# Taming Throughput-Latency Tradeoff in LLM Inference with **Sarathi-Serve**

**Amey Agrawal[1], Nitin Kedia[2]**, Ashish Panwar[2], Jayashree Mohan[2], Nipun Kwatra[2], Bhargav Gulavani[2], Alexey Tumanov[1], Ramachandran Ramjee[2]

*[1]Georgia Institute of Technology, [2]Microsoft Research India*

# Rise of LLMs 🦄

Technology

**ChatGPT sets record for fastest-growing user base - analyst note**

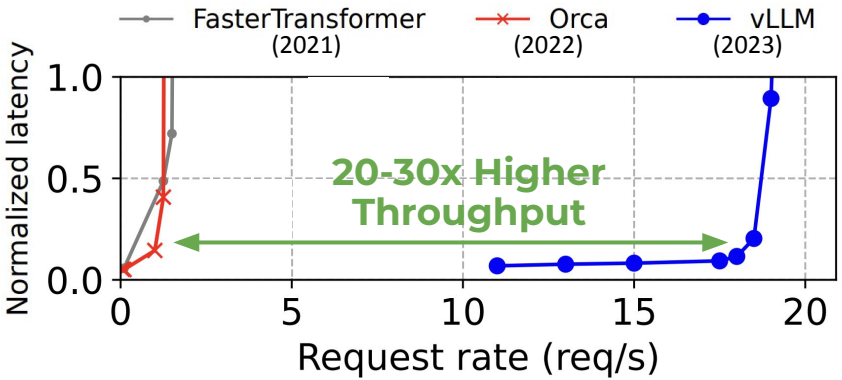
By **Krystal Hu**

February 2, 2023 7:33 AM PST · Updated a year ago

CLIMATE

**Google's carbon emissions surge nearly 50% due to AI energy demand**

PUBLISHED TUE, JUL 2 2024•3:41 PM EDT | UPDATED MON, JUL 8 2024•9:32 AM EDT




# & Inference Systems 🚀

Image credits: Kwon and Li et al.

# Can we maintain low latency with high throughput?

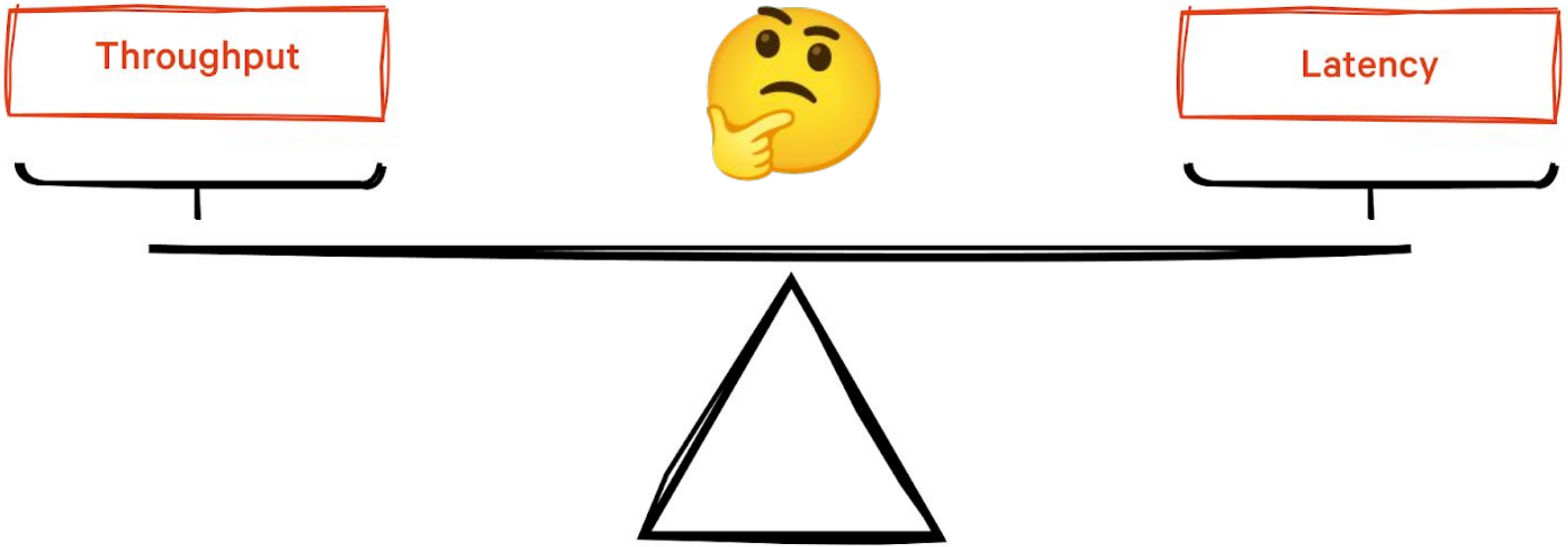



Image credits: Redpanda

# Demo 

# In this talk...

🔬 **Latency-throughput tradeoff:** Analyzing LLM batching policies

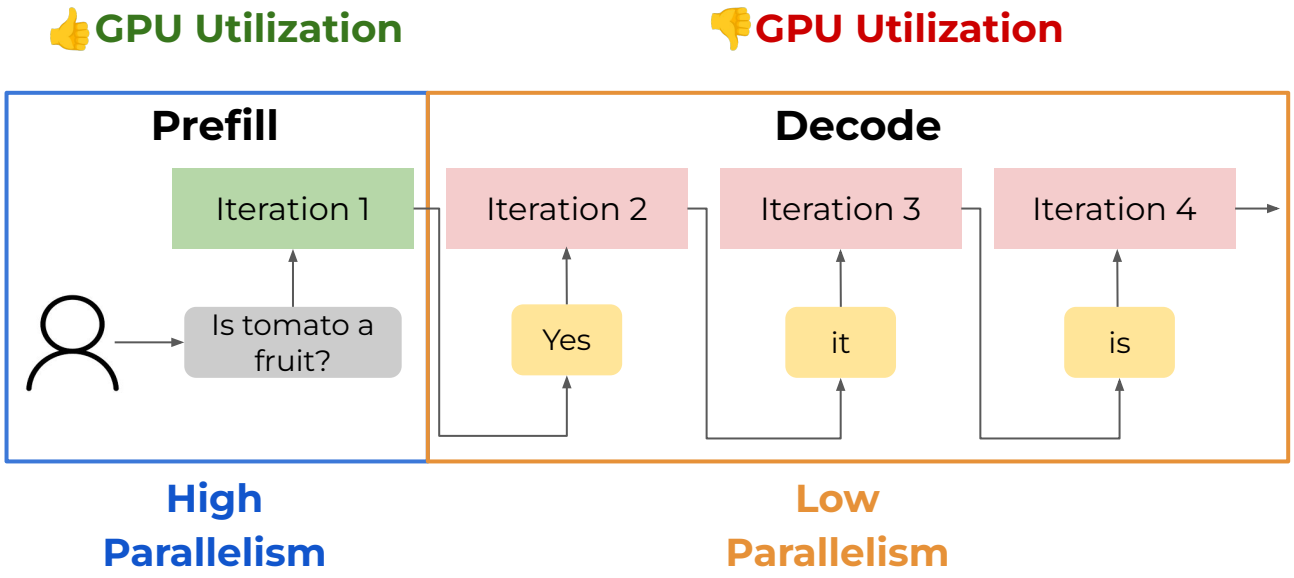
🥗 **Finding a free lunch:** Arithmetic Intensity Slack in LLM Inference

🧠 **Stall-free batching:** Leveraging chunked prefill to overcome the latency-throughput tradeoff

🧮 **Evaluations:** Key results and analysis

What causes the latency-throughput tradeoff in LLM inference systems? 🕵️

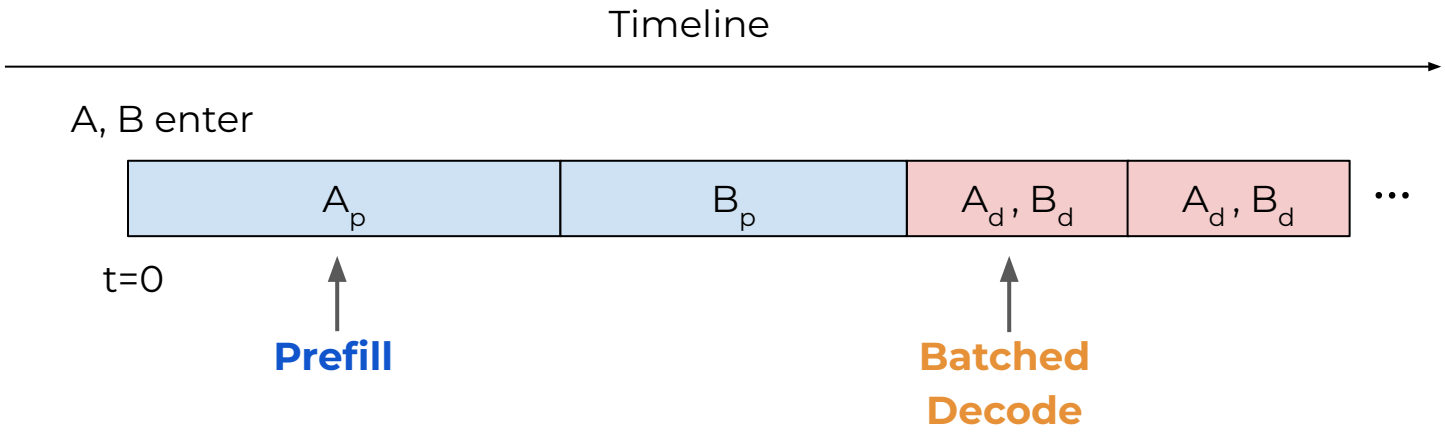
# 🔄 Background: LLM Inference 101



How to improve parallelism during decode phase? 🤔

**Batching** 🤷

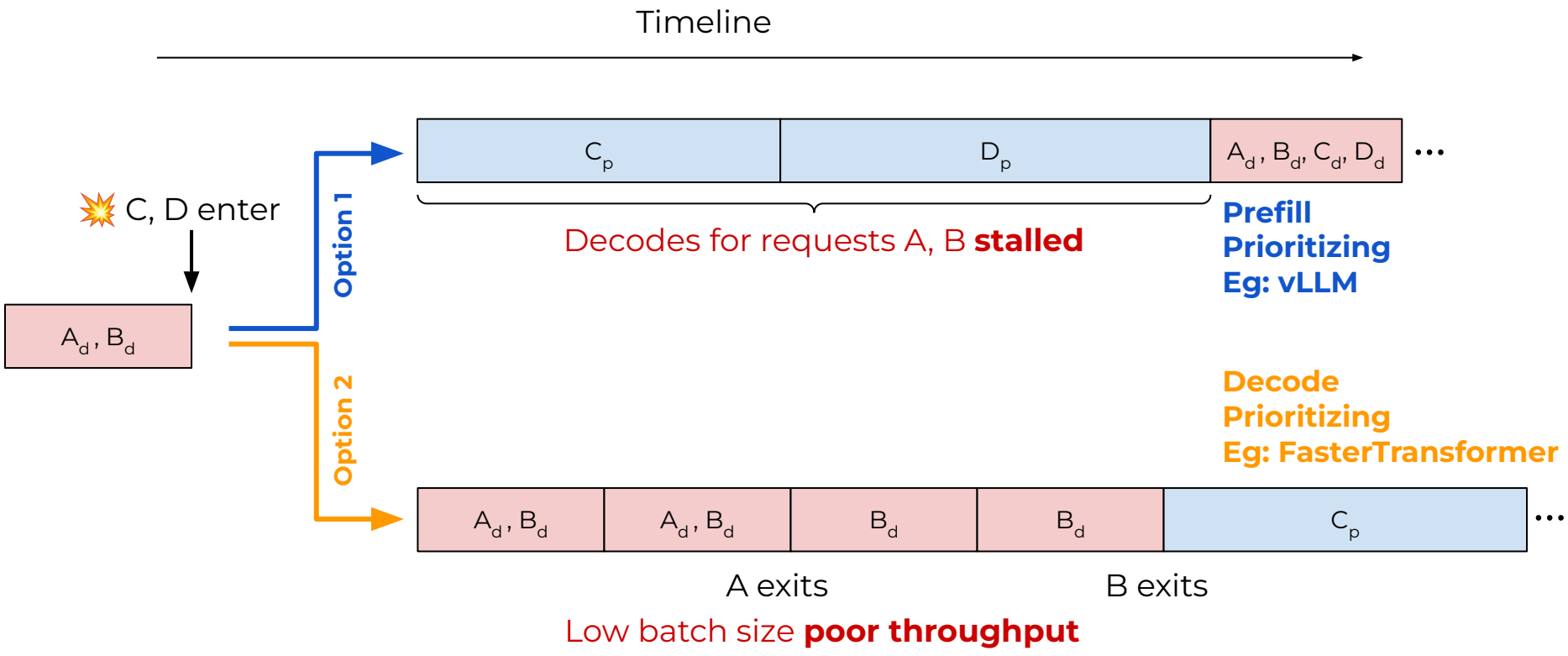
# Background: Batching LLM Inference



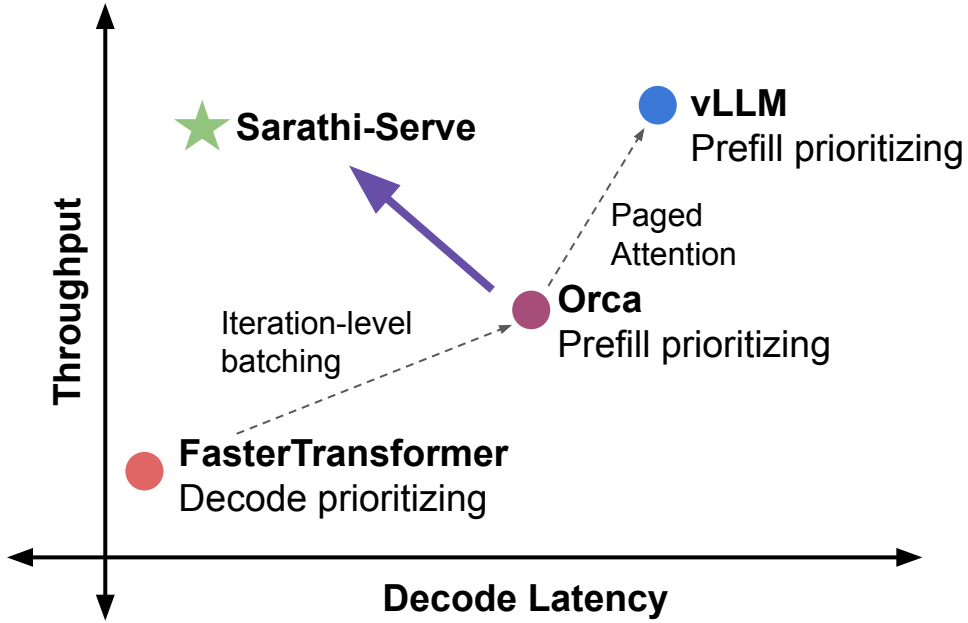

Decode efficiency increases linearly with batch size 🚀

🔼 **Batch size** ⇒ 🔼 **Throughput**

# 🔎 The Prefill-Decode Scheduling Conundrum



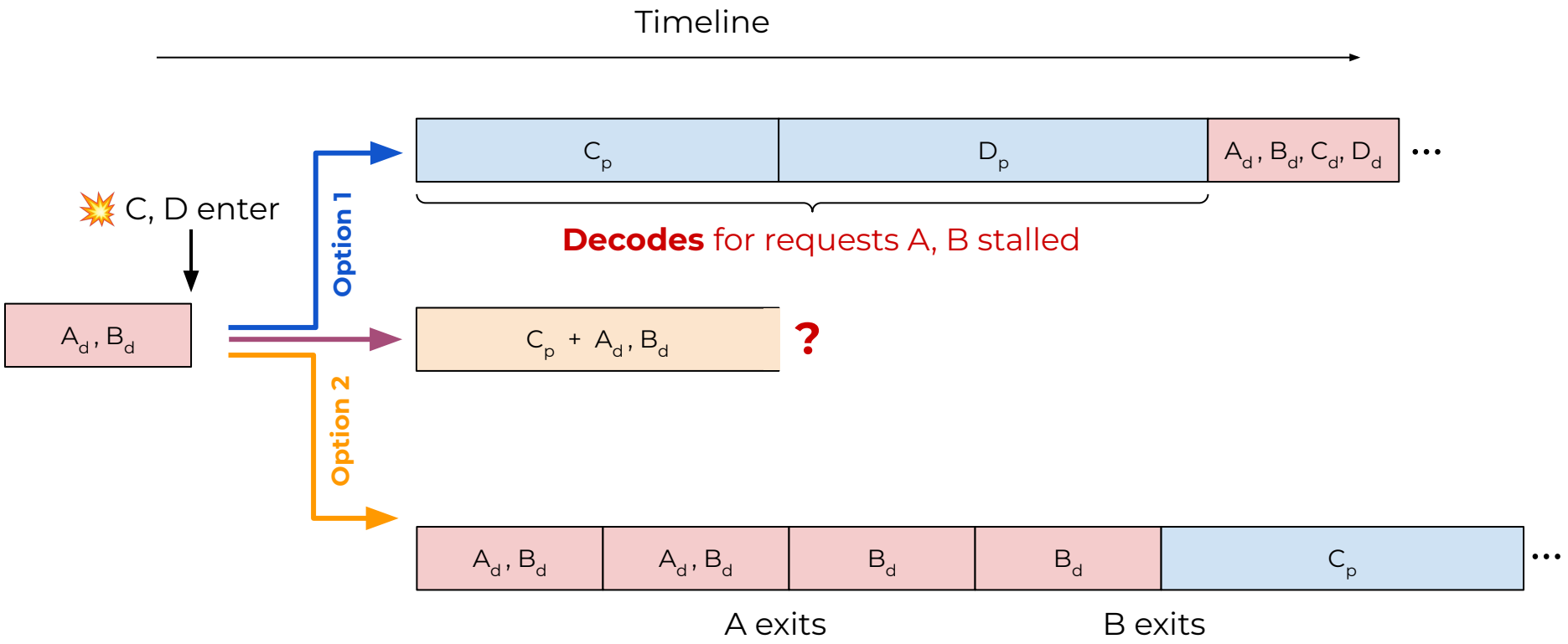
# ⚖️ The Latency-Throughput Tradeoff




**Existing batching policies make a harsh latency-throughput tradeoff ❗**

How can be we achieve both high throughput and low-latency? 🤔

# 🔎 The Prefill-Decode Scheduling Conundrum



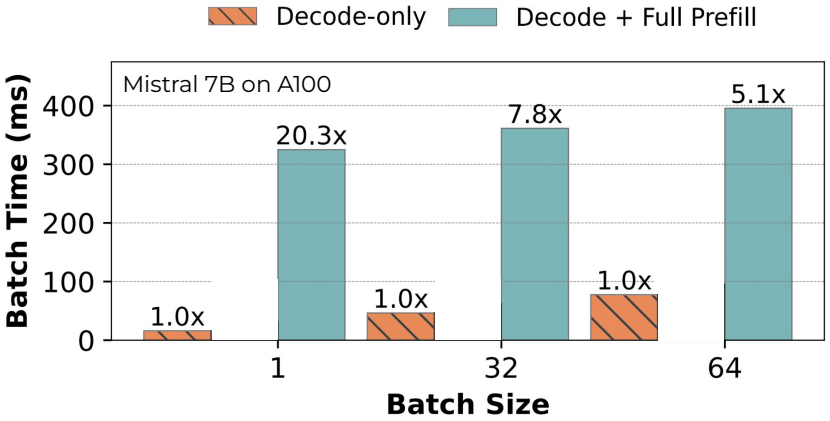
# 🔎 Mixed Batching

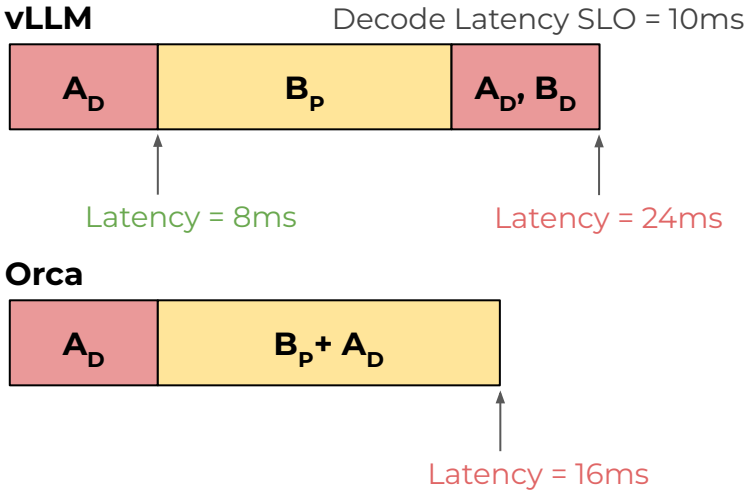
## Idea

👩‍🔧 Fused computation of prefill and decodes

## Challenge

😭 Naively combining prefill and decode operations leads to increase in latency

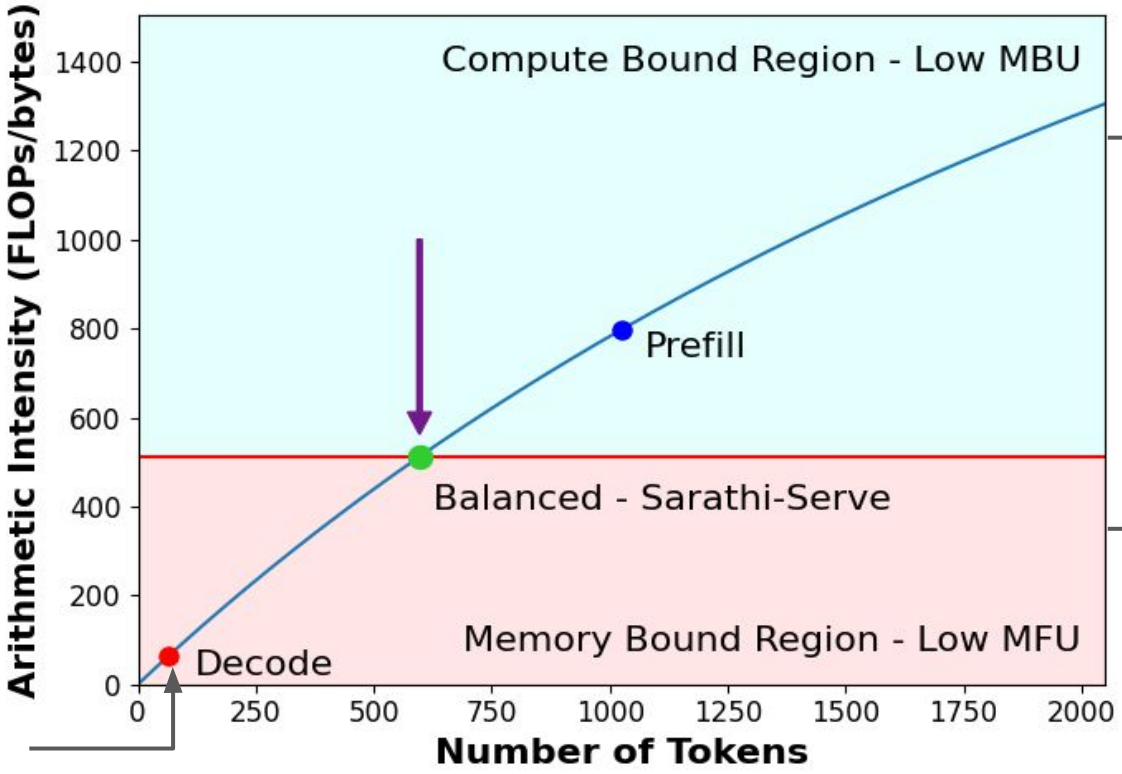


# Key Insight

Prefill computation can be done at a marginal cost with careful batching 💡

# 🔎 Observation: Arithmetic Intensity Slack




Latency dominated by compute

Grows linearly with num tokens

Latency dominated by weight fetch time

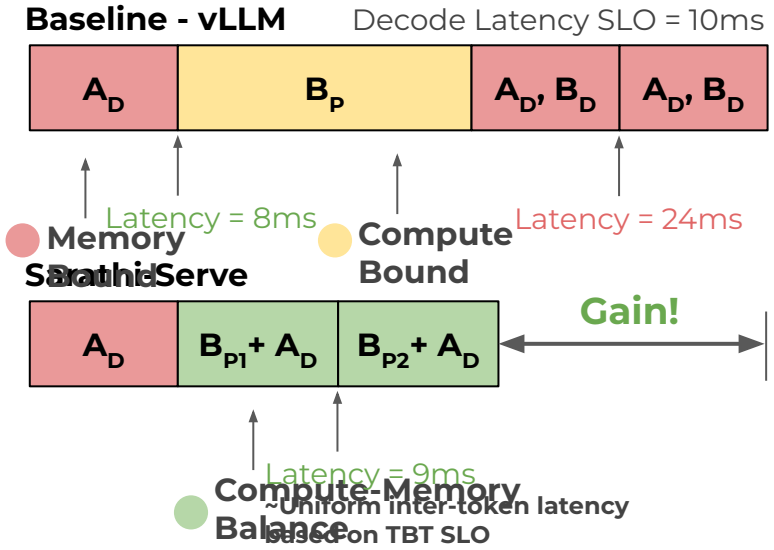
Independent of num tokens

Constrained due to memory overhead in decode phase

# 🧠 Stall-free Batching

## Key Idea

🔪 Split large prefills into smaller chunks – just enough to consume the leftover compute budget in decode batches



# Demo 

# Evaluations 🧮

# 📐 Background: Performance Metrics

🚌 **Time to first token (TTFT):** Time required for the first token to show up from the time user submits a request

🏎️ **Time between tokens (TBT):** Latency between each output token

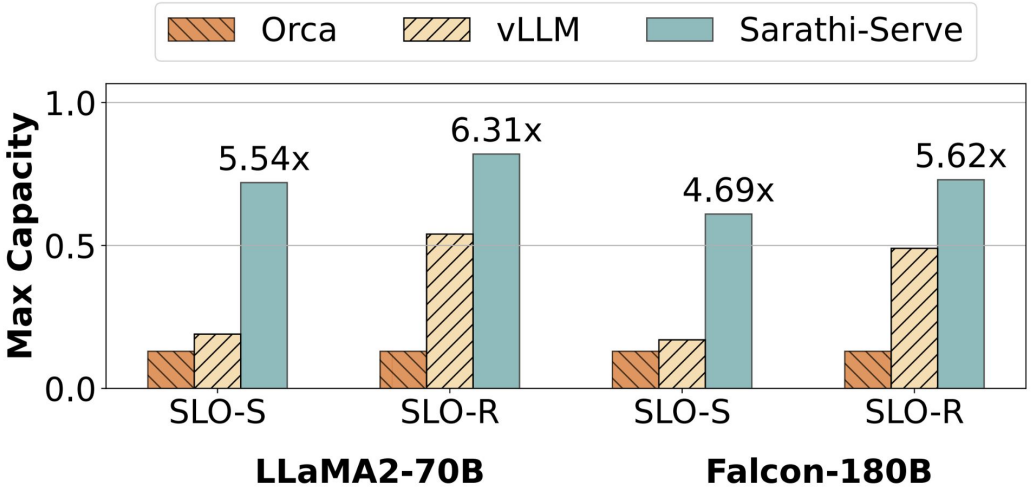
🏋️ **Capacity:** Maximum QPS that can be served while satisfying latency SLOs

# Serving Capacity under SLOs

adapt using different chunk sizes

## Setup

ShareGPT4 trace on on A100 GPUs with strict (S) and relaxed (R) latency SLOs




**5-6x higher capacity** 🎊

# 🗒️ Summary

😨 **Problem:** State-of-the-art systems sacrifice decode latency to achieve higher throughput

💡 **Key Insight** - Low arithmetic intensity of decodes allows for adding compute intensive prefills with negligible decode latency cost

🧮 **Key Results** - We achieve optimality in both latency and throughput simultaneously leading up to 6x higher capacity under SLO constraints

🎉 **Industry Adoption** - Available in all major serving frameworks and more.

vLLM CentML anyscale

Paper GitHub Use in vLLM