AI Factory Benchmarking Framework

Automated Performance Analysis for HPC-Based AI Services

Mario Capodanno Giuseppe Galardi Can Beydogan Thies Weel

January 2026

EUMASTER4HPC 2025–2026
Supervisor: Dr. Farouk Mansouri, LuxProvide

Outline

Introduction

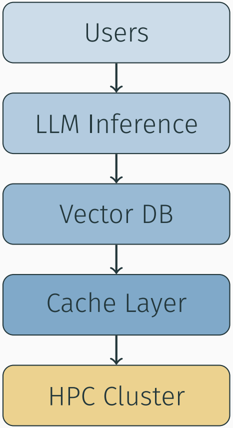
The Challenge: AI Factories on HPC

Modern AI Infrastructure

- LLM inference at scale
- Vector databases for RAG
- High-throughput caching
- GPU-accelerated workloads

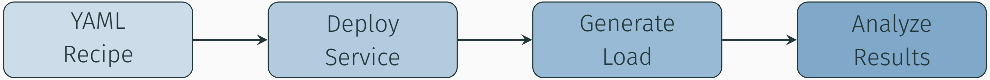
Key Questions

- Max sustainable concurrency?
- Where are the bottlenecks?
- How do configs affect performance?



Our Solution

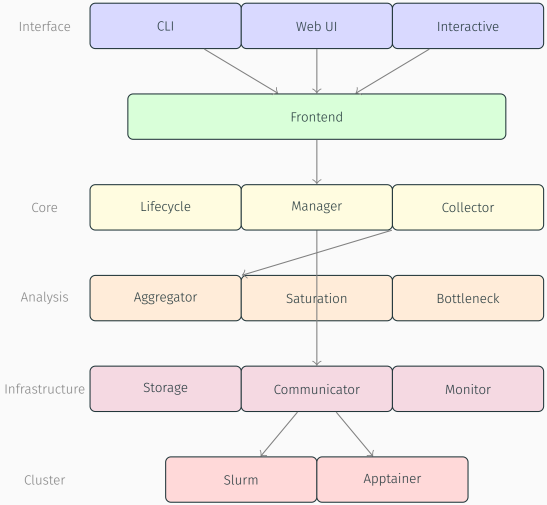
Automated Benchmarking Framework



- Recipe-driven
- Fully automated
- Reproducible

System Architecture

Component Overview



Supported Services

Category	Service	Use Case
Inference	vLLM	High-perf LLM serving
	Ollama	Lightweight LLM
Database	PostgreSQL	OLTP workloads
	Redis	Caching
	MinIO	Object storage
Vector DB	ChromaDB	Embedding search
	Qdrant	RAG pipelines

7 services **3 categories** **Automatic configuration**

Recipe-Driven Benchmarking

Recipe Example

recipe_vllm.yaml

```yaml
configuration:
  target: meluxina

service:
  type: vllm
  partition: gpu
  num_gpus: 1
  settings:
    model: facebook/opt-125m

client:
  type: vllm_smoke
  partition: cpu

benchmarks:
  num_clients: 4
```

Benefits

- Self-documenting
- Version-controllable
- Reproducible
- Schema-validated

One Command

```
python frontend.py recipe.yaml
```

Automated Analysis

Saturation Detection

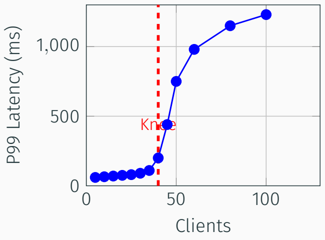
Maximum Curvature Method

Find the “knee” where latency grows superlinearly:

$$\kappa(x) = \frac{|y''(x)|}{(1 + y'(x)^2)^{3/2}}$$

Identifies:

- Latency knee point
- Throughput saturation
- SLO compliance limit



Bottleneck Attribution

Type	Indicators
GPU-bound	GPU util $>$ 80%, stable CPU, rising TTFT
CPU-bound	High CPU time, low GPU, stable memory
Memory-bound	High RSS, OOM errors, latency spikes
Queueing	Throughput plateau, P99 explosion

Each classification includes evidence and actionable recommendations

Real-Time Monitoring

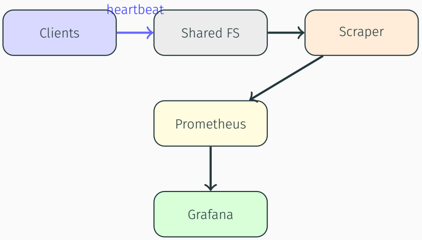
Prometheus + Grafana Integration

Monitoring Stack

- Prometheus for metrics collection
- Grafana for visualization
- Pre-configured dashboards

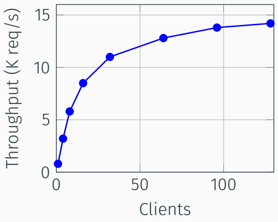
Heartbeat Strategy

- Filesystem-based client tracking
- Works across distributed nodes
- Uses shared Lustre filesystem



Experimental Results

Case Study: Redis Scaling



Findings

- Near-linear scaling to 64 clients
- Saturation at ∼96 clients
- Peak throughput: 14.2K req/s
- Bottleneck: CPU processing

Payload Sensitivity

- Small values (<1KB): linear
- Large values (>16KB): earlier saturation due to memory bandwidth

User Interface

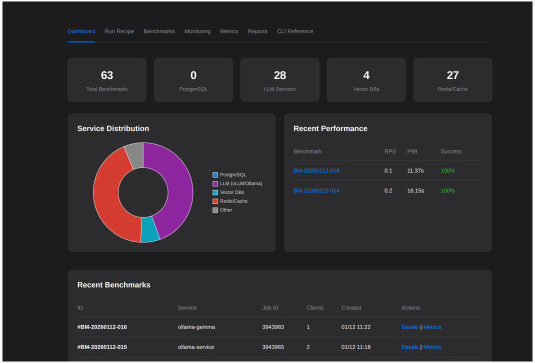
CLI Workflow

```
# Run benchmark
python src/frontend.py recipe.yaml

# Monitor
python src/frontend.py --watch BM-20260112-001

# Generate report
python src/frontend.py --report BM-20260112-001

# Compare runs
python src/frontend.py --compare BM-001 BM-002

# Launch web UI
python src/frontend.py --web
```

Web Interface

Features

- Dashboard
- Deploy benchmarks
- Real-time status
- Log browser
- Metrics visualization
- Report generation



Conclusion

Summary

Key Contributions

- Automated saturation detection
- Bottleneck attribution
- Real time monitoring

Target Platform

- Slurm job scheduling
- Apptainer containers
- GPU and CPU partitions

Future Work

- Multi-node client deployment for higher load
- ML-based anomaly detection
- Integration with PBS, LSF schedulers
- Power-aware efficiency metrics (tokens/watt)

Thank you!

Team1_EUMASTER4HPC2526

Backup: Artifact Structure

```
results/<benchmark_id>/
  run.json            # Complete metadata
  requests.jsonl      # Per-request timing
  summary.json        # Aggregated metrics
  logs/               # Service and client logs

reports/<benchmark_id>/
  report.md           # Human-readable analysis
  report.json         # Machine-readable summary
  plots/              # Visualization PNGs
```

Backup: Regression Detection

Metric	Threshold
P99 Latency increase	$> 10\%$
Throughput decrease	$> 10\%$
Success rate decrease	$> 1\%$
Error count increase	Any

Configurable thresholds for CI/CD integration