Supporting Multiple QoS Class in LLM Inferencing

Anubhav Jana*, Felix George, Kaustabha Ray, Uma Devi, Pratibha Moogi

IIT Bombay* IBM Research

PROBLEM CONTEXT

- ➔ **LLMs are increasingly popular for tasks such as text generation, code creation and conversational AI**
- ➔ **Heavily Reply on GPU for inference**
- ➔ **Different users have different SLO requirements and common approaches tend to over-allocate resources**
- ➔ **The high cost of GPU's prohibits dedicated GPUs for a user class**

OBJECTIVE

Minimize the number of GPU instances (cost + power) while meeting the latency requirements of user class prioritized by their QoS requirements (latency) by statistical multiplexing of user requests among user classes given the rates and latency bounds.

SOLUTION APPROACH

Extensive benchmarking of GPU profiles across various LLM models.

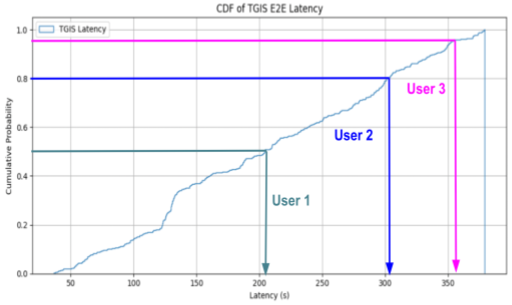
Integer Linear Programming (ILP) to minimize the number of instances required and to decide rate allocation to meet SLOs.

Multi-Instance priority-based routing based on ILP

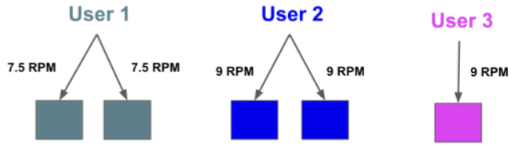
MOTIVATING EXAMPLE

User 1 : RR = 15 L = **204**
User 2 : RR = 18 L = **301**
User 3 : RR = 9 L = **352**

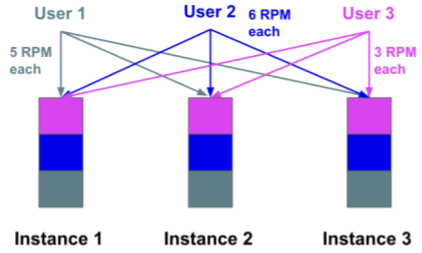
Given Instance Operating at RR = 15 RPM



Simple FCFS scheduling will require at ***least 5 inferencing instances***



Intelligent routing and simple priority scheduling can help meet all SLO's with just 3 instances.



ILP FORMULATION

- $G_{i,j}$: Binary variables indicating GPU i operated in mode $j, 1 \le i \le N, 1 \le j \le R$.
- $A_{i,j}$: Aggregate rate for GPU i in mode j.
- $U_1, U_2, \ldots, U_M$: M user classes with:
  - Request (query) rates $Q_1, Q_2, \ldots, Q_M$.
  - Latency bounds $L_1, L_2, \ldots, L_M$.
- $X_{u,i,j}$: Decision variables denoting the rate allocation to user U_u on GPU i in mode j.
- $f_{i,j}(r)$: Denotes the latency in GPU i mode j for rate r.

Challenge: building the relation of latency with request rate as a function

$$\text{Minimize} \quad \sum_{i=1}^{N}\sum_{j=1}^{R} G_{i,j}$$

$$\text{Subject to:} \quad \sum_{j=1}^{R} G_{i,j} \le 1, \quad \forall i$$

$$\sum_{i,j} X_{u,i,j} = Q_u, \quad \forall u$$

$$\sum_{u,j} X_{u,i,j} \le \sum_{j} G_{i,j} \cdot A_{i,j}, \quad \forall i$$

$$f_{i,j}\left(\sum_{k=1}^{u} X_{k,i,j}\right) \le L_u, \quad \forall i,j,u$$

$$G_{i,j} \in \{0,1\}, \quad \forall i,j$$

OBSERVATIONS

User	Metric	95th Percentile	Mean
Alan	Latency (s)	37.31	31.21
	Queue Time (s)	8.46	3.27
Noel	Latency (s)	102.27	48.71
	Queue Time (s)	75.25	21.11
Hari	Latency (s)	541.06	223.43
	Queue Time (s)	511.48	195.78

User	Metric	95th Percentile	Mean
Alan	Latency (s)	364.95	167.85
	Queue Time (s)	339.08	141.20
Noel	Latency (s)	357.23	170.57
	Queue Time (s)	336.12	143.71
Hari	Latency (s)	361.03	168.14
	Queue Time (s)	332.19	141.67

We were also able to show with priority-based multi-instance routing, latency values were maintained and respected with respect to the priority.

FUTURE WORK

Integrate all the individual components and build an end-to-end inference system, which will handle varying request rates and latency bounds per user class requirements and adapt at runtime.

The ILP model will choose the optimal instance configuration and request rate distribution at runtime based on incoming rates and SLOs.

The multi-instance router will prioritize user class requests to meet SLOs with minimal instances based on the ILP decision outputs.