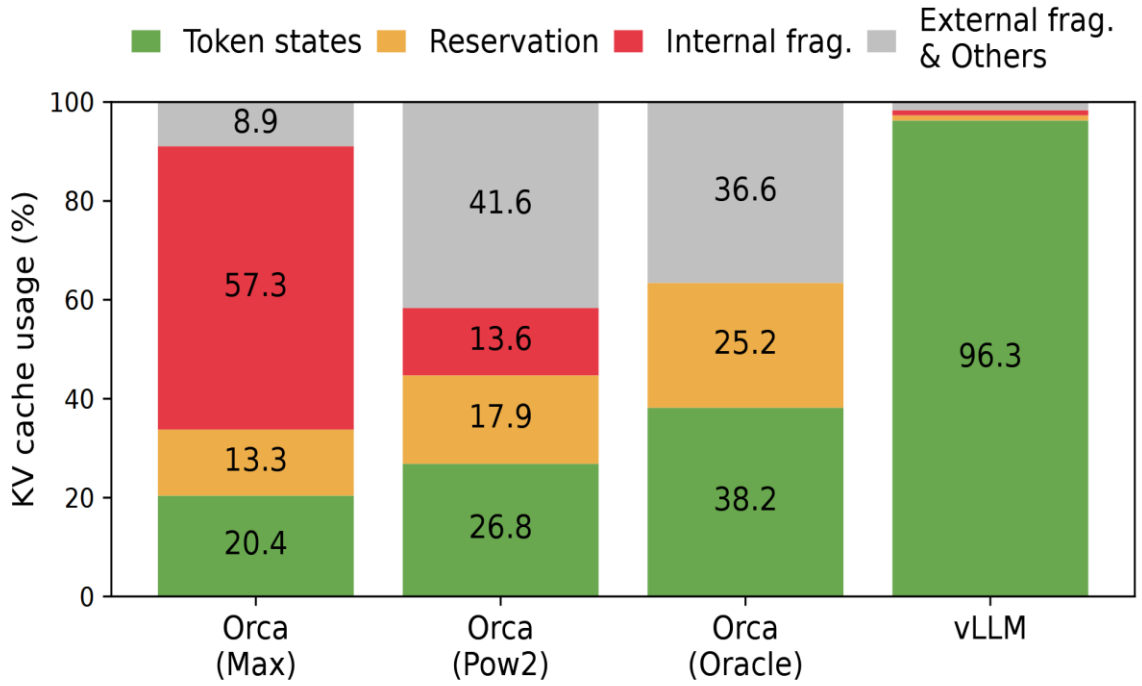
vLLM – Memory Optimizations and Multi-LoRA Serving Prototype

PRESENTED BY: OGNEN PENDAROVSKI

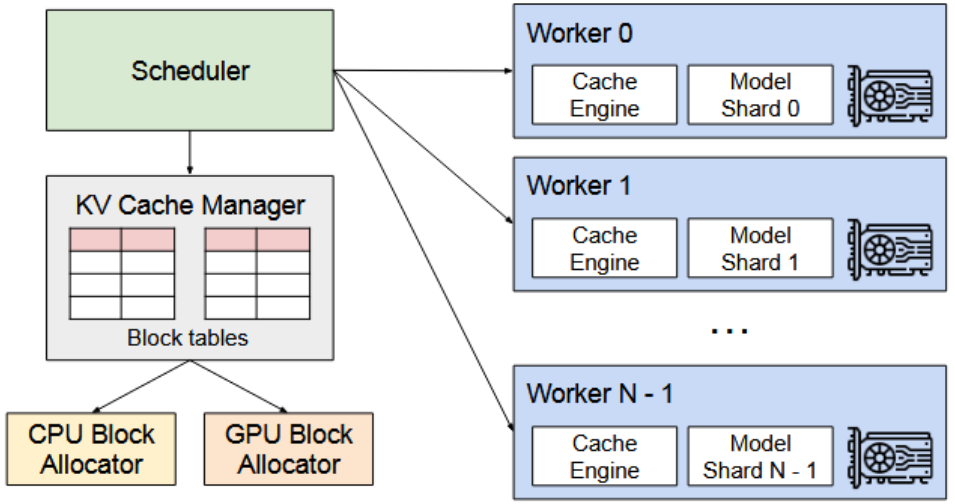
The Problem

- Dynamically changing KV cache size is inefficiently handled by LLM serving systems:
  - Internal and external memory fragmentation
  - Inability to share memory



The Solution

- vLLM introduces PagedAttention:
  - Idea analogous to virtual memory in operating systems
  - Stores continuous KV cache in non-contiguous memory blocks
  - Eliminates most of the fragmentation memory waste
  - Allows more complex decoding methods



Part 1 – Benchmarking

- Main objective – reproduce and test vLLM performance across multiple different LLMs and various serving parameters.
- Metrics tracked:
  - TTFT (Time to First Token) – latency of pre-fill phase
  - TPOT (Time per Output Token) – decode phase throughput
  - Total latency
  - KV cache overhead
- Parameters: batch size, context length, quantization levels

Part 2 – Multi-LoRA Serving Prototype

- Implement prototype serving logic for dynamically routing queries to associated LoRA adapters.
  - One frozen-weights backbone with multiple LoRA kernels
  - Adding additional metadata to queries for association with specific LoRA kernel (specific tasks, writing styles, languages, etc.)
  - Queries processed in heterogeneous batch
- Benchmark the multi-LoRA server on standard benchmarks, as well as switching overhead.
  - Parameters: LoRA count, rank, quantization

Project Timeline

- Phase 1 (Dec 12 – Dec 19):
  - Set up required software environment
  - Select models and parameter values
  - Write and run benchmarking script
- Phase 2 (Dec 20 – Dec 31):
  - Implement multi-LoRA serving
  - Update script with additional metrics
  - Perform evaluation
- Phase 3 (Jan 1 – Jan 20):
  - Compile data
  - Write final report

References

[1] Kwon, Woosuk, et al. "Efficient memory management for large language model serving with PagedAttention." *Proceedings of the 29th symposium on operating systems principles*. 2023.

[2] Hu, Edward J., et al. "Lora: Low-rank adaptation of large language models." *ICLR* 1.2 (2022): 3.

[3] https://github.com/vllm-project/vllm/tree/main