PyTorch FSDP

Fully Sharded Data Parallel

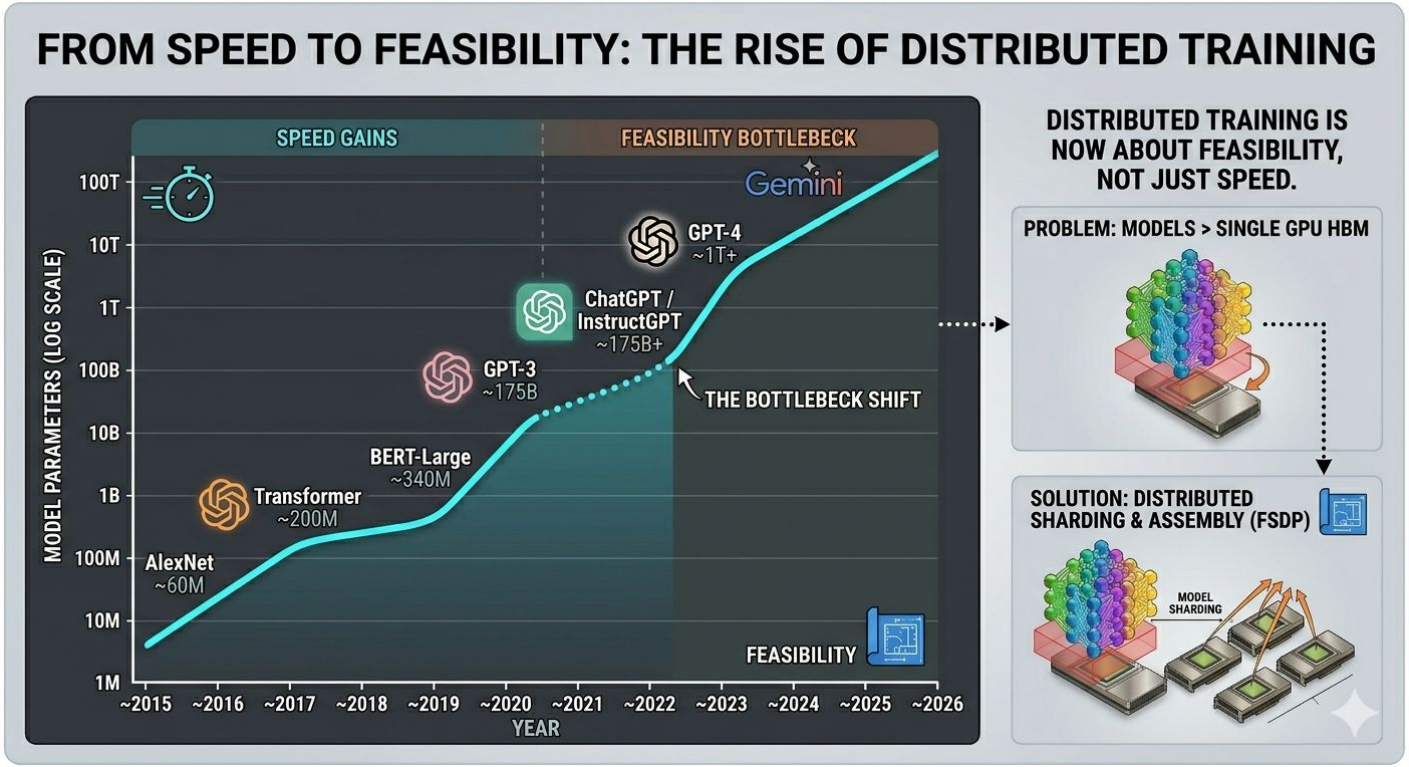
Yanli Zhao, et al. Meta AI

Presenters: Jaeyeon Kim, Kayden Kehe, Theo Lebryk, and Aditya Palaparthi

Presentation Advisors: Qitong Wang and Konstantinos Kopsinis

Why We Need a New System

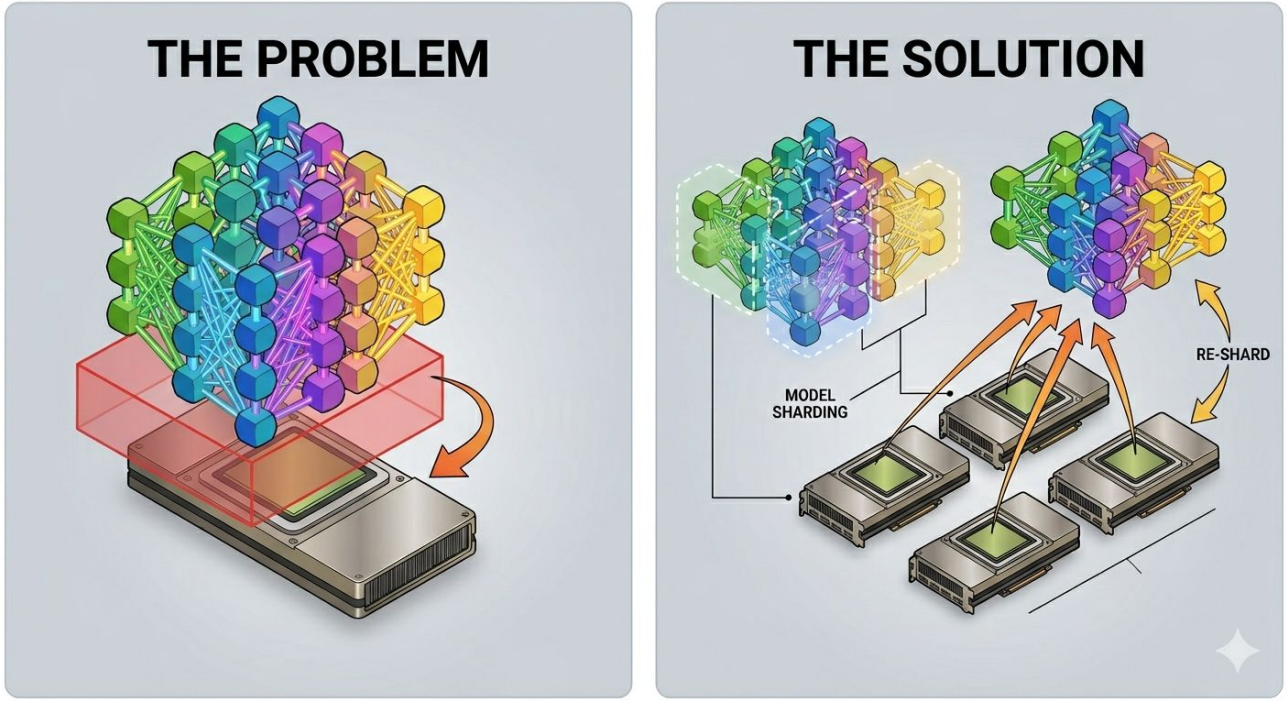
Why Model Scaling Matters



Discussion

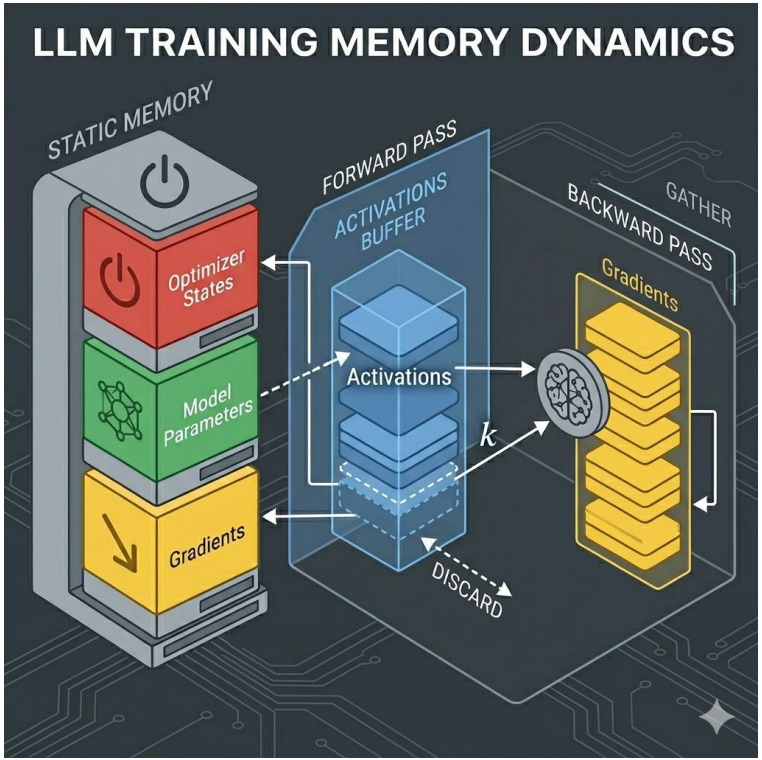
What do you think breaks first when models scale – compute, memory, or communication?

Big Picture



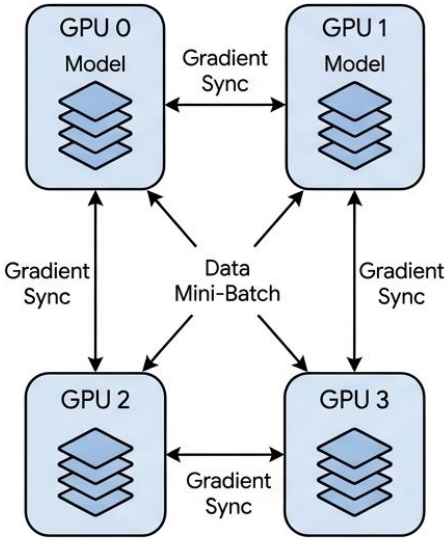
Why Do Existing Methods Break?

Recap on Deep Learning Training



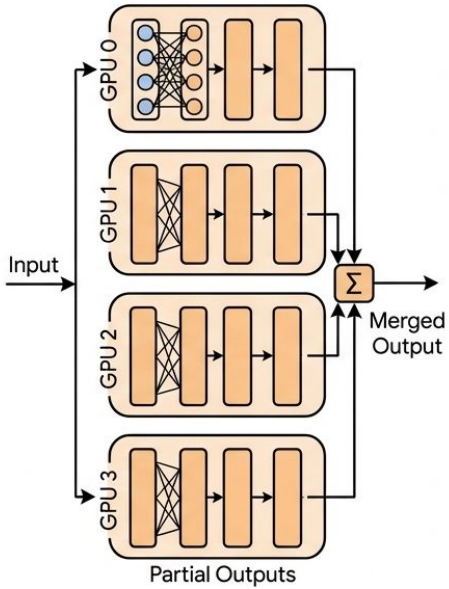
Previous Approaches Break

Data Parallelism (DP)



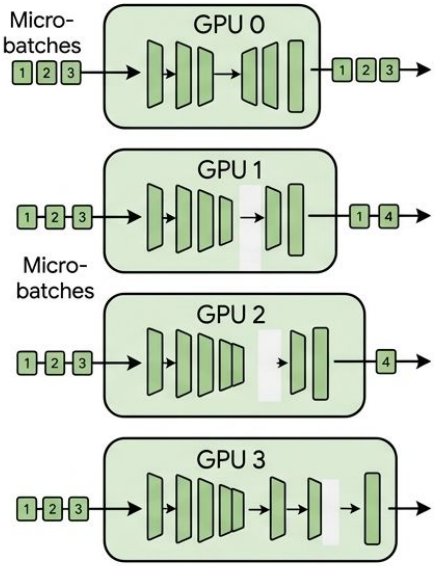
Replicate model, split data

Tensor Parallelism (TP)



Split one layer across GPUs

Pipeline Parallelism (PP)

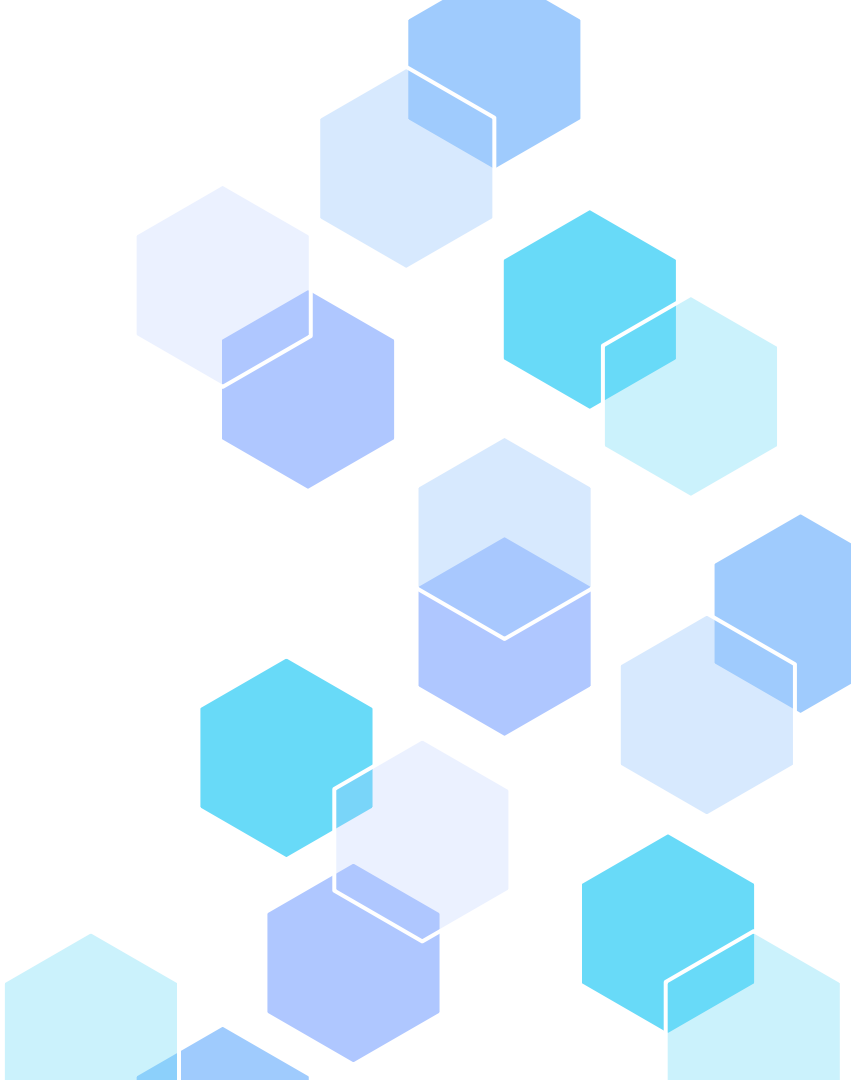


Split layers by stage

Discussion

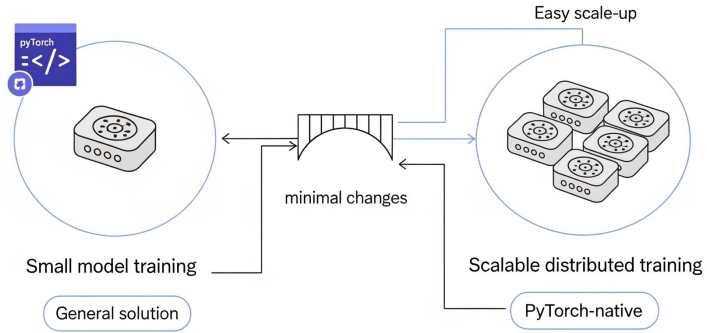
If you had to choose one of these approaches for trillion-parameter models, which would fail first and why?

What Solution Do We Need?

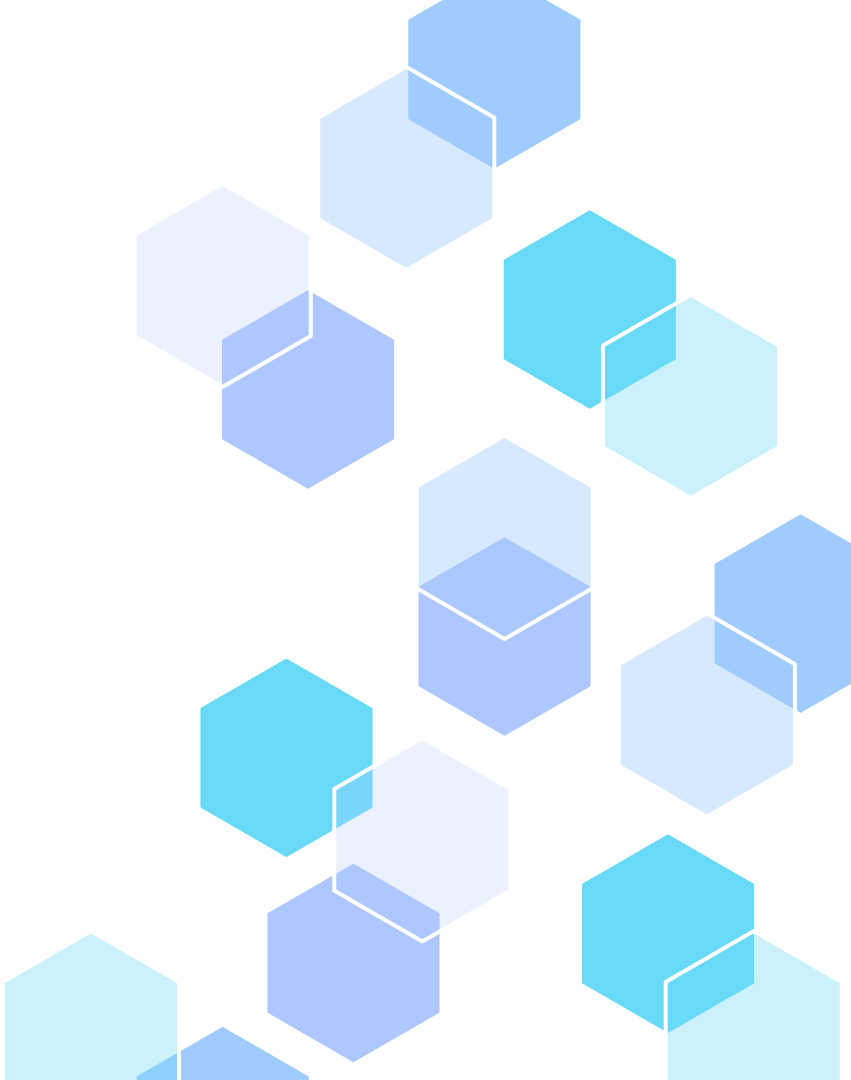


Native Scaling in PyTorch

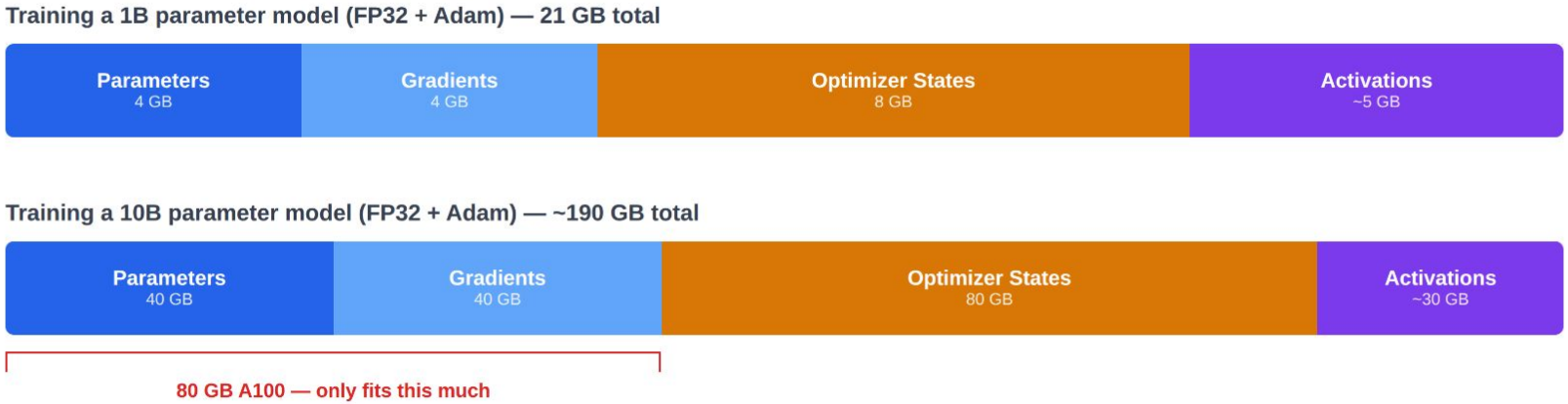
PyTorch-native



Why Is This Difficult?

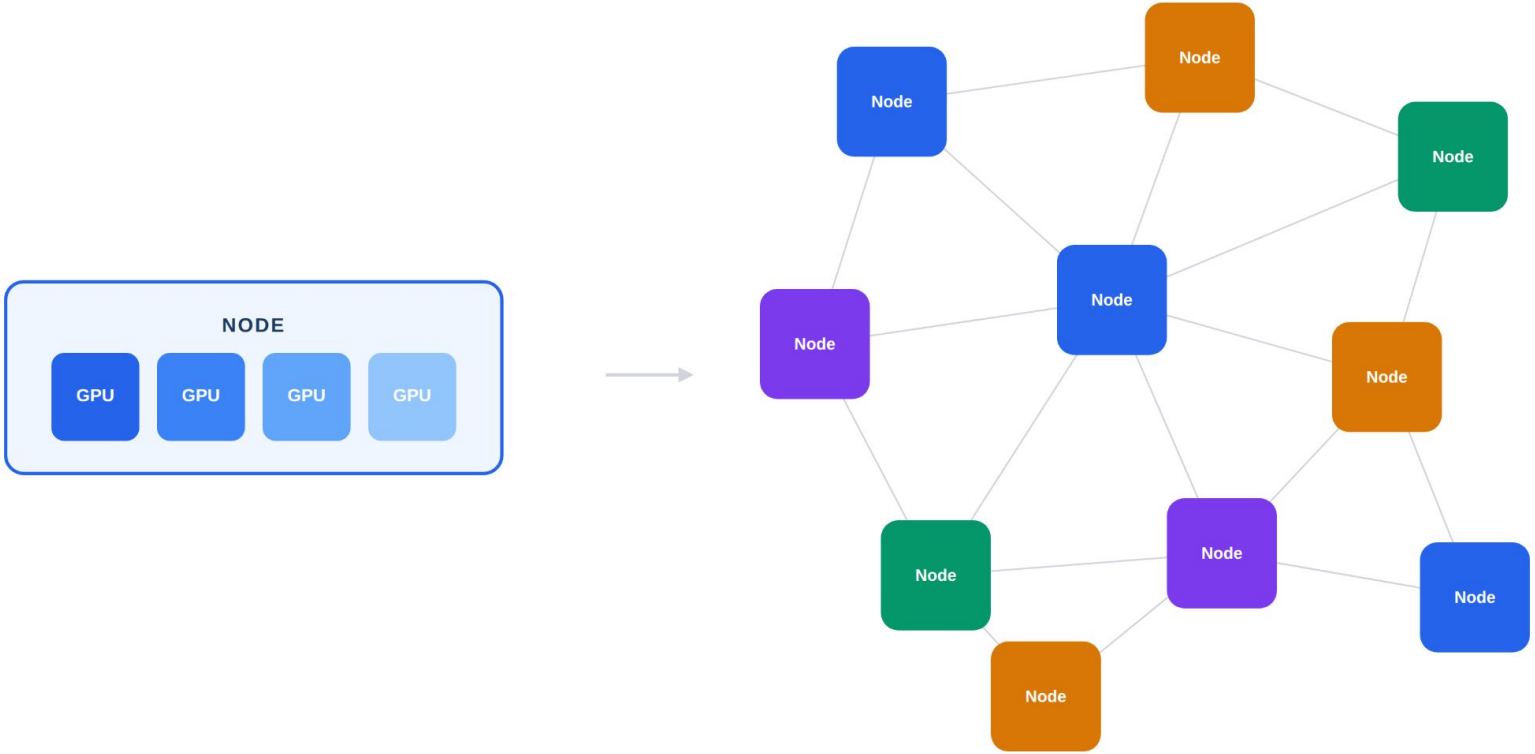


Scaling Up



Some modern frontier LLMs have over ***1 trillion*** parameters

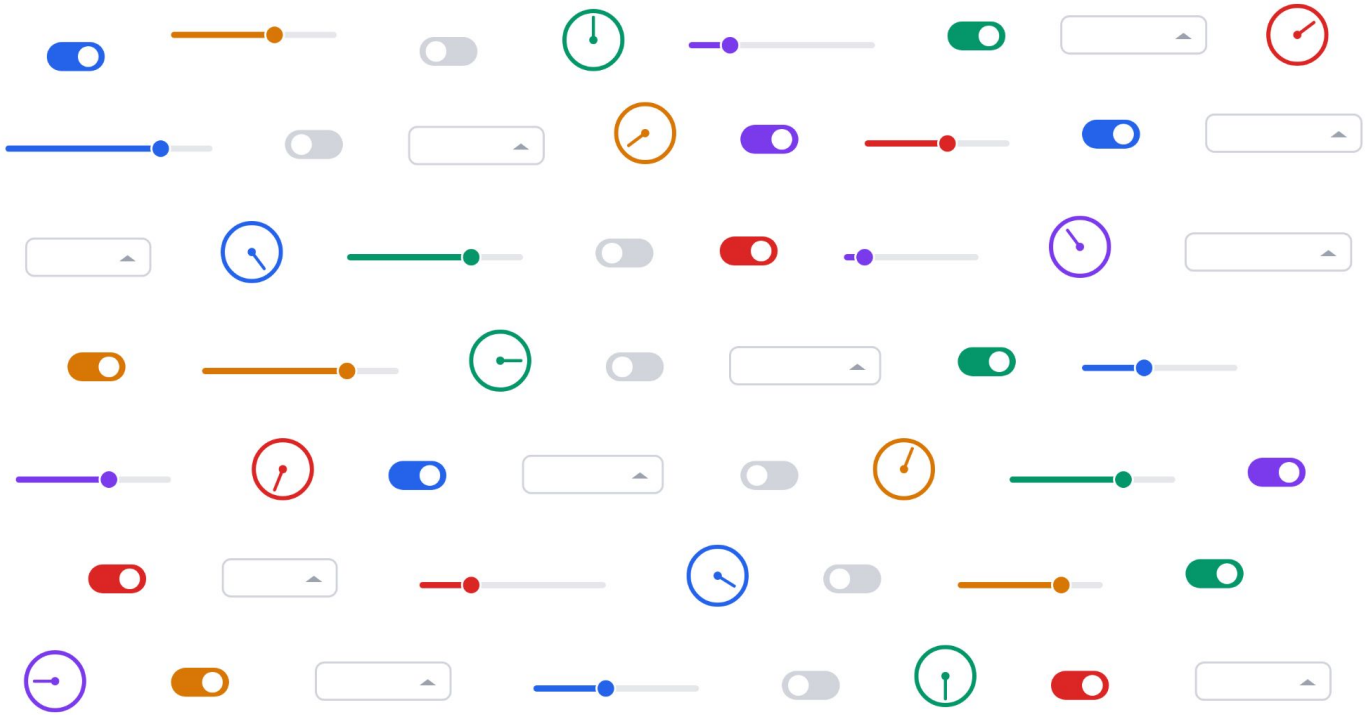
Scaling Up


NODE
GPU
GPU
GPU
GPU
Node
Node
Node
Node
Node
Node
Node
Node
Node
Node

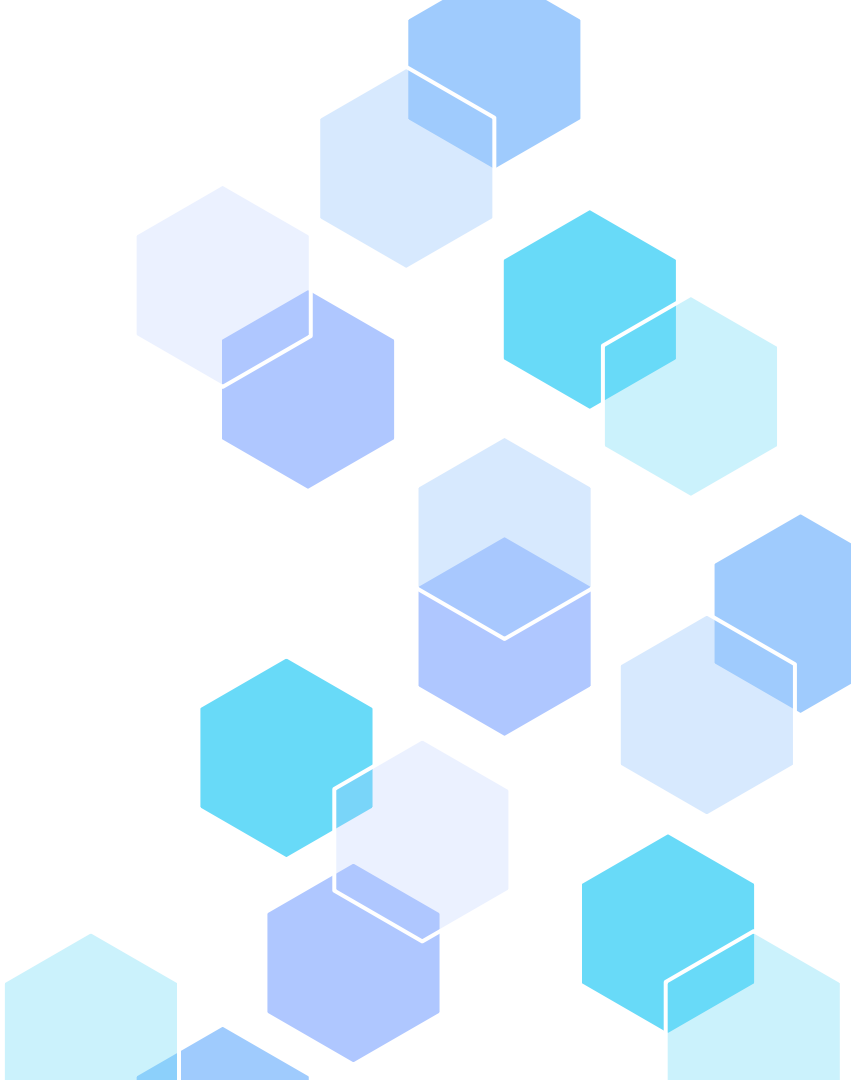
Discussion

What kind of problems might we face when scaling up the number of GPUs?

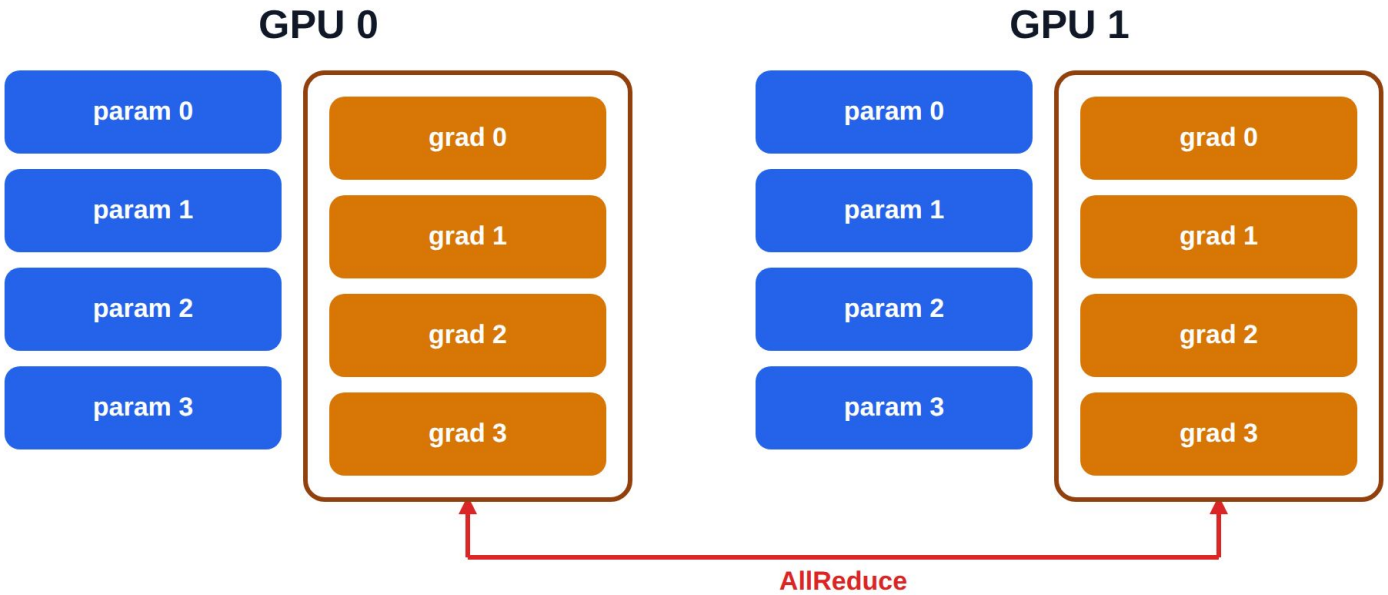
Usability



Existing Solutions



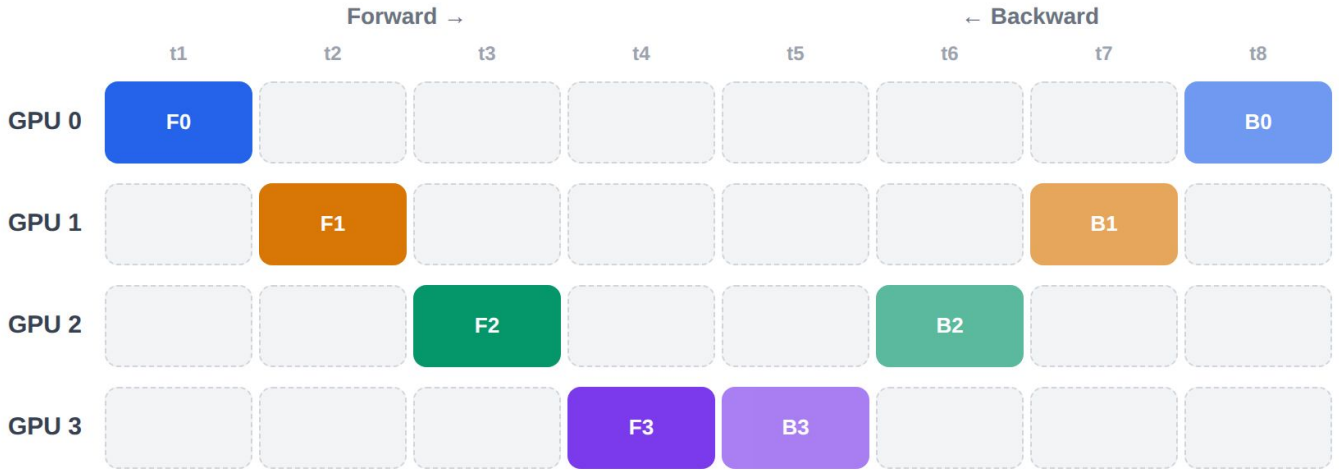
Data Parallelism



DDP - Distributed Data Parallel

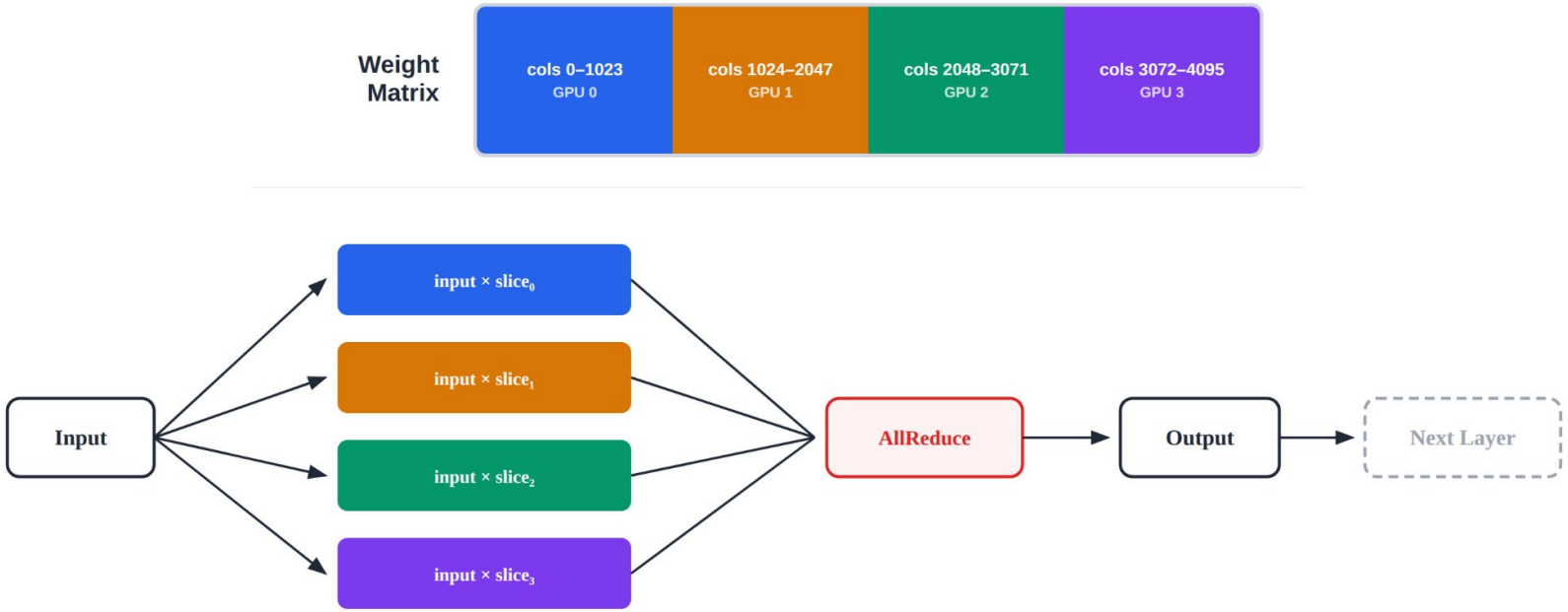
Pipeline Parallelism

Pipeline Parallelism

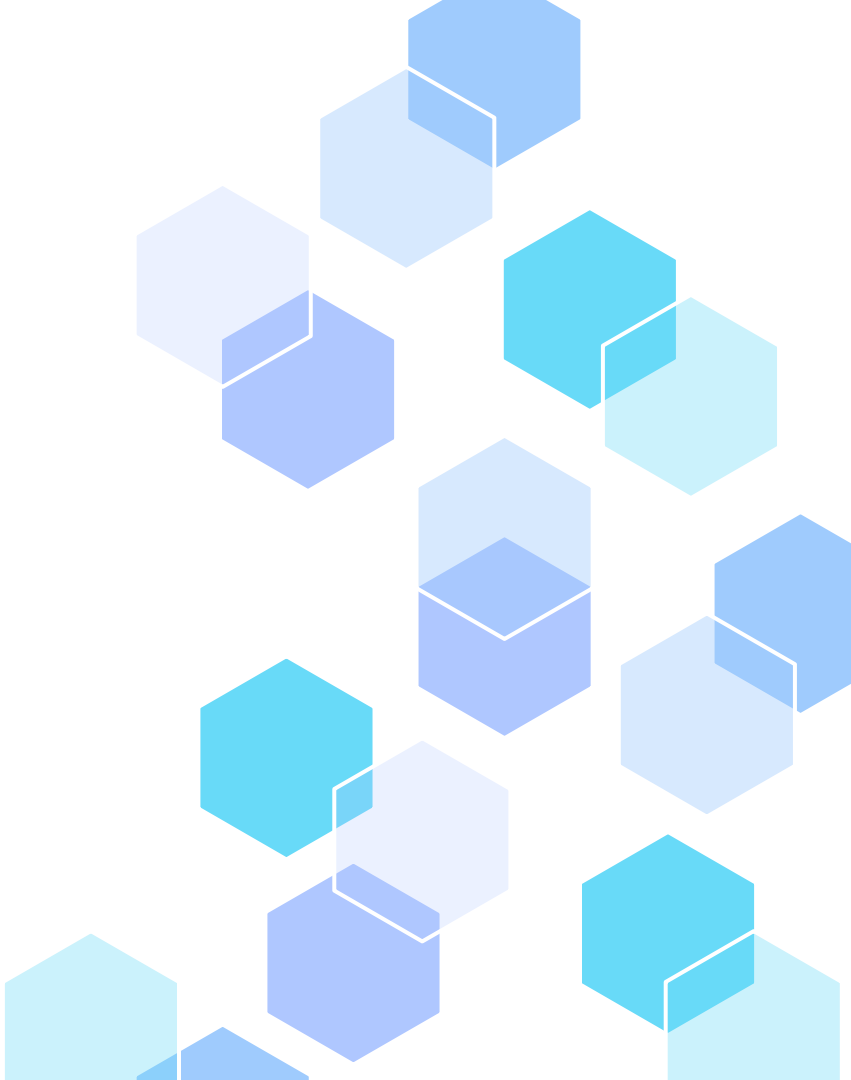


Tensor Parallelism

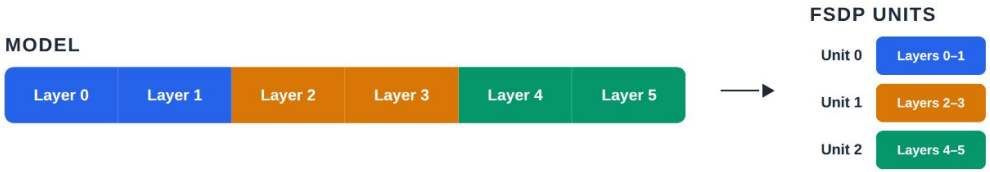
Tensor Parallelism



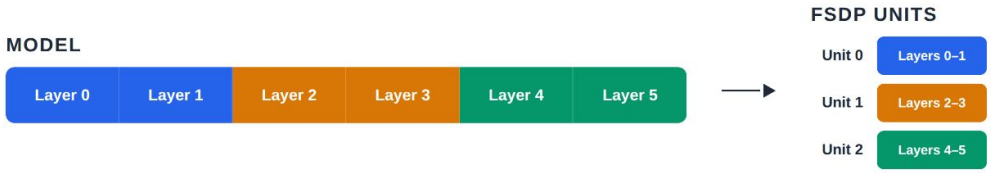
Core Intuition

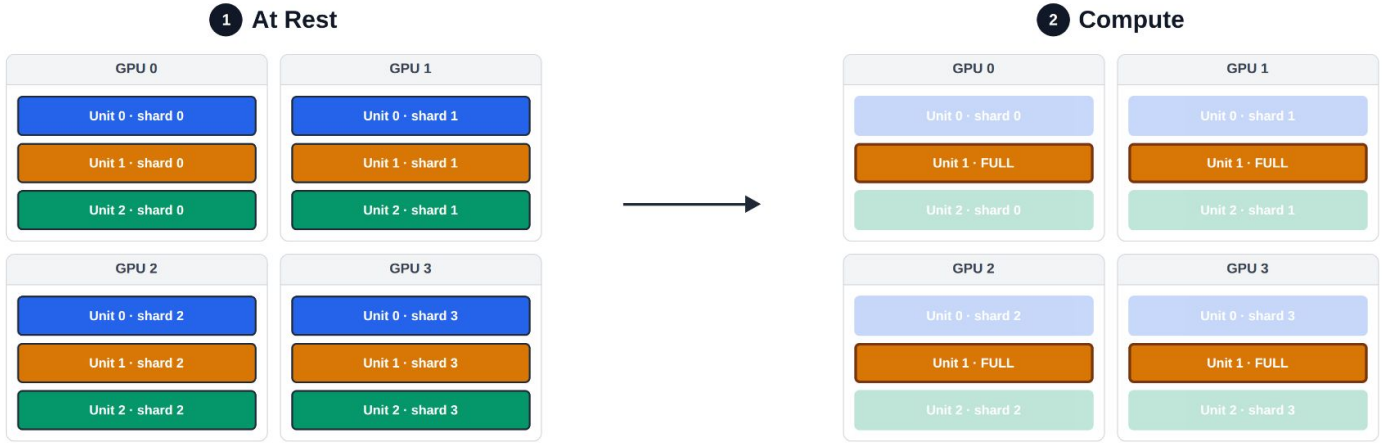


Overview


MODEL
Layer 0
Layer 1
Layer 2
Layer 3
Layer 4
Layer 5
FSDP UNITS
Unit 0
Layers 0–1
Unit 1
Layers 2–3
Unit 2
Layers 4–5

Overview


MODEL
Layer 0
Layer 1
Layer 2
Layer 3
Layer 4
Layer 5
FSDP UNITS
Unit 0
Layers 0–1
Unit 1
Layers 2–3
Unit 2
Layers 4–5


1 At Rest
GPU 0
Unit 0 · shard 0
Unit 1 · shard 0
Unit 2 · shard 0
GPU 1
Unit 0 · shard 1
Unit 1 · shard 1
Unit 2 · shard 1
GPU 2
Unit 0 · shard 2
Unit 1 · shard 2
Unit 2 · shard 2
GPU 3
Unit 0 · shard 3
Unit 1 · shard 3
Unit 2 · shard 3
2 Compute
GPU 0
Unit 0 · shard 0
Unit 1 · FULL
Unit 2 · shard 0
GPU 1
Unit 0 · shard 1
Unit 1 · FULL
Unit 2 · shard 1
GPU 2
Unit 0 · shard 2
Unit 1 · FULL
Unit 2 · shard 2
GPU 3
Unit 0 · shard 3
Unit 1 · FULL
Unit 2 · shard 3

NCCL Operations

Comprehension Check

Why does FSDP rebuild parameters (AllGather) instead of keeping full copies like DDP?

Trading Latency for Memory

Ways to decrease memory pressure

- **↓ unit size**
- **↑ Sharding Factor (full vs hybrid)**
- **Reduce After Forward**
- **Gradient Accumulation with communication**
- **No prefetching**

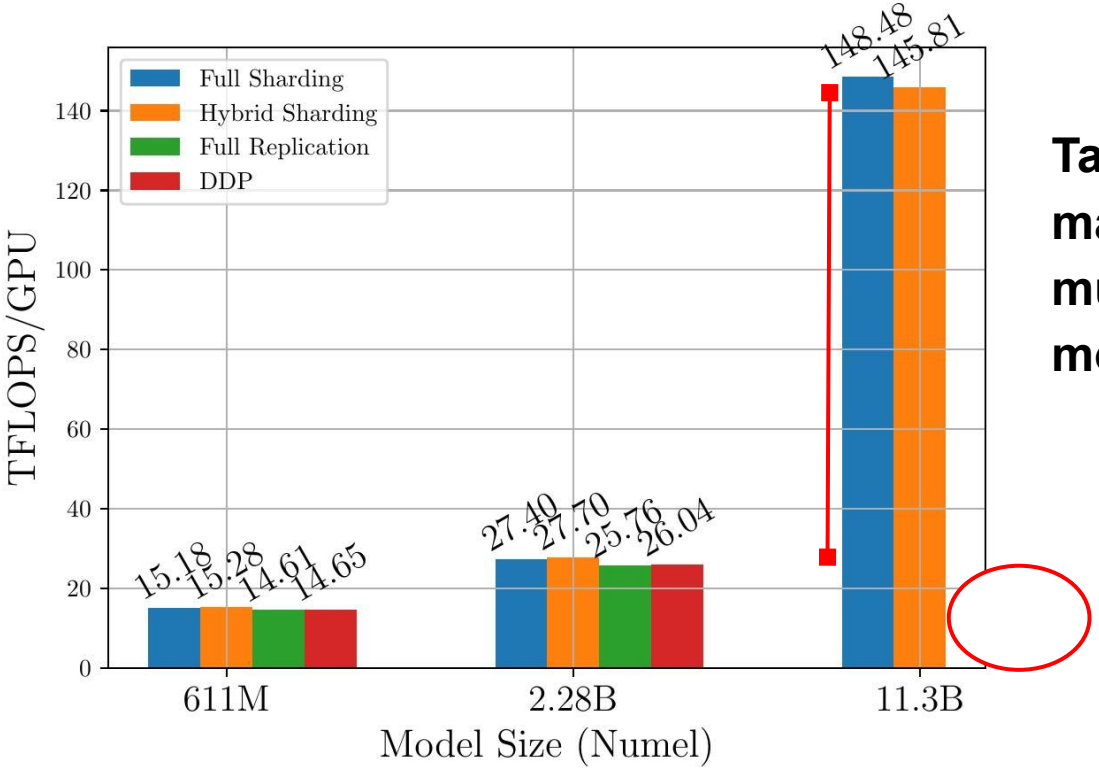
Trading Memory for Latency

Ways to decrease latency

- **↑ unit size**
- **↓ Sharding Factor (full vs hybrid)**
- **No Reduce After Forward**
- **Gradient Accumulation with no communication**
- **Prefetching**

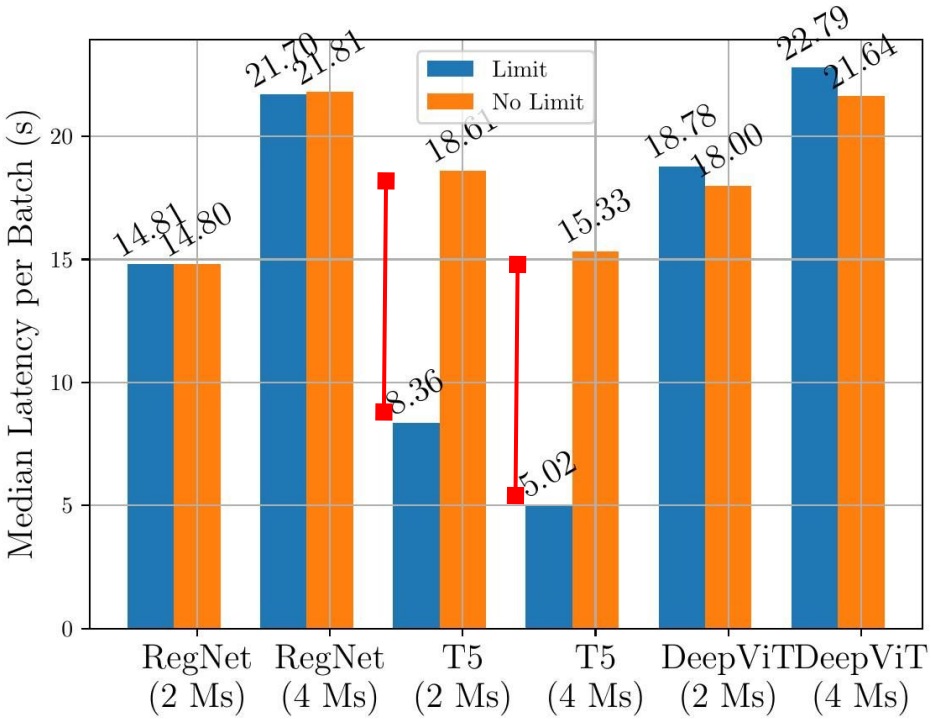
Experimental Results & Analysis

Training Efficiency



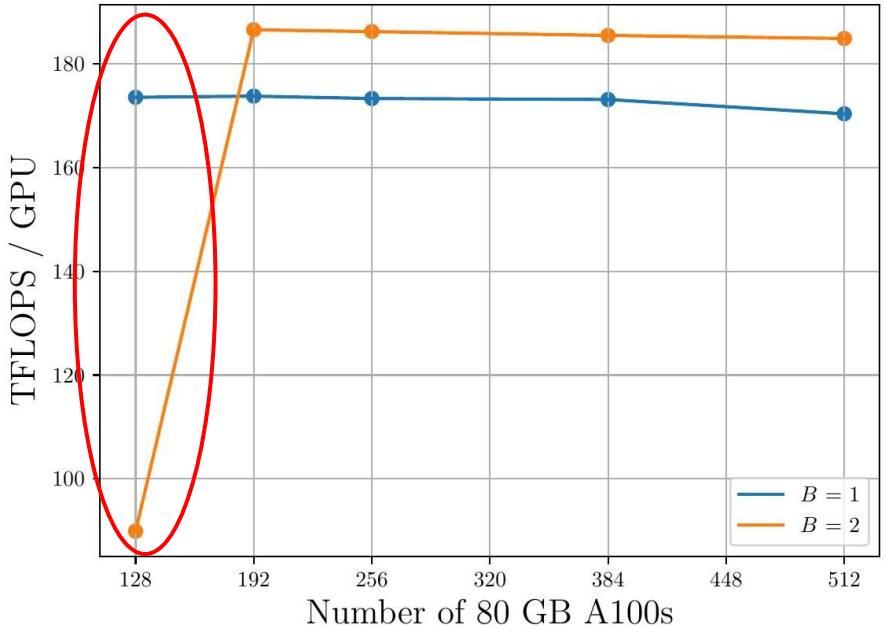
Takeaway: FSDP really matters for multi-billion parameter models

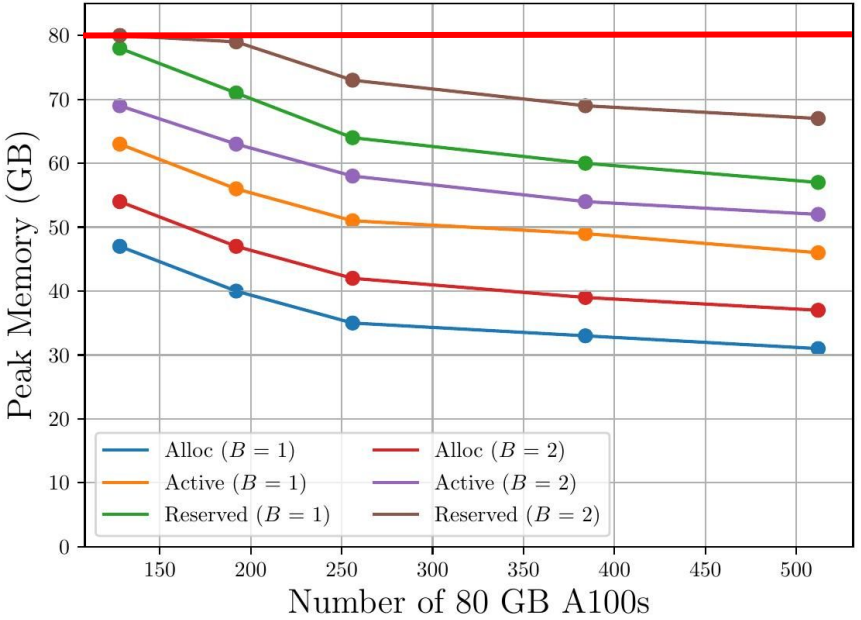
Rate Limiting



Takeaway: Verify if memory defragmentation occurs before rate limiting

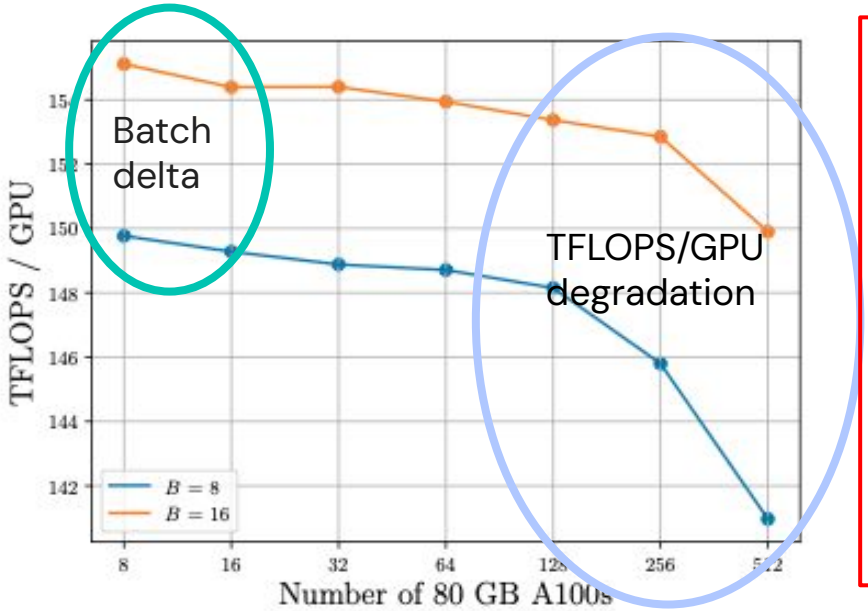
GPT 175B Performance

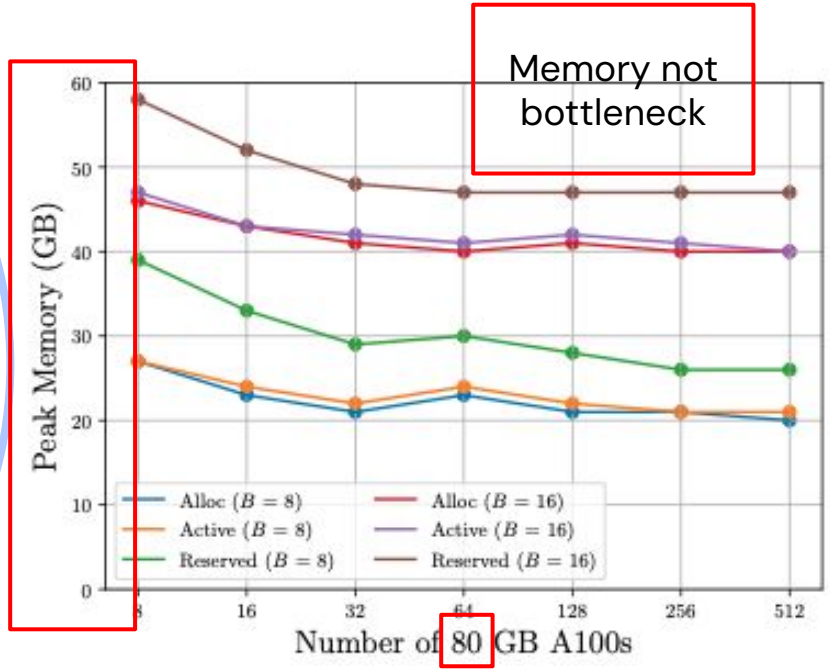




Takeaway: As peak memory approaches 80GB, defragmentation occurs

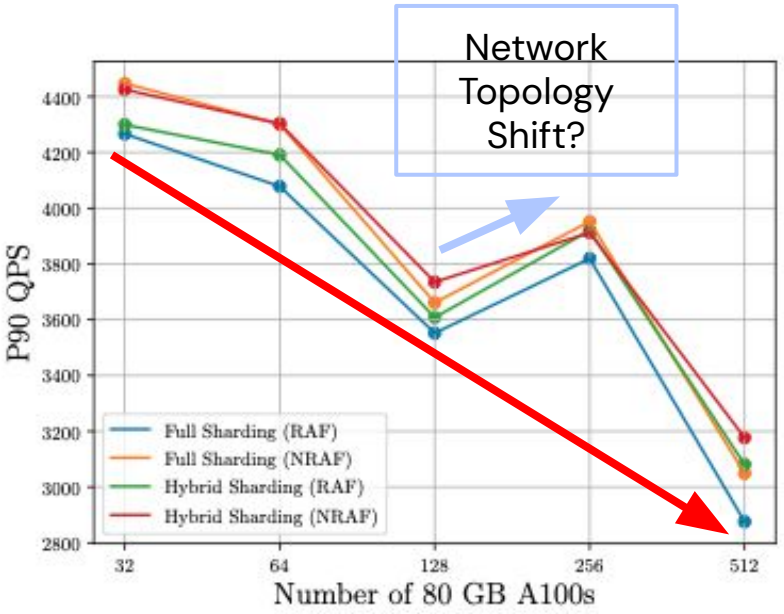
T5-11B Performance

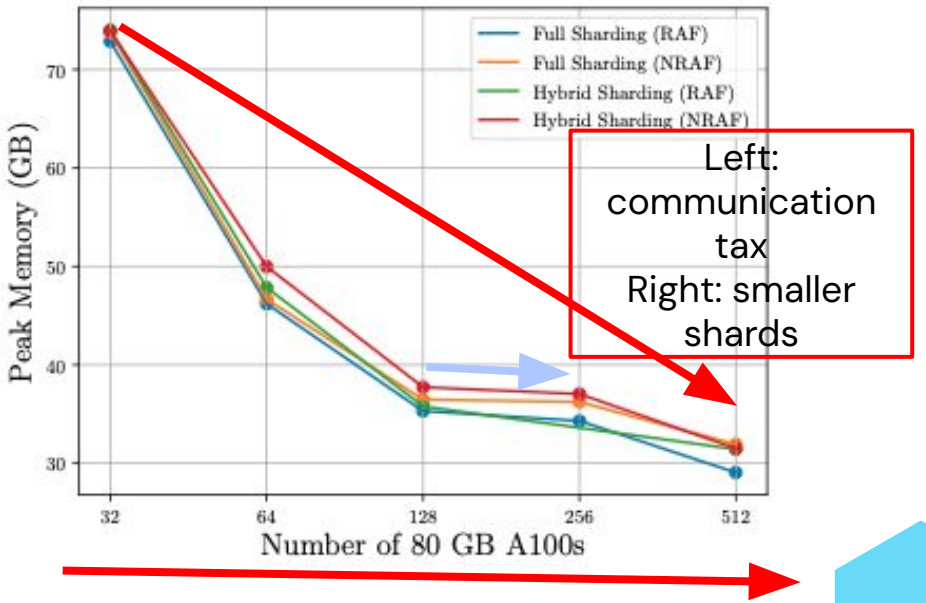




- **Despite ample memory headroom, per-GPU compute efficiency drops ~7% at 512 GPUs because communication overhead outpaces communication overlap at scale**

DHEN (768B Sparse, 550M Dense) Performance





- **FSDP offers a tunable memory-throughput trade-off: RAF minimizes memory, while Hybrid NRAF maximizes throughput**

Are Experiments Sufficient?

Takeaway

- **Convincing for:**
  - **Scaling**
  - **Memory efficiency**
- **Not sufficient for:**
  - **Generality**
  - **System comparisons**

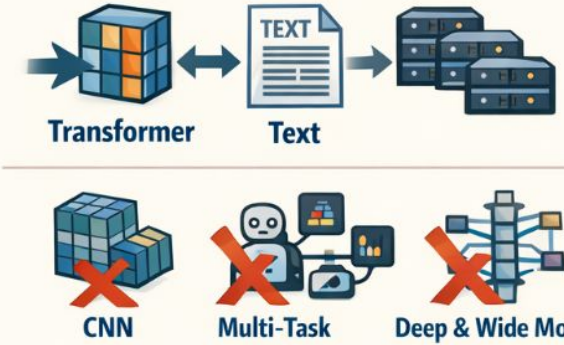
Discussion

Is FSDP a net win for scaling when considering memory vs. communication tradeoffs?

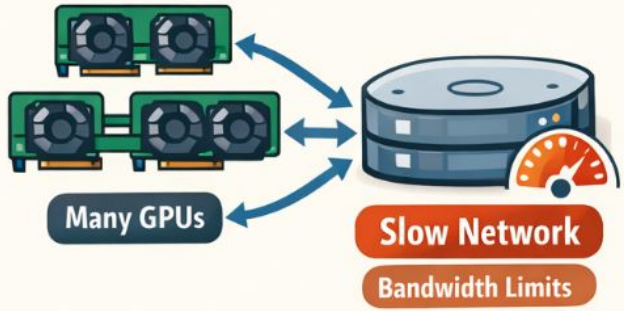
Gaps in the Paper

Potential Limitations

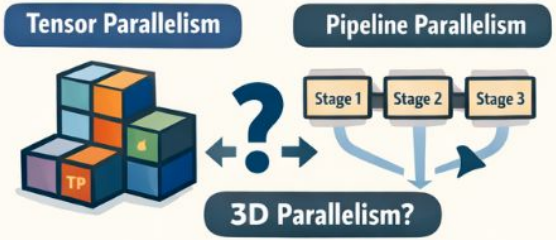
Narrow Evaluation Scope



Network Communication Bottleneck

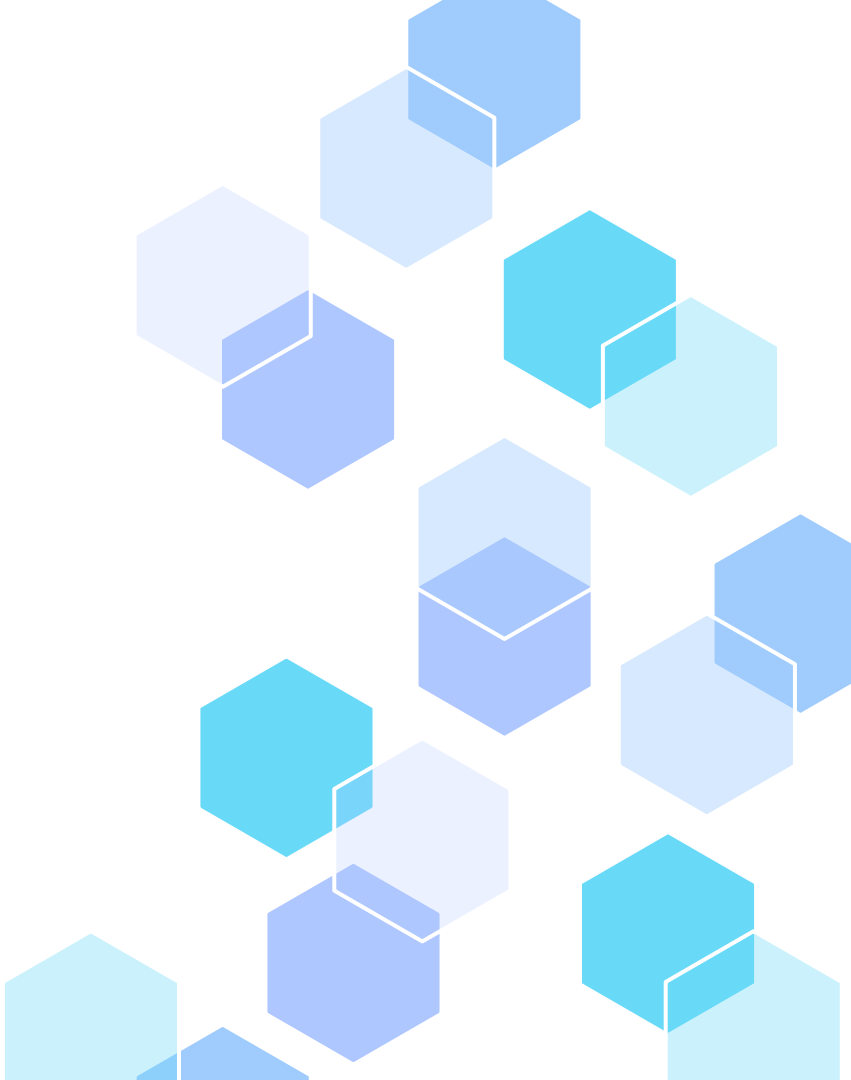


Hybrid Parallelism Gaps



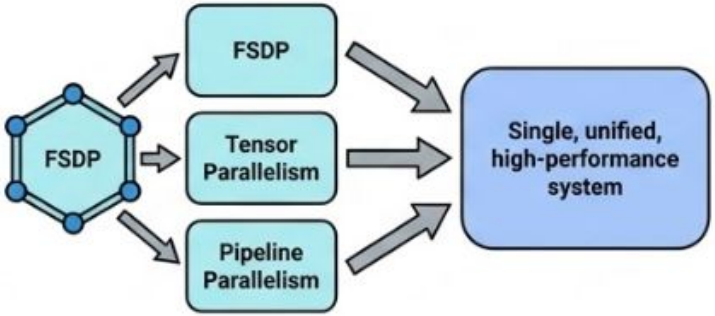
Checkpointing & Fault Tolerance

Next Steps

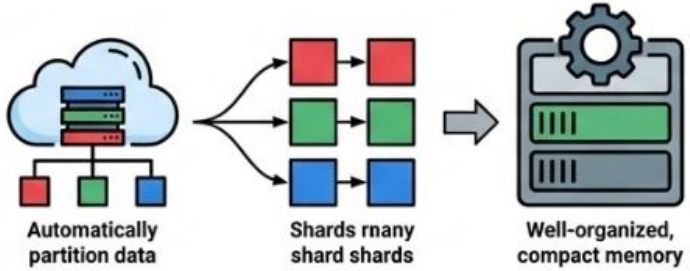


Future Systems

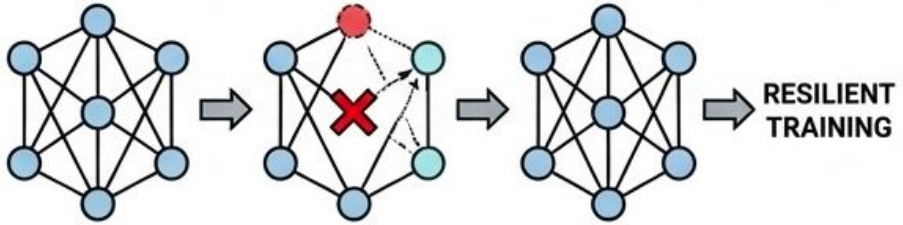
Hybrid Parallelism Systems



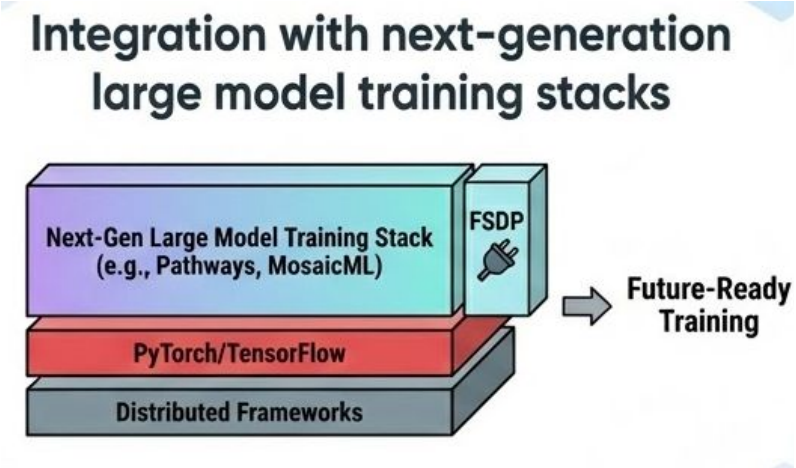
Automatic Sharding & Memory Optimization

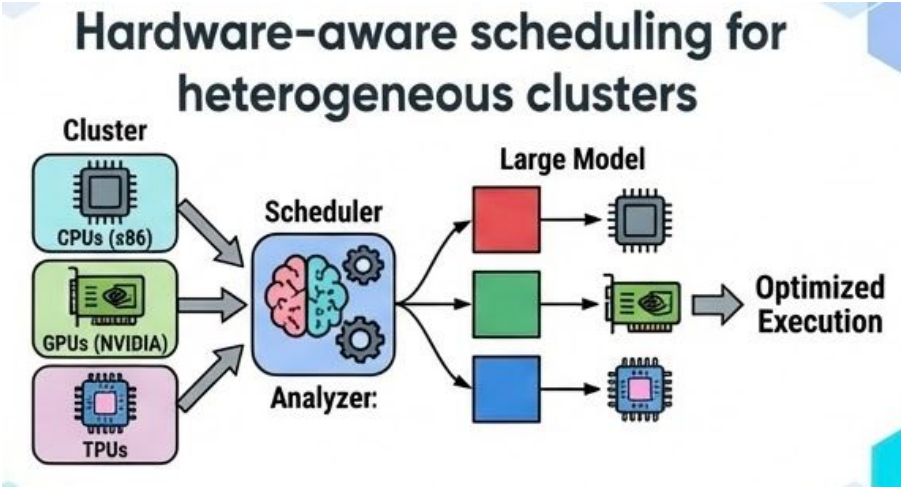


Elastic & Fault-Tolerant Training



Future Systems





Thanks!

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**