

# CS 744: LLUMINIX

Shivaram Venkataraman

Spring 2025

# ADMINISTRIVIA

Grading updates → Midterm 1 out!  
→ Check ins → Canvas soon!

Midterm 2: Thursday!

Poster session: May 1st  
– Details on Piazza

# LLM INFERENCE?

ChatGPT ▾

↳ Big workload

↳ Ideas through the semester

What can I help with?

Ask anything



Search



Reason



# CHARACTERISTICS

Diversity of LLM requests

high priority  
vs.  
normal

Sequence length

Output length

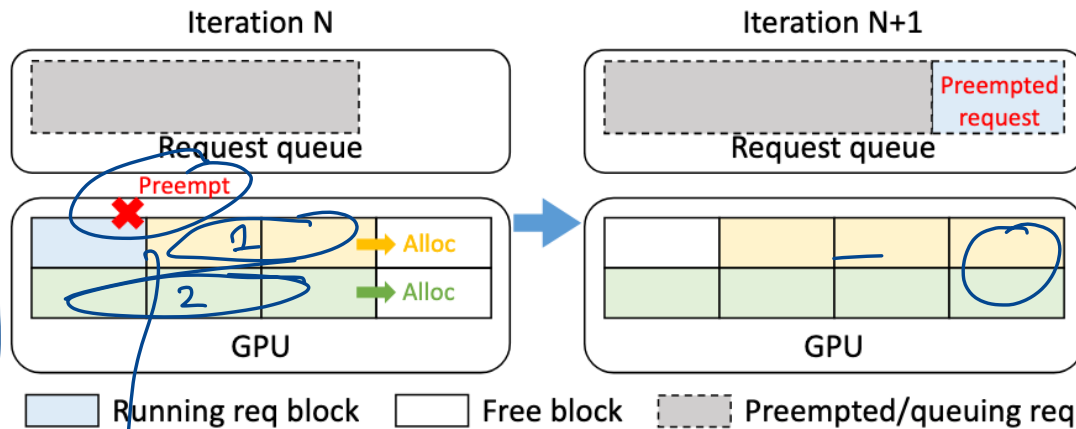
SLOs

inherently  
hard to  
predict  
1 to ???

Continuous batching

Dynamic  
memory  
allocation

4k, 8k ... 128k etc.



out of memory at this point

# CHALLENGES

Pre-empt  
dependencies  
Isolation → many requests same time

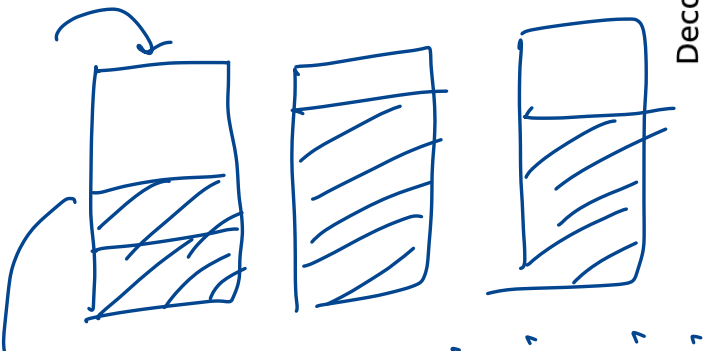
isolate the effect of  
other requests on this

## Fragmentation

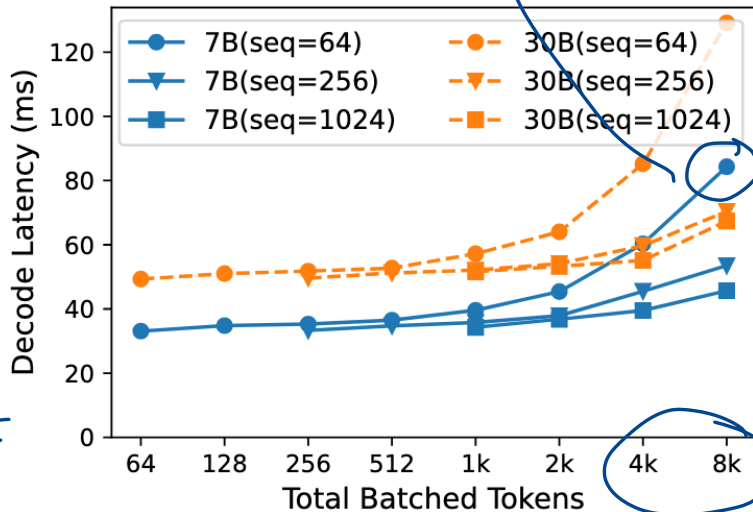
requests  
on which  
you

## Priorities

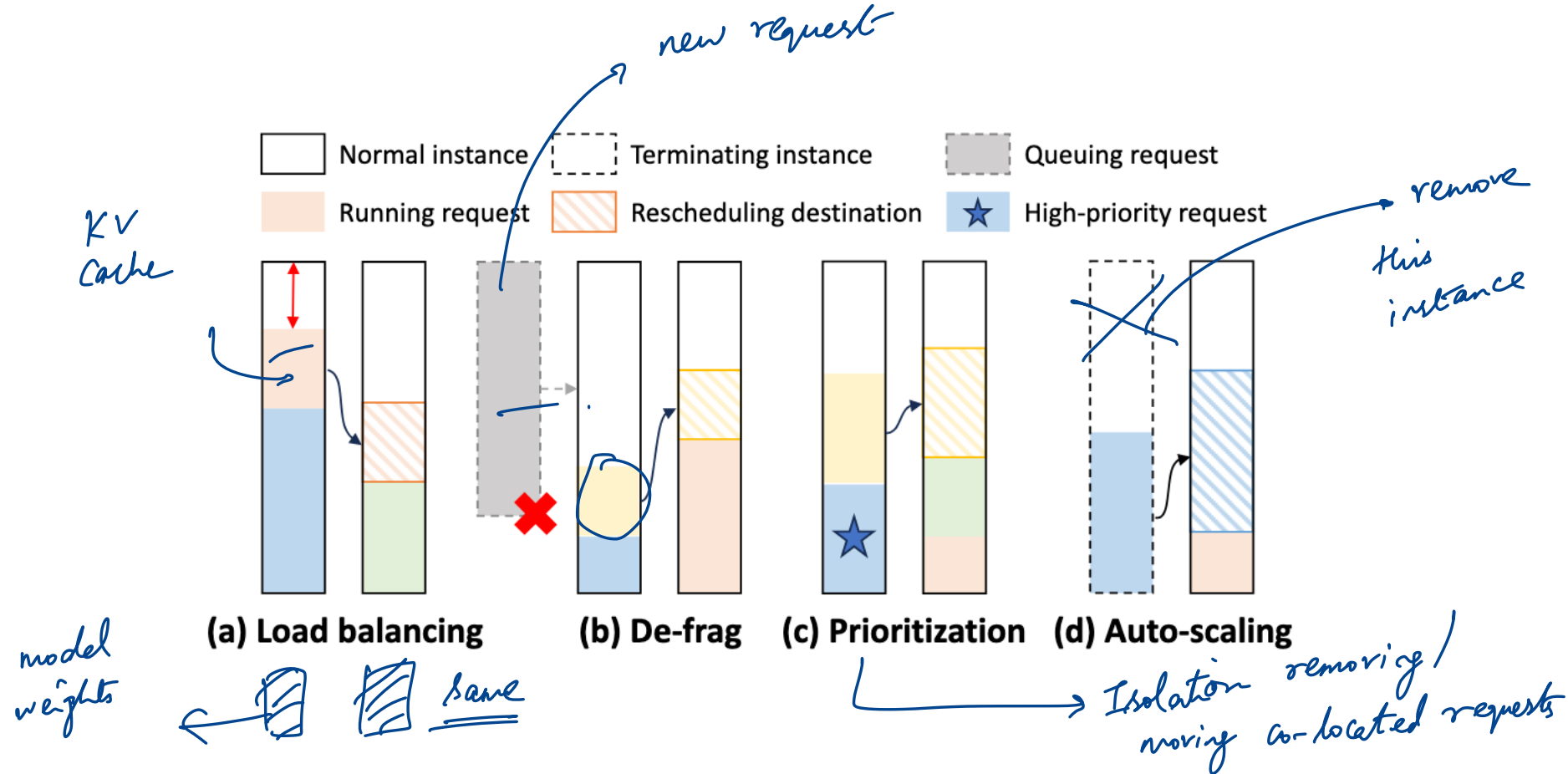
low priority  
vs.  
high priority



slow down  
due to lack  
of isolation



# APPROACH: REQUEST RE-SCHEDULING



# LIVE MIGRATION

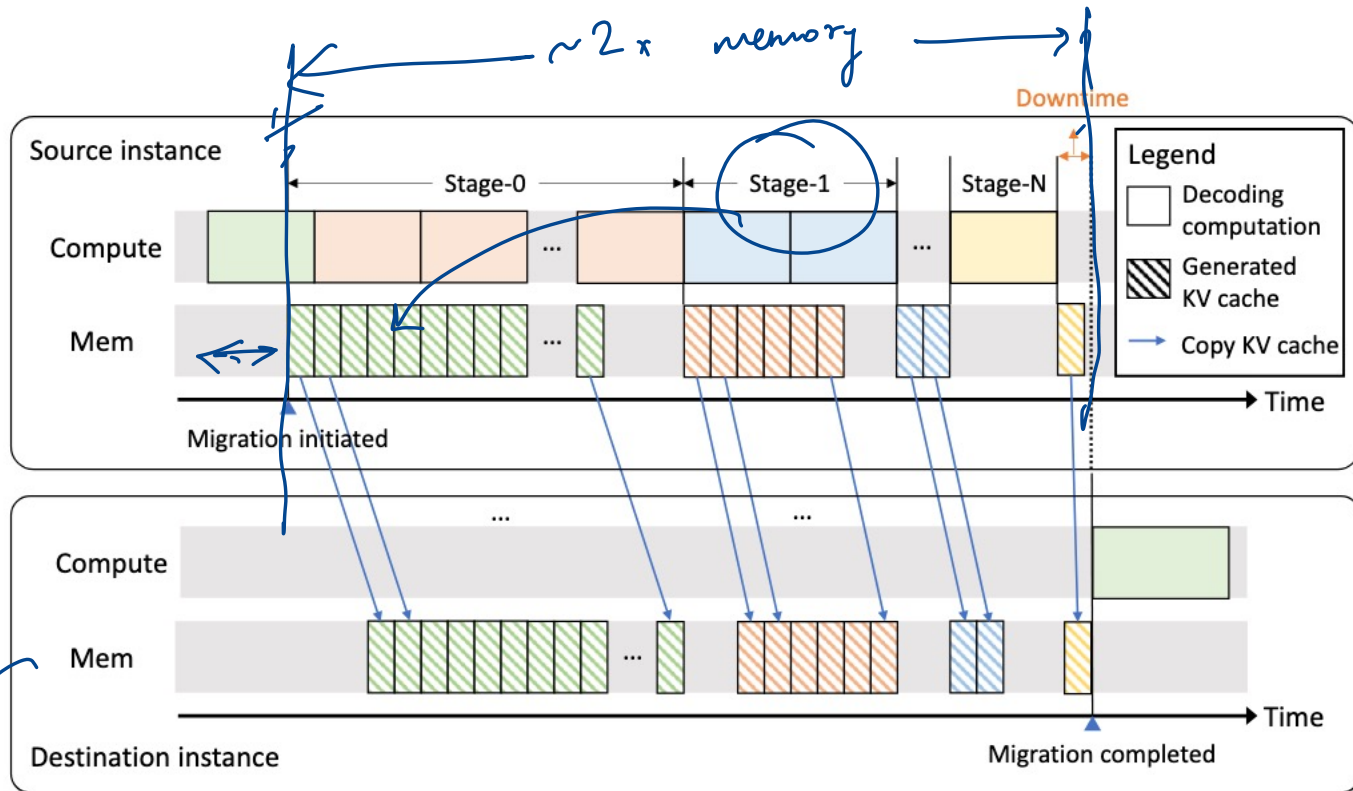
→ VM migration

Goal: Minimize  
downtime

Key Insight:

KV caches are  
"append" - only

↳ Start the copy  
of KV cache in  
background



→ Pipelining compute with migration

# TWO-LEVEL SCHEDULING

Global scheduler

Dispatch new requests

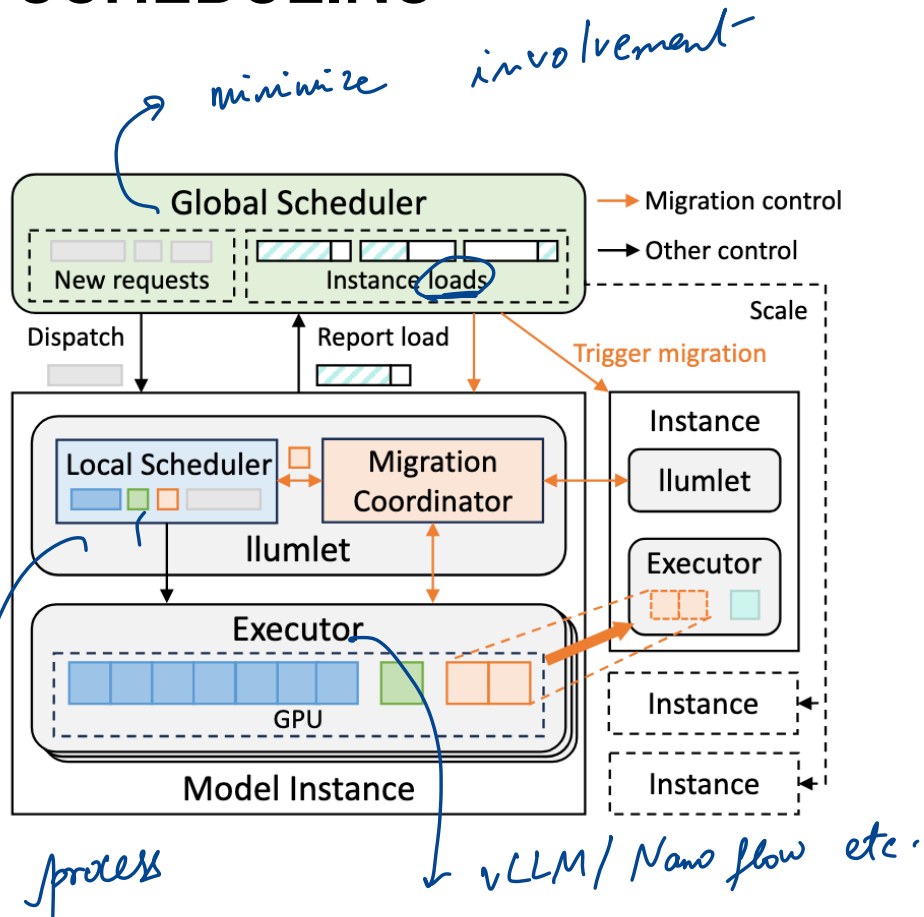
Trigger migration

*select source ↔ destination*

Llumlet

Memory load

Select requests

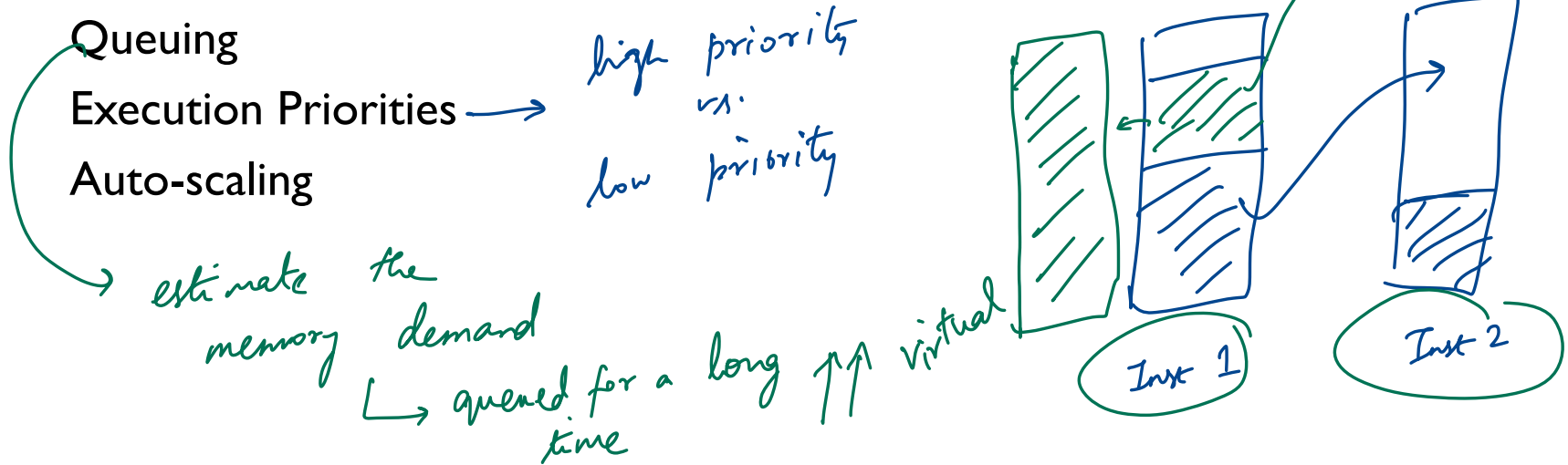


# VIRTUAL USAGE

"Virtual usage" → one metric used to make decision

Unify load balancing, fragmentation, prioritization etc.

Intuition: make the instance virtually overloaded!



# POLICIES

Dispatch requests to “freest” instance

*all pairs  
are equally good*

→ Migration: periodically select source (lowest free) and dest (highest)

Auto-scaling: maintain average freeness within a range

*[20, 80] %*

*new machine to start moving load there*

# SUMMARY

LLM Scheduling → Important new workload

New challenges based on workload properties

Lluminix: Scheduler that reschedules requests at runtime

- Low overhead migration

- Two level scheduling

- Virtual usage based policies



# DISCUSSION

<https://forms.gle/7Cpk6oDYA3UpmuIRA>

What could be some challenges with using Luminix if we want to serve many different kinds of models simultaneously (Llama-3B, 7B, 64B etc.)

↳ If you want all models on all machines → memory use !!

each GPU has part of every model

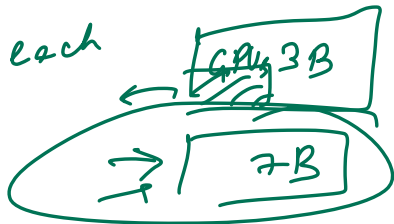
---

hardware homogeneous

→ Scheduling → Dispatch a request 3B  
request 7B

↳ how do I prioritize / schedule them

diff instances of each model



Migration KV-cache & model weights?

↳ limit candidates for migration

If you wanted to apply ML to improve Lluminox, what are some potential areas you would explore?

→ Predict output length  
(use ML model → Past prompts  
→ Past responses)

→ should be small?  
should look at KV cache

→ time of day, who the user is, → auto scaling

↳ Traffic  
→ migrate → better bin packing?

# NEXT STEPS

Next week schedule

Thu: Midterm 2