UNIVERSITY OF CAMBRIDGE

BOute: Cost-Efficient LLM Serving with Heterogeneous LLMs and GPUs

Authors: Youhe Jiang[1], Fangcheng Fu[2], Eiko Yoneki[1]

[1]University of Cambridge, [2]Shanghai Jiao Tong University

Presenter: Ran Yan[3]

[3]Hong Kong University of Science and Technology

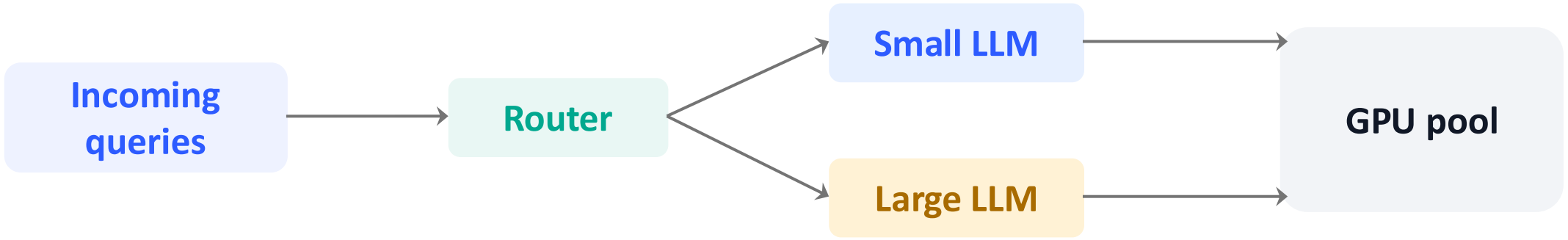
UNIVERSITY OF CAMBRIDGE

Motivation: Serving Cost Grows with LLM Demand

Heterogeneous models: Small models are cheaper and faster; large models are more capable.

Heterogeneous GPUs: GPU types differ in memory, bandwidth, compute, and hourly cost.

Serving objective: Meet latency and quality targets with the lowest possible cost.



Key question: How should routing and deployment be chosen together to minimize the serving cost?

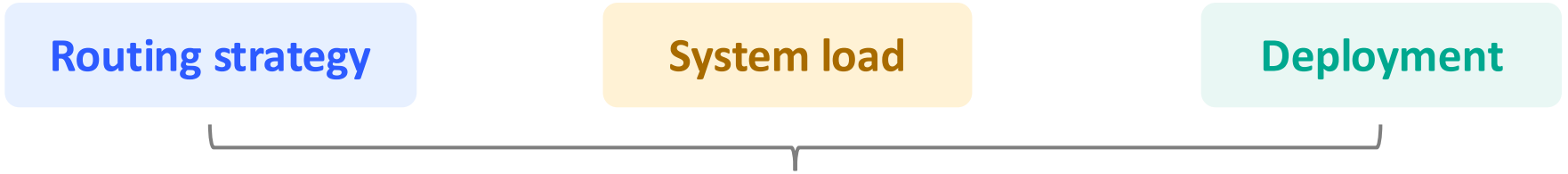
UNIVERSITY OF CAMBRIDGE

Key Challenges:

Effective routing: Routing must balance quality and latency, but model latency changes with deployment.

Efficient deployment: Each model prefers different GPU allocations and parallelism strategies.

Coupled decisions: Routing changes model load; deployment changes achievable latency.

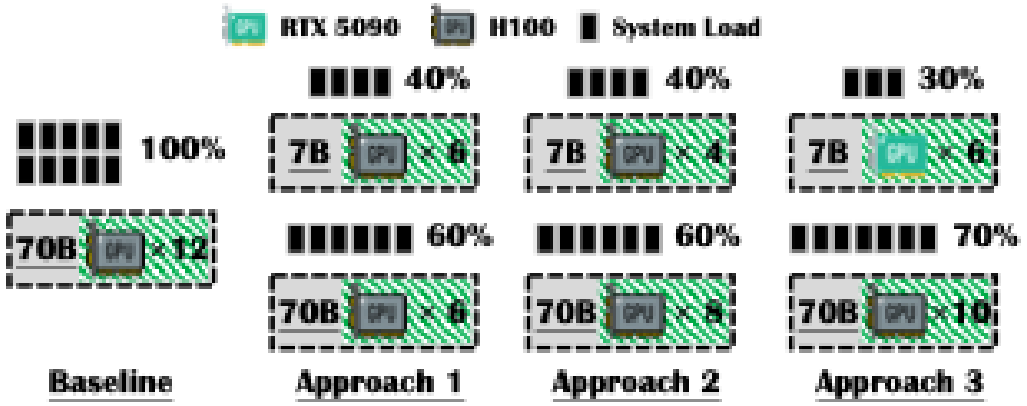


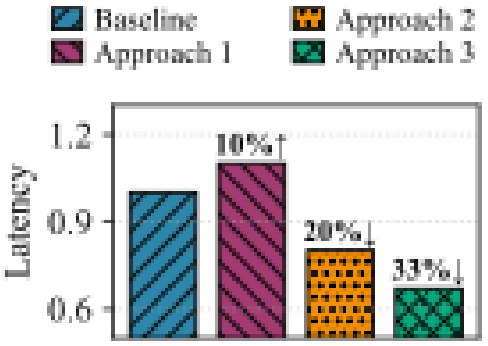
Optimizing one without the other creates bottlenecks.

UNIVERSITY OF CAMBRIDGE

Workload Characterization: Four Approaches

- **Baseline:** Serve all queries with one large model and homogeneous GPUs.
- **Approach 1:** On top of **Baseline**, add routing mechanism (RouteLLM), and allocate resources uniformly for different models.
- **Approach 2:** On top of **Approach 1**, adjust resource allocation according to model load and size, optimizing resource utilization.
- **Approach 3:** On top of **Approach 2**, utilize heterogeneous resources for model serving, boosting serving cost efficiency.





Best characterization result: Approach 3 (heterogeneous deployment) lowers P95 latency to 17.1s.

UNIVERSITY OF CAMBRIDGE

Core Insight: Model-GPU Matching Matters

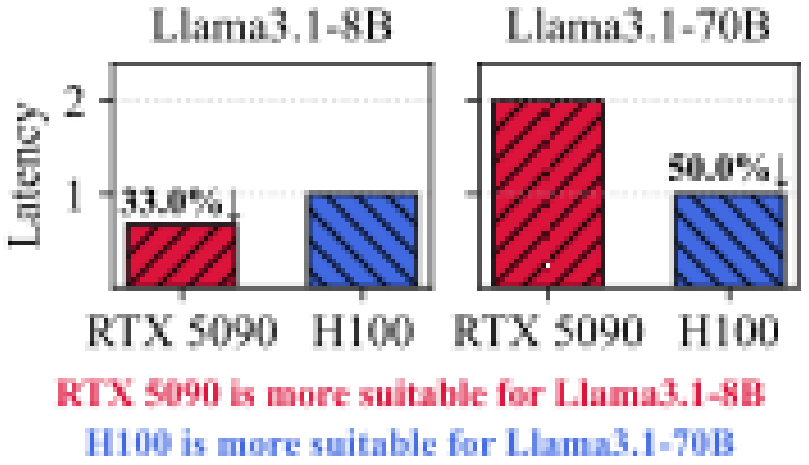
1.5x lower P95 latency for small model on RTX 5090 vs H100.

2x lower P95 latency for large model on H100 vs RTX 5090.

30/70 routing split after heterogeneous deployment in the example.

Llama3.1-8B → **RTX 5090**

Llama3.1-70B → **H100**



Heterogeneity is not just tolerated - it is exploited.

BOute uses this preference during scheduling.

UNIVERSITY OF CAMBRIDGE

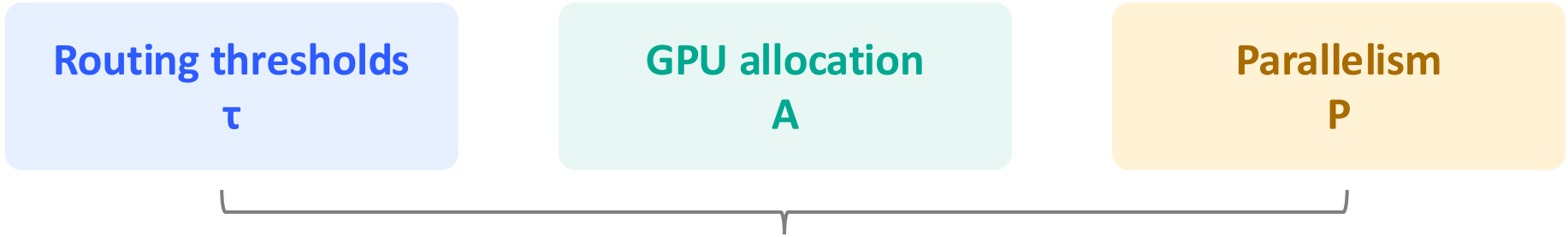
Problem Formulation

Find Pareto-optimal serving plans:

- Minimize system response latency.
- Maximize response quality.
- Respect budget and GPU availability.

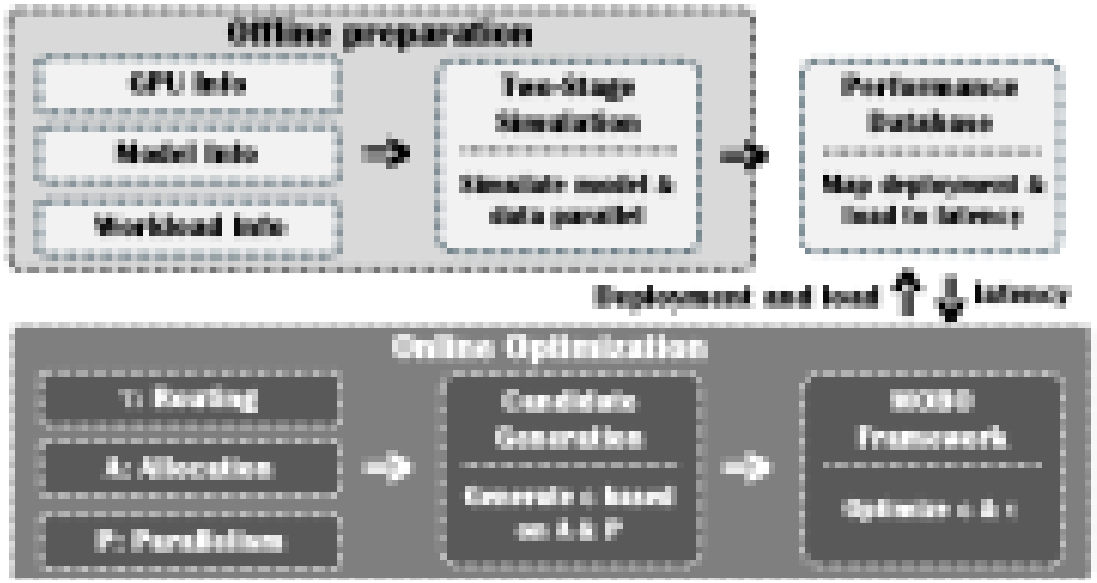
Why MOBO (Multi-Objective Bayesian Optimization)?

The search space is large, discrete, and coupled. Exhaustive search is infeasible.



Solution: Utilizing MOBO to learn the latency-quality frontier efficiently.

BOute System Overview



Offline preparation

Profile and simulate deployments once to build a performance database.

Online optimization

Use MOBO to choose routing and deployment under constraints.

Input: GPU info + model info + workload trace

Output: Routing policy + deployment plan

UNIVERSITY OF CAMBRIDGE

Offline Preparation Phase

1. Profile inputs

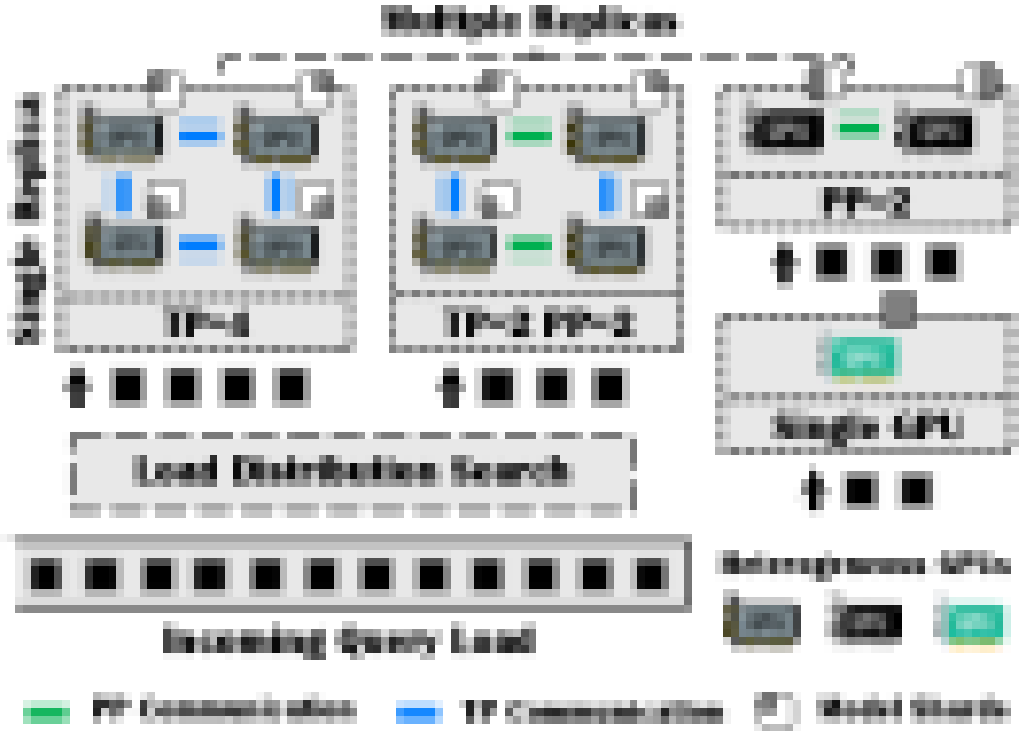
Collect GPU characteristics, model properties, and workload sequence lengths.

2. Two-stage simulation

Simulate model parallelism first, then data-parallel replicas and load distribution.

3. Build performance DB

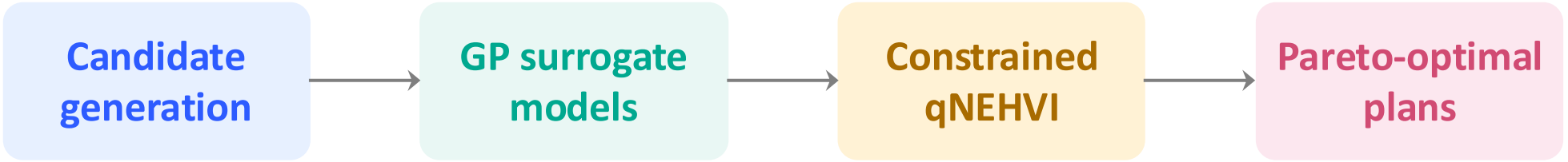
Map each deployment-load pair to estimated P95 latency for fast lookup.



Purpose: Avoid expensive real-time profiling during MOBO.

UNIVERSITY OF CAMBRIDGE

Online Optimization Phase



Structural information improves sample efficiency:

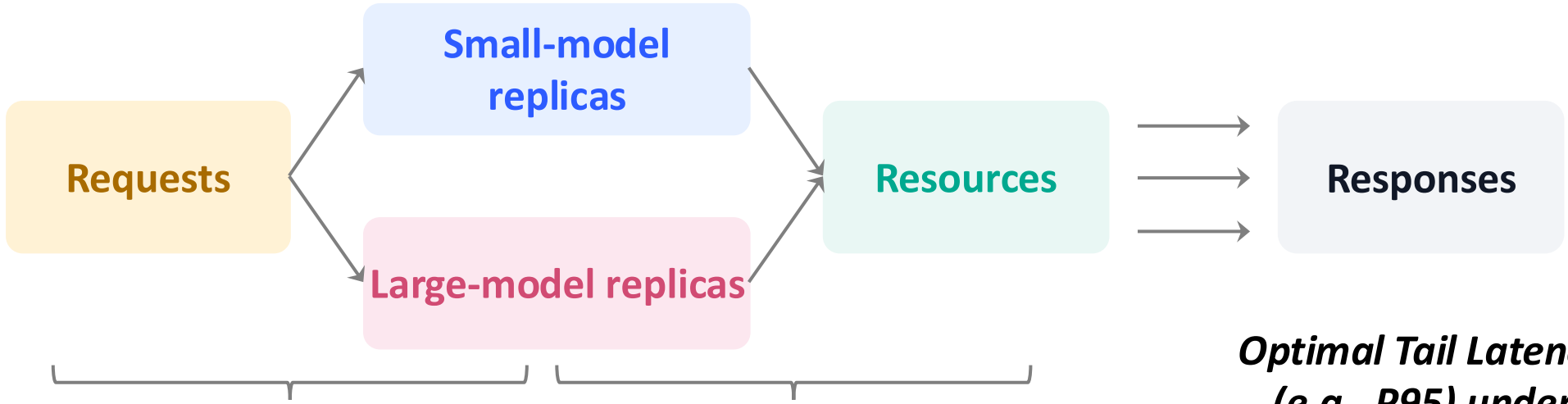
- Load-fraction encoding for routing.
- Hard budget and GPU constraints.
- Model-GPU preference encoding.

The framework converges to Pareto-optimal solutions that balance system latency and response quality under given constraints and requirements.

System Implementation

The MOBO Scheduler determines the **Routing Strategy** and **Deployment Plan**.

- Routing Strategy determines the *request assignment* across different models.
- Deployment Plan determines the *resource allocation* across different models.



UNIVERSITY OF CAMBRIDGE

Experimental Setup

GPU cluster: H100, RTX PRO 6000, RTX 5090, RTX 4090 with Budget: $30/hour.

Models, Workloads: Llama3.1-8B and Llama3.1-70B on GSM8K and MTBench traces.

Metrics: P95 latency, average latency, tail latency, throughput, quality.

Router: We adopt the RouteLLM router for routing decisions.

Baselines:

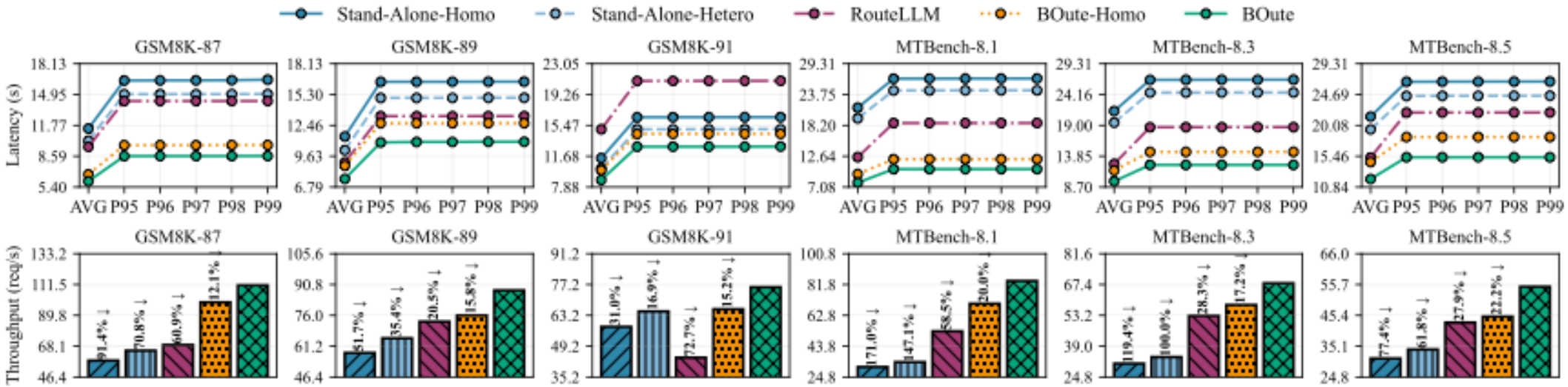
- Stand-Alone-Homo / Stand-Alone-Hetero.
- RouteLLM with uniform resource allocation.
- BOUTE-Homo w/o heterogeneous GPU deployment.

Goal: Compare under the same cost budget and quality requirements.

UNIVERSITY OF CAMBRIDGE

Main Results: Lower Latency and Higher Throughput

- **2.57×** lower P95 latency at best.
- **1.58×** average latency improvement.
- **1.9×** higher throughput at best.



Across GSM8K and MTBench, BOUTE consistently dominates baselines under fixed quality targets.

UNIVERSITY OF CAMBRIDGE

Cost and Scheduling Performance

- **15-61%** cost reduction while maintaining targets.
- **38%** average cost reduction.
- **< 1 min** scheduling for 3-5 GPU types in reported cases.

Table 3. Costs required to meet quality and latency requirements. (87, 8) represents quality and latency requirements of 87 and 8s.

Baseline	**GSM8K (87, 8)**	**GSM8K (91, 12)**	**MTBench (8.1, 10)**	**MTBench (8.5, 15)**
Stand-Alone-Homo	$61.83/h	$41.21/h	$79.93/h	$53.29/h
Stand-Alone-Hetero	$56.24/h	$37.50/h	$73.54/h	$49.02/h
RouteLLM	$53.55/h	$52.38/h	$55.90/h	$44.00/h
BOUTE-Homo	$38.41/h	$36.07/h	$36.33/h	$38.31/h
BOUTE	**$32.25/h**	**$32.18/h**	**$30.98/h**	**$31.23/h**

Table 2. The scheduling (algorithm converge) time and scalability of BOUTE. BOUTE (w/o structural info) disables algorithmic reparameterization, output transformation, and heterogeneous GPU awareness. BOUTE (w/o offline prep) disables the performance database. The algorithm is considered converged when the solution remains stable for more than 20 consecutive iterations.

GPU Types / GPU Number	**BOUTE**	**BOUTE (w/o structural info)**	**BOUTE (w/o offline prep)**
3 / 24	23.5 s	3.6 min	$\approx$ 12 min
4 / 32	32.5 s	5.3 min	$\approx$ 20 min
5 / 40	48.6 s	7.1 min	$>$ 30 min
6 / 48	1.2 min	8.9 min	$>$ 30 min

Routing Strategies and Resource Allocations

Table 1. Routing strategies and resource allocations of BOUTE used in Figure 4.

Trace-Quality	Model Type	Load	Budget	Allocated Resources
GSM8K-87	Llama3.1-8B	73.45%	38.58%	8×5090+5×4090
	Llama3.1-70B	26.55%	61.42%	6×H100
GSM8K-89	Llama3.1-8B	43.95%	17.86%	6×5090
	Llama3.1-70B	56.05%	82.14%	8×H100
GSM8K-91	Llama3.1-8B	29.94%	9.28%	3×5090
	Llama3.1-70B	70.06%	90.72%	6×H100+4×6000
MTBench-8.1	Llama3.1-8B	80.58%	46.34%	8×5090+8×4090+6000
	Llama3.1-70B	19.42%	53.66%	4×H100+2×6000
MTBench-8.3	Llama3.1-8B	61.17%	24.44%	8×5090
	Llama3.1-70B	38.83%	75.56%	6×H100+2×6000
MTBench-8.5	Llama3.1-8B	41.75%	12.04%	4×5090
	Llama3.1-70B	58.25%	87.96%	6×H100+4×6000

UNIVERSITY OF CAMBRIDGE

Routing Strategies and Resource Allocations

Insight 1: Load distribution adapts to quality requirements:

As quality requirements increase, the framework progressively shifts system load and computational budget toward the Llama3.1-70B model.

Insight 2: Resource/Budget allocation reflects model-specific resource requirements.

The framework allocates computational resources based not only on assigned request load but also on intrinsic model characteristics (e.g., compute and memory demands). → Typically, small models require fewer resources even under larger request loads.

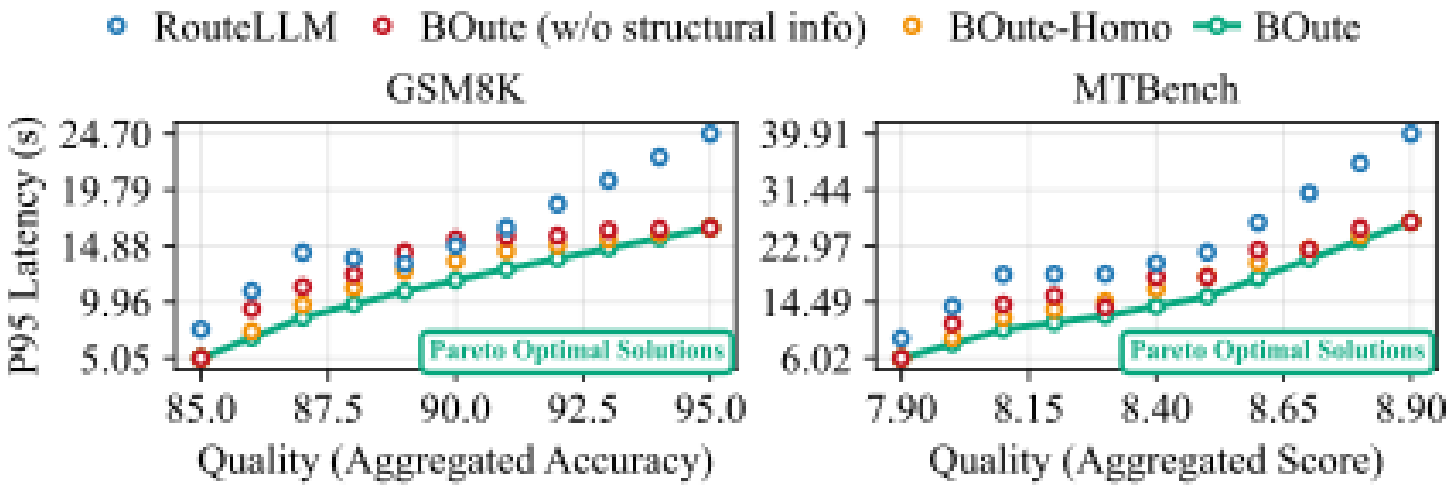
Insight 3: Heterogeneous GPU allocation optimizes resource utilization.

The framework strategically assigns heterogeneous GPU resources to different model types based on their computational requirements and GPU availability.

UNIVERSITY OF CAMBRIDGE

Takeaways

- Routing and deployment must be co-optimized.
- Heterogeneous GPUs improve cost-efficiency when matched to models.
- MOBO finds practical latency-quality Pareto solutions.



BOUTE turns heterogeneity into an optimization opportunity.

UNIVERSITY OF CAMBRIDGE

Thank you for watching!

Contact: yj367@cam.ac.uk