PyTorch Europe Conference — Paris 2026

From GPU Efficiency to Governance

Protecting Latency SLOs in the Blackwell Era

Akhil Gupta | Banking AI Infrastructure

Key claim: Blackwell GPUs improve median latency but destabilise tail latency — governance closes the gap.

THE CHALLENGE

Blackwell Tail Latency Paradox

GPU efficiency gains

Blackwell/NVFP4 push utilization to 90–95%, removing natural bandwidth governors

p99 spikes despite p50 gains

Median latency improves 30–40% but tail latency grows 3–5× without governance

New failure mode

Active sequence count & KV eviction — not bandwidth — now drive tail latency

SLO violations are costly

Fraud scoring & payments face regulatory risk and financial loss from p99 breaches

THE SOLUTION

3-Layer Inference Governance Architecture

01

GOVERNANCE LAYER 1

Admission Control

Enforces QPS caps, concurrency limits, and context-length thresholds to keep GPU utilization below the nonlinear instability zone (u_cap = 0.85)

02

GOVERNANCE LAYER 2

KV Cache Isolation

Per-tenant memory quotas with namespaced eviction — Tier-0 (fraud/payments) KV blocks are never evicted to serve lower-priority batch workloads

03

GOVERNANCE LAYER 3

Priority Scheduling

Deadline-aware queue ordering replaces FIFO batching; fraud-scoring requests preempt long-context jobs at decode-step level to meet SLO deadlines

HOW GOVERNANCE HELPS

60–70%	99.2%+	90%+	3 layers
p99 latency reduction	SLO compliance for fraud scoring	GPU utilization maintained	integrated control plane in vLLM

THE BLACKWELL TAIL LATENCY PARADOX

H100 Era

Memory bandwidth is the dominant bottleneck

GPU saturates early — natural concurrency cap

p99 stays close to p50 — SLOs predictable

Latency range: 100–600 ms (bounded)

Bandwidth acts as a safety governor

Blackwell / NVFP4 Era

Bandwidth freed — concurrency scales 2×+

p50 improves 30–40% — more throughput

p99 grows 3–5× — SLOs violated at rho > 0.90

Latency range: 250–1,750 ms+ (unbounded)

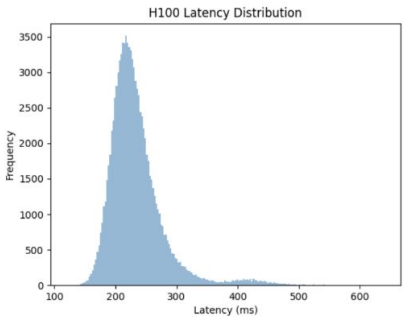
More capacity does not mean safer workloads

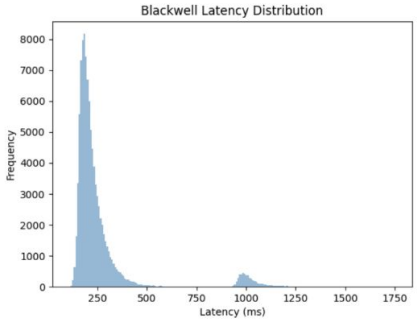
Without governance: fraud-scoring SLO breach rate is 15–25% at 90%+ GPU utilisation.

When bandwidth pressure recedes, interference becomes the new bottleneck

Simulation evidence: *Show that blackwell improves p50 but worsens p99*

https://github.com/akkhil2012/PyTorchParis2026/blob/main/BlackWellH100Compare.py





Metric	H100 Regime	High-Concurrency Regime
Memory Bandwidth Utilization	High correlation	Low correlation
Active Sequence Count	Low correlation	High correlation
KV Eviction Rate	Low correlation	High correlation
Scheduler Wait Time	Moderate	Strong

llm-compressor

Solving the Blackwell p99 Latency Problem

The Problem: Blackwell NVFP4 doubles throughput — but pushes concurrency so high that KV cache evictions explode, making **p99 latency 3–5× worse** even as p50 improves.

How llm-compressor fixes it — 3 recipes, one API call:

1 Shrink model weights

WHAT

Quantize weights to 4-bit (W4A16)

WHY IT HELPS

Frees 2–4× VRAM → bigger KV cache pool → fewer evictions

```
GPTQModifier(scheme="W4A16")
```

2 Shrink KV cache blocks

WHAT

Quantize KV cache tensors to FP8

WHY IT HELPS

Each cached block is 50% smaller → 2× more sequences fit before eviction

```
kv_cache_scheme: {num_bits: 8, type: float}
```

3 Faster token generation

WHAT

FP8 dynamic quantization on H100/Blackwell

WHY IT HELPS

~2× decode speed → sequences finish faster → active count stays low

```
QuantizationModifier(
  scheme="FP8_DYNAMIC")
```

`Pick a recipe` → `oneshot(model, recipe)` → `save_pretrained()` → `Load in vLLM` → `Lower p99` ✓

THREE-LAYER INFERENCE GOVERNANCE ARCHITECTURE

01

Admission Control

Upstream of scheduler

Caps effective utilisation at rho <= 0.85 before the queue enters the non-linear instability zone. QPS limits, concurrency caps and context-length gates applied per tenant.

Prevents unbounded queue growth

02

KV Cache Isolation

Inside inference engine (vLLM BlockSpaceManager)

Per-tier KV memory quotas (Tier-0: fraud/payments, Tier-1: chat, BE: batch). Eviction is local to each tier — a batch job can never evict a fraud-scoring KV block.

Eliminates cross-tenant spikes

03

Priority Scheduling

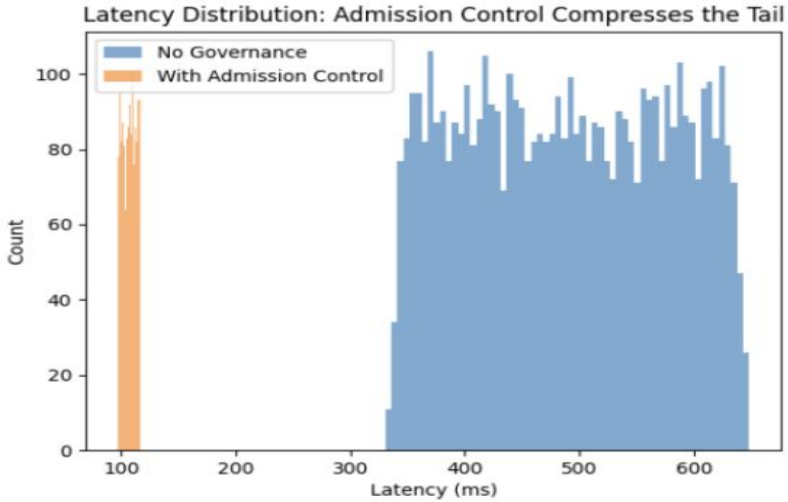
Decoder loop — vLLM step()

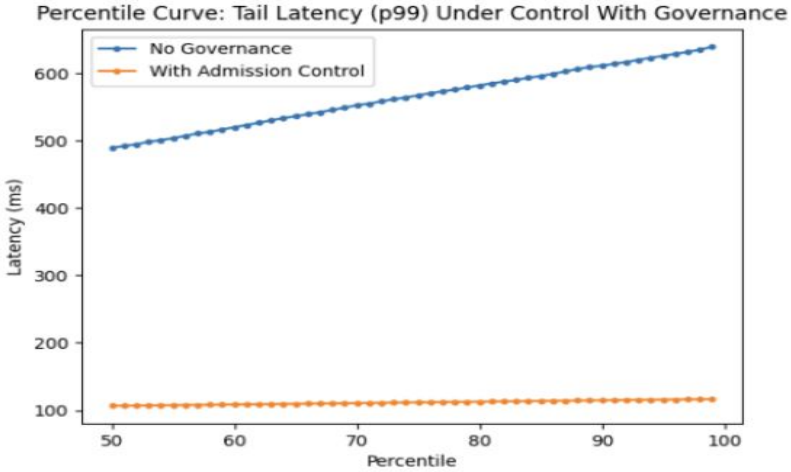
Deadline-aware queue replaces default FIFO. Tier-0 requests preempt lower-priority sequences at the decode-step level, ensuring SLO-critical workloads always advance first.

SLO compliance >= 99.2%

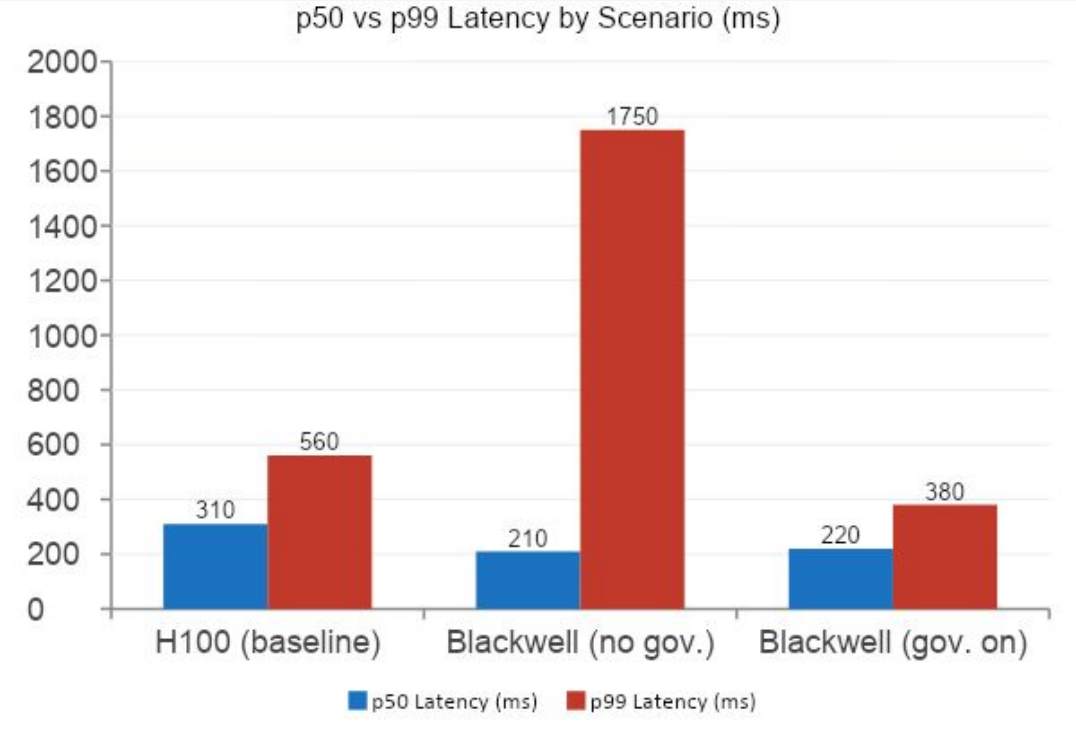
Integrates with vLLM >= 0.4 and PyTorch >= 2.3 on Blackwell GB200 / B200 clusters.

https://github.com/akkhil2012/PyTorchParis2026/blob/main/SimulatorGov.py





SIMULATION RESULTS (5,000 REQUESTS PER EXPERIMENT)



KEY FINDINGS

✔ Blackwell p50 improves 30–40% vs H100 — throughput gains are real.

✗ Blackwell p99 grows 3–5x without governance (560ms to 1,750ms+).

✔ Governance compresses p99 by 60–70% at rho = 0.93 offered load.

✔ SLO compliance 99.2%+ with governance vs. 75–85% without.

→ Trade-off: ~13% throughput reduction for deterministic SLOs.

Simulation validated. Live Blackwell GB200 hardware validation is the immediate next milestone.