OpenTela: Unifying Decentralized Computing Resources for LLM Serving

(Operational System)

Xiaozhe Yao, Youhe Jiang, Ilia Badanin, Qinghao Hu, Robert Matthew Smith, Binhang Yuan, Imanol Schlag, Eiko Yoneki, Ana Klimovic.

July 2026

The Sovereignty Gap in Today's LLMs

Claude OpenAI ChatGPT Gemini

THE PROBLEM

Lack of transparency

How they're trained and served stays hidden.

Lack of control

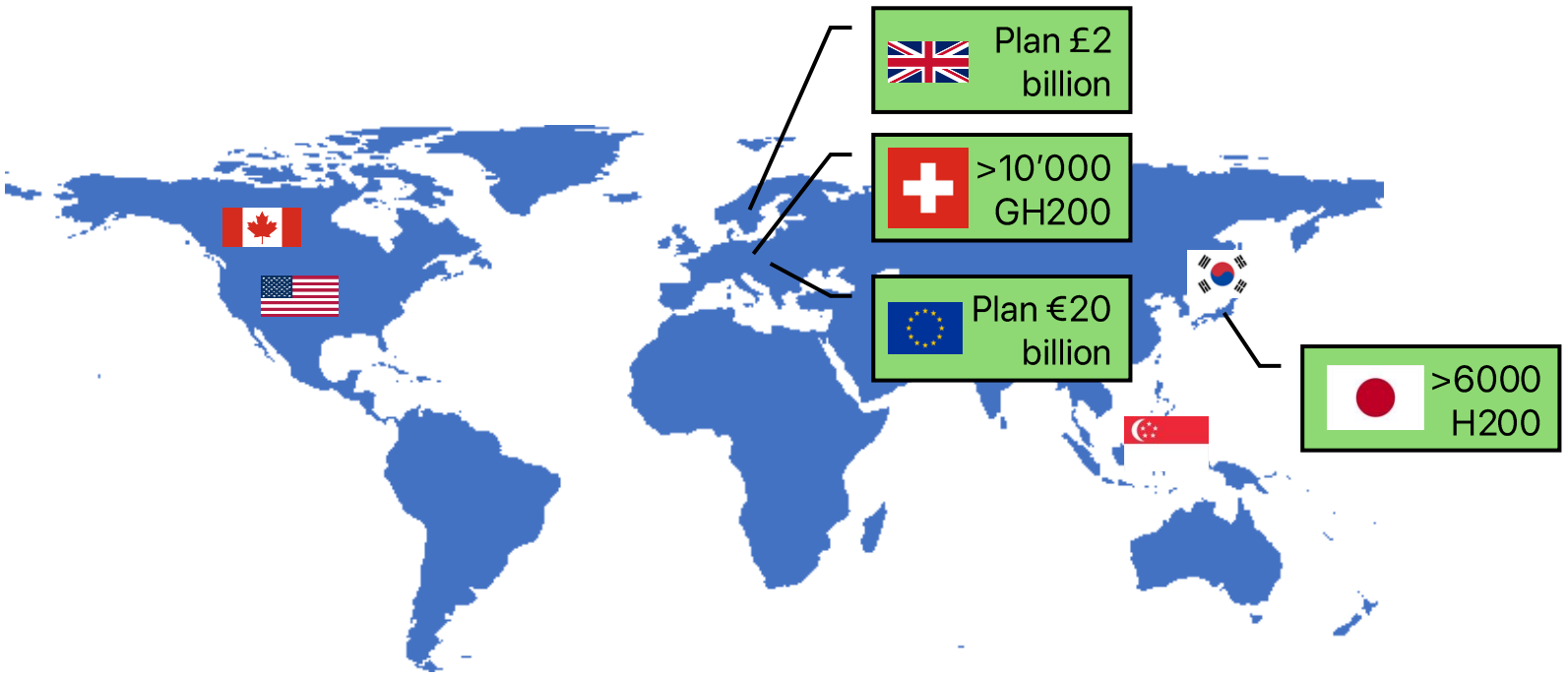
Routing, precision, and data retention are out of our hands.

Lack of access

Best models gated by cost, geography, and policy.

We need *sovereign* AI.

Significant investment in AI Factories





Carsten Binnig

Professor @ TU Darmstadt

*"Access to these models must be treated as critical scientific infrastructure ...what we need are **massive resources for AI inference**"*

McKinsey & Company

*"**Inference workloads** could make up 40% of data-center demand by 2030."*

Compute is being funded at unprecedented scale — and we are seeing massive demand on inference.

Towards Sovereign AI Inference

We have the key ingredients:

Open models

Capable open-weight & fully-open models

Serving engines

Efficient and powerful systems

vLLM SGL

Sovereign compute

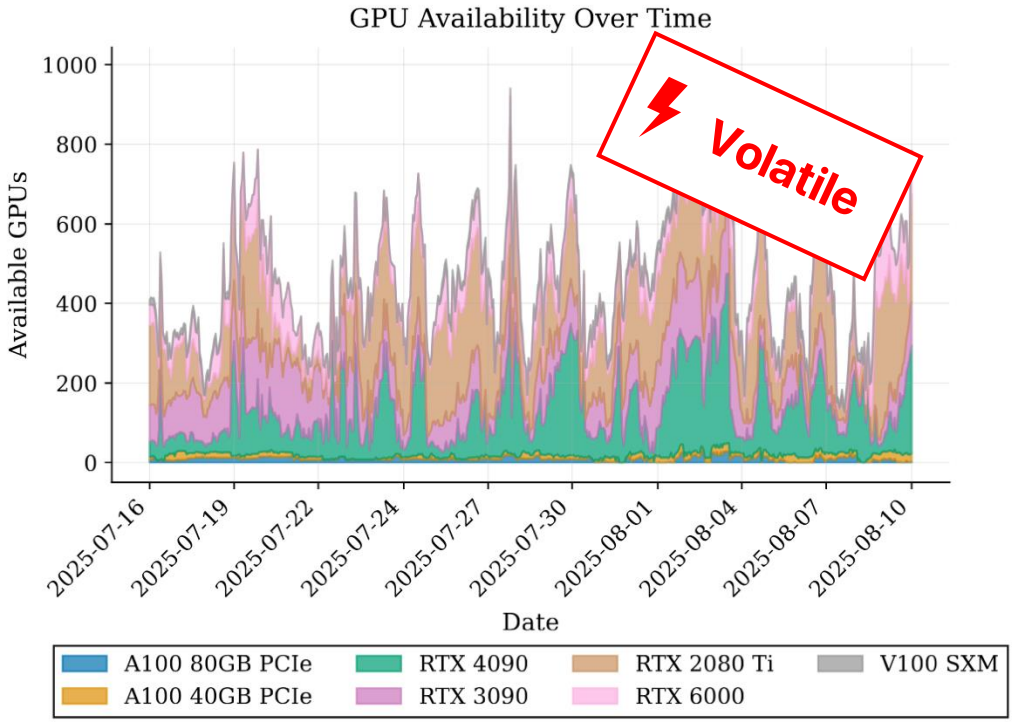
Nations funding AI factories at scale

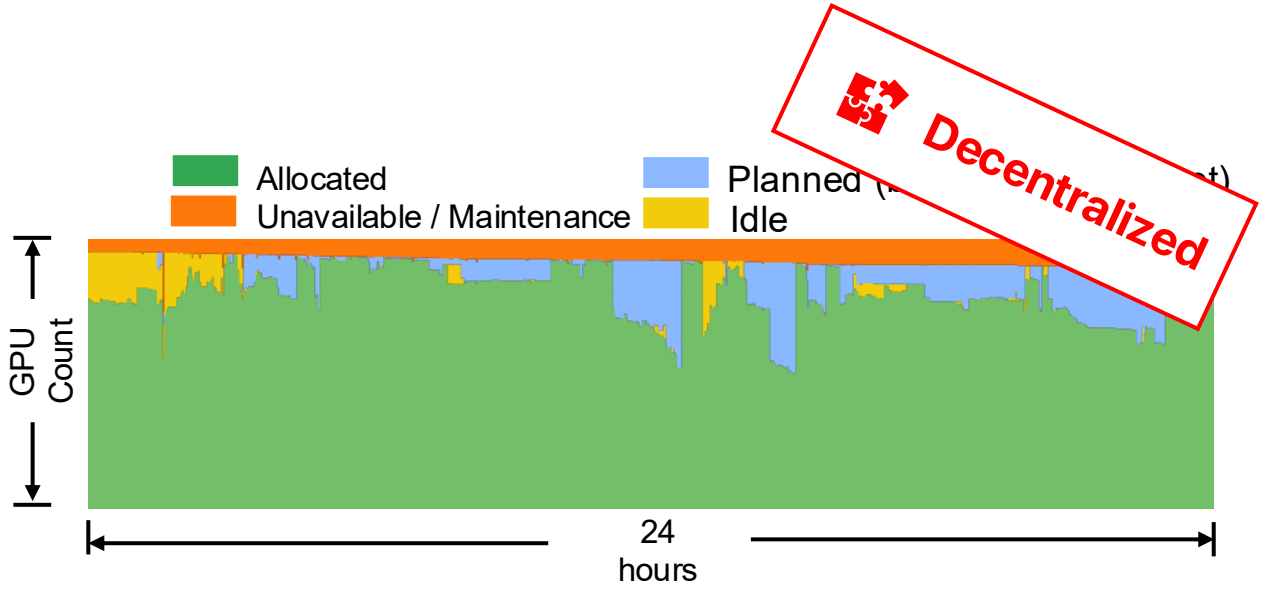


The compute exists — so why can't we use it for sovereign inference?

GPU Compute Availability

Volatile, heterogeneous, decentralized compute resources





Can we harness the volatile, decentralized computing resources, and build a platform for AI inference?

Towards Sovereign AI Inference on HPC Clusters

- We do have the compute resources! However,…
  - **Fragmented across different clusters** managed by incompatible orchestration systems: Kubernetes, Slurm, …
  - >65% of HPCs are using Slurm, but Slurm is **not well-built for serving** workloads: no service discovery, load balancing etc.

- **OpenTela**: **Federate** siloed, fragmented, heterogeneous computing resources into **a unified serving platform**.

Requirements to Unify Decentralized Compute

For Sovereign AI Inference

1 Non-Invasive Deployment



Runs entirely in user space — no root, kernel modules, or admin privileges.

2 Stable Abstraction



Stable service endpoints over transient resources, surviving preemption.

3 Coordinator-Free Federation



Federates across sites with no centralized coordinator or single point of failure.

4 Requirement-Aware Routing

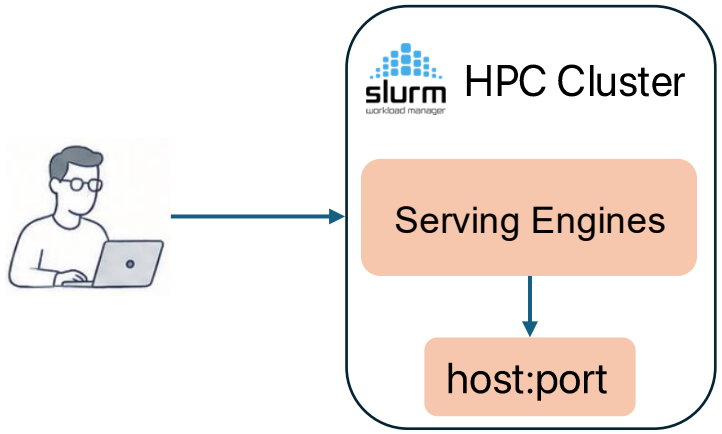


Routes across heterogeneous hardware using heterogeneity- and trust-aware decisions.

OpenTela: A Thin Layer

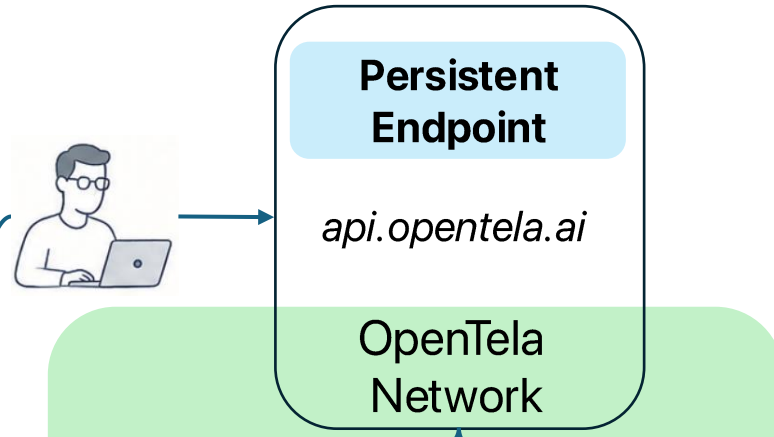
Unifying GPU Resources together for AI Inference

Without OpenTela



Serving Engines ≠ Serving Platform!

With OpenTela



Persistent Endpoint

As long as at least one model replica is online

Access Anywhere

Accessible from anywhere on the internet

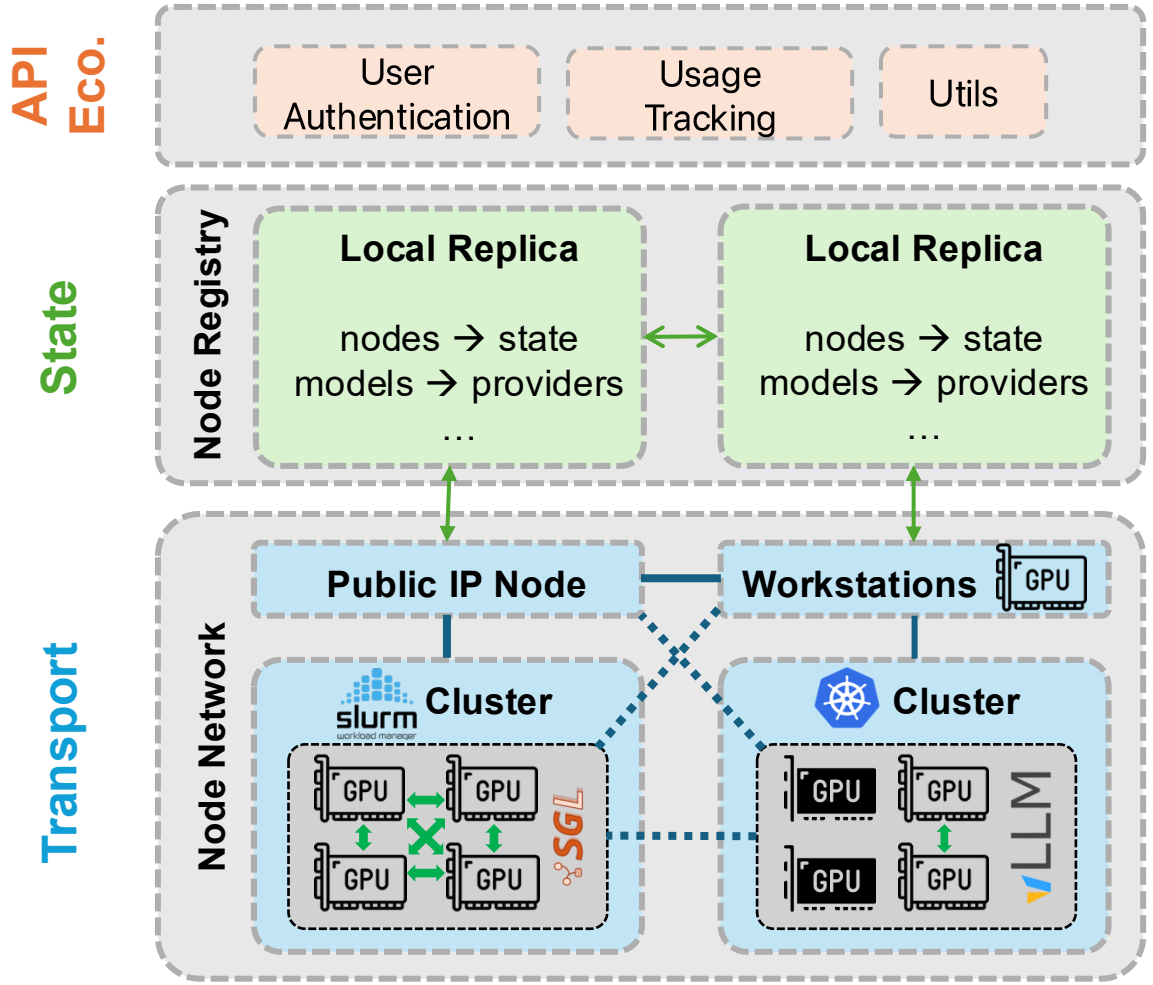
Cross-cluster sharing

Access models across geo-distributed clusters

OpenTela: not a new serving engine, nor a cluster manager
It is a thin coordination and networking layer, making existing serving engines **discoverable**, **persistent** and **reachable**.

OpenTela: System Architecture

Components to support user-space orchestration



Ease of Access
Standard APIs for auth, usage tracking, and shared utilities

CRDT-based Global View
Nodes traverse from Join → Serve → Down → Left

No Central Coordinator Required
Each node maintains a global view of the whole system

Direct / Relay
Hole punching for direct channels or circuit relay

No Network Change
Use outbound network only

Battle-Tested System

From lab experiments to battle-tested system and real deployments

20+ Months



in continuous operation

1'000+ Researchers



from ETH/EPFL, as well as federal tribunal and other partners

142+ Models



ranging from hundreds millions to ~1T parameters

15B+ Tokens



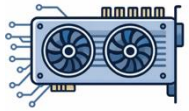
from ~13 million requests

Peak 2B+ Tokens



in a single day

200+ GPUs managed



of varying types

View it live: *https://serving.swissai.cscs.ch/*

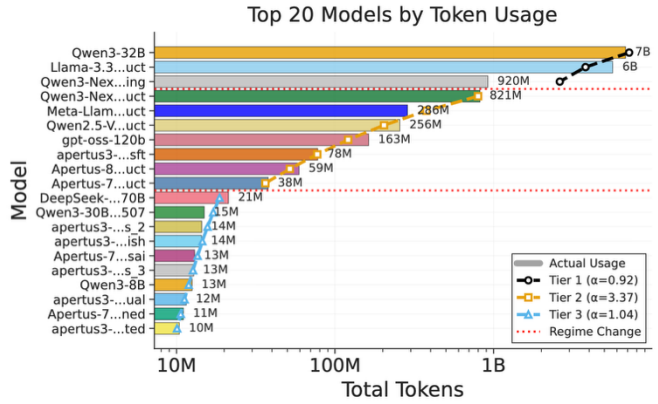


Collected from July 2024 to October 2025

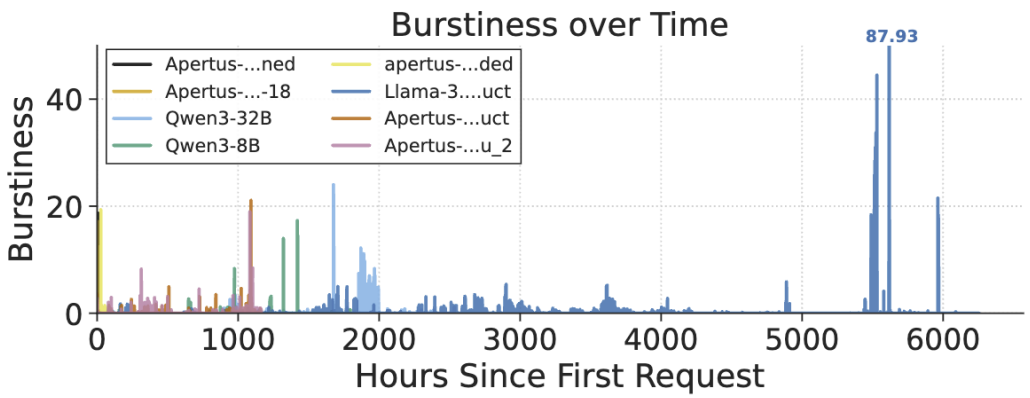
Open Traces

We released our production traces openly — here's what they reveal

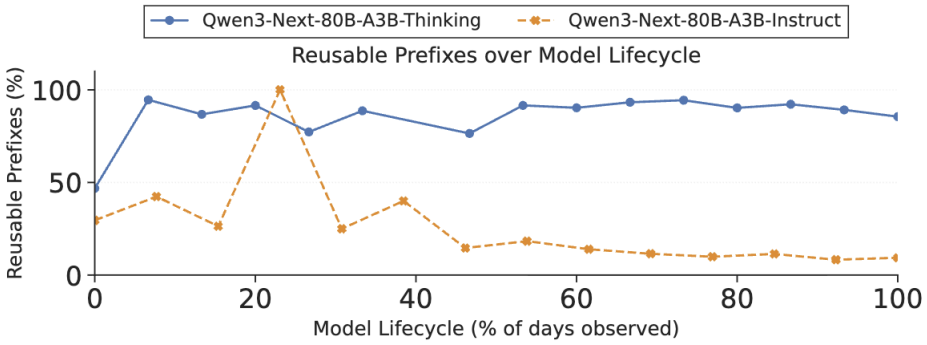
Overall model token usage



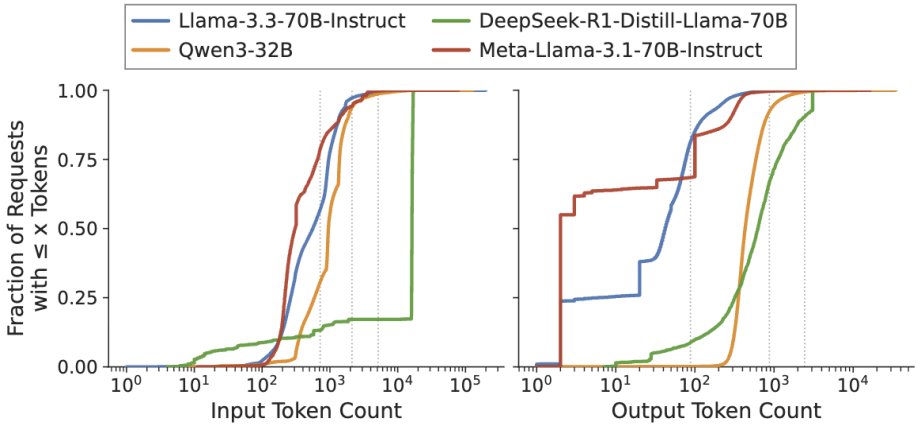
Burstiness over Time



Reusable prefix ratio



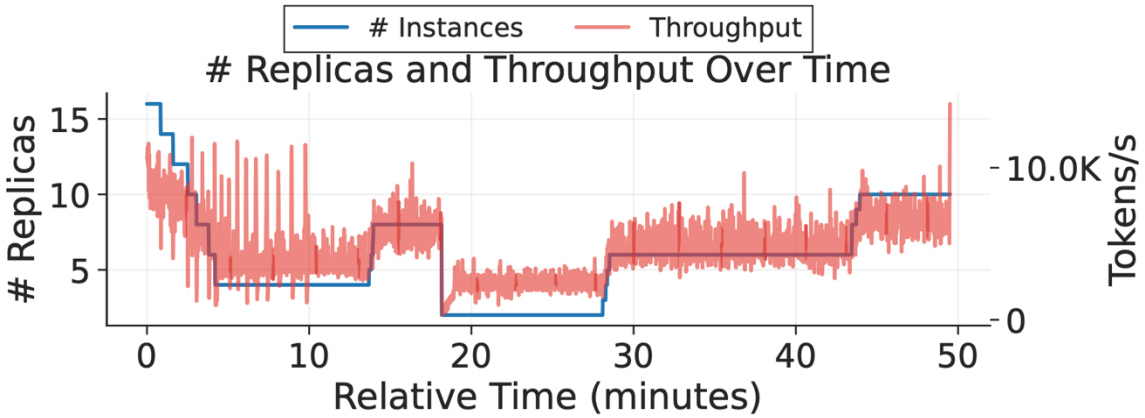
Input/Output Token Distribution



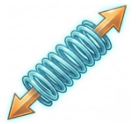
OpenTela: Elasticity and Fault Tolerance

The ability to leverage volatile resources

For this experiment, we deploy OpenTela to up to 64 GH200 GPUs serving a continuous request stream, and randomly adjust the available GPUs.



Number of active replica vs. achieved throughput.



Elasticity: Throughput scales proportionally to the number of active replicas → effective use of available capacity.



Fault Tolerance: Zero user-facing HTTP errors during the entire experiment. OpenTela successfully routed requests to healthy nodes, ensuring service continuity.

Read our paper for more trace analysis and evaluation ☺

Lessons Learnt

from building and operating in decentralized environments

1

Flexibility & Interoperability

Clean rebuilds cost too much — **interoperate across existing serving engines / orchestration systems.**

2

Expressive Abstraction

OpenAI-compatible APIs hide quantization and model configs — **expose them with a richer abstraction.**

3

Introspection & Oversight

Importance to detect node failures early — **remove nodes from network and debug offline**

swiss-ai/Apertus-1-70B-FP8-32K-Tool-SFT-RL-DPO-SDPO-Low-Less-Refuse-Feedback-Think

Summary

- **OpenTela**: A thin layer between current HPC environment and LLM serving engines, that enables heterogeneous and decentralized serving.
- OpenTela makes existing serving engines **discoverable**, **persistent** and **reachable,** on HPC and other volatile computing resources.
- We have been operating the system for ~20 months. We open source the system, the traces we collected, and all the utilities we built.

 eth-easl/swissai-serving-trace  eth-easl/OpenTela  opentela.ai

- **All together**: Make LLM serving ***cost-efficient***, ***usable*** on existing geo-distributed HPC clusters, AI factories and fragmented, heterogenous hardware/software resources.
- **Future**: More sites to deploy, more heterogenous hardware, and broaden access to more researchers around the world.