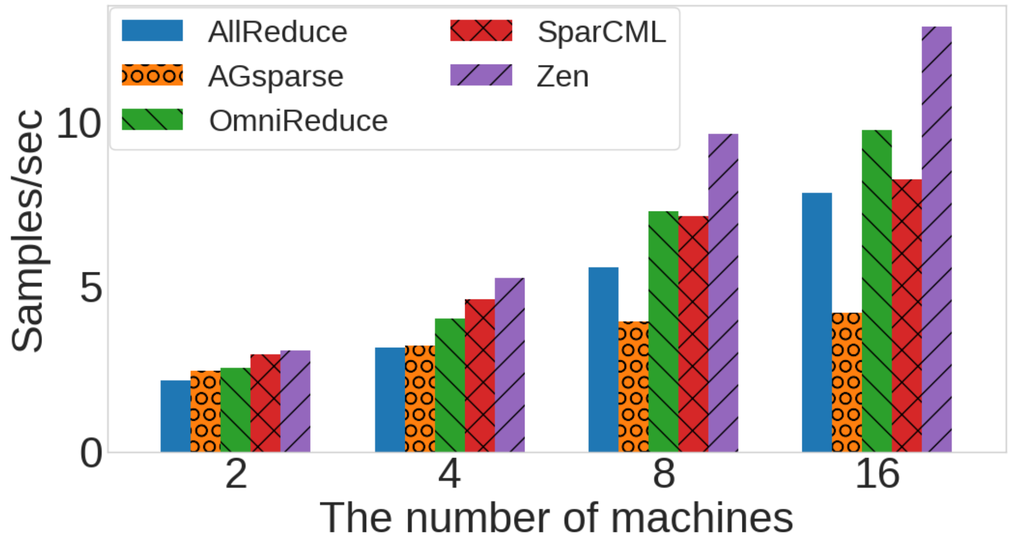
End-to-end training improvement

128 V100 GPUs

- Training throughput



LSTM, natural tensor sparsity
1.5x speedup



Llama3.2-3B, gradient compression
1.3x speedup

Summary

- We study **tensor sparsity** in popular models
- We propose Zen with near-optimal synchronization for sparse tensors
  - find **provably optimal schemes** from the constructed design space
  - propose **a hierarchical hashing algorithm** for efficient realization on GPUs