



SimServing: Native-Execution Simulation for Evolution-Resilient LLM Serving Configuration Tuning

Yujie Huang · Jiazhen Gu · Zhihan Jiang · Michael R. Lyu
The Chinese University of Hong Kong



➤ LLM Serving: The backbone of AI Applications

Applications



Chat Assistant



Coding Agent



Deep Research

LLM Serving Software



MindIE





LLM Serving: The backbone of AI Applications

Applications



Chat Assistant



Coding Agent



Deep Research

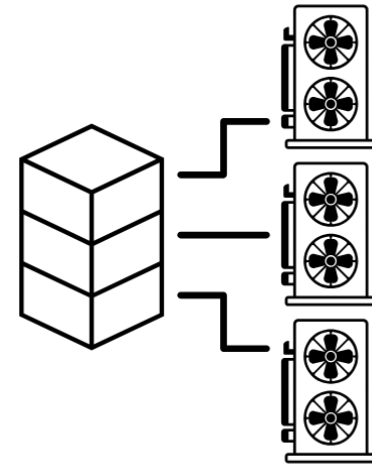
LLM Serving Software



MindIE



Hardware Cluster





LLM Serving: The backbone of AI Applications

LLM Serving Software



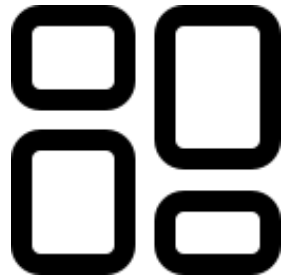
MindIE



- **Scheduler policy**
- **Parallelism policy**
- Max batch size
- Max batch tokens
- Max prefill batch size
- Max decode batch size
- Max context length
- Prefill time per req
- Concurrency
- Request rate
- ...



➤ Expensive Configuration Tuning



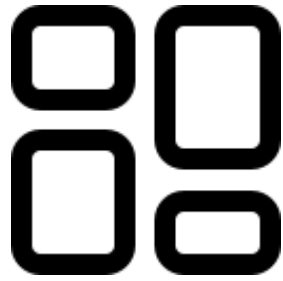
Large
Search Space



On-device
Benchmarking



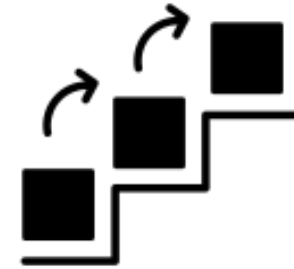
➤ Expensive Configuration Tuning



Large
Search Space



On-device
Benchmarking



Upstream
Evolution



LLM Serving System

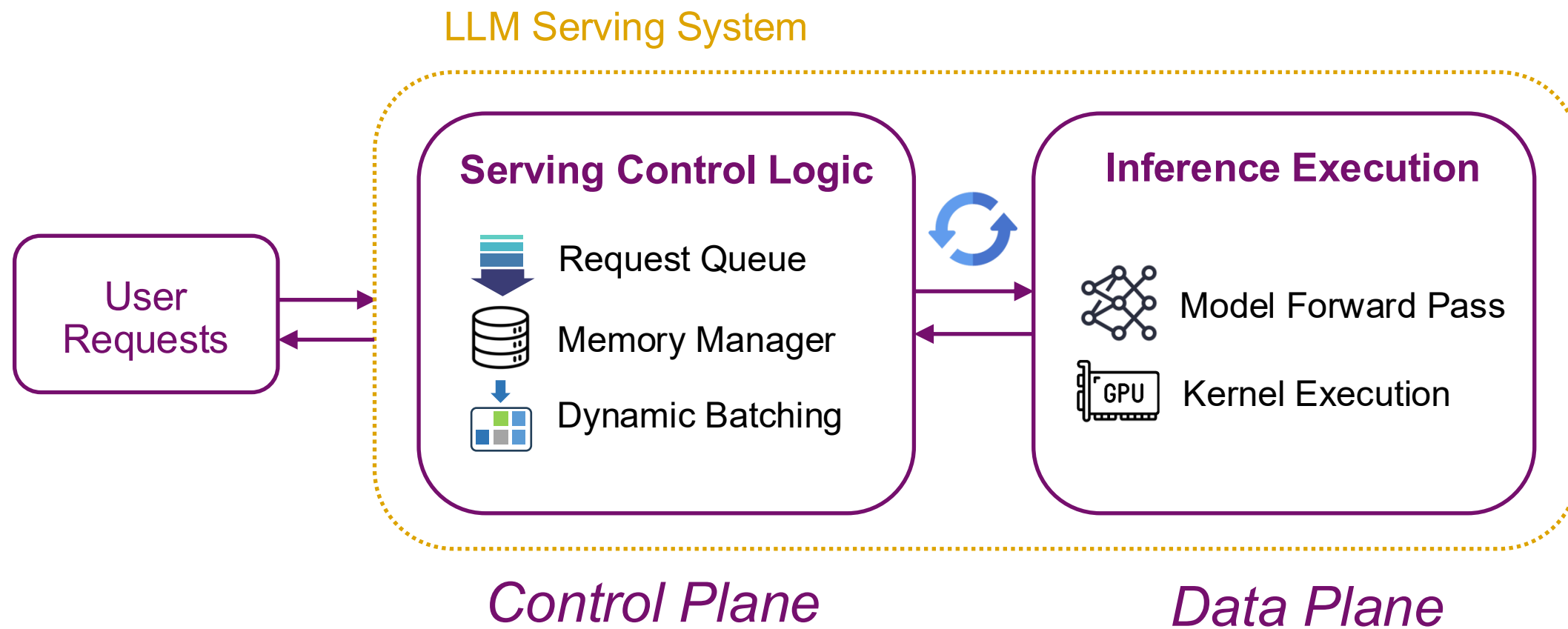


Native

Simulation



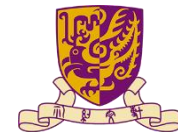
LLM Serving System



Native

Simulation

LLM Serving Simulator



Native

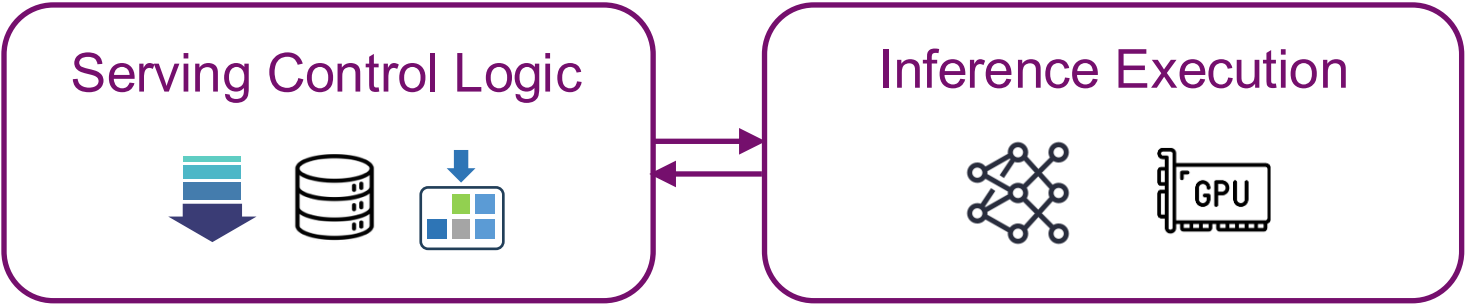
Simulation



LLM Serving Simulator

Control Plane

Data Plane



Native

Simulation



LLM Serving Simulator

Control Plane

Serving Control Logic



Data Plane

Inference Execution



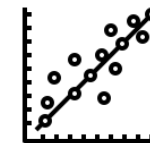
Native

Simulation

Simplified Re-implementation



Op-level Predictors

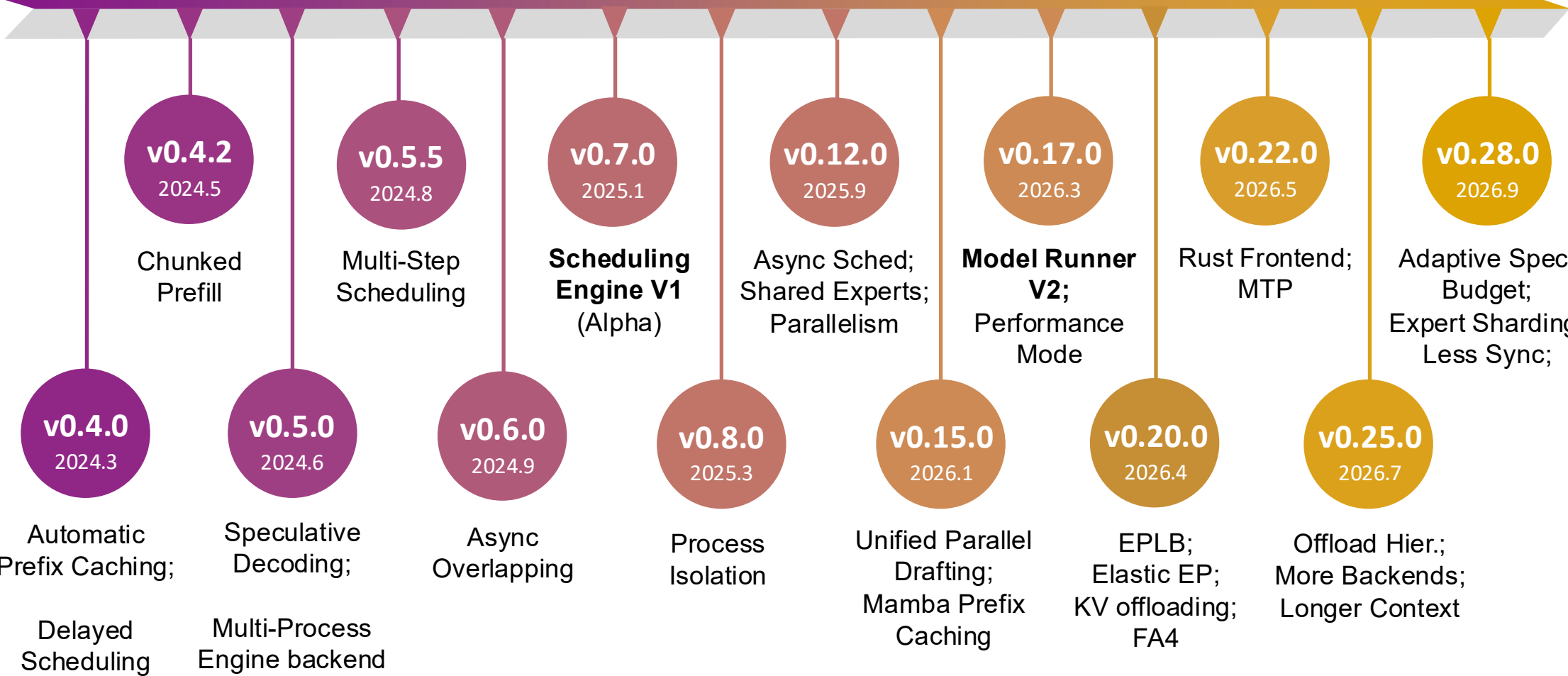


- MLP
- Attention
- Comm.
- ...



Version Drift in Simulators

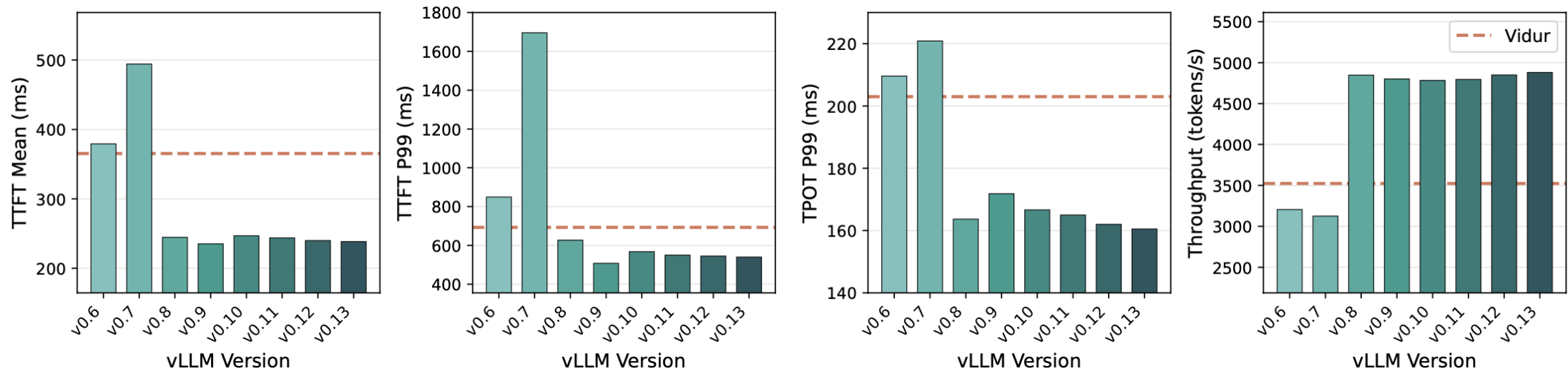
LLM serving software is **constantly evolving**.



Version Drift in Simulators



Reimplemented control plane **drifts** as production software evolves.

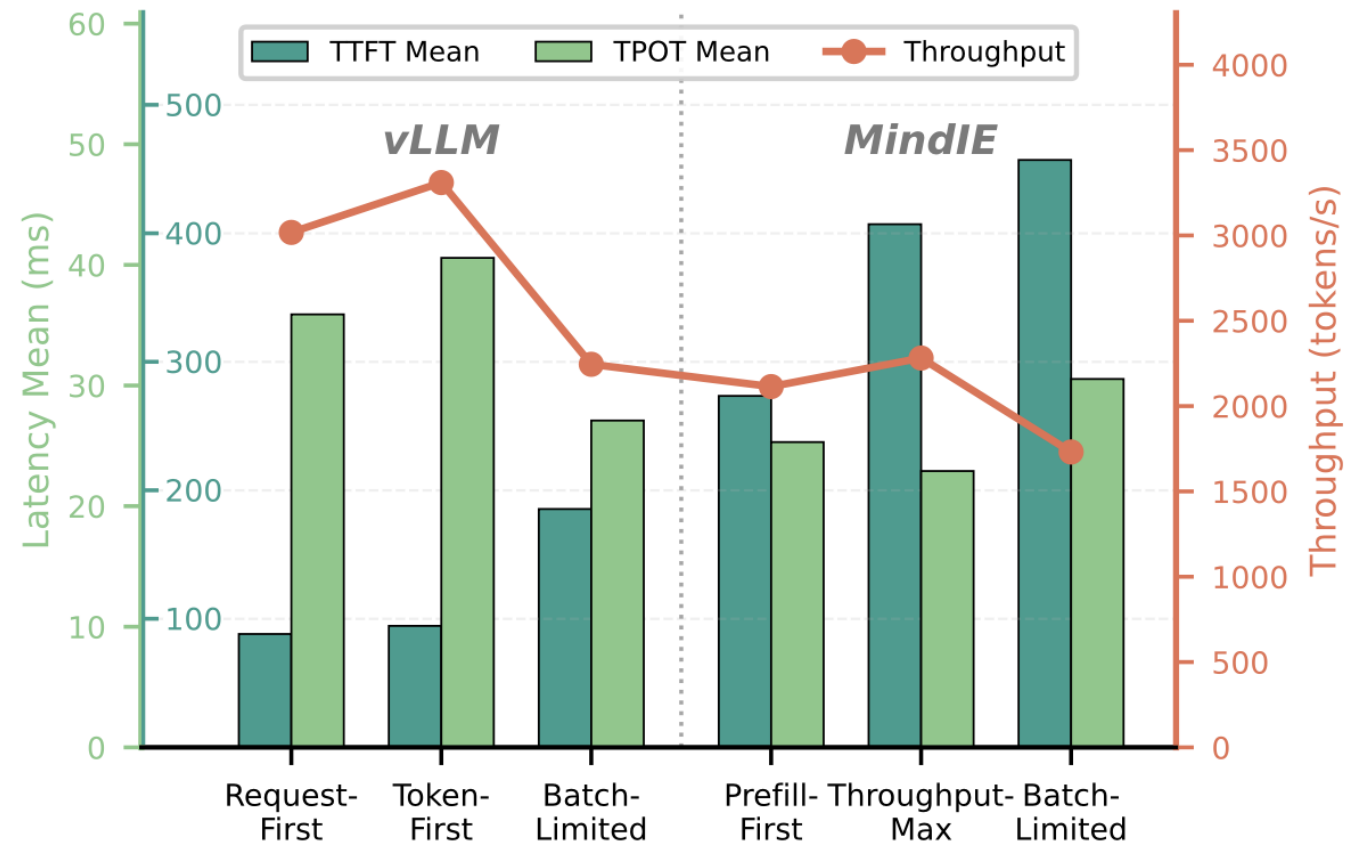


The simulator becomes untrustworthy exactly when we need it most,
to tune the newest release.



Observation: Impact of Serving Control Plane

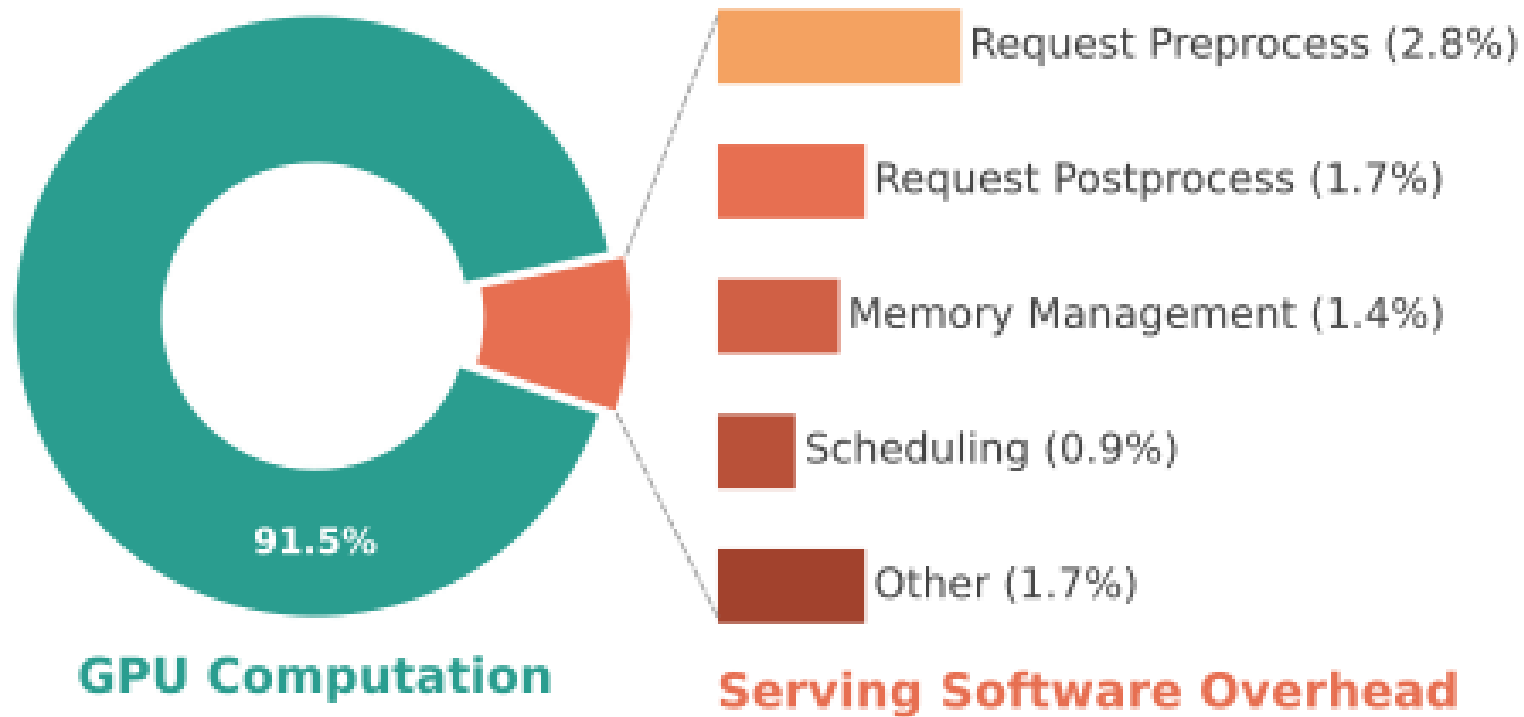
- Different settings of control logic can shift the TTFT by up to 67%





➤ Observation: The Asymmetry

- GPU computation accounts for **> 90%** of runtime
- Control plane dominates performance, but takes **<10%**





➤ Simulation at the Right Boundary

Execute what evolves. Estimate what costs.

EXECUTE NATIVELY

Serving control logic

Request queue
Scheduling and batching
Memory management



ESTIMATE LATENCY

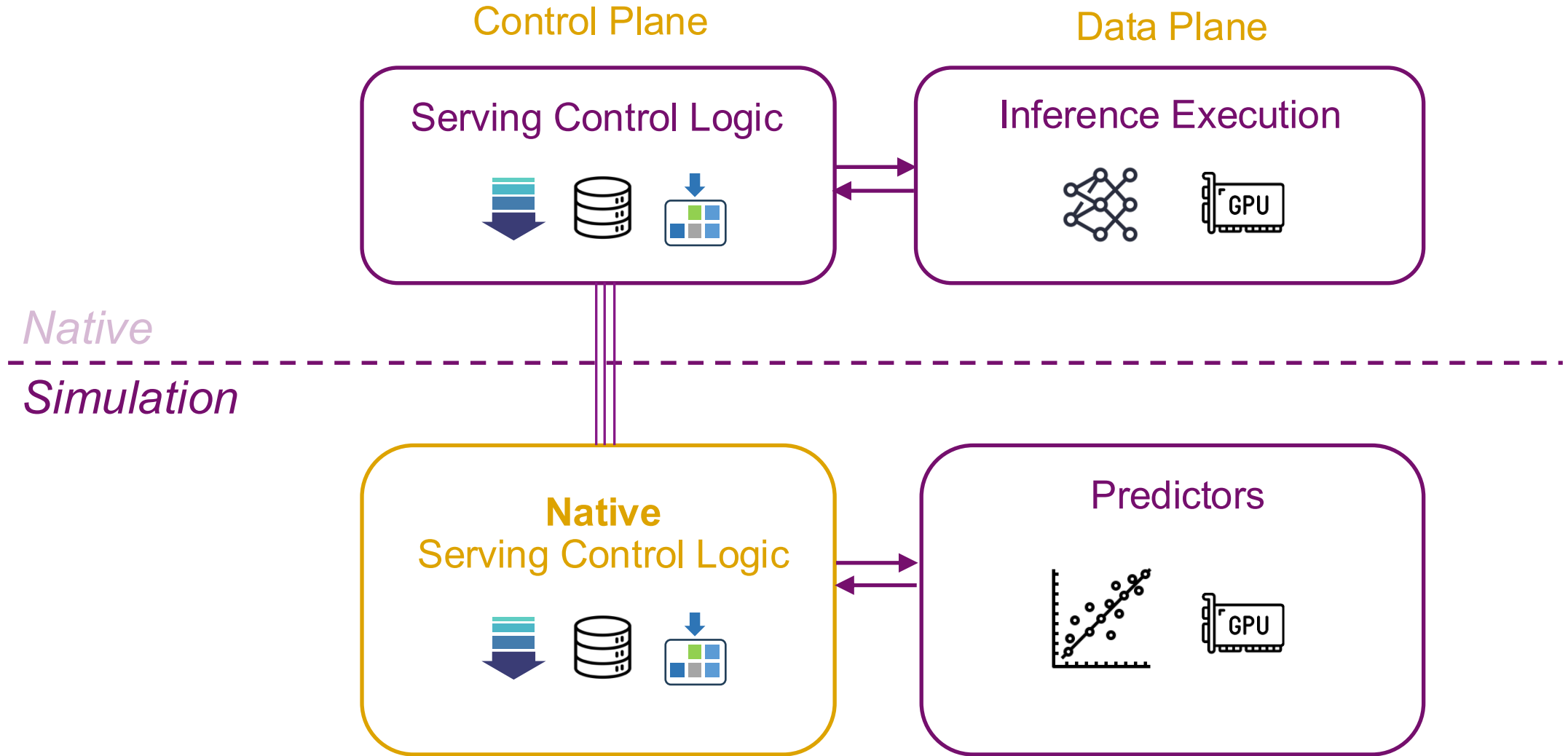
Inference execution

Skip GPU execution
Pluggable latency predictor
Virtual time clock

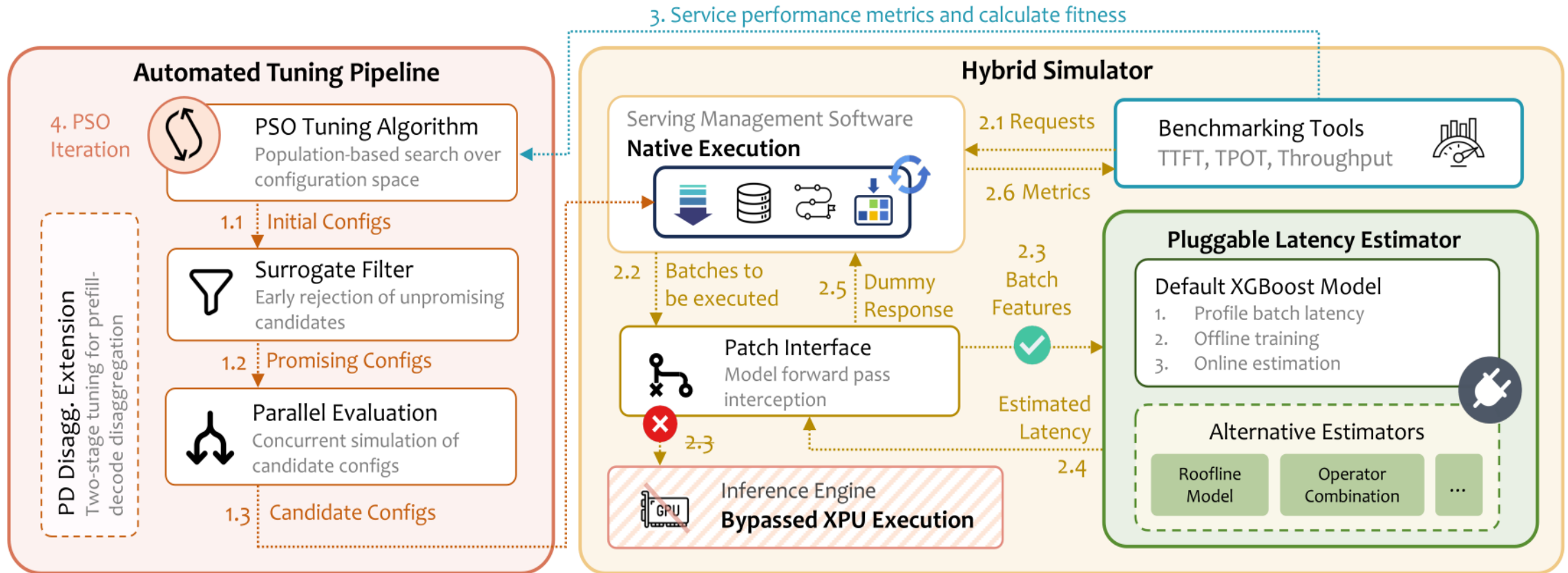
Native control follows the target version; the estimator removes GPU execution cost.



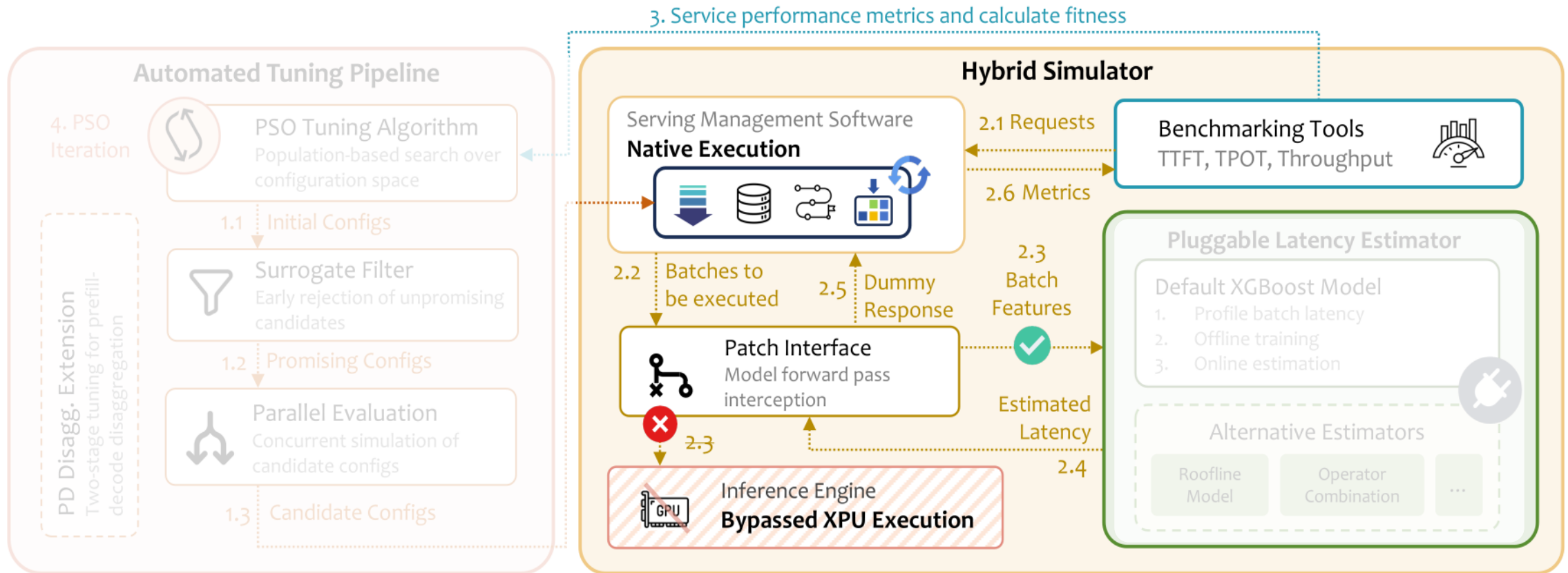
➤ Simulation at the Right Boundary



SimServing Overview

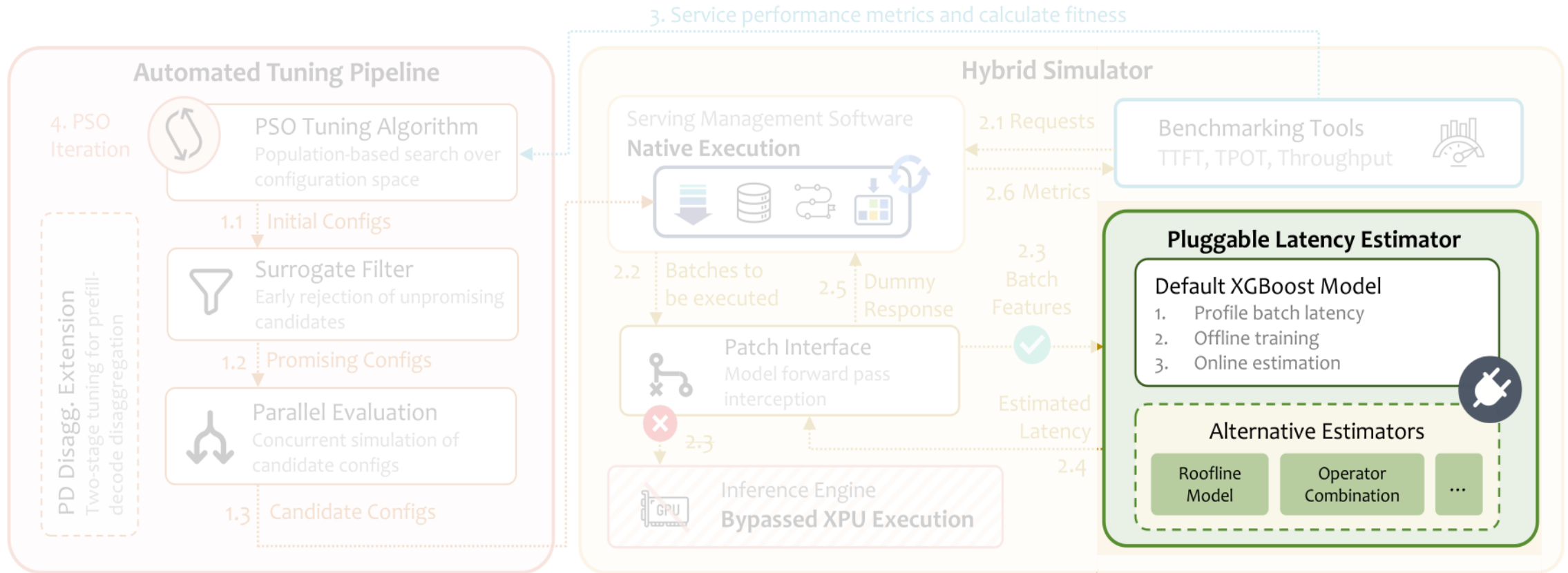


Hybrid Simulation Architecture





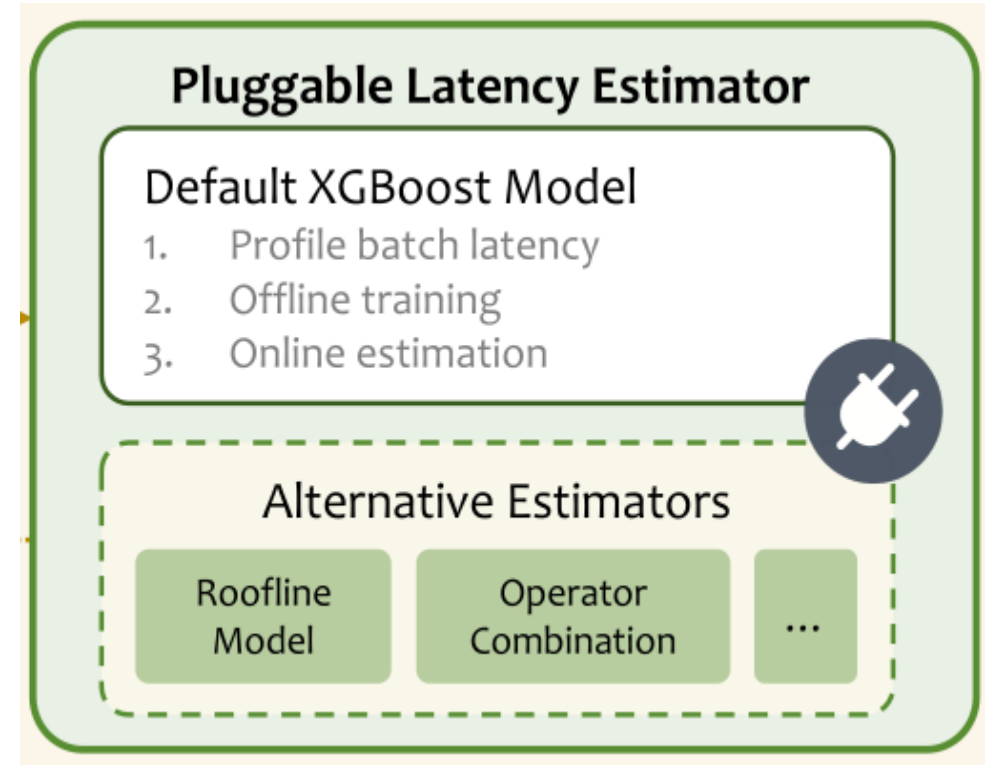
➤ Pluggable Latency Estimator





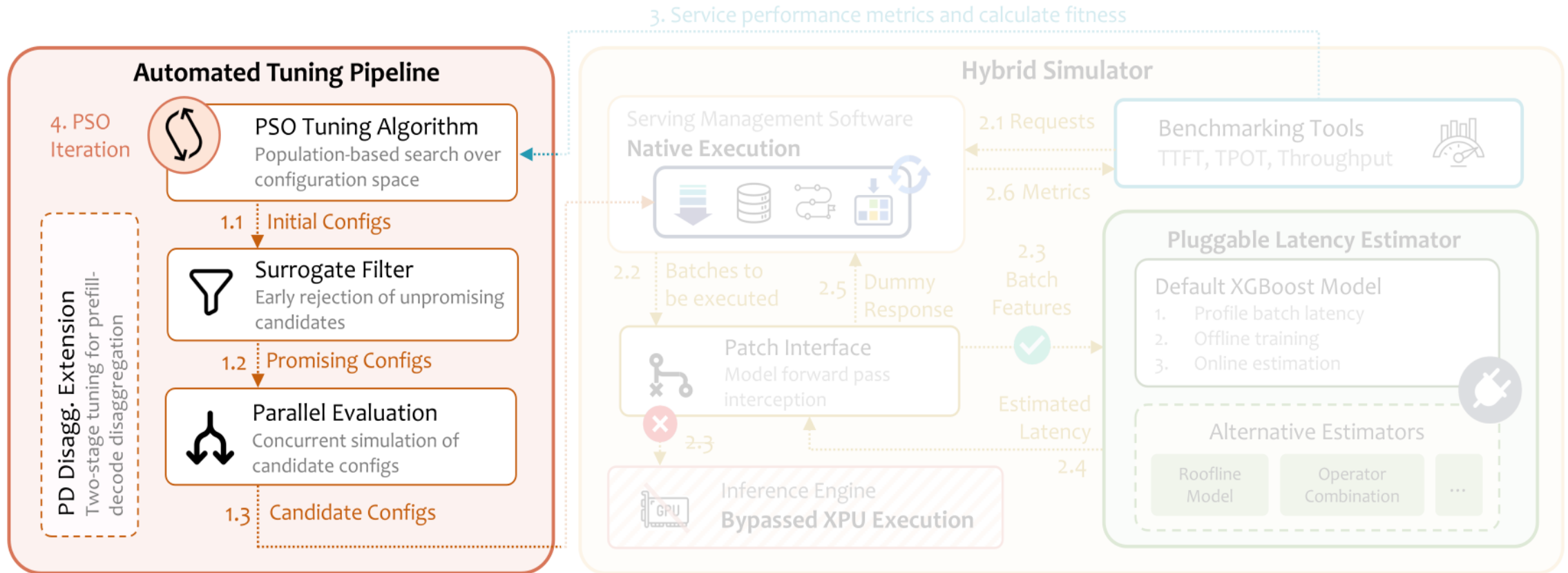
➤ Pluggable Latency Estimator

- **XGBoost predictor**
 - 30min of offline training
 - 2ms / online estimation
- **Pluggable interface** for more models

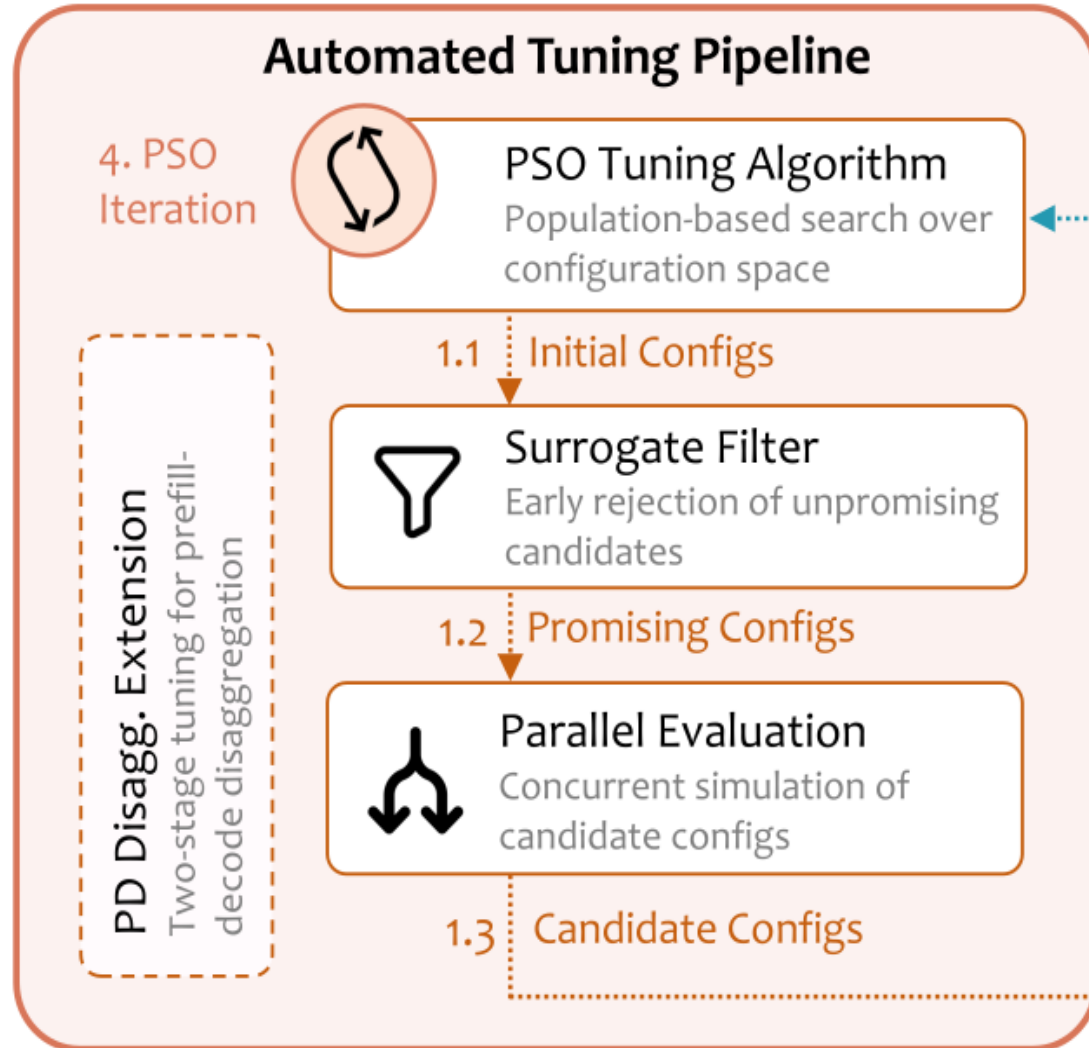




Configuration Tuning Pipeline



Configuration Tuning Pipeline



- **Particle Swarm Optimization**
 - General algorithm
 - Allowing parallel evaluation
 - Also a replaceable interface for more optimization algorithms

- How does SimServing's **hybrid simulation approach** compare to prior work in fidelity and efficiency?
- How effective is SimServing as an **end-to-end configuration tuning** framework compared to existing methods?
- How accurate and efficient is the **latency estimator**?

Model:

- Llama-3.1-8B
- Qwen3-30B-A3B

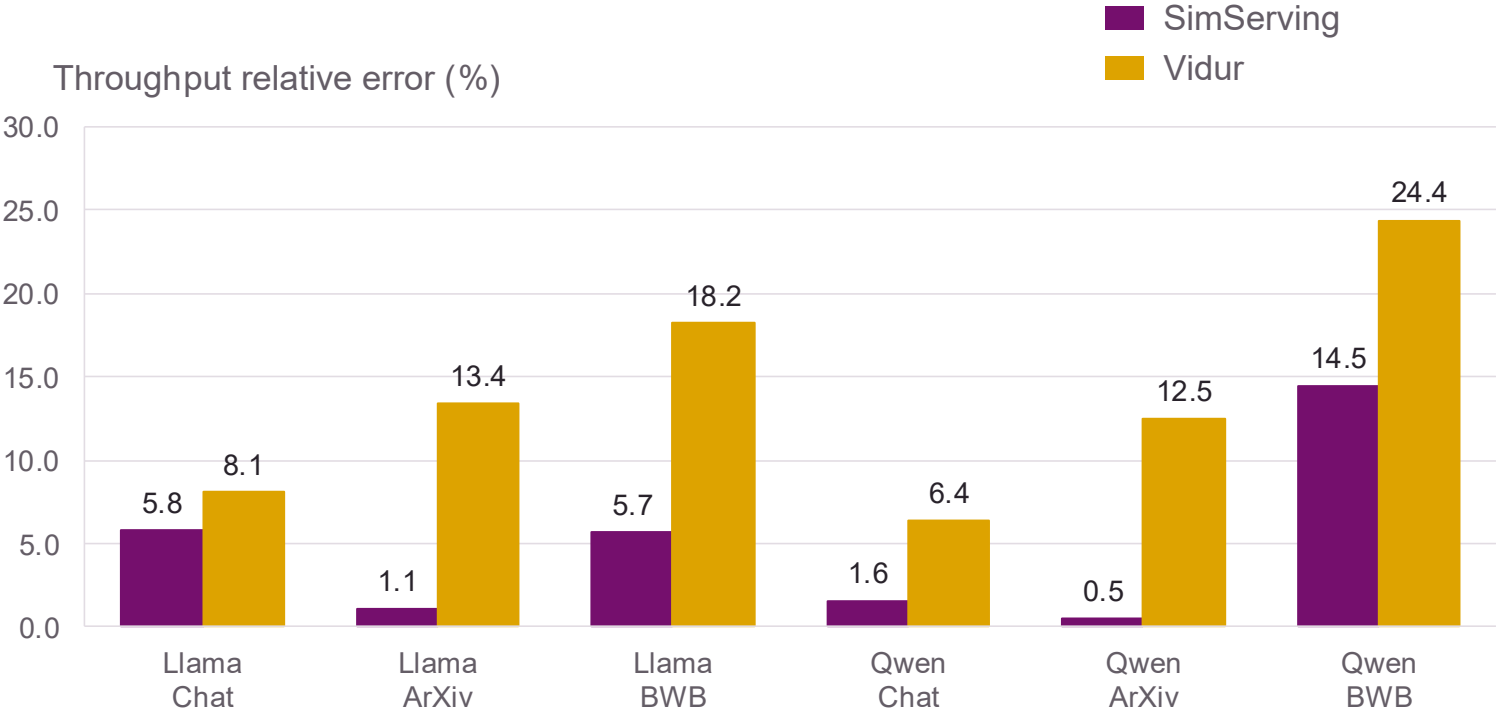
Workload:

- Arxiv-Summarization (Prefill-intensive)
- BWB (Decode-intensive)
- LMSYS-Chat-1m (Balanced, multi-turn)



➤ First: Simulation Fidelity and Speed

Lower throughput prediction error across all six model–workload settings.



13.8% → 4.9%

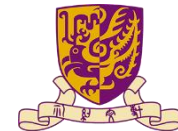
average throughput error

21.18×

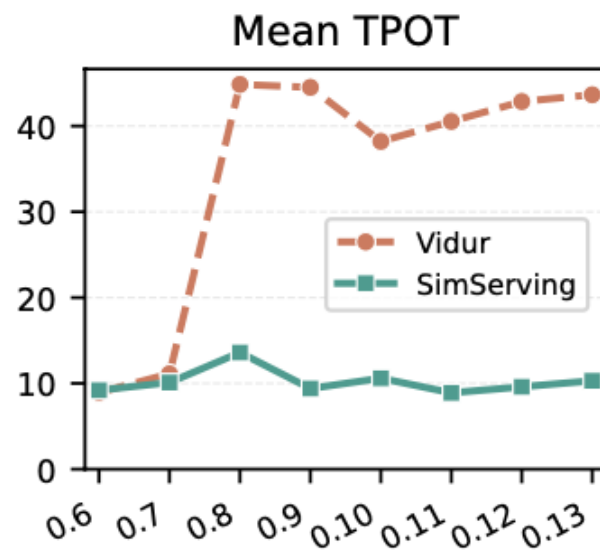
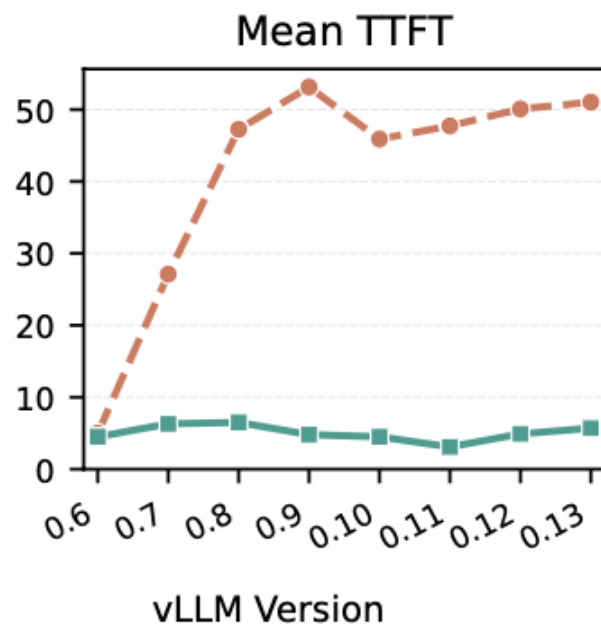
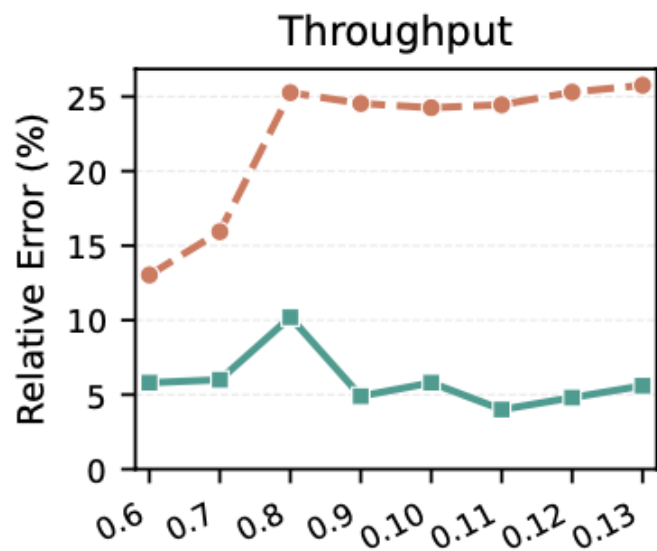
maximum simulation speedup
over native benchmarking.



First: Simulation Fidelity and Speed



Hybrid simulation architecture is **evolution-resilient!**





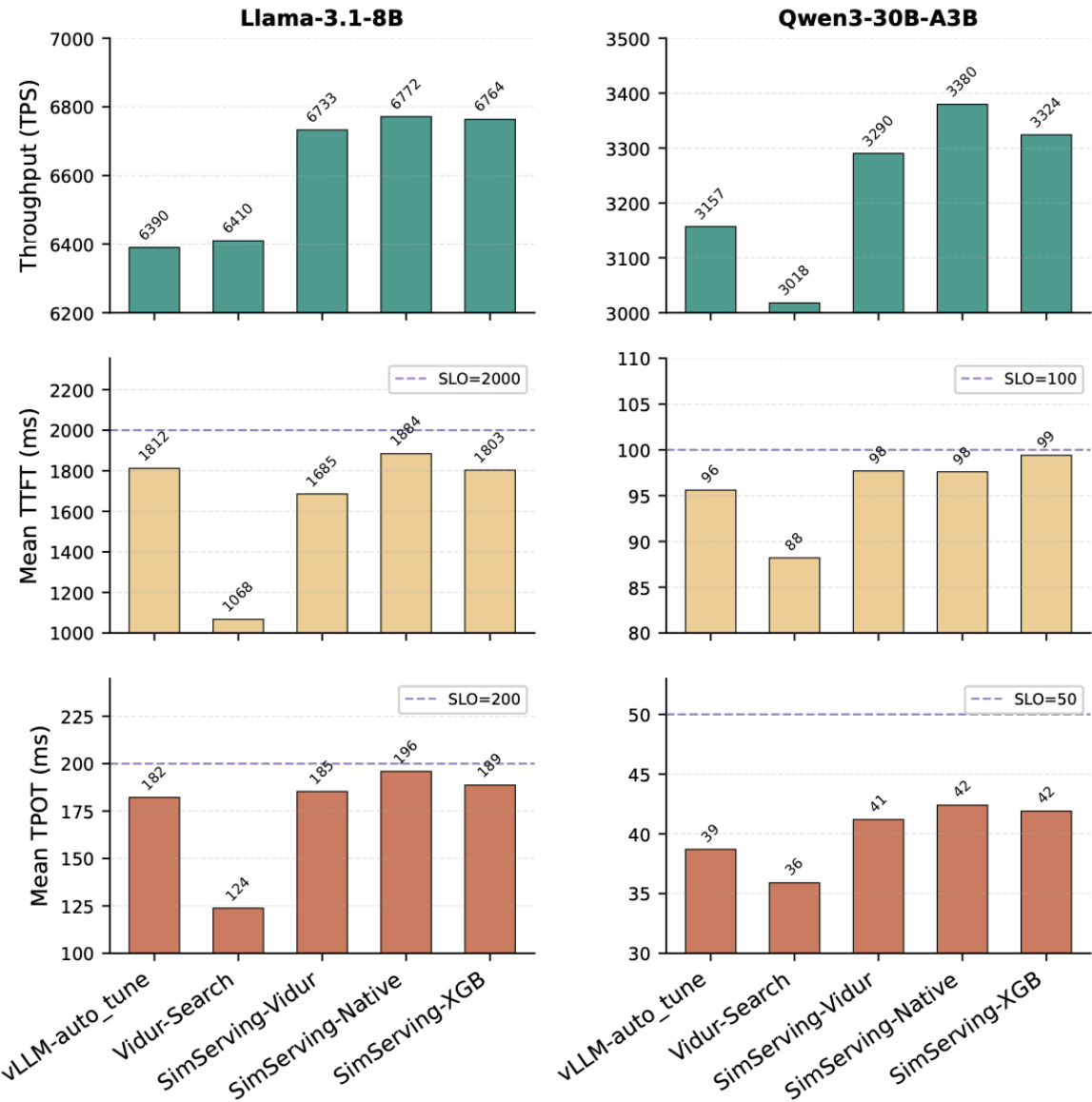
➡ Second: Better Configurations, Less Tuning Time

Comparing Tuning Algorithms

- vLLM auto-tune
- Vidur-Search
- SimServing

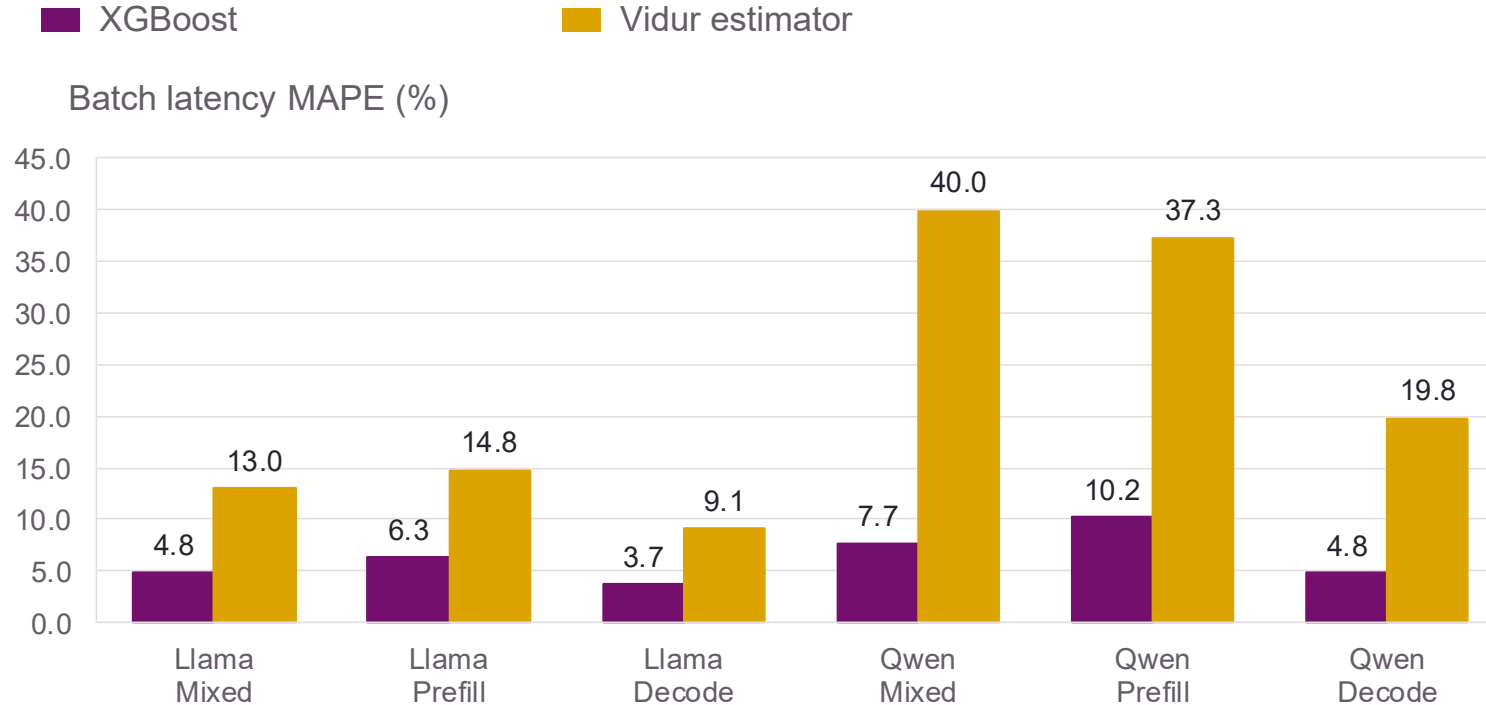
End-to-end Tuning Speed up (Llama)

- vLLM auto-tune: >24h
- SimServing-Native: 5.18h
- SimServing-XGB: 0.39h





Third: Accurate, Lightweight Latency Estimation



6.25%

average batch latency MAPE

1.92 ms

average CPU prediction cost



➤ Extensions to Other Serving Settings

The same interception and tuning interface supports additional deployments.

Prefill-Decode Disaggregation

Tune the allocation of 8 nodes



3P : 5D +17.0%

27,375.9 tokens / s

vs manual 4P:4D; Llama-3.1-8B

MindIE · Ascend 910B

Native framework integration

Qwen3-32B industrial deployment

720 → 750

tokens / second · +4.2%

SLO: TTFT < 3 s; TPOT < 0.05 s



Execute what evolves. Estimate what costs.

Keep the native serving control logic and
Replace the expensive model inference execution.

4.9%

average throughput error

21.18×

maximum simulation speedup

+10.2%

throughput over Vidur-Search

Thank you!!

Yujie Huang · yjhuang@cse.cuhk.edu.hk