

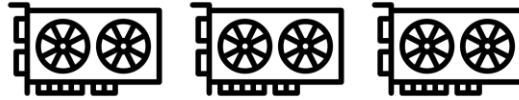
Understanding Performance Implications of LLM Inference on CPUs

[Seonjin Na](#), Geonhwa Jeong, Byung Hoon Ahn, Jeffery Young,
Tushar Krishna, Hyesoon Kim

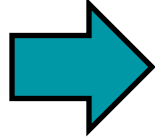
IISWC 2024

Large Language Models (LLM) are widely adopted

Data centers equip with
GPUs, NPUs to accelerate LLM inference

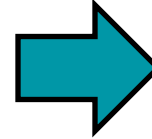


Input Prompt



Meta

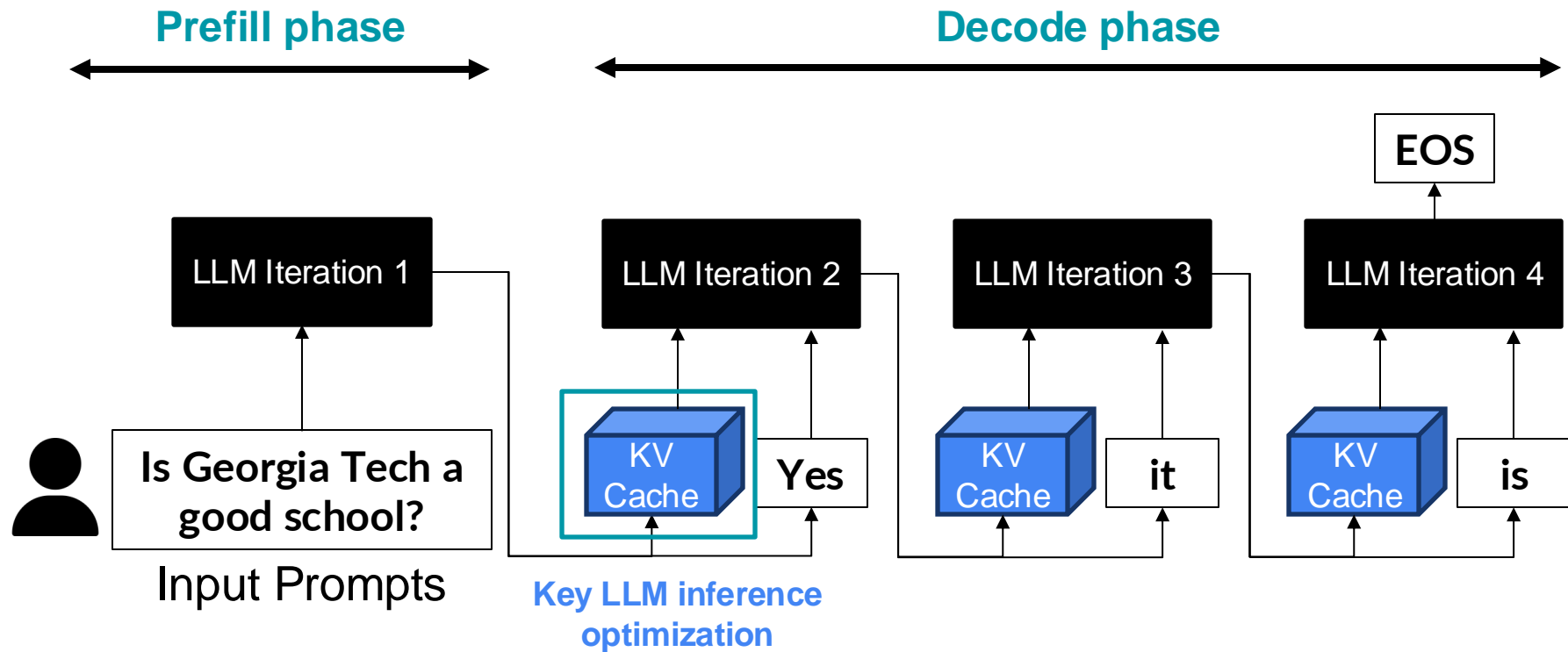
LLM Services



Output



Overview of LLM Inference Procedure



Prefill Phase vs Decode Phase

Prefill phase



Input Prompts

Process **all input prompts**
in parallel

Compute bound

Decode phase

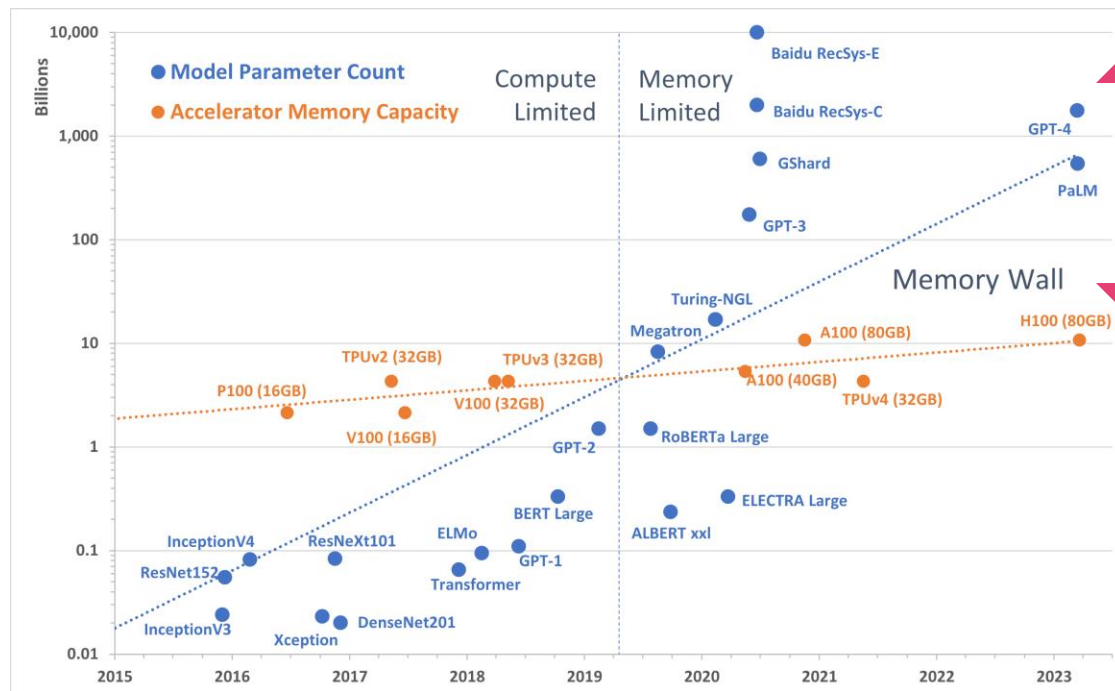


Output tokens

Process **one token at a time**

Memory bound

Challenges in LLM Inference: Huge Model Size



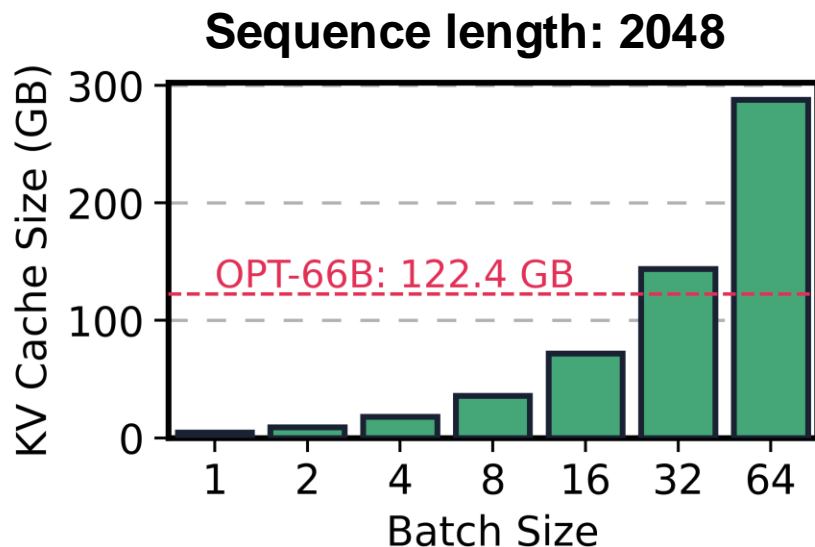
Gap between
GPU memory vs model size

LLMs continue to
grow increasingly larger,
driven by scaling law [2]

The **memory wall** of LLMs [1]

Challenges in LLM Inference: KV Cache Size

- KV Cache size **linearly** scales with **the sequence length and batch size**
 - The size of KV Cache = 2 (Key/ Value) * 2 (BF16) * d_layer * d_model * seq_len * batch_size

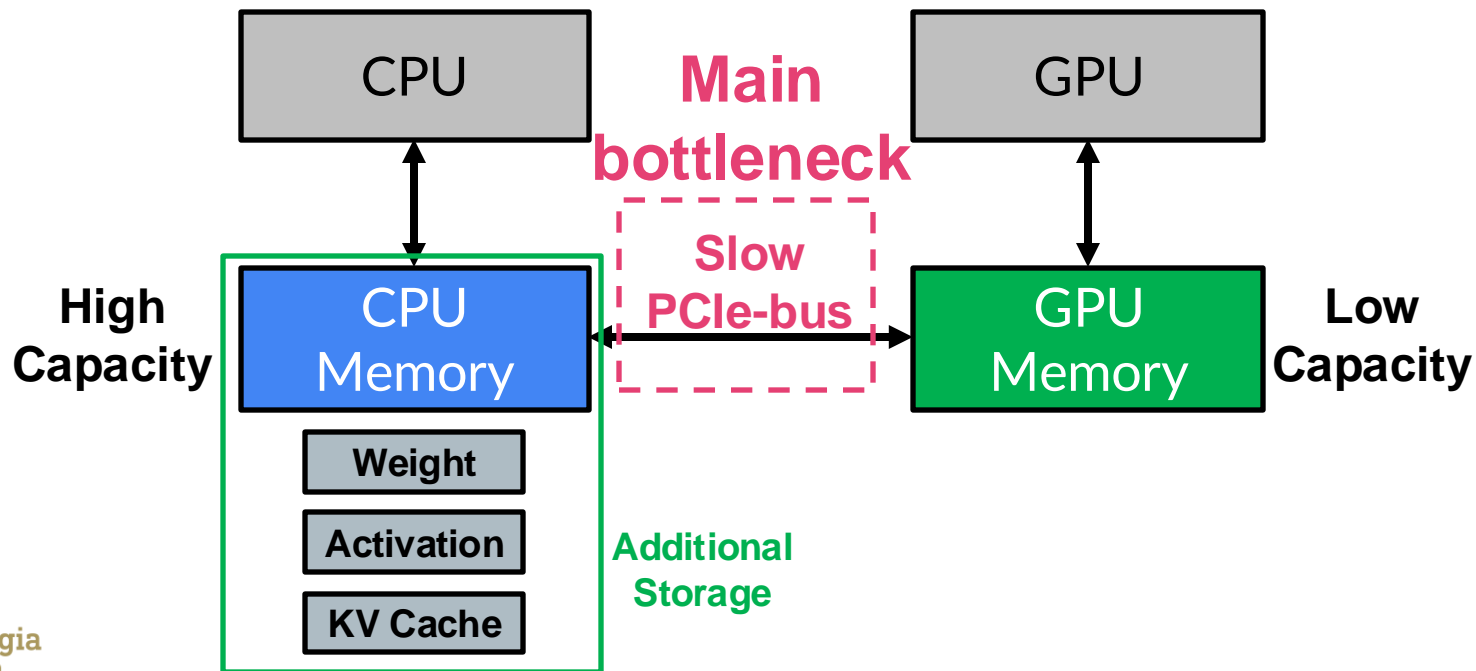


KV cache size is **288GB (FP16)**
with 2048 sequence length,
64 batch size for OPT-66B

**Requires at least
4 H100-80GB GPU**

Offloading-based LLM Inference on GPUs

- LLM weights, activation, KV cache are offloaded to CPU memory



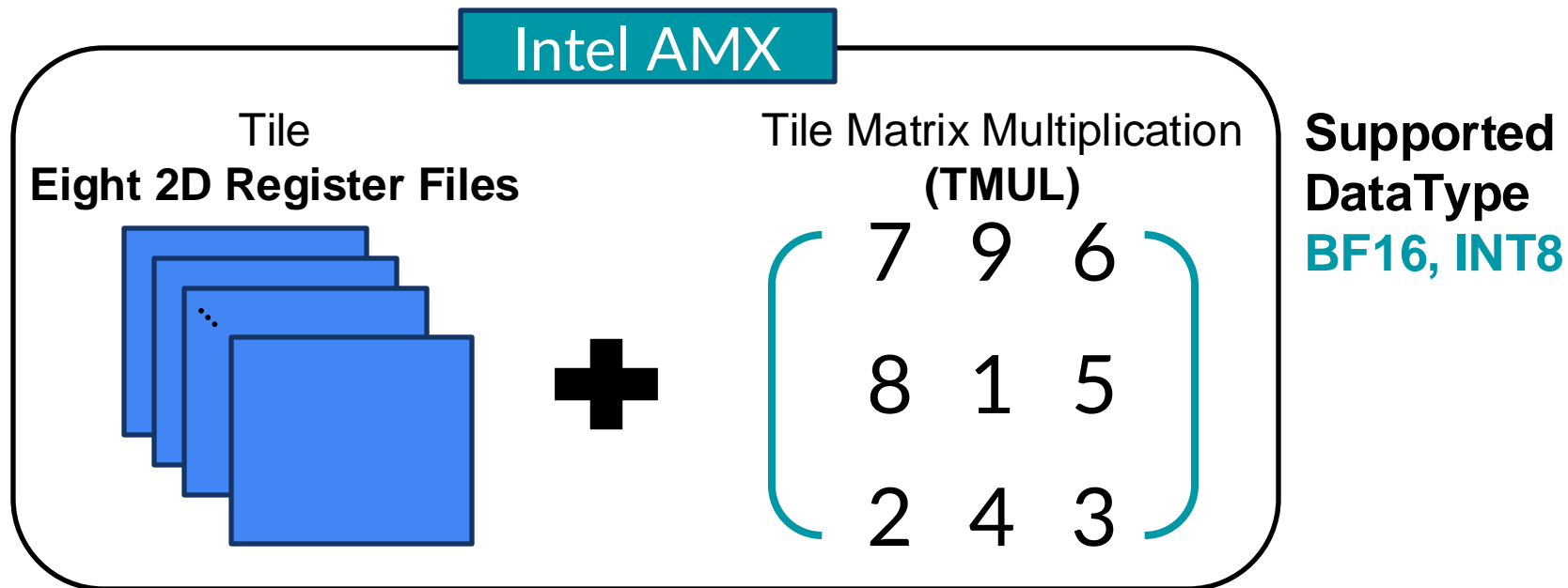
Possible Hardware Options for LLM Inference

Options	Cost	Accuracy	Latency
CPU	Low	High	Low-High
Single-GPU with CPU offloading	Medium	High	Low- High
Single-GPU with quantization (without CPU-offloading)	Medium	High- Medium	Low
Multi-GPUs	Very High	High	Very Low

**Main
Focus**

Opportunities in Latest CPUs: (1) Dedicated Accelerators

- Recent CPUs offer **GEMM accelerators with extended ISA support**
 - Intel Advanced Matrix eXtension (AMX), ARM Scalable Matrix Extension (SME), etc.



Opportunities in Latest CPUs: (2) Large Memory Capacity

- CPU servers provide **larger memory capacity** than that of GPUs

CPU

Could be expanded

There are two key opportunities for CPU LLM inference

1. **Dedicated accelerator with ISA extension**
2. **Larger memory capacity with HBM**

High Capacity
Low bandwidth

Low Capacity
High bandwidth

8X higher capacity
NVIDIA H100 GPU
HBM 80GB

Evaluation Methodology

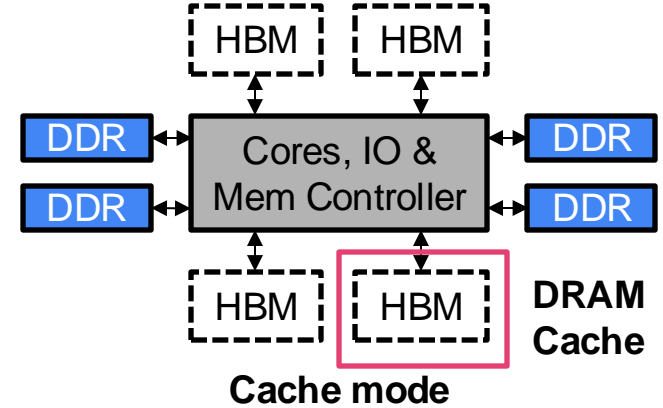
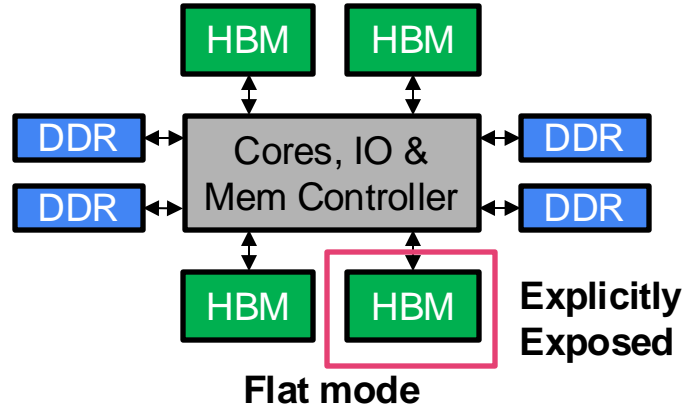
- Use Intel Extension for Pytorch (IPEX) for CPU LLM inference
- Evaluated LLMs: OPT (1.3B, 6.7B, 13B, 30B, 66B) , LLaMA2 (7B, 13B, 70B)
- Metrics: End-to-End Latency & Throughput (Generated output tokens/s)

	Sapphire Rapids CPU (SPR)
CPU Model	Xeon 4 th Max 9468
# of Cores (Per socket) / # of Socket	48 / 2
Compute Throughput	25.6 (AVX-512) / 206.4 (AMX) TFLOPS
L1/L2 (per core)	48KB/ 2MB
LLC	105MB
Memory Capacity	DDR5 512GB, HBM 128GB
Memory Bandwidth	DDR5: 233.8 GB/s, HBM: 588 GB/s

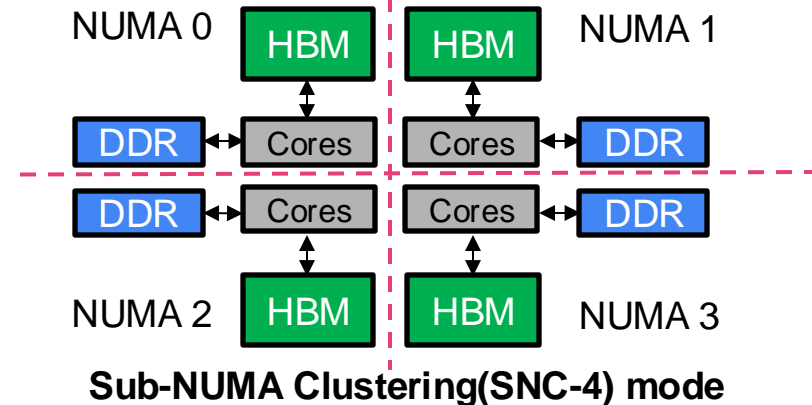
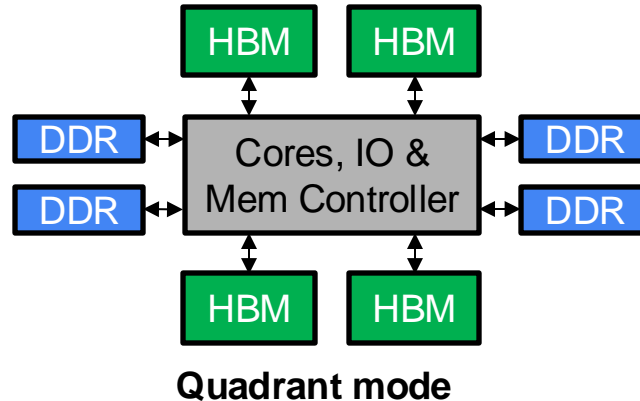
Memory bandwidth is measured on single-socket using STREAM benchmark

Key Intel CPU Configurations: Memory, Clustering Modes

Memory Mode



Clustering Mode



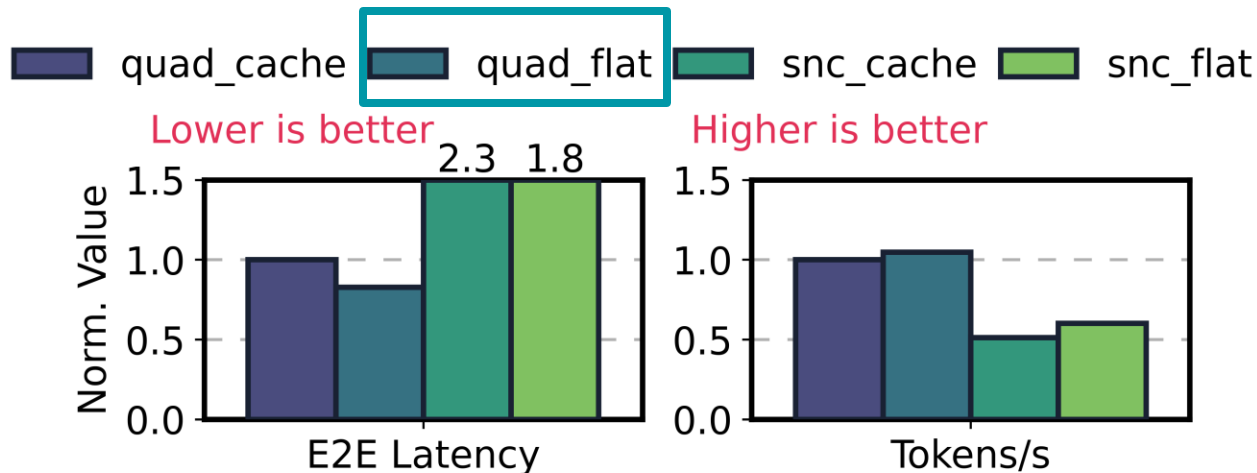
Questions We Aim to Answer for Optimal Performance

- What is **the optimal clustering and memory configuration** for LLM inference?
- What is **the optimal number of CPU cores** for LLM inference?

Performance Impact of Clustering and Memory Modes

- Compare the averaged performance across all LLMs and batch sizes (1 to 32)
 - Each result is normalized to **Quadrant_Cache** (quad_cache) configuration
 - HBM memory is prioritized for **flat mode using Linux numactl**

Best configuration



Performance Impact of the Number of CPU Cores

- Compare the averaged performance across all LLMs and batch sizes (1 to 32)
 - Each result is normalized to **12 cores** configuration
 - All configurations use **quad_flat** mode



Using Quad with Flat and 48 cores delivers the best results

GPU Server Configurations

- We use **FlexGen** for offloading-based LLM inference on GPUs

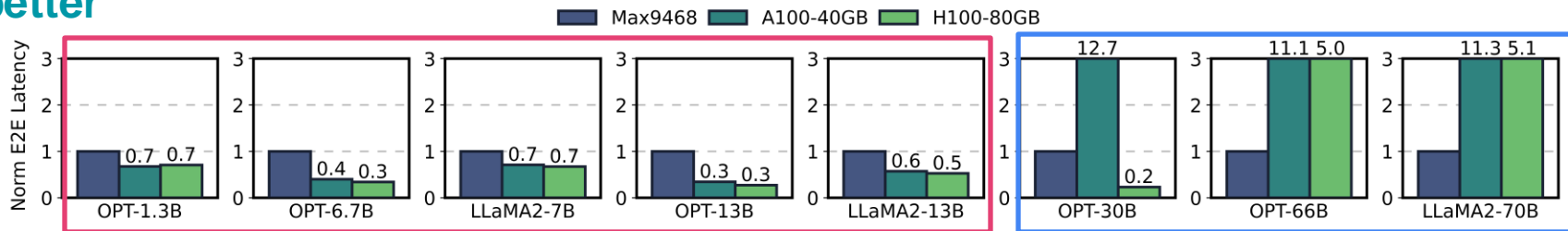
	A100-40GB GPU	H100-80GB GPU
# of SMs	108	132
Compute Throughput	312 TFLOP	989 TFLOP
L1/L2	192KB / 40MB	256KB / 50MB
Memory Capacity	HBM 40GB	HBM 80GB
Memory Bandwidth	1299.9 GB/s	1754.4 GB/s
Interconnect	PCIe 4.0, 64GB/s	PCIe 5.0, 128GB/s

Performance Comparison: SPR Max CPU vs GPUs

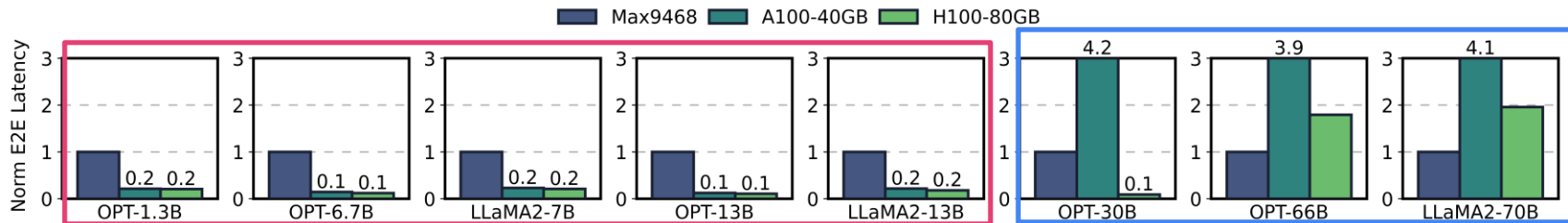
Lower
is better

Batch size 1

Input length: 128
Output length: 32



Batch size 16



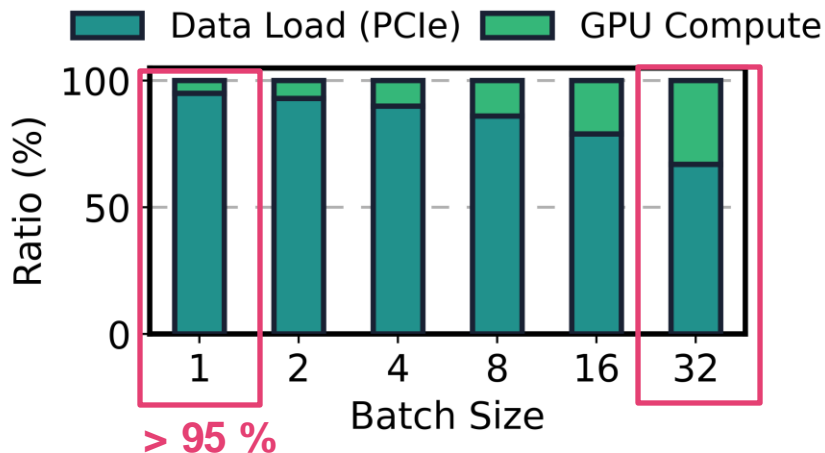
GPUs outweigh CPU
for smaller models

CPU performs better than GPUs
for larger models

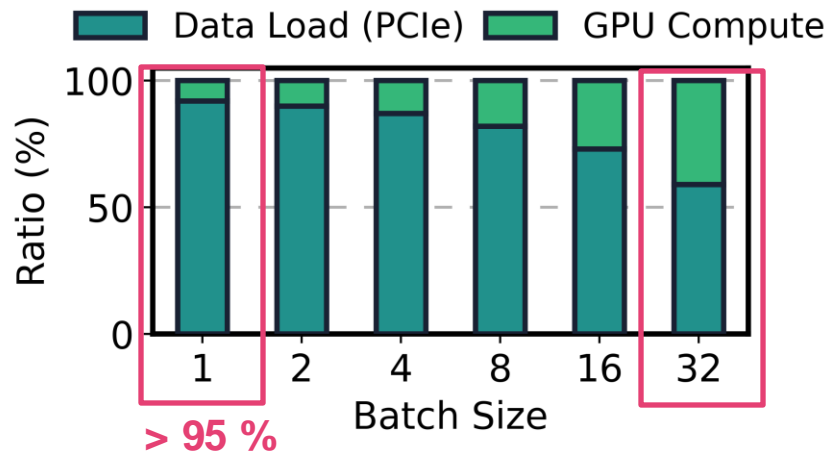
GPU Execution Time Breakdown

- Offloading-based LLM inference suffers from **significant PCIe transfer times**

OPT-30B model in A100-40GB



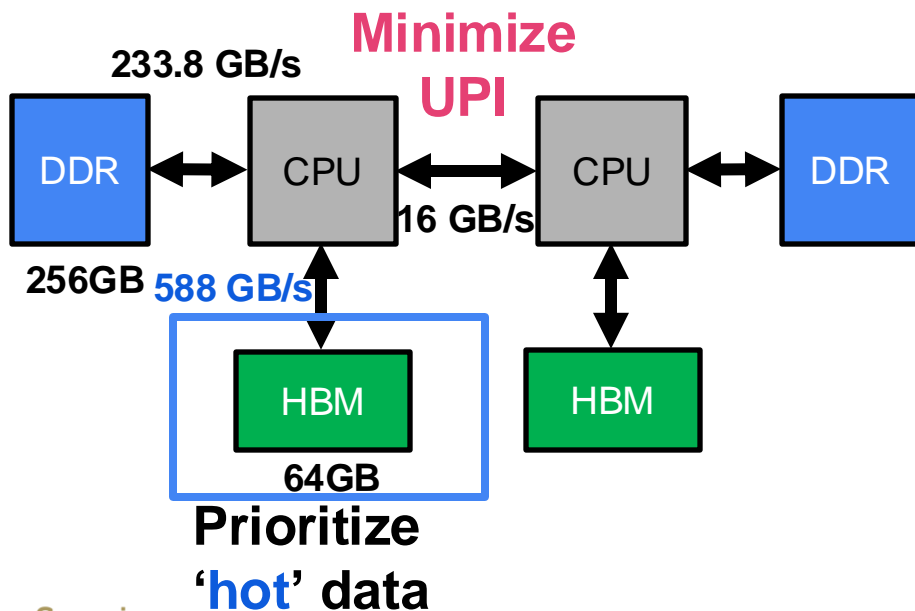
OPT-66B model in H100-80GB



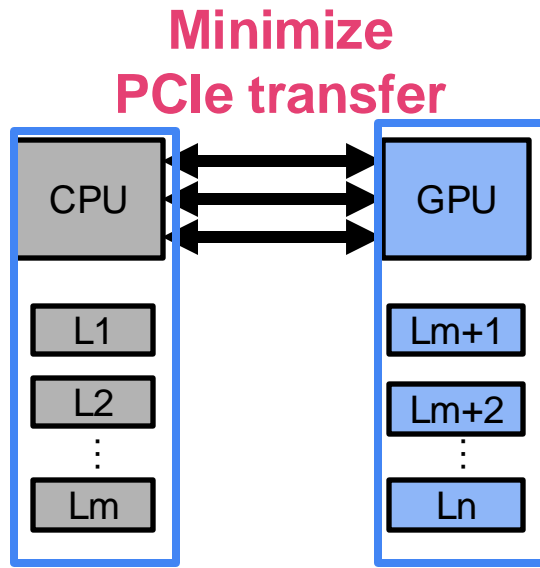
Compute intensive

Future Research Direction for Efficient CPU LLM Inference

NUMA aware data placement



CPU-GPU Hybrid LLM Inference



Splitting Layers to
fully **utilize both CPU and GPU**

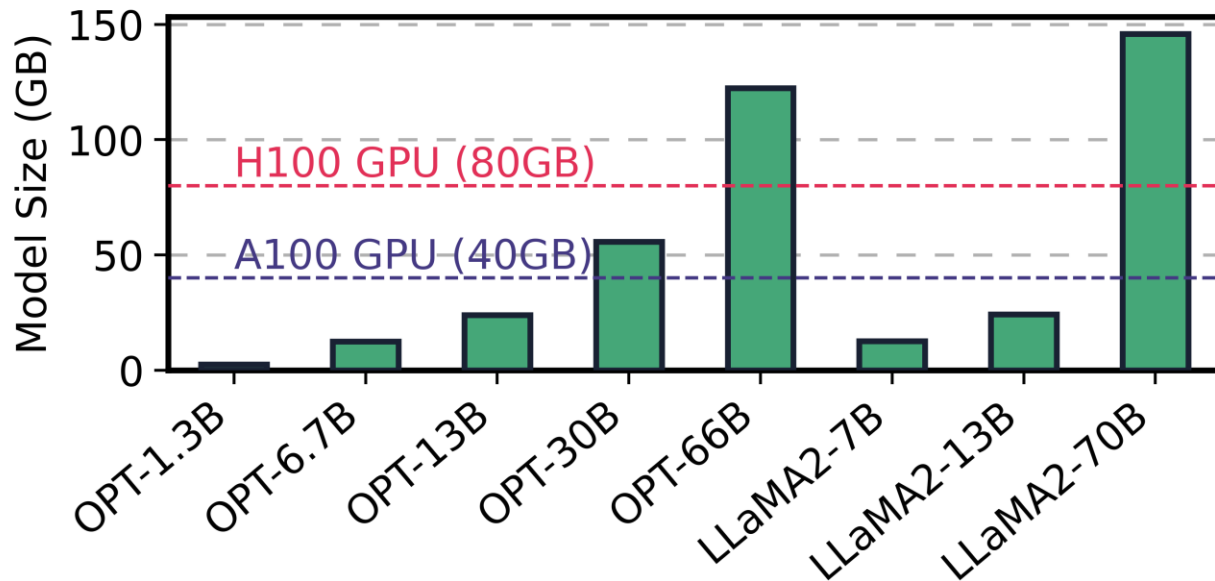
Conclusion

- LLM inference demands substantial memory, often exceeding GPU memory
 - Offloading-based LLM inference suffer from performance degradation due to PCIe transfer
- Recent CPUs have potential for LLM inference
 - Dedicated GEMM Accelerators with ISA support
 - Larger memory capacity with HBM that could be further expanded CXL
- CPUs can perform better than GPUs, specifically for larger models

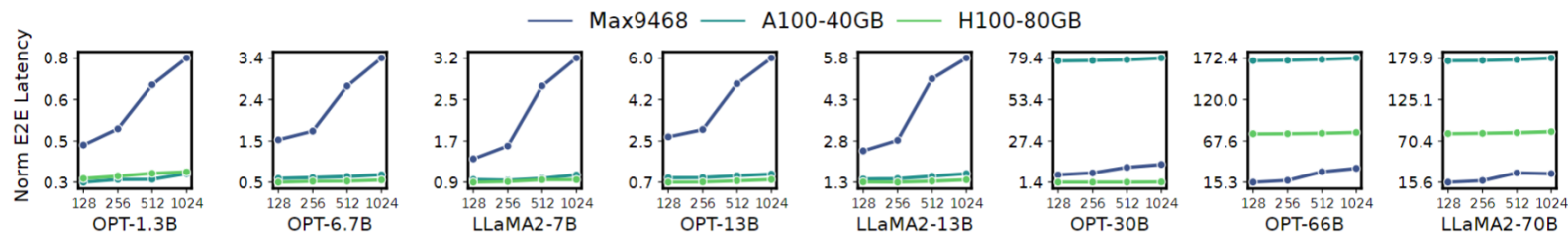
Backup Slides

Challenges in LLM Inference: Huge Model Size

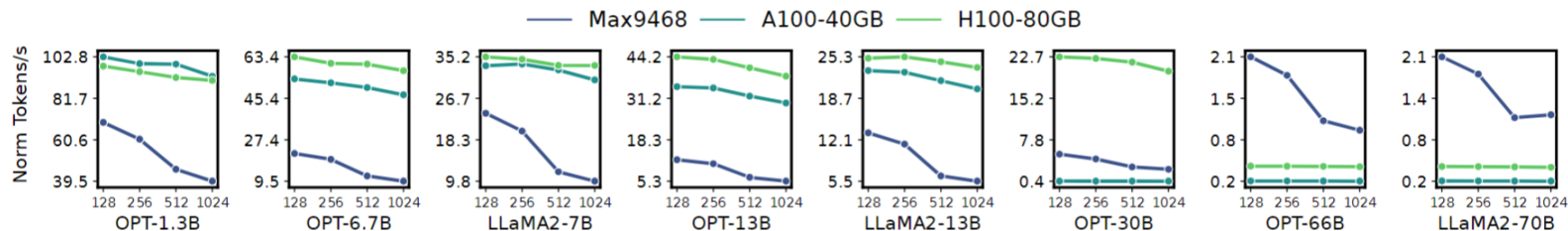
- LLMs are growing larger due to scaling laws [1]



Sensitivity Study: Sequence Length



(a) End-to-End latency comparison



(b) Throughput comparison

Evaluation Methodology

- Use Intel Extension for Pytorch (IPEX) for CPU LLM inference
- Evaluated LLMs: OPT (1.3B, 6.7B, 13B, 30B, 66B) , LLaMA2 (7B, 13B, 70B)
- Metrics: End-to-End Latency & Throughput (Generated output tokens/s)

	IceLake CPU (ICL)	Sapphire Rapids CPU (SPR)
CPU Model	Xeon 3 rd 8352Y	Xeon 4 th Max 9468
# of Cores (Per socket) / # of Socket	32 / 2	48 / 2
Compute Throughput	18.0 TFLOP (AVX-512)	25.6 (AVX-512) / 206.4 (AMX) TFLOPS
L1/L2 (per core)	48KB/ 1.25MB	48KB/ 2MB
LLC	48MB	105MB
Memory Capacity	DDR4 256GB	DDR5 512GB, HBM 128GB
Memory Bandwidth	156.2 GB/s	DDR5: 233.8 GB/s, HBM: 588 GB/s

Memory Bandwidth is measured on single-socket using STREAM benchmark

Performance Comparison: ICL CPU vs SPR CPU

- We use 32 cores for ICL CPU and 48 cores for SPR CPU
 - Each result is normalized to **ICL CPU results at the same batch size.**

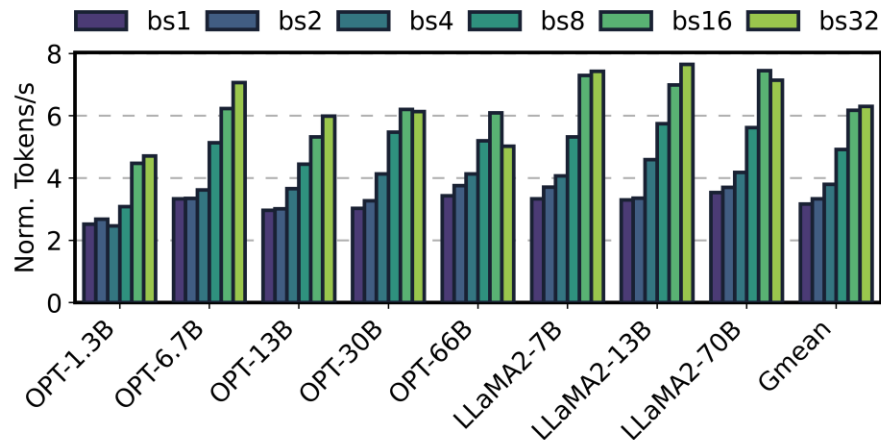
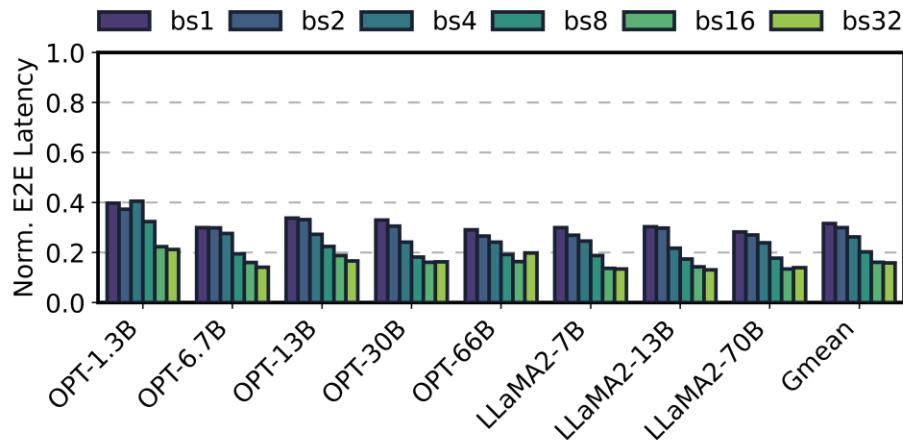
Lower
is better

Compute intensive



Higher
is better

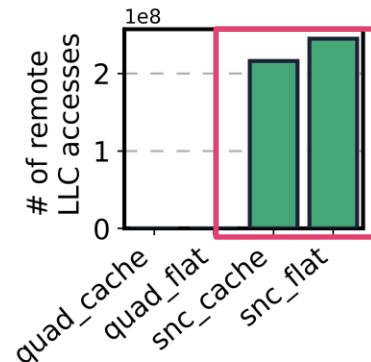
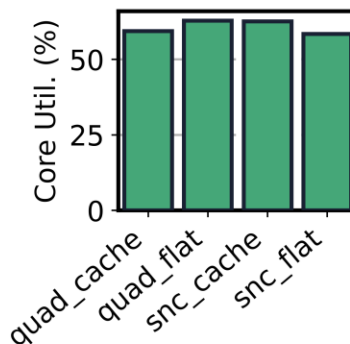
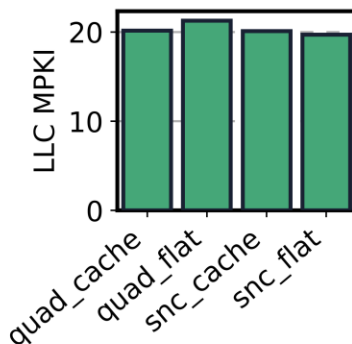
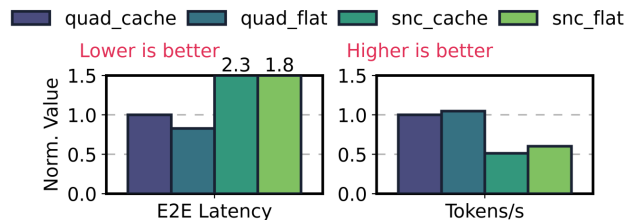
Input length: 128
Output length: 32



Performance Impact of Clustering and Memory Modes

- Compare the averaged performance across all LLMs and batch sizes (1 to 32)
 - Each result is normalized to **Quadrant_Cache** (quad_cache) configuration
 - HBM memory is prioritized for **flat mode using Linux numactl**

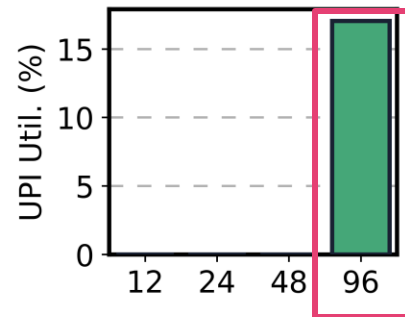
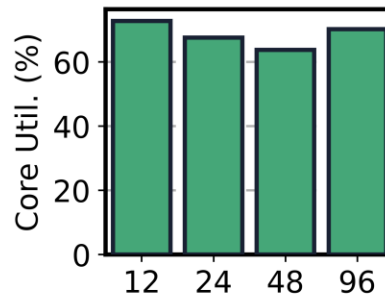
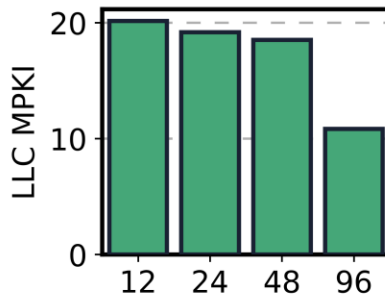
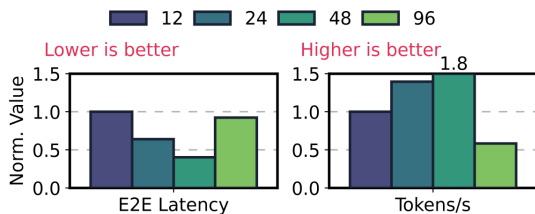
LLaMA2-13B model with batch size 8



Performance Impact of the Number of CPU Cores

- Compare the averaged performance across all LLMs and batch sizes (1 to 32)
 - Each result is normalized to **12 cores** configuration
 - All configurations use **quad_flat** mode

LLaMA2-7B model with batch size 8



inter-socket
communication