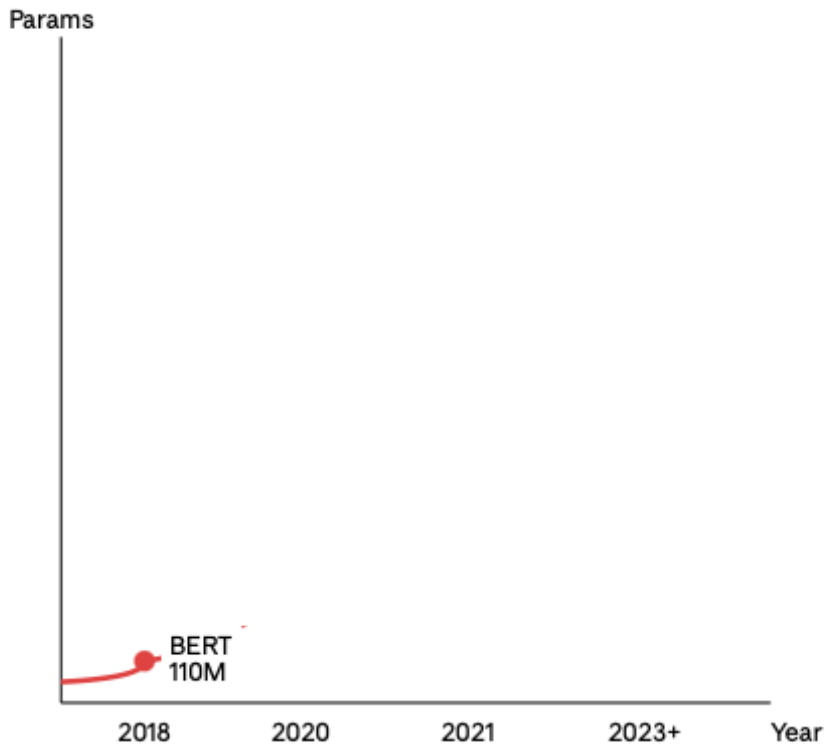


# ZeRO: Memory Optimizations Toward Training Trillion Parameter Models

Samyam Rajbhandari\*, Jeff Rasley\* , Olatunji Ruwase, Yuxiong He

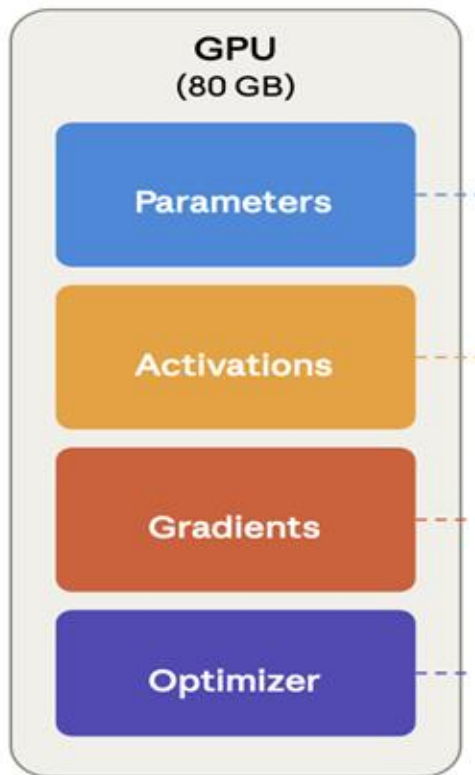
Model size is growing exponentially, but GPU memory is not



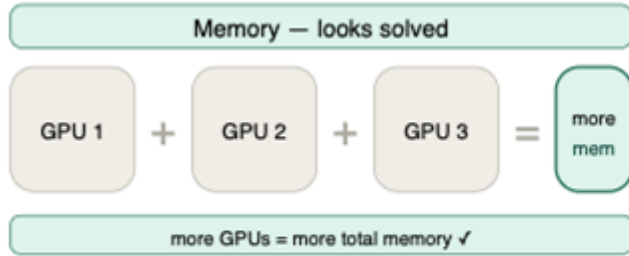
# How is a model trained?

---

# What is stored in a single GPU?



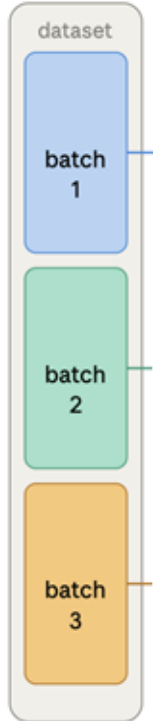
# Naive Solution



How do existing methods attempted to solve this problem?

# 1. Data Parallelism

Data parallelism: same model, different data



## 2. Model Parallelism

Tensor parallelism:  $W$  is split across GPUs — each GPU receives the full input activation



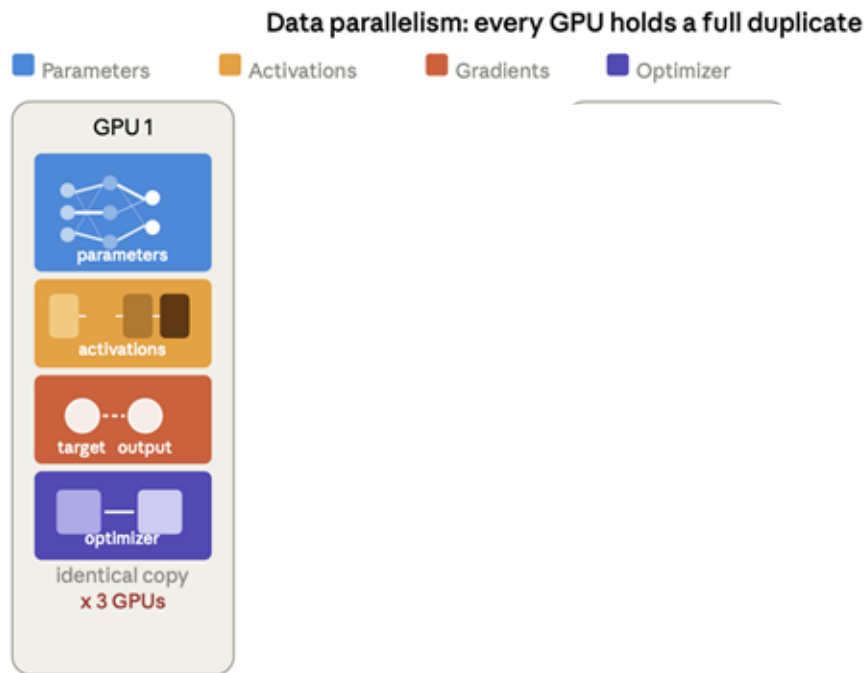
■ activation (yellow)

■ weight shard  $W$  (blue, lighter = further shard)

□ redundant copy stored for backprop

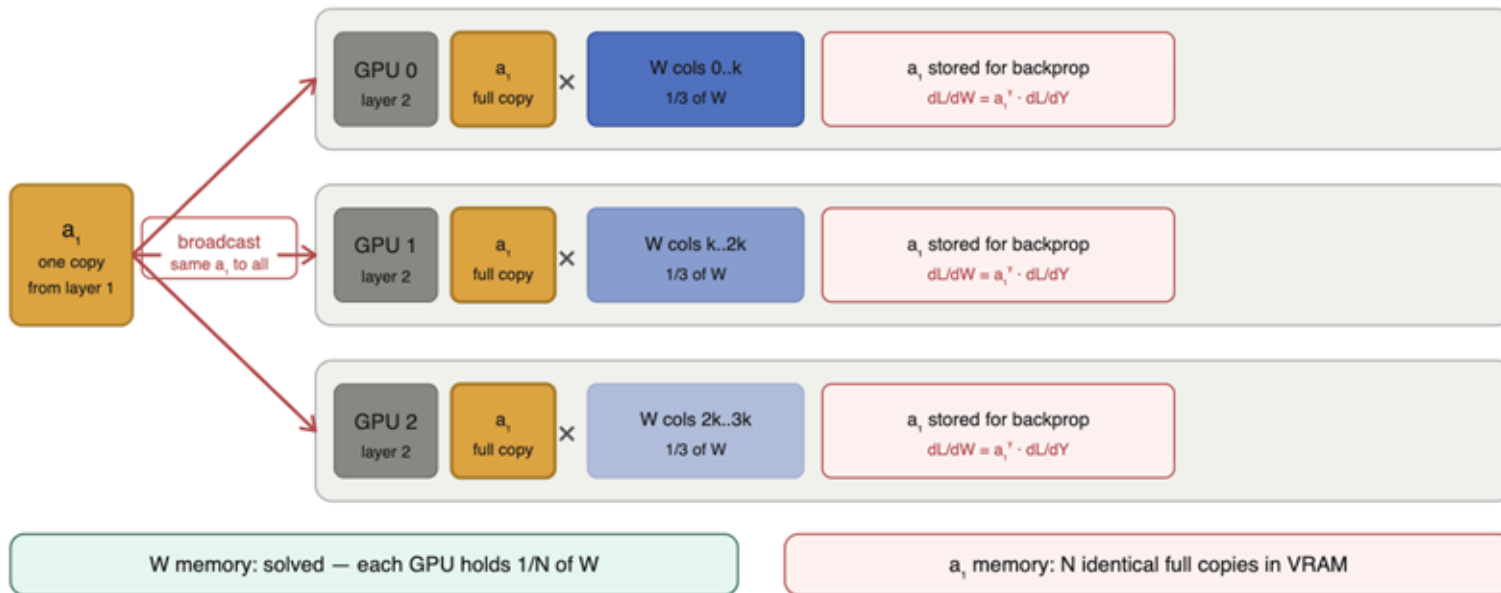
Where do Existing methods Fail?

# 1. Problem with Data Parallelism:



## 2. Problem with Model Parallelism

Tensor parallelism:  $W$  is split across GPUs — but  $a_i$  must be broadcast to every GPU



# Redundancy is the key Issue

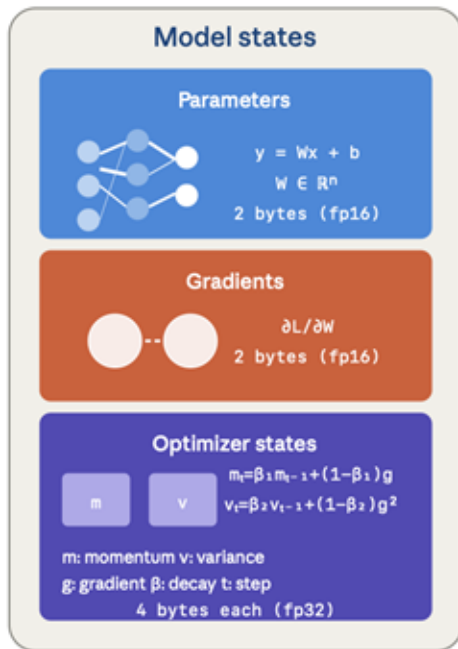
Model states — data parallelism

Activations — tensor parallelism

---

# ZeRO Full Spectrum Analysis

What fills GPU memory during training



majority of memory

The Main Intuition is: Partition What persists, Checkpoint  
What Disappears

# Model States Persist

Model states persist across all three phases and are solved by partitioning across N GPUs



■ parameters ( $2\Psi$ )

■ gradients ( $2\Psi$ )

■ optimizer states ( $8\Psi$ )

□ idle or not computed

$\Psi$  = number of parameters

# Residual States Disappears

Residual states are transient within a phase and cannot be solved by partitioning

## Activations

Forward pass

Backward pass

Weight update

layer 1 activation

freed

layer 2 activation

consuming

...

...

all freed

grows layer by layer

shrinks layer by layer

nothing remains

Partitioning does not help: data is gone before another GPU could benefit

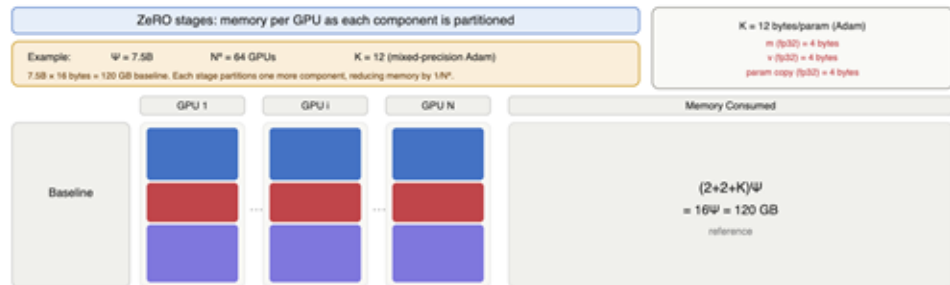
Solution: activation checkpointing

store only at selected checkpoint layers

recompute the rest during backward pass

Both residual state problems are within a single GPU and within a single phase, partitioning across GPUs cannot help

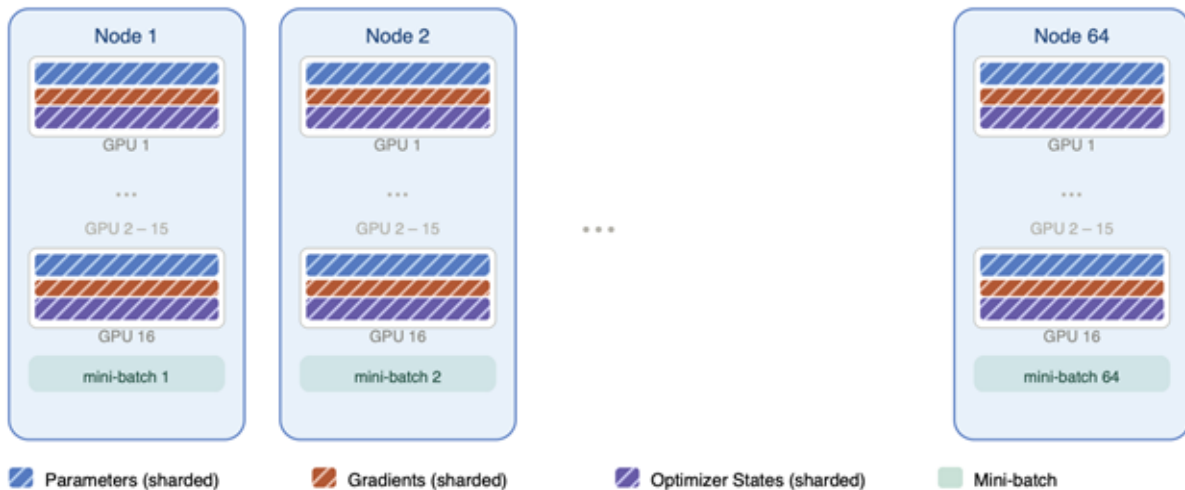
# ZeRO-DP: Optimizing Model State Memory



# Result: Enables 1T-parameter Model Training

ZeRO-DP: 64 nodes × 16 GPUs = 1024 GPUs

Each node holds a ZeRO-sharded replica · each processes a different mini-batch



64 nodes × 16 GPUs = 1024 GPUs total

ZeRO shards P, G, Os — each node holds 1/64 of model states · 1T model fits

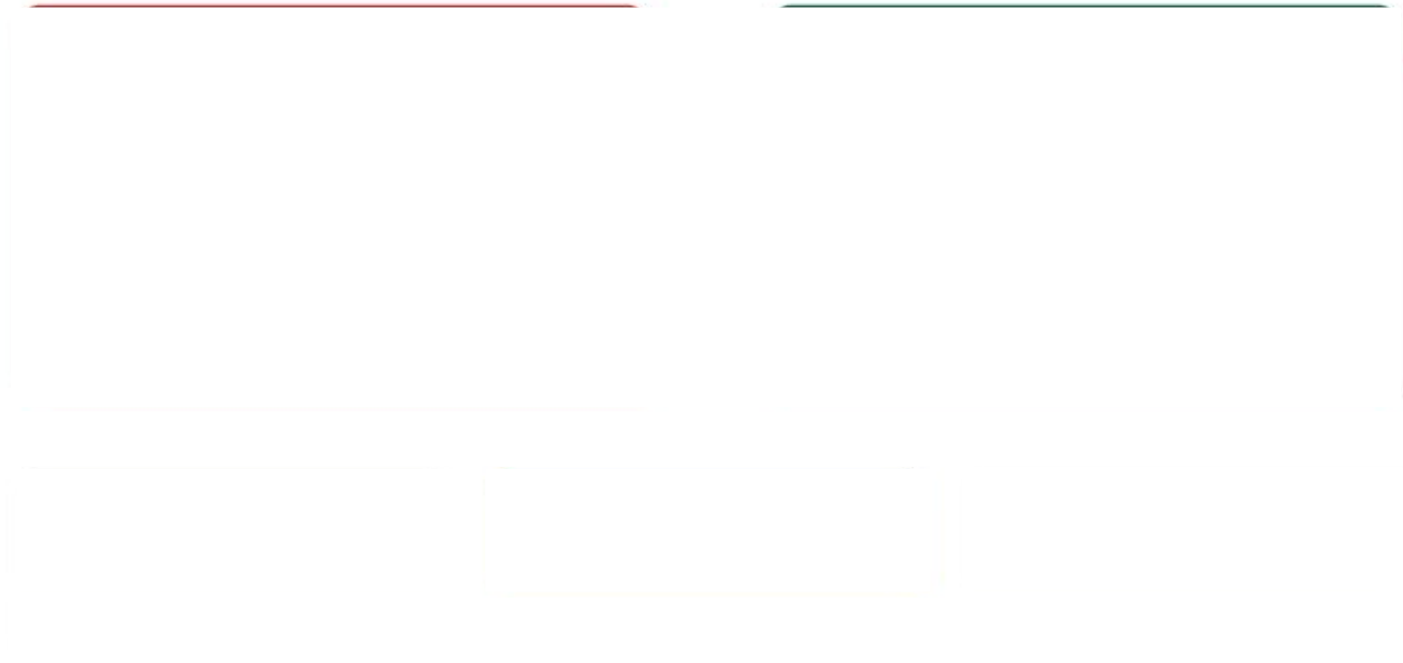
# ZeRO-R: Optimizing Activations

Activation Partitioning

# ZeRO-R: Optimizing Residual State Memory

ZeRO-R: Fixed-size buffers for all-reduce communication


The temporary buffer used to stage gradient data during all-reduce grows with model size without ZeRO-R



# ZeRO-R: Optimizing Residual State Memory

Memory Defragmentation

 Long-lived (params)

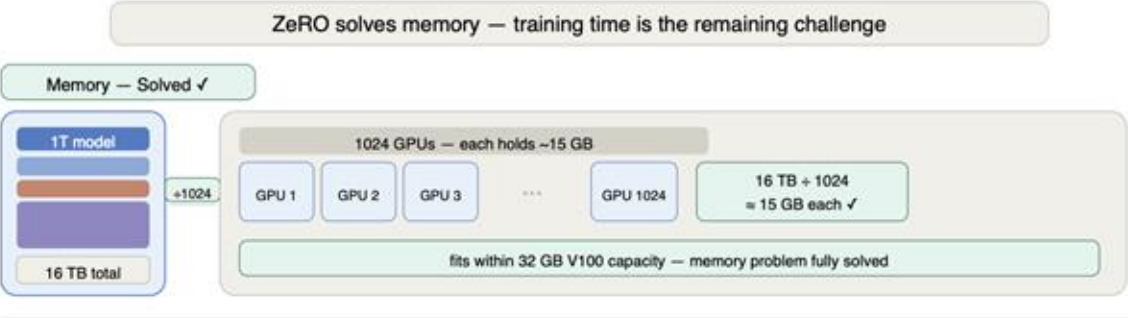
 Short-lived (activations)

 Free gaps

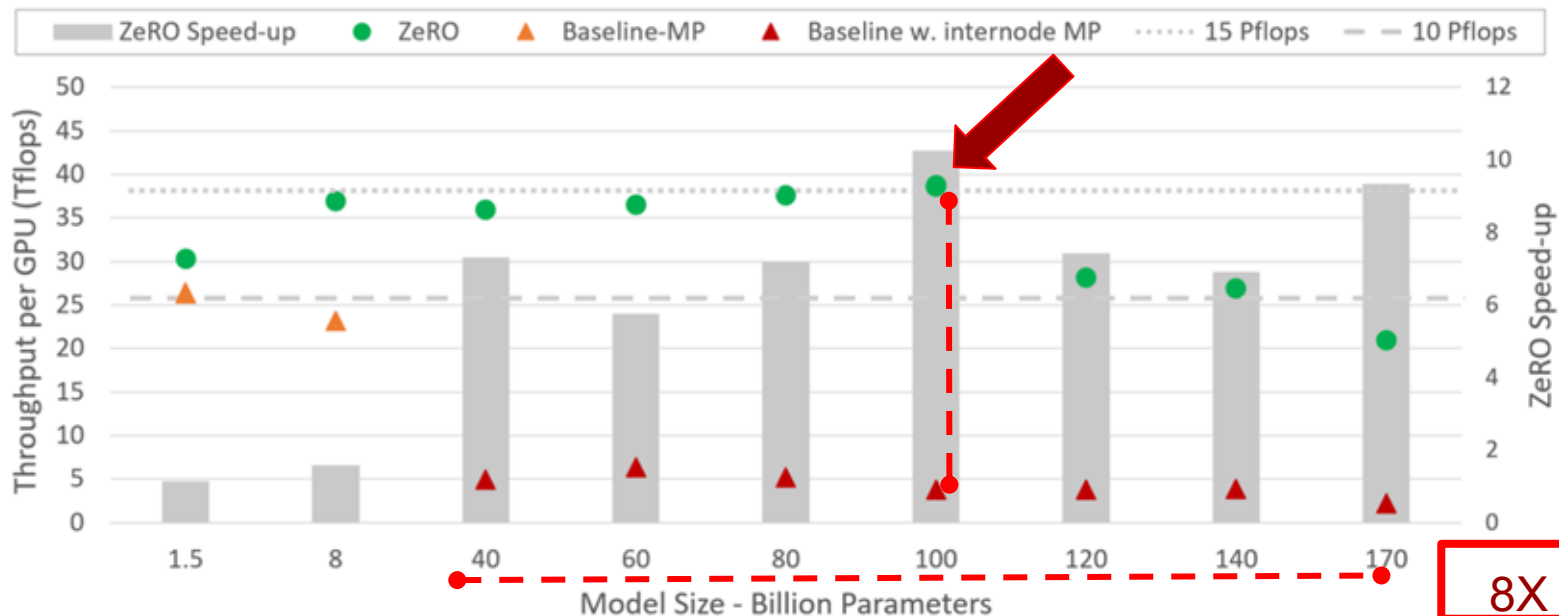
**Zero-DP + Zero-R = ZeRo**

# Evaluation

# Reshift Target: 1 Trillion Parameter Model

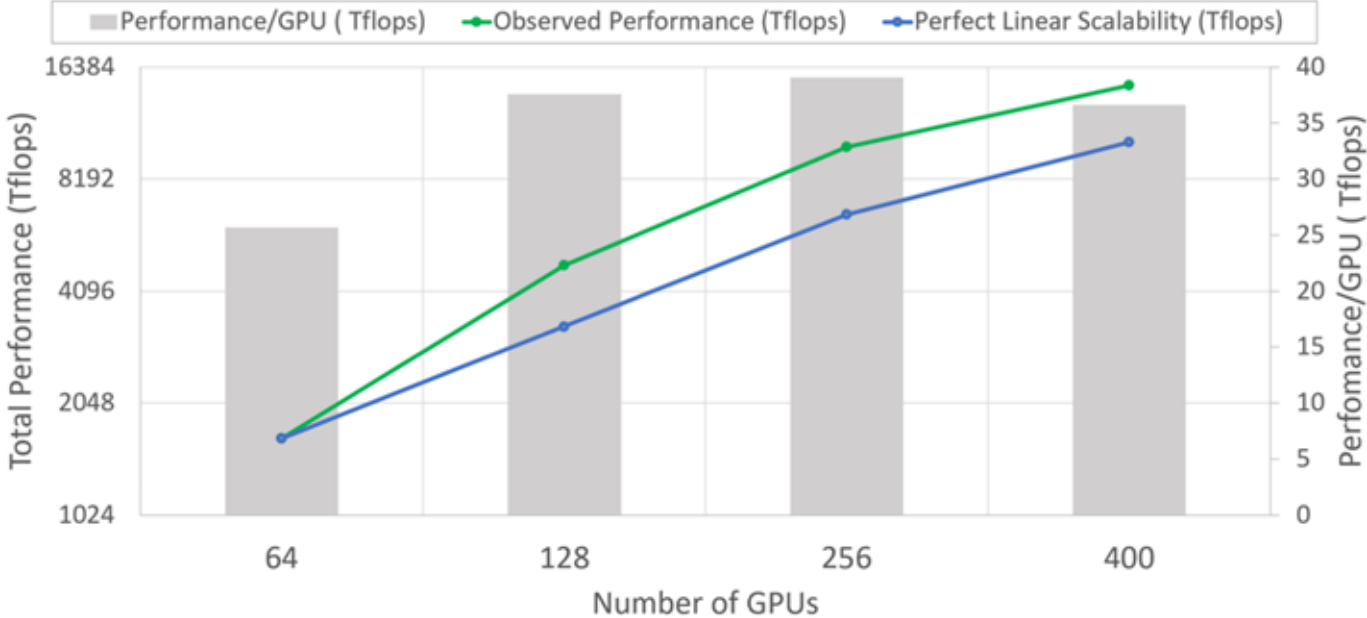


# Memory Efficiency and Speed with model size



8X

# Scalability





# Drawbacks

- **Higher communication overhead:** Bottleneck on slow interconnects (e.g.,

Ethernet vs NVLink)

states; harder to maintain

**Thank you!**