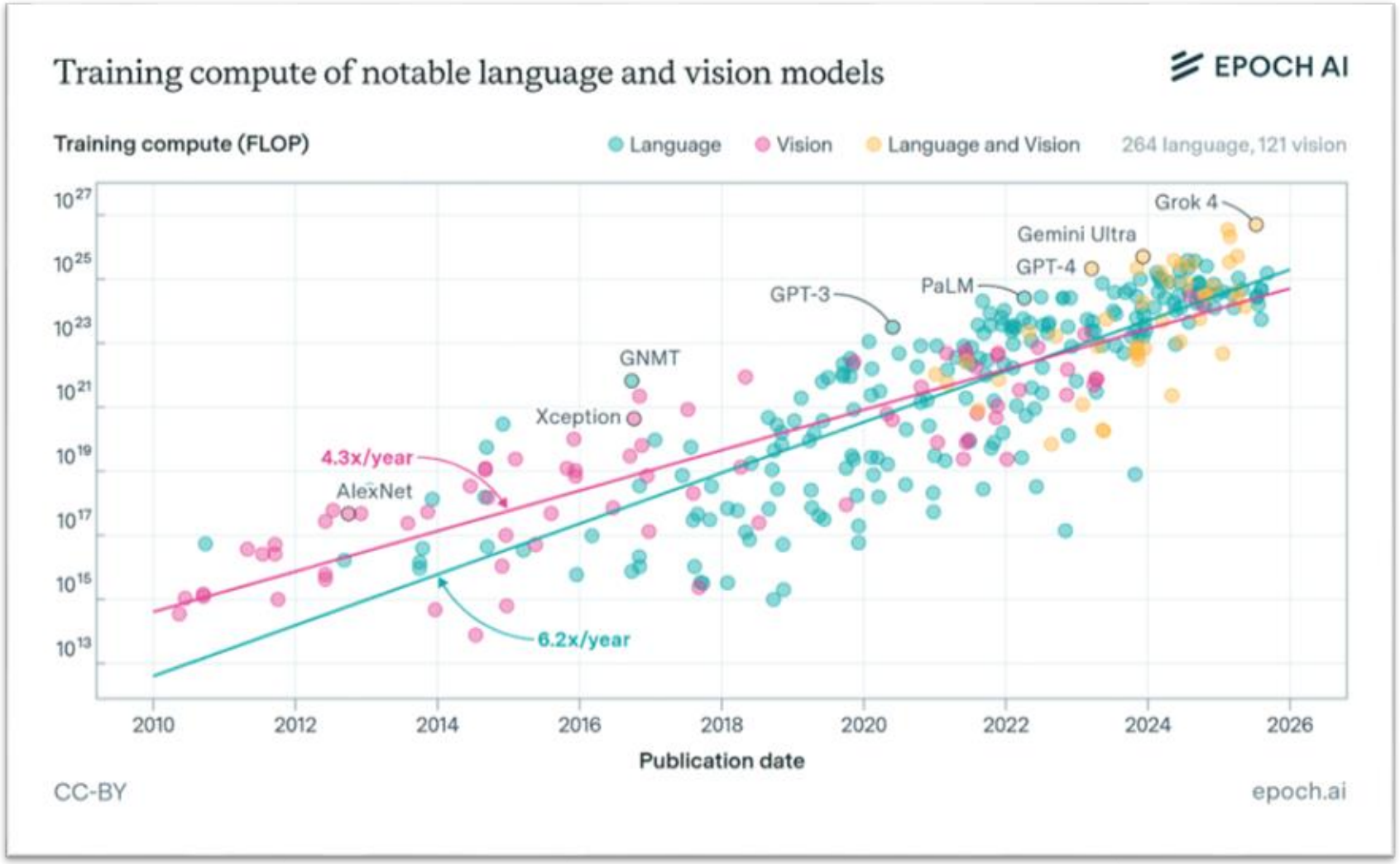
Checkmate: Zero Performance Overhead Model Checkpointing via Network Gradient Replication

Ankit Bhardwaj*, Weiyang “Frank” Wang*, Jeremy Carin, Adam Belay, and Manya Ghobadi

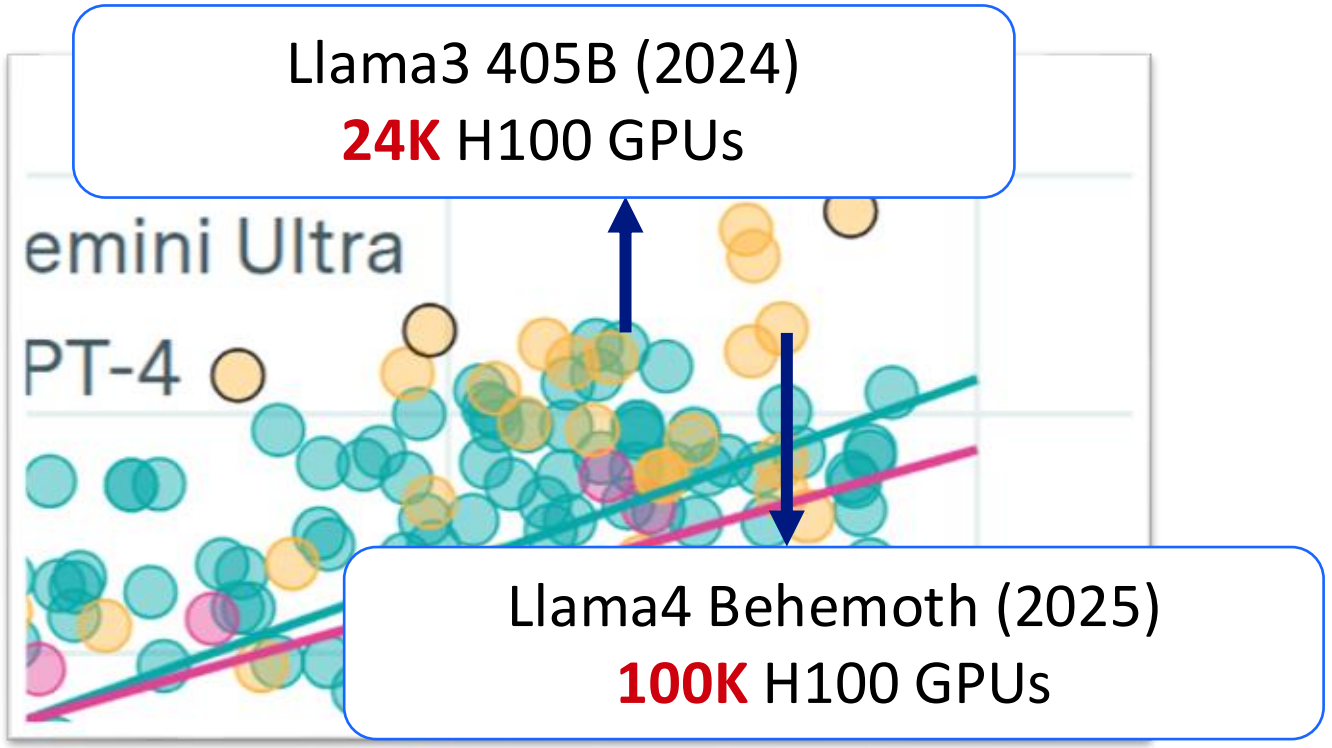
NSDI 2026

Modern machine learning (ML) is demanding



Robi Rahman and David Owen, "Training compute has scaled up faster for language than vision". Published online at epoch.ai, CC-BY license

Scaling out ML: More GPUs, more network



Robi Rahman and David Owen, "Training compute has scaled up faster for language than vision". Published online at epoch.ai, CC-BY license

Problem: Failures are common at scale

Component	Category	Interruption Count	% of Interruptions
Faulty GPU	GPU	148	30.1%
GPU HBM3 Memory	GPU	72	17.2%
Software Bug	Dependency	54	12.9%
Network Switch/Cable	Network	35	8.4%
Host Maintenance	Unplanned Maintenance	32	7.6%
GPU SRAM Memory	GPU	19	4.5%
GPU System Processor	GPU	17	4.1%
NIC	Host	7	1.7%
NCCL Watchdog Timeouts	Unknown	7	1.7%
Silent Data Corruption	GPU	6	1.4%
GPU Thermal Interface + Sensor	GPU	6	1.4%
SSD	Host	3	0.7%
Power Supply	Host	3	0.7%
Server Chassis	Host	2	0.5%
IO Expansion Board	Host	2	0.5%
Dependency	Dependency	2	0.5%
CPU	Host	2	0.5%
System Memory	Host	2	0.5%

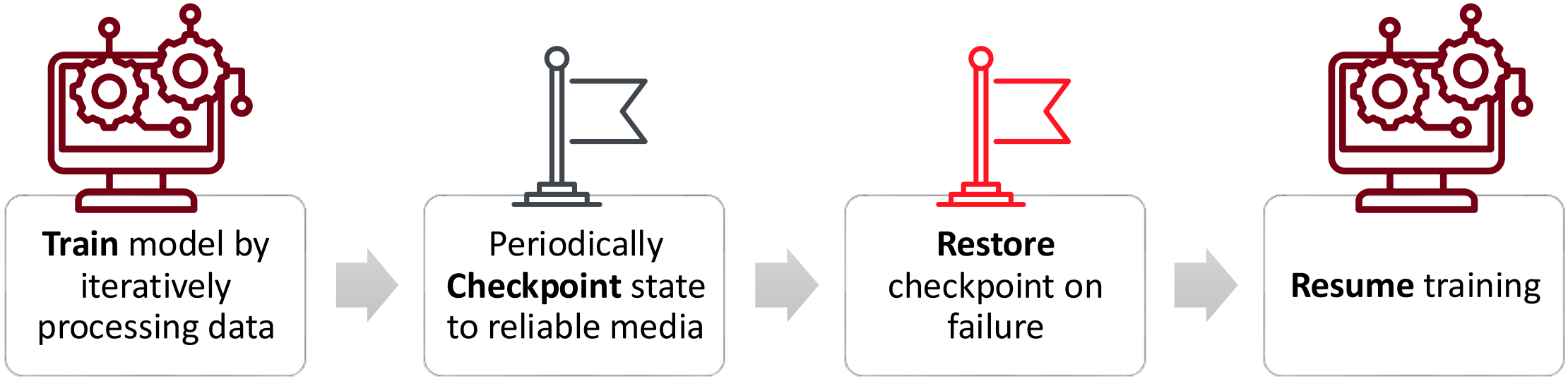
Table 5 Root-cause categorization of unexpected interruptions during a 54-day period of Llama 3 405B pre-training. About 78% of unexpected interruptions were attributed to confirmed or suspected hardware issues.

Failure types during Llama 3 training

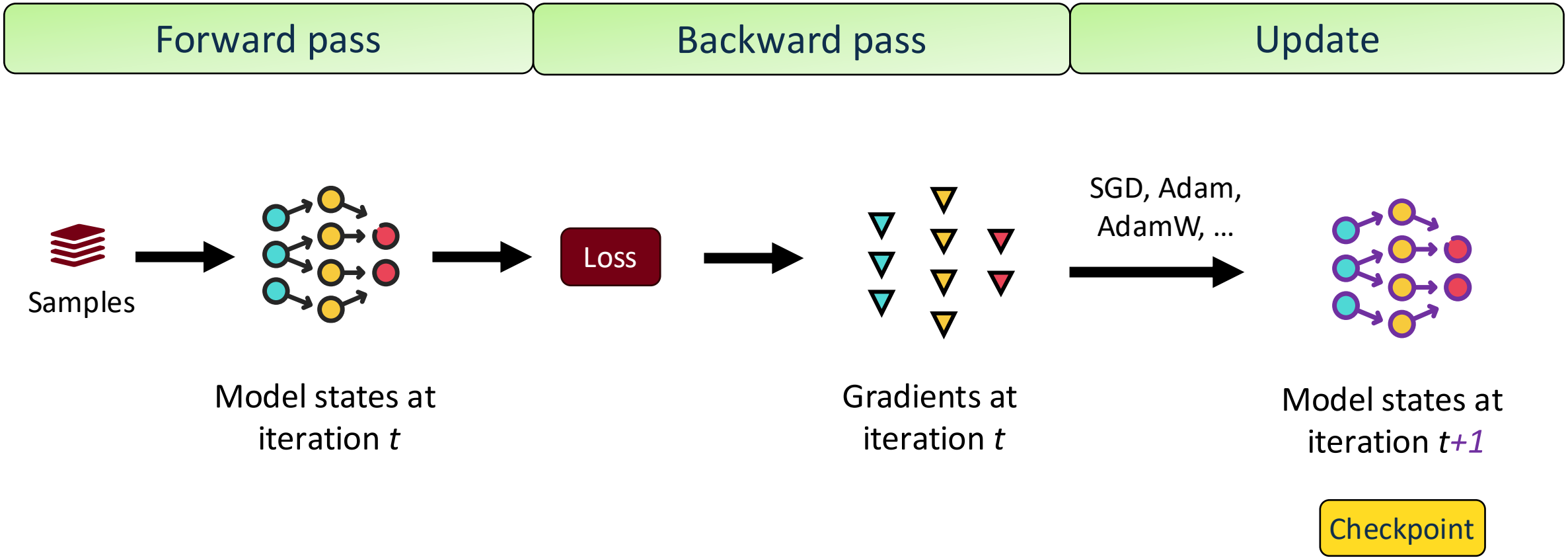
419 unexpected interruptions over 54 days

~2 million wasted GPU-hours

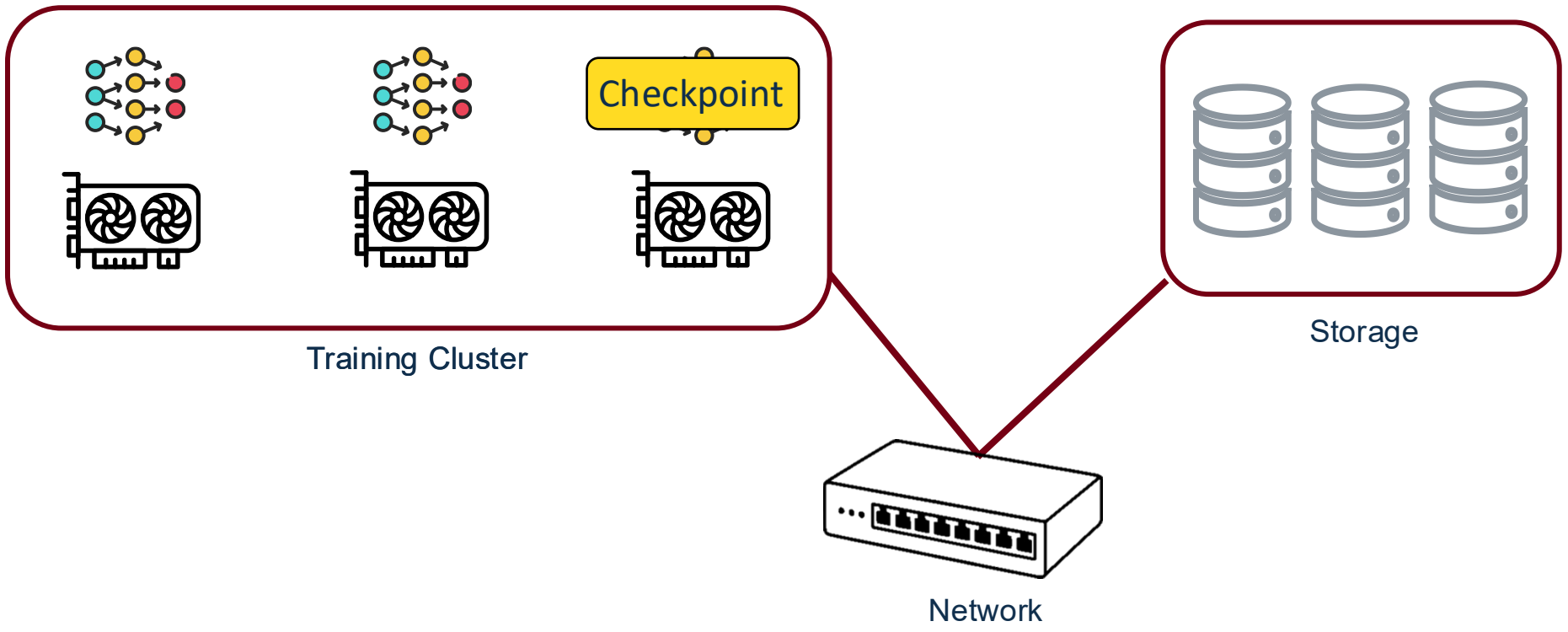
Checkpoint for failure recovery



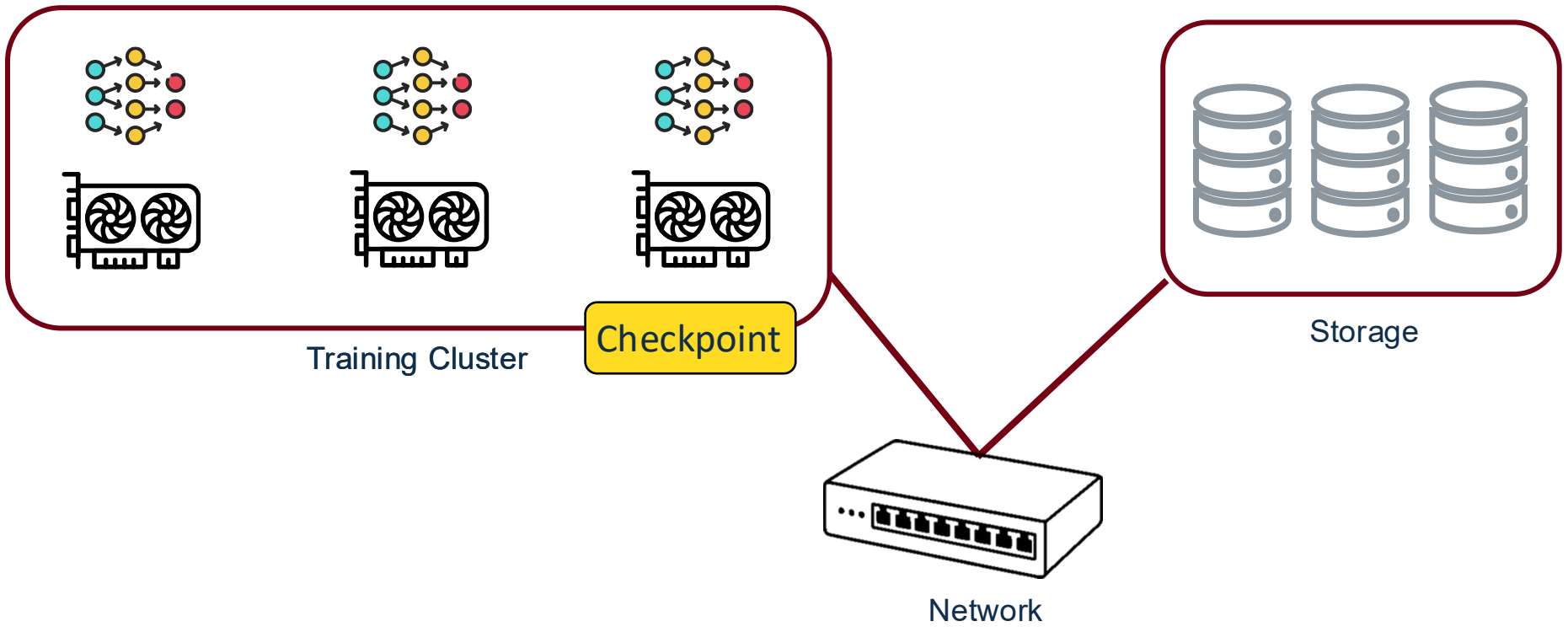
Background: DNN training iteration and model update



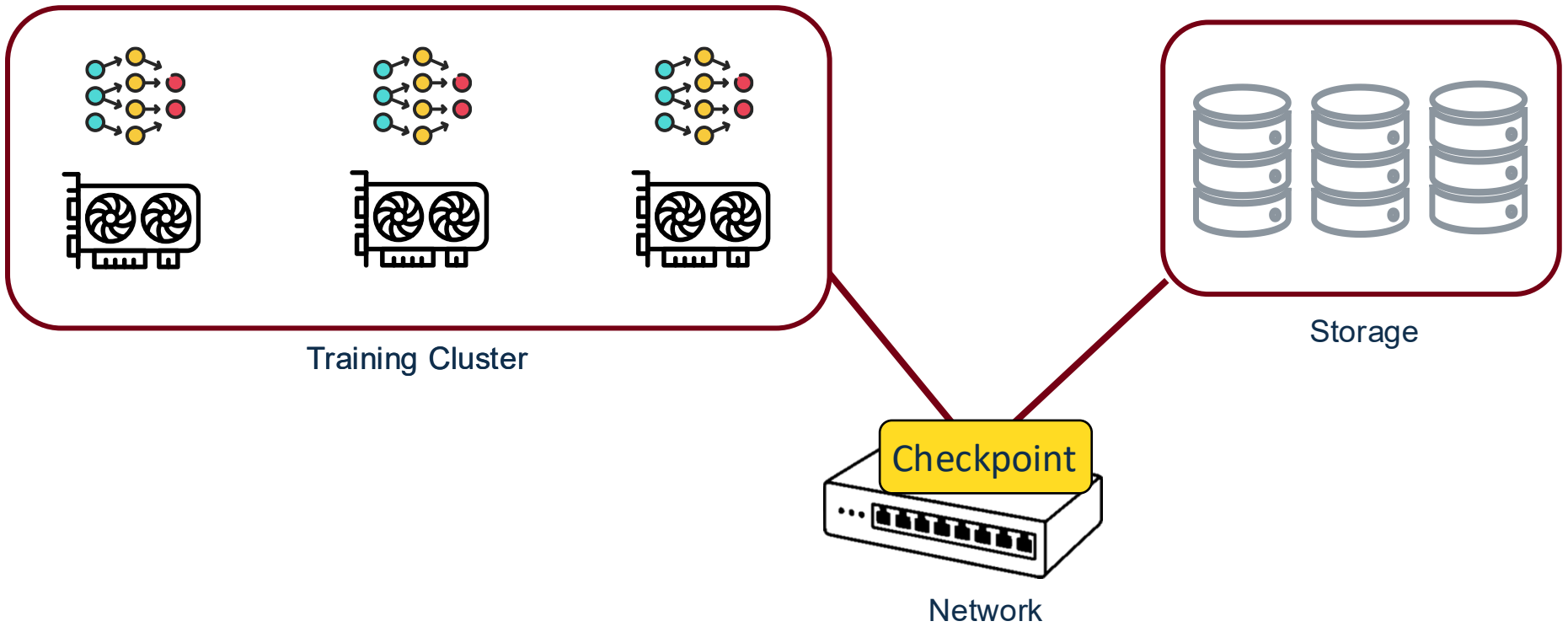
Step 1: `copy` out of GPU memory



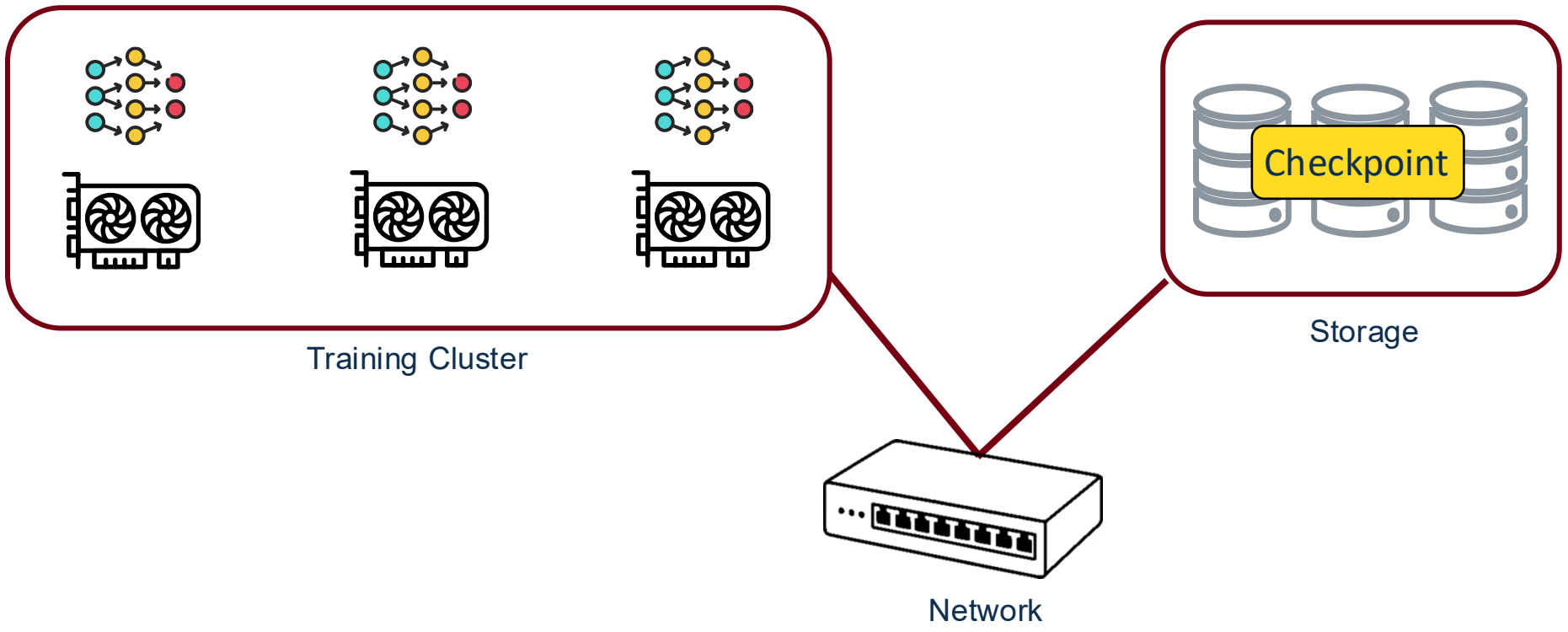
Step 1: `copy` out of GPU memory



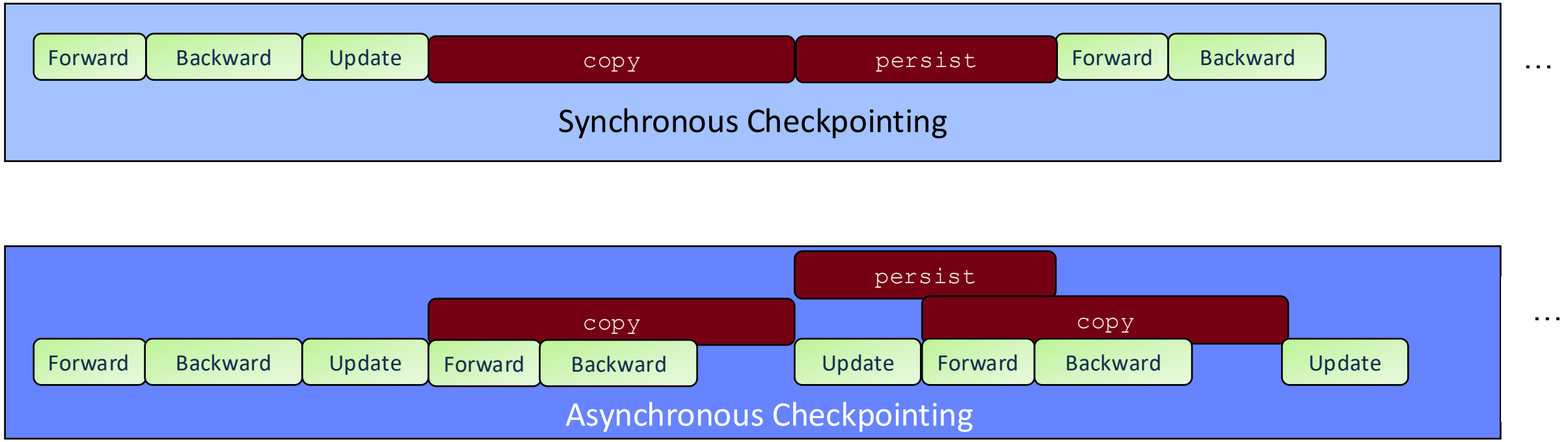
Step 1: `copy` out of GPU memory



Step 2: `persist` to remote storage

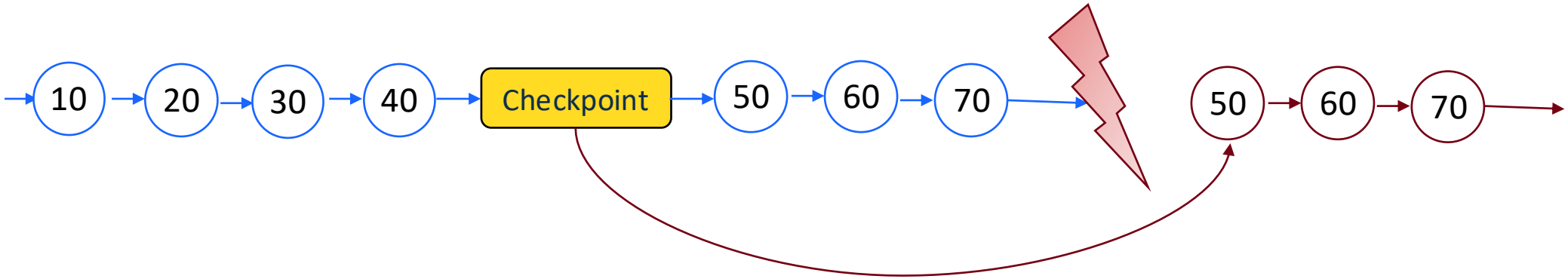


`copy-persist` stalls GPUs



Low checkpointing frequency

Low checkpointing frequency leads to repeated work

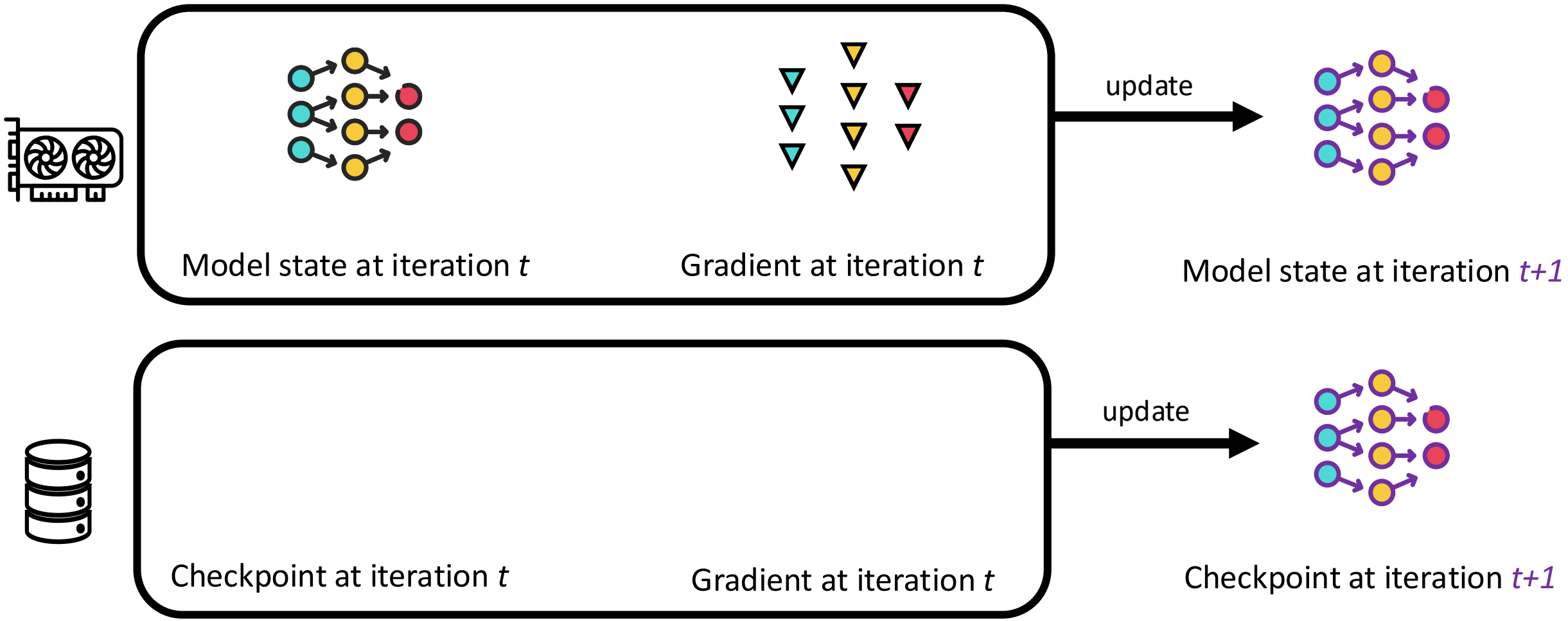


Trade-off:
checkpointing often - high GPU overhead
checkpointing seldomly - more repeated work

“We aim to minimize GPU pause time during checkpointing and increase checkpoint frequency to reduce the amount of lost work after a recovery.”
– Llama-3 Paper

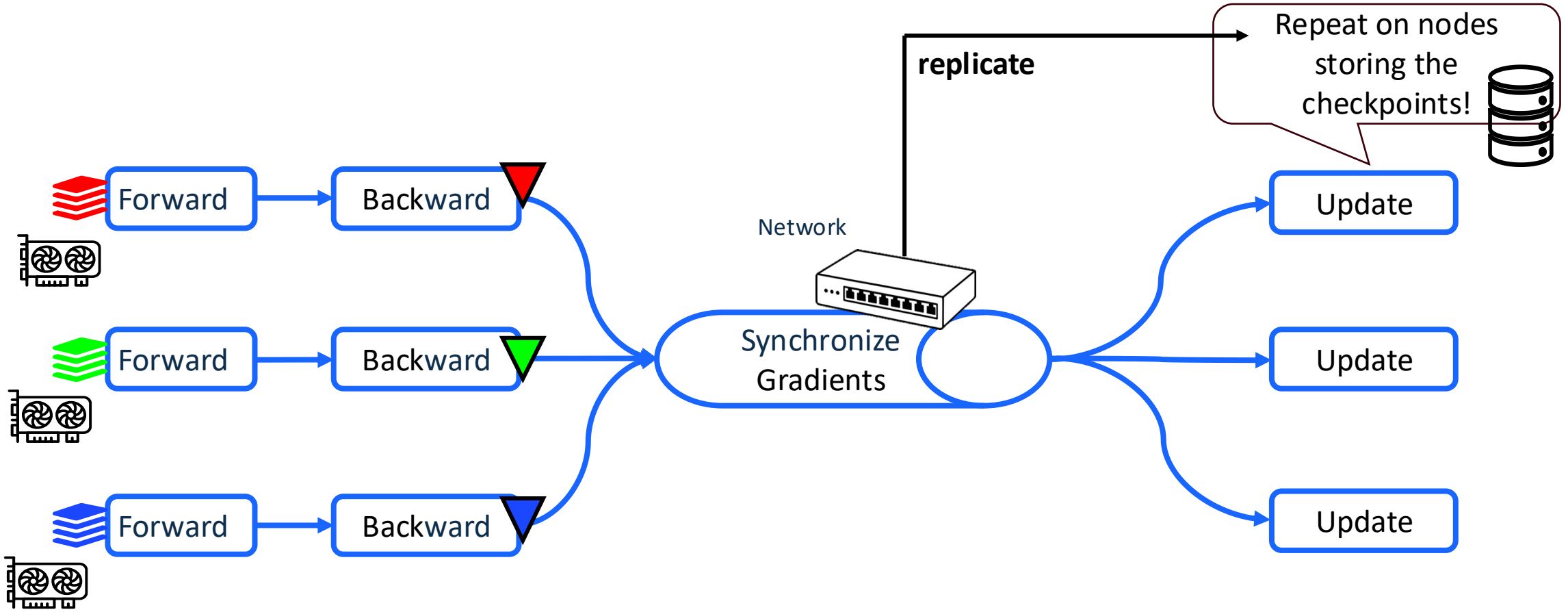
Checkmate: per-iteration checkpointing without GPU overhead

Observation: Model update step on the checkpoint



Great, where do we get the gradients for the nodes storing checkpoints?

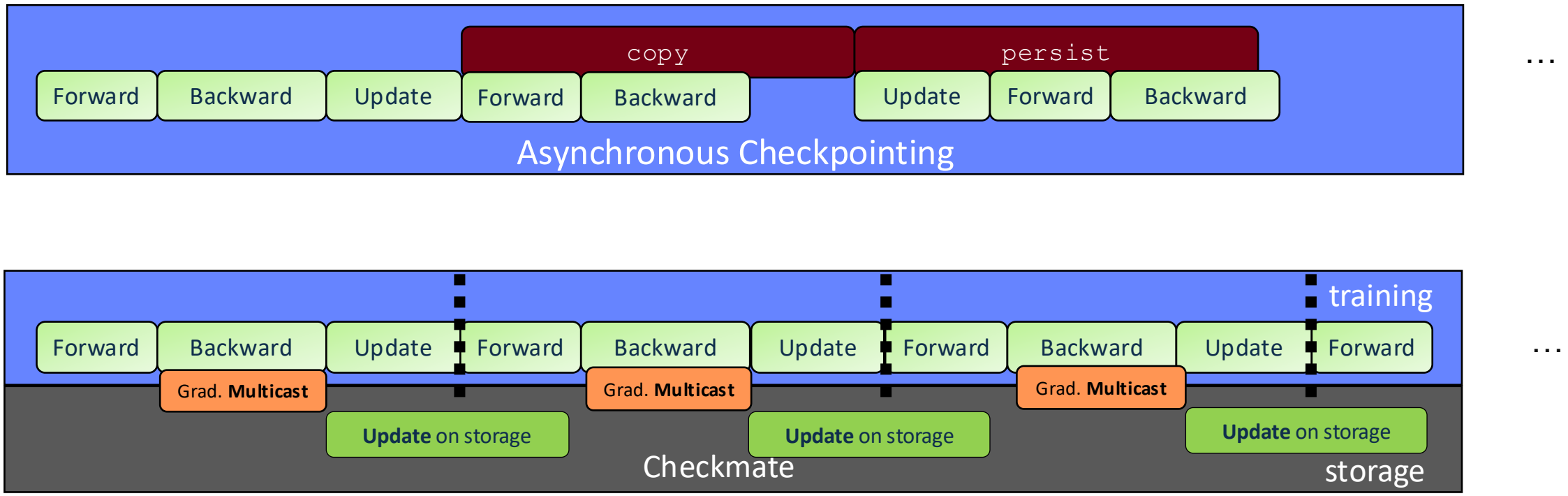
Key idea 2: Checkpointing gradients for optimization



Checkmate Overview

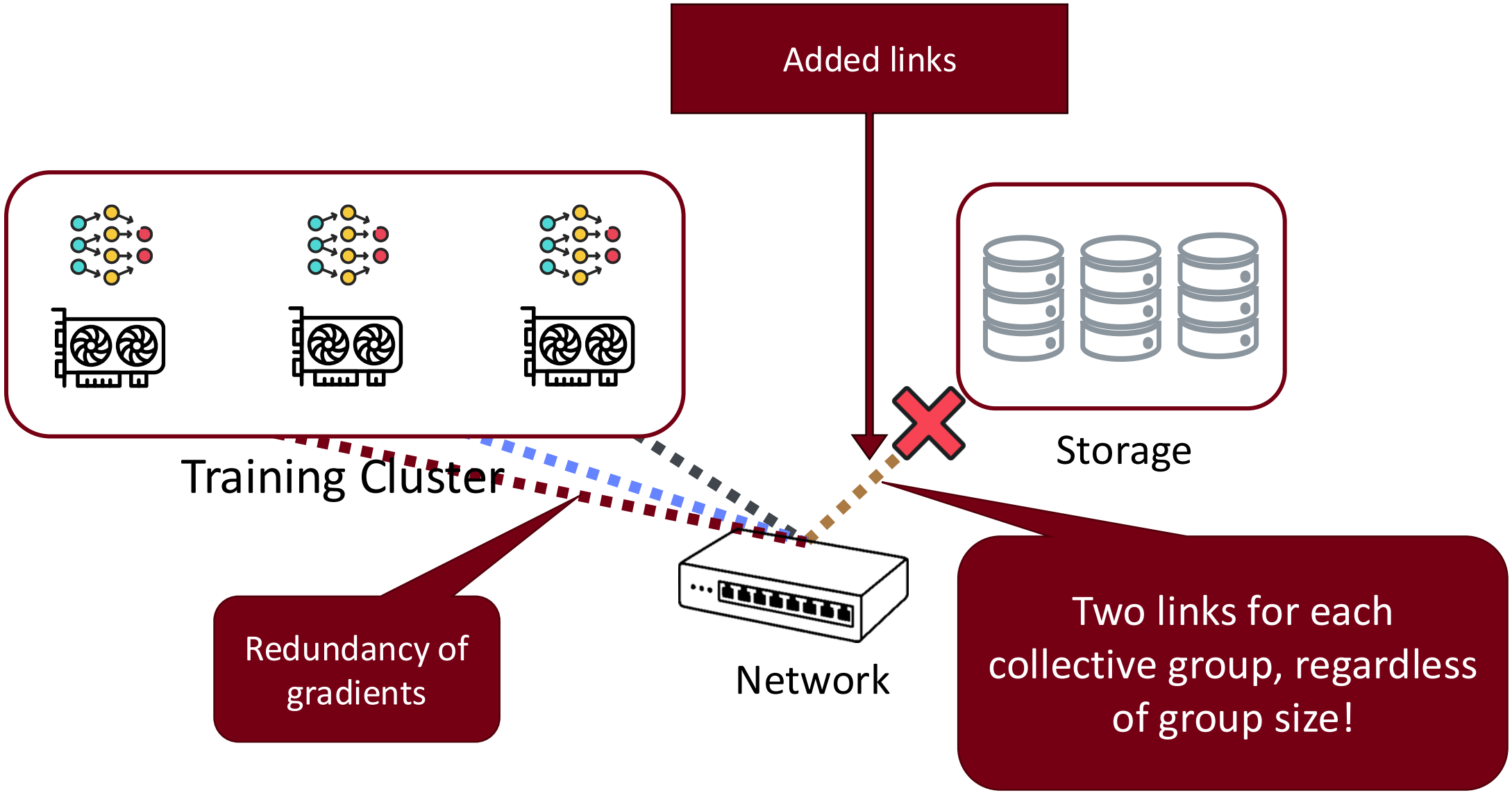
1. Un-interrupted and unmodified DNN training
   - NCCL-Plugin to assist network gradient replication
2. Programmable Switch
   - **Multicast** gradients to training and storage nodes in parallel
3. Storage node
   - Use gradients to maintain the **shadow** model and optimizer

`Multicast-update` abstraction in Checkmate



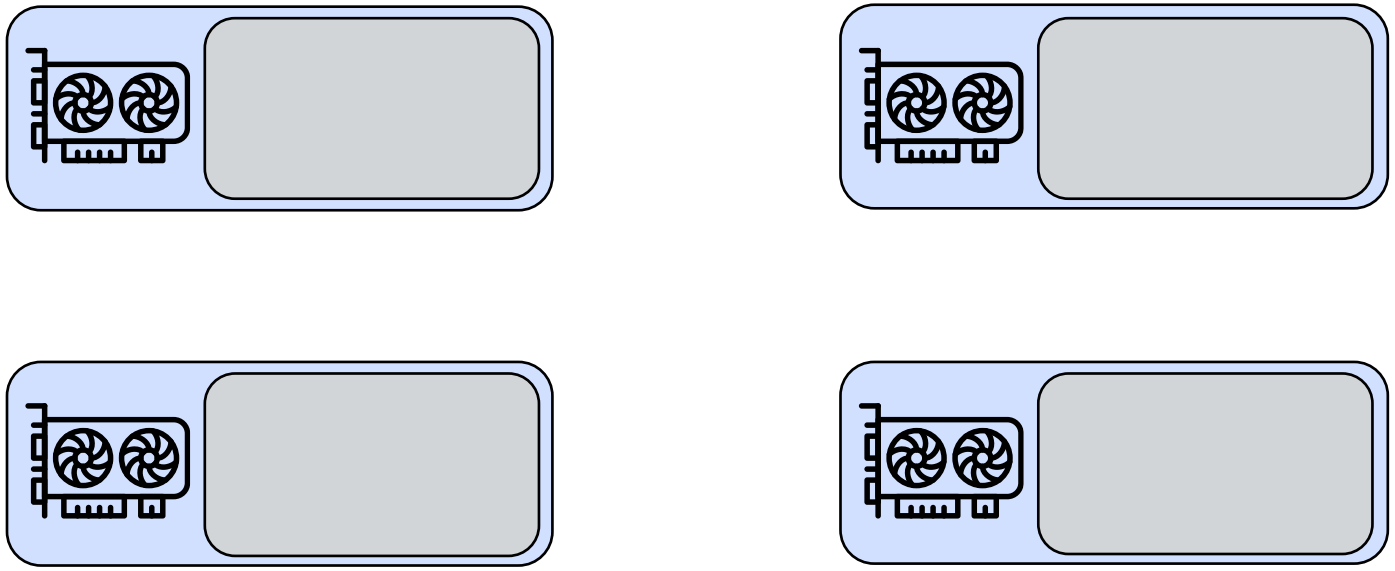
Time

Challenge 1: Too much gradient traffic



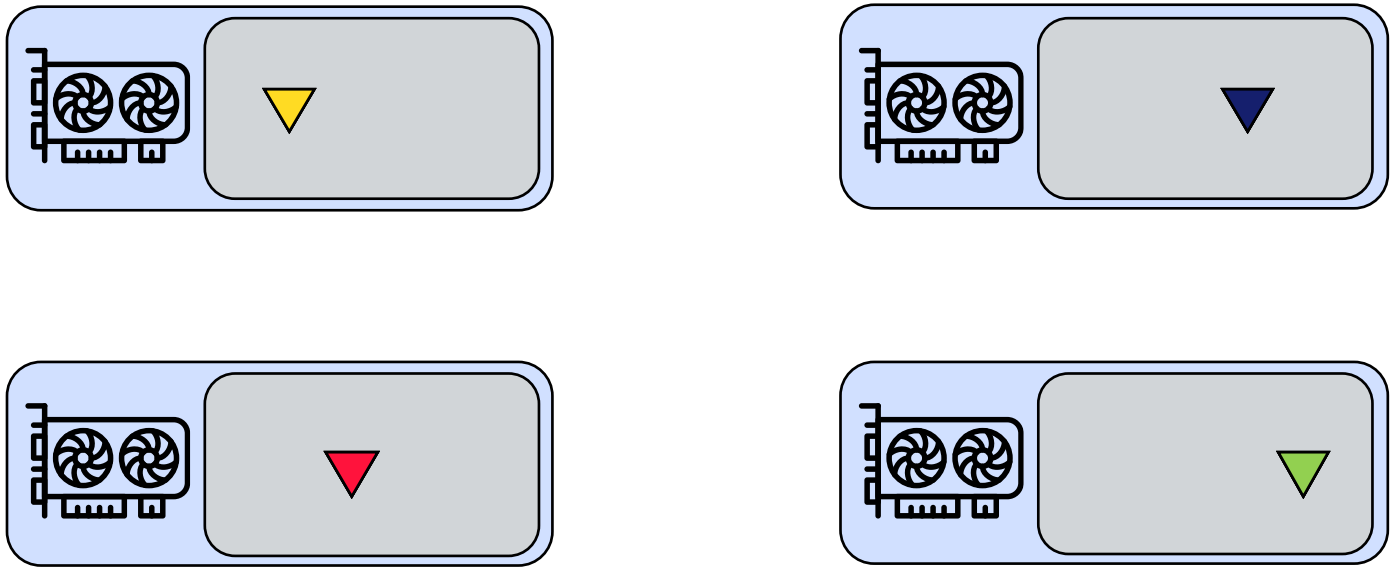
Solution 1: Selectively replicate gradients during AllReduce

- Synchronizing gradients  across four GPUs



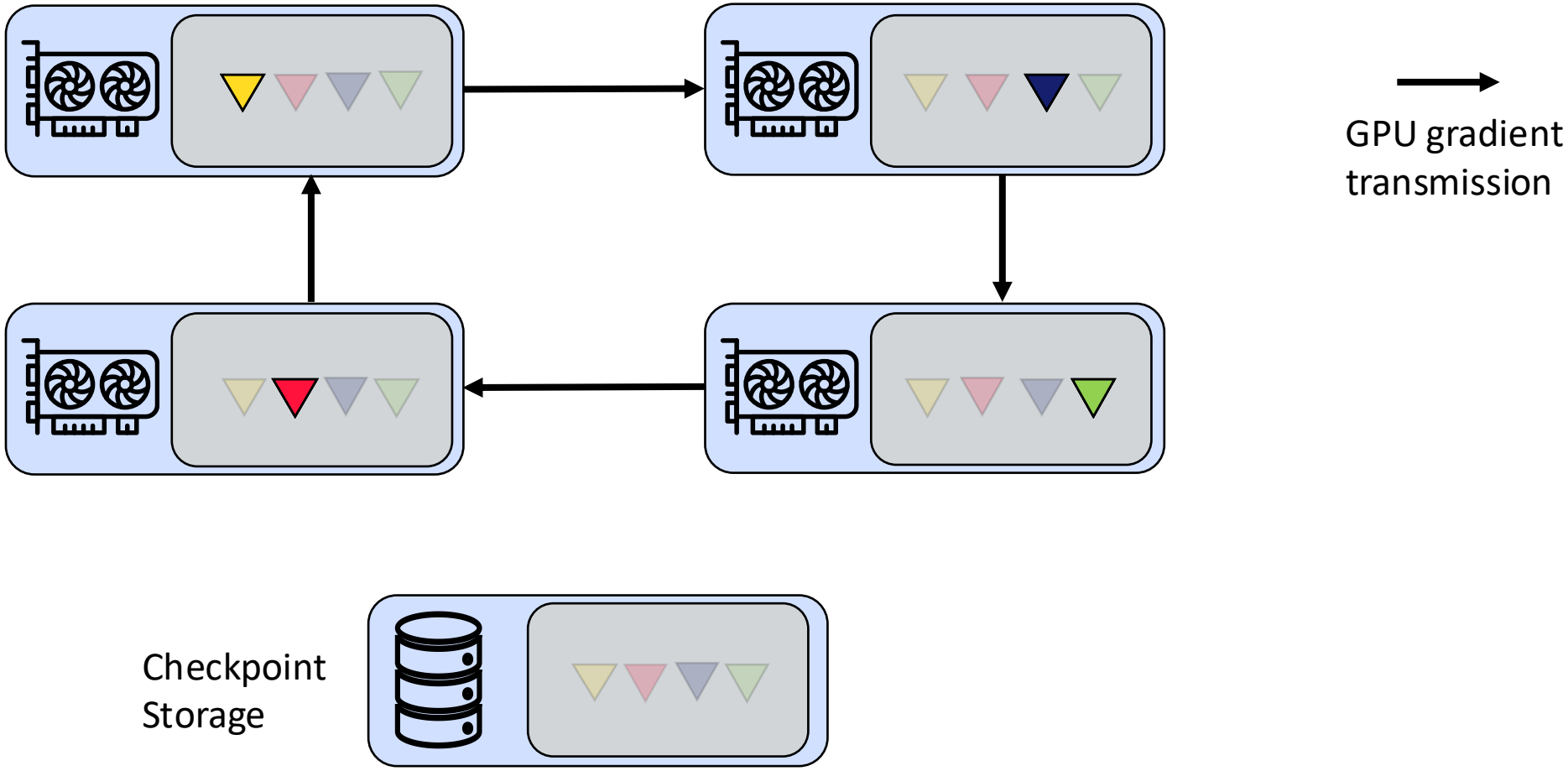
Solution 1: Selectively replicate gradients during AllReduce

- Synchronizing gradients  across four GPUs



Solution 1: Selectively replicate gradients during AllReduce

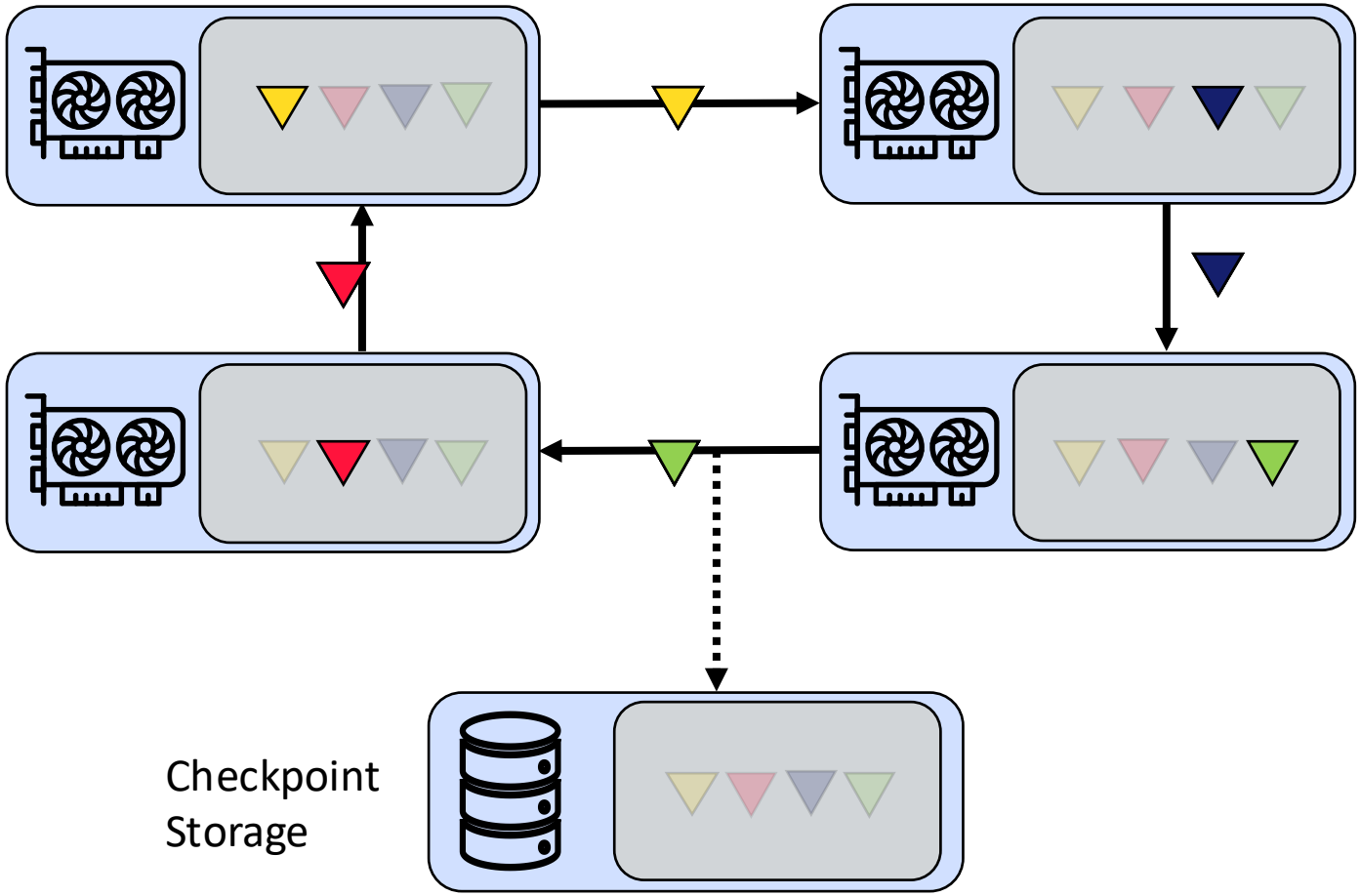
- Synchronizing gradients  across four GPUs

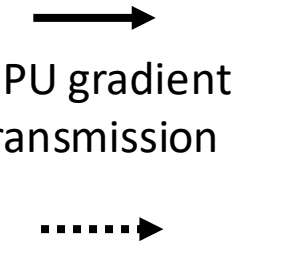


Solution 1: Selectively replicate gradients during AllReduce

- Synchronizing gradients  across four GPUs

Round 1

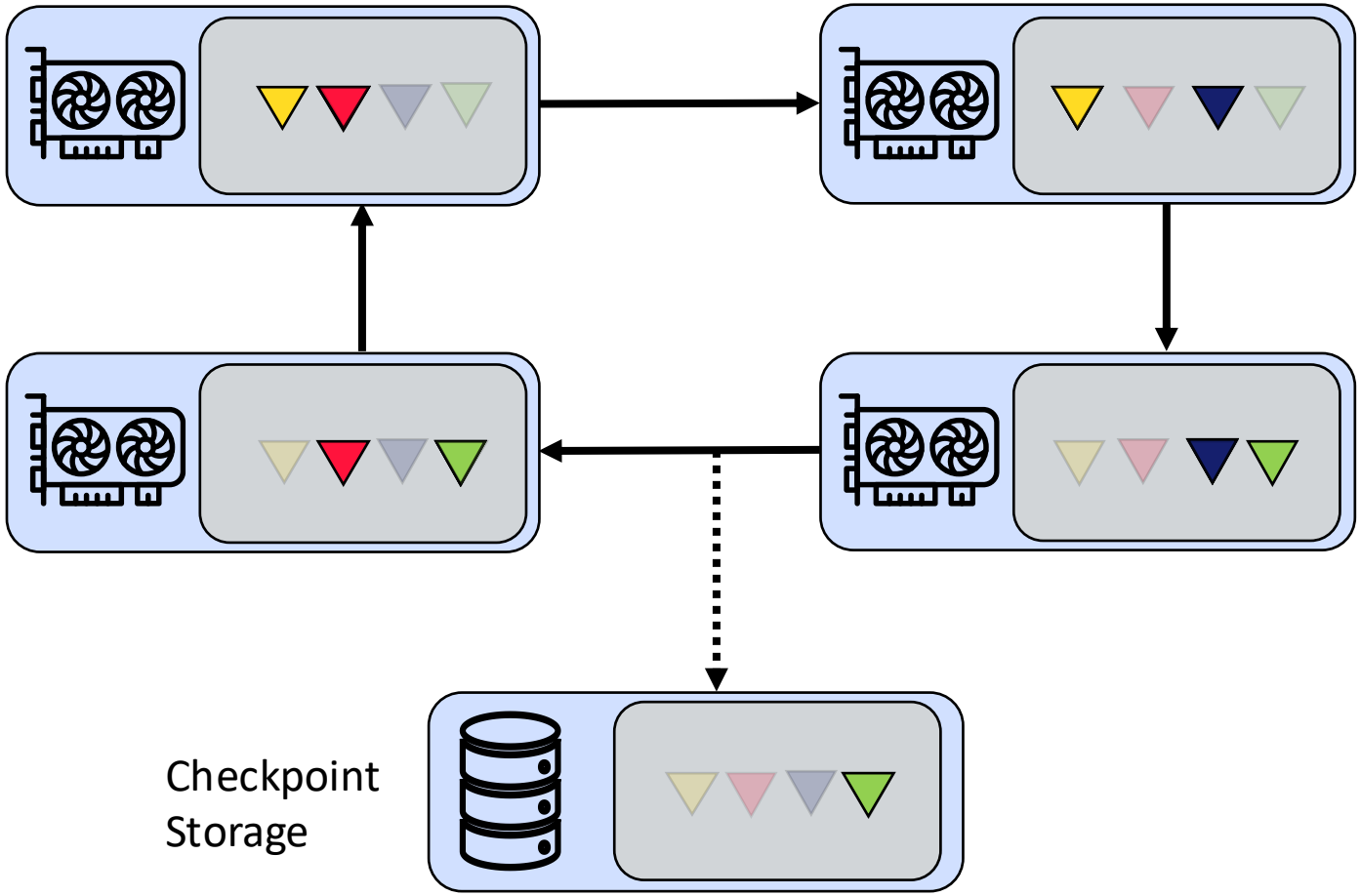


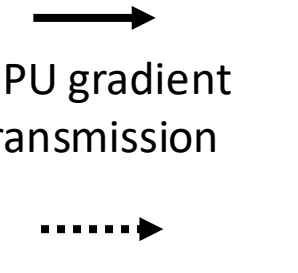


Solution 1: Selectively replicate gradients during AllReduce

- Synchronizing gradients  across four GPUs

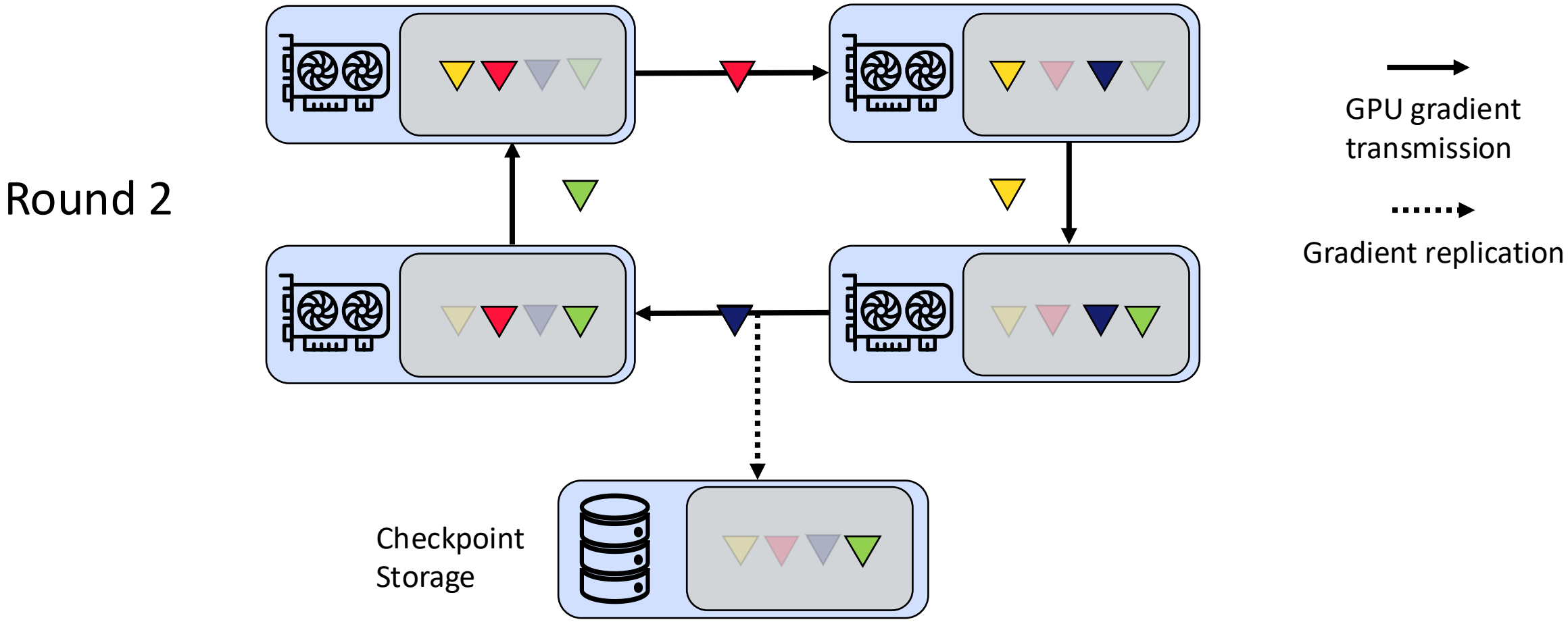
Round 1





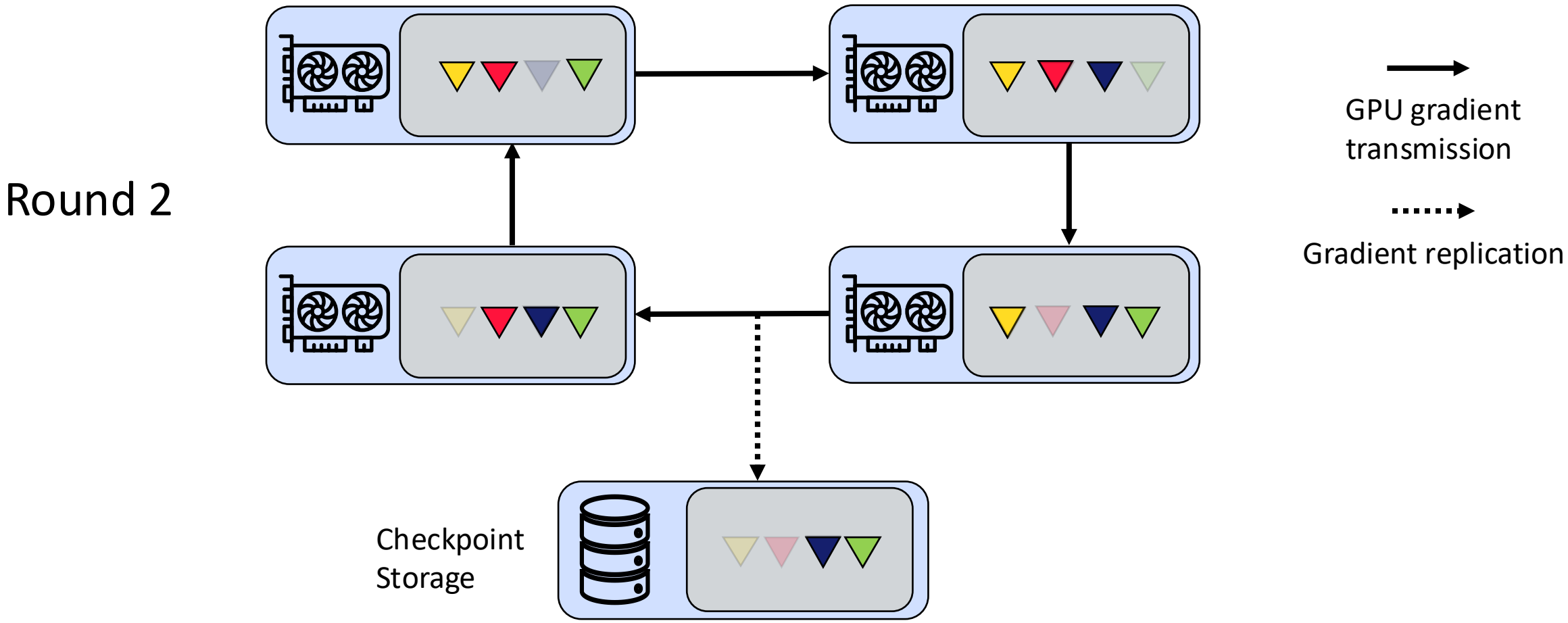
Solution 1: Selectively replicate gradients during AllReduce

- Synchronizing gradients  across four GPUs



Solution 1: Selectively replicate gradients during AllReduce

- Synchronizing gradients  across four GPUs

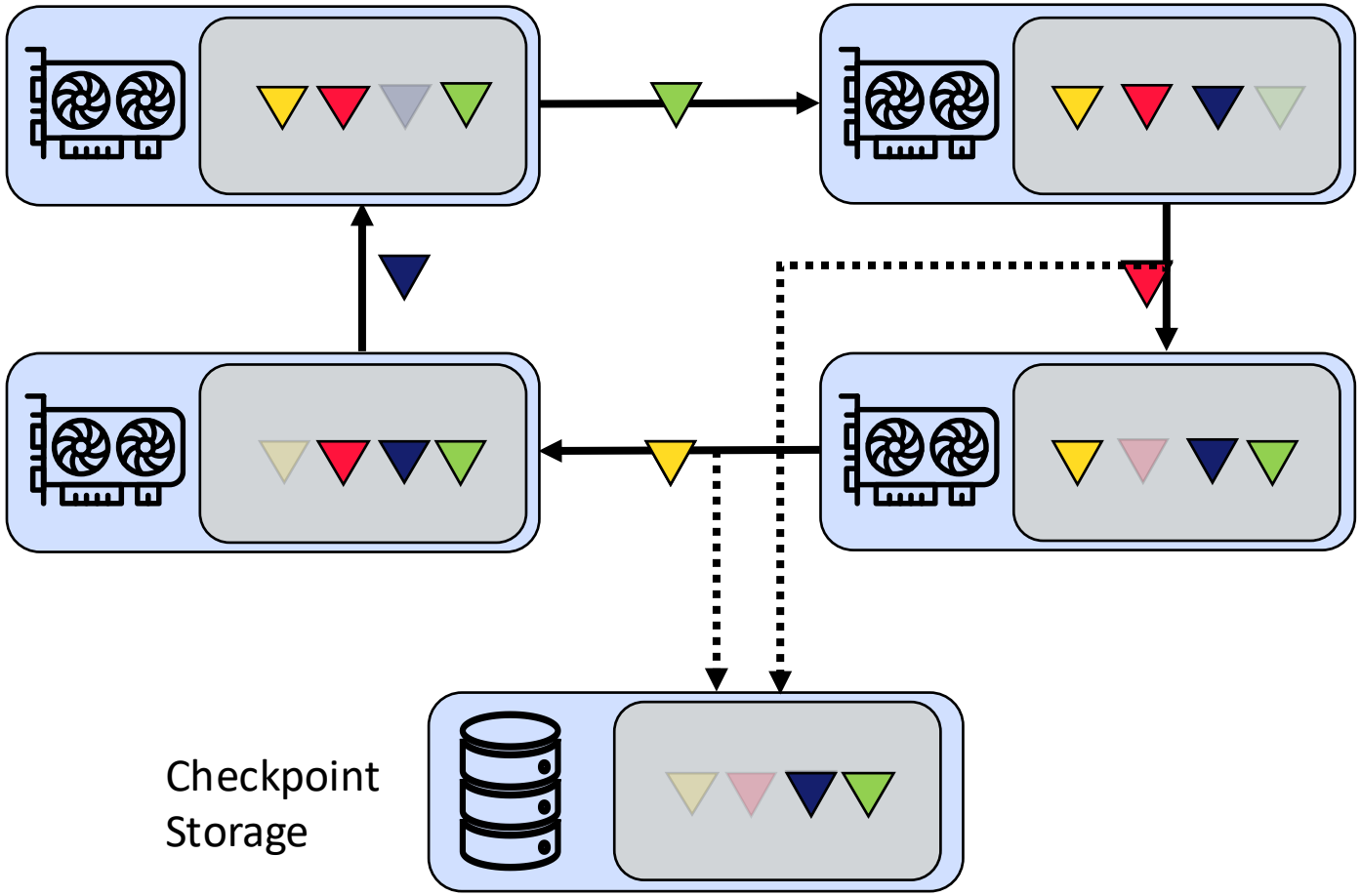


Solution 1: Selectively replicate gradients during AllReduce

- Synchronizing gradients  across four GPUs

Round 3

Last round!

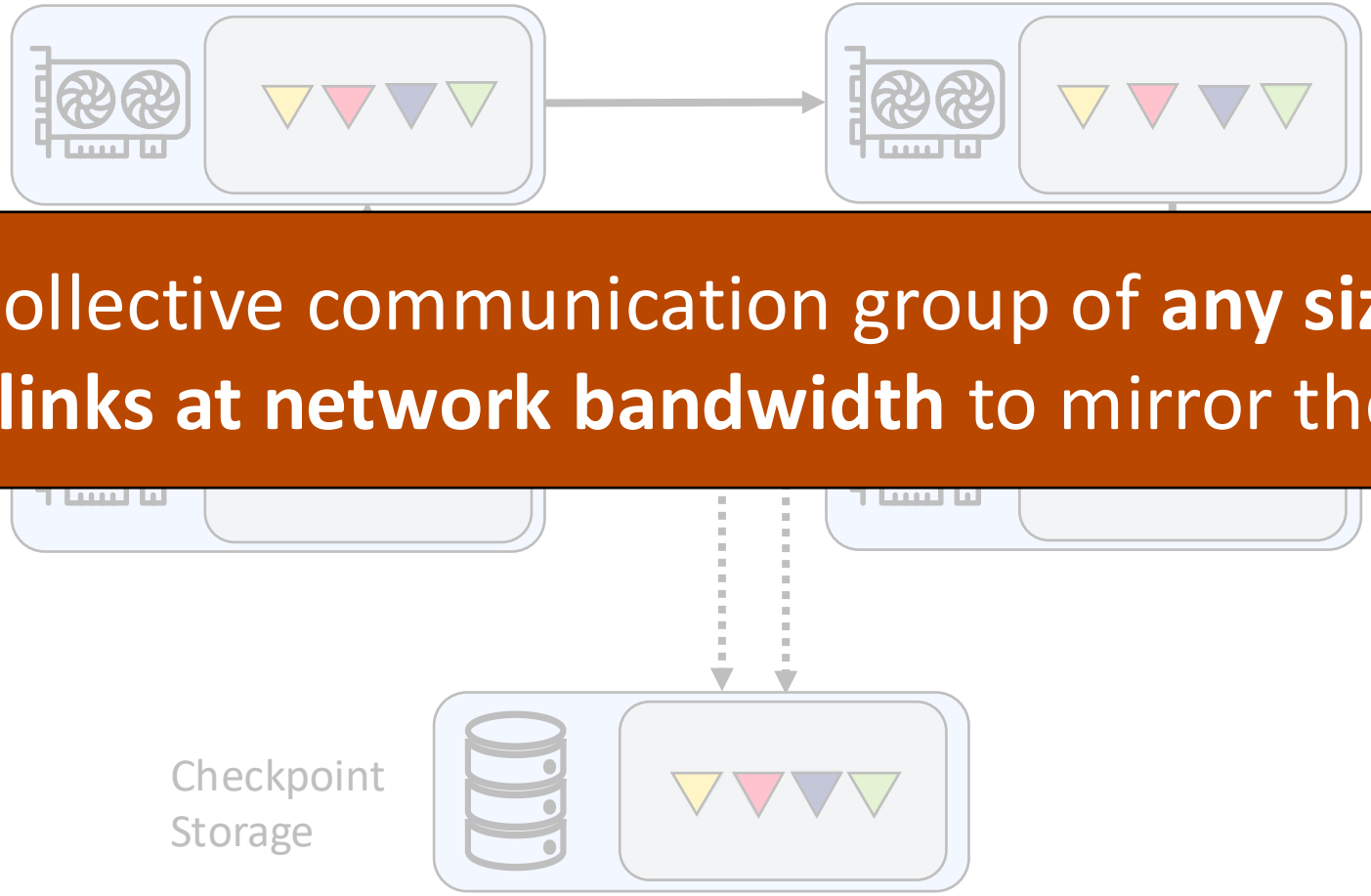


GPU gradient transmission

Gradient replication

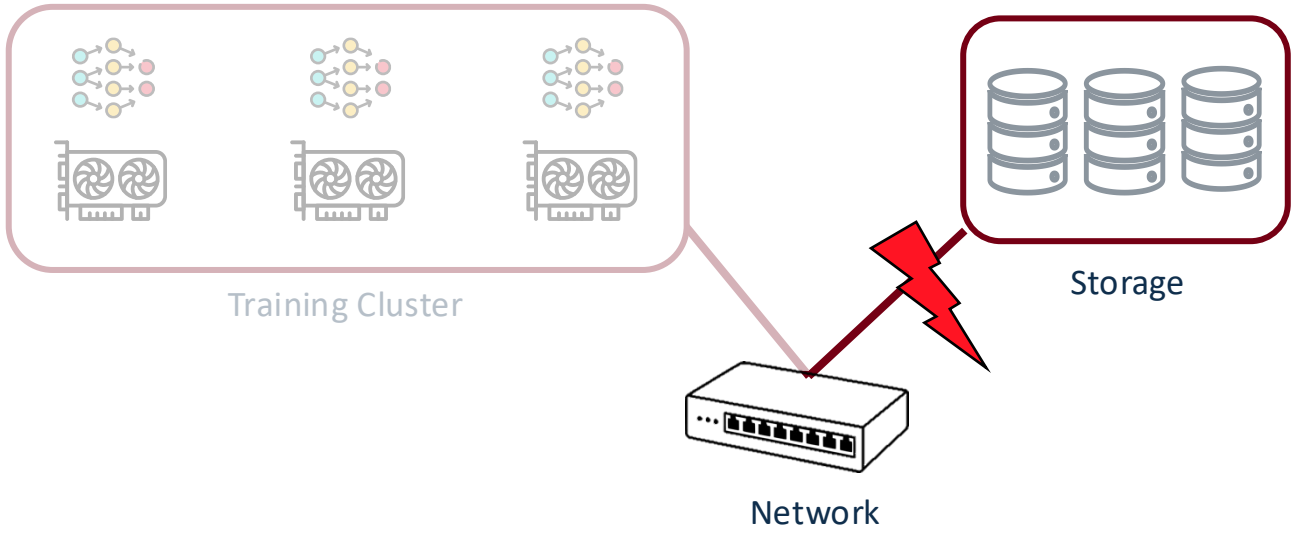
Solution 1: Selectively replicate gradients during AllReduce

- Synchronizing gradients  across four GPUs



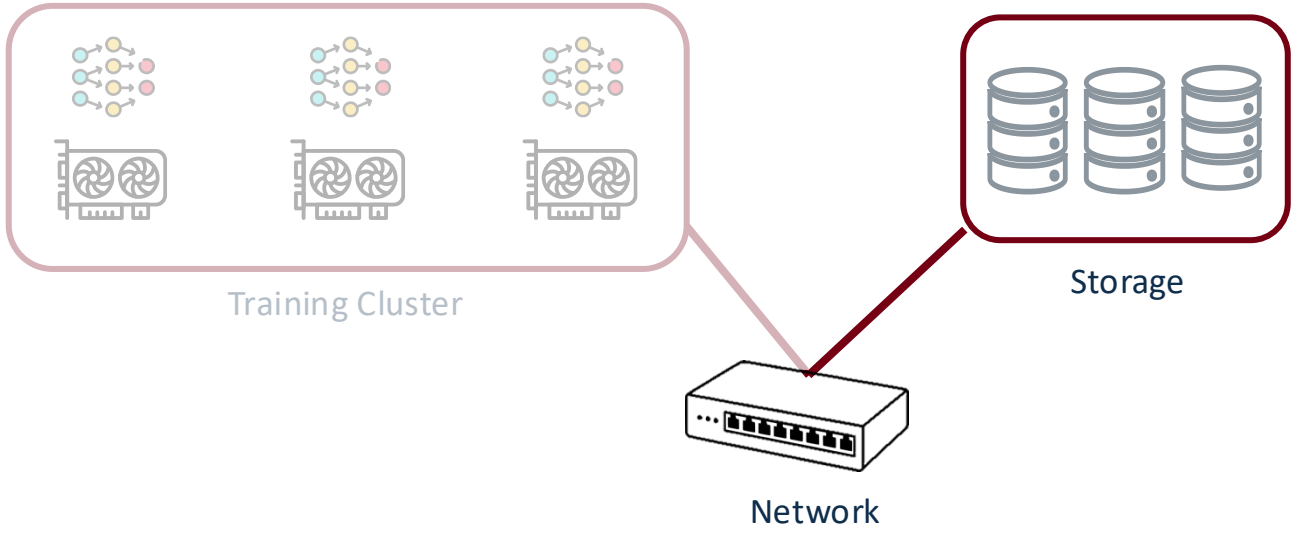
For each collective communication group of **any size**, we only need **two links at network bandwidth** to mirror the gradients

Challenge 2: Reliable delivery



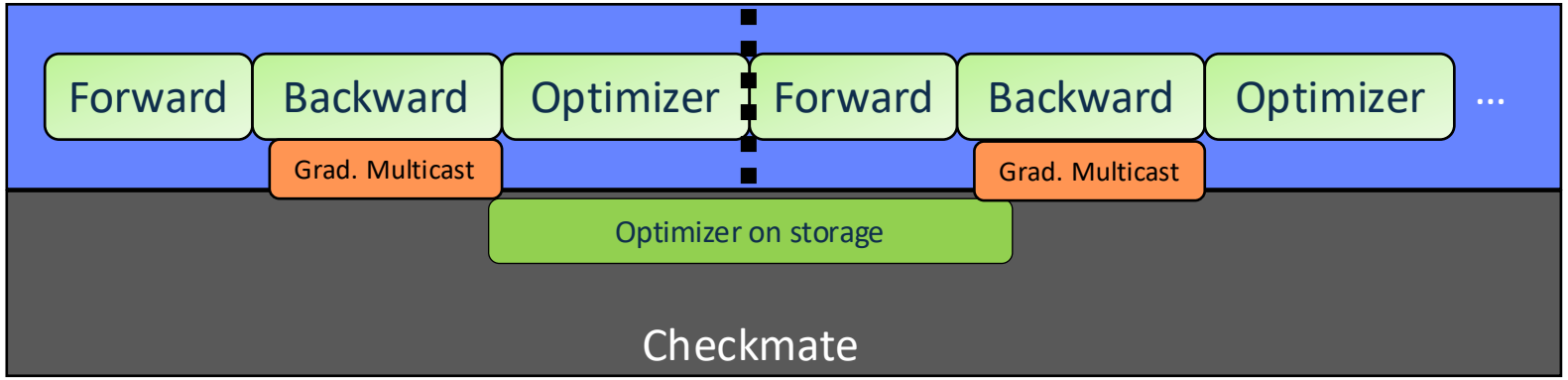
- The network must replicate *all* gradients to the storage cluster for consistency
- Need reliable communication for the multicast streams

Solution 2: Reliable delivery

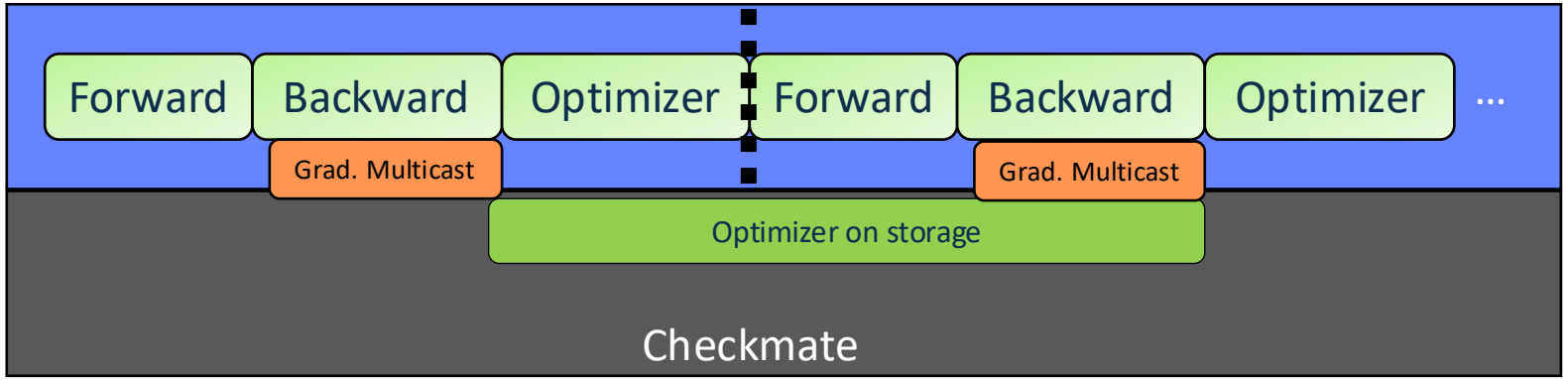


- Priority-based Flow Control (PFC)
  - Pause transmission when buffer is full
  - All streams in a multicasting flow can backpressure upstream sender
  - Tuned switch and NIC buffers
- Switch to storage connection
  - Handshake mechanism between switch and storage
  - Forward error correction (FEC) codes to handle data corruption

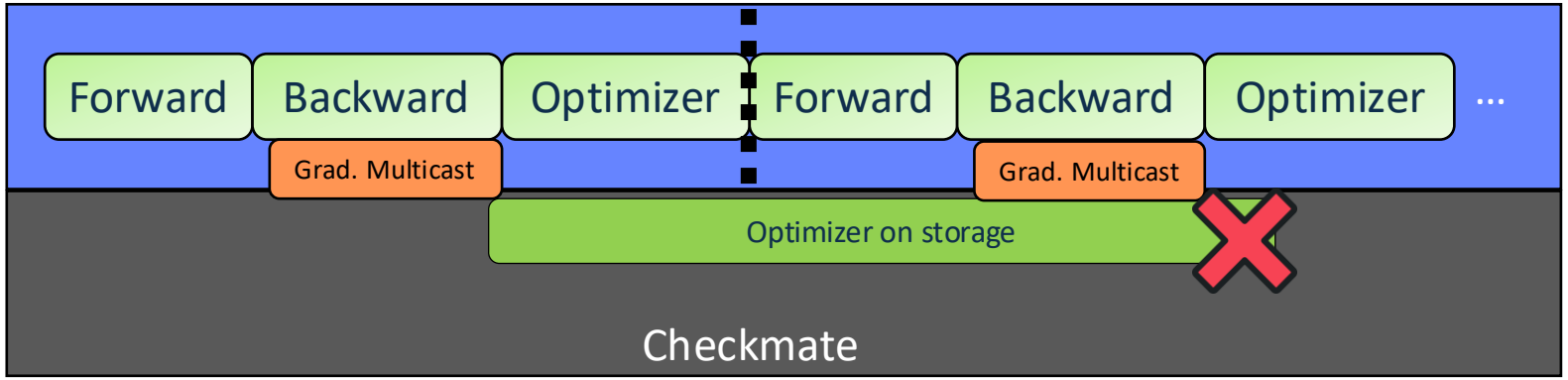
Challenge 3: Compute resource constraint on storage nodes



Challenge 3: Compute resource constraint on storage nodes

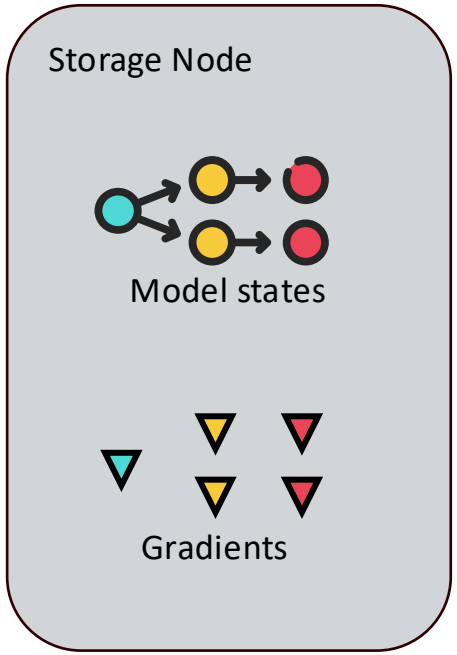


Challenge 3: Compute resource constraint on storage nodes



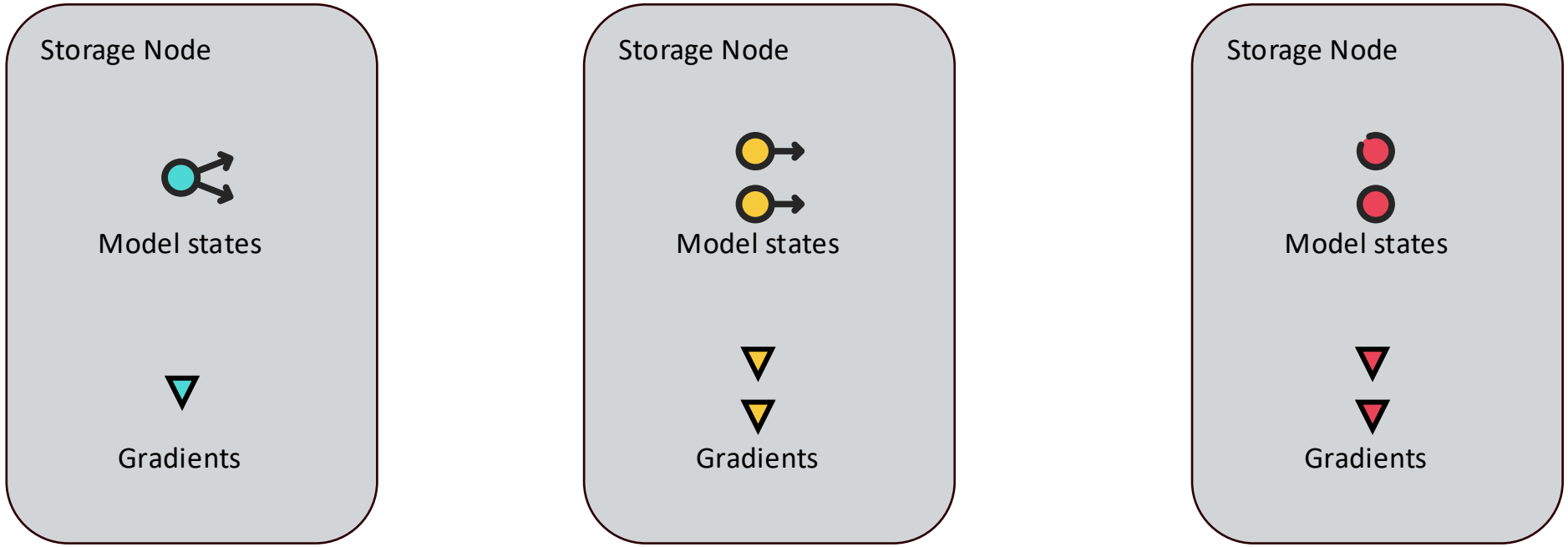
Solution 3: partition the checkpoint

- Most popular optimizers (e.g. SGD, Adam, AdamW) are *functional* and *independent:*
  - Each parameter updates independently, and the update depends only on the parameter itself, the gradient and the optimizer state



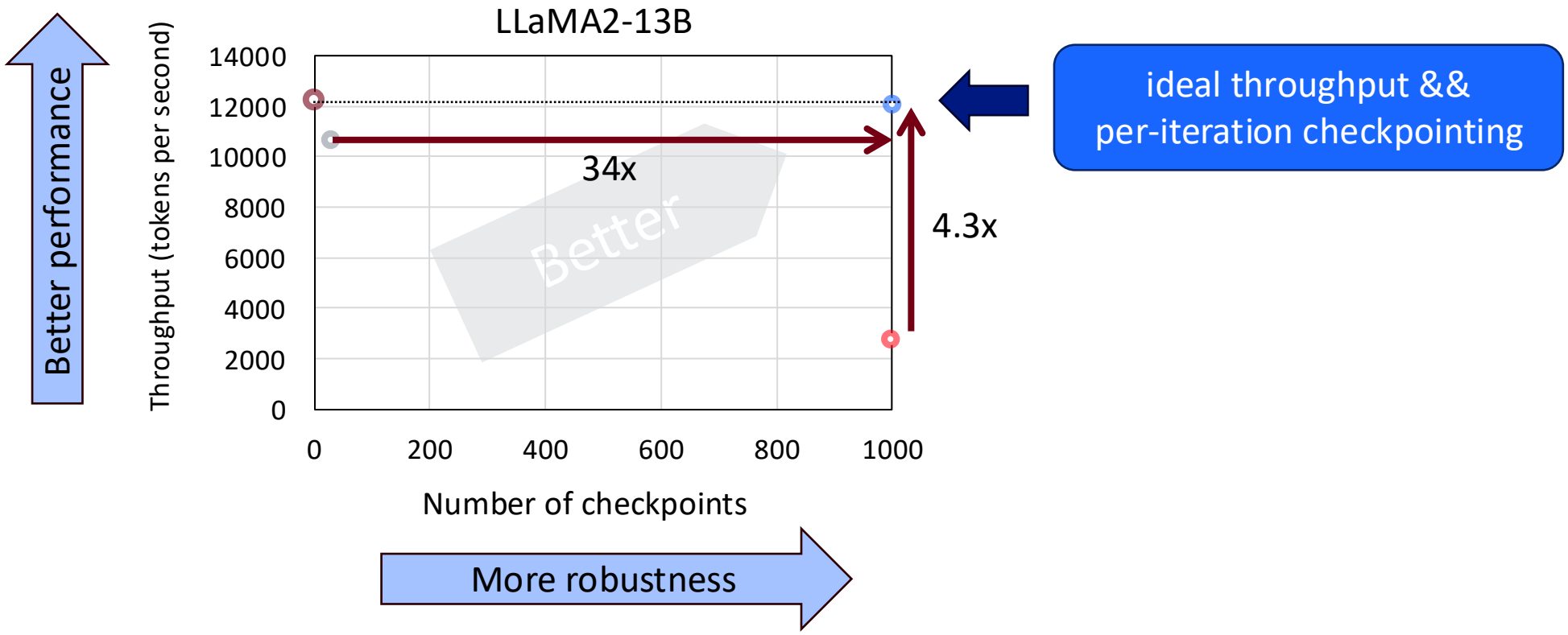
Solution 3: partition the checkpoint

- Most popular optimizers (e.g. SGD, Adam, AdamW) are *functional* and *independent:*
  - Each parameter updates independently, and the update depends only on the parameter itself, the gradient and the optimizer state



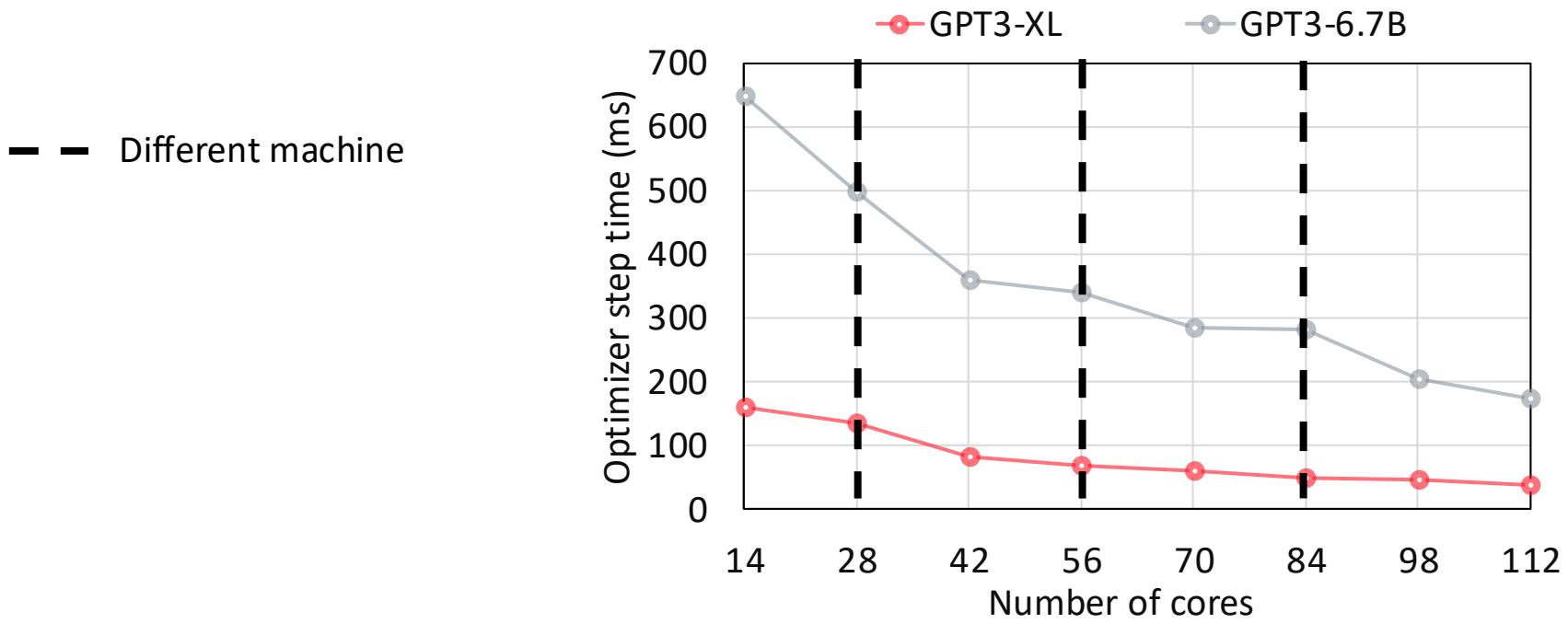
Result: Robustness at the ideal throughput

- Twelve A100 GPUs with 100 Gbps network; four storage CPU servers



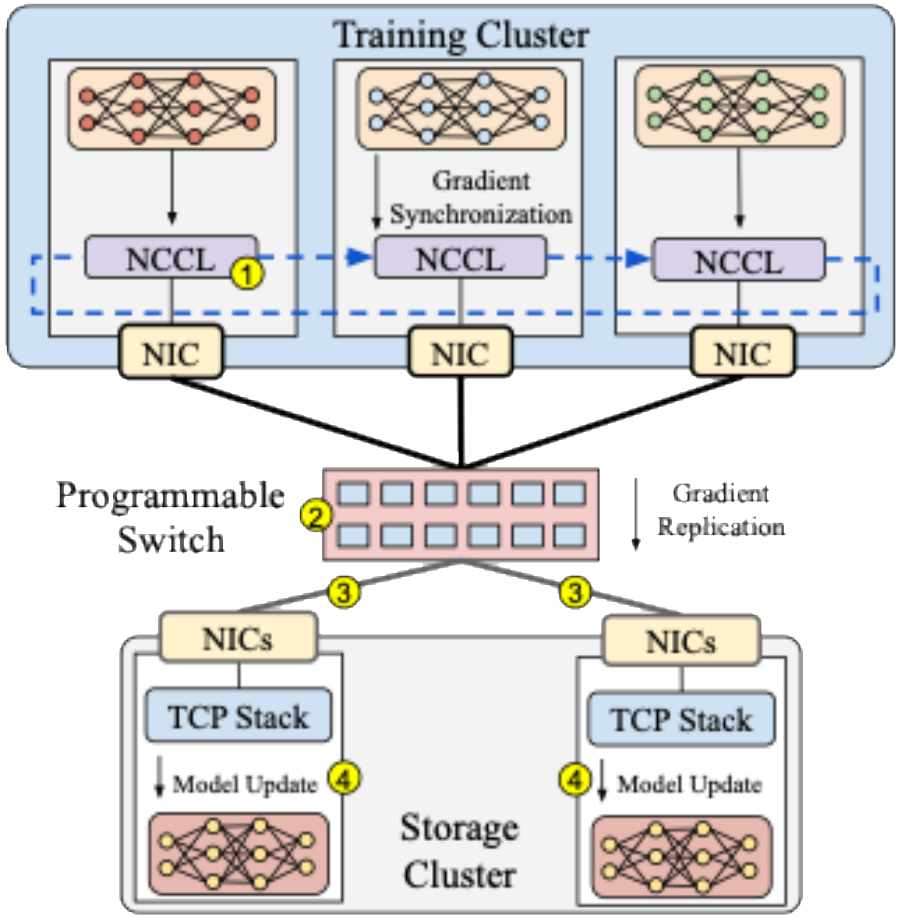
Result: Storage optimizer performance scaling

- One to four 28-core CPU servers running optimizer steps



- A single 28-core CPU server keeps up with training a 6.7B-parameter GPT model with reasonable batch size (local batch size of 5 sequences)

Summary



- Checkmate: zero-GPU overhead checkpointing with network assistant
- Multicast-update abstraction to replicate gradient in the network and repeat update step on the storage node
- 5 - 34.5× more frequent checkpointing
- 1.3 - 6.5× throughput at the same checkpointing frequency
- https://github.com/hipersys-team/checkmate