Towards the Next-Generation AI Agent Infrastructure

Yiying Zhang - GenseeAI and UCSD

GenseeAI

UCSDCSE
Computer Science and Engineering

About Yiying Zhang

- Associate professor at UCSD CSE, leading the WukLab (wuklab.io)
- Founder and CEO of GenseeAI (gensee.ai), an agent infrastructure startup
- 15 years of experience in systems and infrastructure
- ~5 years in SysML, recent years focusing on LLM serving and agents

- Interested in joining GenseeAI as summer intern or founding engineer?
  - email yiying@gensee.ai


AI AGENTS - AGENTIC APPLICATIONS
User Input
LLM
Tools
HITL
Web Explorer
www
Weather API
Search API
Maths Library
RAG API
RAG
Observation
AI Agent Output
fter an Action is completed,
he Agent enters the Observation
tep.
rom Observation step Agent shares a
hought;
f a final answer is not reached,
he Agent cycles back to another
ction in order
o move closer to a Final Answer.
Final Answer Is Reached
COBUS GREYLING & AI

The Status of AI Agent

Easy to demo, extremely hard to productionize

A large majority of organizations have deployed less than a third of their GenAI experiments into production

Deloitte.

Avg Time Nine Months – – Prototype to Production

Time to Develop AI Prototype to Production
Percentage of Respondents

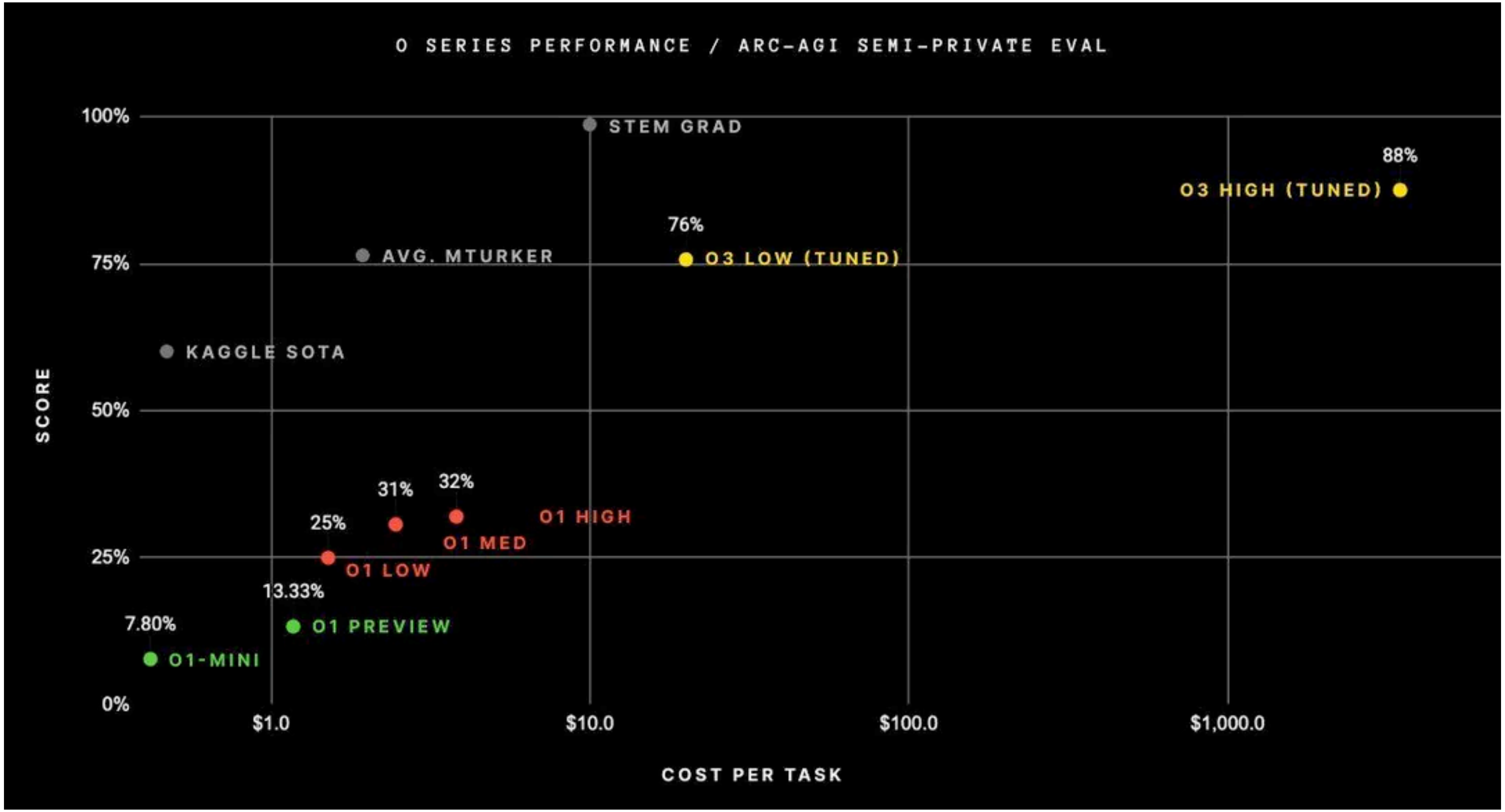
Gartner.

5x more time to productionize [AI Refinery] than to demo

accenture

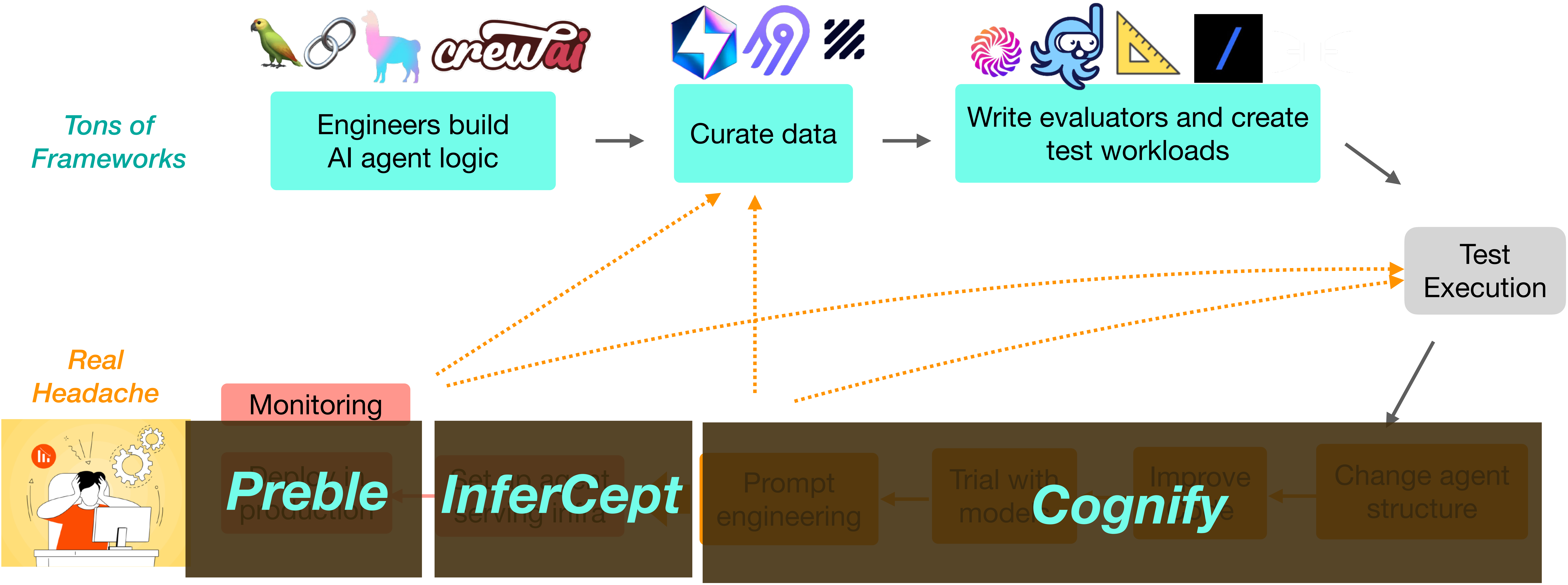
The Status of AI Agent

Rising inference cost, esp. as agent involves more model calls



https://arcprize.org/blog/oai-o3-pub-breakthrough

Today's AI agent development and deployment life cycle



Today's Talk

Cognify [arxiv'25]: AI agent optimizer

Preble [ICLR'25]: long & shared prompt serving

InferCept [ICML'24]: compound LLM serving

Cognify: Multi-Facet AI Agent Optimization

Under submission, open source at https://github.com/GenseeAI/cognify

- AI Engineers spend a lot of effort tuning AI agents and workflows
- Cognify: aims to autotune AI agents and workflows with a small budget
- Key challenge: the search space is huge and proper searching needs $$
  - A simple 4-step workflow could need $168K and weeks to search brute force
- AdaSeek: adaptive hierarchical BO-based search
  - Uses $5 and 24 minutes to autotune the above workflow with 2.8x higher quality

"Cognify: Supercharging Gen-AI Workflows With Hierarchical Autotuning", Zijian He*, Reyna Abhyankar*, Vikranth Srivatsa, *Yiying Zhang* (* equal contribution)

Challenge: Large Search Space but Limited Budget

Assuming 3 LLM steps in an agent workflow

12 configurations in total, each with 3 options

Grid search requires: (3)^12 = 531,441 runs !!!

Assume workflow

- use gpt-4o: $10/1M tokens
- each execution output 10K tokens

➡ $53K

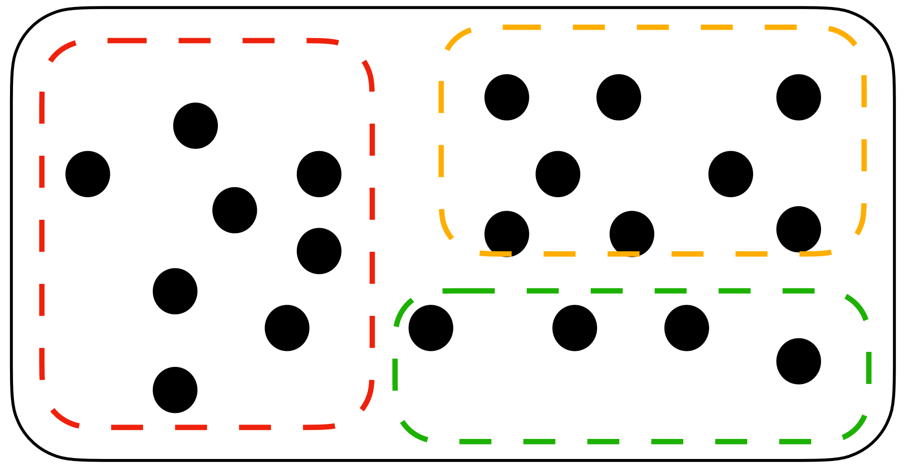
What's the best I can get with 128 trials ?

Our Approach: Adaptive Hierarchical Search

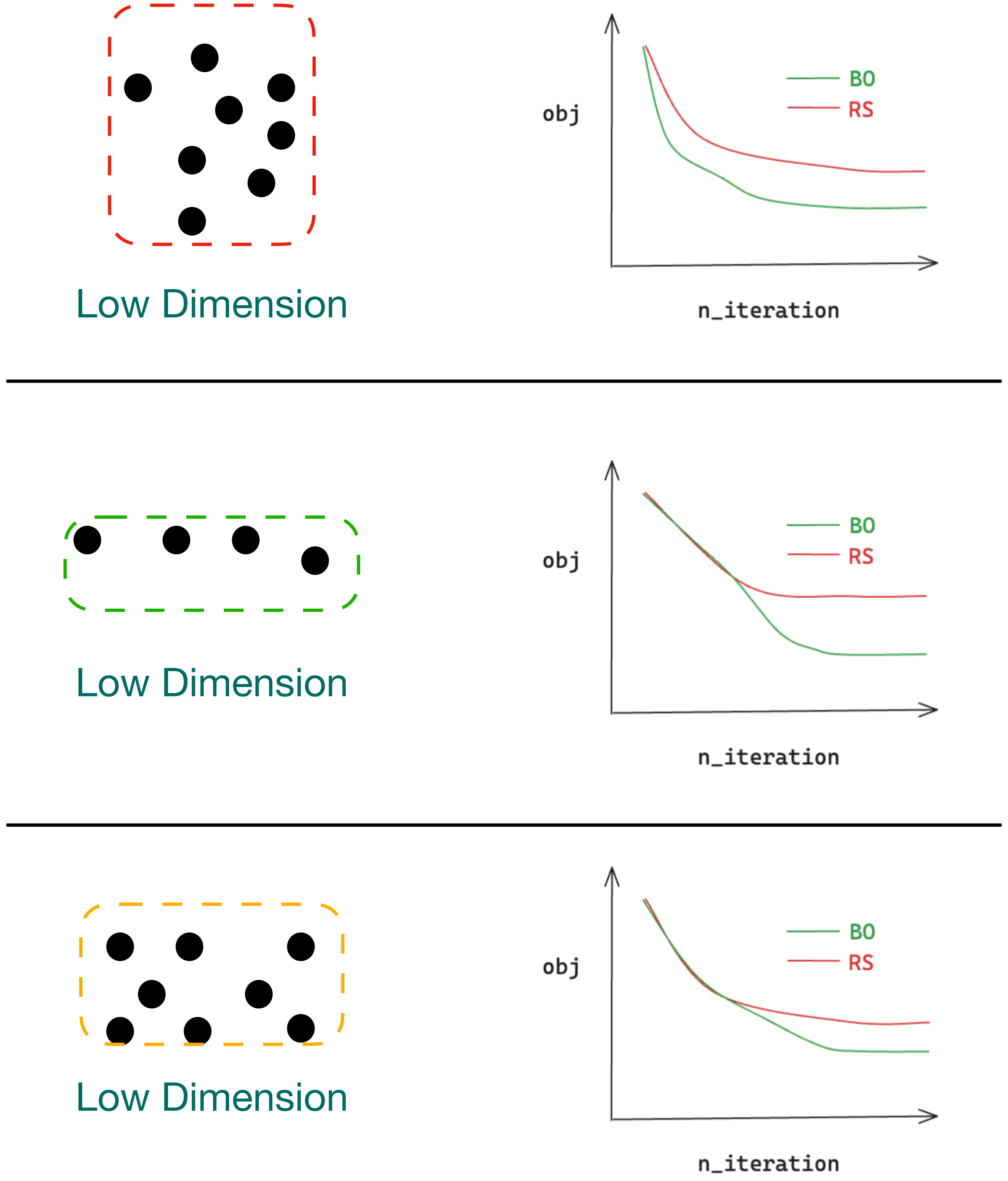
1. Hierarchical search with layered cog search space
2. Result-driven budget distribution
   A. Search space partition
   B. Search budget partition
3. Dynamic resource re-distribution

Cognify's Solution: Hierarchical Search

Flattened Search Space



High Dimension



Cognify: Organize tuning knobs (cogs) hierarchically

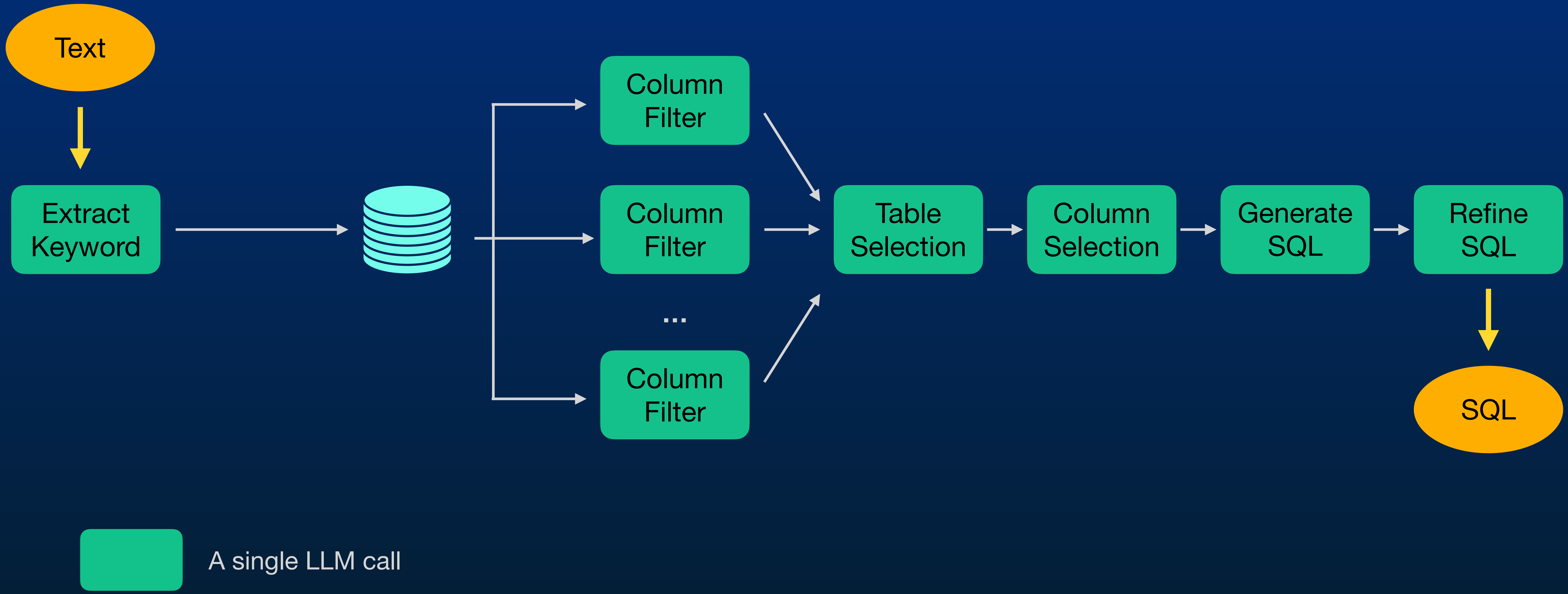
Top layer: Architecture Cogs	Task Decomposition	Task Ensemble
Middle layer: Step Cogs	Model Selection	Code Rewriting
Bottom layer: “Weight” (Prompt) Cogs	Few-Shot Example	Reasoning Prompts

Cognify’s Solution: Adaptive Search

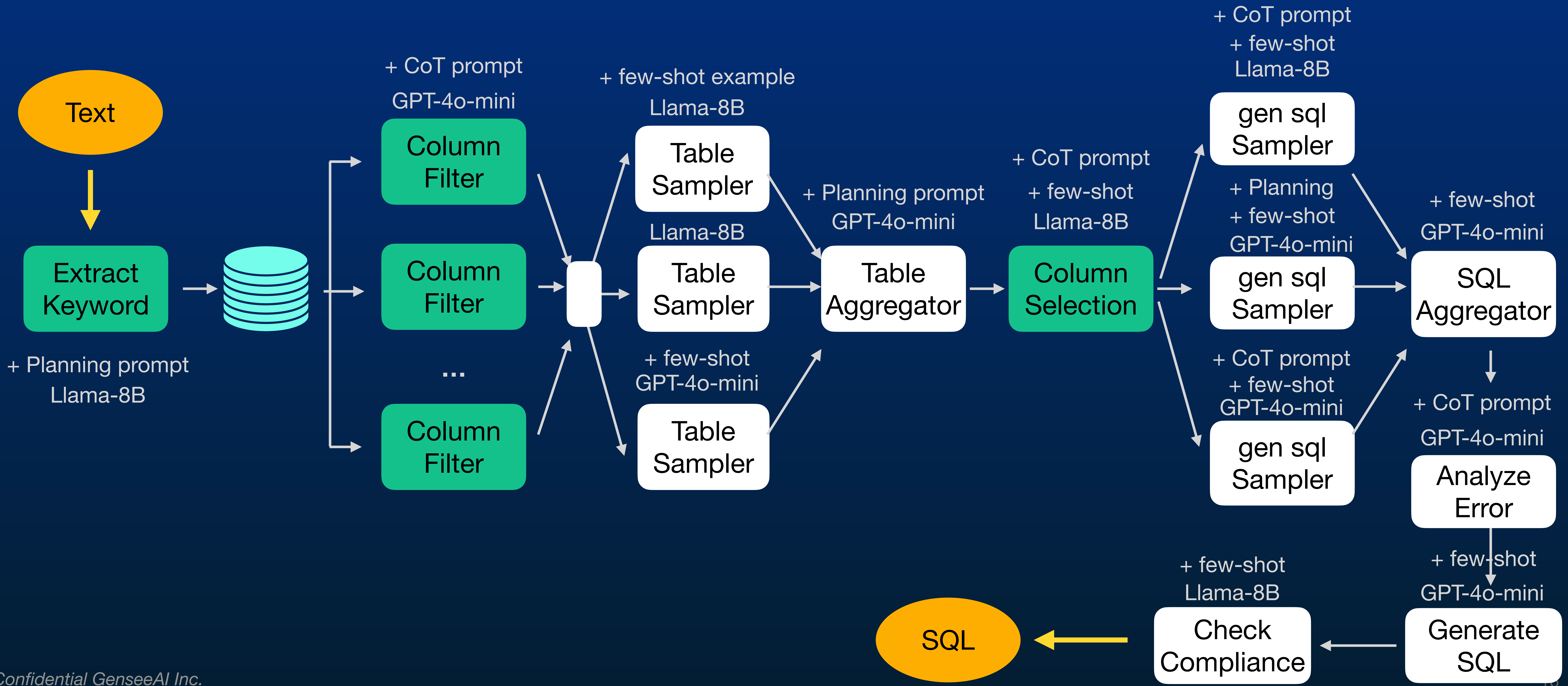
- Partition search budget across hierarchies according to layer complexity
- Direct search budget to more promising configurations using SH


Architecture Cogs
Step Cogs
Weight Cogs
Evaluate Workflow
Search Budget
(32 iterations)
Best Results

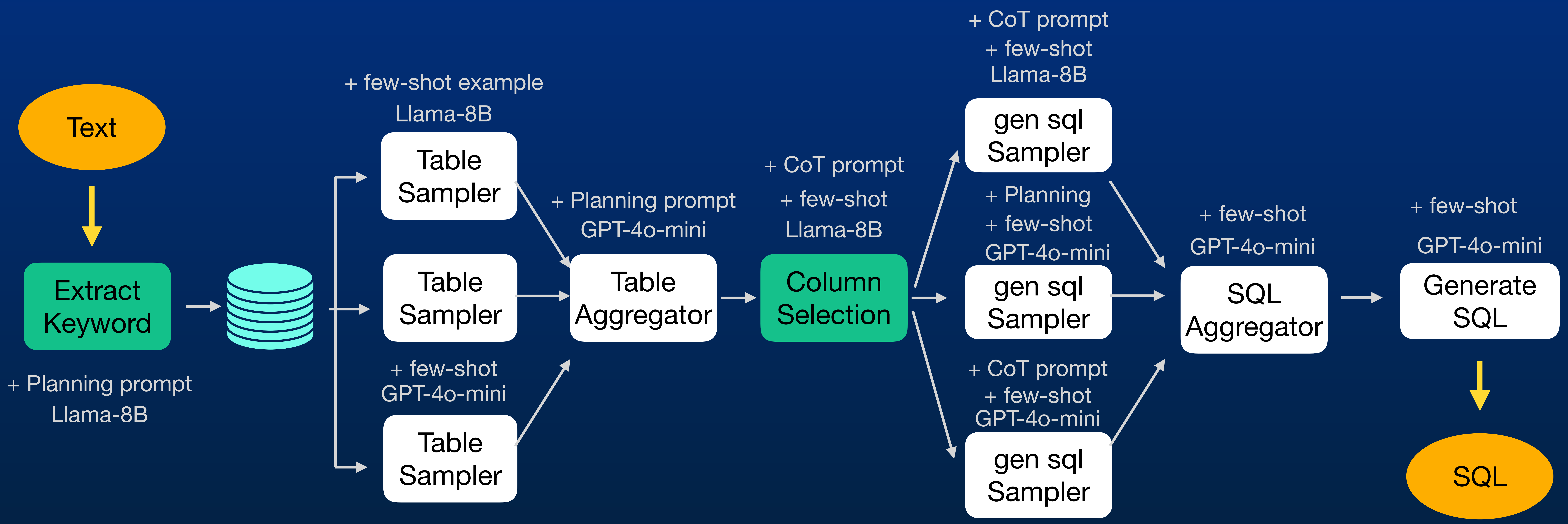
A real gen-AI workflow: text-to-SQL



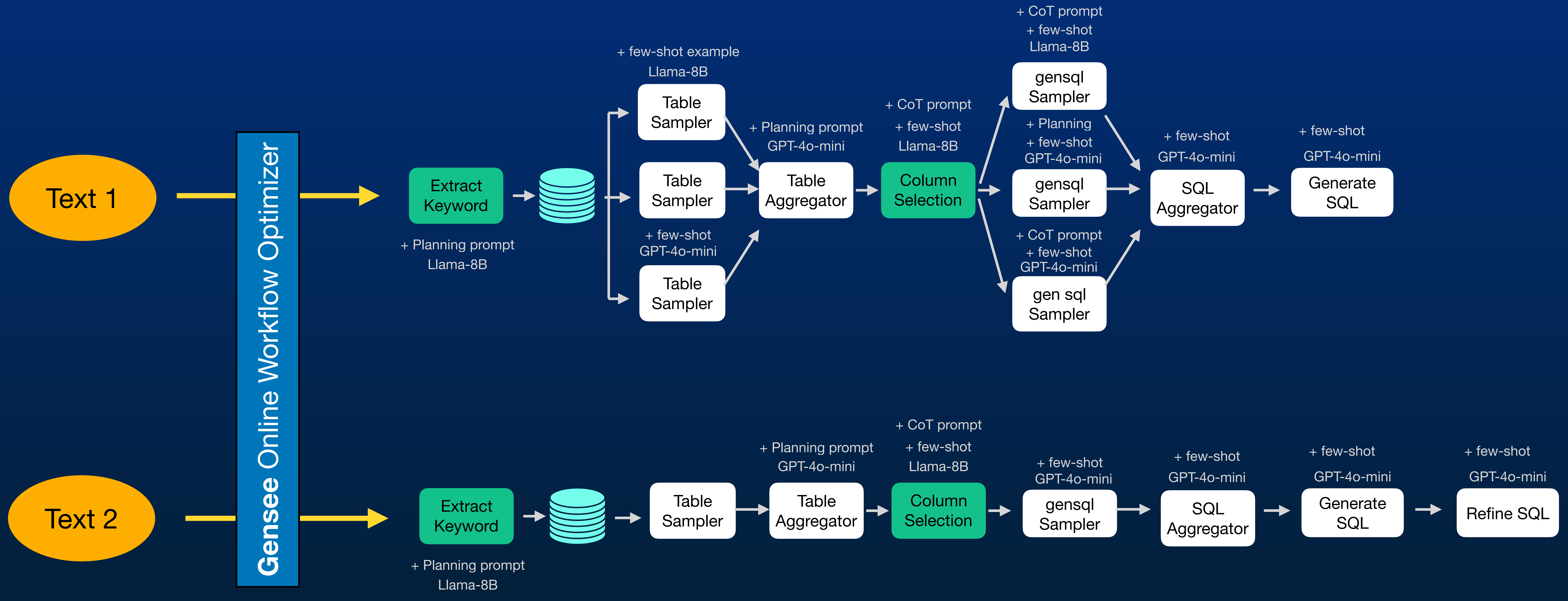
Gensee's optimized text-to-SQL (training+AutoML)



Another Gensee's optimized text-to-SQL (latency-oriented)



Gensee's online per-request optimization (serving)



Cognify Results

Six workloads: RAG-based QA, text-to-SQL, data visualization, financial analysis, code generation, BigBench

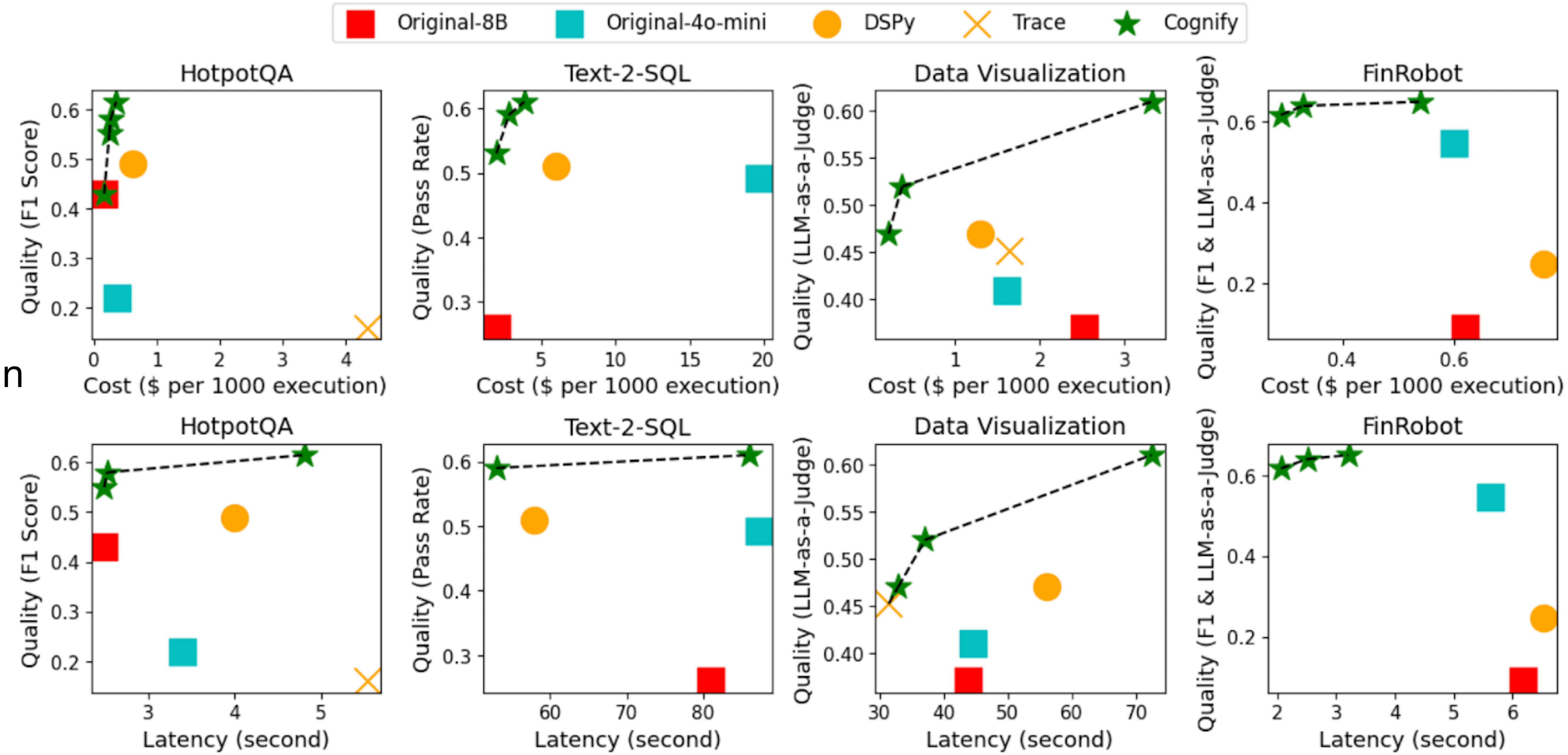
Up to 2.8x quality improvement

Up to 2.7x latency reduction

Up to 10x cost saving

Multiple choices on Pareto frontier

Low optimization costs: $1.5-$10, 8-42min (roughly half of DSPy)



Cognify Takeaways

- Production-grade agents require manual input to incorporate business logic
- But manual efforts for tuning production agents can be avoided
- Cognify is the first production-ready autotuning tool for AI agents
- Please join our discord and star our repo to learn more!
- https://github.com/GenseeAI/cognify
  https://discord.gg/8TSFeZA3V6
-

Today’s Talk

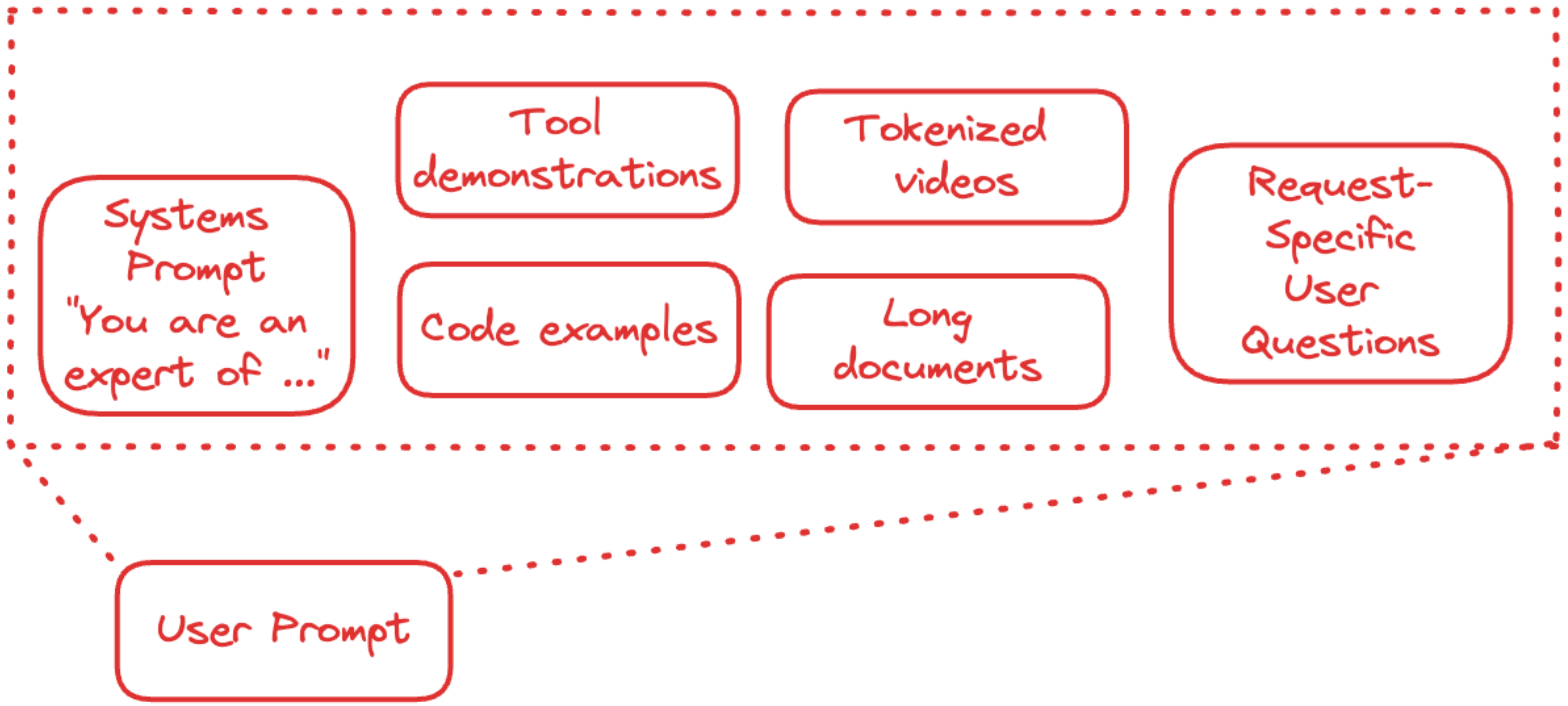
Cognify [arxiv’25]: AI agent optimizer

Preble [ICLR’25]: long & shared prompt serving

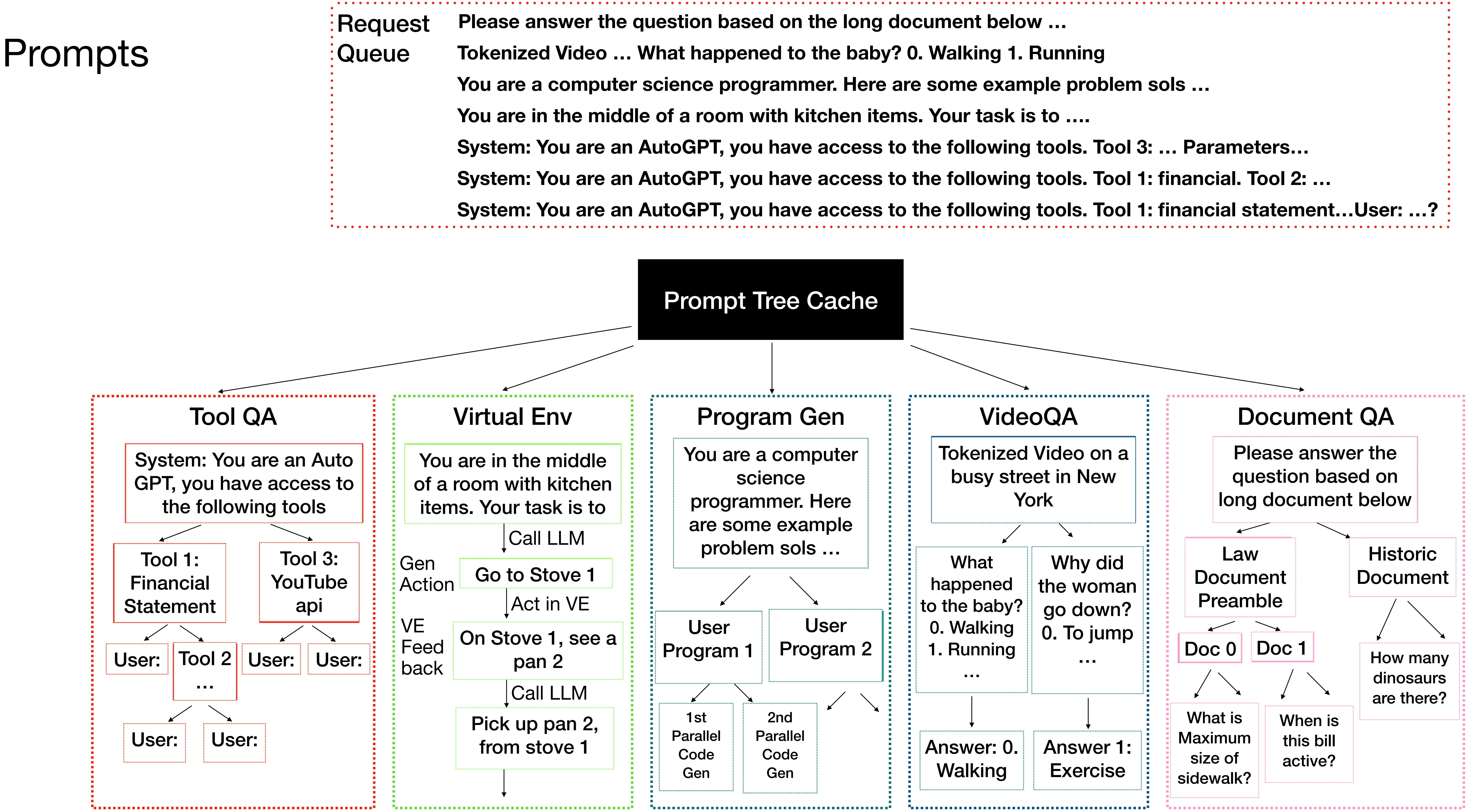
InferCept [ICML’24]: compound LLM serving

It’s all about prompting

- Agent prompts are more than just a simple question

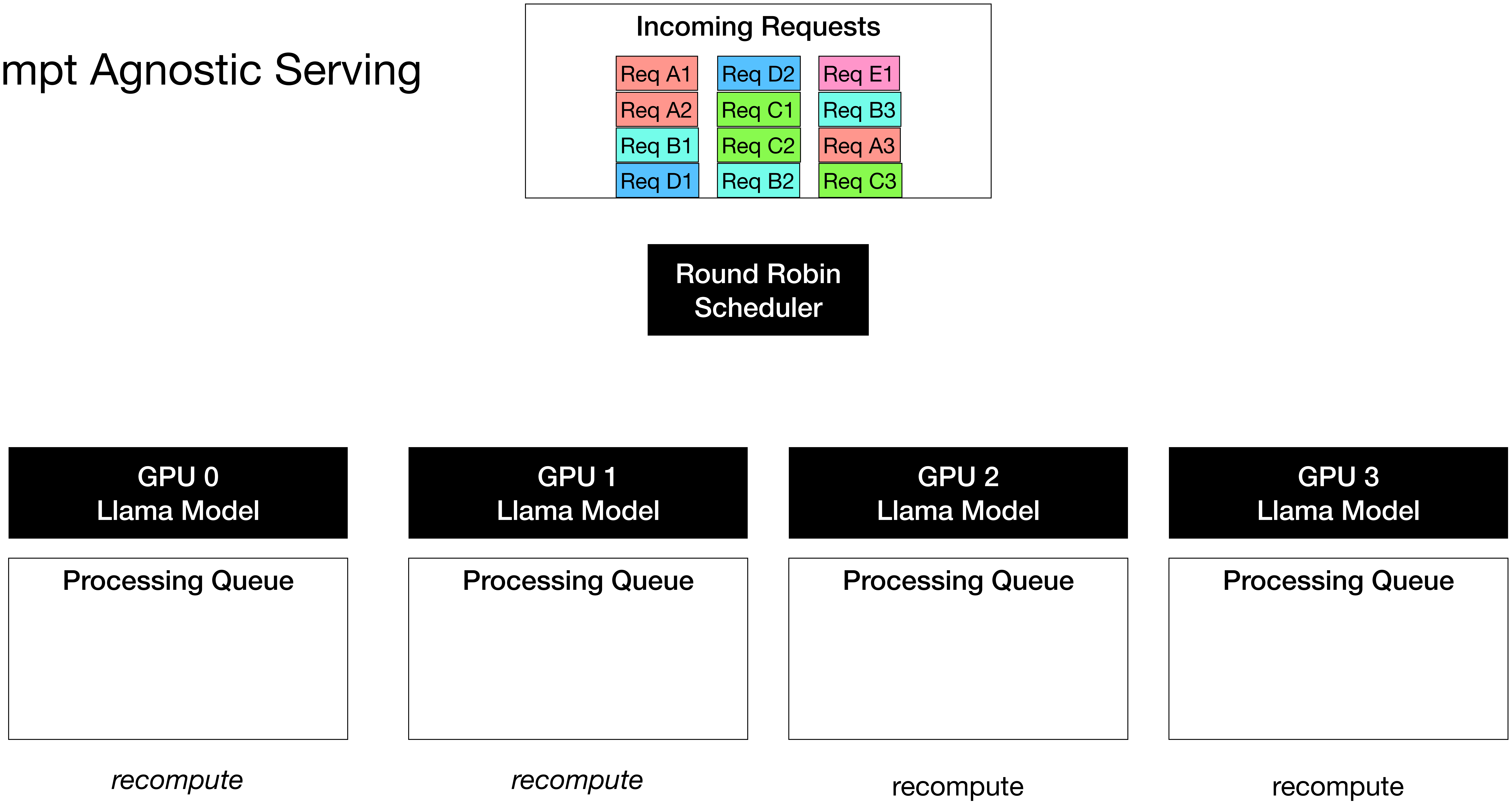


https://towardsdatascience.com/how-i-won-singapores-gpt-4-prompt-engineering-competition-34c195a93d41



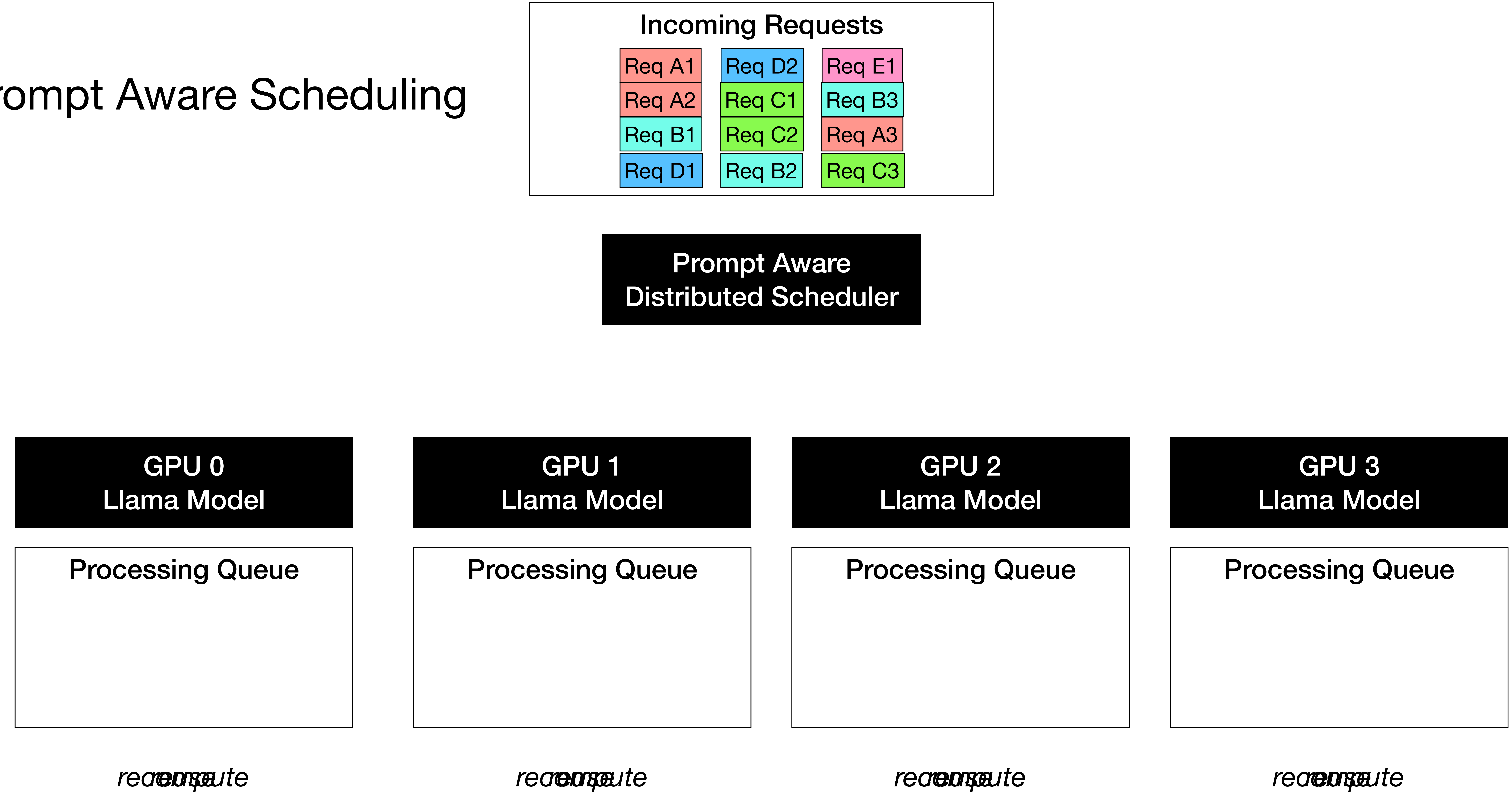
Prompts
Request Queue
Please answer the question based on the long document below ...
Tokenized Video ... What happened to the baby? 0. Walking 1. Running
You are a computer science programmer. Here are some example problem sols ...
You are in the middle of a room with kitchen items. Your task is to ....
System: You are an AutoGPT, you have access to the following tools. Tool 3: ... Parameters...
System: You are an AutoGPT, you have access to the following tools. Tool 1: financial. Tool 2: ...
System: You are an AutoGPT, you have access to the following tools. Tool 1: financial statement...User: ...?
Prompt Tree Cache
Tool QA
System: You are an Auto GPT, you have access to the following tools
Tool 1: Financial Statement
Tool 3: YouTube api
User:
Tool 2 ...
User:
User:
User:
User:
Virtual Env
You are in the middle of a room with kitchen items. Your task is to
Call LLM
Gen Action
Go to Stove 1
Act in VE
VE Feed back
On Stove 1, see a pan 2
Call LLM
Pick up pan 2, from stove 1
Program Gen
You are a computer science programmer. Here are some example problem sols ...
User Program 1
User Program 2
1st Parallel Code Gen
2nd Parallel Code Gen
VideoQA
Tokenized Video on a busy street in New York
What happened to the baby? 0. Walking 1. Running ...
Why did the woman go down? 0. To jump ...
Answer: 0. Walking
Answer 1: Exercise
Document QA
Please answer the question based on long document below
Law Document Preamble
Historic Document
Doc 0
Doc 1
What is Maximum size of sidewalk?
When is this bill active?
How many dinosaurs are there?

Prompt Agnostic Serving



All Requests are recomputed

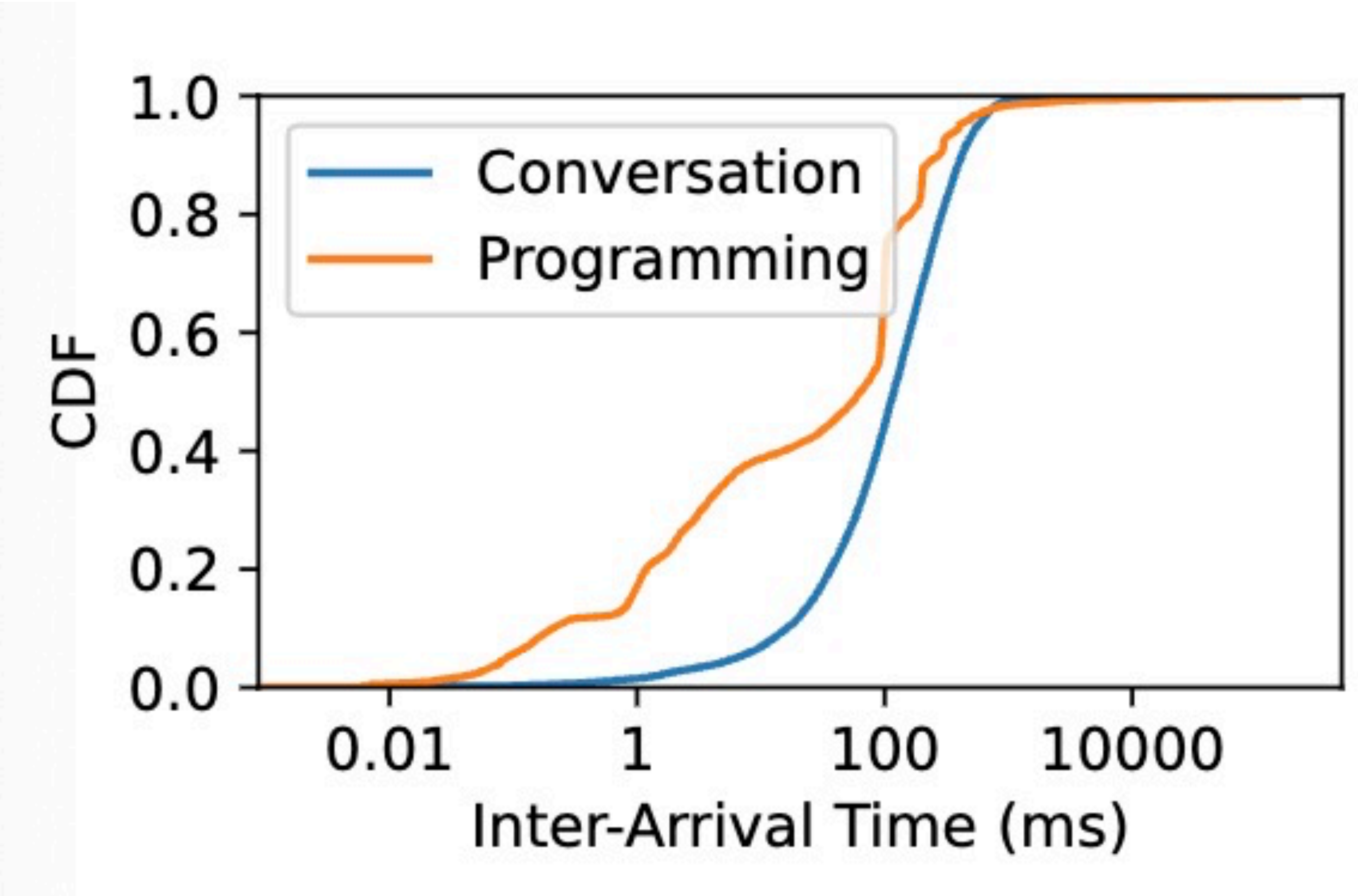
Prompt Aware Scheduling

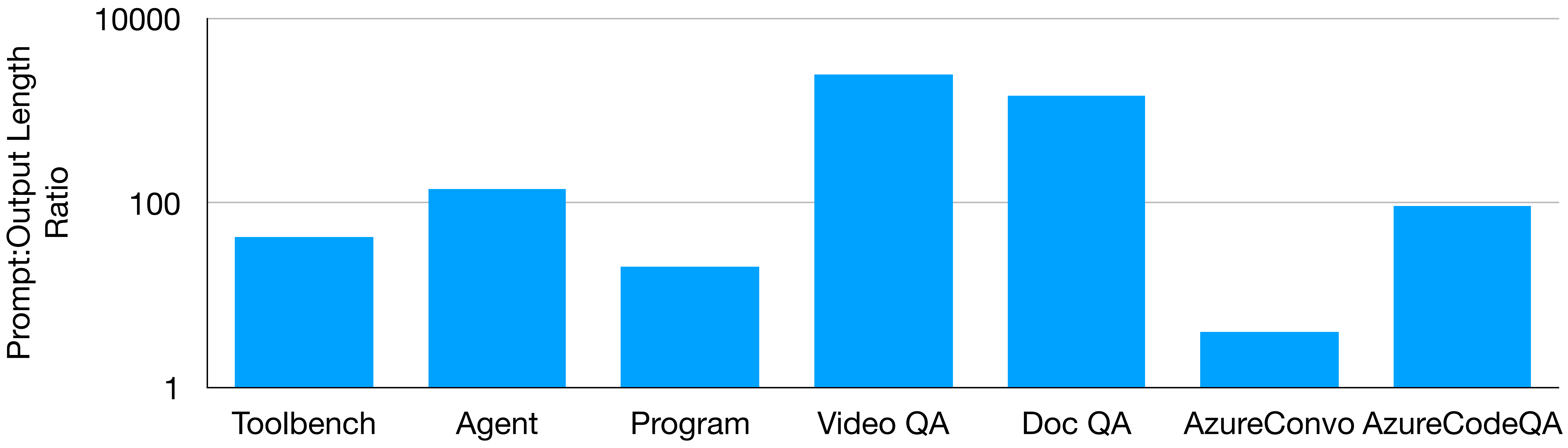


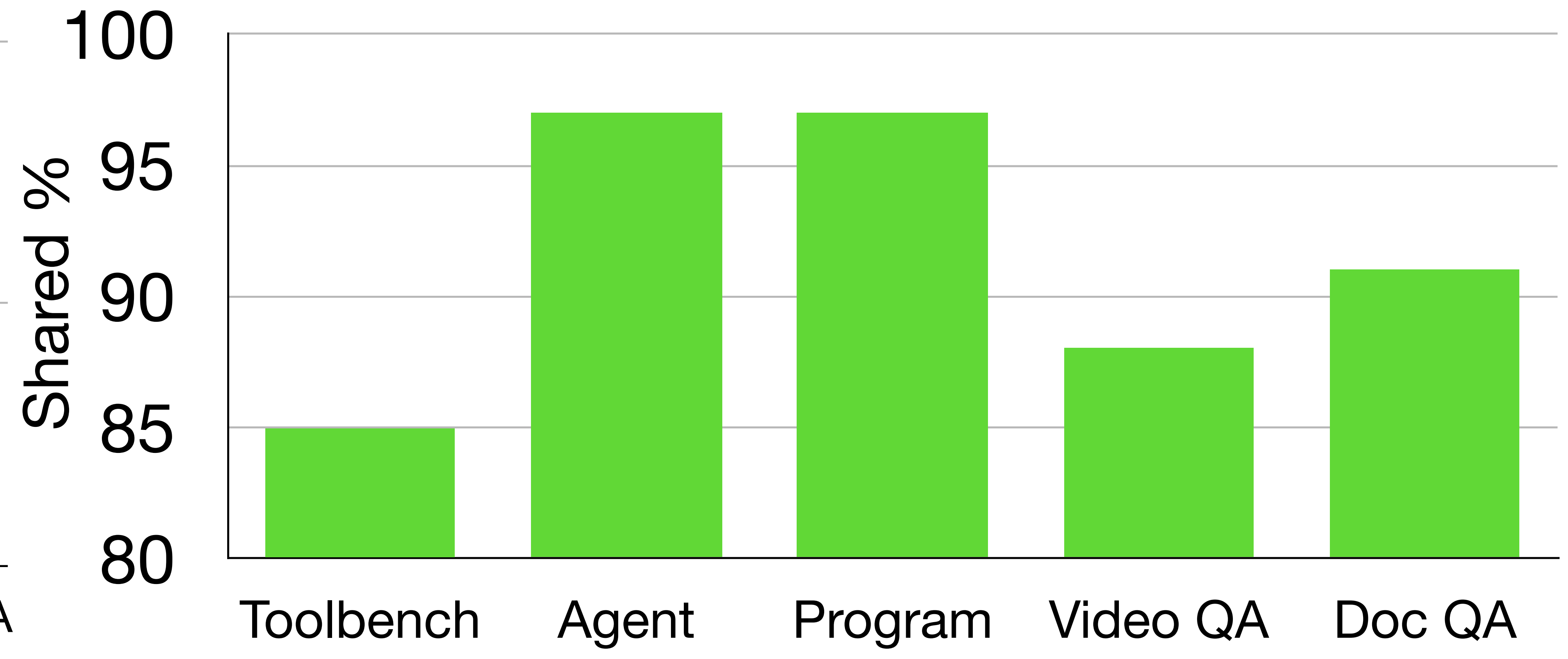
Faster completion due to prefix reusing

How are real-world prompts like?
- A study of prompts from systems perspective

- Studied 5 workloads and 1 real LLM request trace
  - tool use, embodied agents, program generation, video QA, and long document QA
  - 2023 Azure LLM Inference Trace
- Long prompts followed by short output
- High sharing degree
- Variation in request load





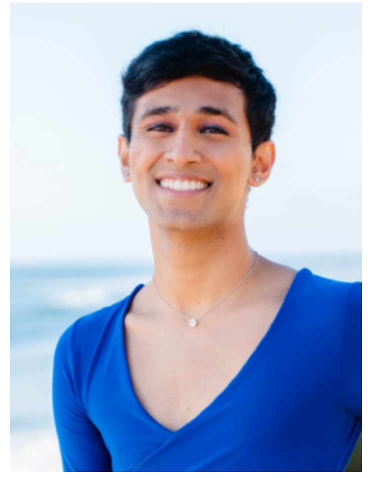


Preble: Distributed LLM Serving for Long and Shared Prompt

ICLR'25, open source at https://github.com/WukLab/preble

- A distributed serving system targeting long and shared prompts
- Co-designs prefix sharing and load balancing
- Centers around a new **E2** distributed scheduling algorithm
- Two-level scheduler for scalability
- Optimizes avg and p99 latency (up to 14.5X and 10X improvement over SoTA)

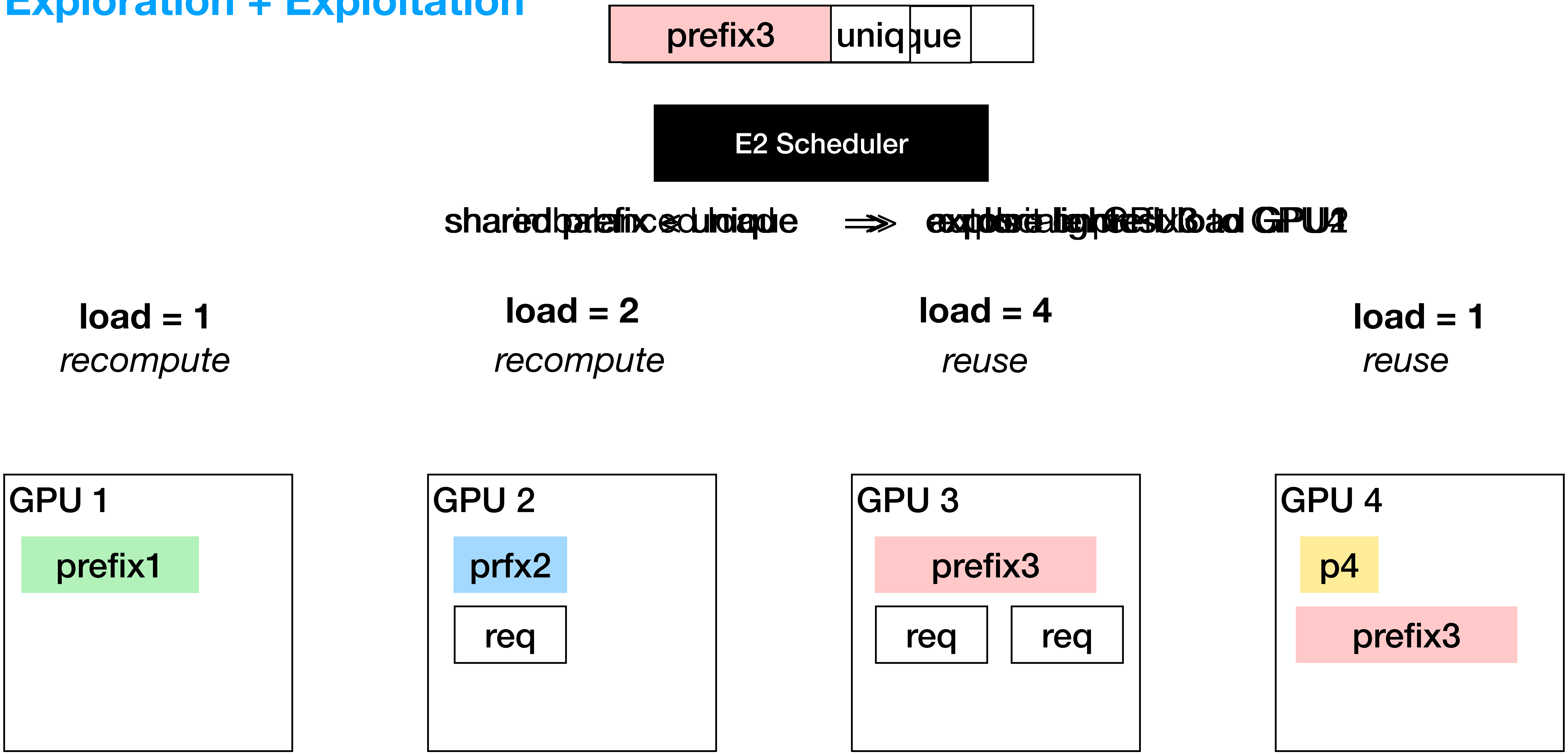





"Preble: Efficient Distributed Prompt Scheduling for LLM Serving", Vikranth Srivatsa*, Zijian He*, Reyna Abhyankar, *Yiying Zhang, arxiv:2407.00023* (* equal contribution)

Preble's E2 Scheduling:

Exploration + Exploitation



Exploit and Explore Heuristic

Greedy Exploit

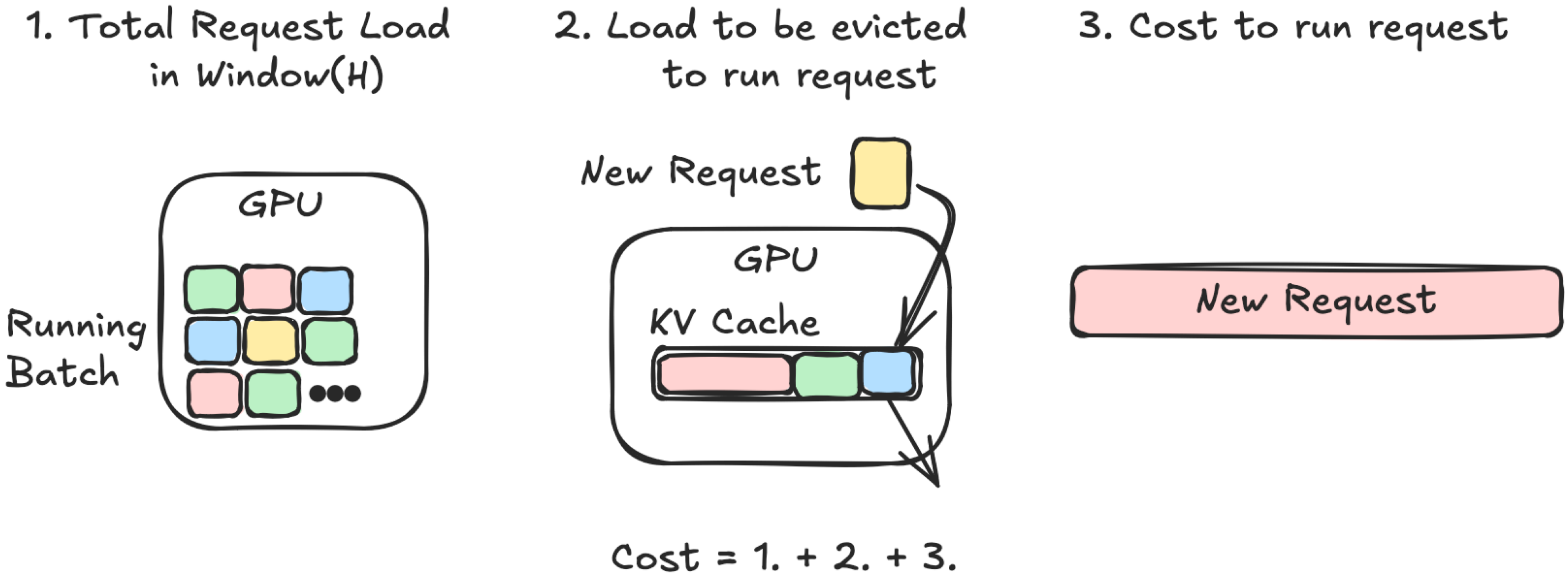
- Length of the prefix is proportional to compute
- Shared *prefix > rest of prefix* then exploit

Intuition on the computation:

- New Server: Shared + unique portion of prefix cost
- Existing Server: unique portion of prefix cost + eviction cost(unique # tkns)

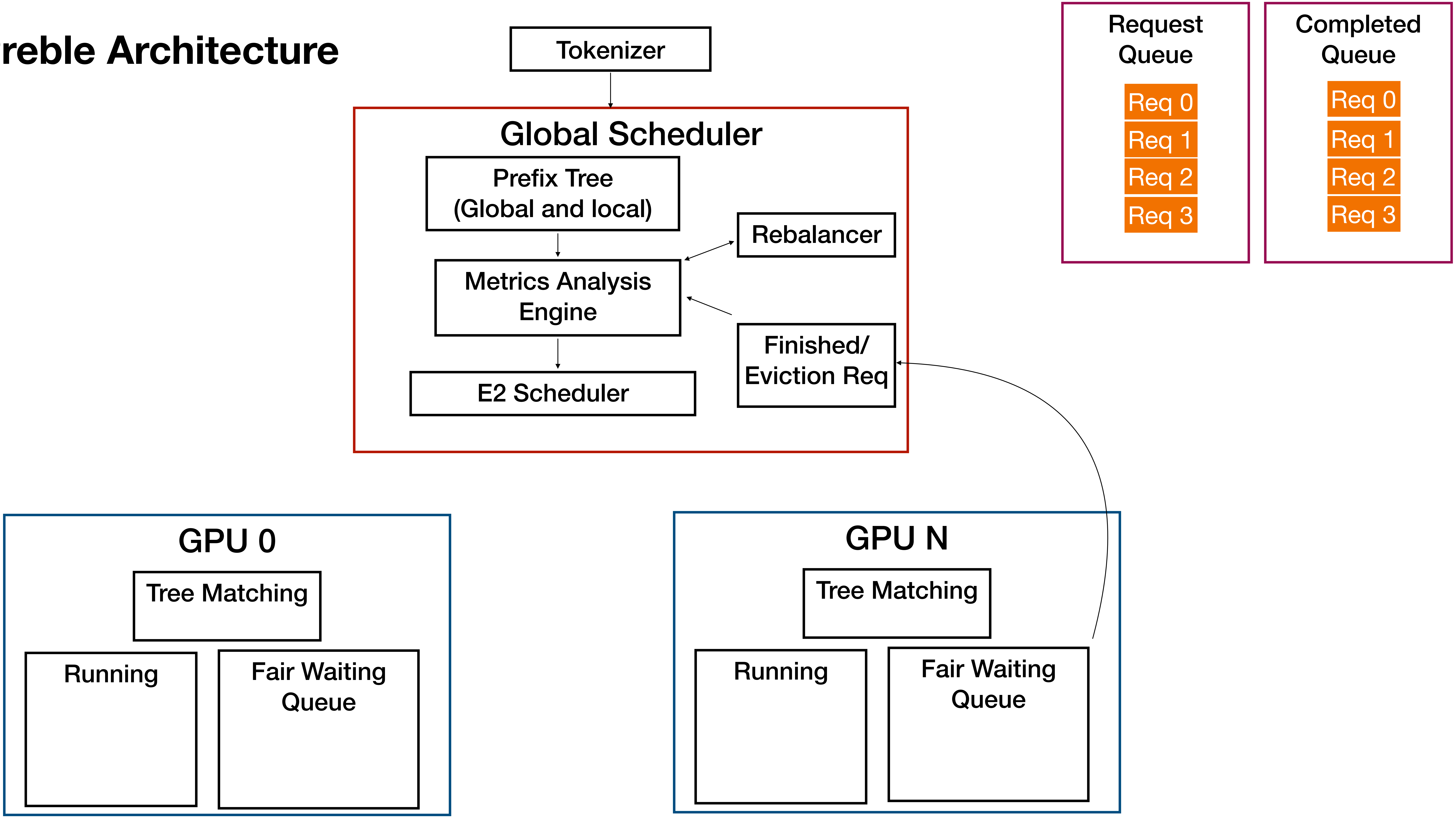
The amount of recomputation saved is larger than new computation

How do you Explore?
Prefix aware exploration



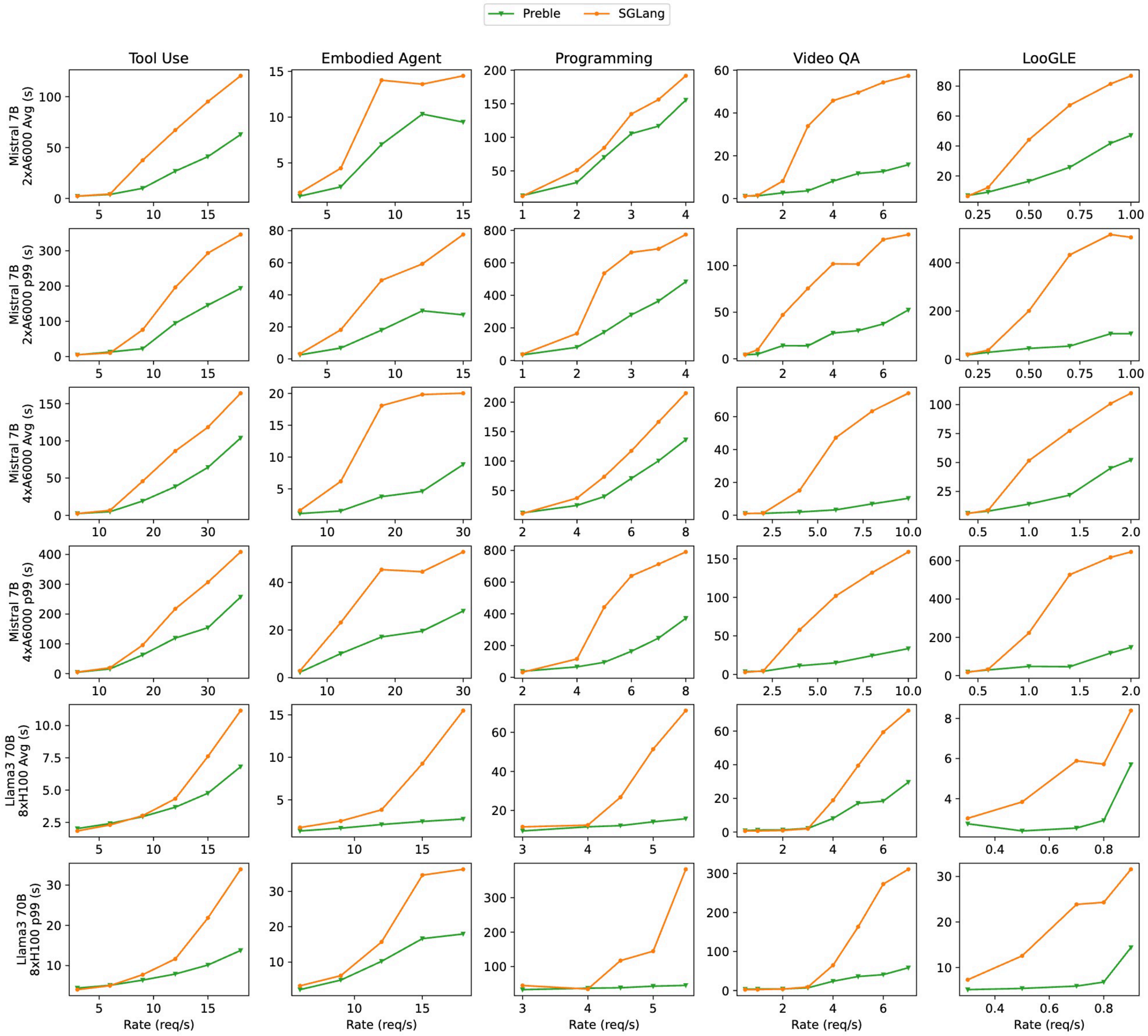
Calculate a load cost for each GPU and pick the min one

Preble Architecture



Initial Results

- Mistral-7B on 2 A6000
- Mistral-7B on 4 A6000
- Llama3-70B on 8 H100
- Five workloads
- cmp SGLang, Oracle partition
- avg & p99 req latency, RPS
- 1.5X to 14.5X on avg
- 2X to 10X on p99



Impact in Real World

Paper released May 2024

Cheaper **Prefix Sharing** in **Real World Production** systems

ANTHROP\C August 2024

OpenAI October 2024

Gemini October 2024

Moonshot AI MoonShot AI

- Directly inspired by Preble

OpenAI Open AI

- Heuristic based

Preble Takeaways

- LLM Serving is getting **more expensive** using more **complex prompting**
- Workloads are **longer and shared**
- **Preble(ICLR ’25)** enables **cache** and **load** to be effectively utilized for **performance**
  - Utilizing **E2 scheduler** and **fair waiting queue**

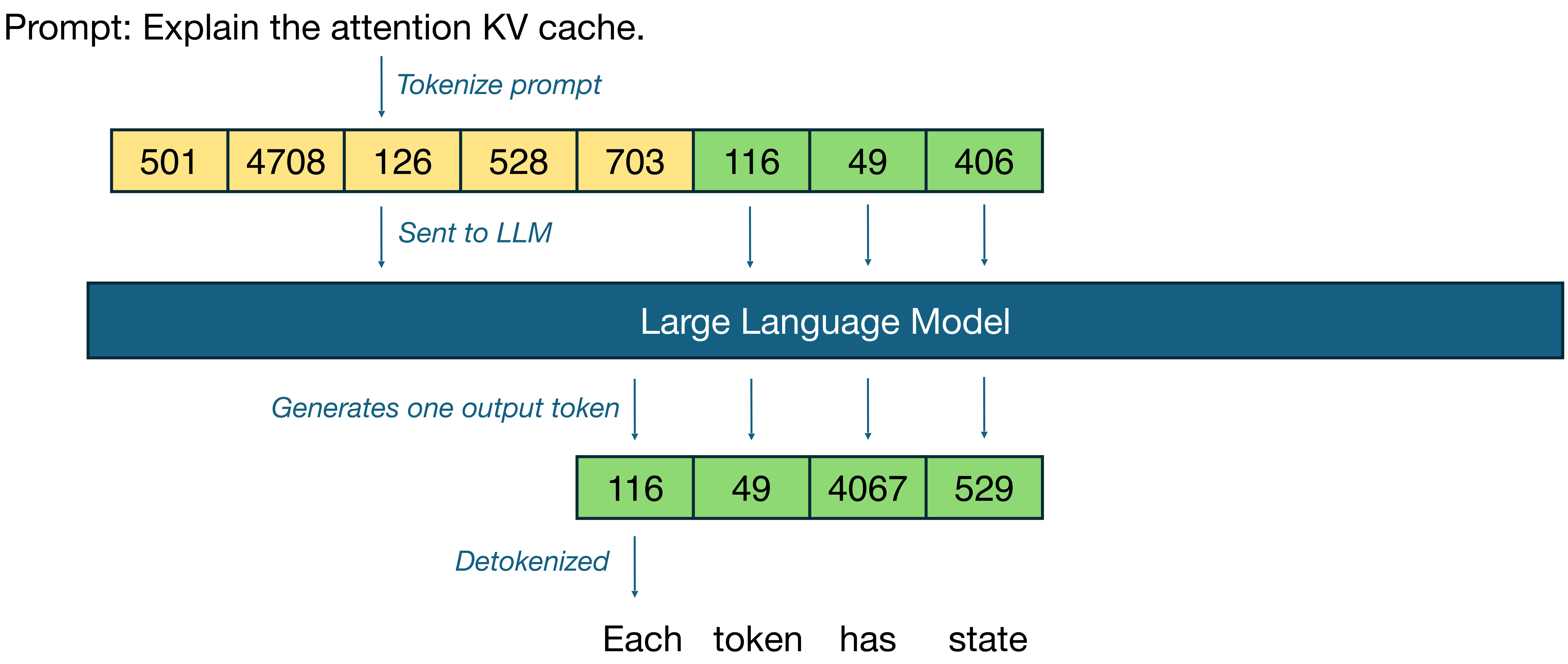
Today's Talk

Cognify [arxiv'25]: AI agent optimizer

Preble [ICLR'25]: long & shared prompt serving

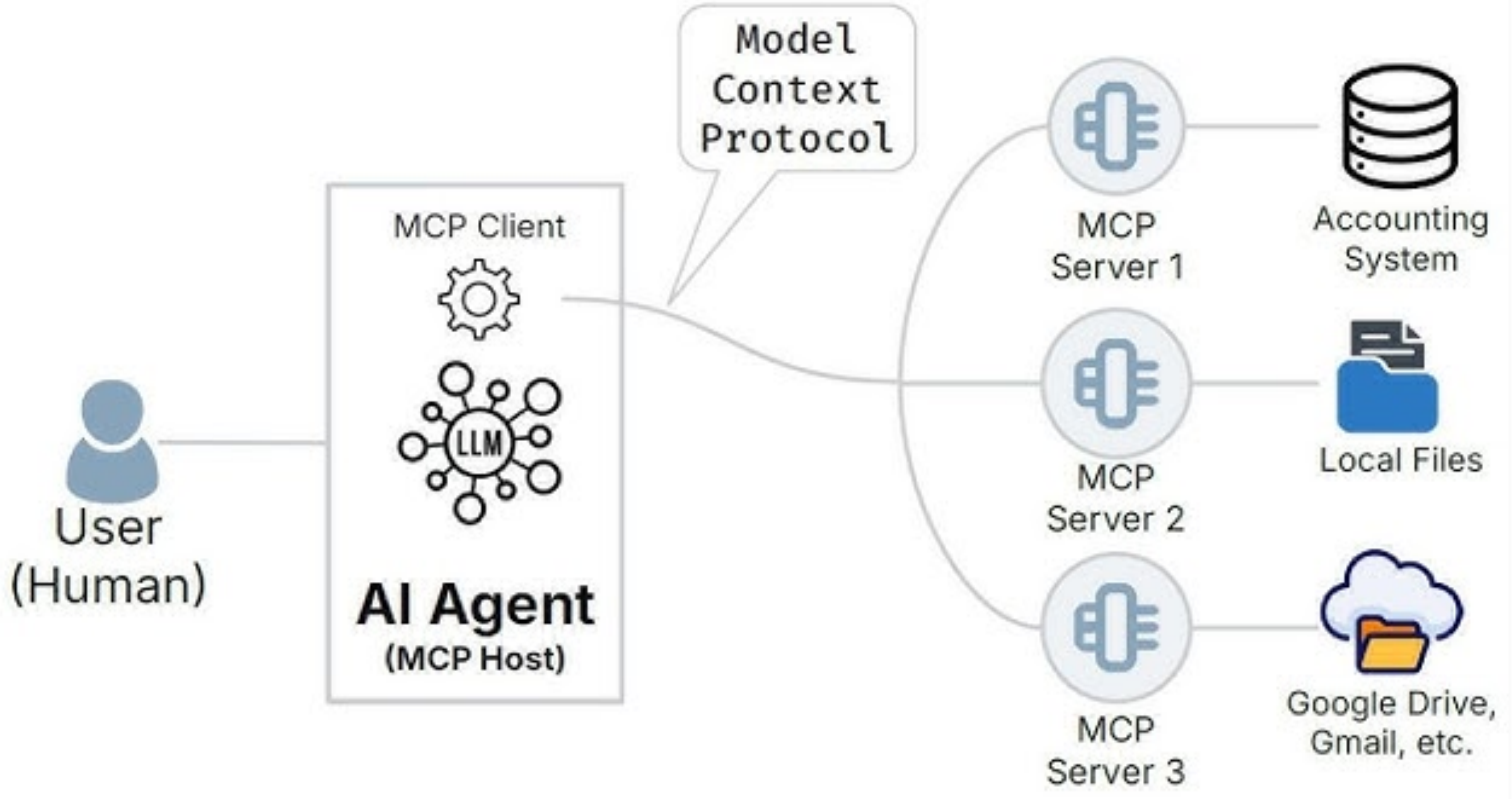
InferCept [ICML'24]: compound LLM serving

Typical LLM Inference: A closer look

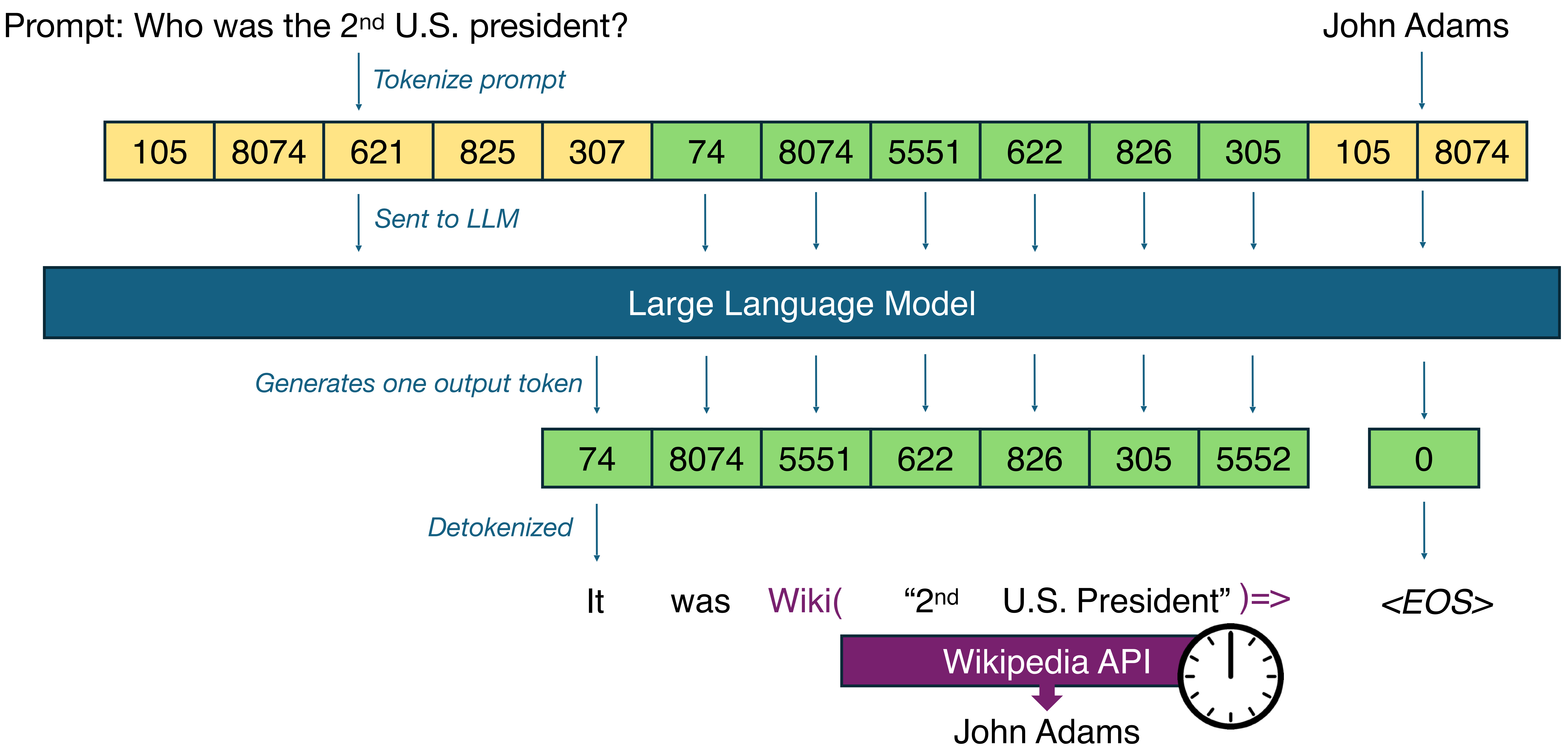


Agent Inference with Tool/Data Calling

Model Context Protocol (MCP)



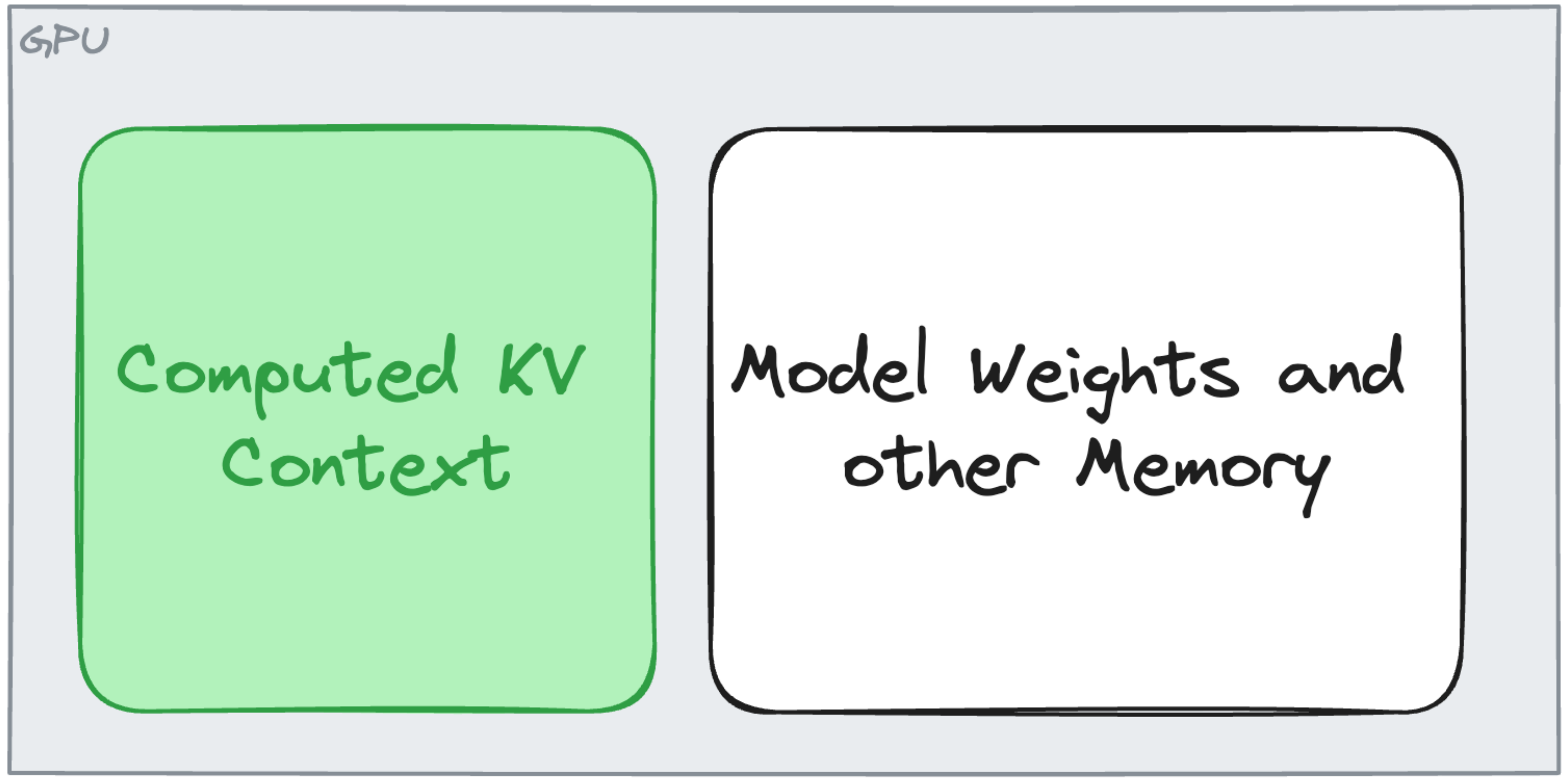
Agent Inference by Augmenting LLM Calls



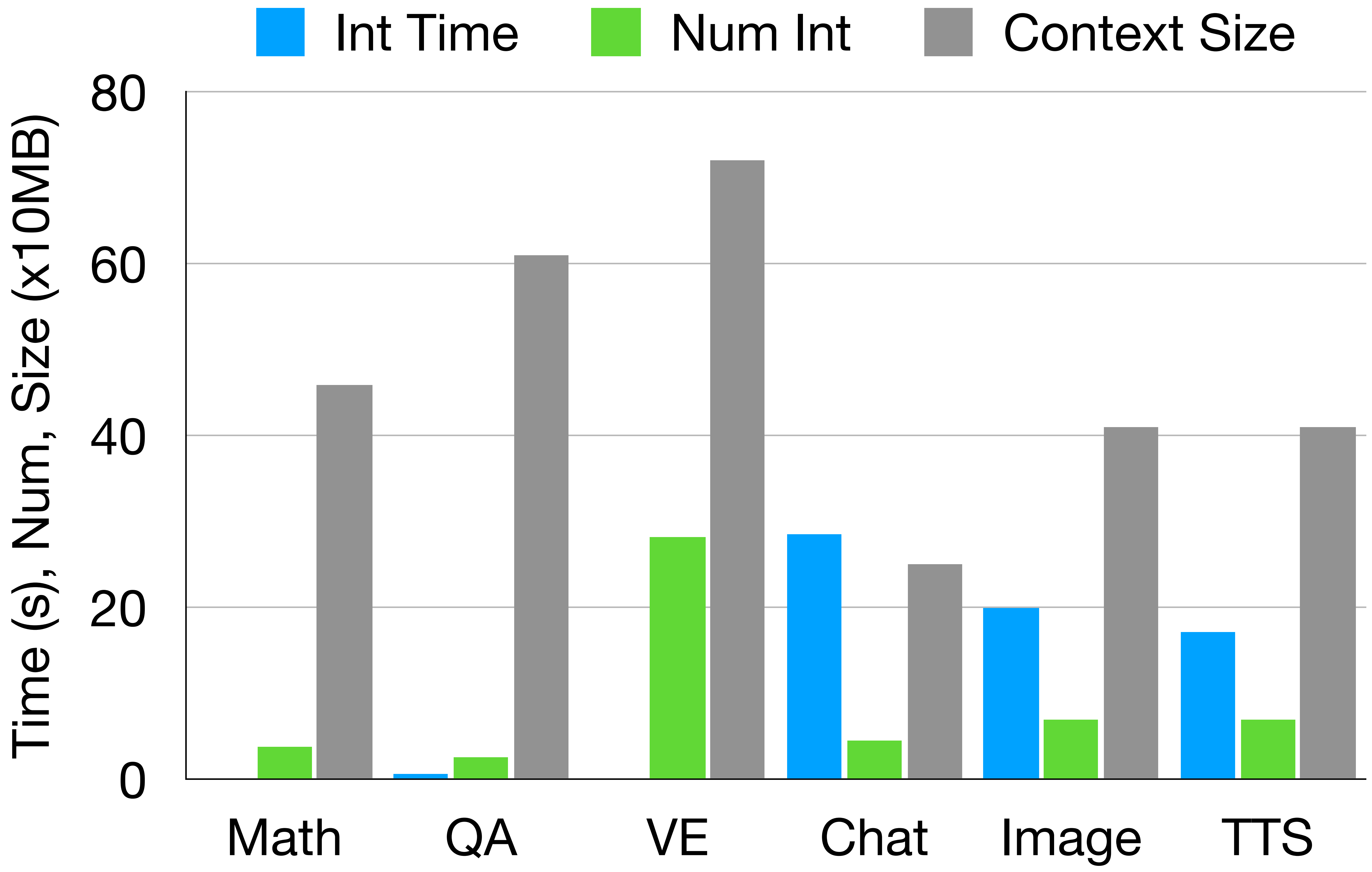
How much is LLM intercepted?

- A study of six sets of real-life compound LLM workloads

- Intercepting for microseconds to minutes
- 2.5 to 28 interceptions per request
- Context 248MB to 720MB per req*



* Assuming a 70B model



How are LLM interceptions handled now?

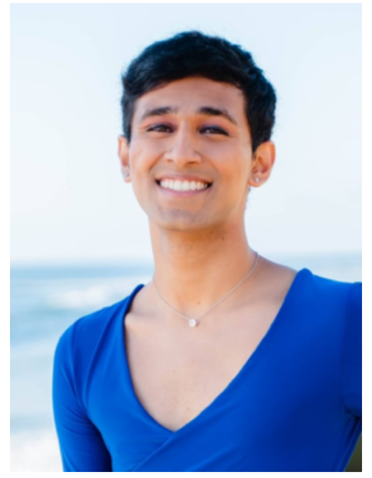
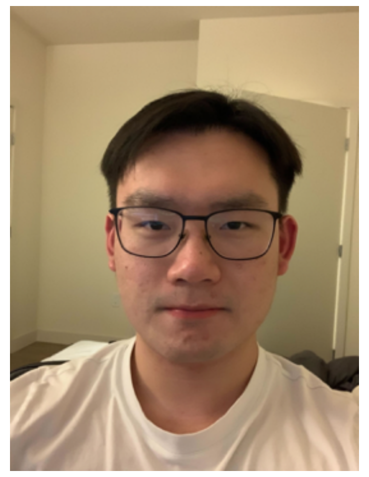
- They are not!

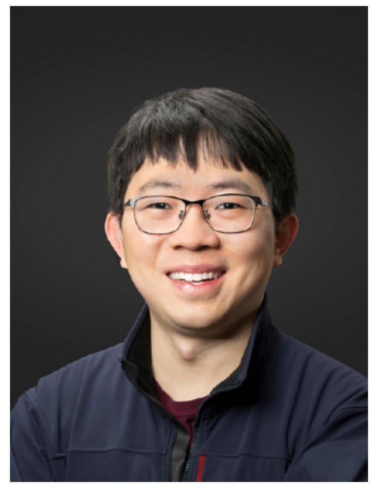
- SoTA LLM serving systems treat LLM interceptions as end of requests
  - Discard all KV context
  - (Re)compute KVs for tokens in context when interception ends
  - 37% - 40% e2e request latency spent on recomputation
  - Wastes 40% GPU resources

InferCept: Adaptive LLM-Centric Workflow Inference

ICML'24, open source at https://github.com/WukLab/infercept

- Pause a request upon intercepting
- Adaptively choose strategies for dealing with KV context
- Efficient implementation of intercept strategies
- Multiple intercepting endpoints supported (tool, other model, human, …)
- 1.6x to 10x improvement over vLLM (SoTA LLM serving system)




"InferCept: Efficient Intercept Support for Large-Language Inferencing", Reyna Abhyankar*, Zijian He*, Vikranth Srivatsa, Hao Zhang, *Yiying Zhang, ICML'24* (* equal contribution)

Three Intercepting Strategies
- when dealing with KV context

- Discard KV context and recompute upon return
- Preserve KV in GPU memory during interception
- Swap KV to CPU memory during interception

Strategy 1: Discard + Recompute

GPU memory

Discard interrupted request

Recompute all context upon return

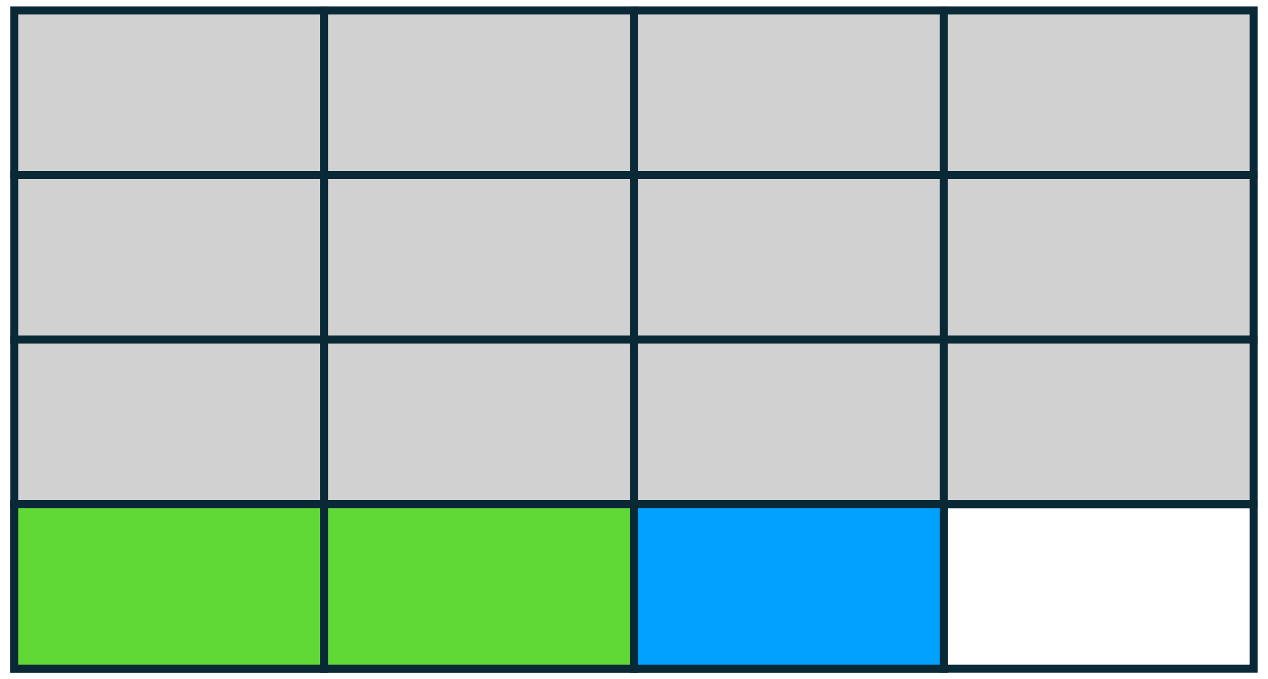
Typical iteration time:
40ms

Other running requests waiting for 4x+ more time!

**Typical recomputation time:
200+ms**

Strategy 2: Preserve

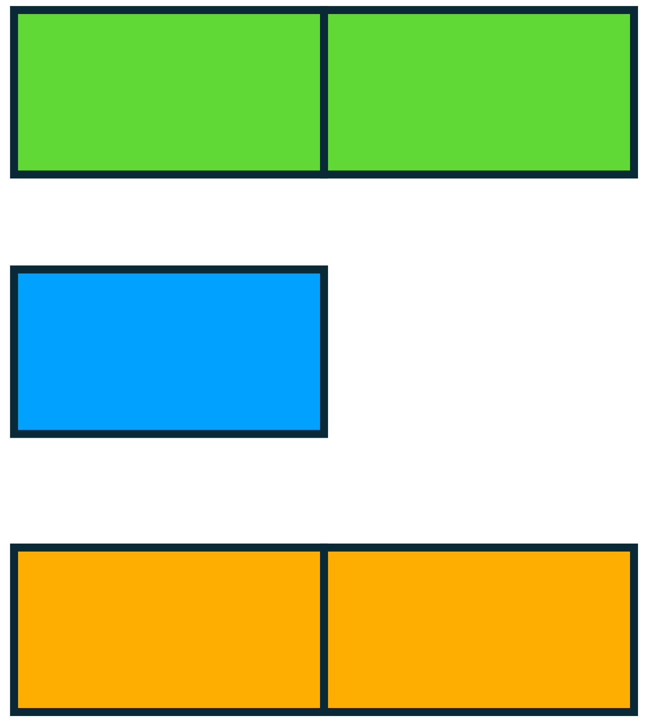
GPU memory



Preserve intercepted request

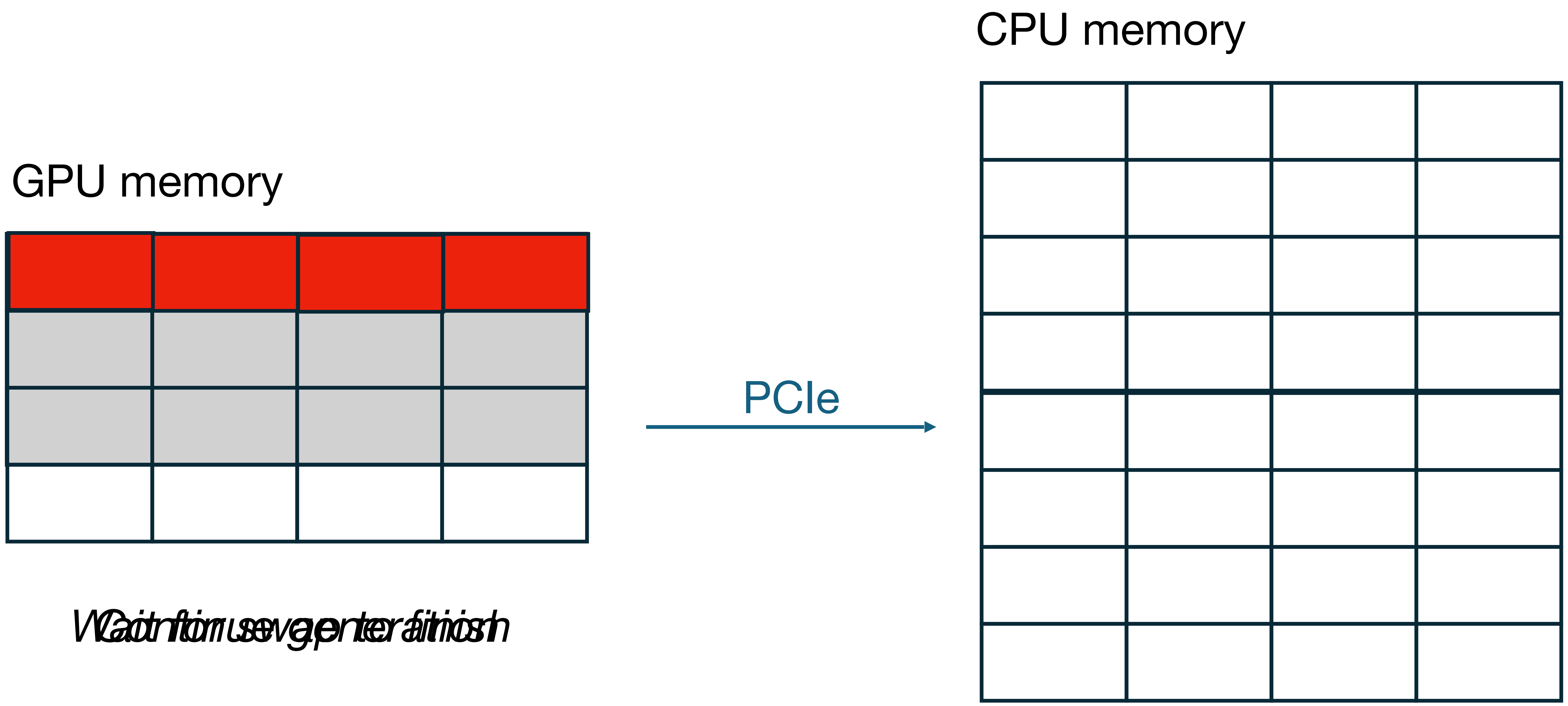
Prefill other requests

<u>Waiting Queue</u>



NOT ENOUGH MEMORY

Strategy 3: Swap to CPU memory



Each strategy has different tradeoffs

- **Discard** + recompute KV — *Recomputing cost + stalls running requests*
- **Preserve** KV in GPU — *Memory unused during interception*
- **Swap** KV to CPU memory — *Swap bandwidth limited*

Which is the best strategy for a given request?
How do we determine this?

Minimizing Waste

A unified measurement for all strategies

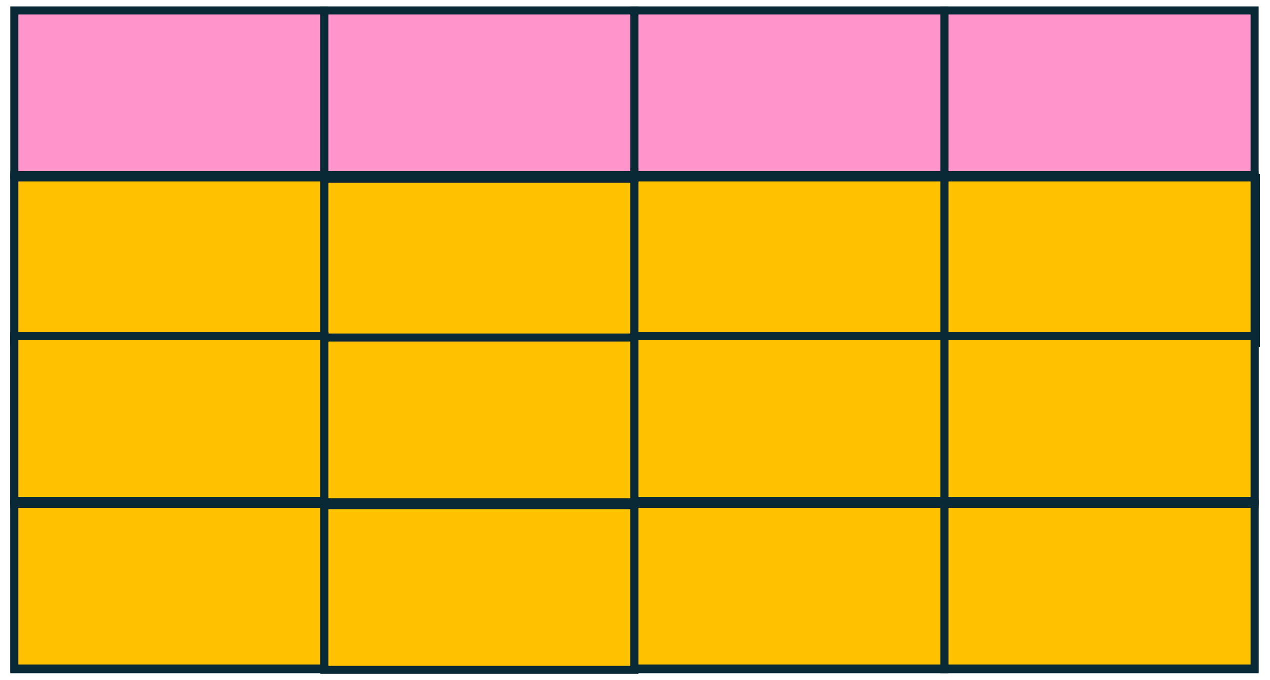
- ***Waste = unused GPU memory * time***
- Accounting for intercepted request and remaining request

For each intercepted request, choose the minimal-waste strategy

Can we improve the existing strategies further?

MinWaste Discard: Chunk Recomputation

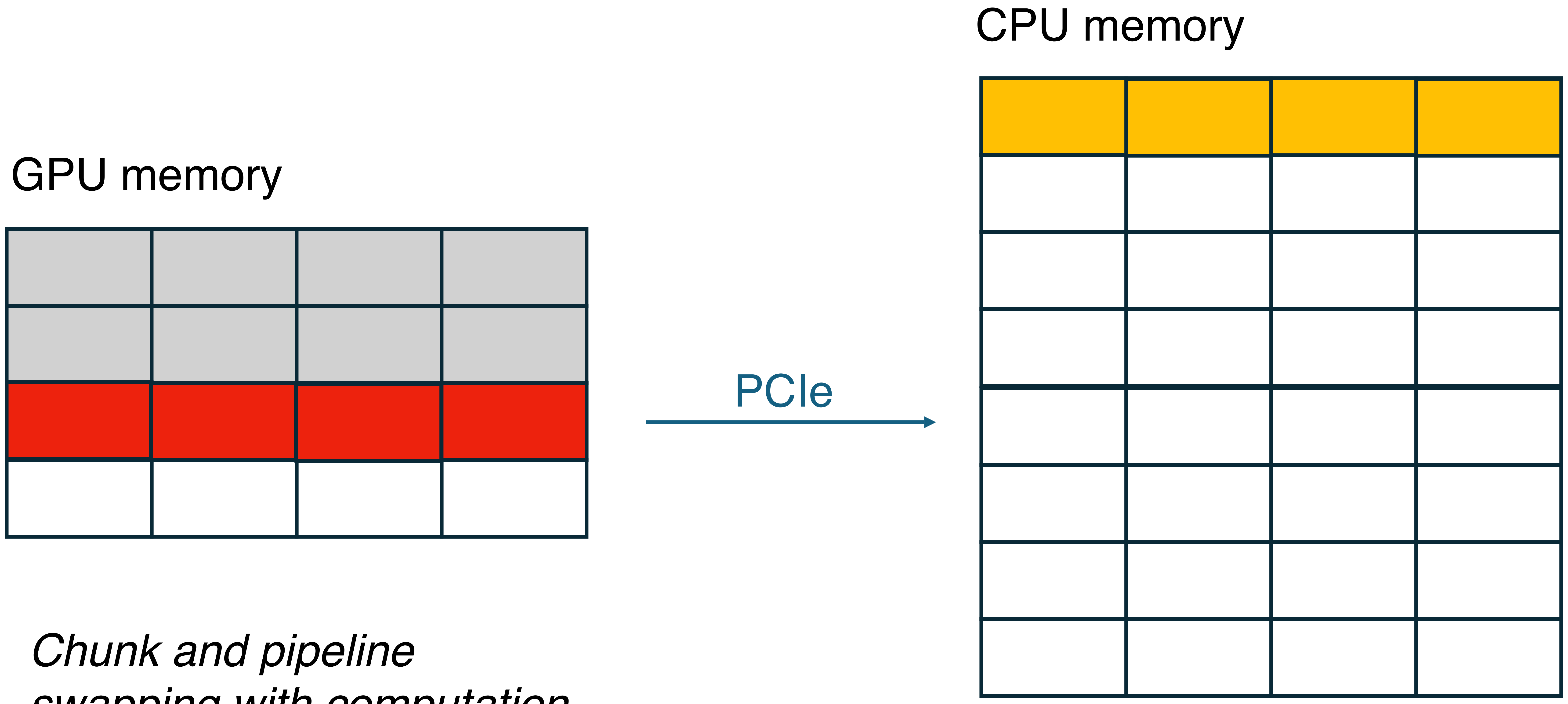
GPU memory



Discarded memory reused by other requests

Recompute one chunk at a time to not stall other running requests

MinWaste Swap: Hide Swap Latency



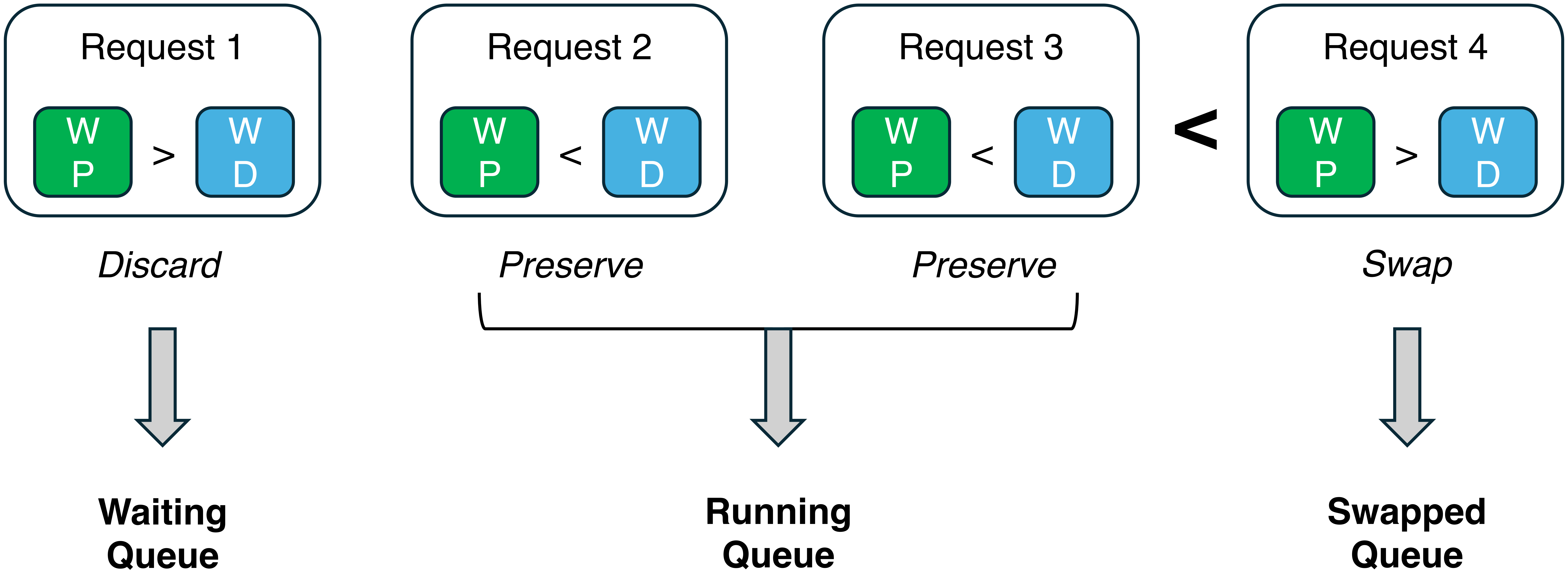
Chunk and pipeline swapping with computation

Swap (within a limit) is completely free!

Scheduling Across Requests

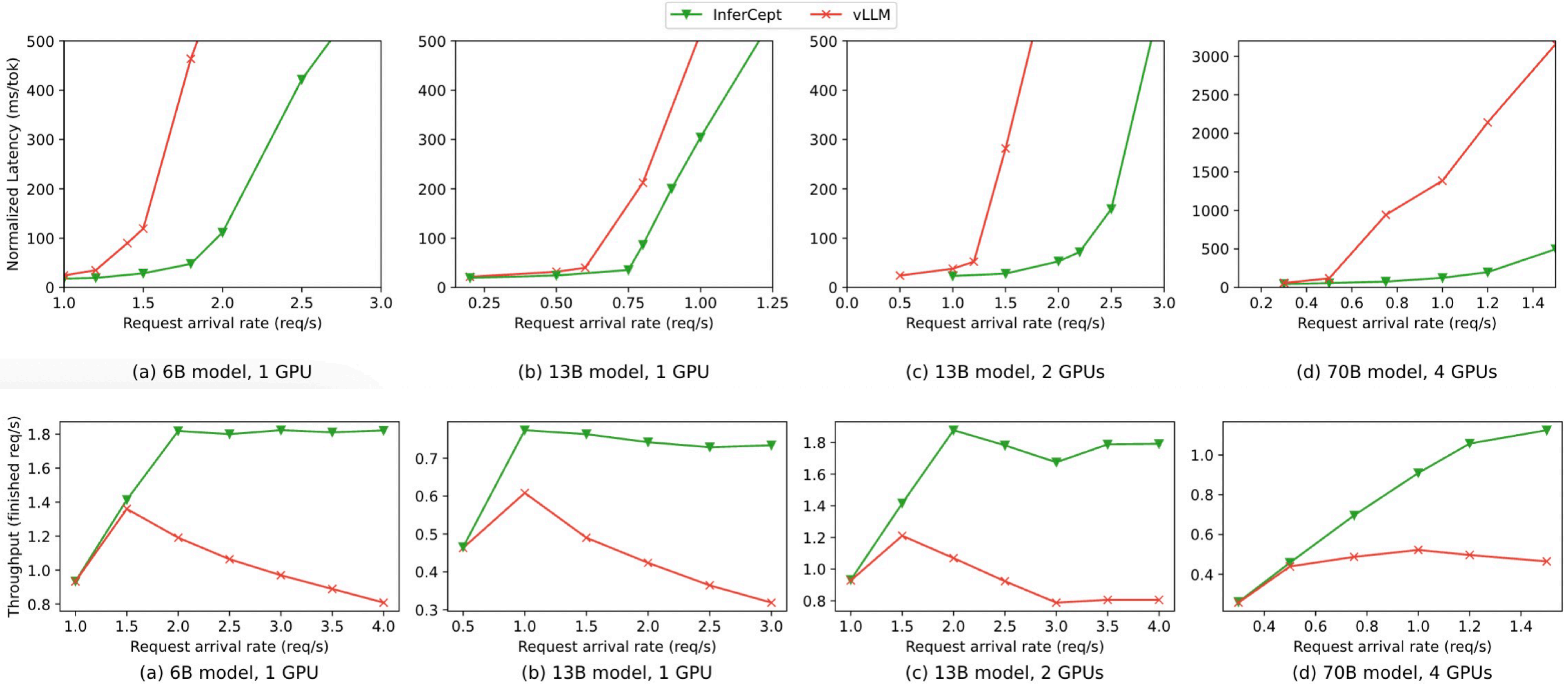
Out of all intercepted requests in an iteration

- Use the swap budget for otherwise **most** wasteful requests
- Choose the smaller waste of preserve and discard for the remaining requests



Results

- 6B GPT-J and 13B Vicuna calling six different tools
- Sustains 1.6x to 2x higher request load and 1.3x to 12x lower latency



InferCept Takeaways

- Model calls are increasingly accompanied by external tool and data calling
- KVs need to be properly managed when external entities intercept model calls
- Three basic strategies, each with pros and cons
- InferCept: first work to manage model/non-model interactions at the system level

Conclusion

- 2025 is the year of AI agents
- Many efforts in agent development and development frameworks and tools
- AI agent infrastructure is largely an unexplored area
- WukLab and GenseeAI are building a cross-stack AI agent platform
- Stay tuned and follow gensee.ai and mlsys.wuklab.io

GenseeAI

UCSDCSE
Computer Science and Engineering