



Elastic Expert Parallelism in vLLM

Reconfiguring live multi-GPU inference during traffic

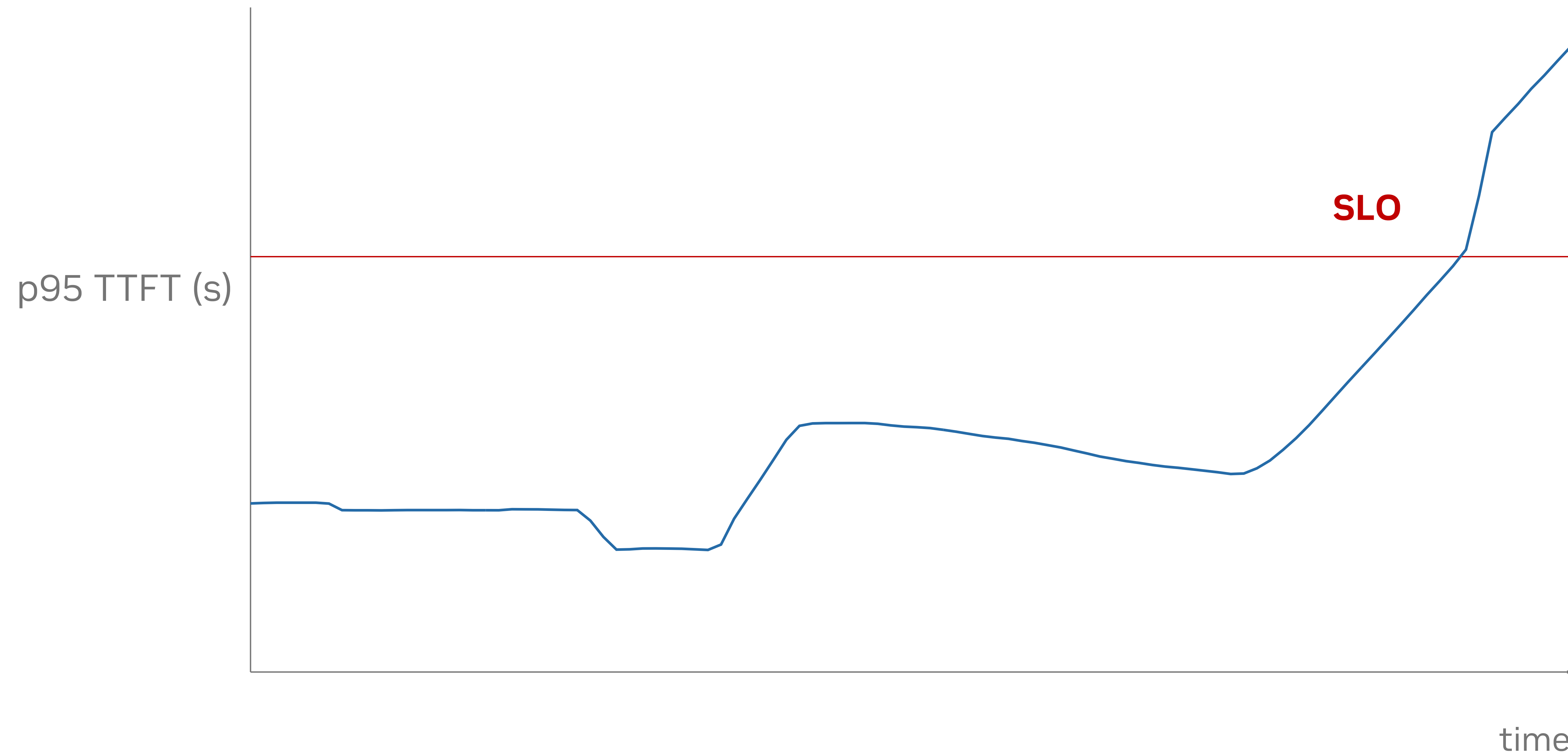
Itay Alroy | Senior Software Architect, NVIDIA

PyTorch Conference NA 2026



Inference workloads change over time

DeepSeek V4 Pro | 16 GB200 GPUs | SLO: p95 TTFT below target



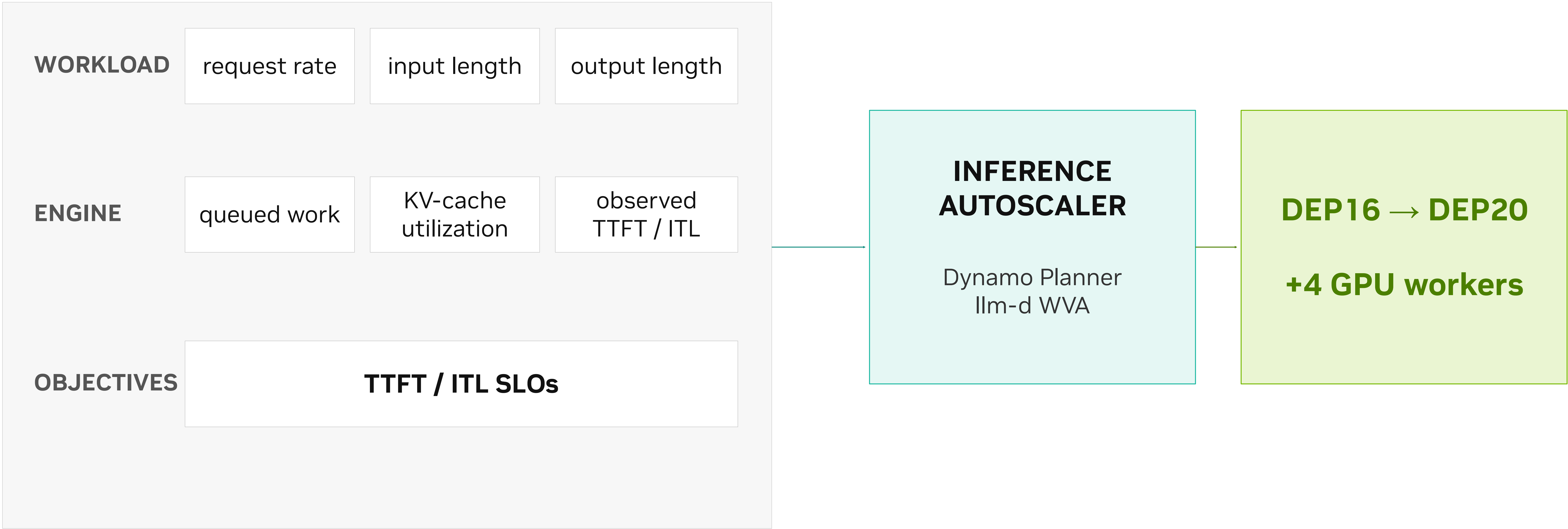
WORKLOAD CHANGES

- More concurrent requests
- Longer prompts (ISL)
- Longer generations (OSL)

Autoscaling for inference

Workload + engine signals → capacity target

SIGNALS AN INFERENCE AUTOSCALER CAN CONSIDER



Static engines force overprovisioning or downtime

Before Elastic EP, vLLM could not change GPU membership while serving

HORIZONTAL SCALING



VERTICAL SCALING



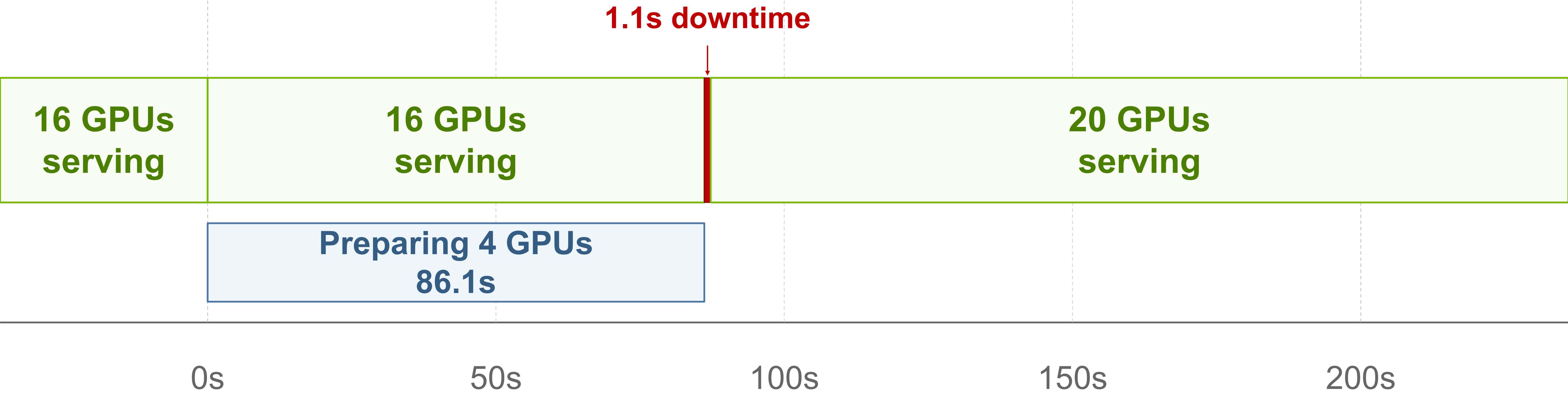
Elastic EP enables fine-grained scaling

DeepSeek V4 Pro | GB200 | 16 → 20 GPUs

Destroy/recreate
16 → 20 GPUs



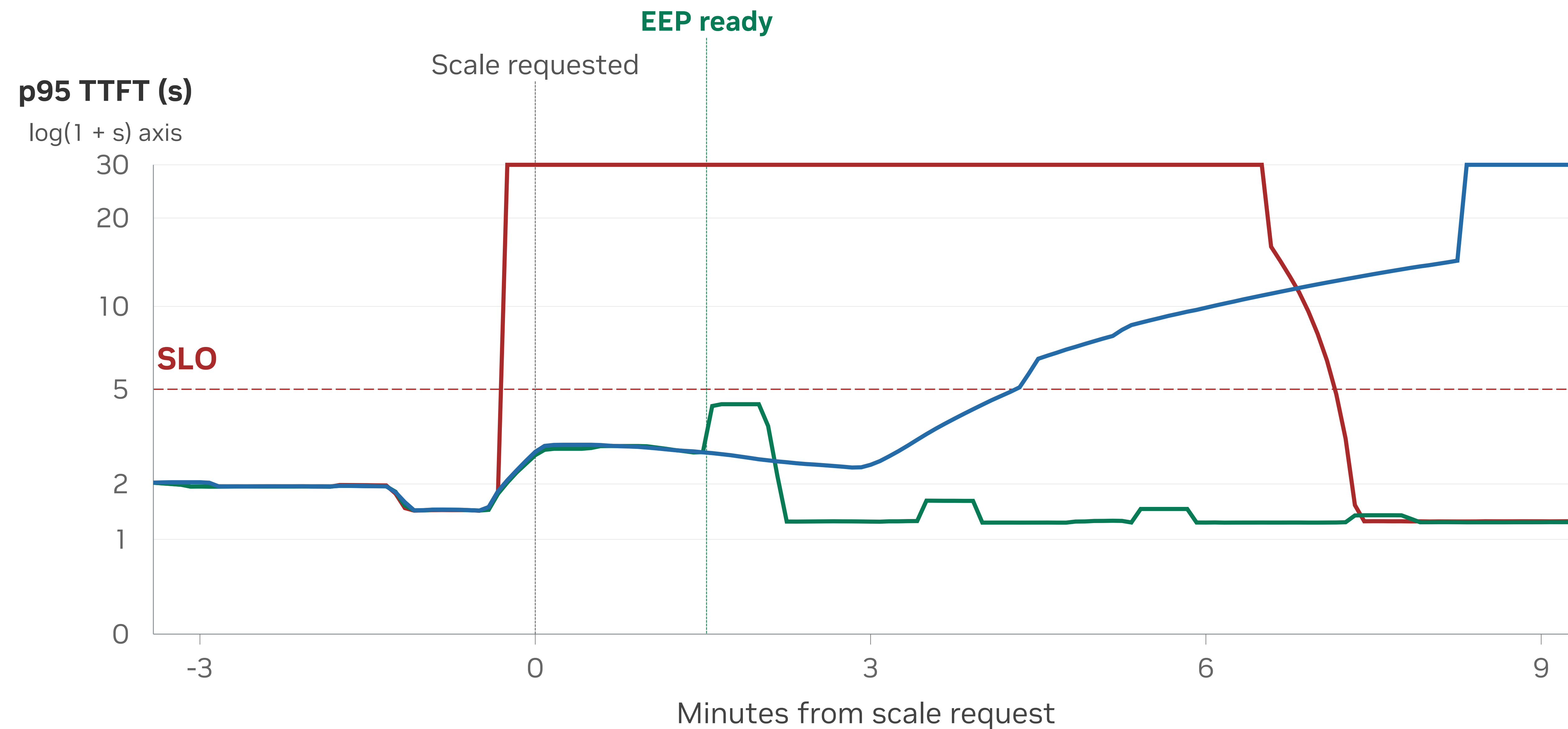
Elastic EP
with NIXL EP
16 → 20 GPUs



elapsed seconds after resize request

Elastic EP kept p95 TTFT below the SLO

DeepSeek V4 Pro | GB200 | same 11,391-request trace, replayed three ways



THREE APPROACHES

STAY AT 16 GPUS

p95 crosses the SLO and stays above it through the run.

RESTART TO 20 GPUS

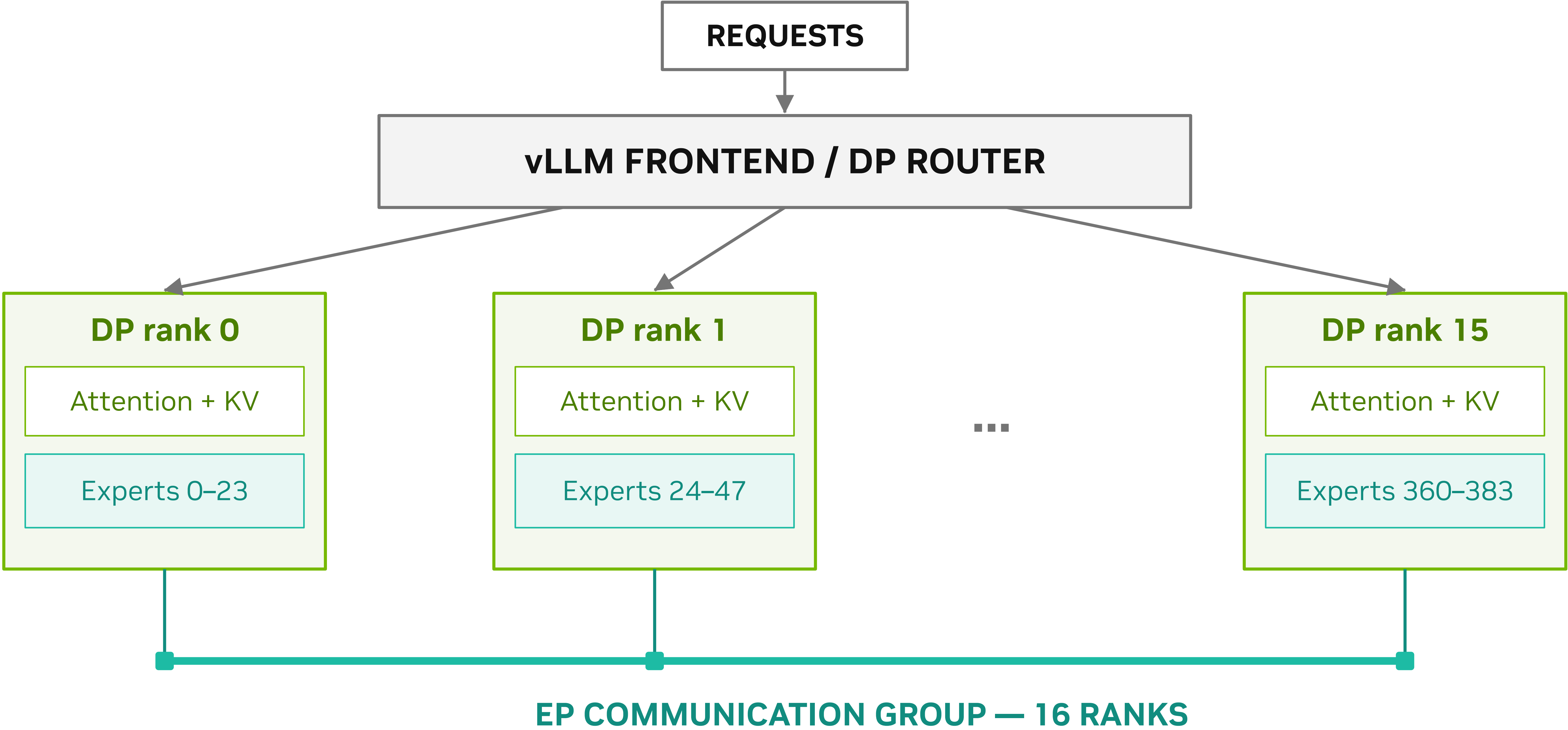
Output stops for 197 s; p95 returns below the SLO at +7 min.

ELASTIC EP TO 20 GPUS

p95 stays below the SLO throughout scale-up.

Wide EP system overview

DP16 | TP1 | EP16 | each request is assigned to one DP rank

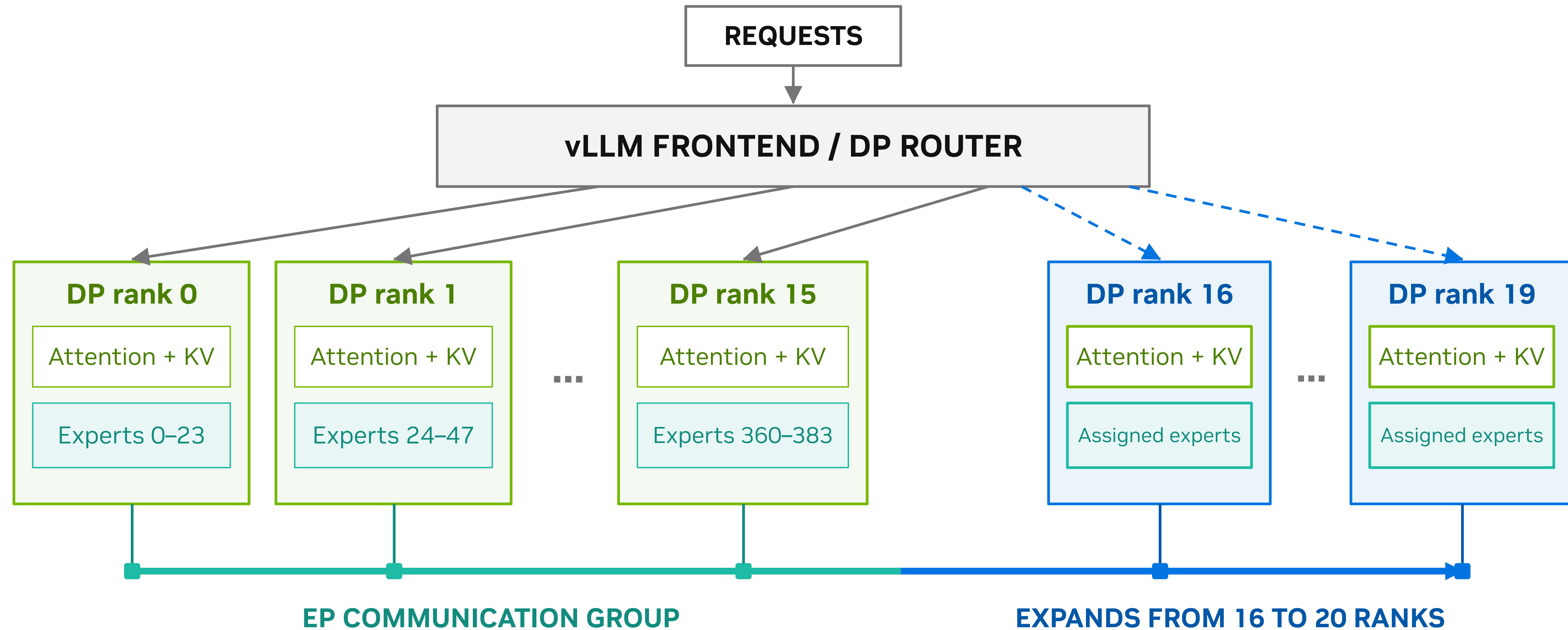


Attention + KV cache stay local to the selected DP rank

384 routed experts / layer — 24 / EP rank

Dynamic reconfiguration of Wide EP

DEP16 + 4 joining ranks = one DEP20 / EP20 engine



1 START RANKS

launch four joining processes

2 LOAD WEIGHTS

attention weights from
retained ranks
via NVLink / RDMA

3 PLACE EXPERTS

choose placement
copy expert weights

4 EXPAND EP GROUP

connect new ranks
without pausing serving

Existing EP libraries could not expand a live communicator

Scale-up meant preparing a second communicator

TWO EP COMMUNICATORS

Both include DP0–DP15

EP16 COMMUNICATOR / RETIRING

RANKS DP0, DP1, ..., DP15

EP communication buffers

Peer connections, registrations / handles

EP20 COMMUNICATOR / SERVING

RANKS DP0, DP1, ..., DP19

EP communication buffers

Peer connections, registrations / handles

2× EP COMMUNICATION HBM

NO REUSE OF MEMORY REGISTRATIONS,
CUDA IPC HANDLES, OR CONNECTIONS

SWITCHING COMMUNICATORS REQUIRES CUDA GRAPH RECAPTURE
57 SECONDS OF SERVING DOWNTIME
DEEPSEEK V4 PRO · DEP16

NIXL EP expands the live communicator in place

SAME NIXL EP COMMUNICATOR THROUGHOUT

Membership changes behind the same NIXL agent and buffers



RETAINED CUDA GRAPHS STAY VALID

EP launch configuration, kernel arguments, and input/output tensor addresses stay stable

Joining ranks still need CUDA graphs

NIXL EP lets each joining rank capture locally while retained ranks keep serving

WITHOUT NIXL EP

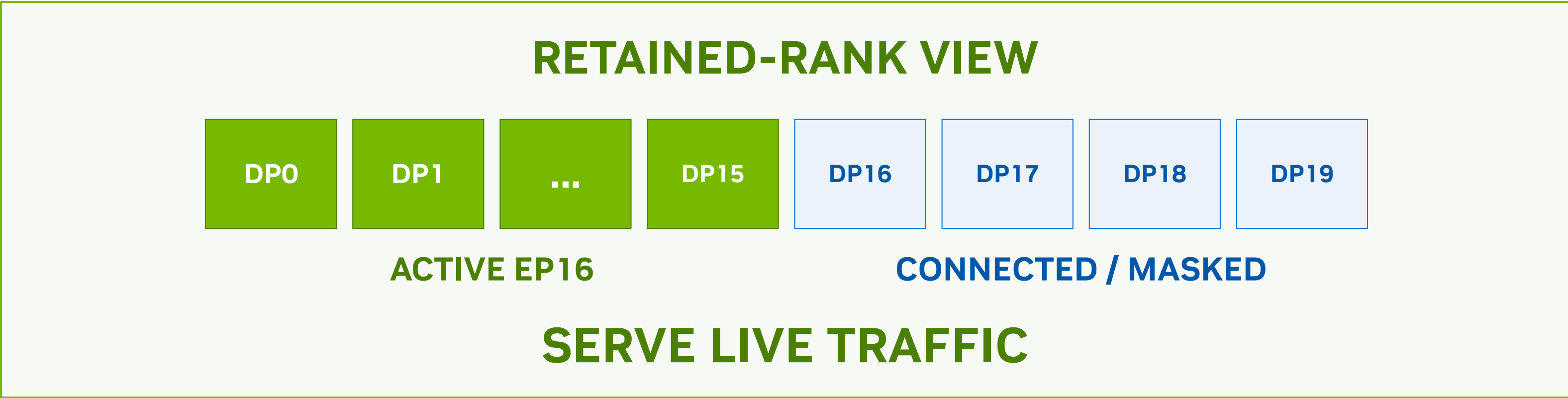
GRAPH PREPARATION RUNS DISTRIBUTED MODEL FORWARDS



Retained ranks must join preparation — serving pauses

WITH NIXL EP

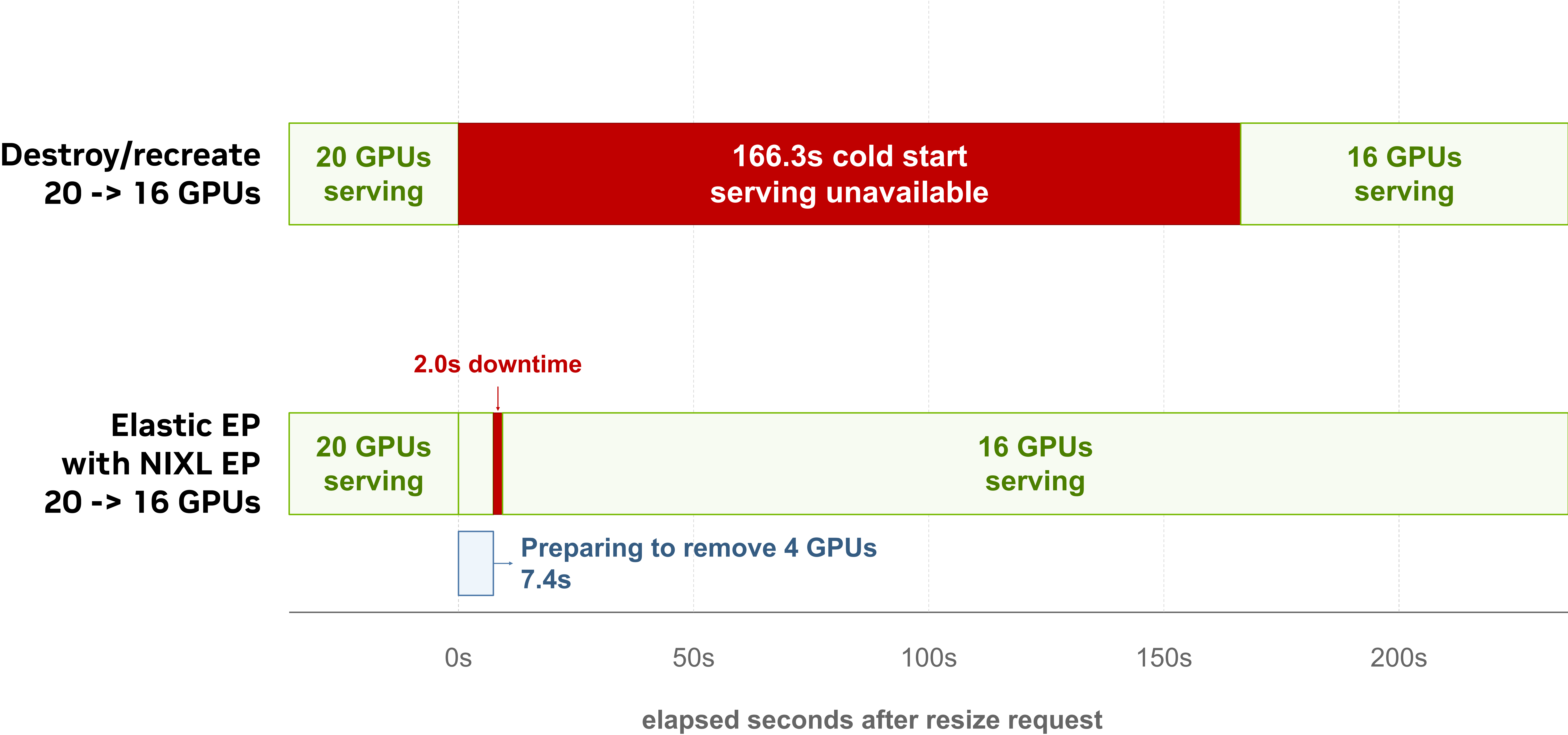
ALL REMOTE RANKS ARE MASKED DURING CAPTURE



RETAINED RANKS SERVE WHILE JOINING RANKS CAPTURE INDEPENDENTLY

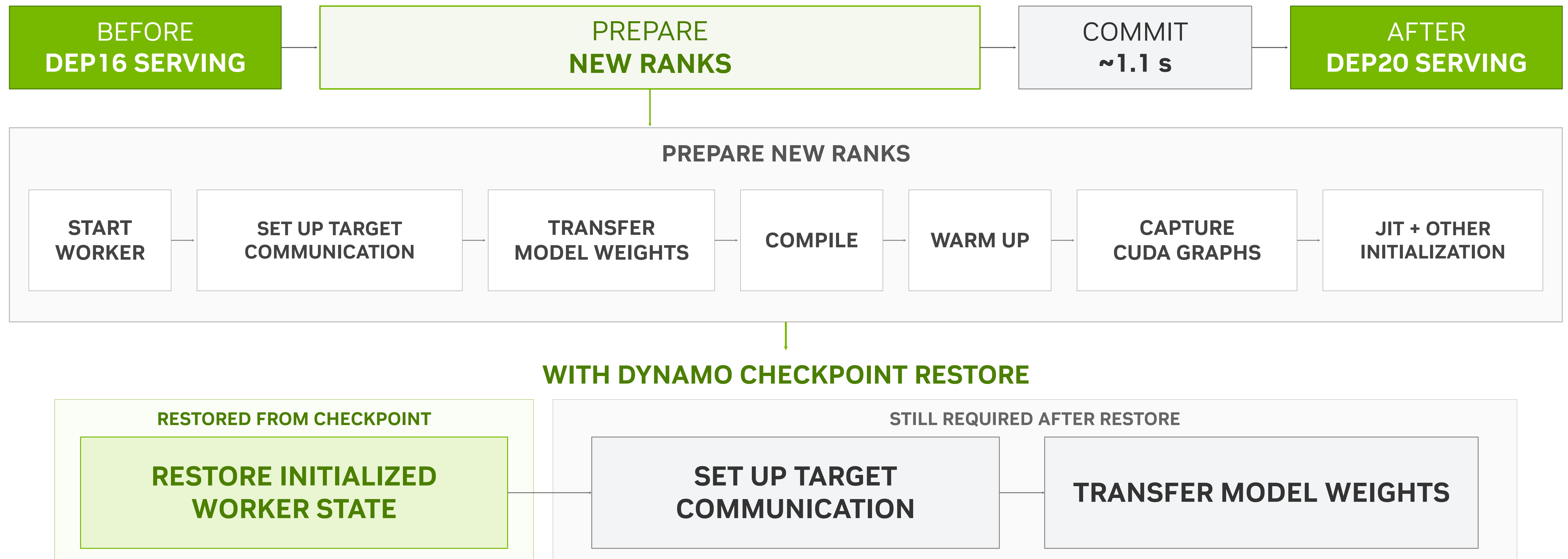
Elastic EP scale-down: 9.4s end to end, 2.0s downtime

DeepSeek V4 Pro | GB200 | 20 → 16 GPUs



Dynamo checkpoint restore can shorten preparation

Restore initialized worker state; still transfer model weights and set up target communication



Elastic EP works out of the box.

Everything shown today is available
with the latest vLLM and NIXL releases.

START HERE

[vLLM](#)

[NIXL](#)

[Elastic EP example](#)

[Elastic EP blog post](#)

Deploying inference at scale?

Let's get Elastic EP running in production.