

TEXAS
The University of Texas at Austin

Hardware-Software Co-design for AI Workloads in Datacenters

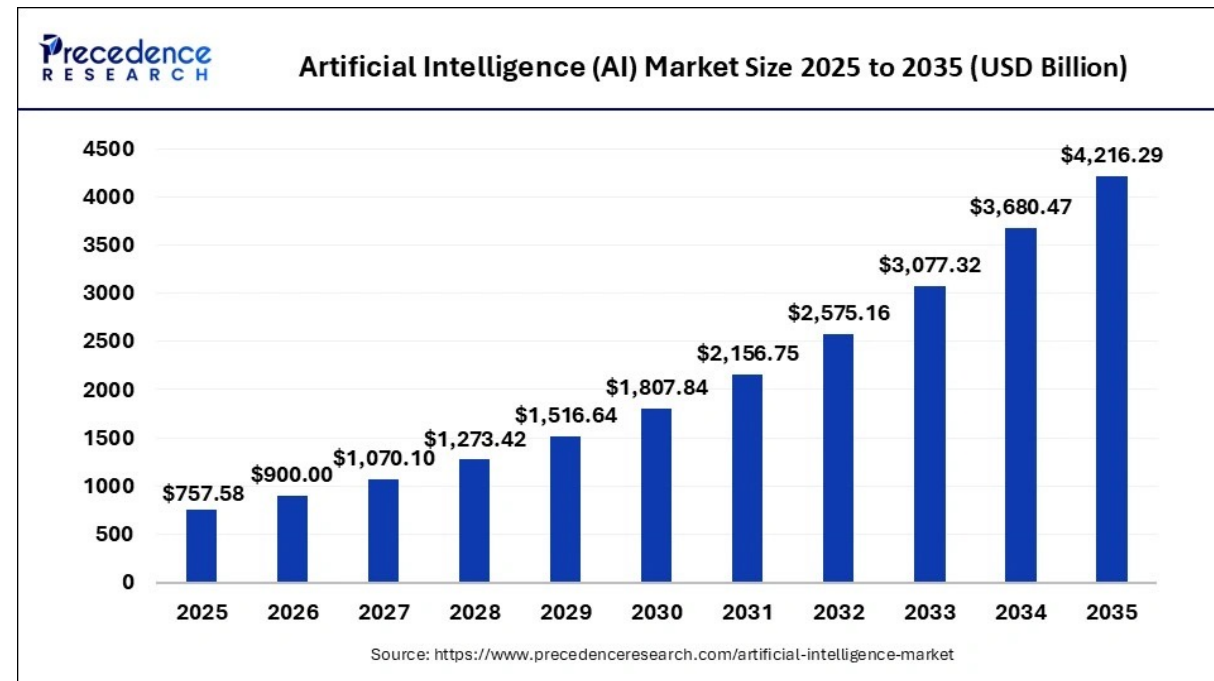
Guest Lecture, Columbia University
September 2026

Jovan Stojkovic

University of Texas at Austin, Computer Science Department

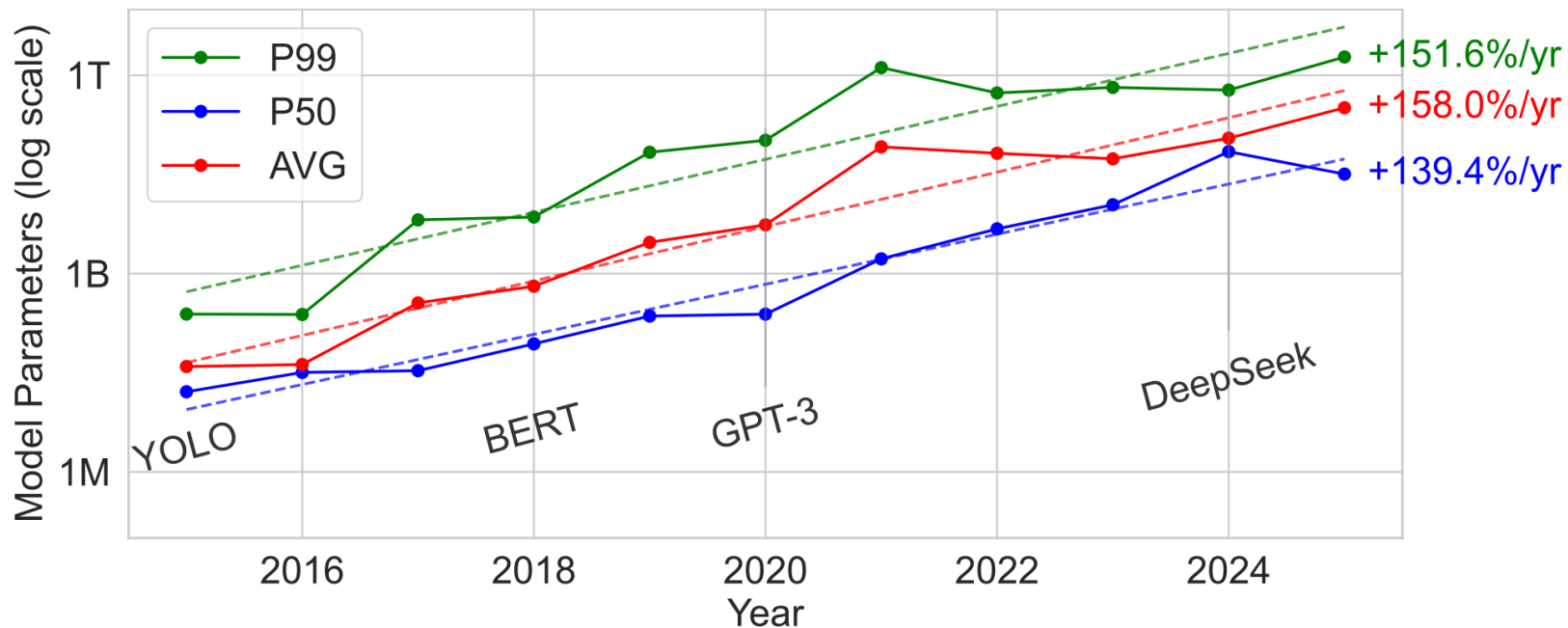
AI Inference in Datacenters

- AI inference becoming a dominant datacenter workload



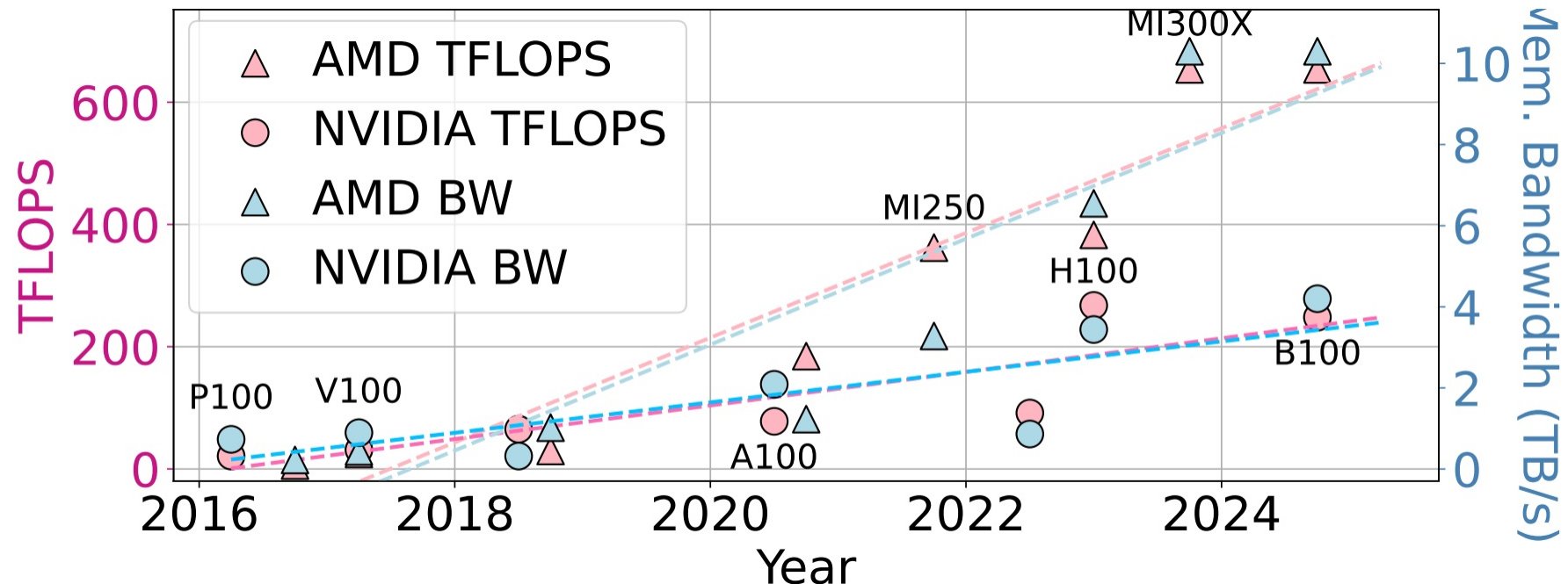
AI Inference in Datacenters

- AI inference becoming a dominant datacenter workload
- AI workloads and hardware are fast evolving



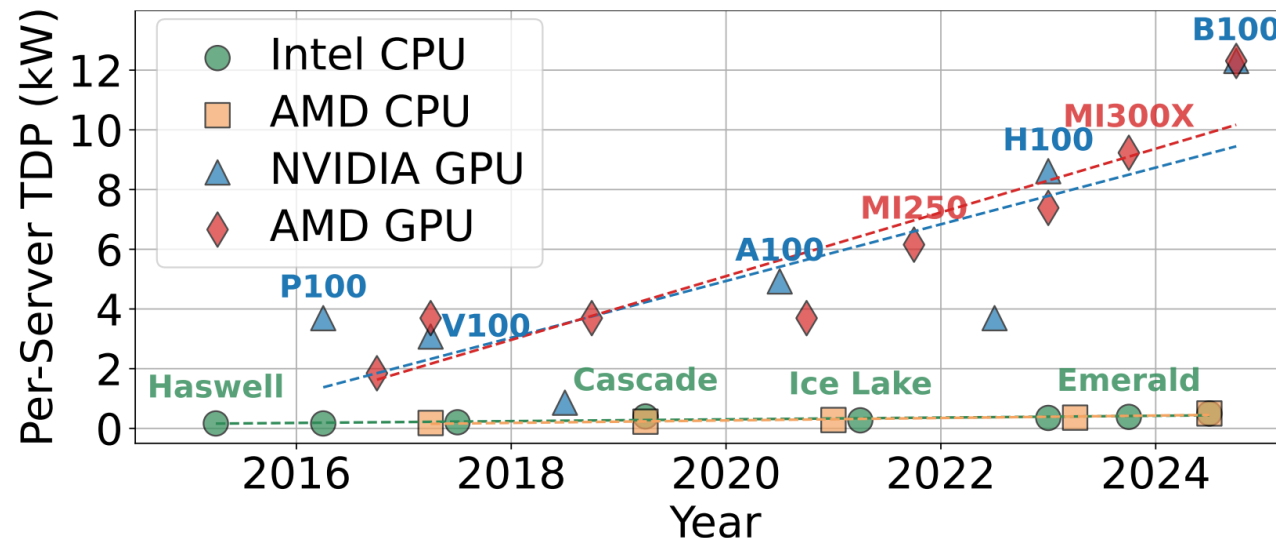
AI Inference in Datacenters

- AI inference becoming a dominant datacenter workload
- AI workloads and hardware are fast evolving



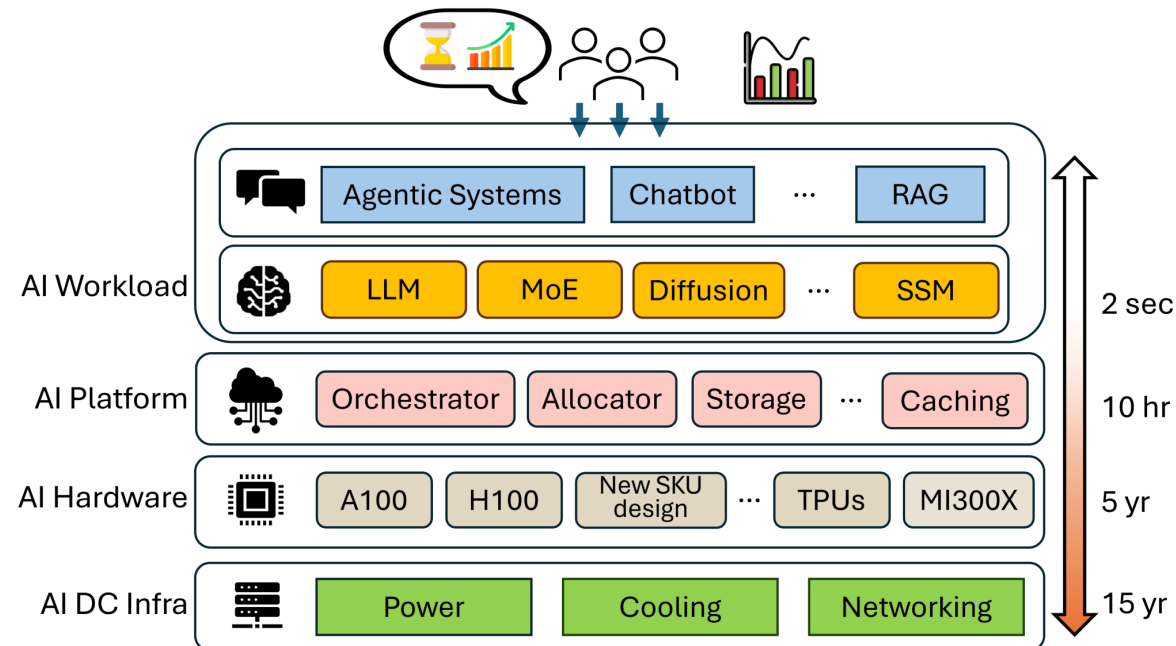
AI Inference in Datacenters

- AI inference becoming a dominant datacenter workload
- AI workloads and hardware are fast evolving
- Traditional datacenter design was tuned for CPUs



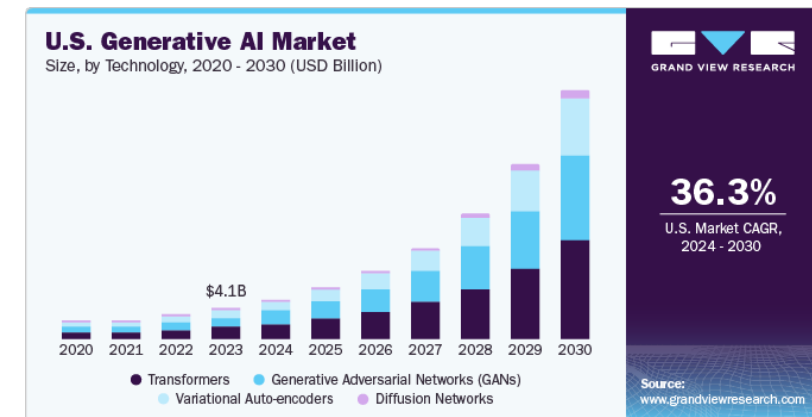
AI Inference in Datacenters

- How should cloud providers build, provision, and operate AI datacenters to minimize long-term TCO?

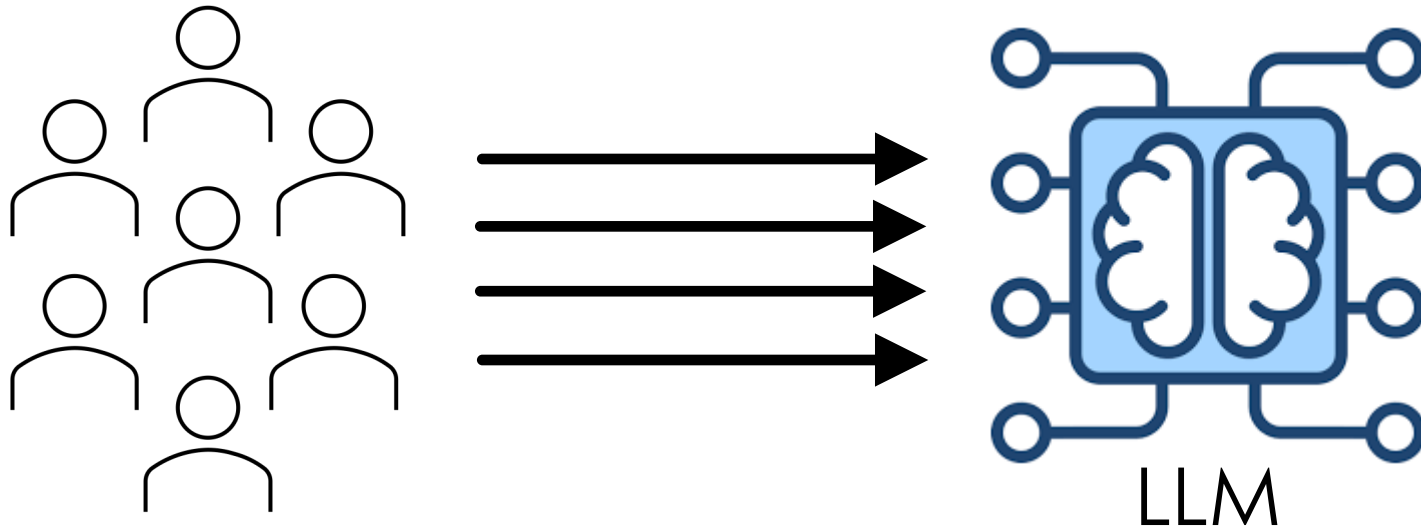


LLM Inference in Datacenters

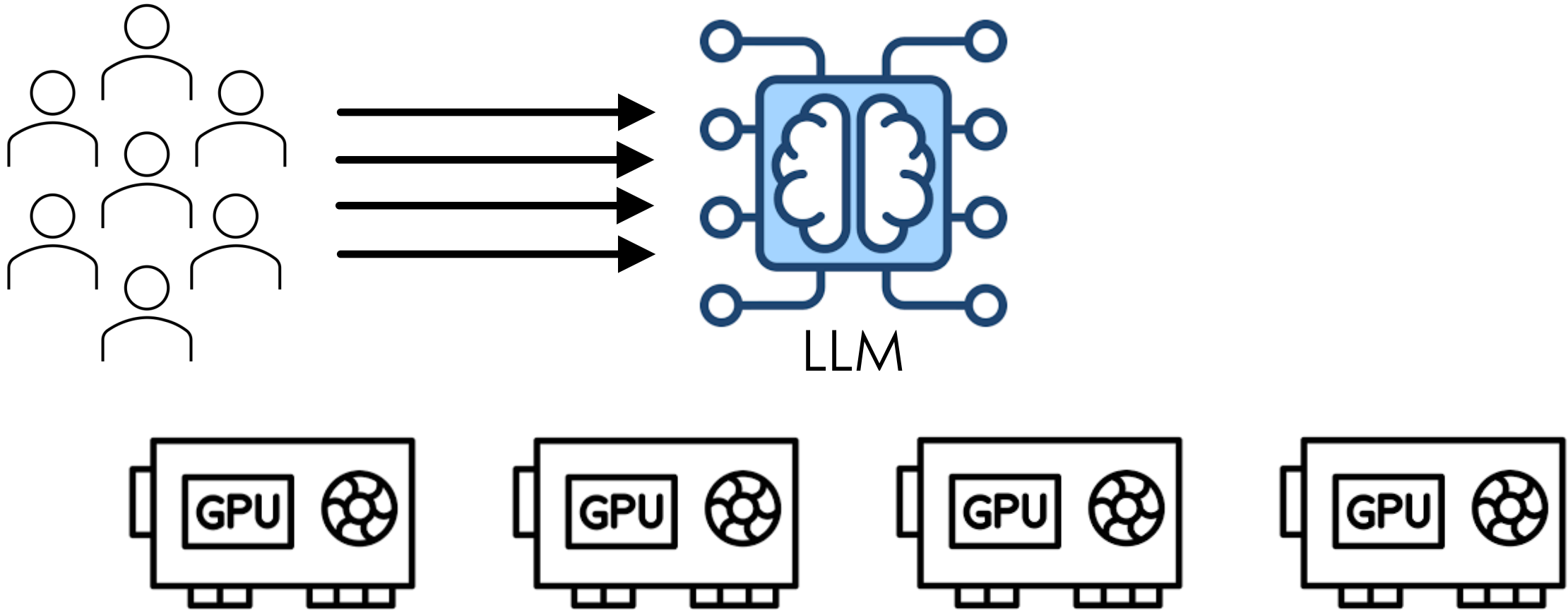
- Modern generative LLMs are turning ubiquitous
 - Use cases: programming, chat-bots, education, healthcare



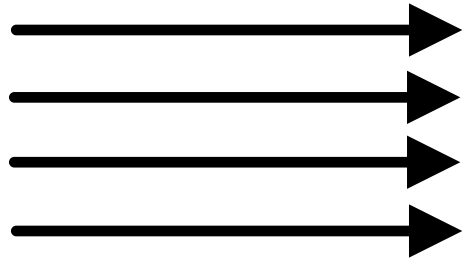
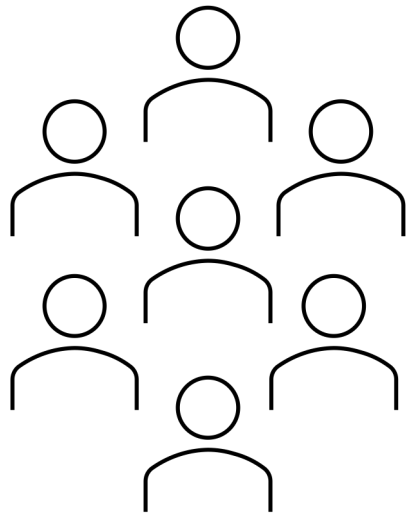
LLM Inference Stresses Cloud Infrastructure



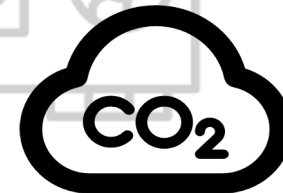
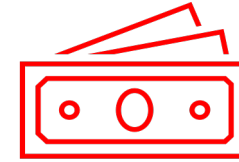
LLM Inference Stresses Cloud Infrastructure



LLM Inference Stresses Cloud Infrastructure



LLM



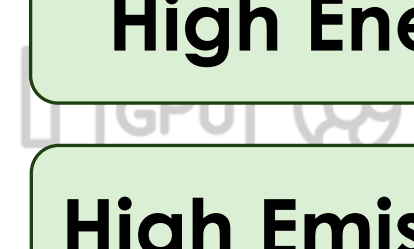
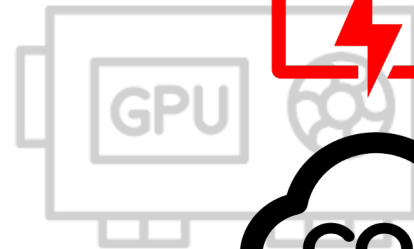
Expensive!

High Power!

High Temp!

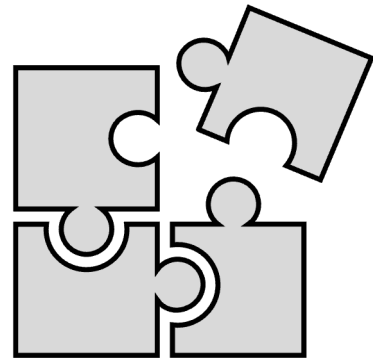
High Energy!

High Emissions!



How to Tame the LLMs?

- Lots of work on performance, accuracy, scalability
- LLMs cause the datacenter cost, energy, and carbon emissions to skyrocket



Energy, power, and cost efficiency of LLMs is a missing piece of a puzzle!

How to Tune LLMs?

Knob	Energy	Power	Perf	Quality

How to Tune LLMs?

Knob	Energy	Power	Perf	Quality
Model Size ↓	↓	↓	↑	↓ ↓

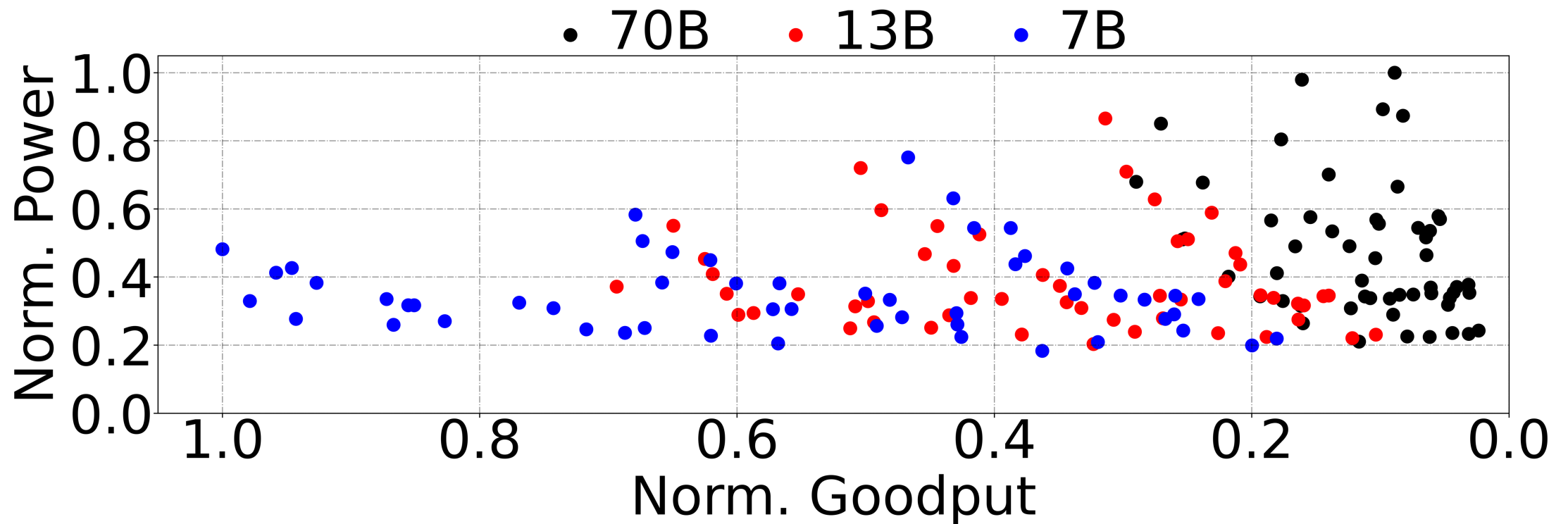
How to Tune LLMs?

Knob	Energy	Power	Perf	Quality
Model Size 				 
Quantize 				

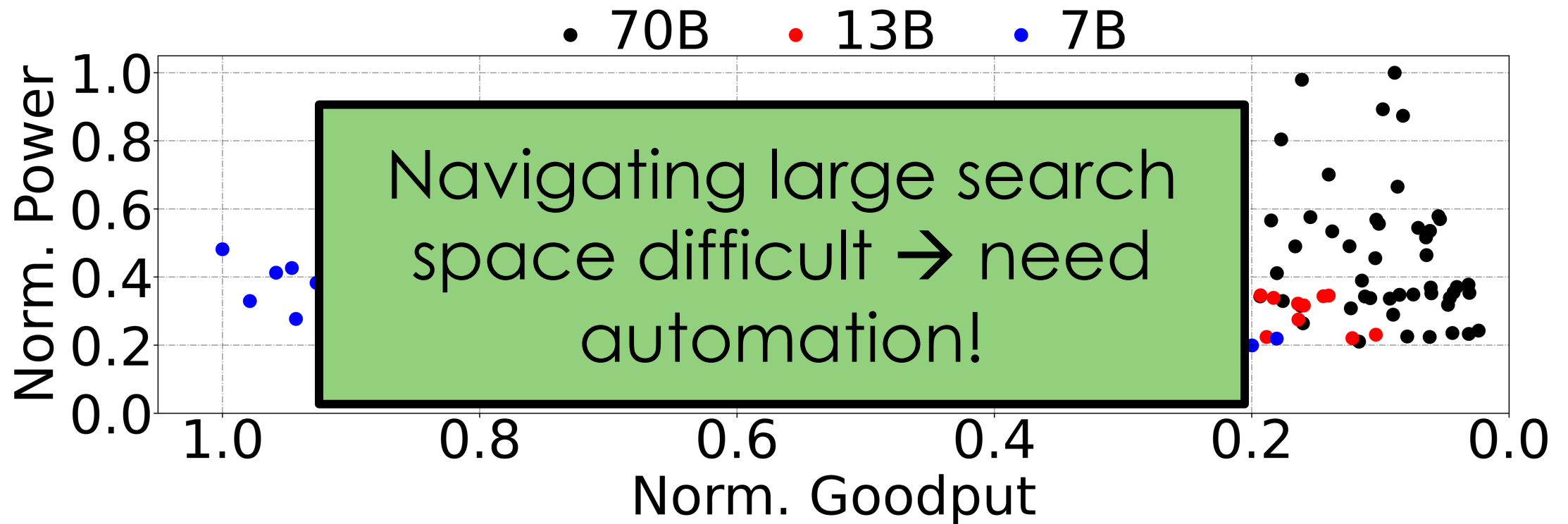
How to Tune LLMs?

Knob	Energy	Power	Perf	Quality
Model Size ↓	↓	↓	↑	↓ ↓
Quantize ↓	↓	↓	↑	↓
Parallelism ↓	↑ ↓	↓	↓	
Frequency ↓	↑ ↓	↓	↓	
Batch size ↓	↑ ↓	↓	↓	

Large Configuration Search-Space



Large Configuration Search-Space

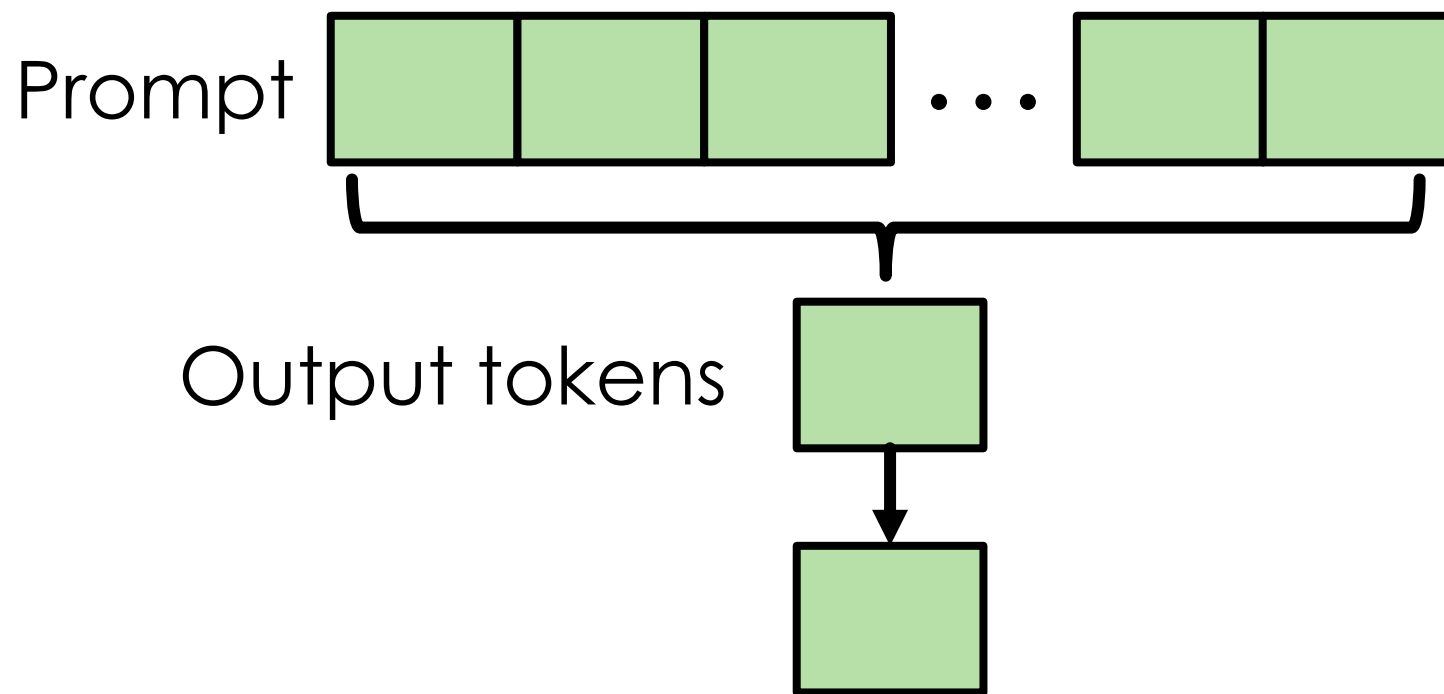


Goal: Make LLMs energy-efficient

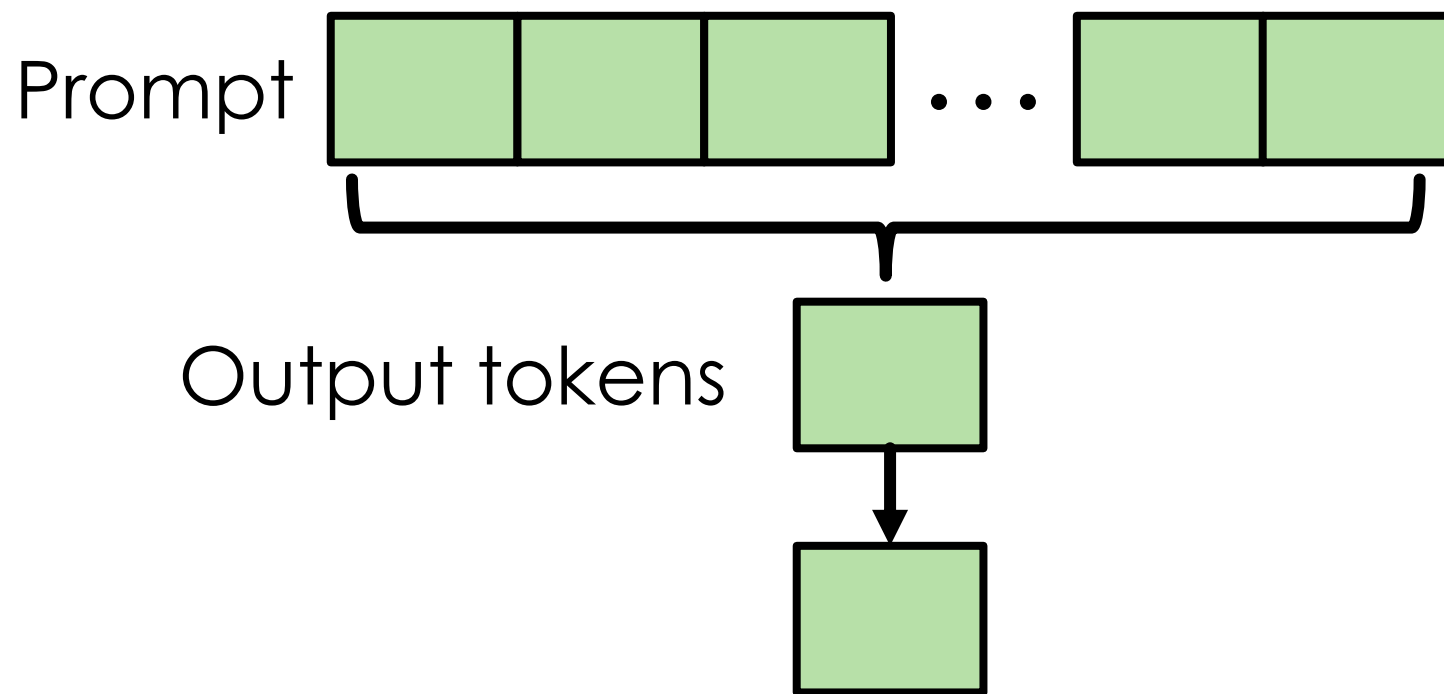
○ Challenges

1. Request heterogeneity
2. Workload dynamics
3. Reconfiguration overheads

Challenge #1: Request Heterogeneity

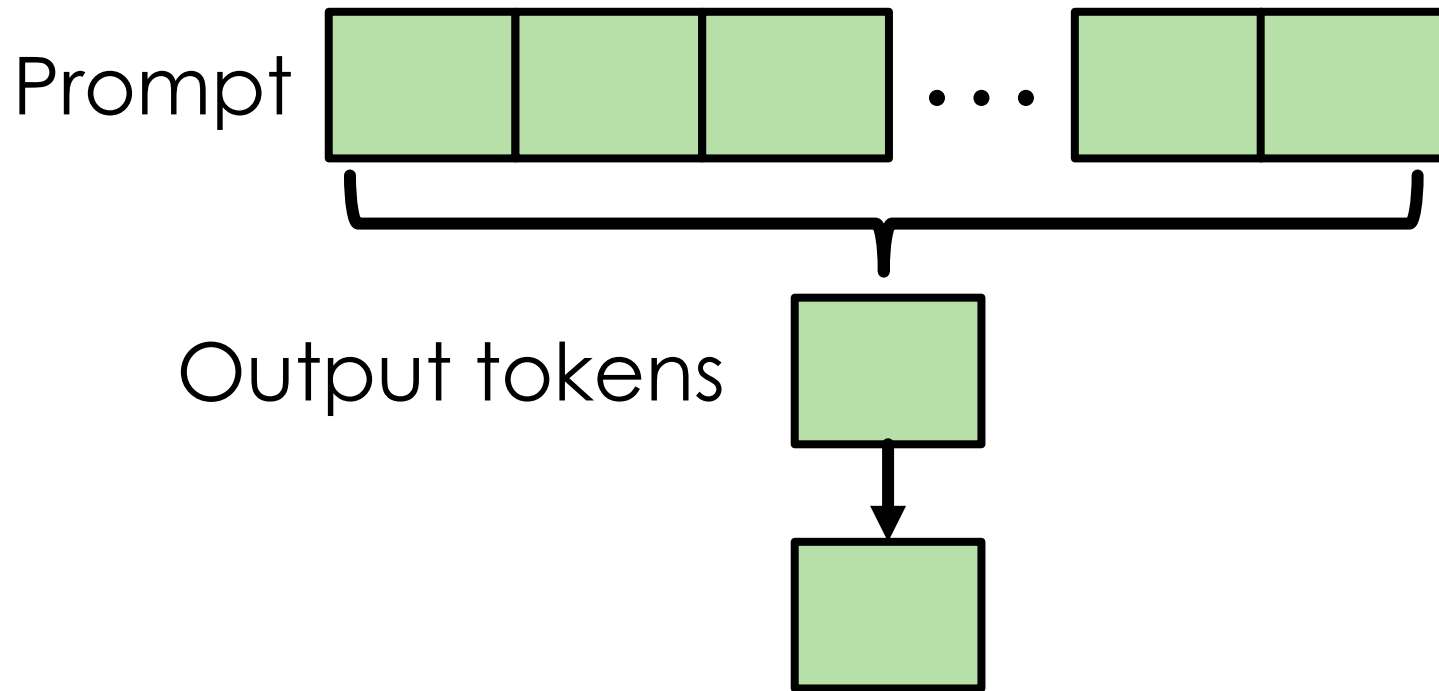


Challenge #1: Request Heterogeneity



**Long input,
short output**

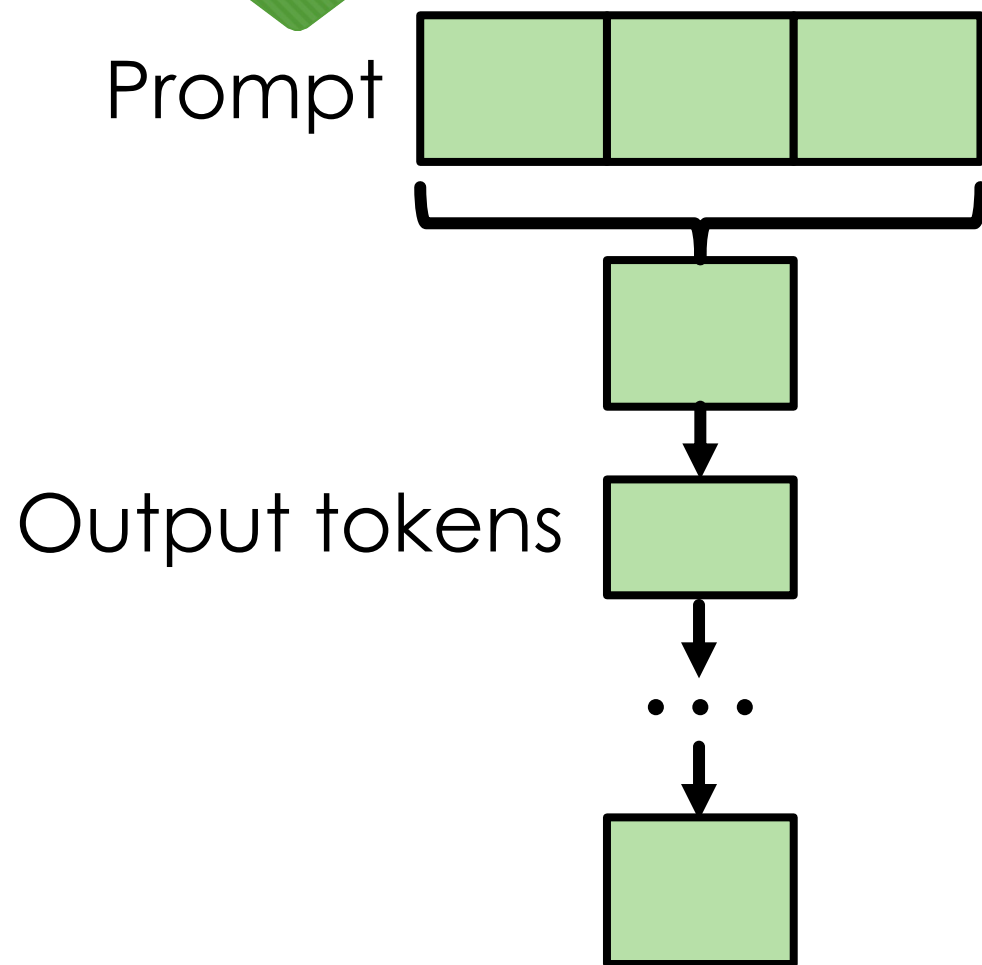
Challenge #1: Request Heterogeneity



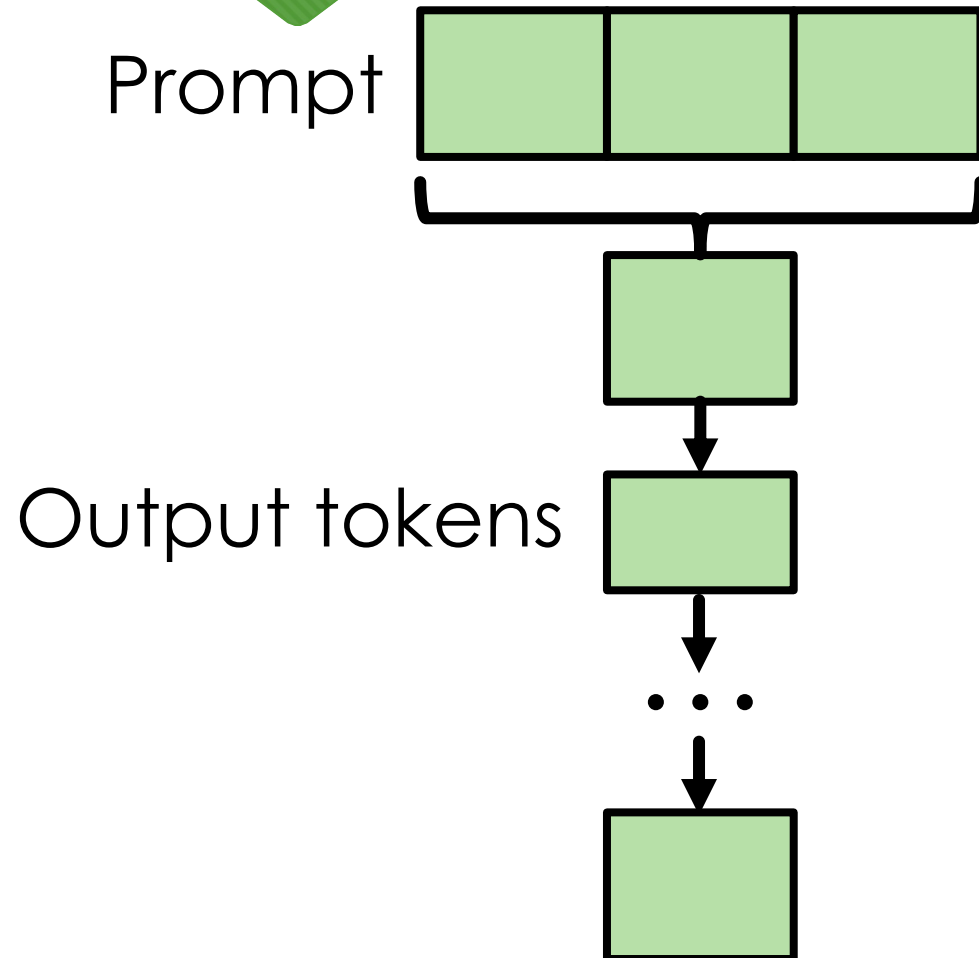
**Long input,
short output**

Compute-bound

Challenge #1: Request Heterogeneity

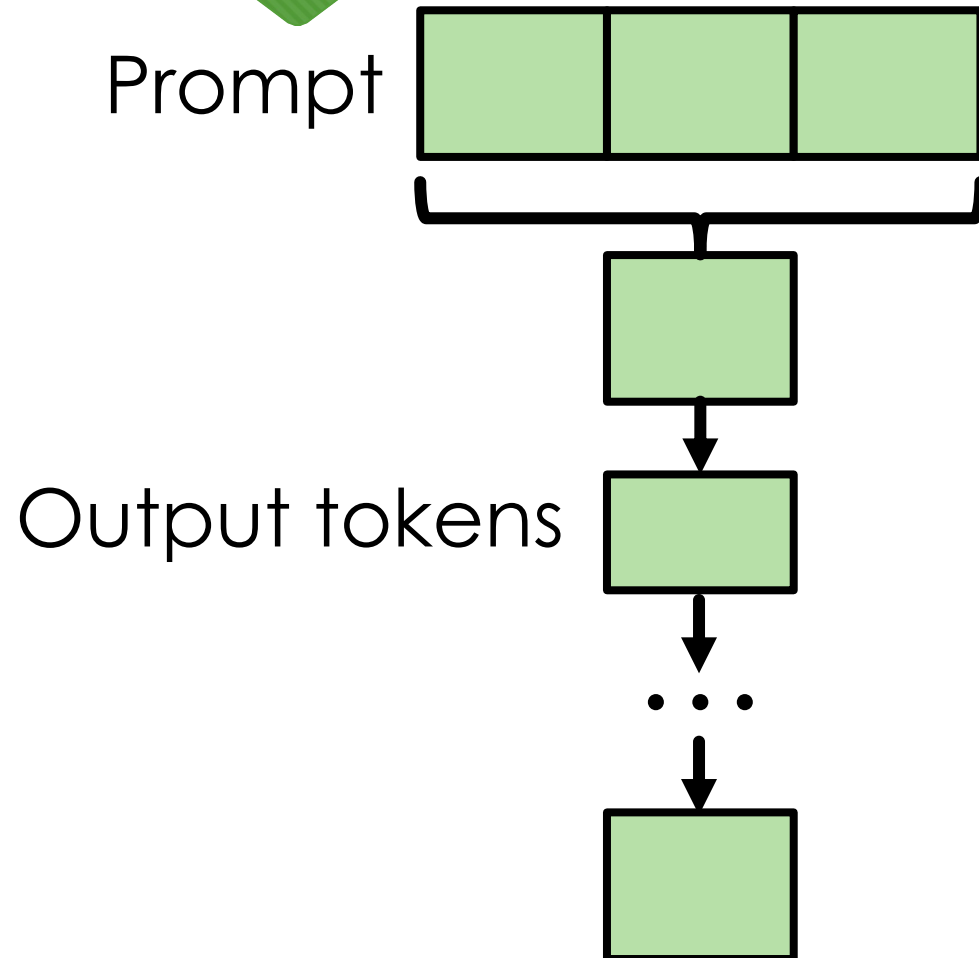


Challenge #1: Request Heterogeneity



**Short input,
long output**

Challenge #1: Request Heterogeneity



**Short input,
long output**

Memory-bound

Challenge #1: Request Heterogeneity

[illegible]

Challenge #1: Request Heterogeneity

Shard	TP2				TP4				TP8			
Freq [GHz]	0.8	1.2	1.6	2	0.8	1.2	1.6	2.0	0.8	1.2	1.6	2.0
SS ←	Short input, short output											
SL												
LS												
LL ←	Long input, long output											

Challenge #1: Request Heterogeneity

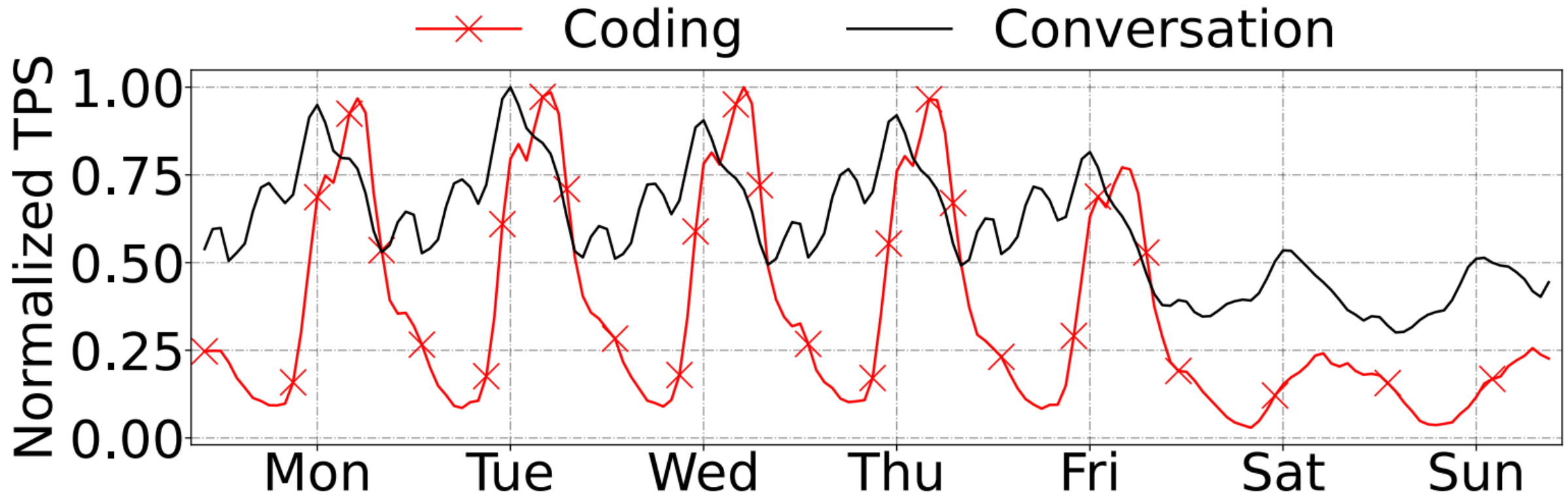
[illegible]

Challenge #1: Request Heterogeneity

Shard	TP2				TP4				TP8			
Freq [GHz]	0.8	1.2	1.6	2	0.8	1.2	1.6	2.0	0.8	1.2	1.6	2.0
SS		✓										
SL												
LS						✓						
LL											✓	

Challenge #1: Different request types require different configurations

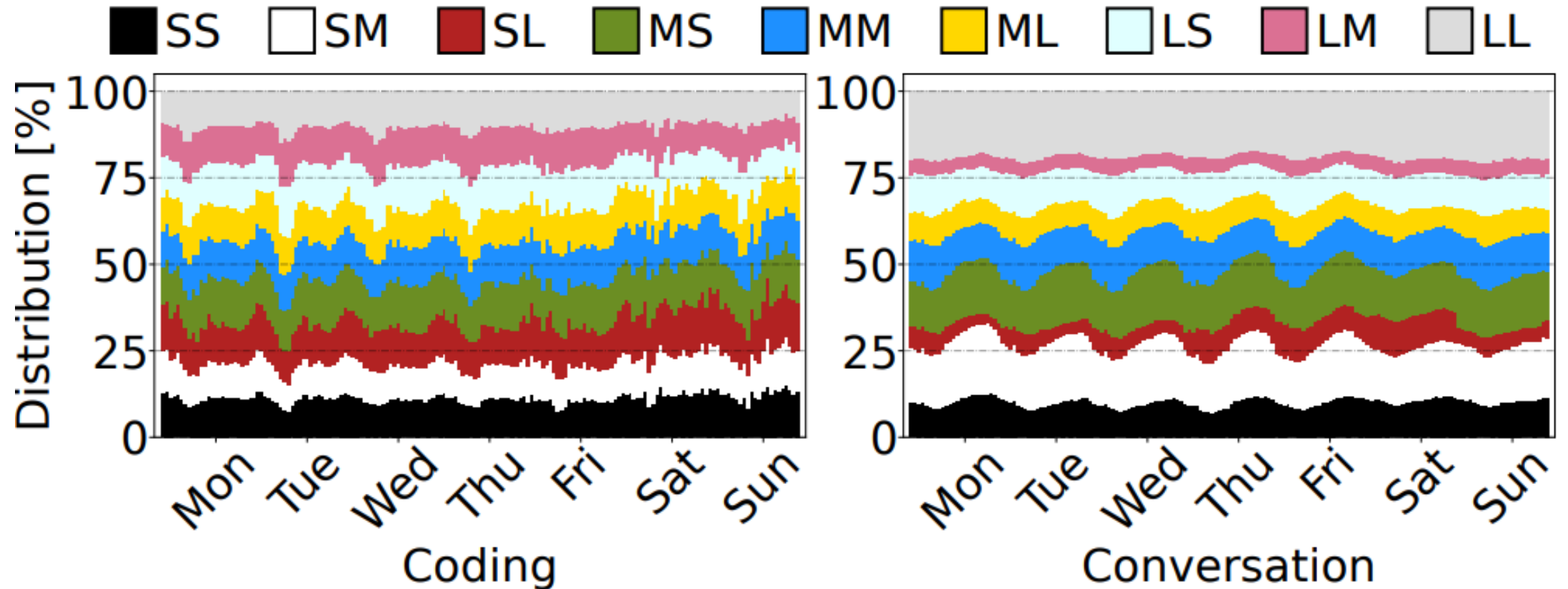
Challenge #2: Workload Dynamics



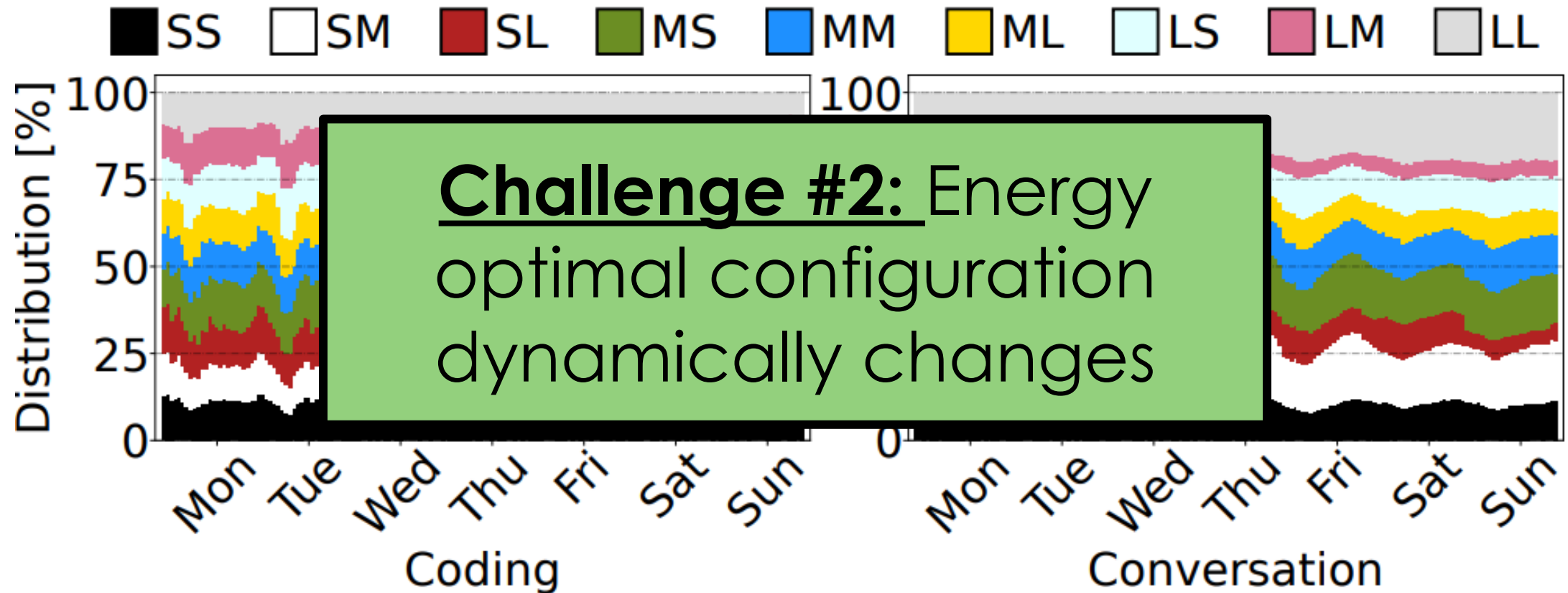
Challenge #2: Workload Dynamics

Sharding	TP4			
Frequency	0.8 GHz	1.2 GHz	1.6 GHz	2.0 GHz
Low Load		✓		
Med Load	✗		✓	
High Load	✗	✗		✓

Challenge #2: Workload Dynamics



Challenge #2: Workload Dynamics



Challenge #3: Re-configuration expensive

- Reconfiguration
 - i. Scale in/out
 - ii. Shard in/out
 - iii. Scale up/down

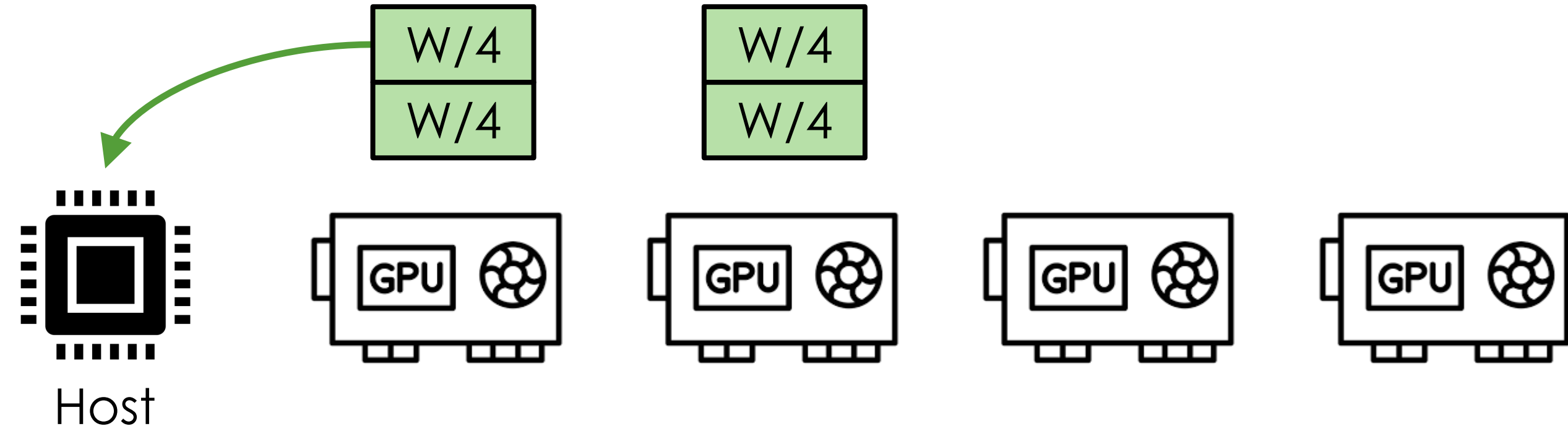
Challenge #3: Re-configuration expensive

i) Scale in/out costs

Overhead source	Time
Create an H100 VM	1-2 min
Init distributed environment	2 min
Download model weight	3 min
Setup inference engine	20 seconds
Install weights and KV cache on GPU	15 seconds
Total	6-8 min

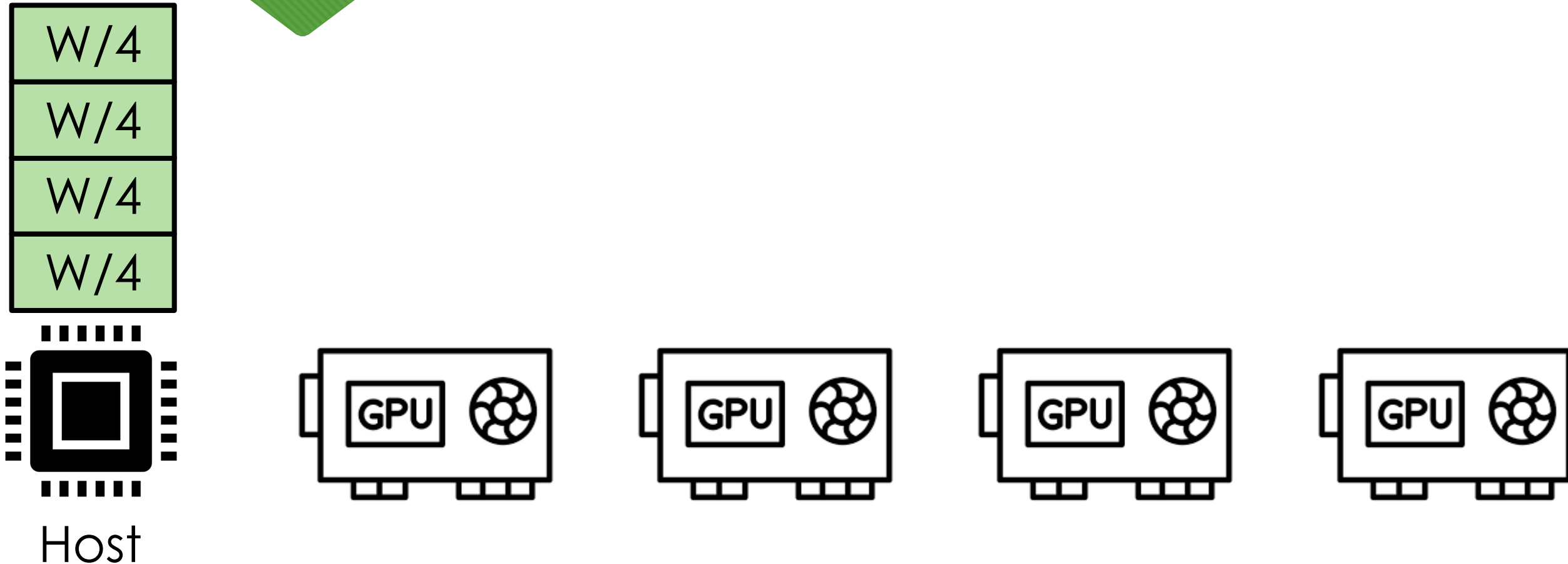
Challenge #3: Re-configuration expensive

ii) Shard in/out costs



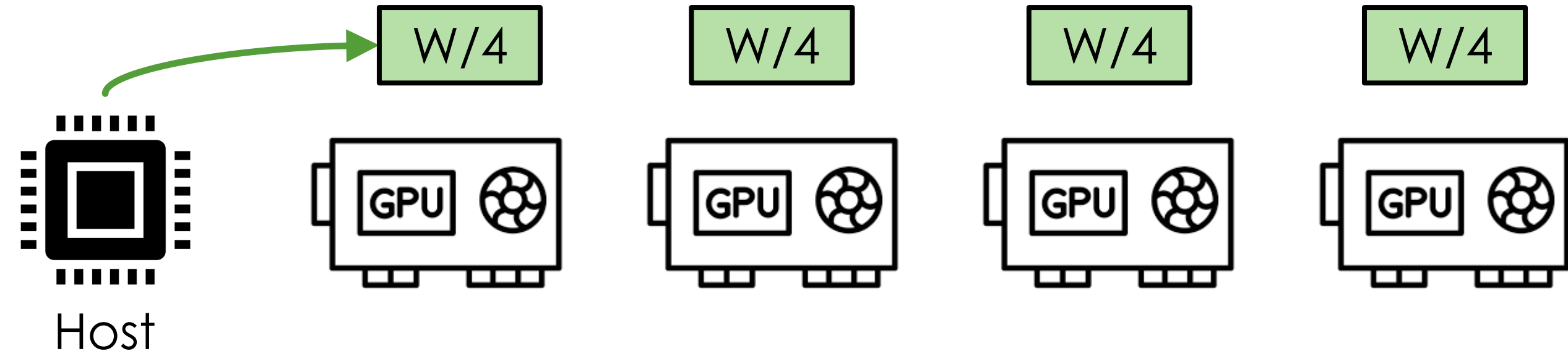
Challenge #3: Re-configuration expensive

ii) Shard in/out costs



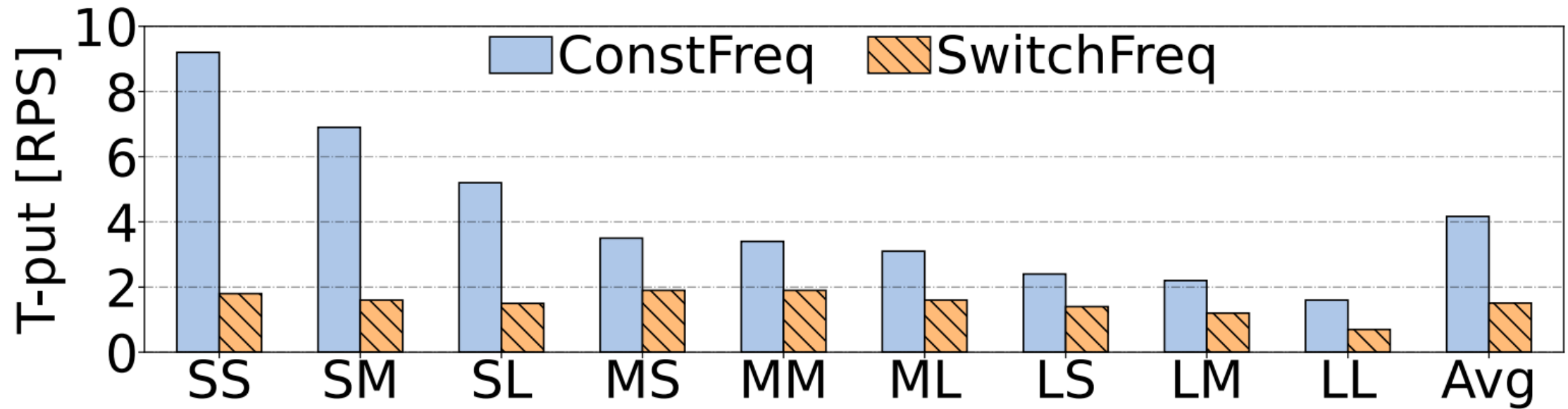
Challenge #3: Re-configuration expensive

ii) Shard in/out costs



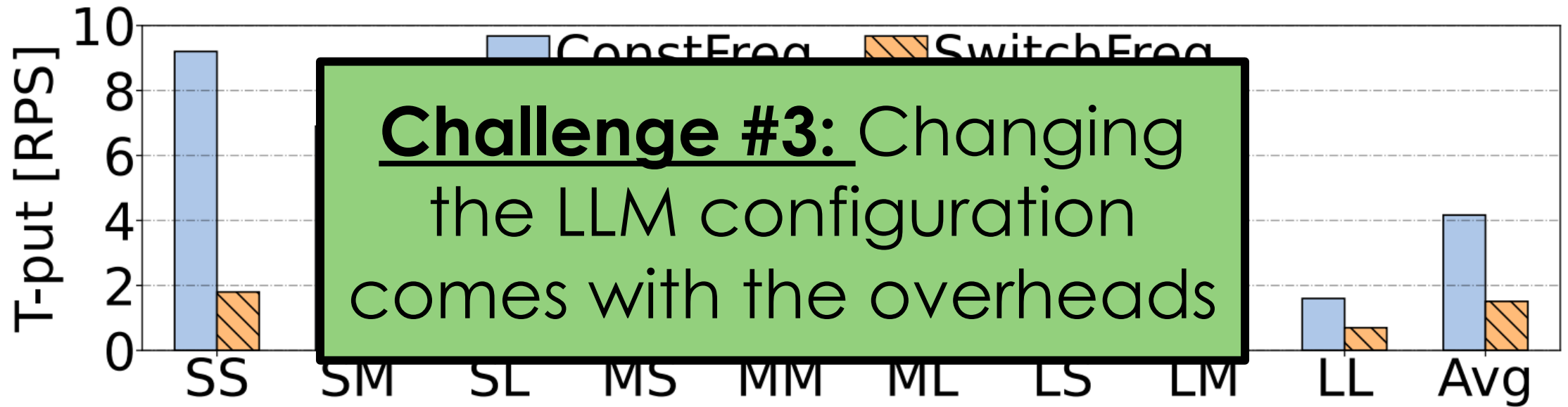
Challenge #3: Re-configuration expensive

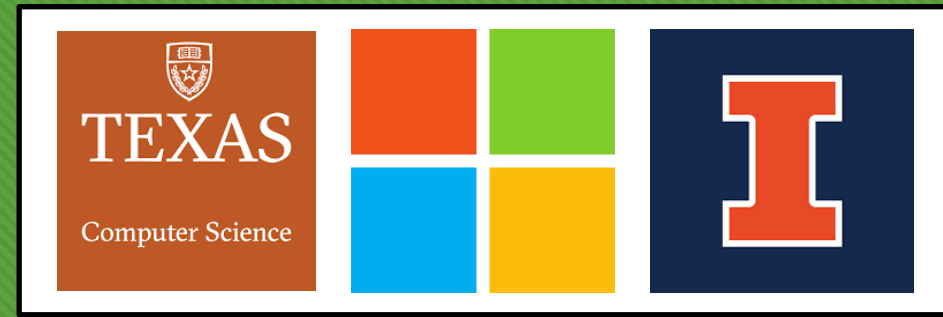
iii) Scale up/down costs



Challenge #3: Re-configuration expensive

iii) Scale up/down costs





DynamoLLM: Designing LLM Inference Clusters for Performance and Energy Efficiency

Best Paper Award 🏆
HPCA 2025

Open-source
production traces



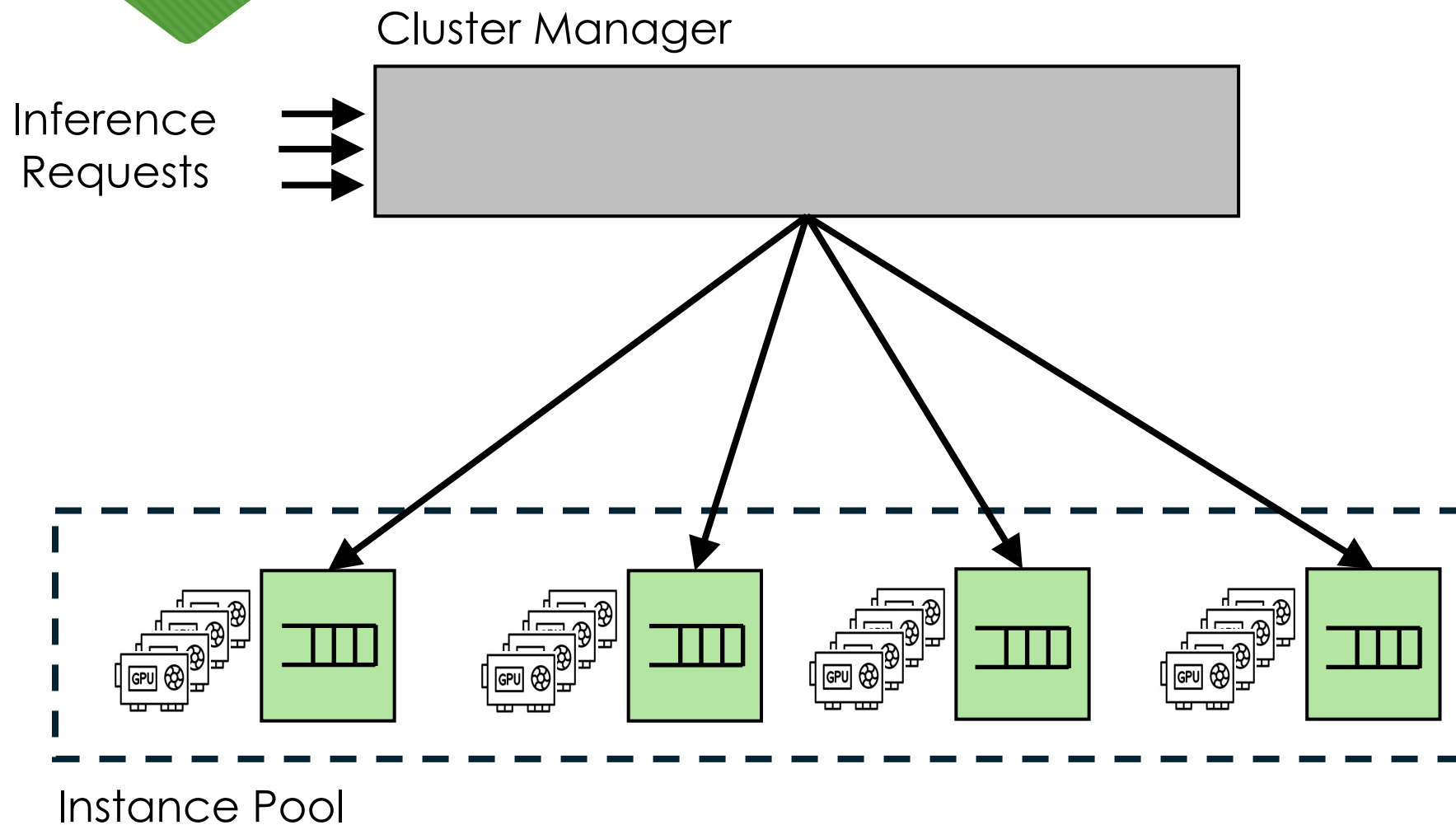
Jovan Stojkovic, Chaojie Zhang*, Íñigo Goiri*, Josep Torrellas, Esha Choukse*

University of Illinois at Urbana-Champaign, *Azure Research Systems

DynamoLLM: Energy-Management Framework for LLM Inference Clusters

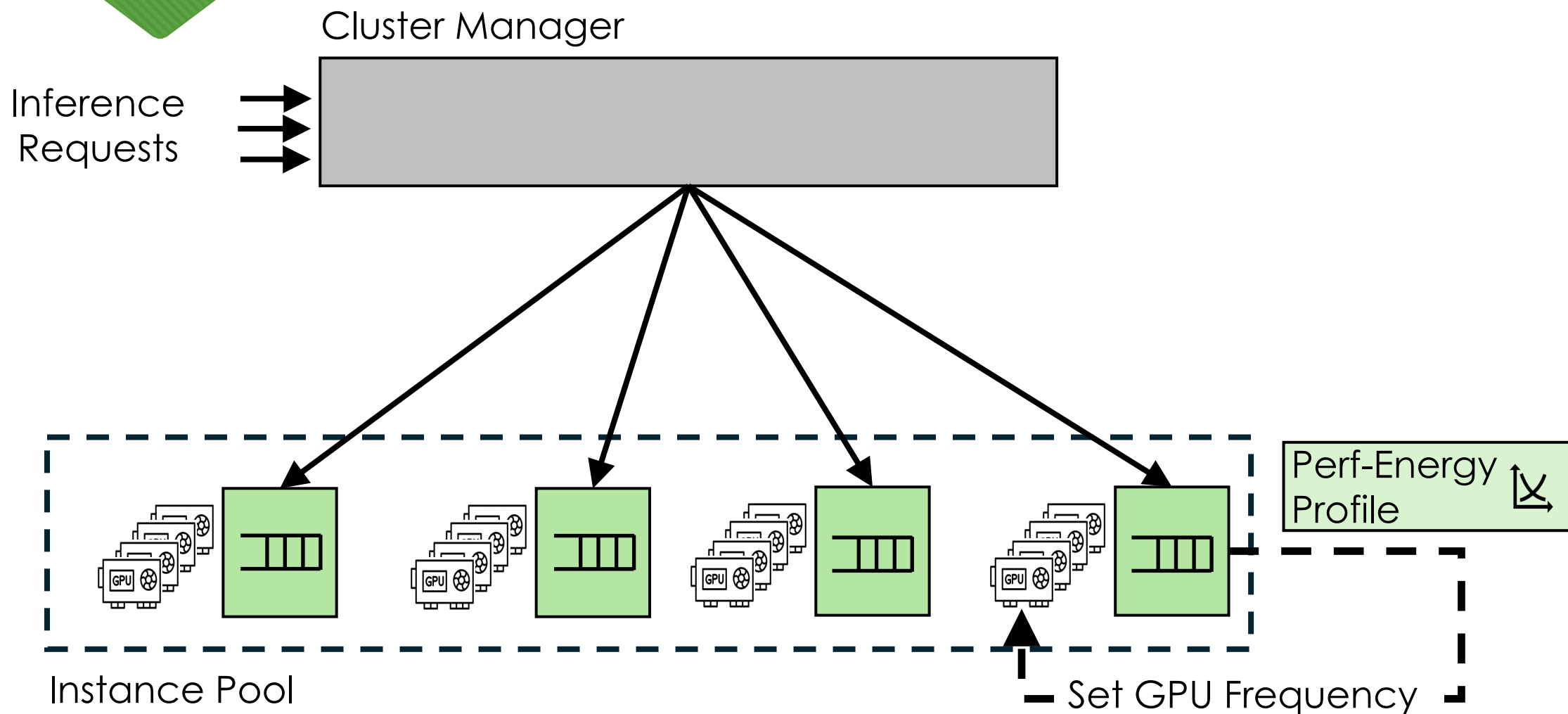
- DynamoLLM
 - i. Profile-driven energy configuration setting
 - ii. Instance pools for diverse workloads
 - iii. Hierarchical control for dynamic load

DynamoLLM: Energy-Management Framework for LLM Inference Clusters



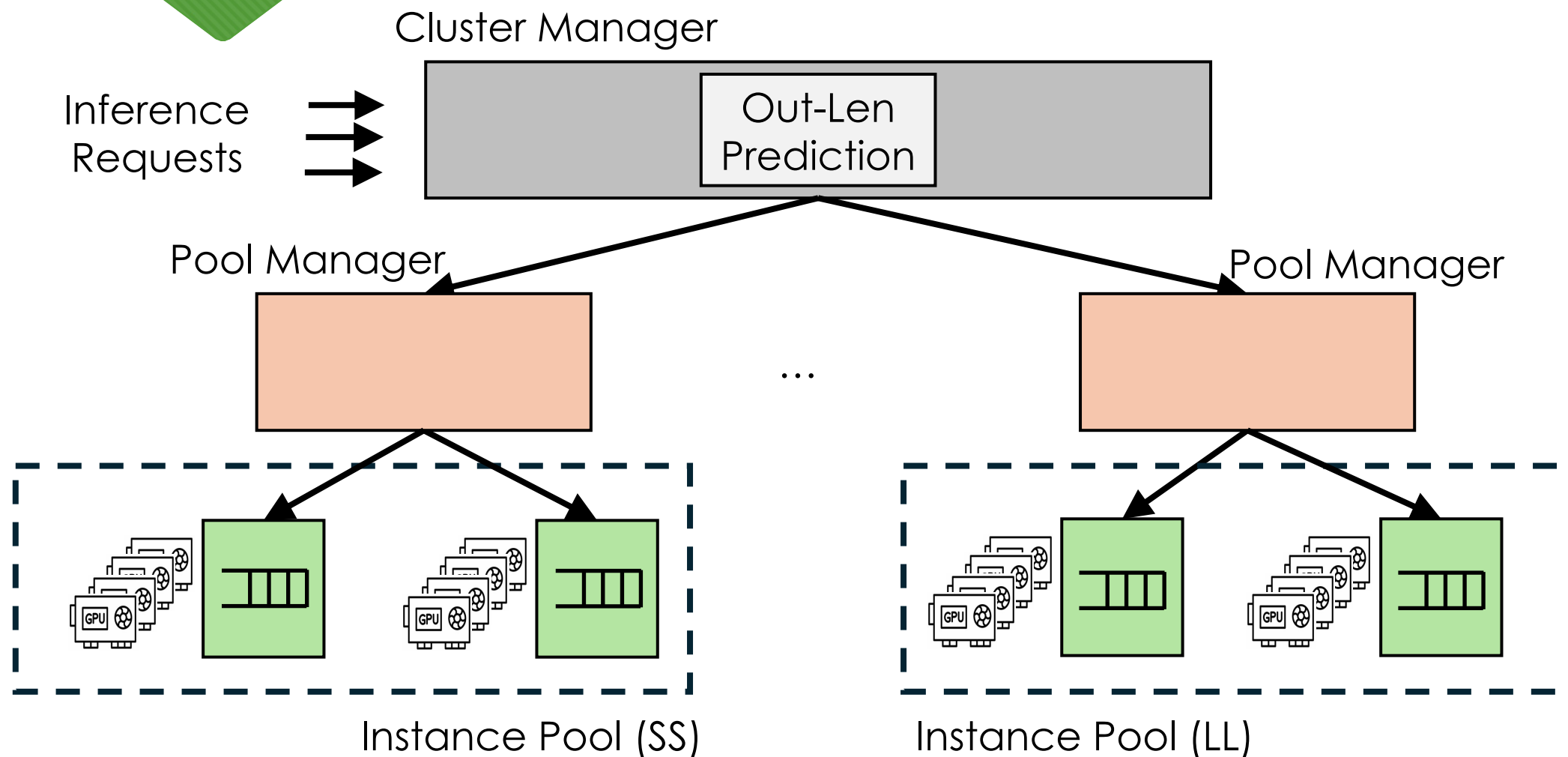
DynamoLLM:

1. Profile-Driven Energy Configuration Setting



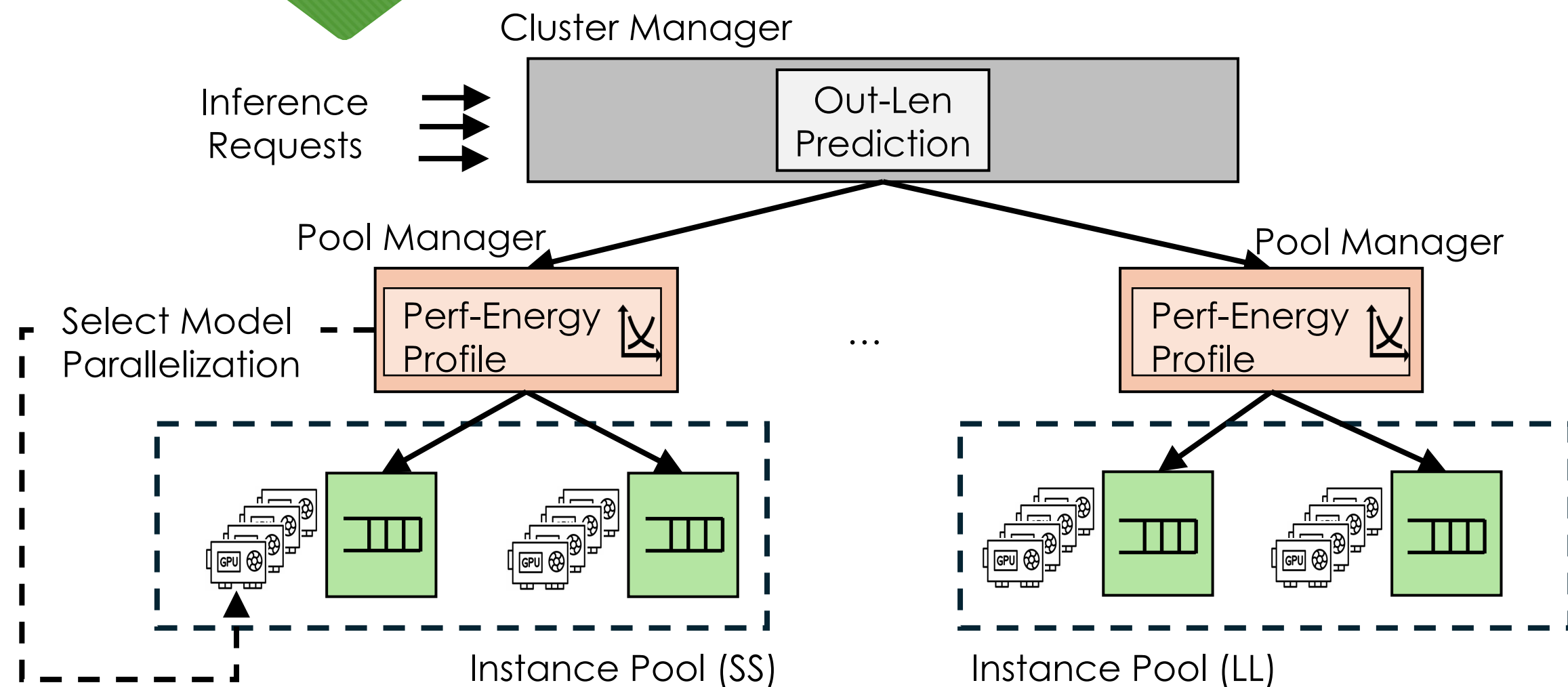
DynamoLLM:

2. Instance Pools for Diverse Workload



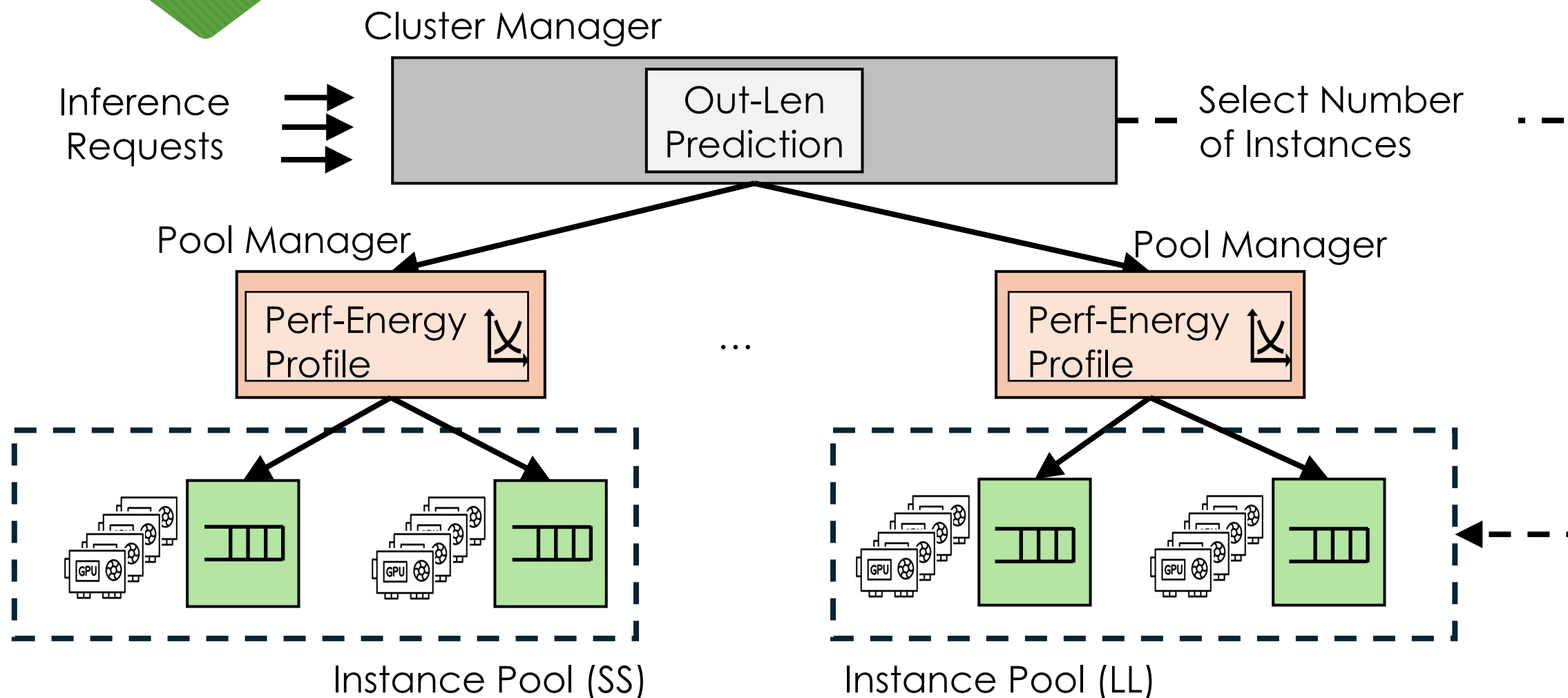
DynamoLLM:

2. Instance Pools for Diverse Workload



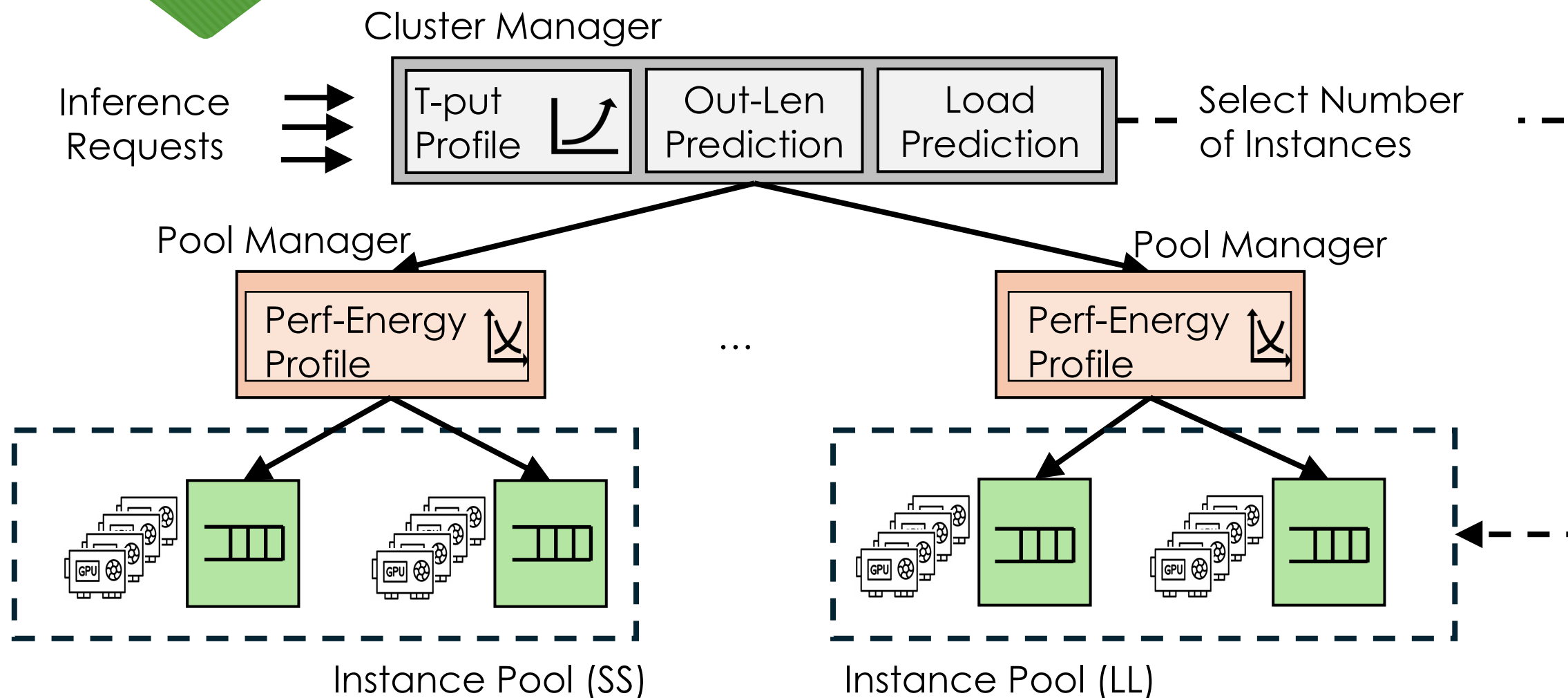
DynamoLLM:

3. Hierarchical Control for Dynamic Load



DynamoLLM:

3. Hierarchical Control for Dynamic Load



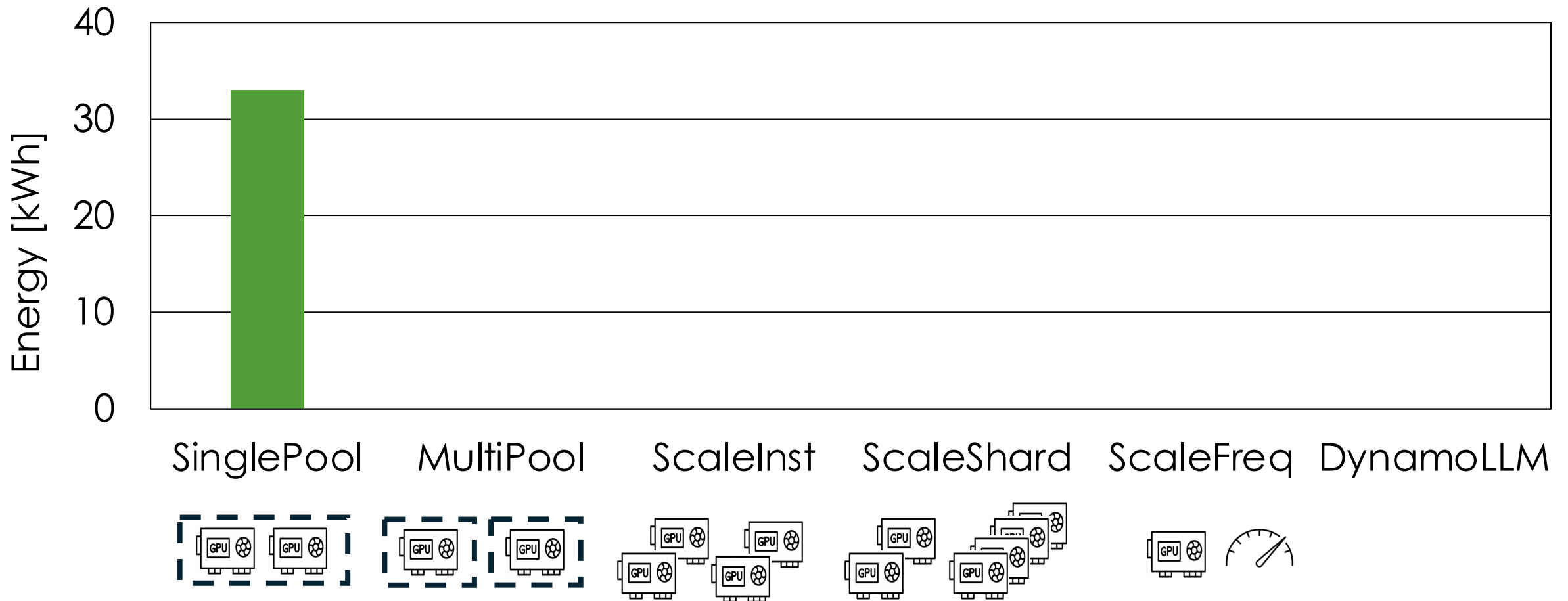
More in the paper

- MILP formulation for optimal energy-efficiency
- Techniques to reduce re-configuration overheads
 - Proactive VM creation
 - Graph-matching algorithm for re-sharding

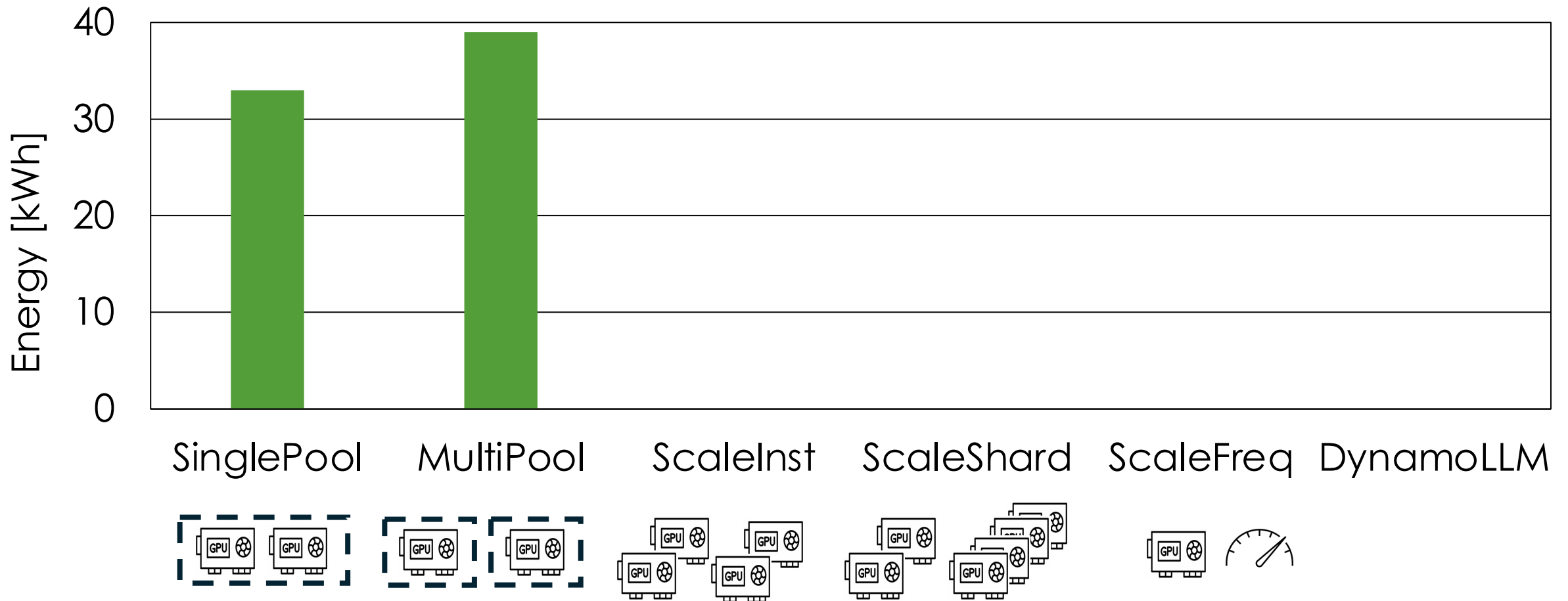
DynamoLLM Evaluation

- A peak hour (open-source) production traces from Azure
 - 12 x 8 H100 VMs for Baseline
- 1-week production traces from Azure
 - Simulate 40 x 8 H100 VMs for Baseline
- Llama2-70B model

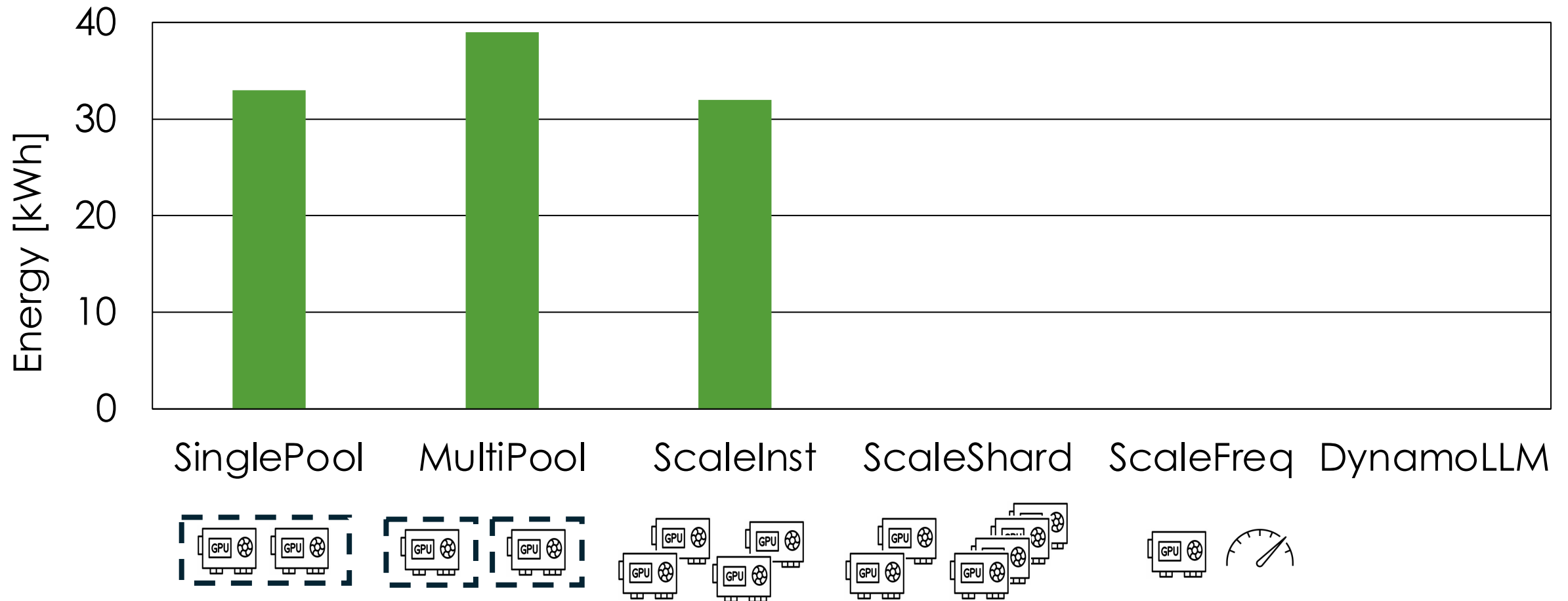
DynamoLLM significantly reduces energy!



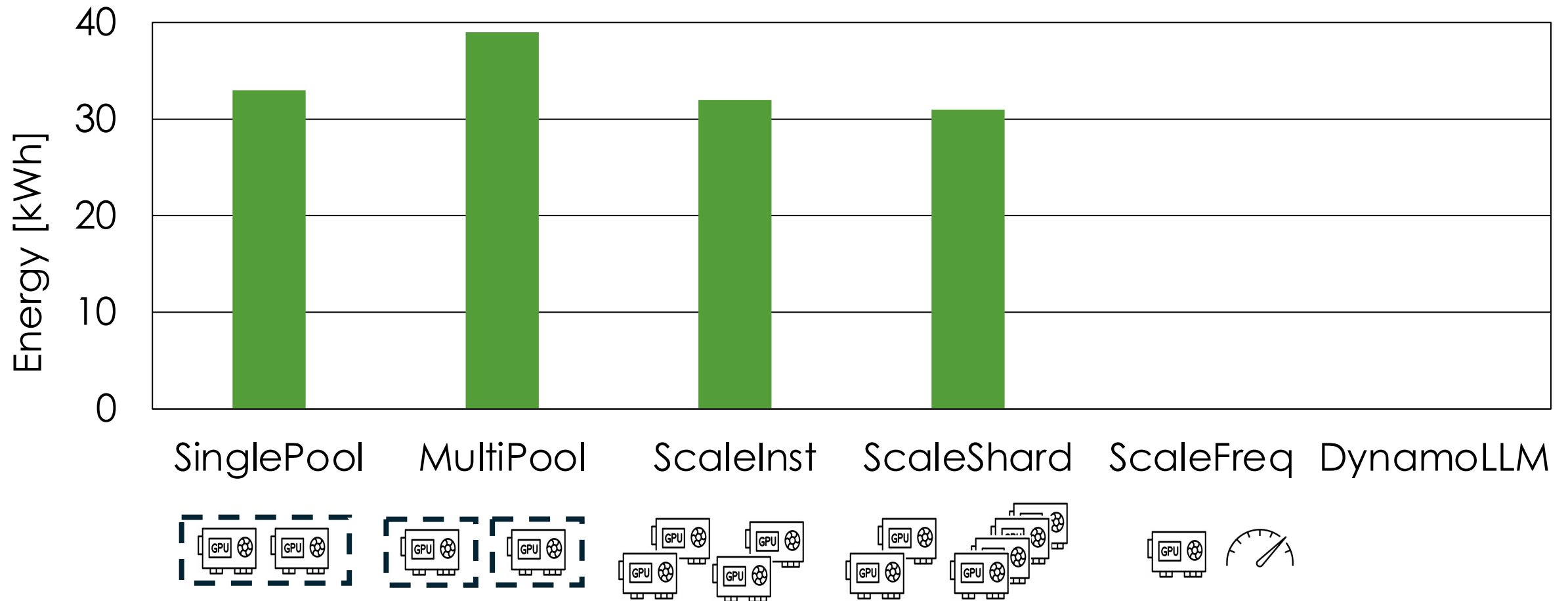
DynamoLLM significantly reduces energy!



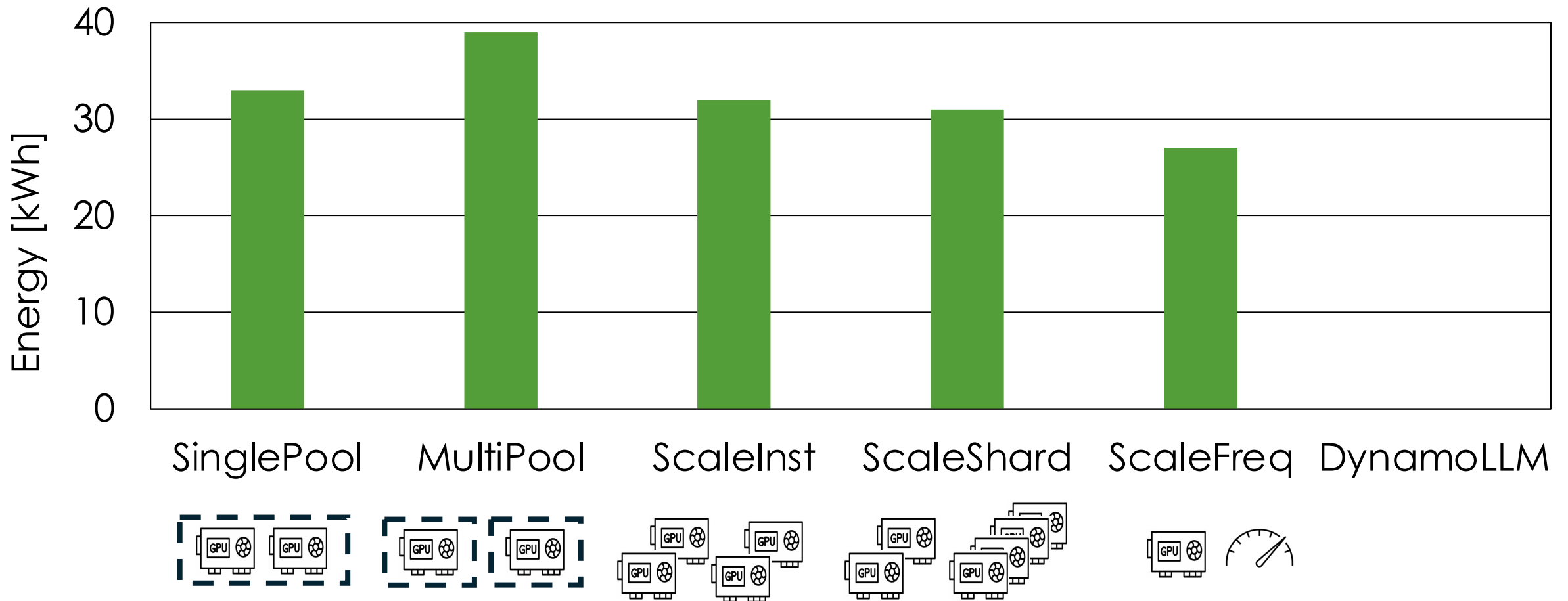
DynamoLLM significantly reduces energy!



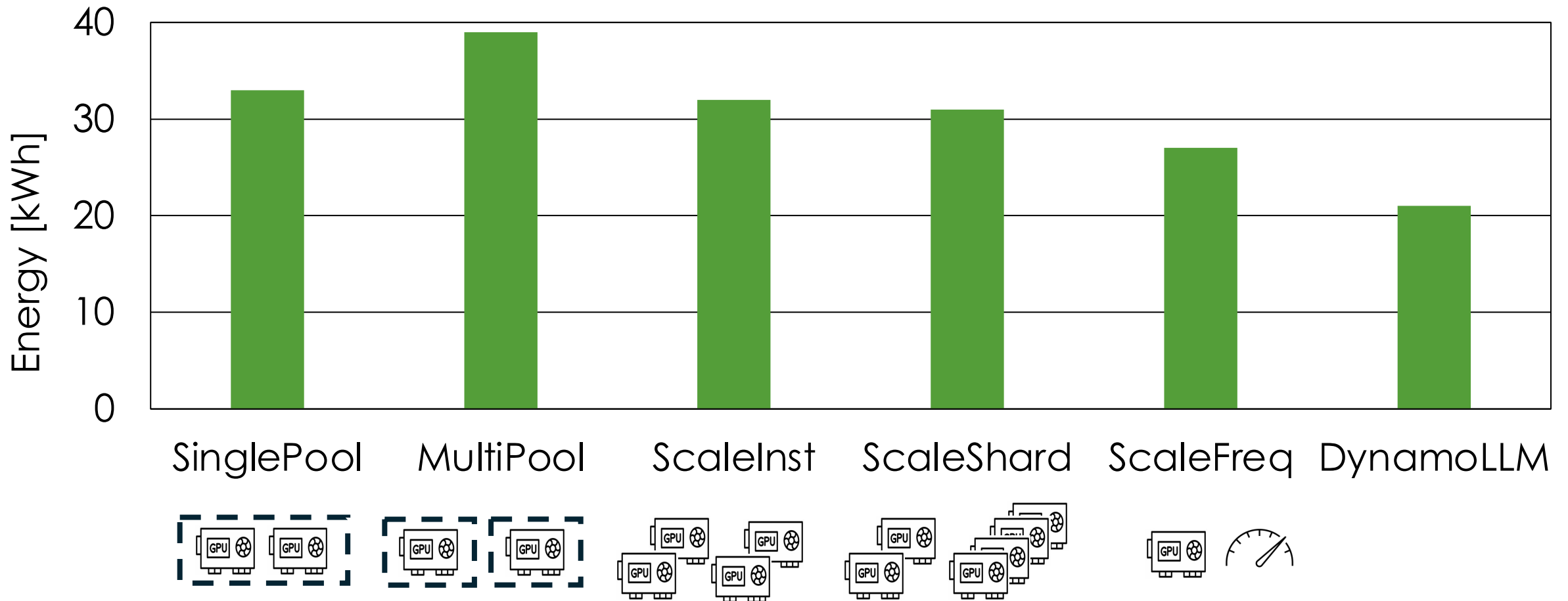
DynamoLLM significantly reduces energy!



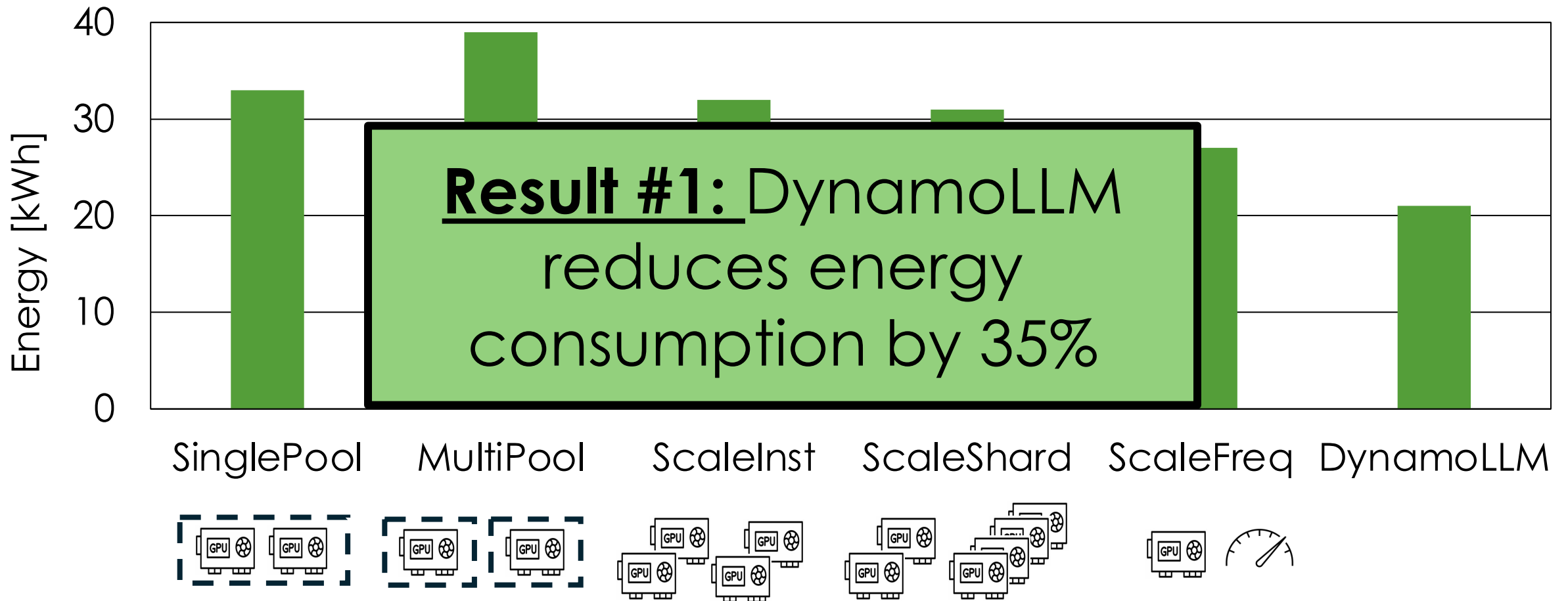
DynamoLLM significantly reduces energy!



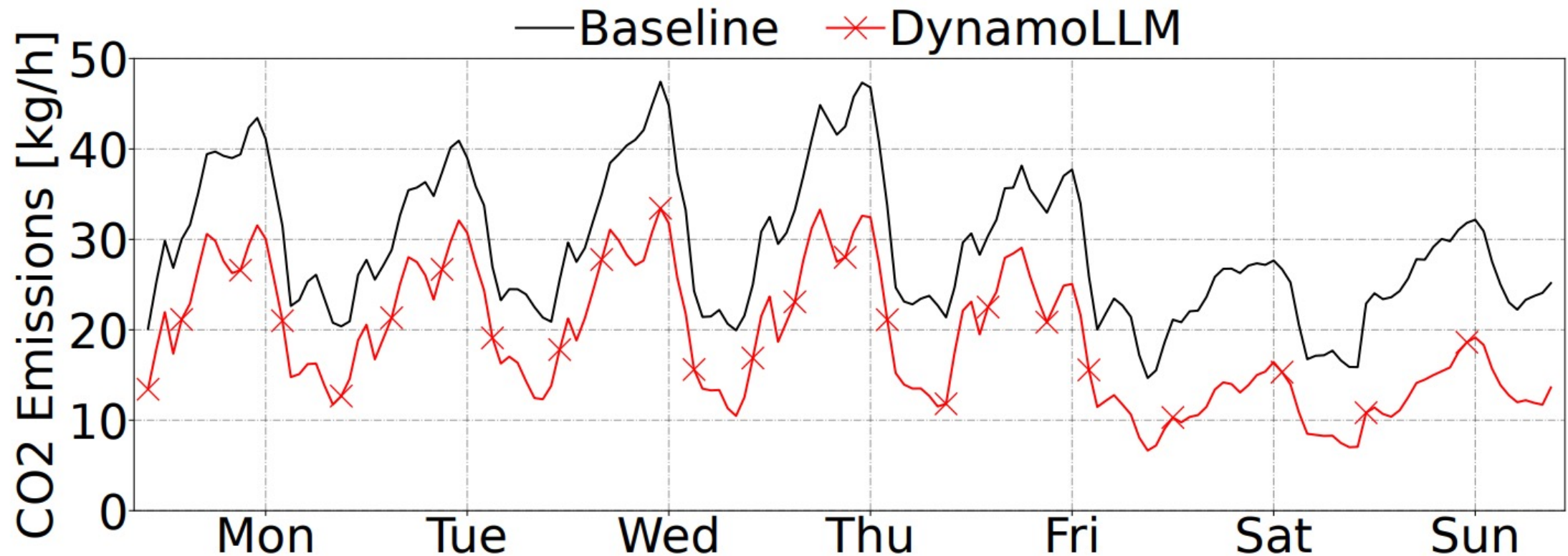
DynamoLLM significantly reduces energy!



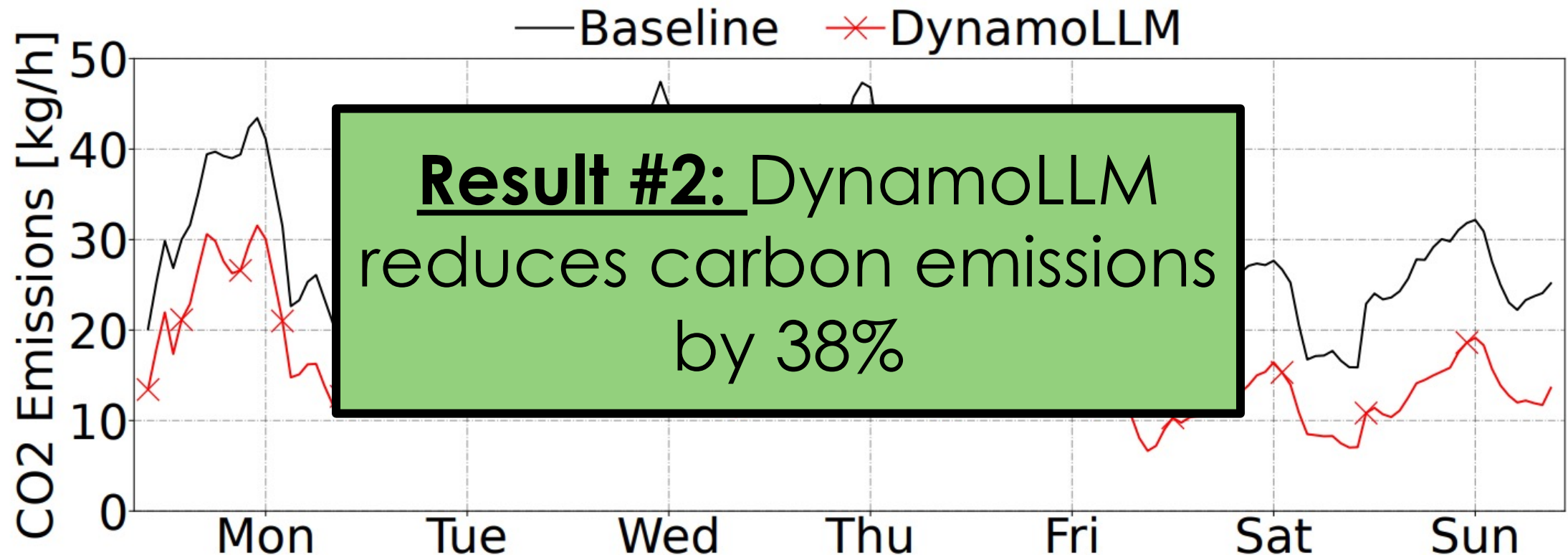
DynamoLLM significantly reduces energy!



DynamoLLM significantly reduces carbon!



DynamoLLM significantly reduces carbon!

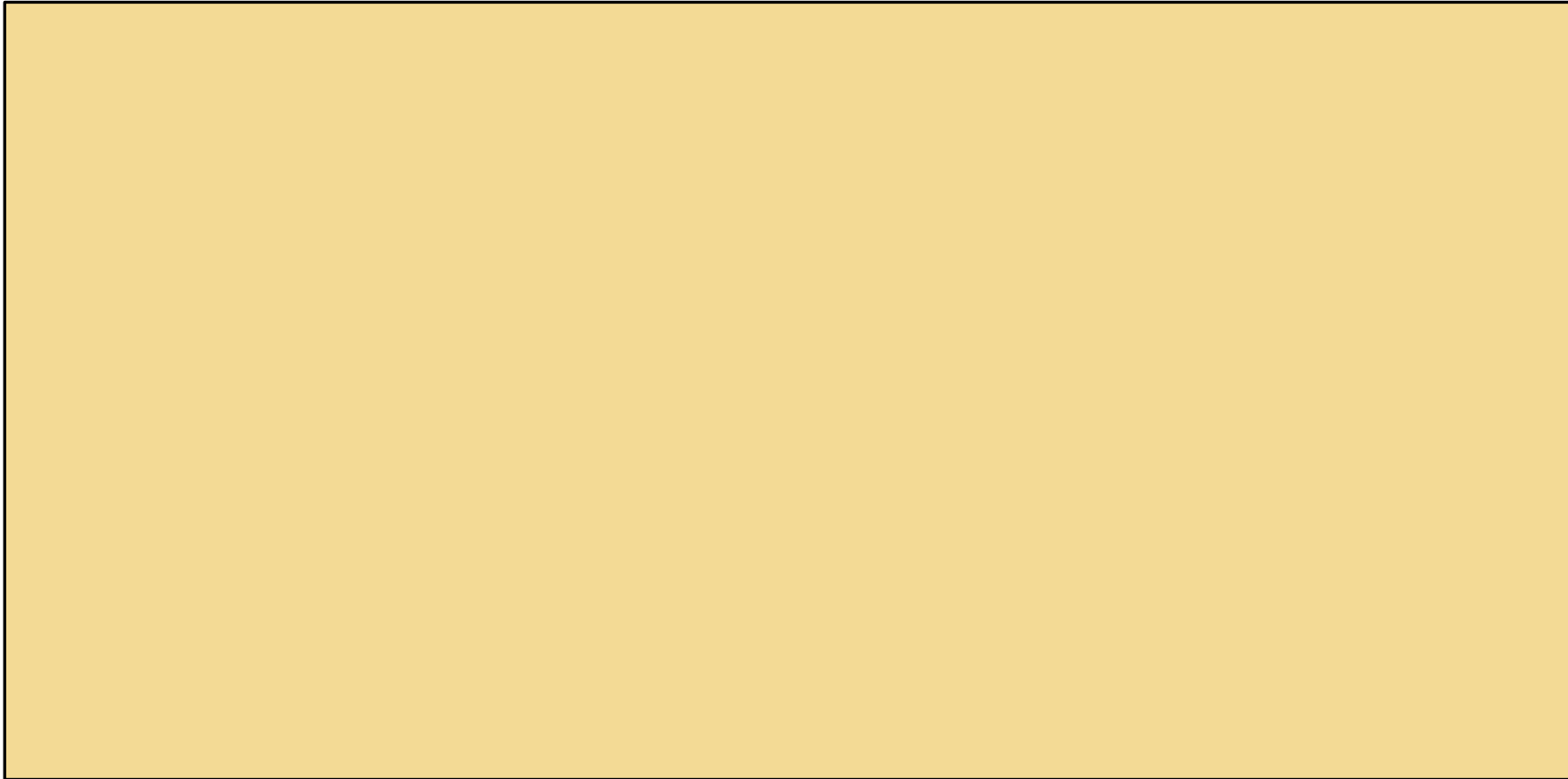


Goal: Make LLMs Thermal- and Power-Efficient

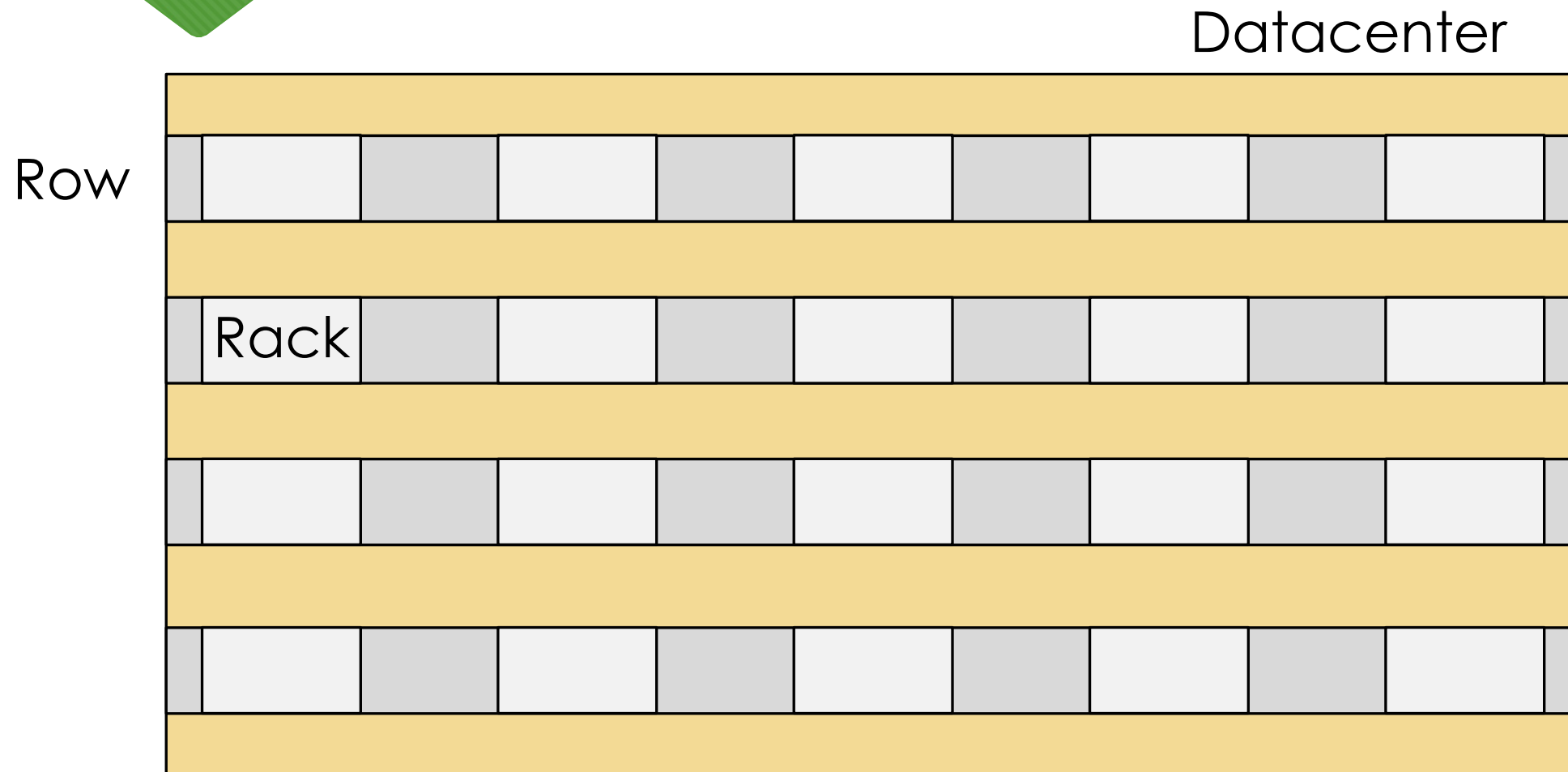
- Challenges
 - Cooling management
 - Power management
 - Workload heterogeneity

Challenge #1: Cooling Management

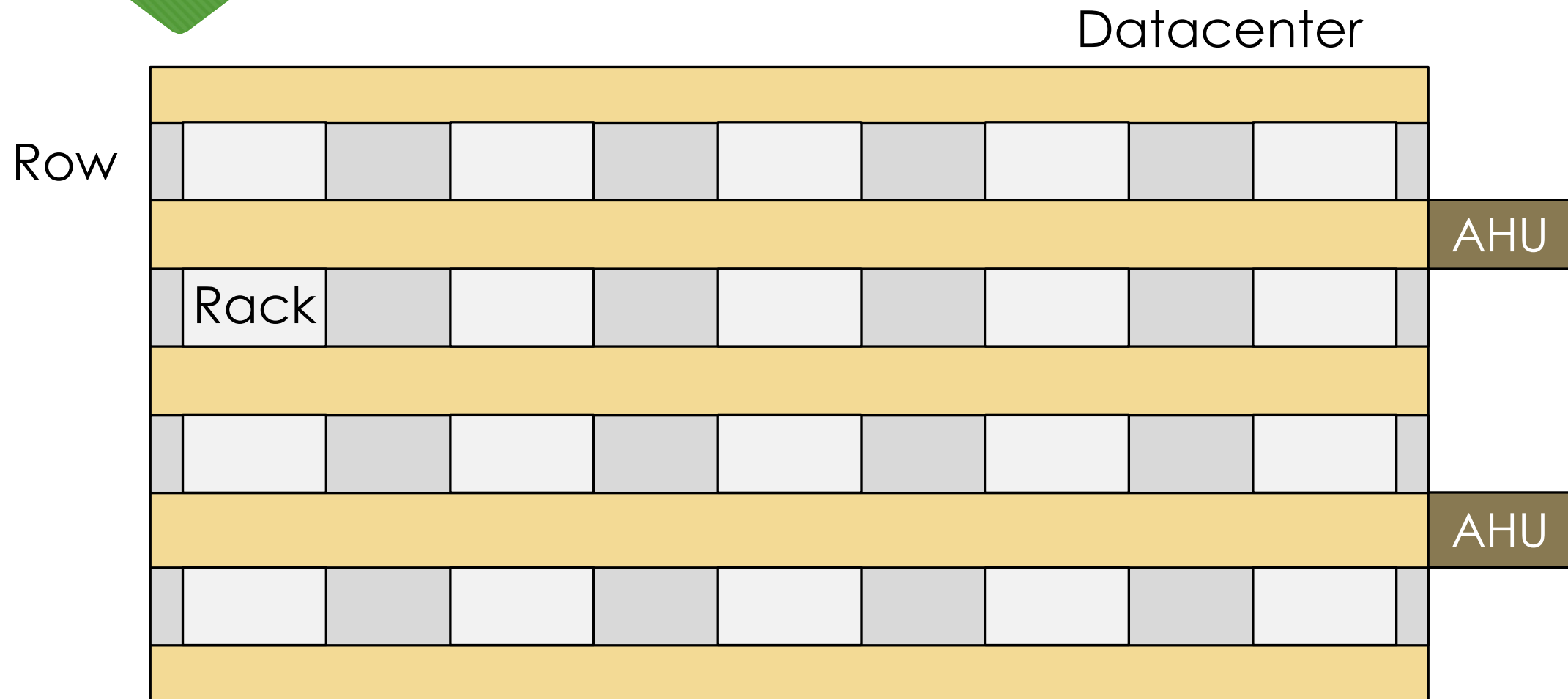
Datacenter



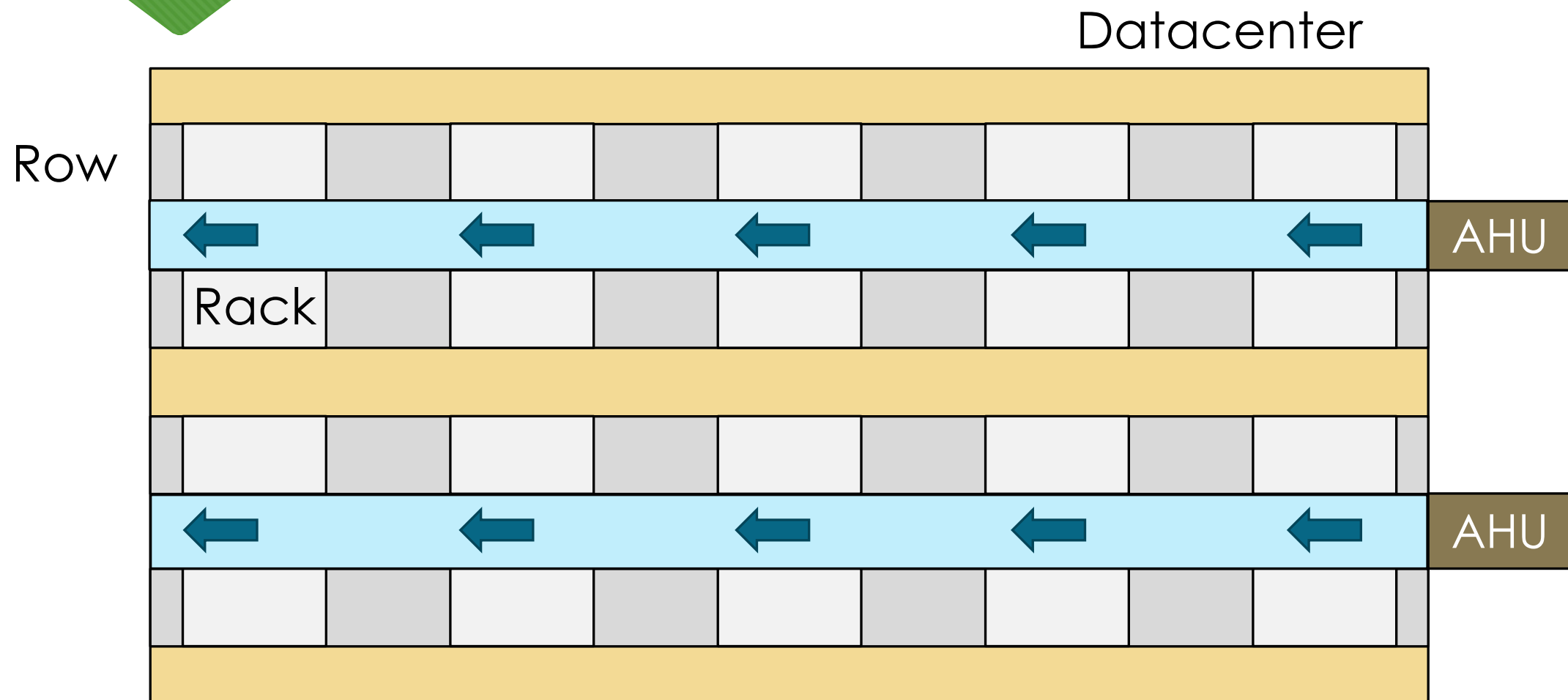
Challenge #1: Cooling Management



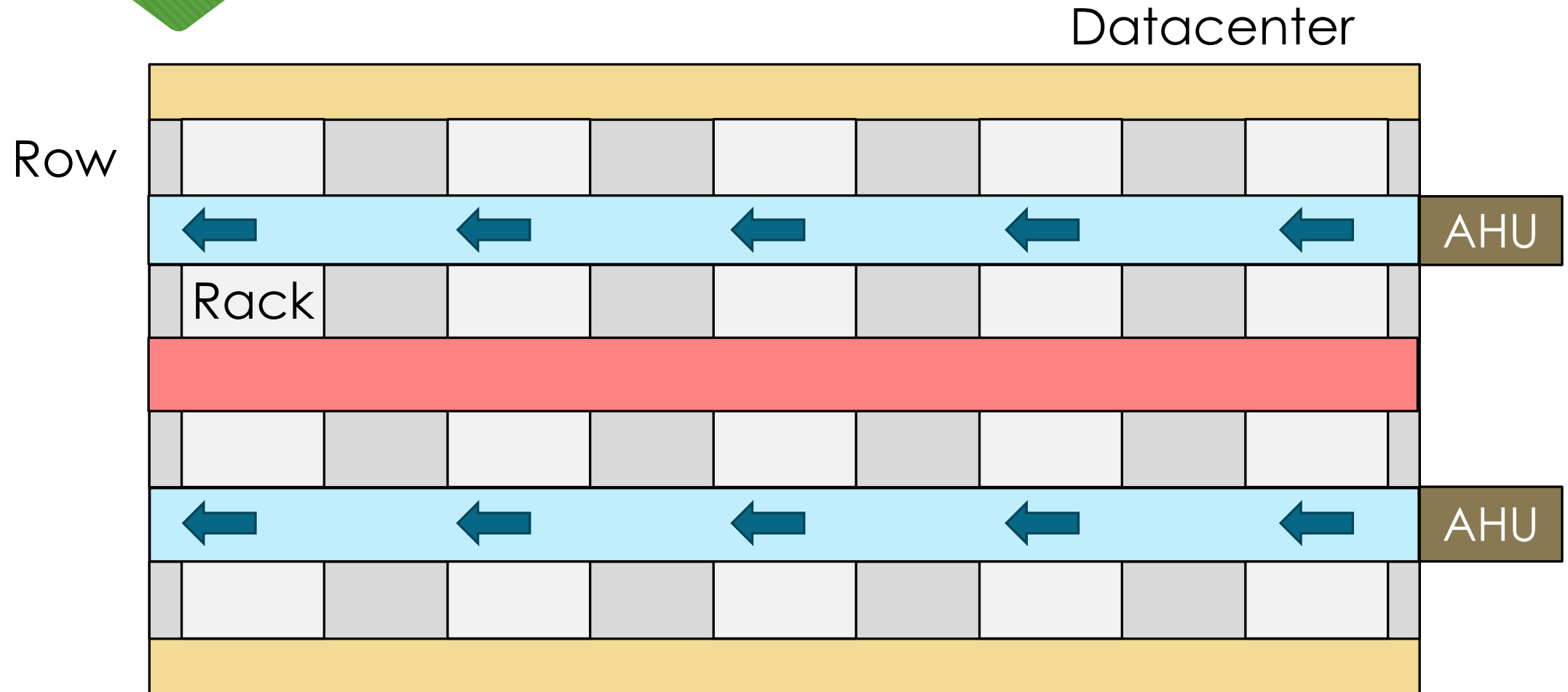
Challenge #1: Cooling Management



Challenge #1: Cooling Management



Challenge #1: Cooling Management



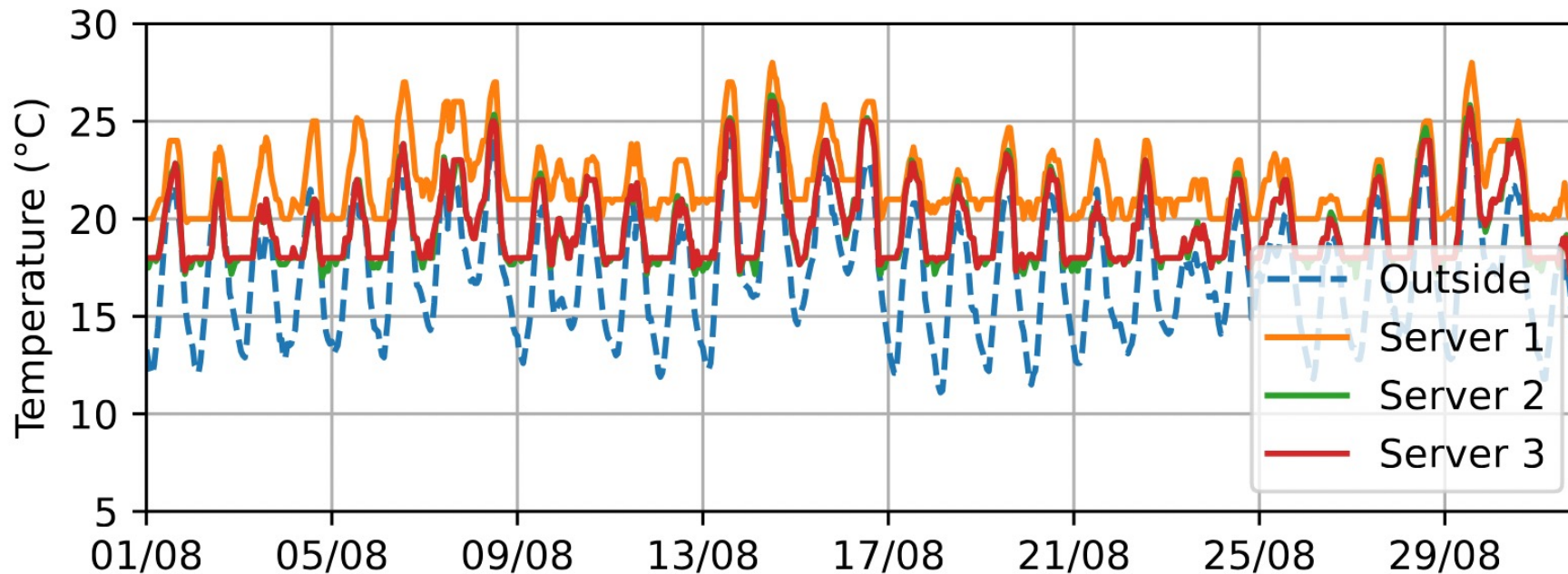
Challenge #1: Cooling Management

- Datacenter operators provision cooling infrastructure to sustain **peak load**
 - Enough airflow in each aisle to prevent heat recirculation
 - Enough cooling to lower the temperature within operating conditions

Challenge #1: Thermal Hotspots!

○ Temperature of a GPU server depends on:

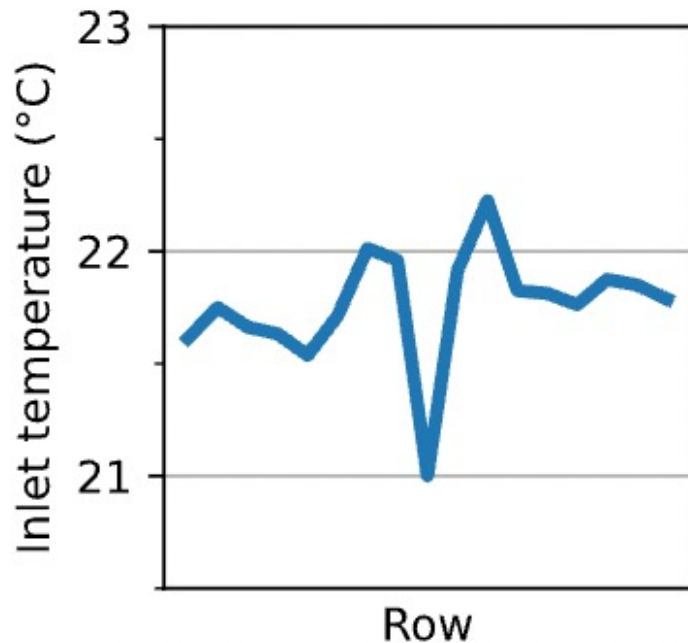
1. Outside temperature



Challenge #1: Thermal Hotspots!

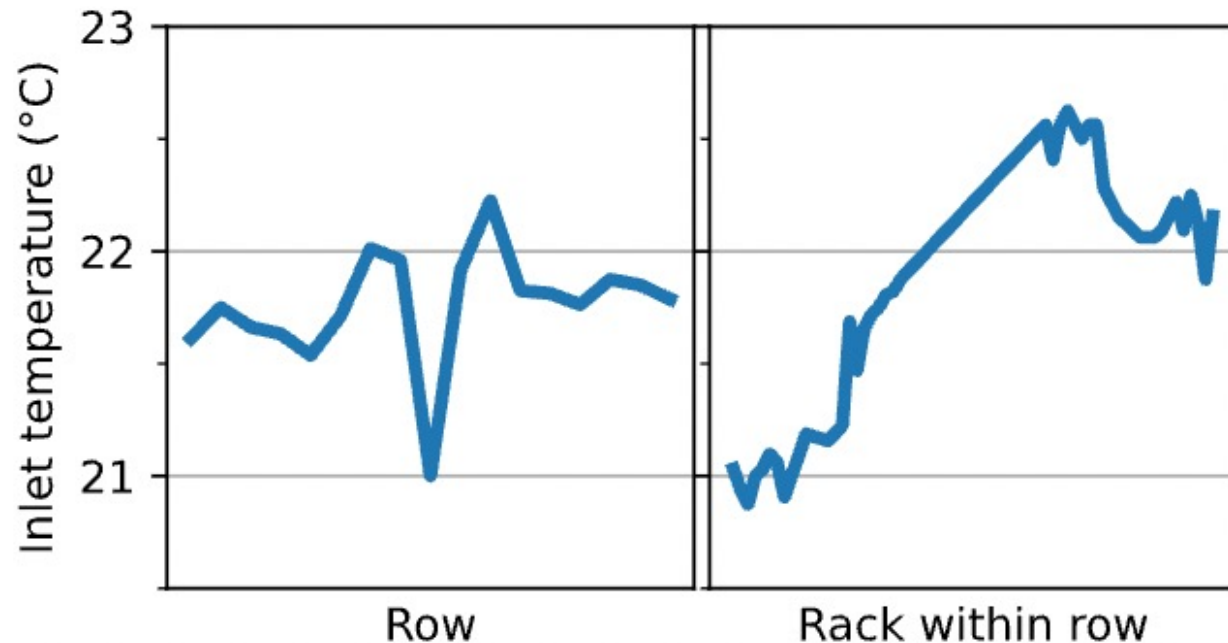
○ Temperature of a GPU server depends on:

2. Datacenter layout



Challenge #1: Thermal Hotspots!

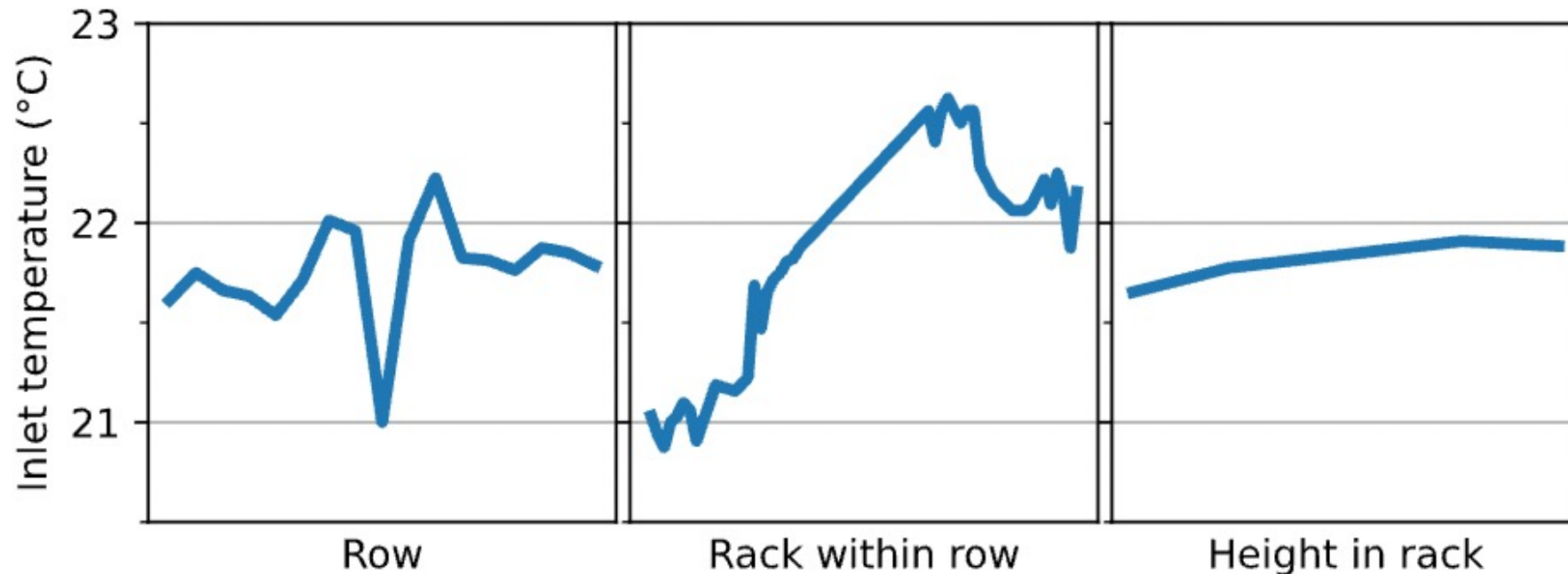
- Temperature of a GPU server depends on:
 - 2. Datacenter layout



Challenge #1: Thermal Hotspots!

○ Temperature of a GPU server depends on:

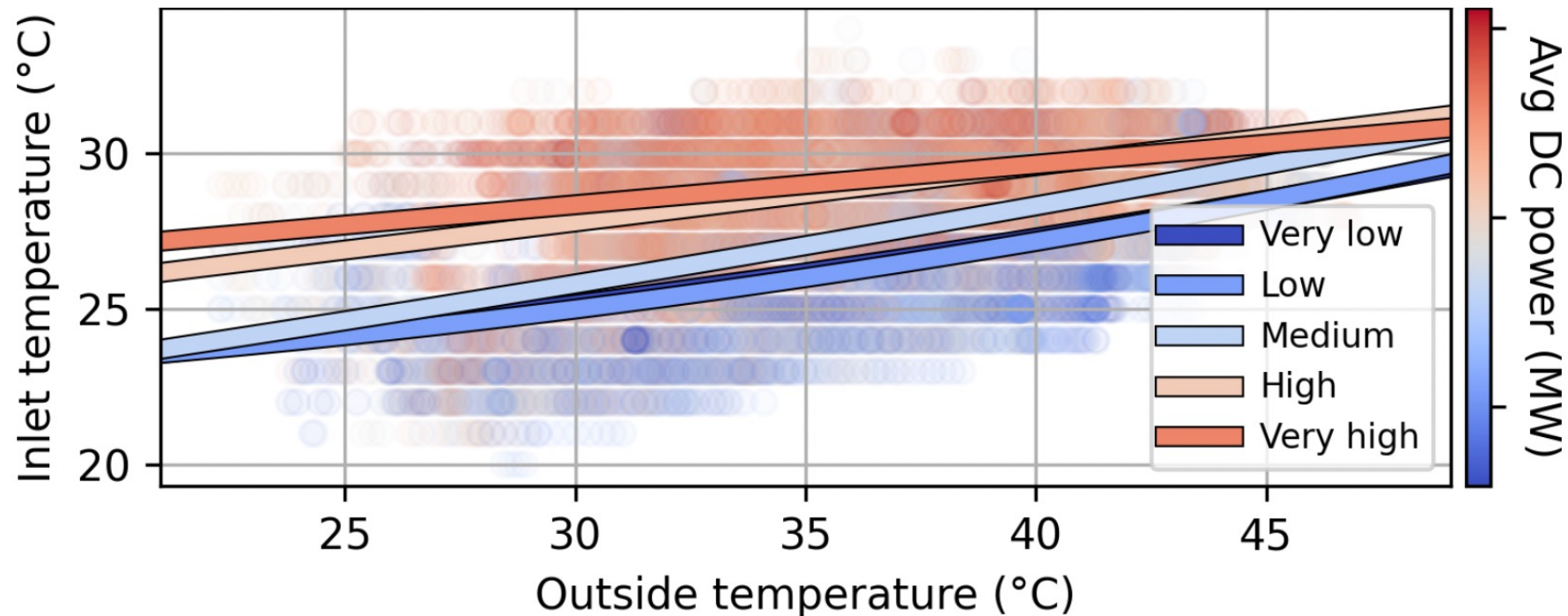
2. Datacenter layout



Challenge #1: Thermal Hotspots!

○ Temperature of a GPU server depends on:

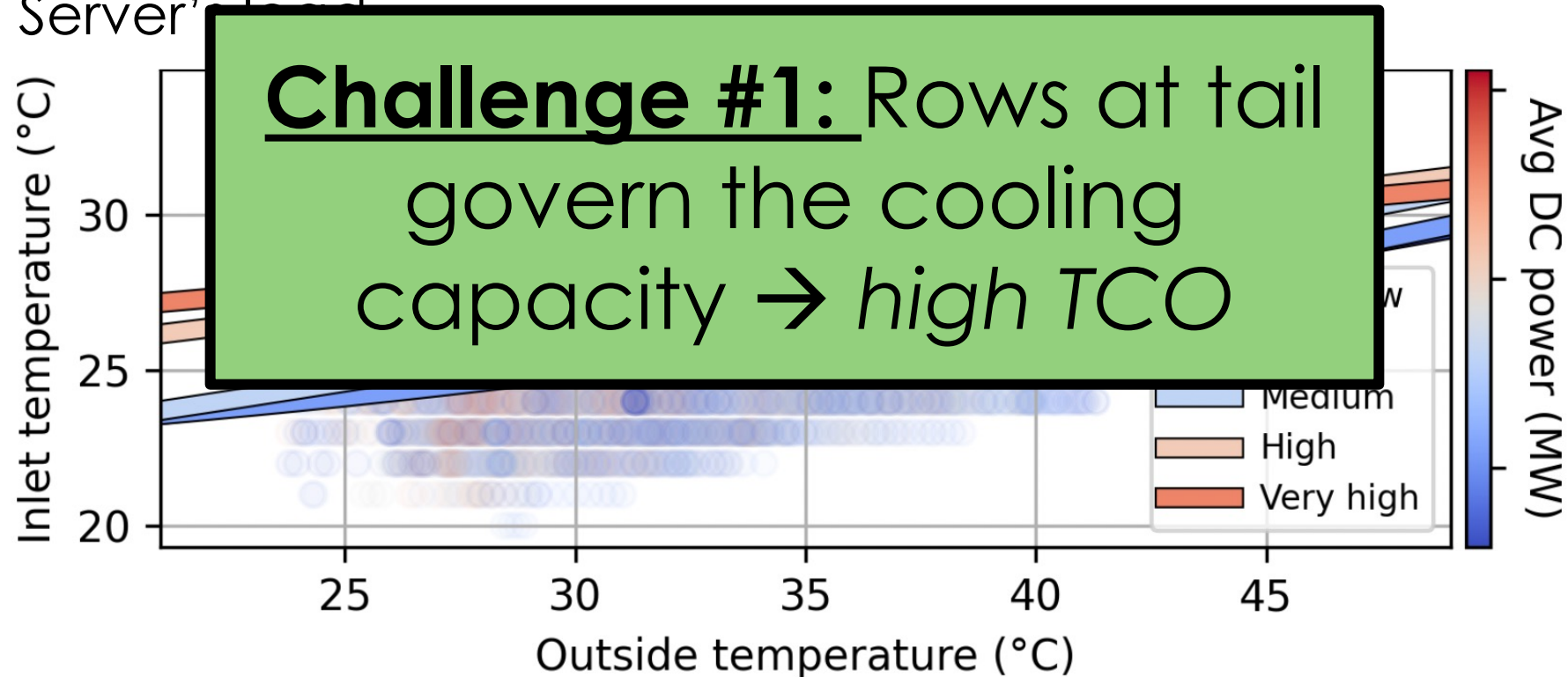
3. Server's load



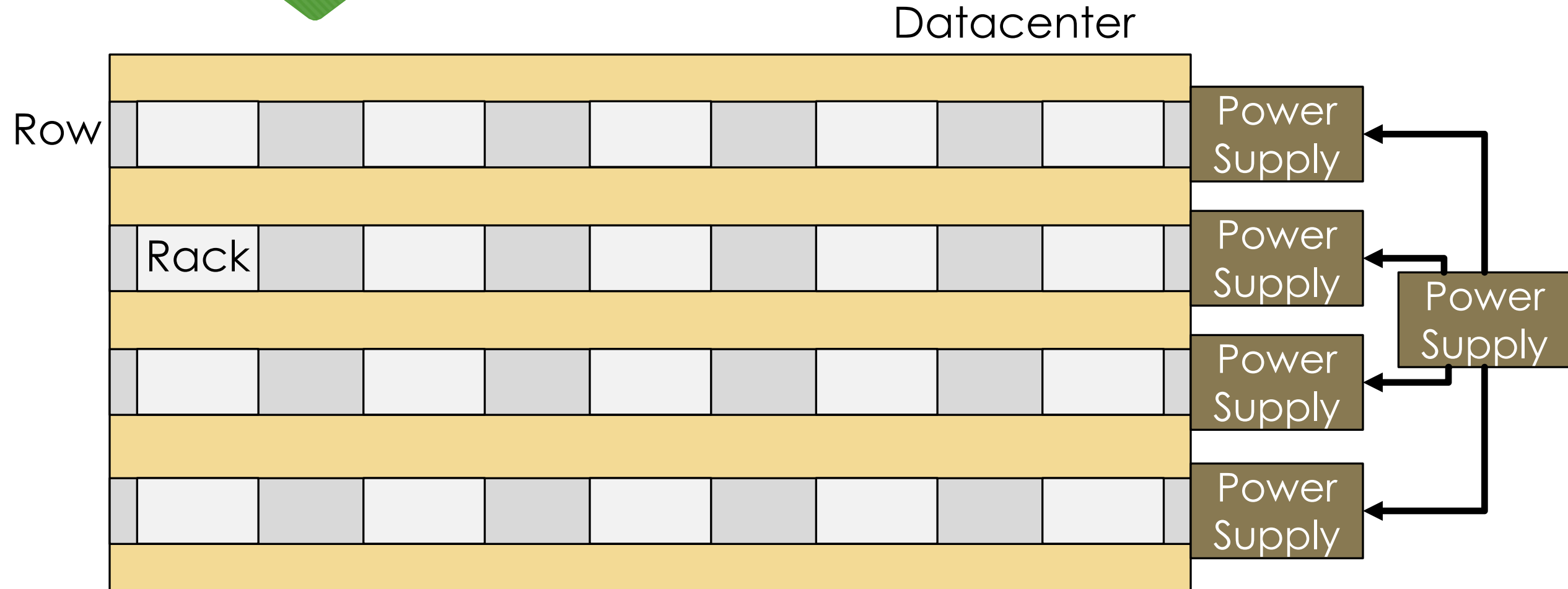
Challenge #1: Thermal Hotspots!

○ Temperature of a GPU server depends on:

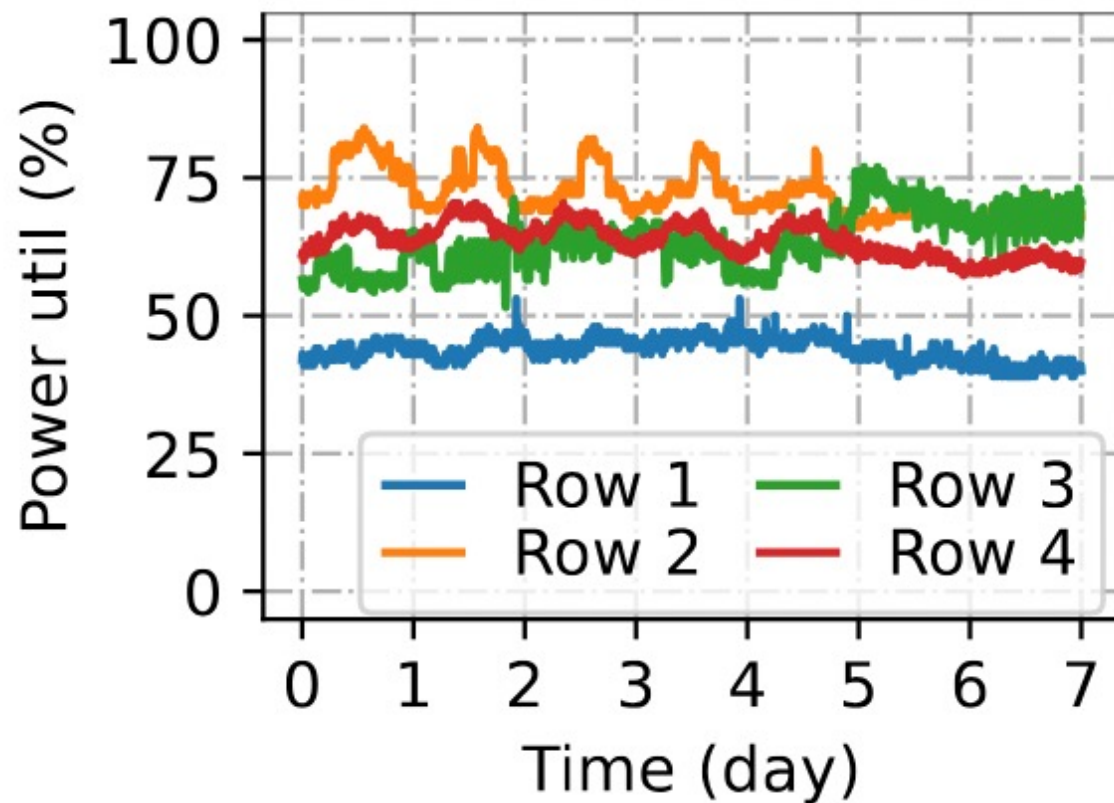
3. Server's load



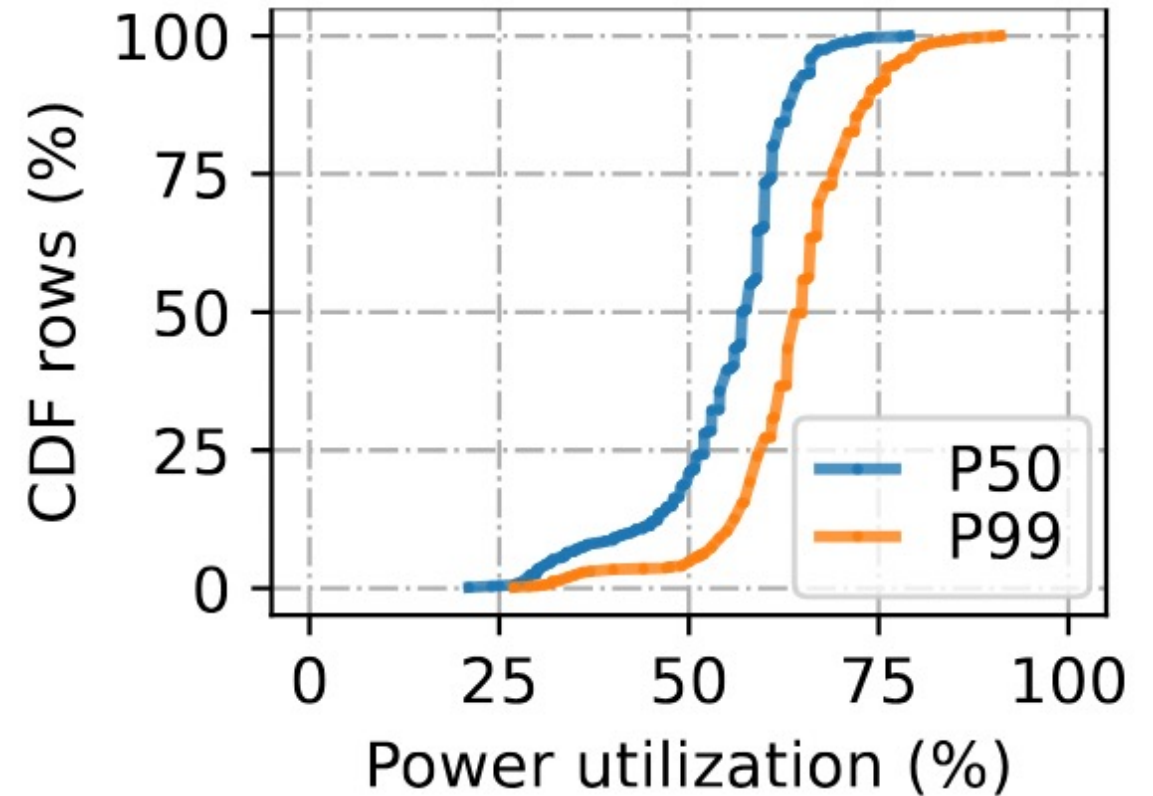
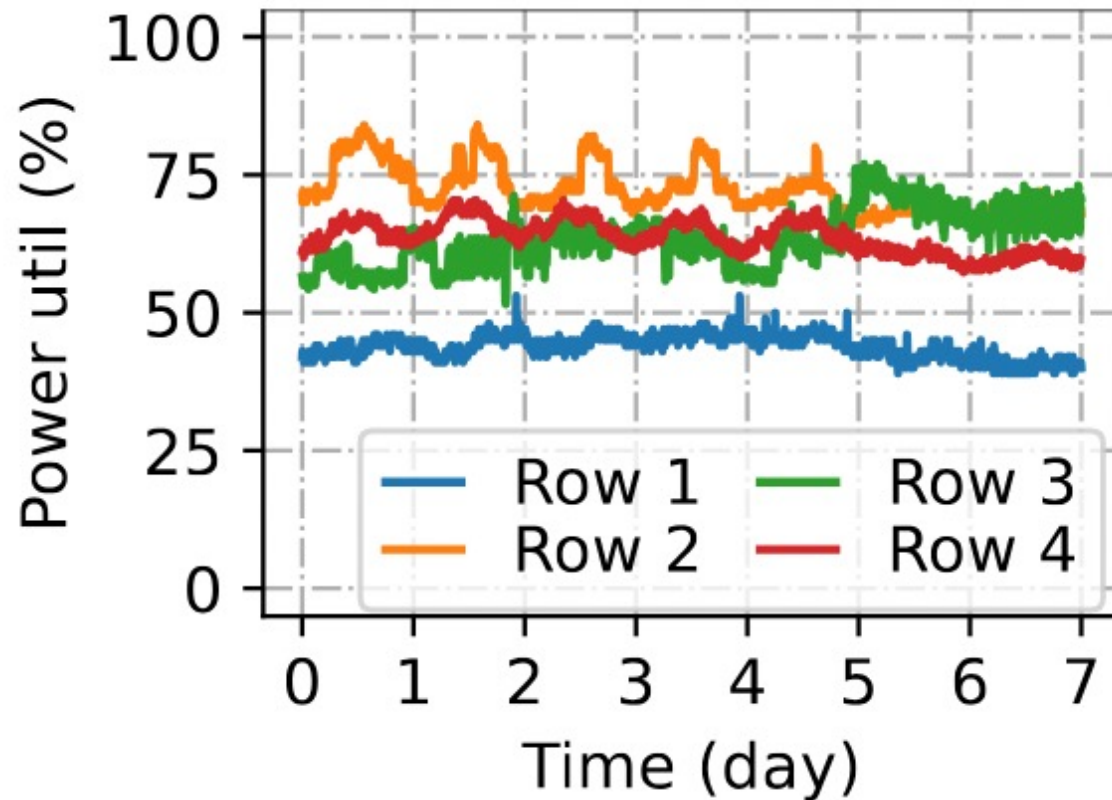
Challenge #2: Power Management



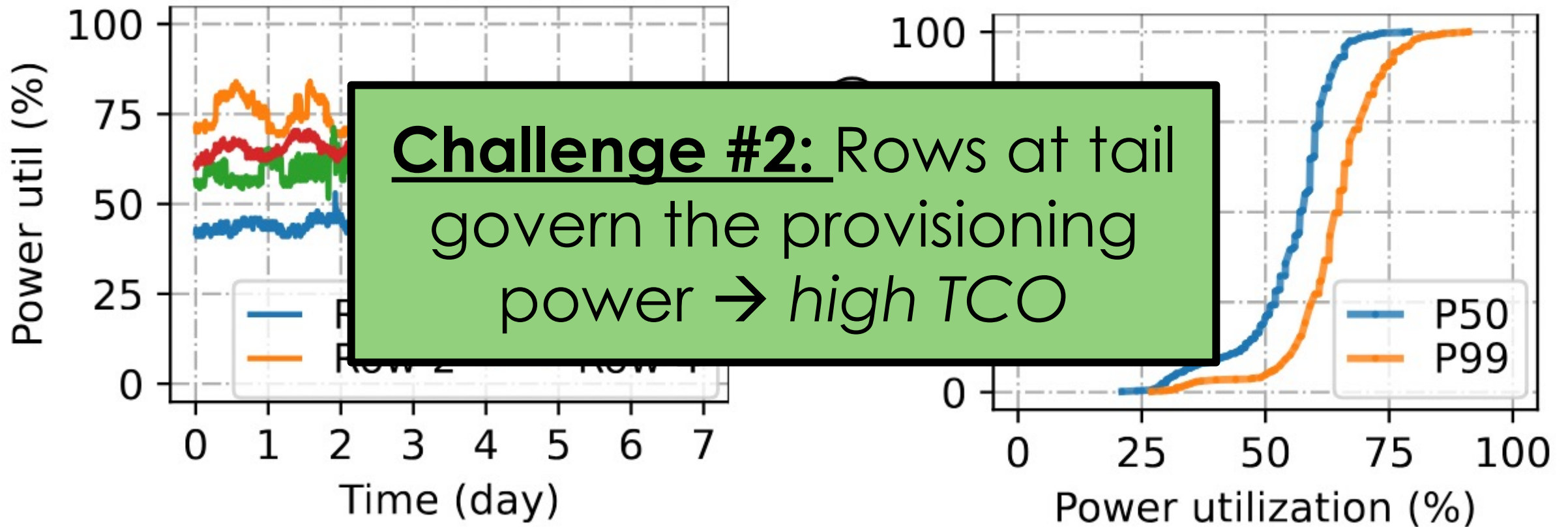
Challenge #2: Power Imbalance!



Challenge #2: Power Imbalance!



Challenge #2: Power Imbalance!



Challenge #3: Heterogeneous Workloads

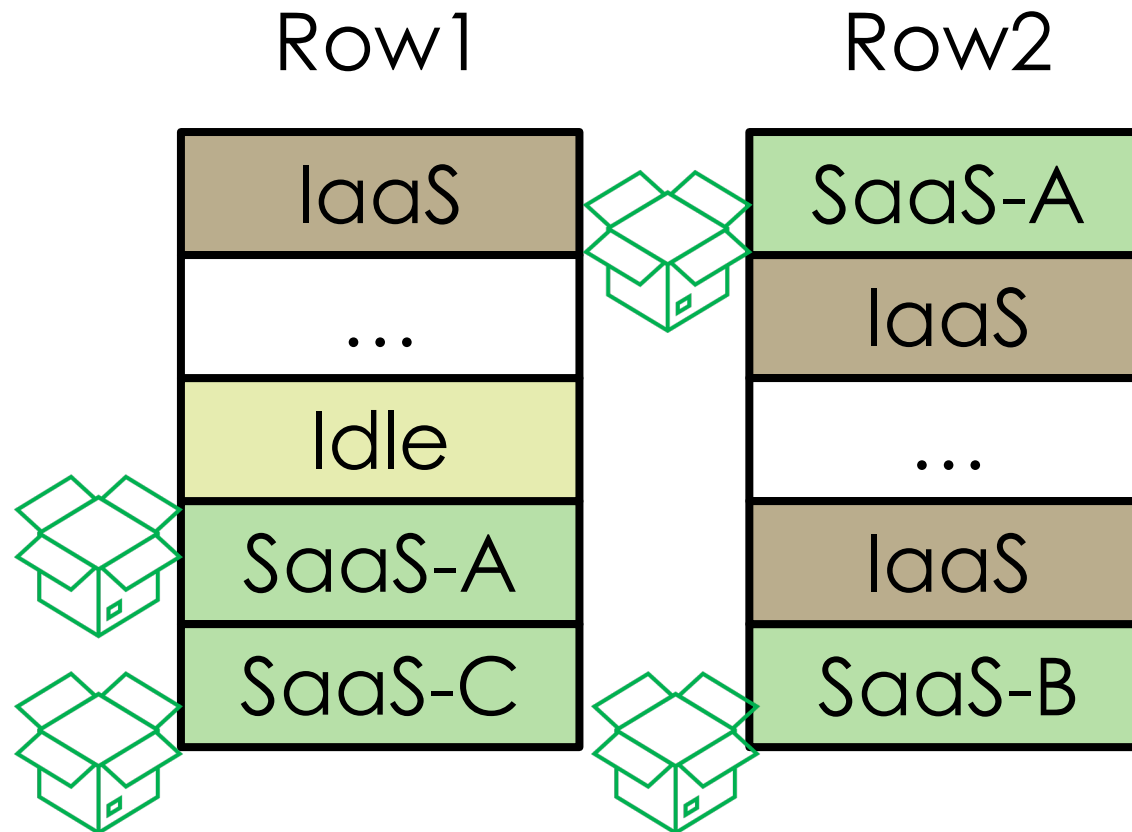
Row1

IaaS
...
Idle
SaaS-A
SaaS-C

Row2

SaaS-A
IaaS
...
IaaS
SaaS-B

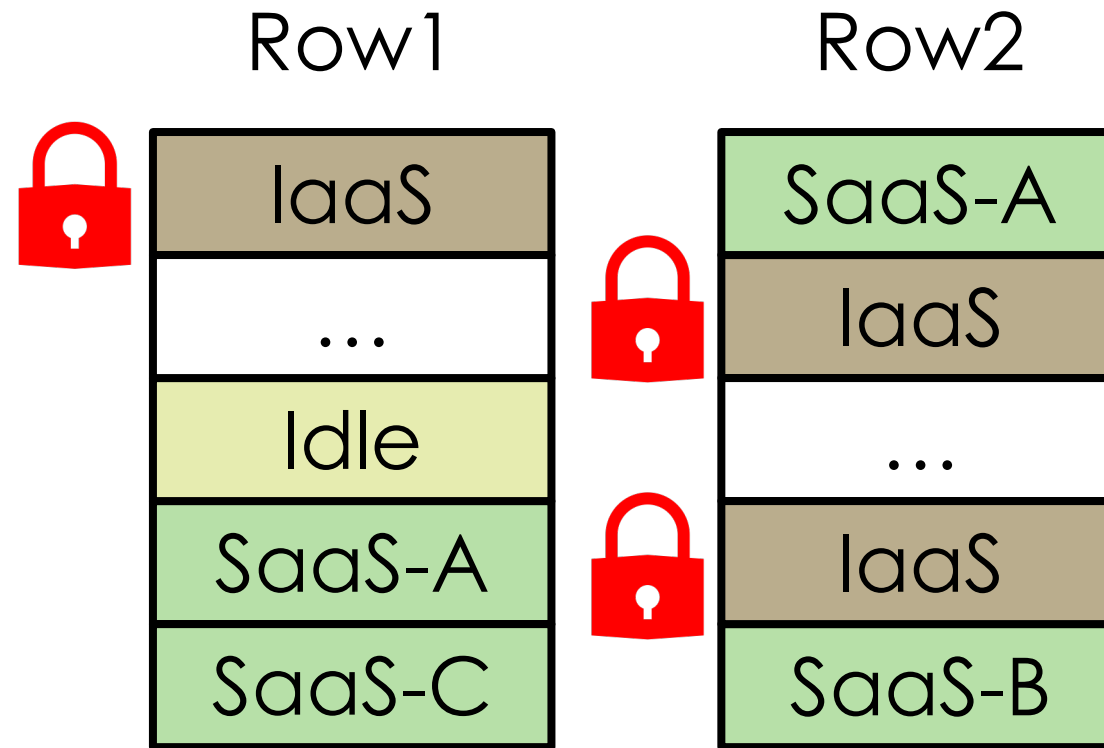
Challenge #3: Heterogeneous Workloads



SaaS services

First-party
Good visibility
Good flexibility

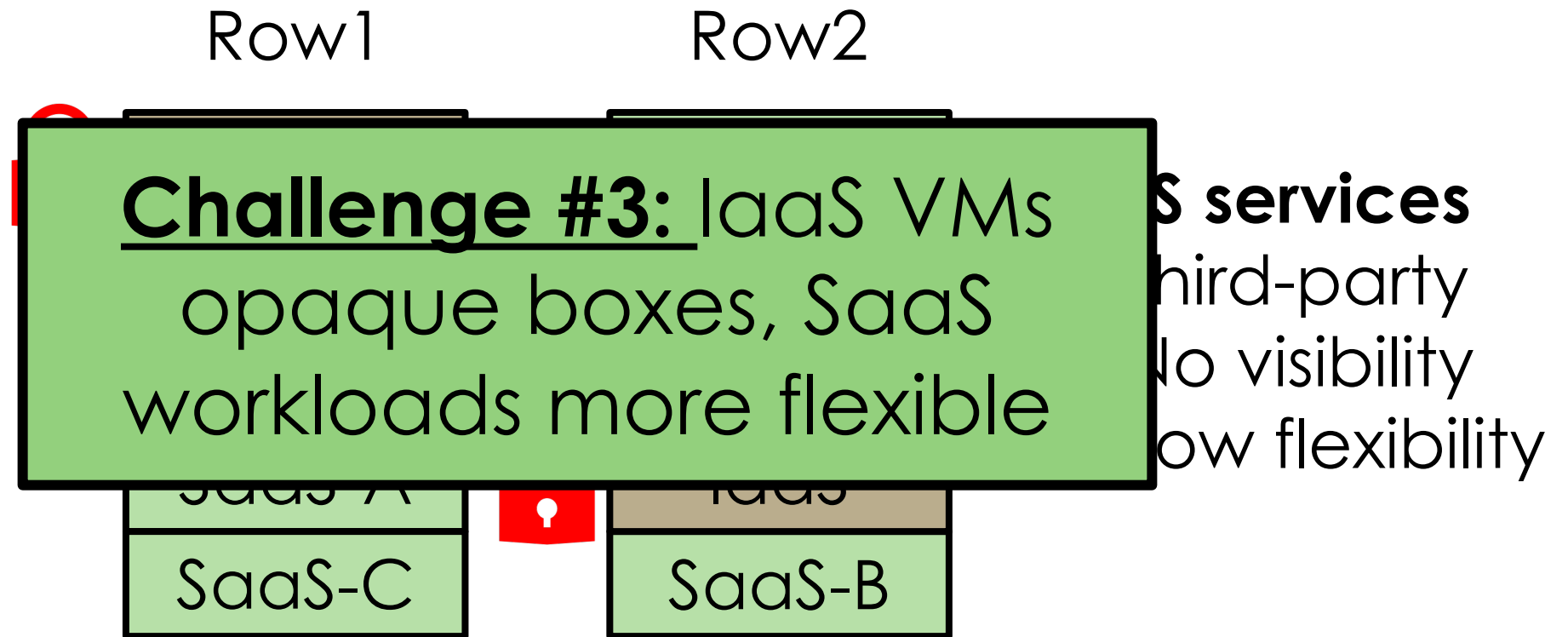
Challenge #3: Heterogeneous Workloads



IaaS services

Third-party
No visibility
Low flexibility

Challenge #3: Heterogeneous Workloads

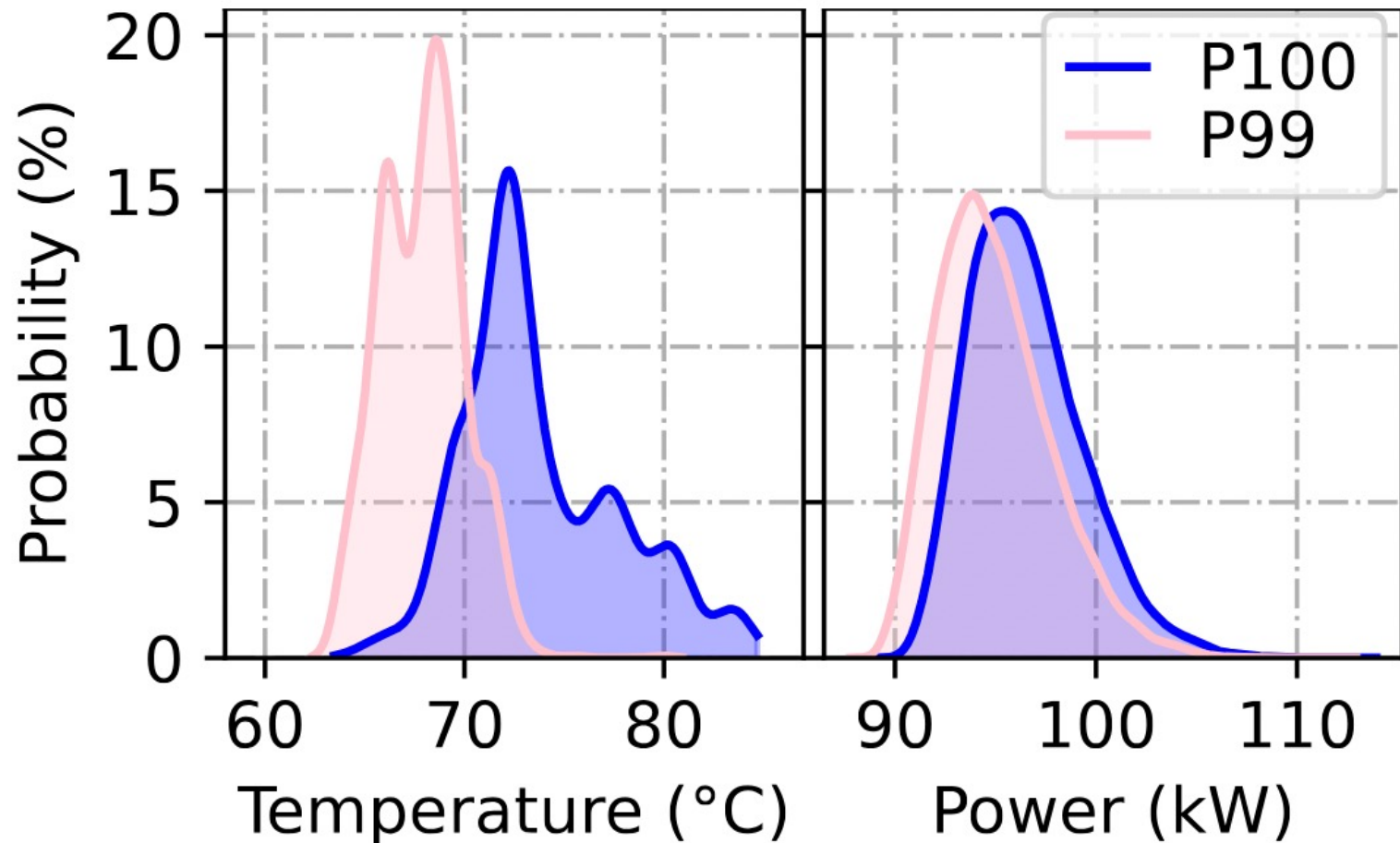


How to Address the Challenges?

- Thermal- and power-aware:
 - Workload placement
 - Instance configuration
 - Request scheduling

Opportunity #1: Smart Workload Placement

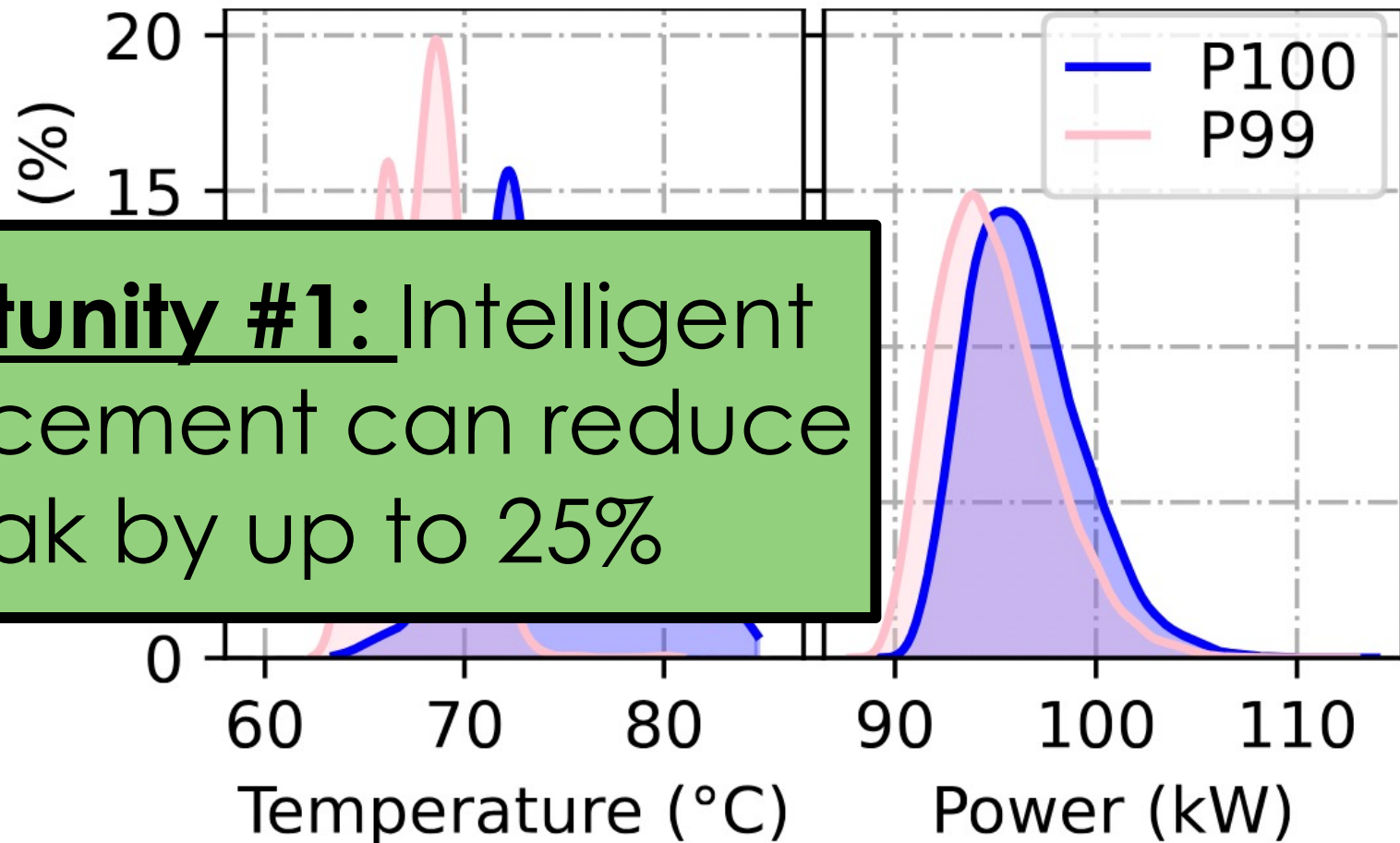
- 100K random VM placements
- Generate diverse thermal and power distributions



Opportunity #1: Smart Workload Placement

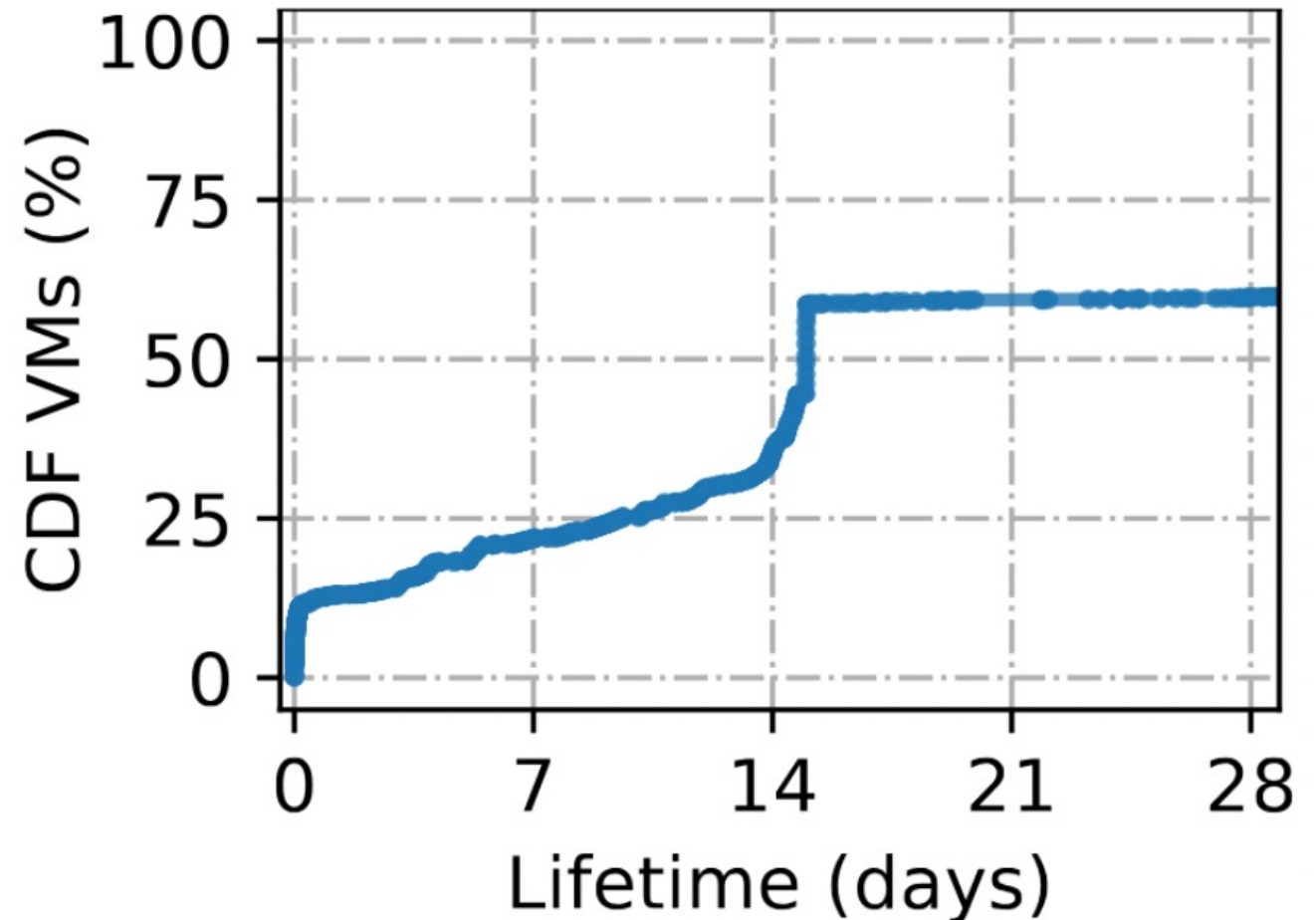
- 100K random VM placements
- Generate thermal and power distributions

Opportunity #1: Intelligent VM placement can reduce peak by up to 25%



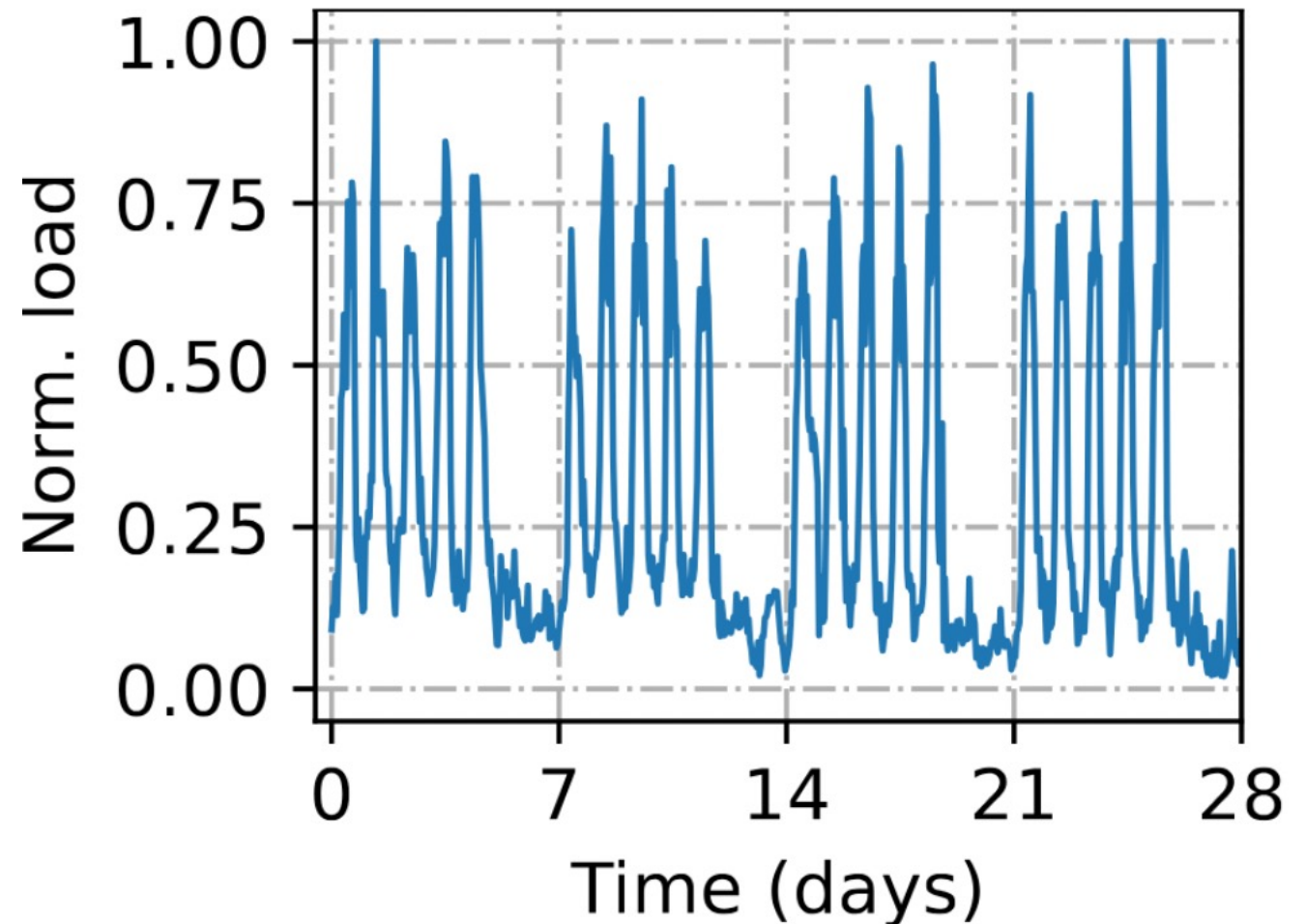
Opportunity #1: Smart Workload Placement

- LLM inference VMs are long lived



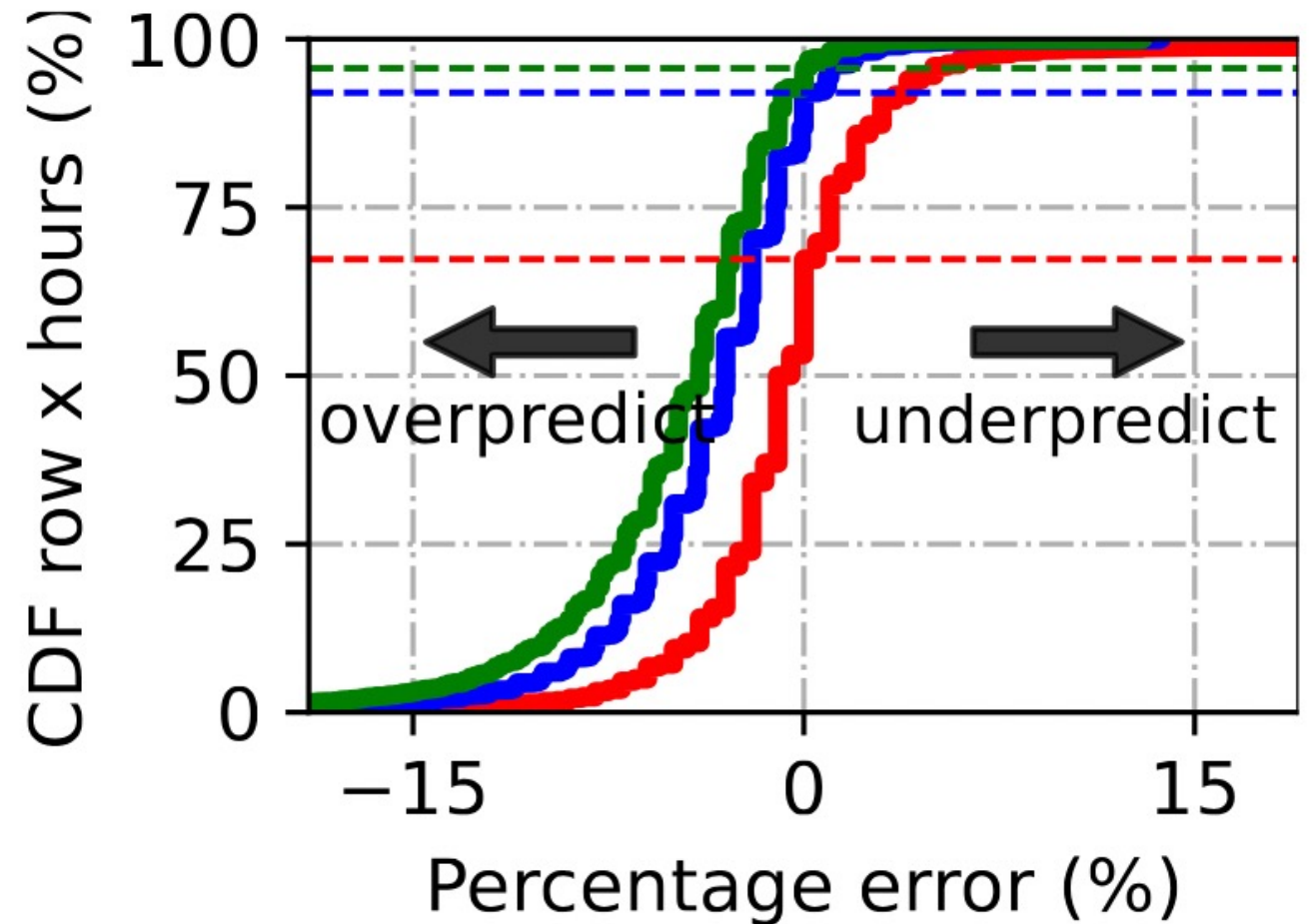
Opportunity #1: Smart Workload Placement

- LLM inference loads are periodic

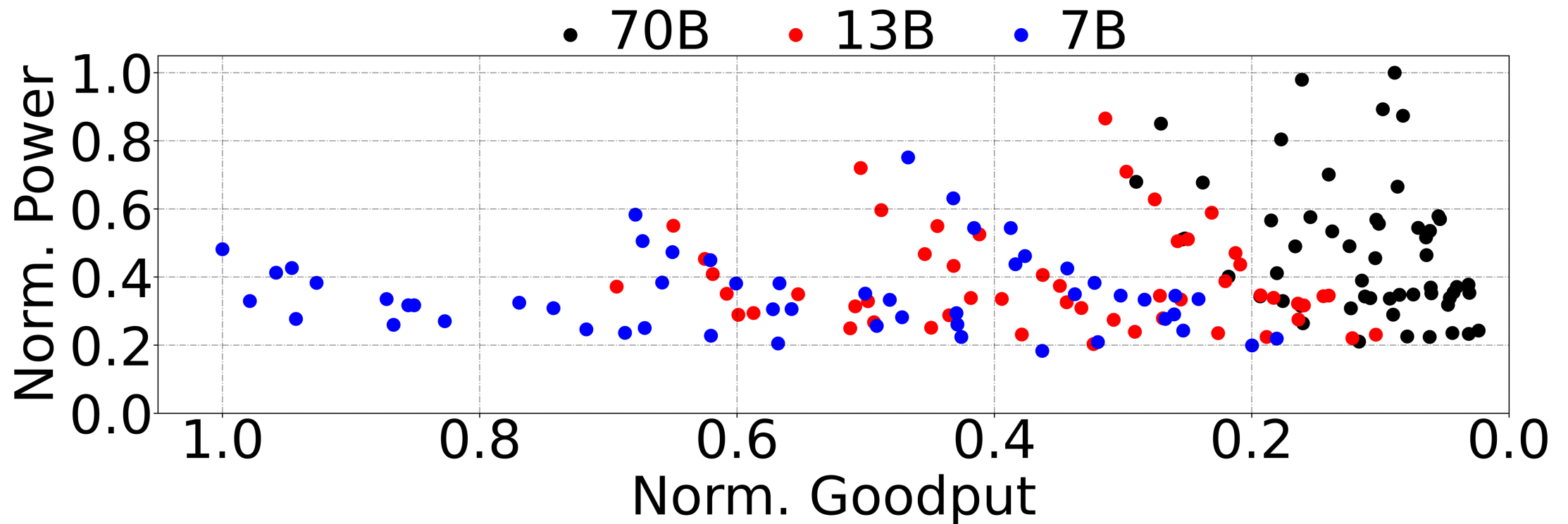


Opportunity #1: Smart Workload Placement

- Long-lived instances with periodic loads
→ high predictability
- Accuracy over 90%

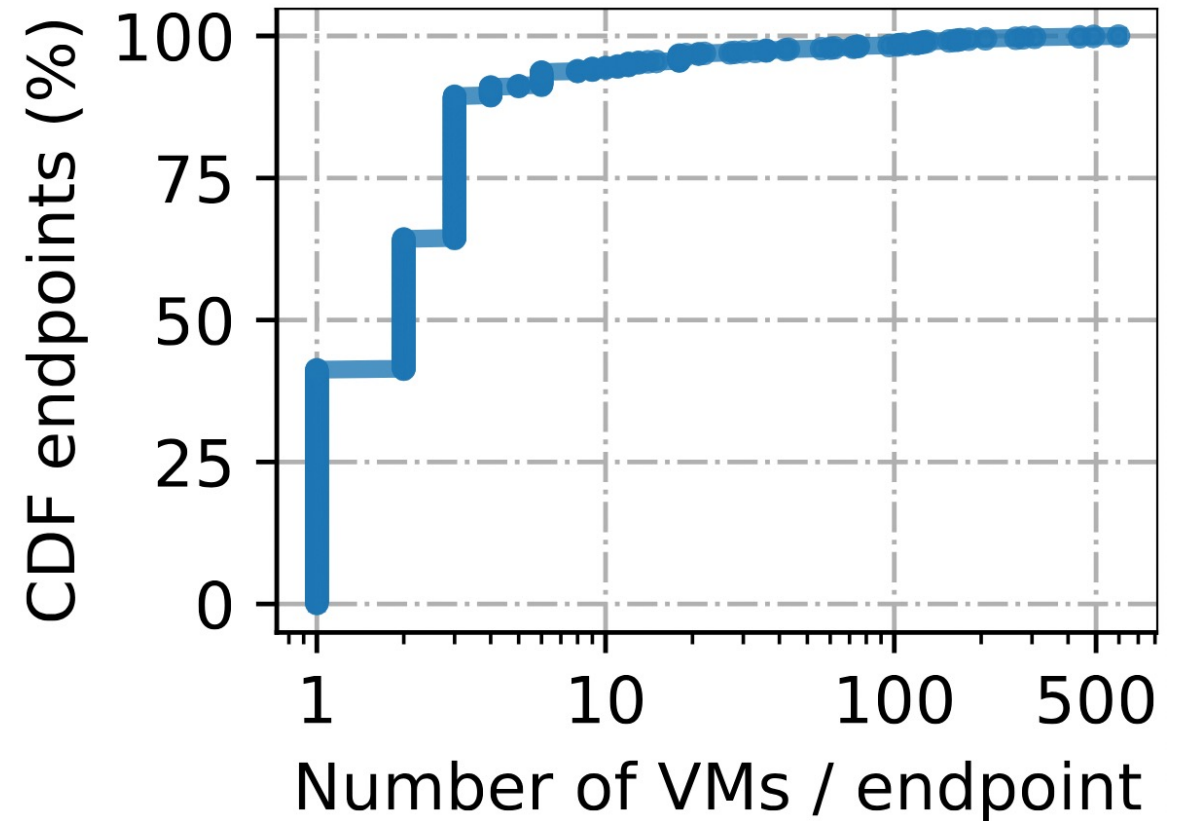


Opportunity #2: Tune Flexible SaaS LLM VMs



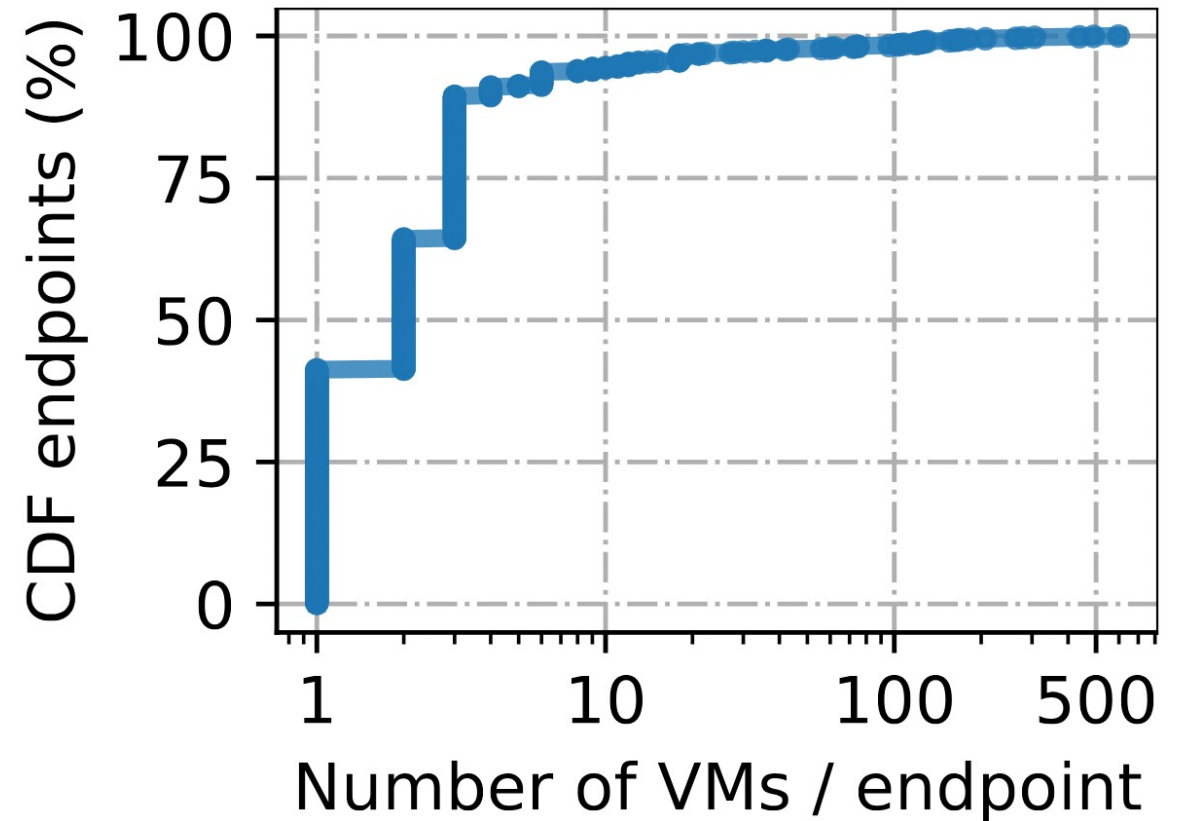
Opportunity #3: Smart Request Routing

- LLM inference service spans many VMs



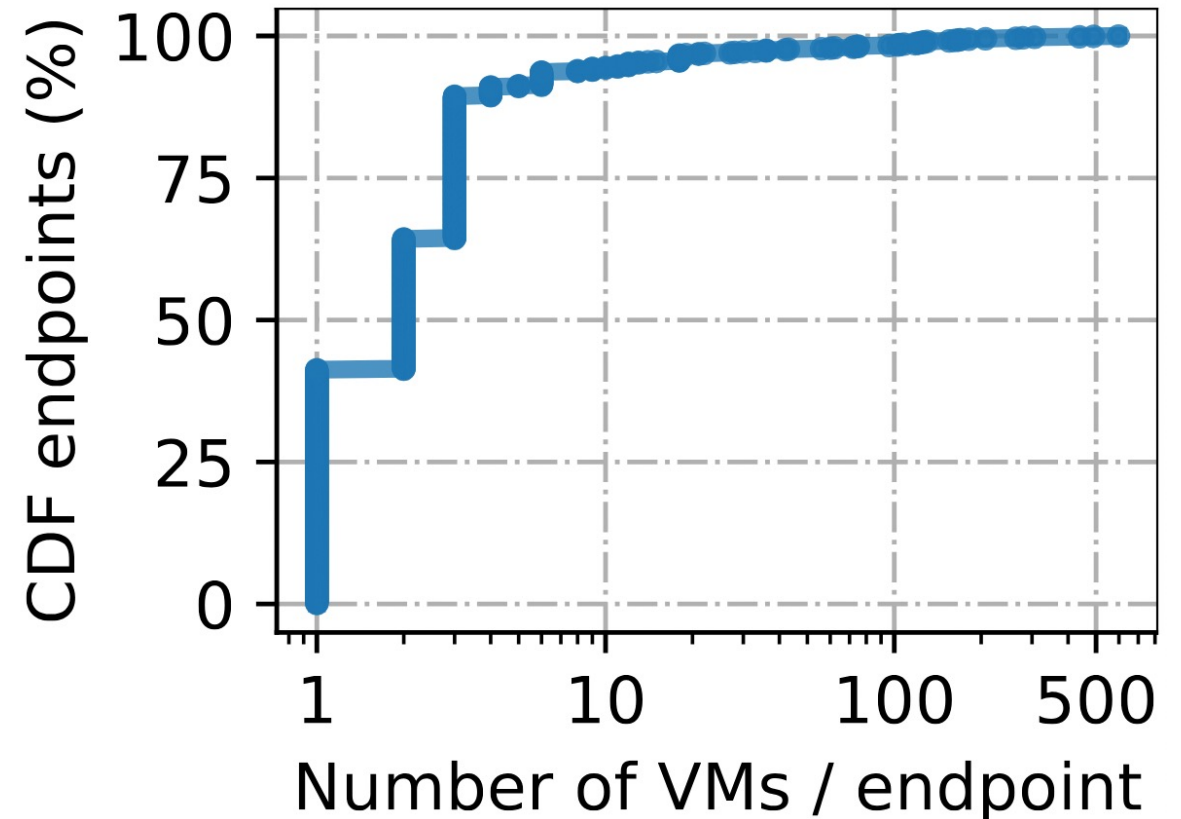
Opportunity #3: Smart Request Routing

- LLM inference service spans many VMs
- VMs of a given service can be placed in different rows



Opportunity #3: Smart Request Routing

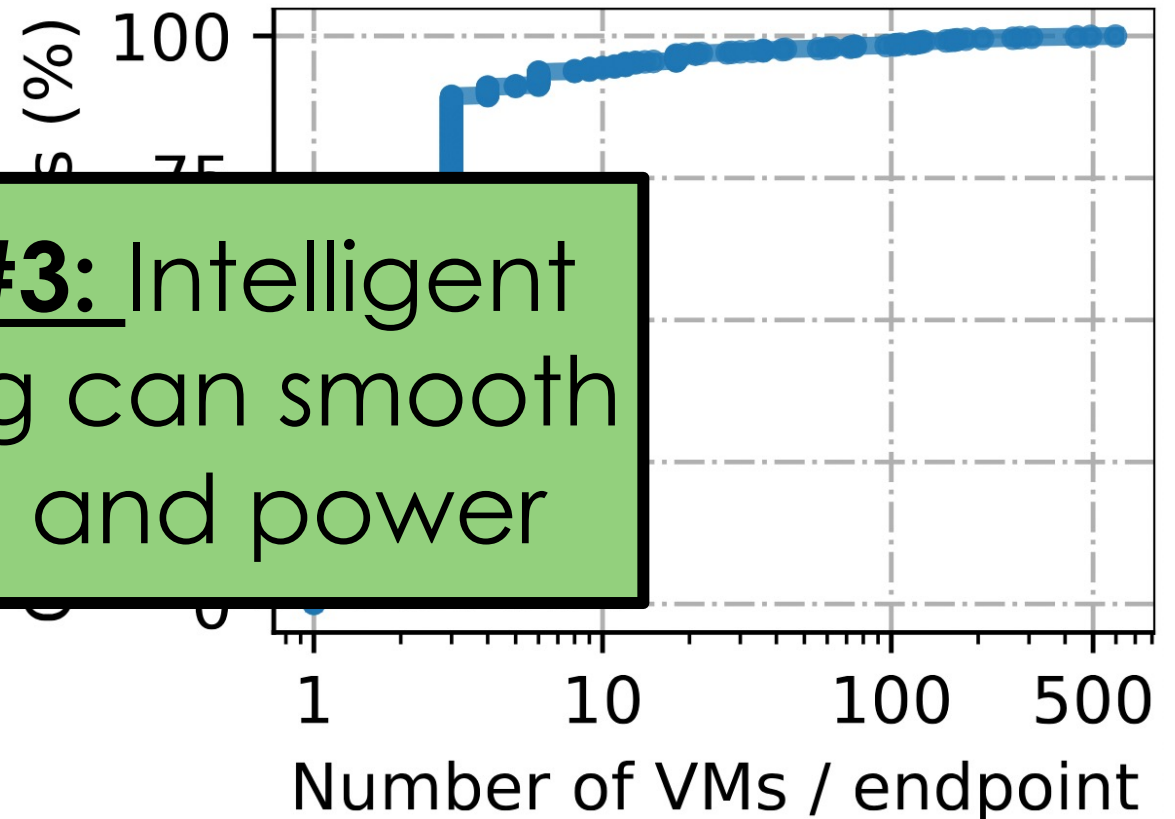
- LLM inference service spans many VMs
- VMs of a given service can be placed in different rows
- Load balance the requests in a thermal- and power-aware manner

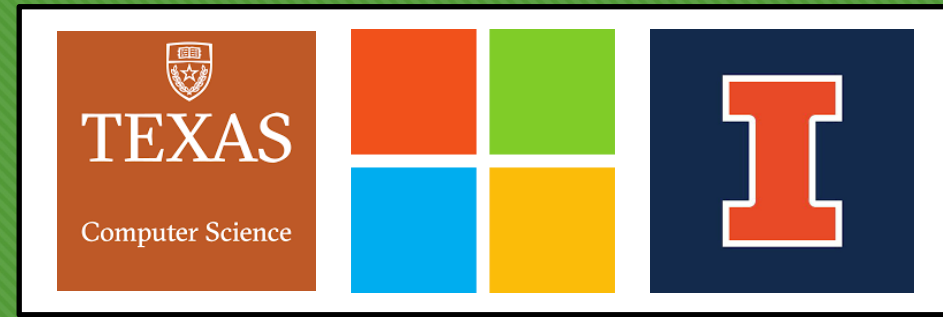


Opportunity #3: Smart Request Routing

- LLM inference service spans many VMs
- VMs of a given service can be placed in different locations
- Load balancers can route requests in a thermal- and power-aware manner

Opportunity #3: Intelligent request routing can smooth temperature and power





TAPAS: Thermal- and Power-Aware Scheduling for LLM Inference in Cloud Platforms

ASPLOS 2025

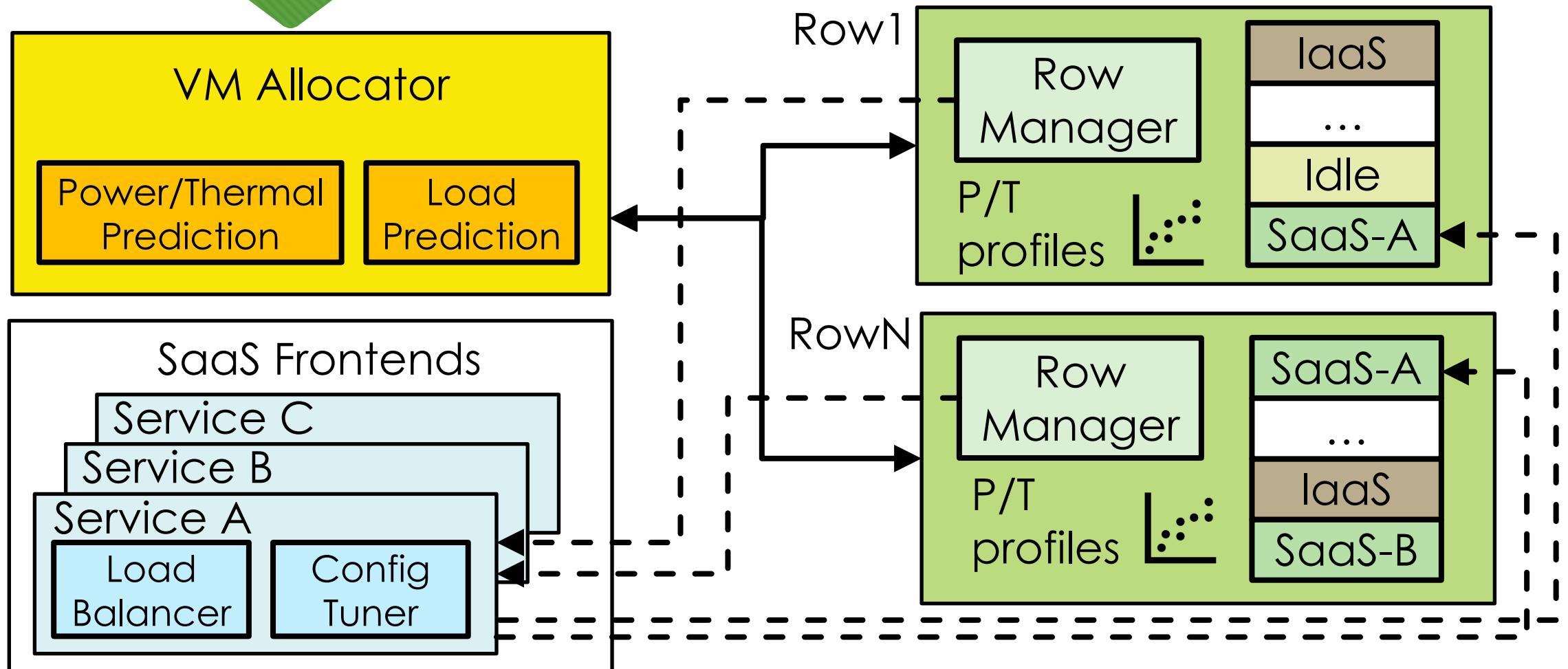
Jovan Stojkovic, Chaojie Zhang*, Íñigo Goiri*, Esha Choukse*, Haoran Qiu*, Rodrigo Fonseca*, Josep Torrellas, Ricardo Bianchini*

University of Illinois at Urbana-Champaign, *Azure Research – Systems

TAPAS: Thermal- and Power-Aware Scheduling for LLM Inference in the Cloud

- VM Placement
- Instance Configuration
- Request Routing

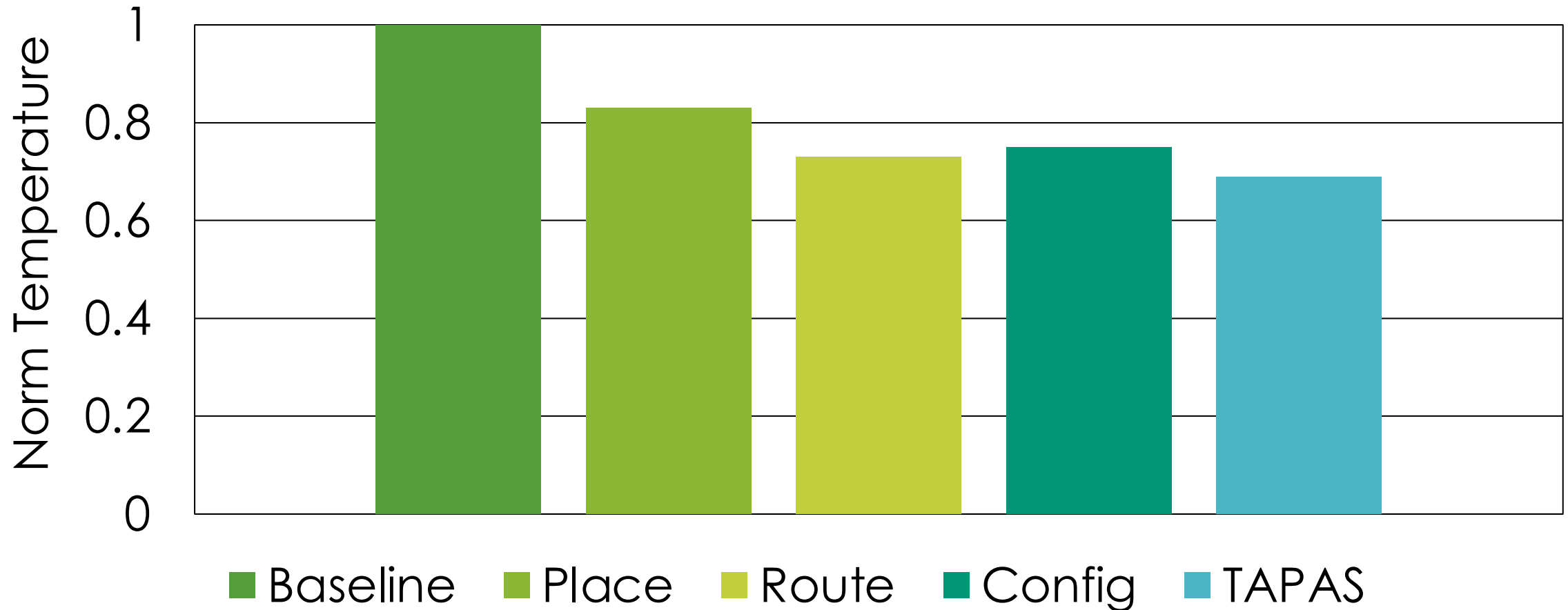
TAPAS: Thermal- and Power-Aware Scheduling for LLM Inference in the Cloud



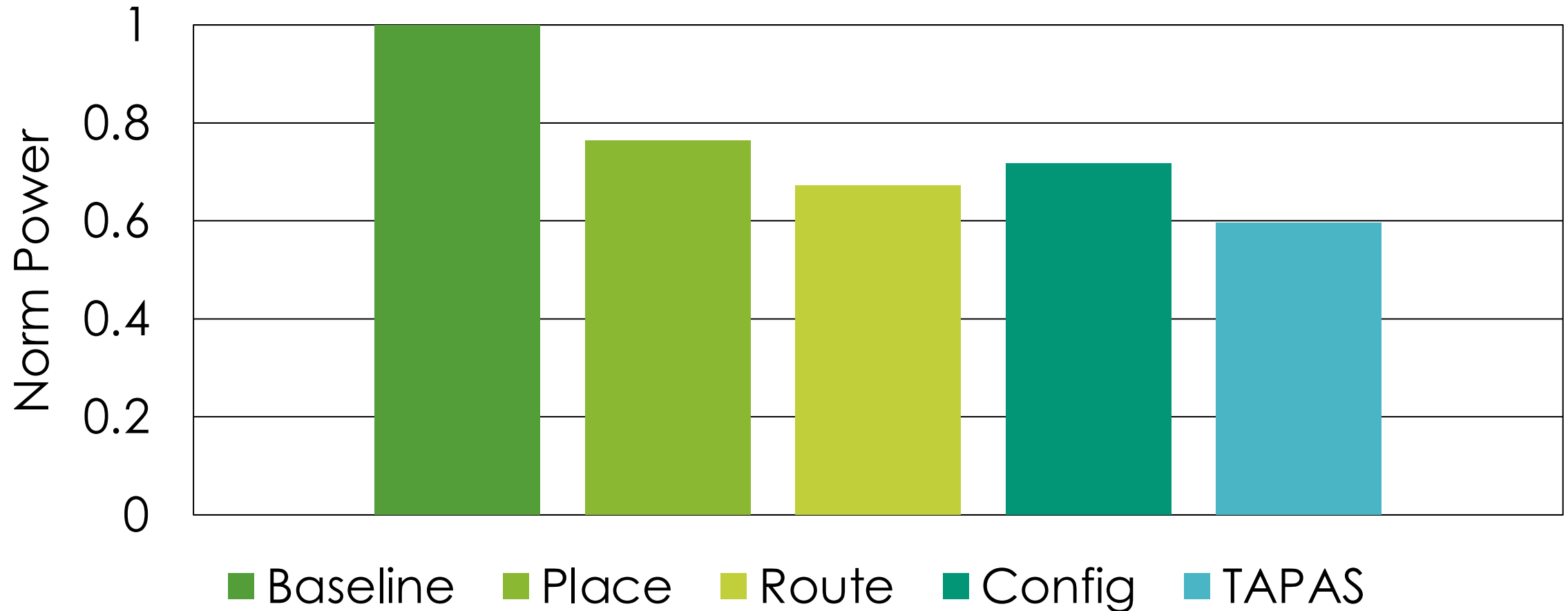
TAPAS Evaluation

- Real cluster experiments
 - Two rows of 80 A100 servers for 1 hour
 - IaaS: direct power and thermal readings
 - SaaS: replay production traces with Llama2 models
- Simulations
 - Week production trace from one of the A100 datacenters
 - Thermal and power modeling using our derived equations

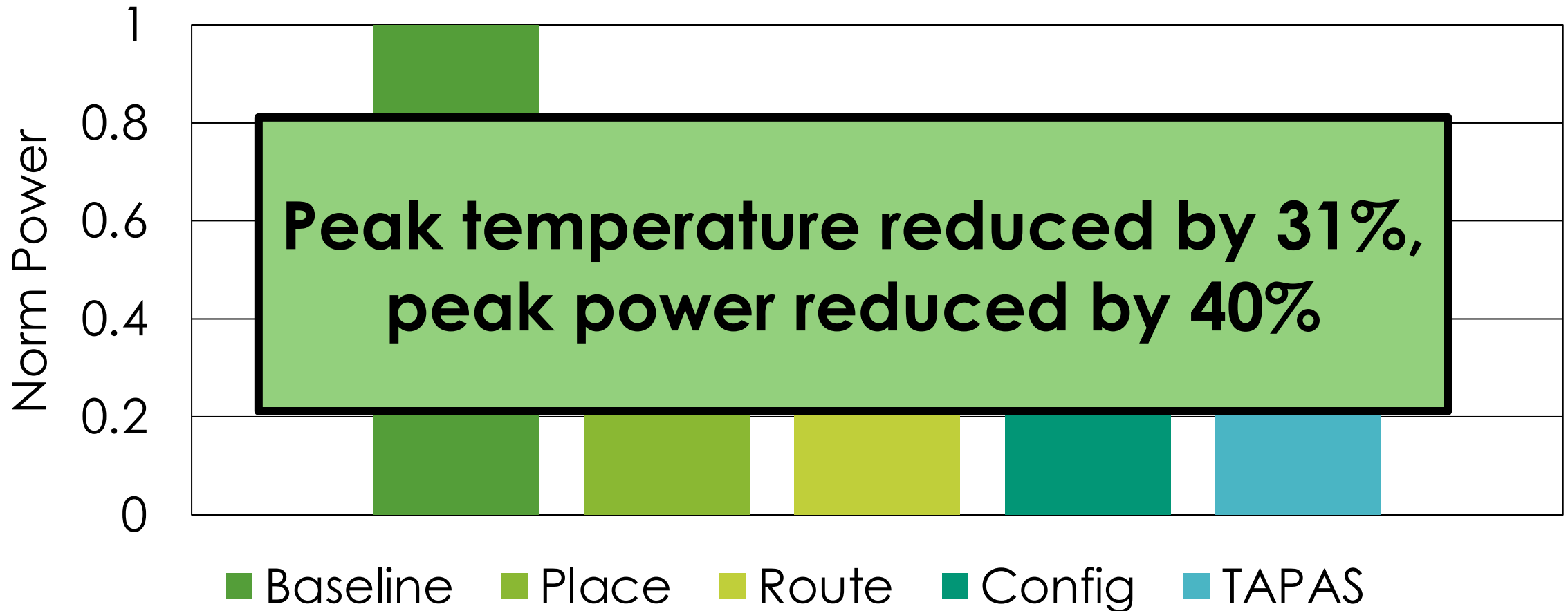
TAPAS Reduces Peak Temperature and Power!



TAPAS Reduces Peak Temperature and Power!



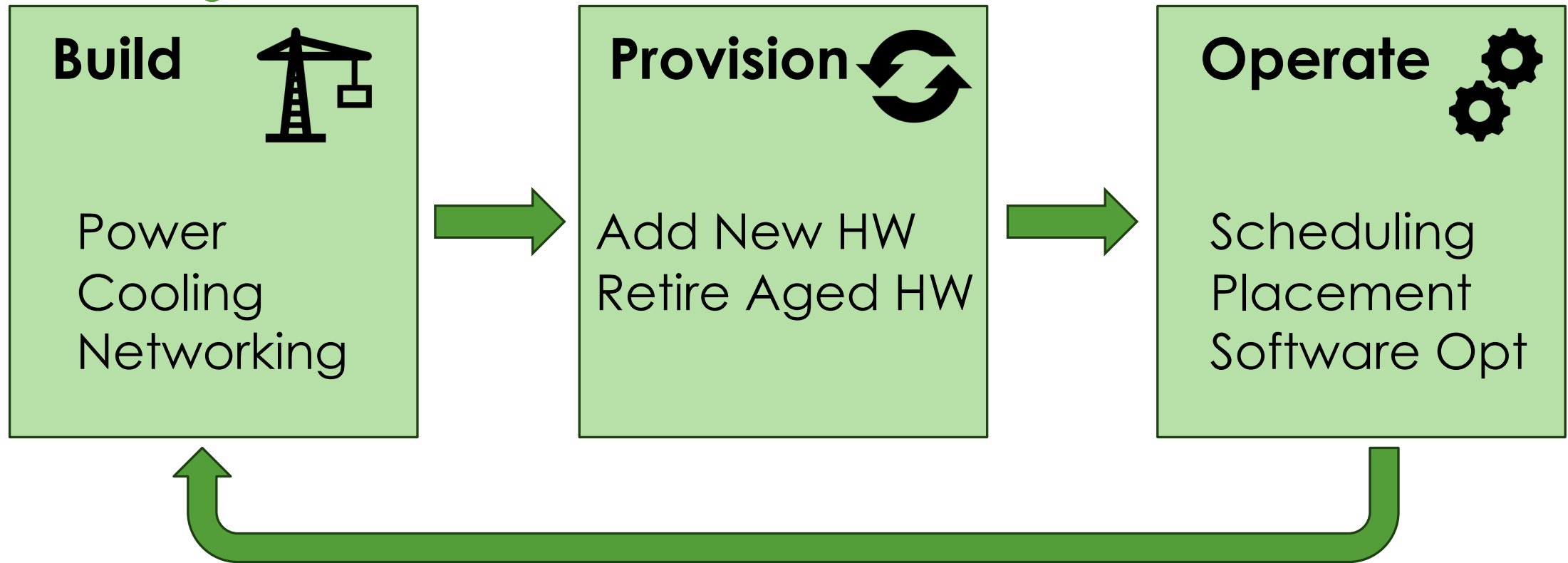
TAPAS Reduces Peak Temperature and Power!

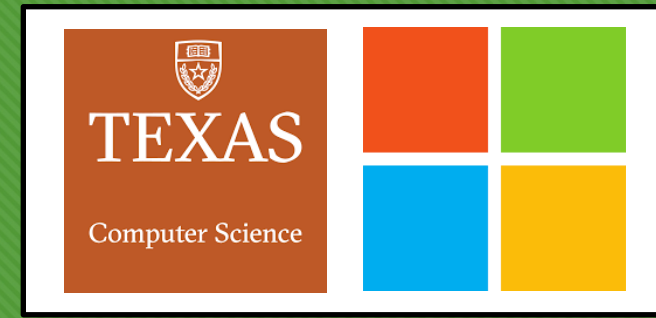


Goal: Make LLMs Cost-Efficient

- Rethink the AI datacenter lifecycle across build, provision, and operation stages through a TCO-driven framework

A Datacenter Lifecycle





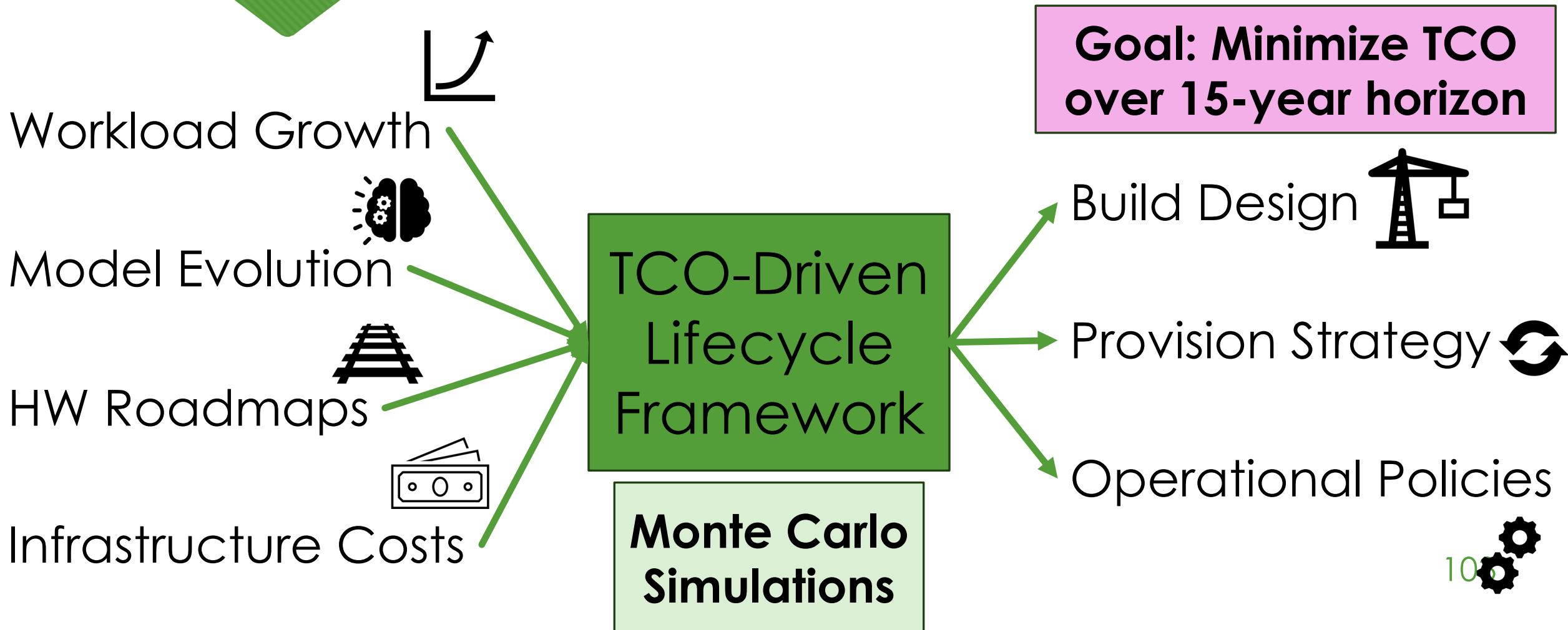
Rearchitecting the Datacenter Lifecycle for AI

ISCA 2026

Jovan Stojkovic, Chaojie Zhang*, Íñigo Goiri*, Ricardo Bianchini*

The University of Texas at Austin, *Azure Research – Systems

A Datacenter Lifecycle

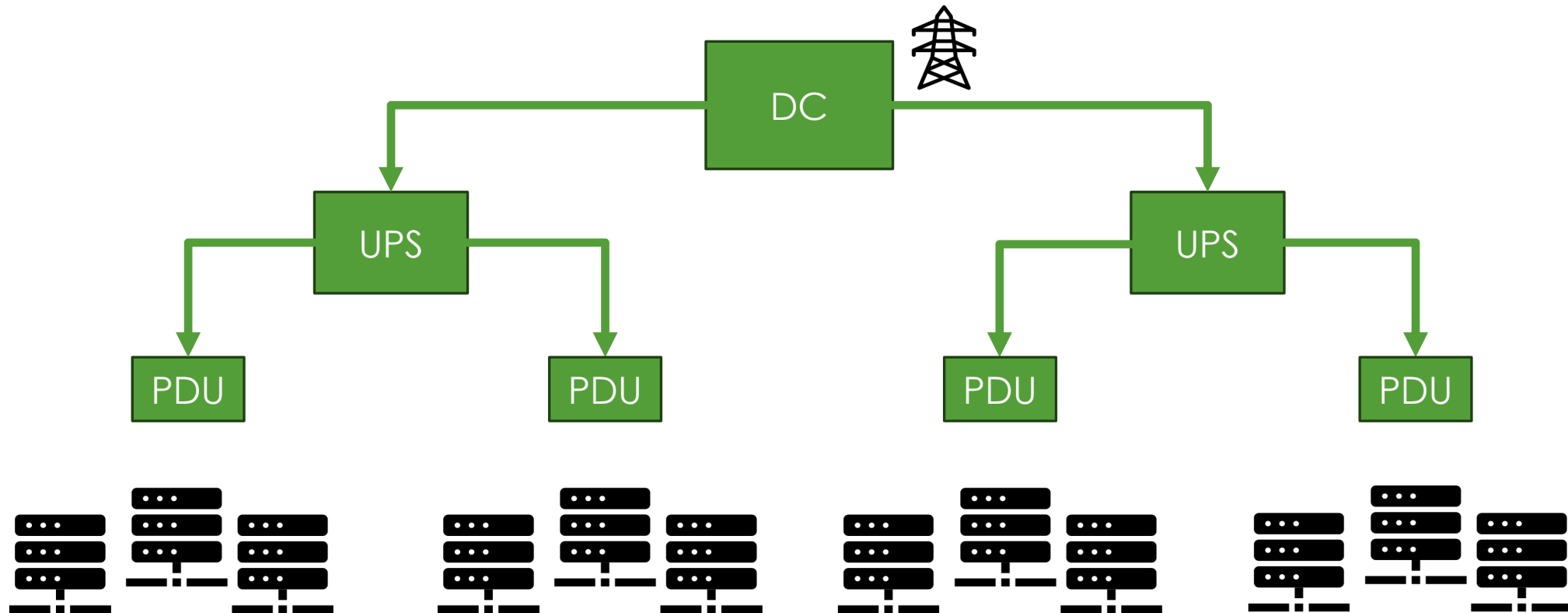


Build-Stage Findings: Hierarchical Power Topology → Fragmentation

- Traditional datacenters: hierarchical power distribution

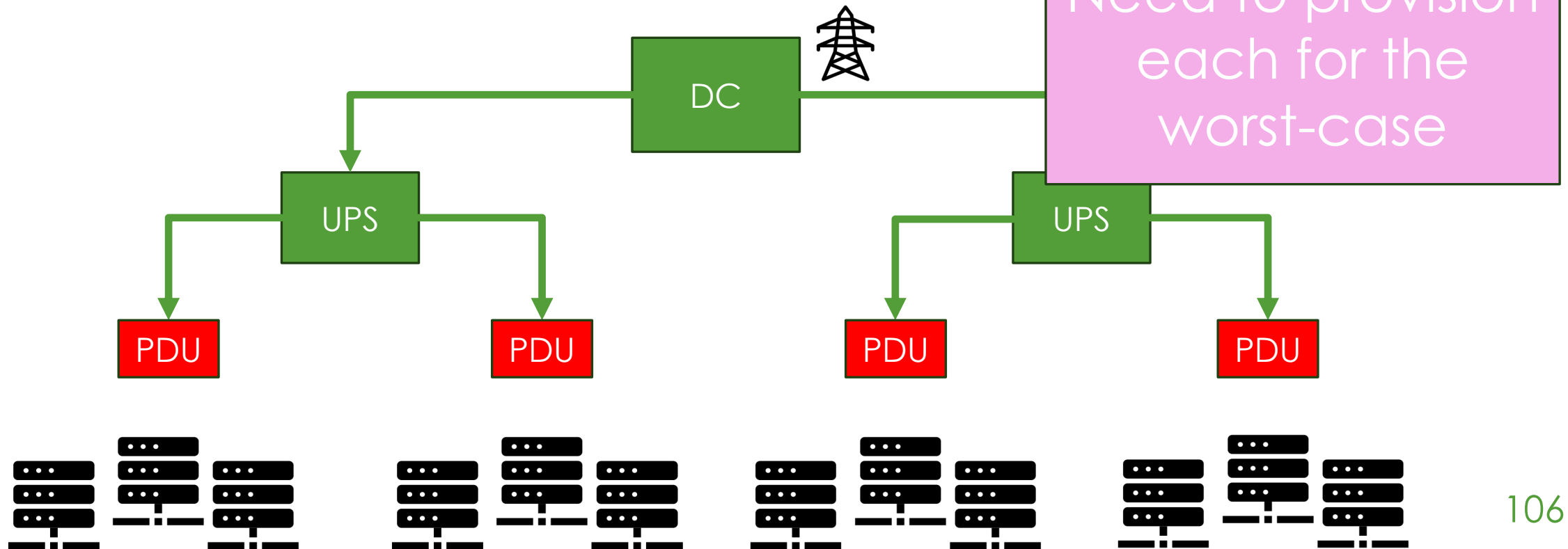
Build-Stage Findings: Hierarchical Power Topology → Fragmentation

- Traditional datacenters: hierarchical power distribution



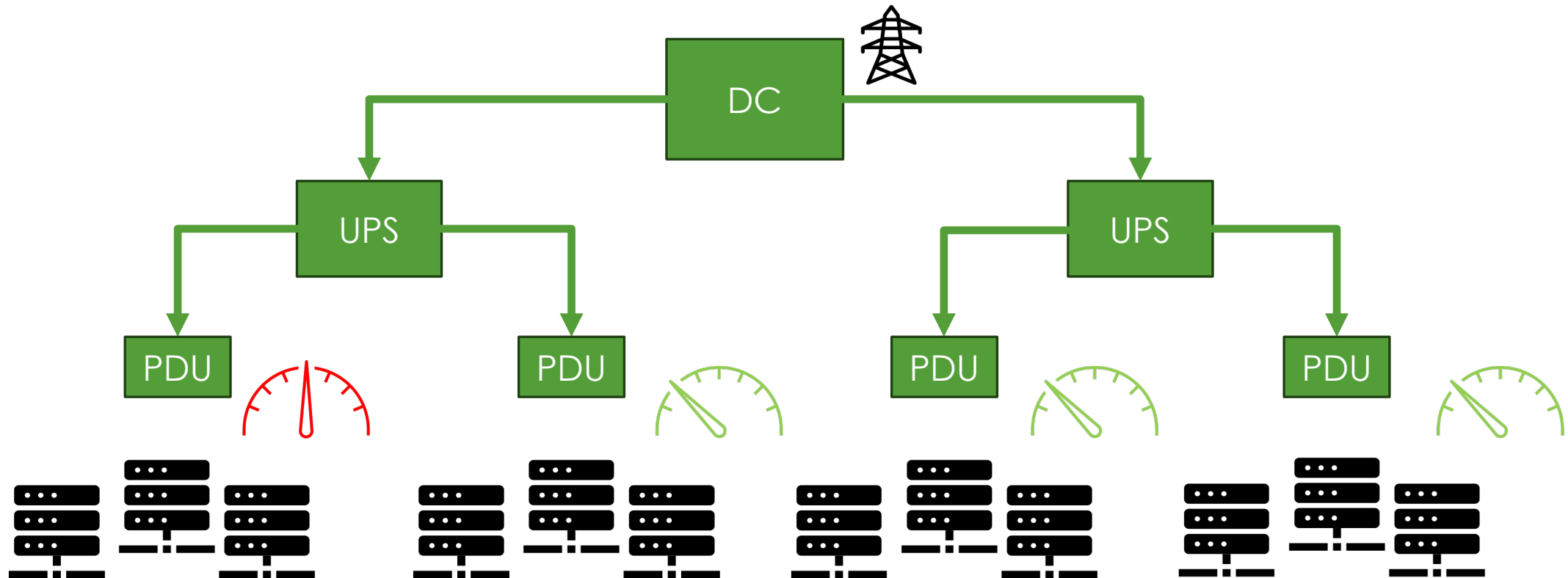
Build-Stage Findings: Hierarchical Power Topology → Fragmentation

- Traditional datacenters: hierarchical power



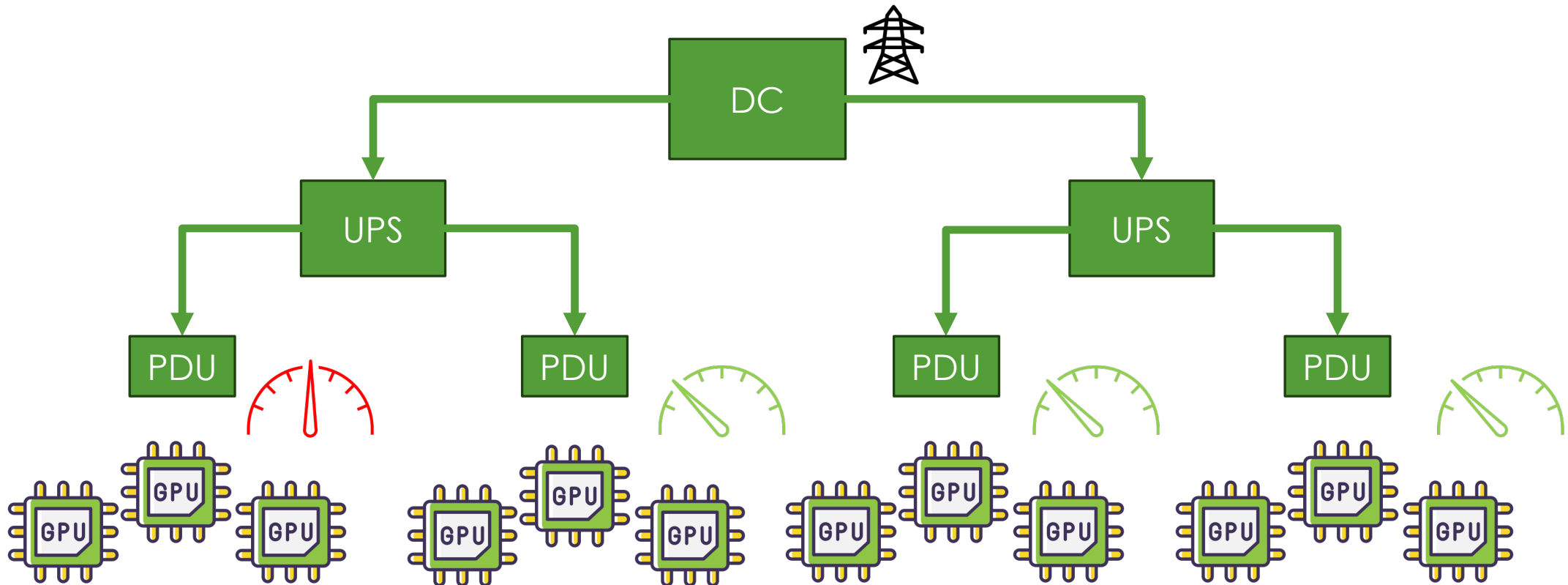
Build-Stage Findings: Hierarchical Power Topology → Fragmentation

○ Traditional datacenters: hierarchical power distribution



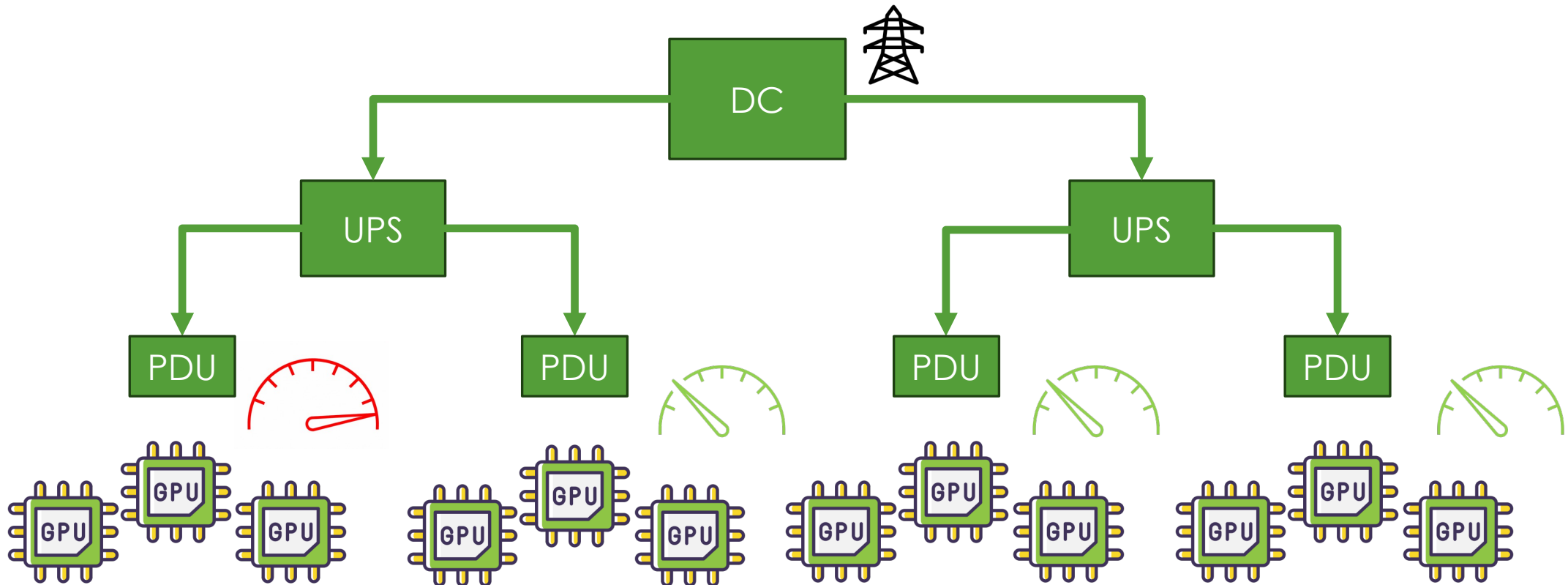
Build-Stage Findings: Hierarchical Power Topology → Fragmentation

○ Traditional datacenters: hierarchical power distribution



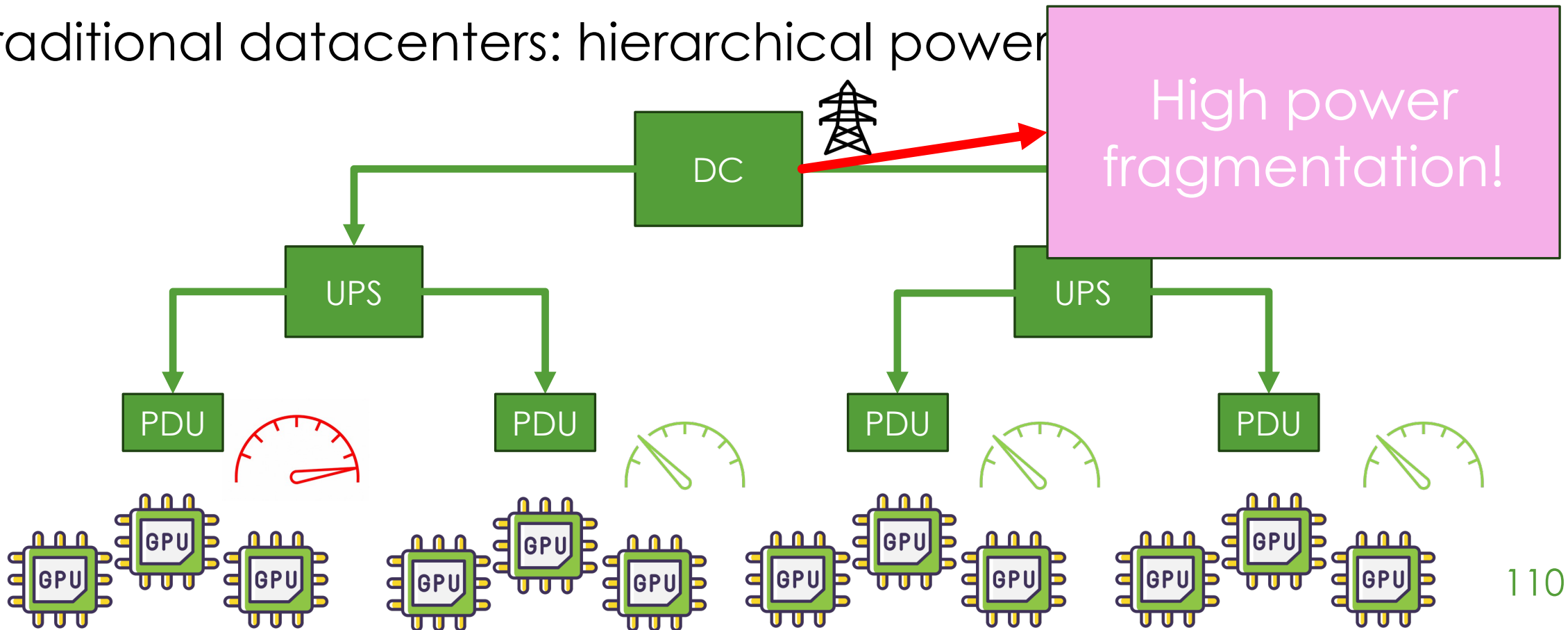
Build-Stage Findings: Hierarchical Power Topology → Fragmentation

○ Traditional datacenters: hierarchical power distribution



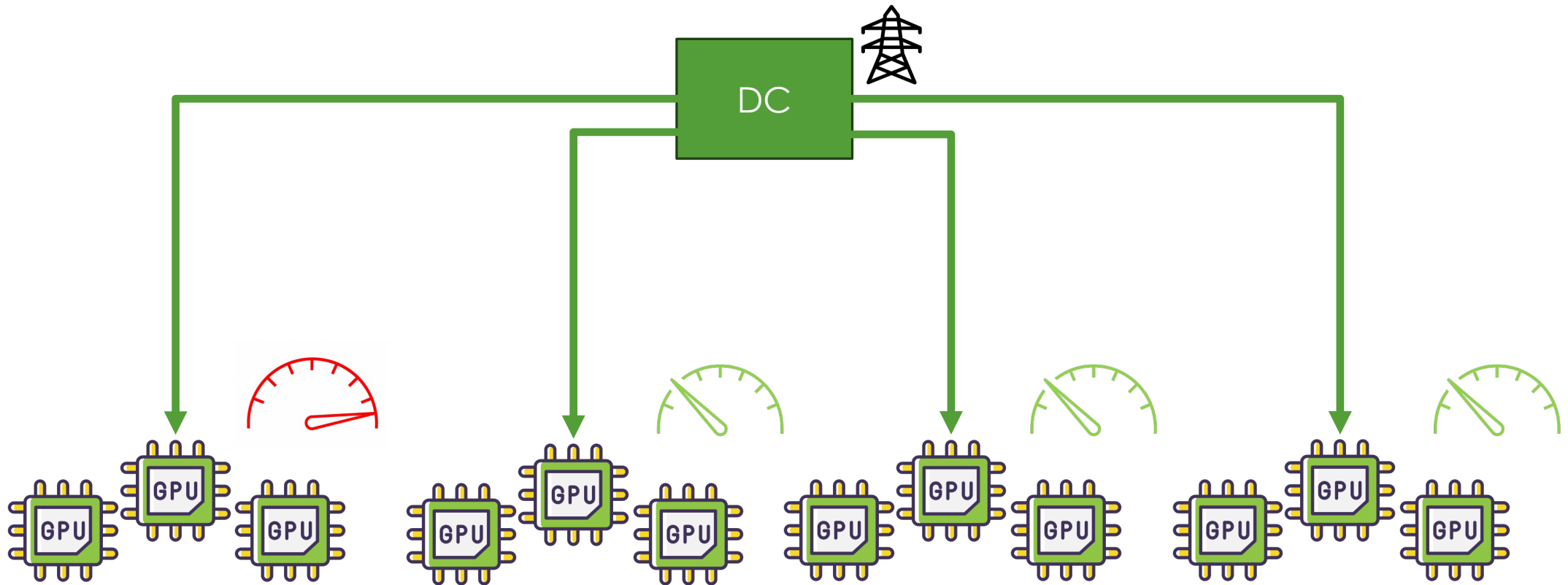
Build-Stage Findings: Hierarchical Power Topology → Fragmentation

○ Traditional datacenters: hierarchical power



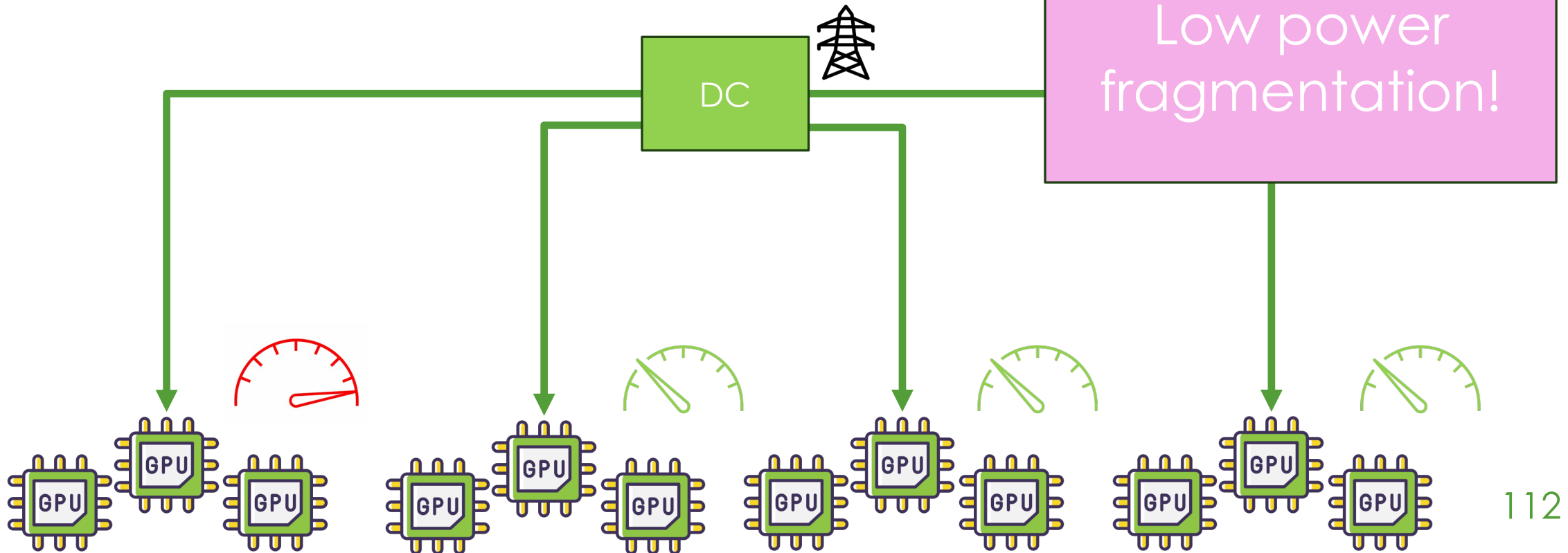
Build-Stage Findings: Need for a Flat Power Topology

- AI datacenters: need flatter power distribution topologies



Build-Stage Findings: Need for a Flat Power Topology

- AI datacenters: need flatter power distribution

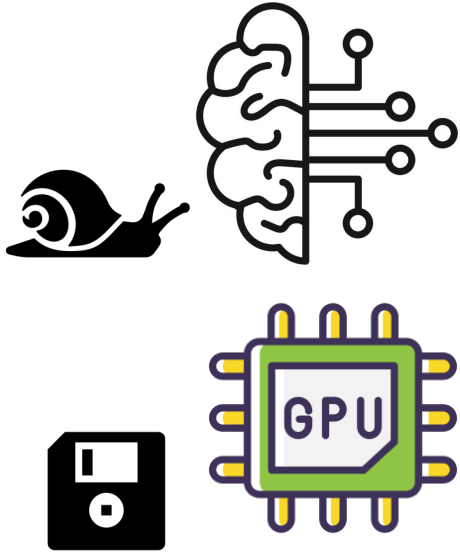


Provision-Stage Findings: AI Workload Affects Provisioning Policies

- Workload-driven refresh signals:

Provision-Stage Findings: AI Workload Affects Provisioning Policies

- Workload-driven refresh signals:
 - Smaller models remain efficient on older GPUs

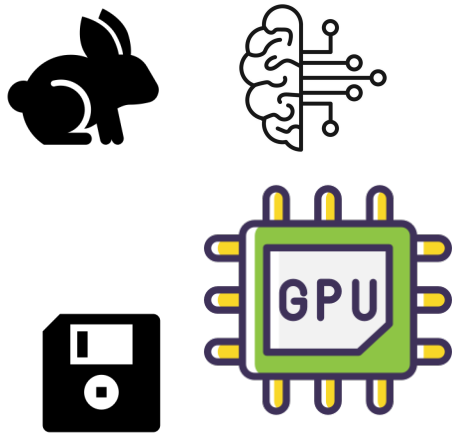


Old GPU

Provision-Stage Findings:

AI Workload Affects Provisioning Policies

- Workload-driven refresh signals:
 - Smaller models remain efficient on older GPUs

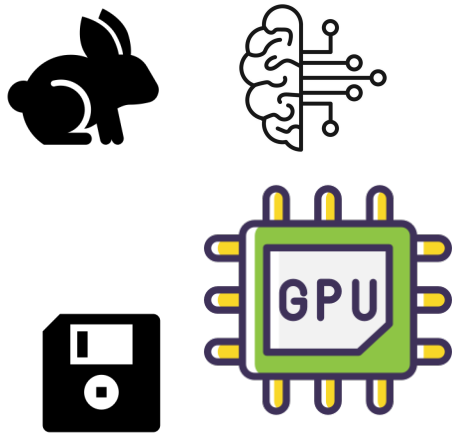


Old GPU

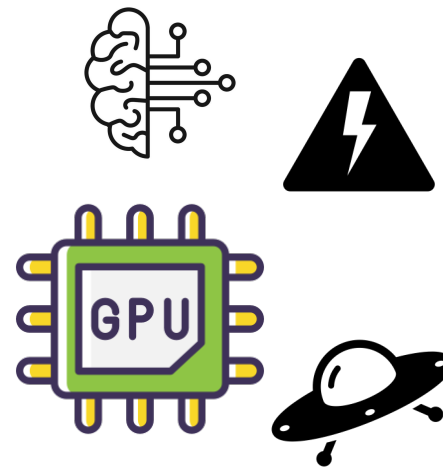
Provision-Stage Findings:

AI Workload Affects Provisioning Policies

- Workload-driven refresh signals:
 - Smaller models remain efficient on older GPUs



Old GPU



New GPU

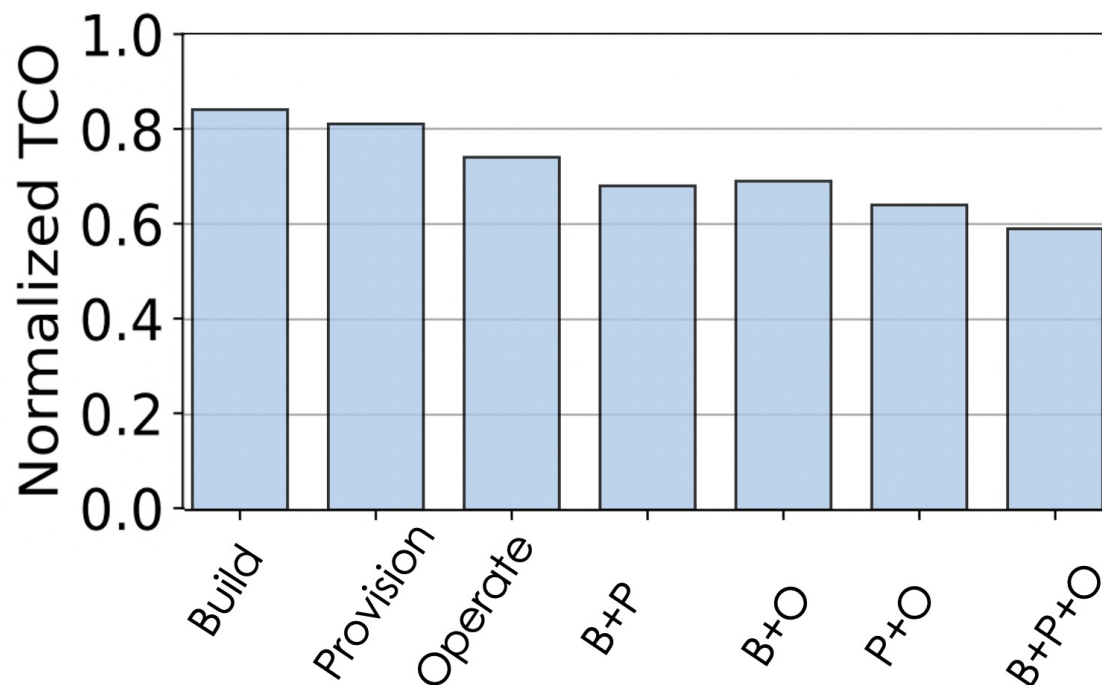
Provision-Stage Findings:

AI Workload Affects Provisioning Policies

- Workload-driven refresh signals:
 - Smaller models remain efficient on older GPUs
 - Sparse (MoE) models scale better on older GPUs
 - State-space models are more hardware-efficient for older GPUs

Cross-Stage Optimizations

- The largest TCO savings come from cross-stage rearchitecting the end-to-end AI datacenter lifecycle



Holistic approach
reduces AI datacenter
TCO by 40%!

From Monolithic LLM to Agentic AI

01 Agentic AI is becoming an important datacenter workload

02 Existing AI servers optimize monolithic LLM inference

03 Agentic workflows add orchestration, tools, and cross-agent dependencies

What does agentic AI demand from future servers?

From Monolithic LLM to Agentic AI

Example: One request repeatedly crosses the CPU-GPU boundary

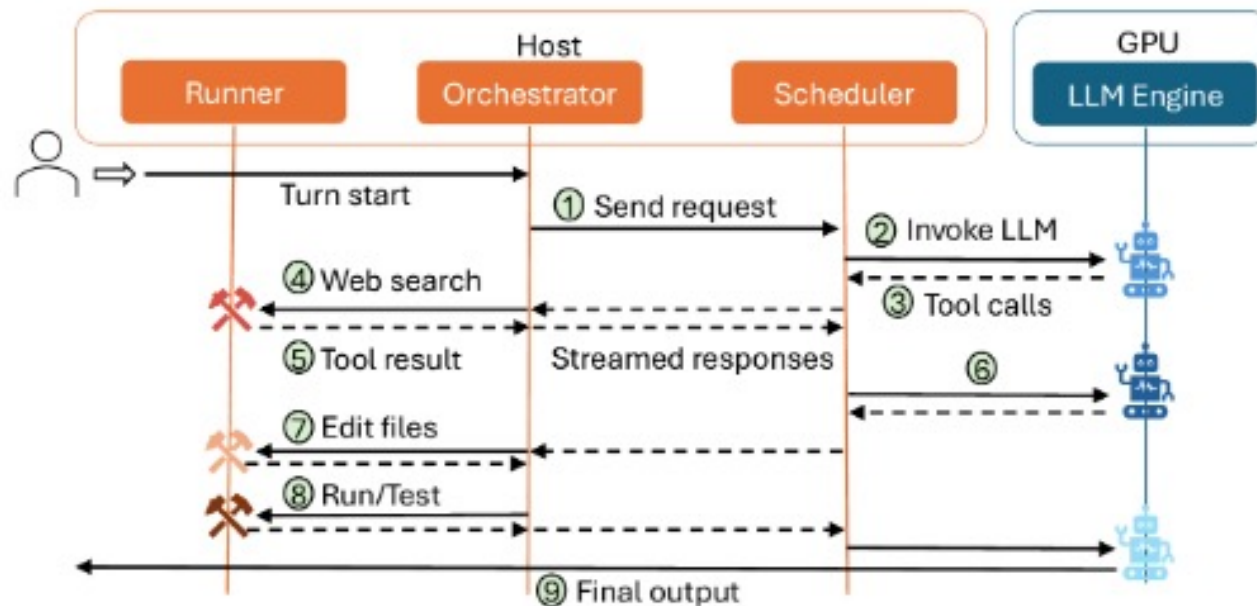


Fig. 1. An example of agentic AI workflow execution.

Model calls separated by host-side routing, tool execution, state updates

Architectural Implications?

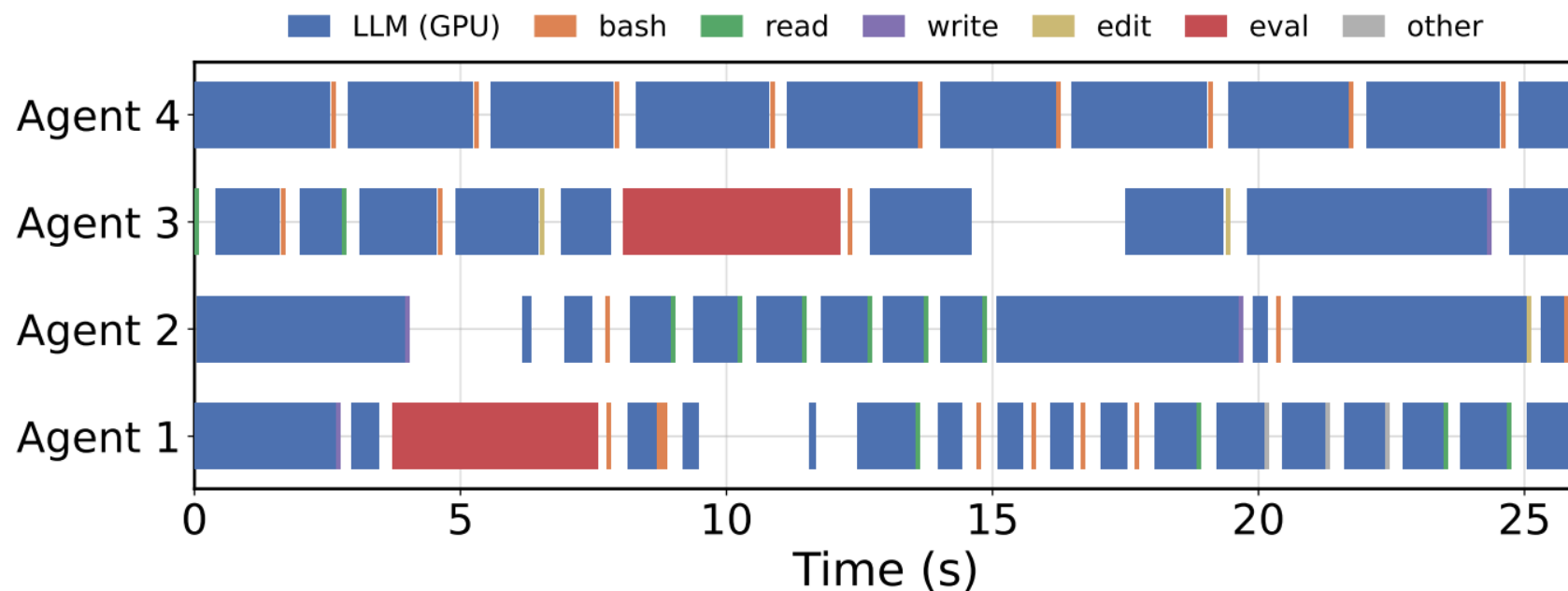
Architectural Implications of Agentic AI Workflows

Jirong Yang[†], Peizhe Liu[†], Chaojie Zhang^{*}, Jovan Stojkovic[†]

[†]The University of Texas at Austin

^{*}Microsoft Azure

01 Highly-fragmented execution behavior (CPU-GPU ping-pongs)



Architectural Implications?

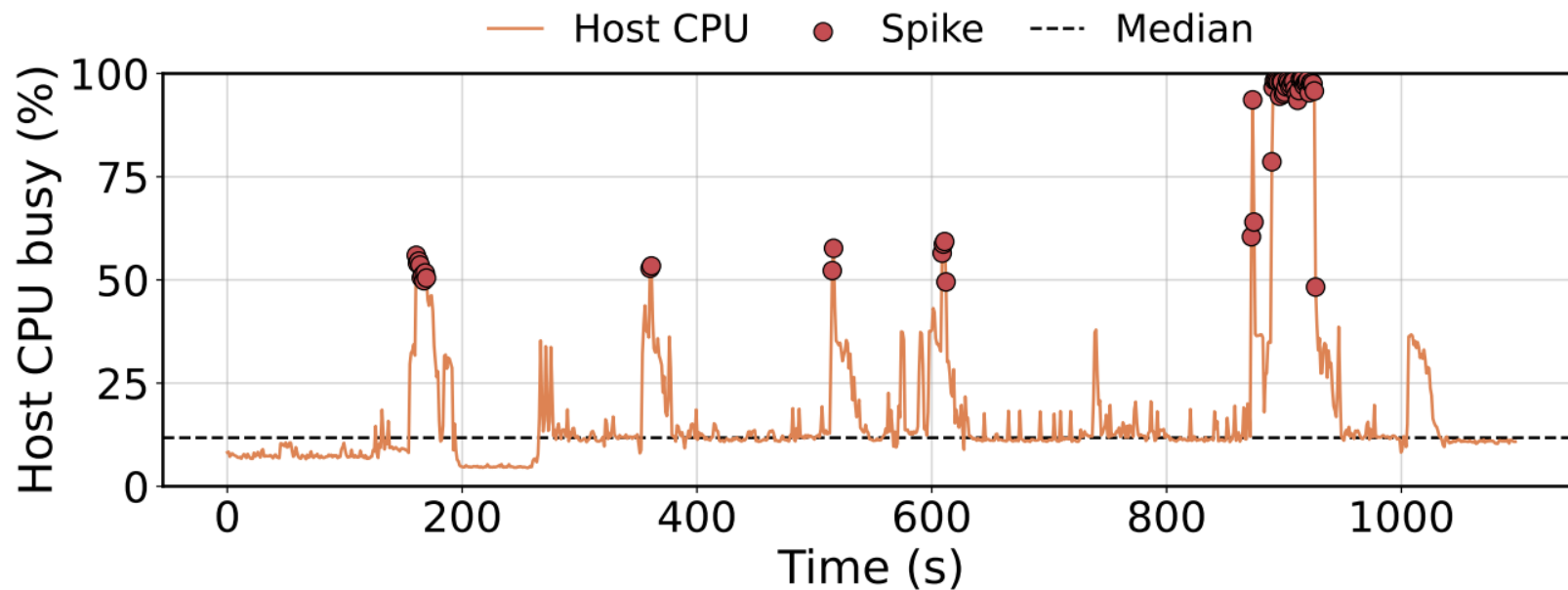
Architectural Implications of Agentic AI Workflows

Jirong Yang[†], Peizhe Liu[†], Chaojie Zhang^{*}, Jovan Stojkovic[†]

[†]The University of Texas at Austin

^{*}Microsoft Azure

02 Bursty CPU execution on the critical path



Architectural Implications?

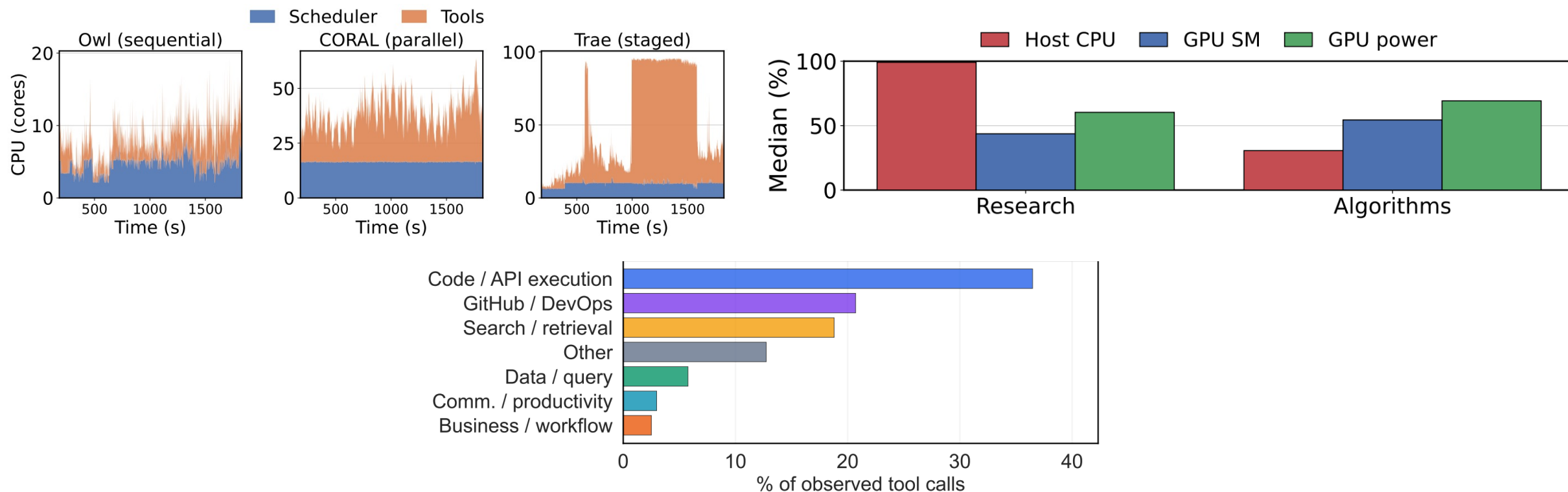
Architectural Implications of Agentic AI Workflows

Jirong Yang[†], Peizhe Liu[†], Chaojie Zhang^{*}, Jovan Stojkovic[†]

[†]The University of Texas at Austin

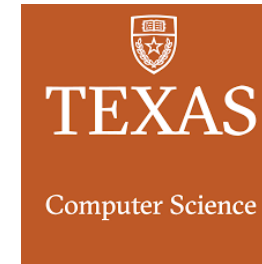
^{*}Microsoft Azure

03 Highly heterogeneous resource needs across tasks



Conclusions

- 01** AI inference an ever-growing datacenter workload
- 02** Rapid evolution of AI models, workloads, and hardware
- 03** Rethinking the entire datacenter stack for AI
- 04** Largest efficiency gains from co-design across system layers



TEXAS
The University of Texas at Austin

Hardware-Software Co-design for AI Workloads in Datacenters

Guest Lecture, Columbia University
September 2026

Jovan Stojkovic

University of Texas at Austin, Computer Science Department