Sparing Strategies to Minimize Reliability Impact on Large Training Jobs

An Analytical Framework for LLM Training Cluster Optimization

Kevin J. Quirk, Matthew Lennie, **Ehsan K. Ardestani**, et al. — Meta Platforms

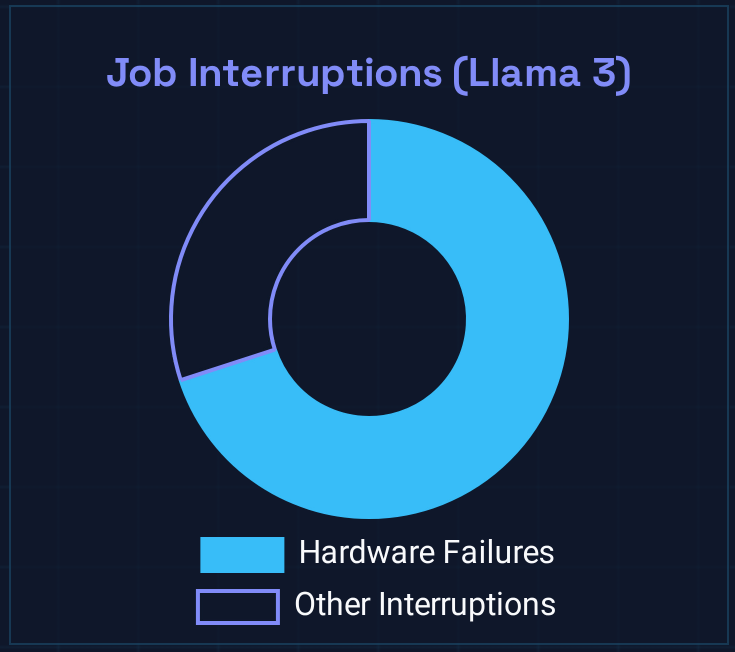
MLSYS 2026, BELLEVUE, WA

LLM Training at Hyperscale Demands Fault Tolerance

Training frontier LLMs requires massive, long-running distributed jobs that are highly vulnerable to hardware failures.

- **Llama 3** trained on 16K H100 GPUs over 15 trillion tokens
- **Llama 4 Behemoth** scaled to 32K H100 GPUs and 30 trillion tokens
- Hardware failures caused **>70% of job interruptions** during Llama 3 pre-training
- Synchronous training means **one failure can halt the entire job**

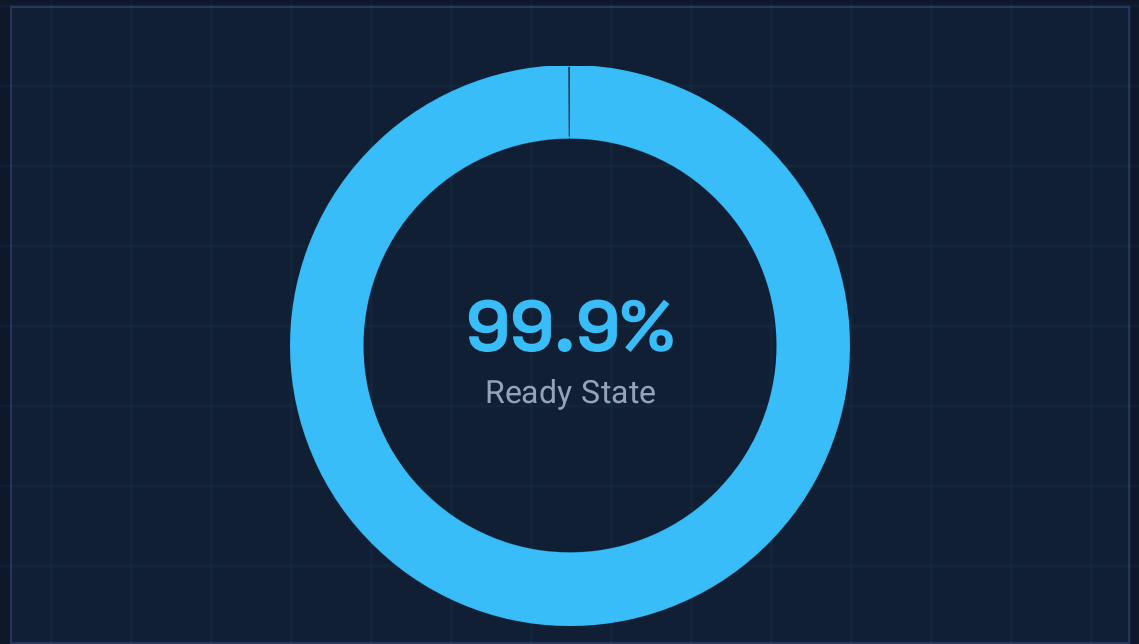
> *As cluster sizes grow, the frequency and complexity of failures rise — making fault tolerance a first-class design concern.*



Availability vs. Reliability: Two Pillars of Fault Tolerance

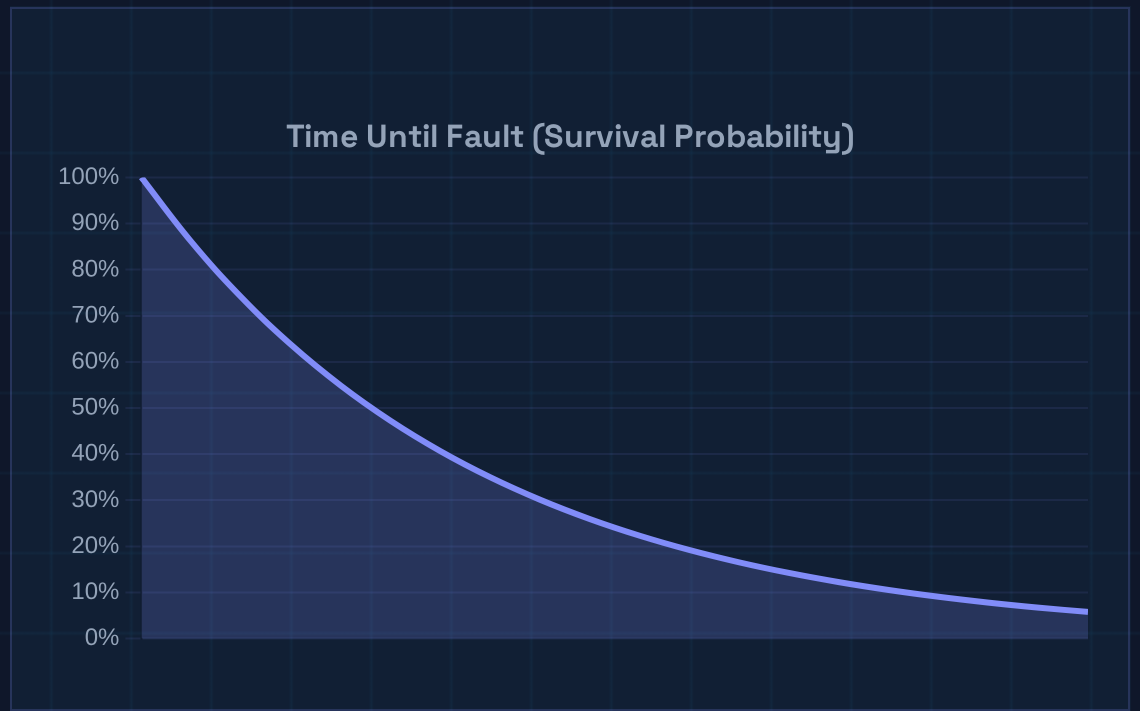
Availability

A measure of the fraction of a resource that is **ready to enter a working state**. It represents the proportion of the cluster that can be allocated to a job at any given time.



Reliability

A measure of **how long that resource can do work** (i.e., until an error/fault) after entering a working state. It dictates the frequency of job interruptions.



Addressing Failures: Sparing & Checkpoint Recovery

Two complementary mechanisms that protect Availability and Reliability

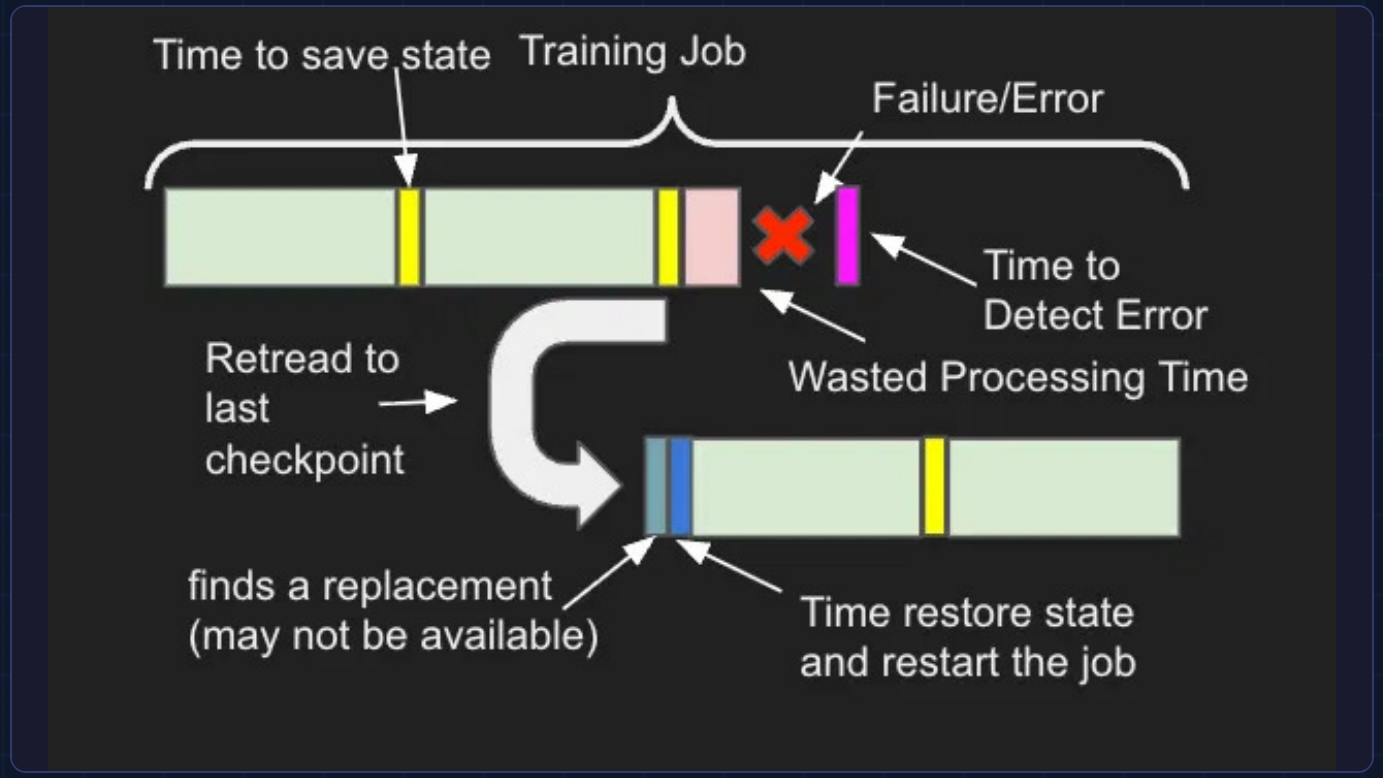
Sparing

Pre-allocates a fraction of resources to be used in the event of failures — increasing **availability** by ensuring a replacement is ready before a fault occurs.

Checkpoint Fault Recovery

Mitigates the impact of failures by periodically saving job state. When a fault occurs, the job **retreads to the last checkpoint** — trading checkpoint overhead for reduced wasted work.

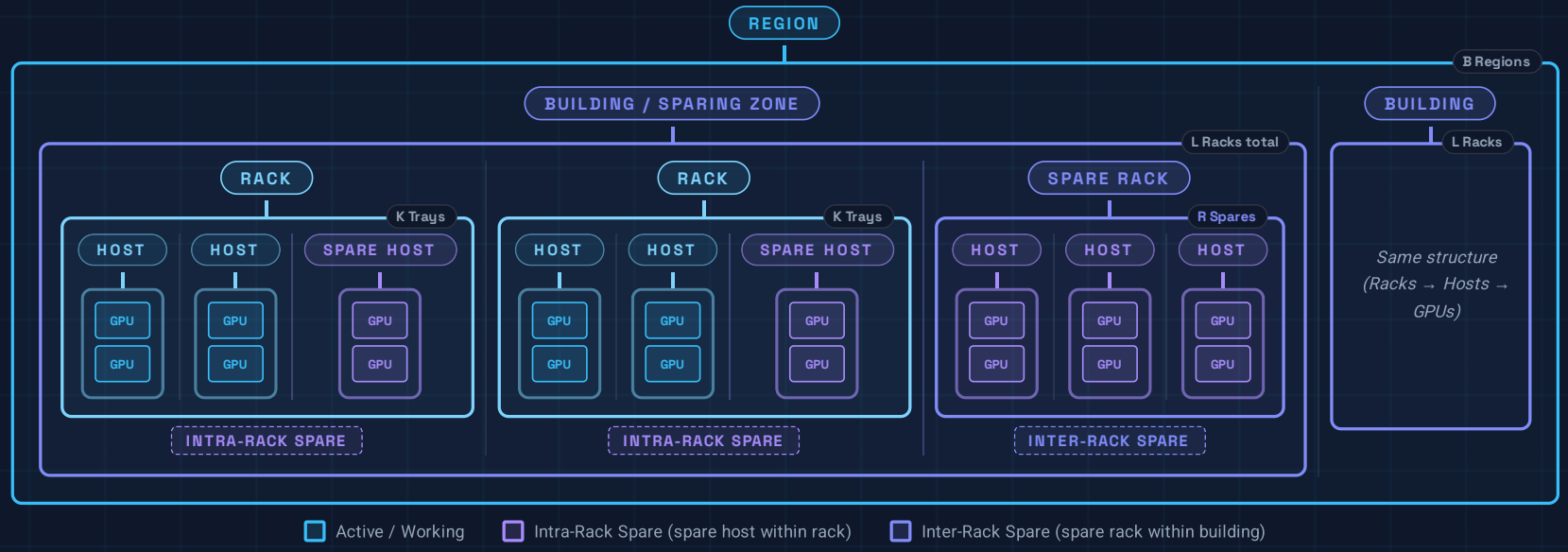
CHECKPOINT RECOVERY TIMELINE



A failure triggers a retread to the last checkpoint. Wasted time includes: time since last checkpoint, time to detect the error, and time to restore state and restart the job.

Sparing: Pre-Allocating Reserves to Absorb Failures

> **Sparing** — pre-allocating idle compute resources that can instantly replace failed components, keeping training jobs running without interruption.



The Core Trade-off: More spares = less job blocking time, but more idle (wasted) resources.

Goodput: The True Measure of Cluster Efficiency

Goodput captures the productive output of the entire cluster:

$$\text{Goodput} = \text{Cluster Size} \times \text{CETT} \times \text{TPS Scale(Hardware)} \times \text{TPS Scale(LLM)}$$

CETT

Cluster Effective Training Time: The fraction of total GPU time actually spent on productive work.

TPS Scale(Hardware)

Accounts for performance gains from power-limit redistributions due to sparing decisions.

TPS Scale(LLM)

Accounts for parallelism efficiency and scale-up network utilization under different block configurations.

CETT is reduced by: idle spare GPUs, job-blocking time, checkpoint overhead, and wasted work after failures.

Analytical Framework: Markov Chains Model Failure Dynamics

The framework uses **probability theory and continuous-time Markov chains** to model stochastic failure and repair events — moving beyond simple linear rate assumptions.

Spare Tray Model 01

Derives compute block MTBF accounting for tray-level and rack-level failures and intra-block spares.

Spare Block Model 02

Computes probability of job blocking when all inter-block spares are exhausted.

Checkpoint Fault Recovery Model 03

Quantifies wasted time from checkpointing, fault detection, and job restart.

Closed-form expressions enable fast, first-order design decisions without simulation overhead.

Job Placement Constraints Create Stranded Resources

Parallelism strategies (tensor, data, expert parallel) impose **co-location constraints** that restrict which GPUs can be used together.

- Tensor parallelism must stay within a single compute block (scale-up network)
- Data parallel groups may need to be co-located within a sparing zone (scale-out network)
- Placement constraints force the number of working blocks per zone to be a multiple of group size P

$$L - R = k \cdot P \quad , k \in \mathbb{Z}^{+}$$

Stranded Blocks

Non-working blocks beyond the minimum needed for sparing — they are wasted capacity even though they could be opportunistically reused.

Production Configuration: Meta's Cluster Parameters

ARCHITECTURE & RELIABILITY

Sparing Zones (B)	4
Compute Blocks per Zone (L)	256 racks
GPUs per Rack	72 (2 per tray, 36 trays)
Tray MTBF	20,000 hours
Rack-level MTBF	10,000 hours
Mean Time to Repair (MTTR)	24 hours

FAULT RECOVERY

Checkpoint Period	250 seconds
Checkpoint Save Time	50 ms
Fault Detection Time	60 seconds
Job Restart Time	6 minutes

Evaluated Strategies: Block sizes of 72, 36, and 18 GPUs — with and without intra-block spares.

72-GPU Block with Intra-Block Spares Maximizes Goodput

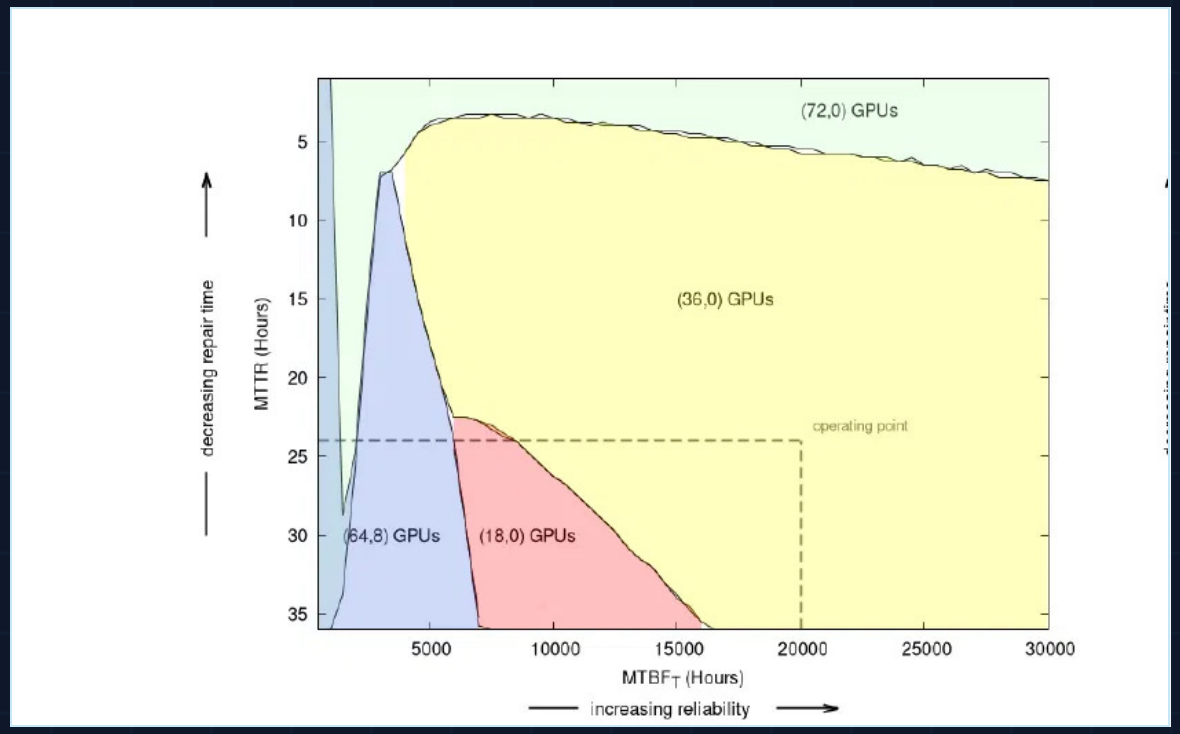
CONFIG	WORKING GPUS	INTRA SPARES	INTER SPARES	CETT	TPS (HW)	TPS (LLM)	GOODPUT (GPUS)
72 GPUs, no intra	72	0%	8.6%	68.8%	1.000	1.18x	59,821
72 GPUs, 8 intra	**64**	**11.1%**	**1.4%**	**68.5%**	**1.034**	**1.17x**	**61,134**
36 GPUs, no intra	36	0%	4.7%	68.0%	1.000	1.12x	56,110
36 GPUs, 4 intra	32	11.1%	1.0%	67.8%	1.034	1.11x	57,330
18 GPUs, no intra	18	0%	2.6%	66.4%	1.000	1.02x	49,912
18 GPUs, 2 intra	16	11.1%	0.9%	66.0%	1.034	1.00x	50,307

KEY INSIGHT

72-GPU block with 8 intra-block spares achieves **1.024x more goodput** than the second-best strategy, driven by power redistribution and larger scale-up network gains.

Optimal Strategy Varies with Reliability & Repair Time

FIGURE 2 — CETT-ONLY OPERATING REGION MAP



Each colored region shows the sparing strategy (block size, intra-block spares) that maximizes CETT for a given MTBF$_T$ (x-axis, increasing reliability →) and MTTR (y-axis, decreasing repair time ↑). No model/hardware gains or placement constraints applied.

Scope: CETT only — this analysis optimizes the *Cluster Effective Training Time* component of Goodput in isolation.

⚠ **Goodput = Cluster Size × CETT** × ~~TPS Scale(HW)~~ × ~~TPS Scale(LLM)~~
Hardware gains, model gains, and placement constraints are excluded here; adding them changes the optimal strategy (see Figures 3–5).

KEY TAKEAWAYS

(36,0) GPUs — **36-GPU blocks (no intra spares)** dominate the majority of the operating space — smaller blocks need proportionally fewer spares.

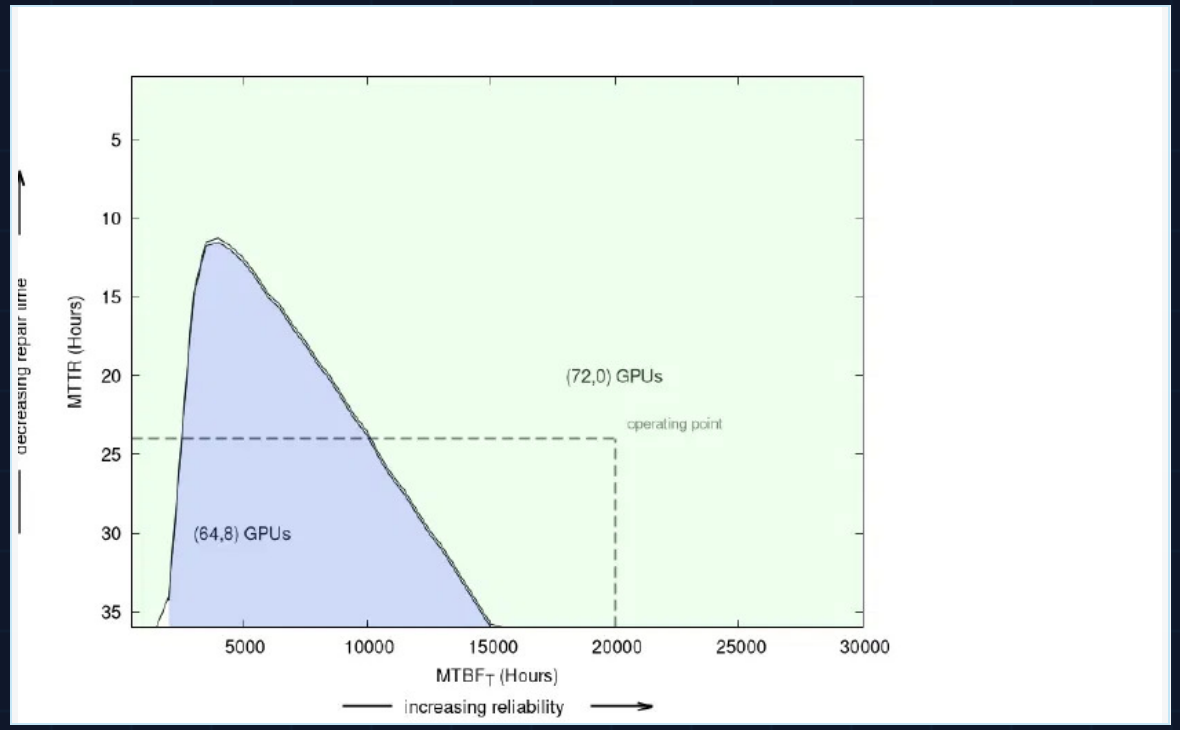
(72,0) GPUs — **72-GPU blocks (no intra spares)** become optimal at **very short repair times** — fast recovery reduces the cost of larger blocks.

(64,8) GPUs — **72-GPU blocks with 8 intra-block spares** win at **very low MTBF** — intra-block sparing absorbs frequent tray-level failures.

(18,0) GPUs — **18-GPU blocks** appear only under extreme failure rates — fine granularity helps but yields lower overall goodput without model gains.

Adding Model Gain: 72-GPU Block Dominates Everywhere

FIGURE 3 — WITH MODEL (LLM) GAIN, NO PLACEMENT CONSTRAINTS



Same axes as Figure 2 (MTBF$_T$ vs. MTTR). Incorporating the LLM model performance gain for larger block sizes collapses the multi-region map into just two regions, both at 72 GPUs per block.

⚠ **Scope: CETT + Model Gain** — hardware TDP gain and placement constraints still excluded.

Goodput = Cluster Size × **CETT** × ~~TPS Scale(HW)~~ × **TPS Scale(LLM)**

KEY TAKEAWAYS

↑ **Dramatic shift from Figure 2:** smaller blocks (36, 18 GPUs) disappear entirely — model gain strongly favors larger block sizes.

(72,0) GPUs

72-GPU blocks (no intra spares) dominate under **higher reliability and shorter repair times** — fast recovery makes intra-block sparing unnecessary.

(64,8) GPUs

72-GPU blocks with 8 intra-block spares win under **lower reliability and longer repair times** — intra spares absorb frequent tray-level failures before they block the job.

- **Meta's operating point** (MTBF$_T$ ≈ 20,000 h, MTTR ≈ 25 h) now falls in the **(72,0) region** — hardware gain and placement constraints will further refine this choice.

Reliability & Repair Sensitivity Shapes Optimal Strategy

The best sparing strategy shifts depending on MTBF and MTTR:

- **CETT-only (no model/hardware gains):** Smaller block sizes (36, 18 GPUs) dominate most of the MTBF–MTTR space due to lower sparing requirements.
- **Adding hardware power gain:** Intra-block sparing becomes advantageous across a wider operating region.
- **Adding LLM model gain + placement constraints:** Larger block sizes (72 GPUs) dominate — the performance gain from a larger scale-up domain outweighs sparing savings from smaller blocks.
- **Key tension:** Halving block size roughly halves inter-block spares needed, but placement constraints can strand an equal amount of capacity — net gain is zero.

> ***"No single sparing strategy is universally optimal — the right choice depends on the full system context."***

Simulation Validates Analytical Model with <1% Error

A Monte Carlo end-to-end simulator (**Arcadia**) cross-validates the analytical framework:

- Simulator matches analytical results under identical assumptions with **<1% relative error in CETT**
- Validated for the composite repair process assumption (auto-remediated vs. ticket-based repairs)
- Simulation extends analysis to complex scenarios: job queue scheduling, mixed fault recovery modes, and fine-grained failure interactions

<1%

RELATIVE ERROR IN CETT

Composite repair simplification holds:

The error from using a single composite MTTR is not sensitive to the proportional mix of auto-remediated vs. ticket-based failures — validating the analytical model's practical applicability.

Framework is Now a Cornerstone of Meta’s AI Infrastructure Design

IMPACT IN PRODUCTION

- Used to guide architectural decisions for Meta's hyperscale LLM training clusters
- Enables rapid, data-driven trade-off analysis at early design stages
- Balances reliability, efficiency, and cost across cluster generations

KEY TAKEAWAYS

1. **Goodput optimization** requires jointly considering CETT, hardware TPS, and LLM TPS — not just availability.
2. **Intra-block sparing** unlocks power redistribution gains that can outweigh its overhead cost.
3. **Placement constraints** can negate sparing savings — stranding must be accounted for.
4. **Markov chain closed-form models** are practical and accurate for first-order cluster design decisions.
5. **Simulation** provides a robust validation layer for complex, dynamic production scenarios.

Thank You

✉ ehsanardestani@meta.com

AUTHORS

- Kevin J. Quirk
- Matthew R. Bergeron
- Xu Zhang
- Chunqiang Tang
- Matthew Lennie
- Andrew Grier
- Abhinav Triguna
- **Ehsan K. Ardestani**
- Zhaodong Wang
- Ying Zhang
- Satyajeet Singh Ahuja
- Mustafa Ozdal
- Mathew Oldham

Meta Platforms, Inc. · Proceedings of the 9th MLSys Conference, Bellevue, WA, 2026


Meta