Efficient Memory Management for Large Language Model Serving with *PagedAttention*

Authors: Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, Ion Stoica

Venue: SOSP, October 2023

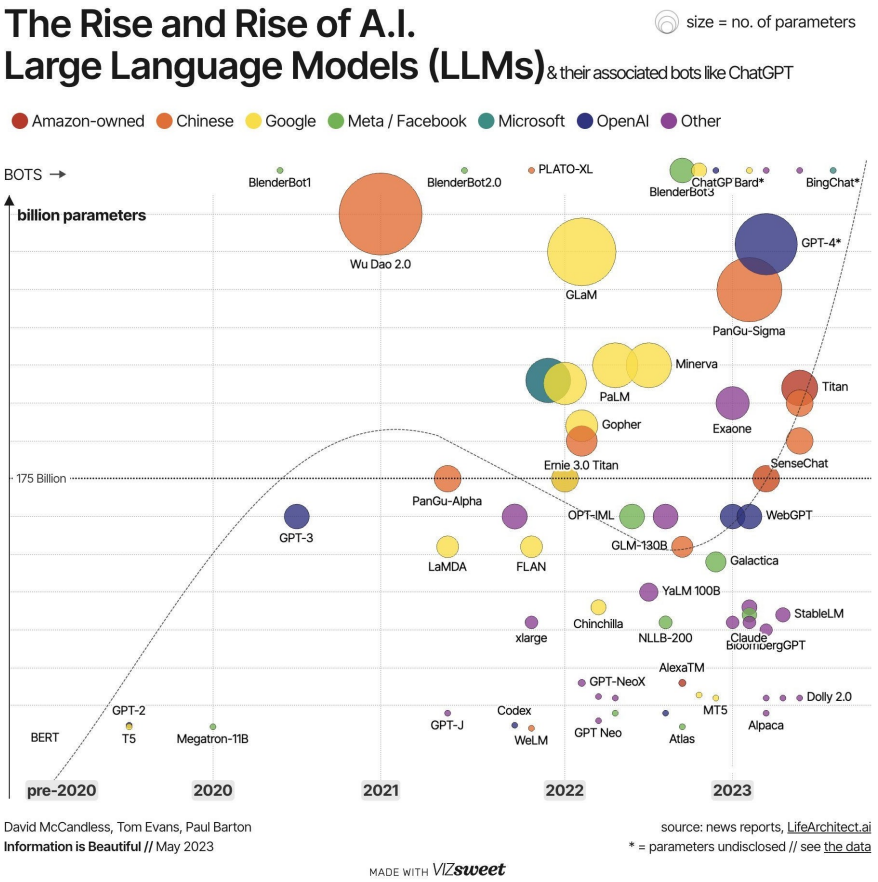
Presenter: Darius Grassi

Slides adapted from: Venkataraman CS 744 '24, Maddison CSC 2541 '25

Serving LLMs is Expensive

- Many GPUs required for production-scale LLM services
- Each GPU can only serve a handful of requests per second
  - For LLaMA-13B and moderate-size inputs, 1 A100 can process < 1 requests per second

Goal: maximize serving **throughput** and GPU **utilization**

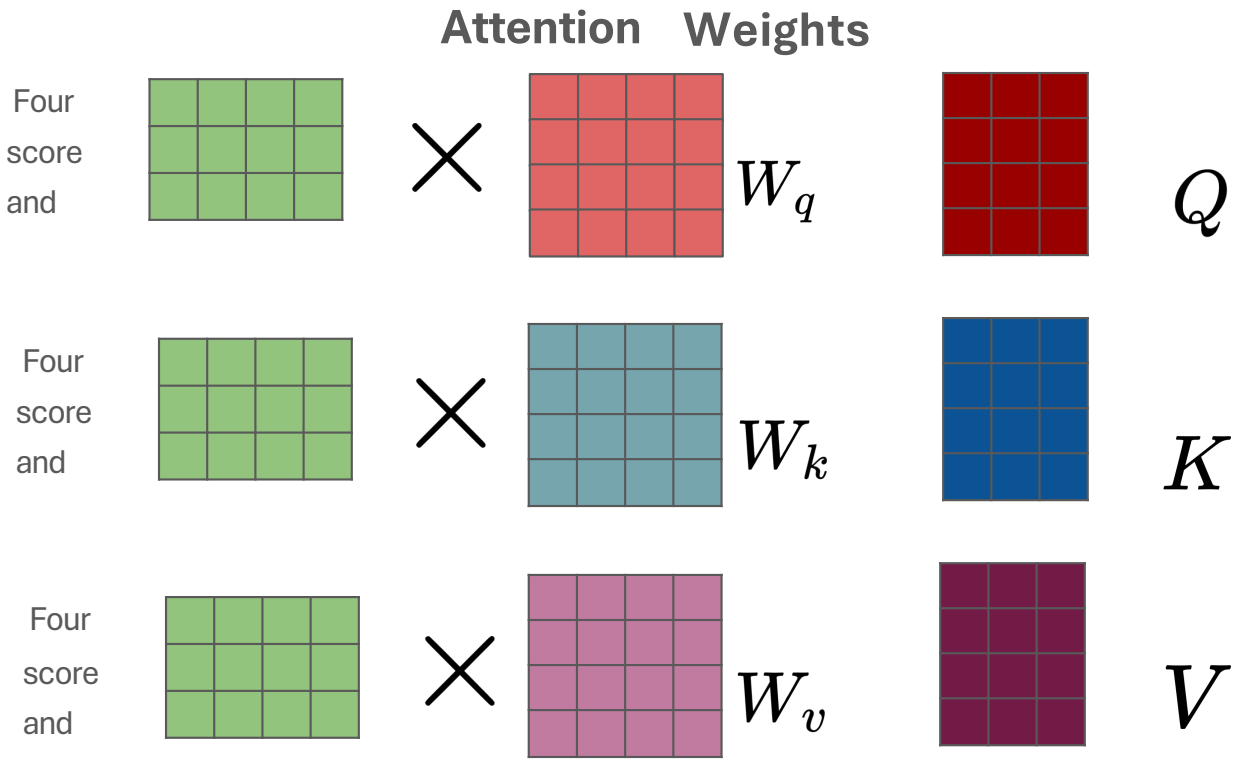


Zuckerberg's Meta Is Spending Billions to Buy 350,000 Nvidia H100 GPUs

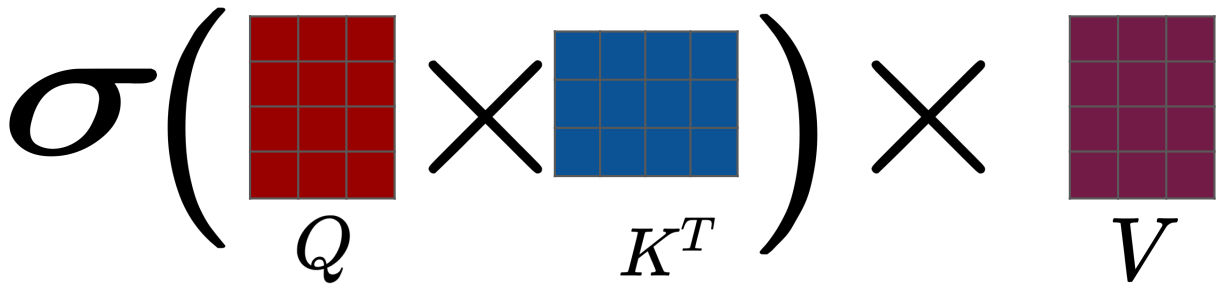
In total, Meta will have the compute power equivalent to 600,000 Nvidia H100 GPUs to help it develop next-generation AI, says CEO Mark Zuckerberg.

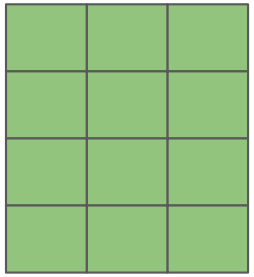
Background: Self-Attention

Example Sentence :Four score and



Background: Self Attention

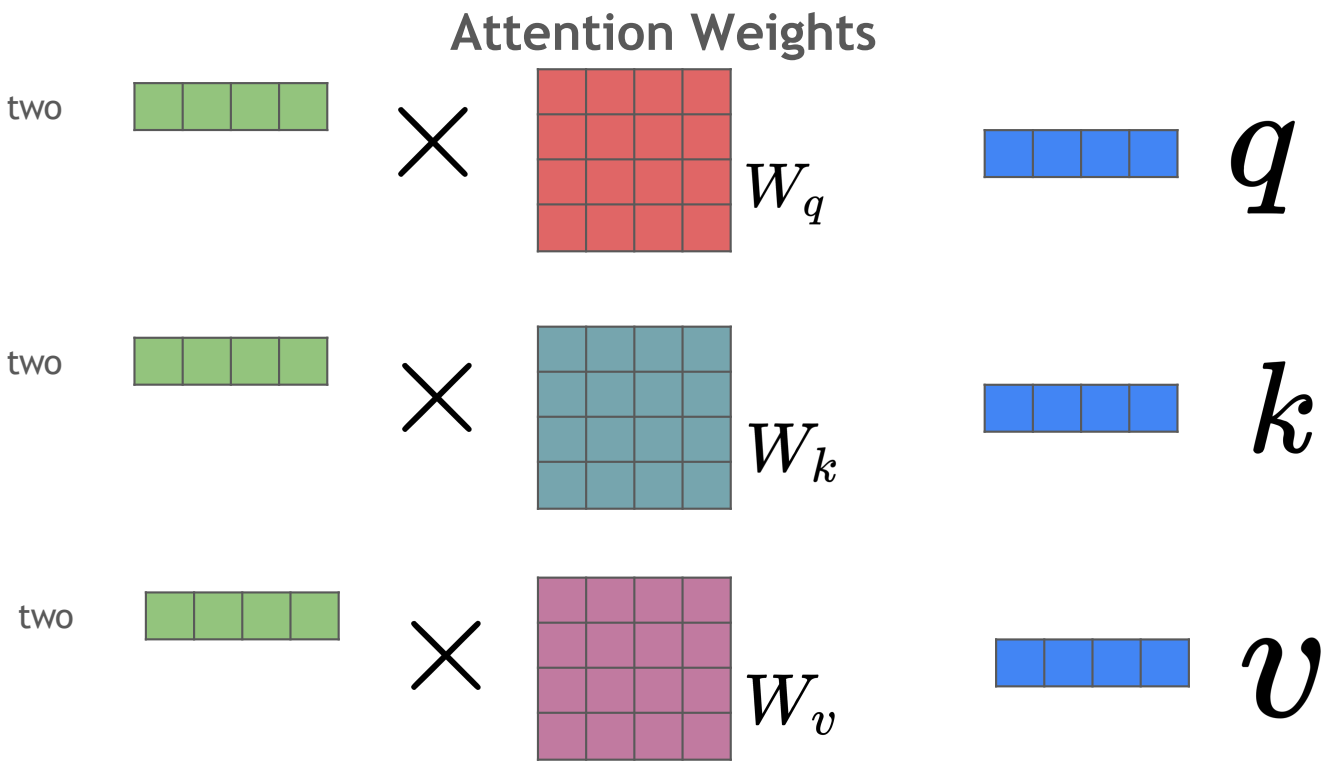




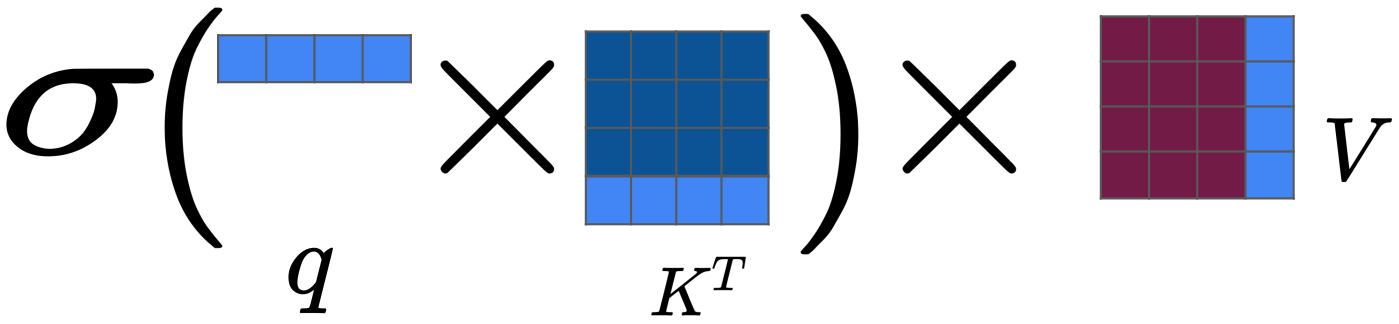
Four score and

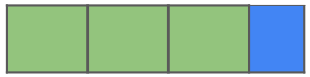
Background: Auto Regressive Decoding

Example Sentence : Four score and two



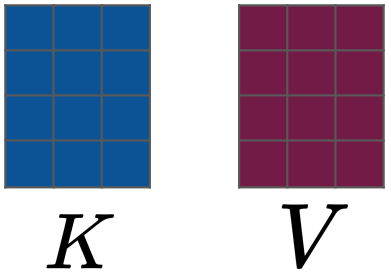
Background: Auto Regressive Decoding





Four score and two

Background: K,V cache

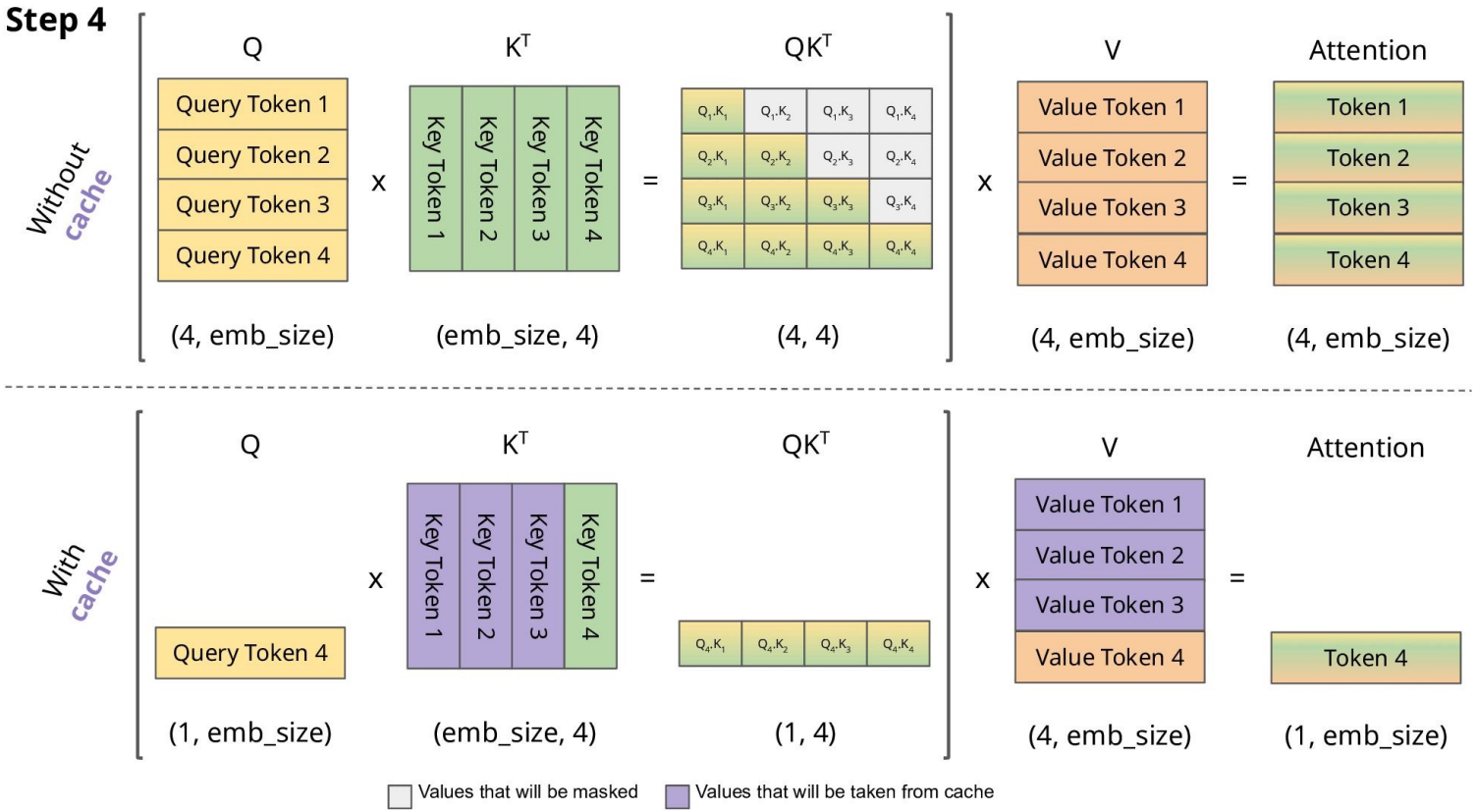


- Store Key and Value vectors associated within the context length
- Minimize re-generation of key and value vectors associated with prior tokens.

Self-Attention with KV Cache

T : Sequence length

d : Model embedding size



$O(T^2 d)$ Time

$O(Td)$ Time

$O(Td)$ Space

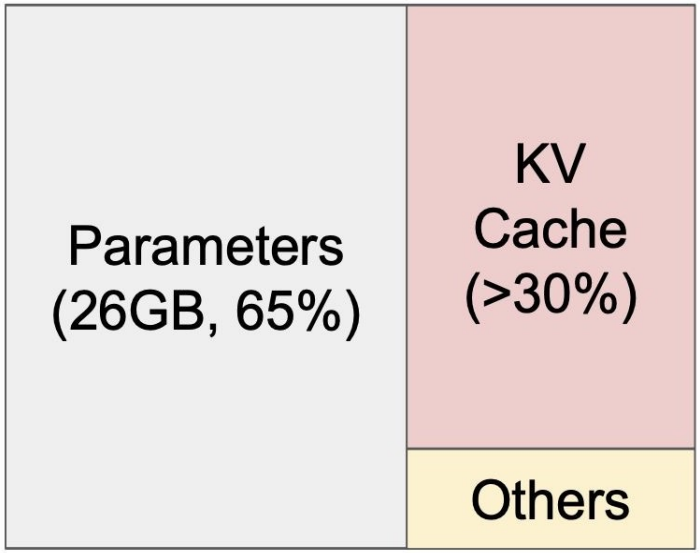
Motivation

- To store K,V cache usually memory is allocated for the full context length.

- Example - Context length for LLaMa models is 2048.
  - Solutions prior to PagedAttention pre-allocated memory for the K,V for full 2048
  - Request needs only 40 tokens → wastage of memory

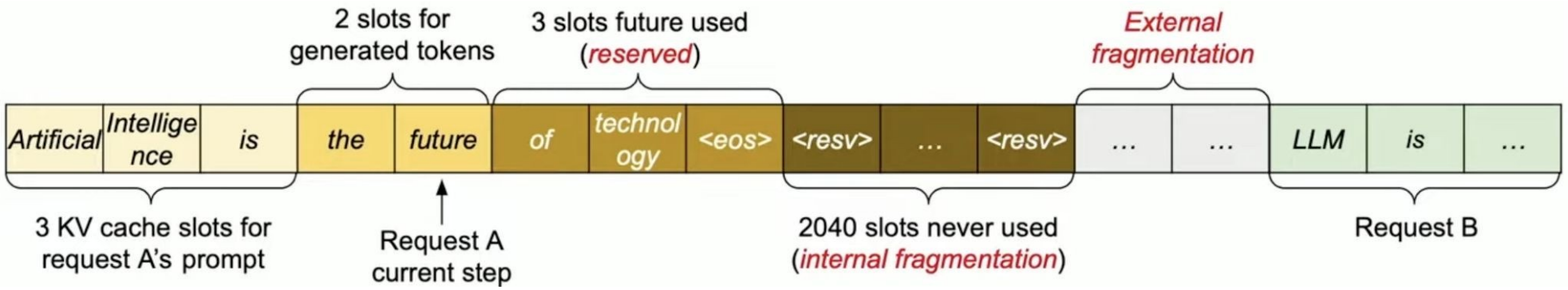
Motivation

- Efficient management of KV cache is crucial for high-throughput LLM serving
- Batched requests → shared KV cache
- Workload throughput becomes **memory-bound**



NVIDIA A100 40GB

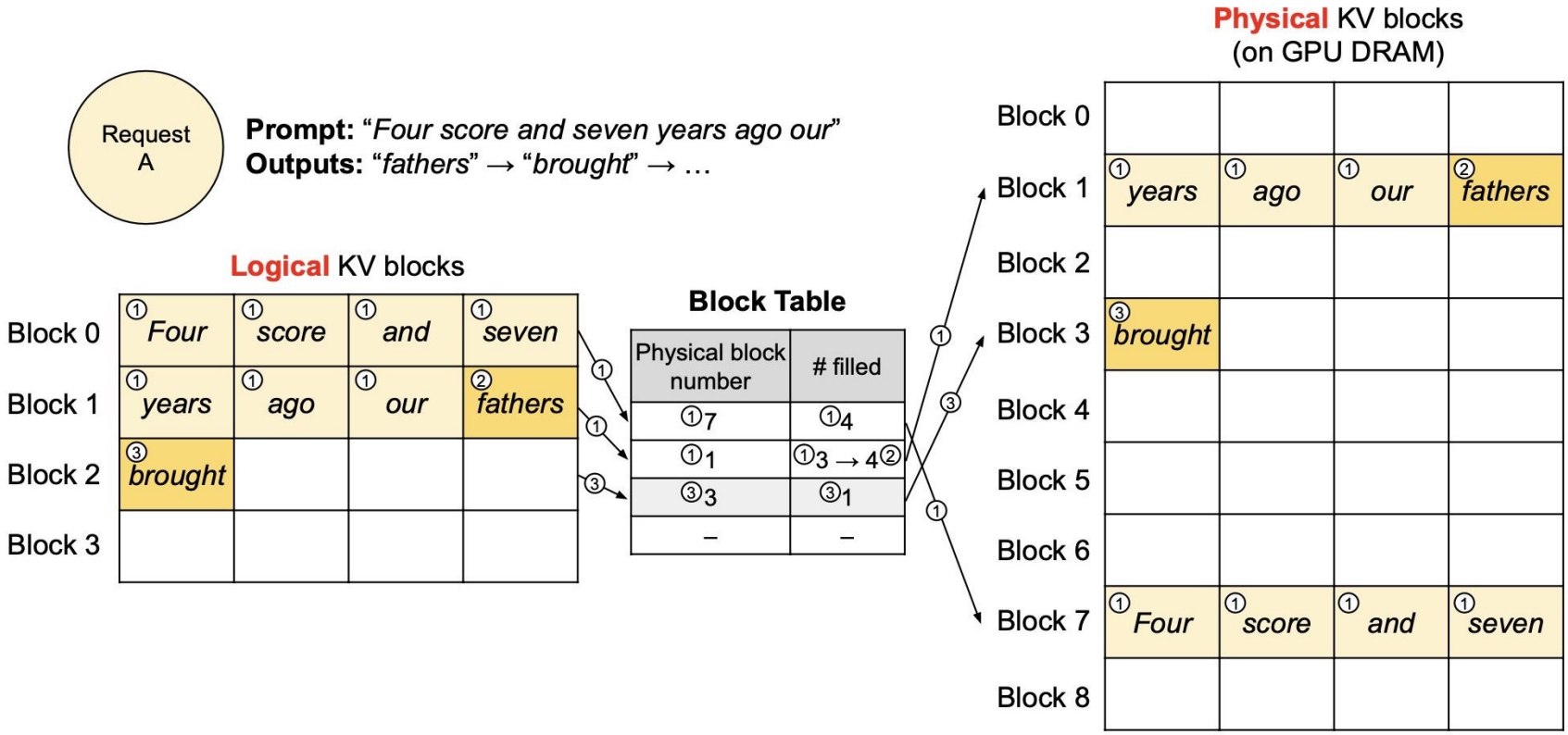
Memory waste in shared KV Cache



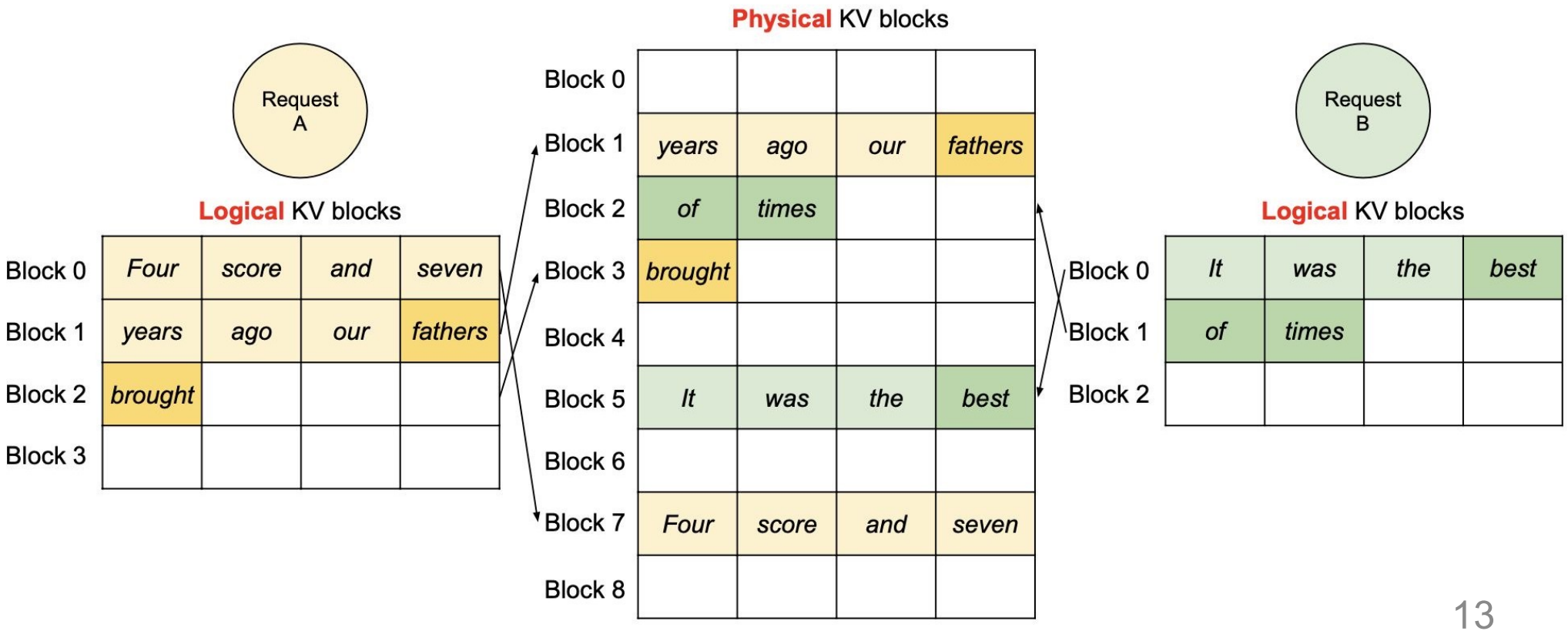
- **Internal fragmentation**: memory reserved for a request but left unused because the final output length is shorter than expected
- **Reservation**: memory currently unused but reserved for future use by the same request
- **External fragmentation**: small unused memory gaps between allocations caused by varying sequence lengths across different requests

PagedAttention

- Use a mechanism similar to a page table used by operating systems

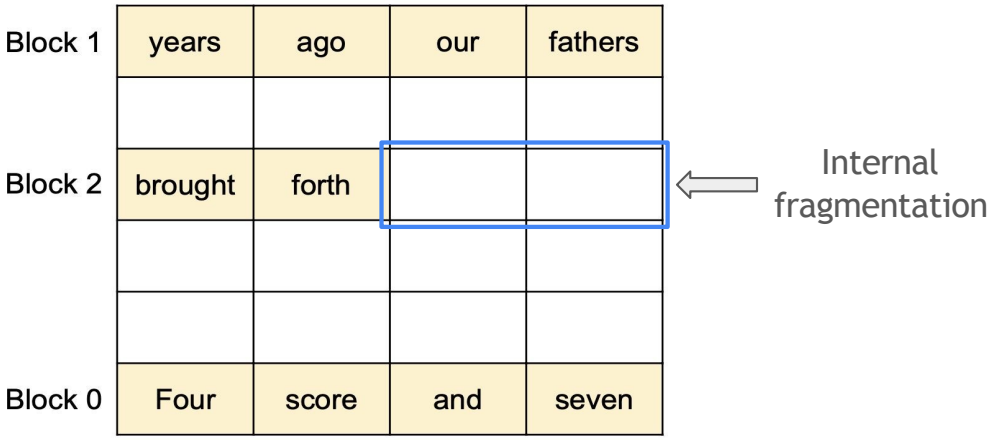


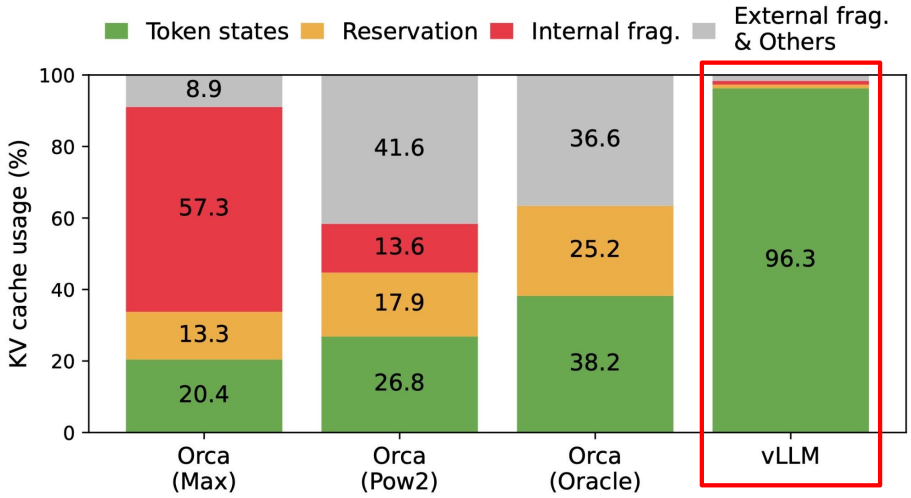
Handling multiple requests (continuous batching)



Memory Efficiency of PagedAttention

- Minimal internal fragmentation
  - # of wasted tokens per sequence < block size
- No external fragmentation





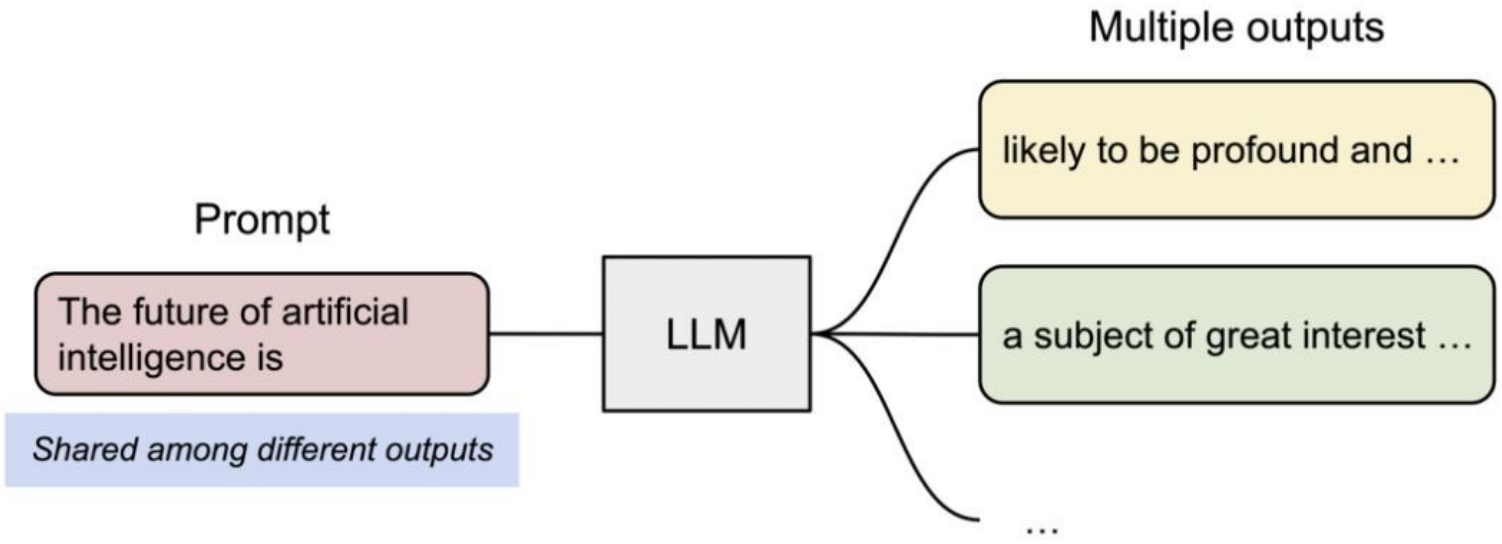
- With PagedAttention, wasted KV cache space is < 4% (3-5x improved memory utilization)

PagedAttention

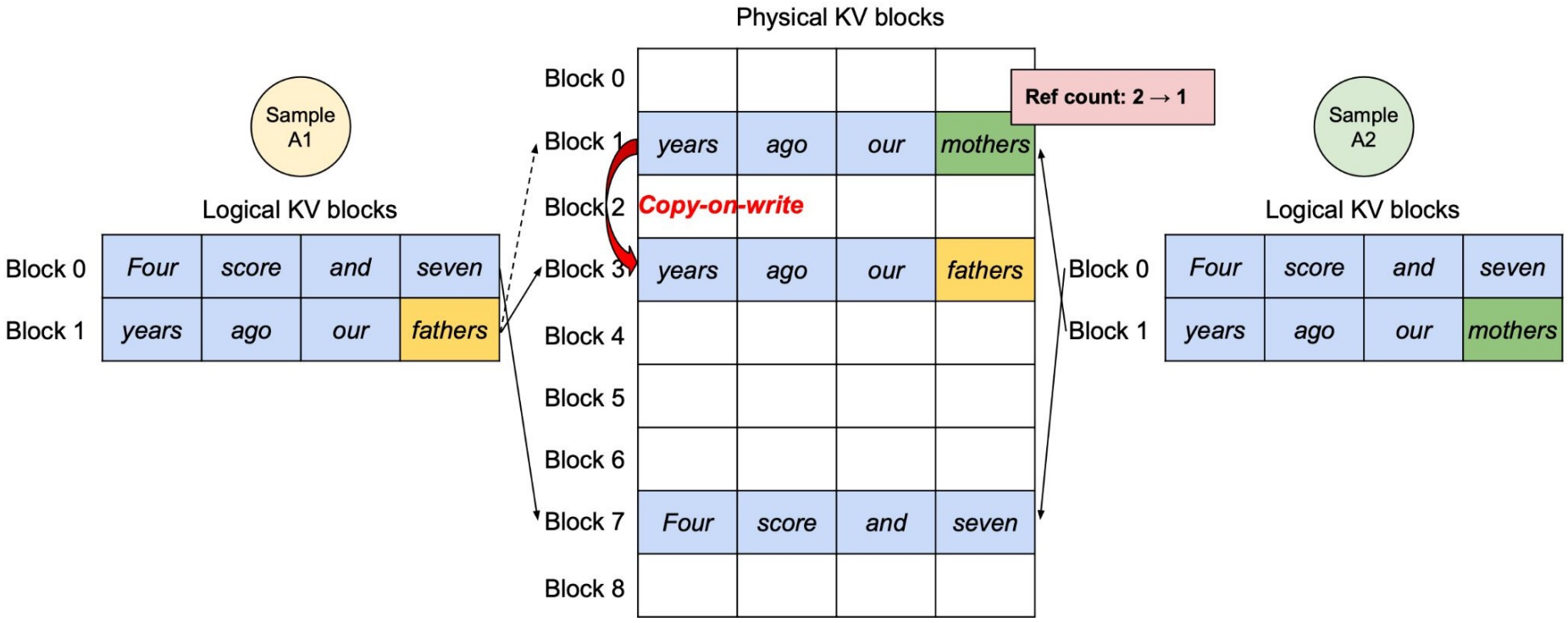
- Use a block manager to keep block tables associated with each GPU.

- Enable efficient sharing of memory in case of parallel sampling, shared prefixes, beam search.

Memory Sharing: Parallel Sampling

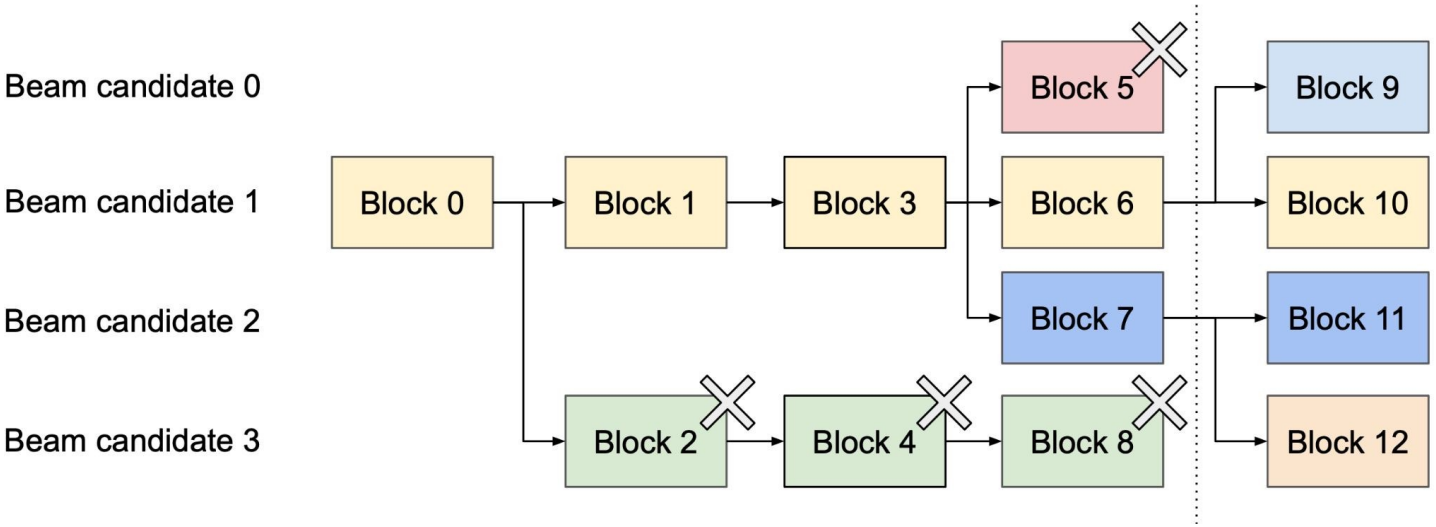


Parallel Sampling Copy-on-Write

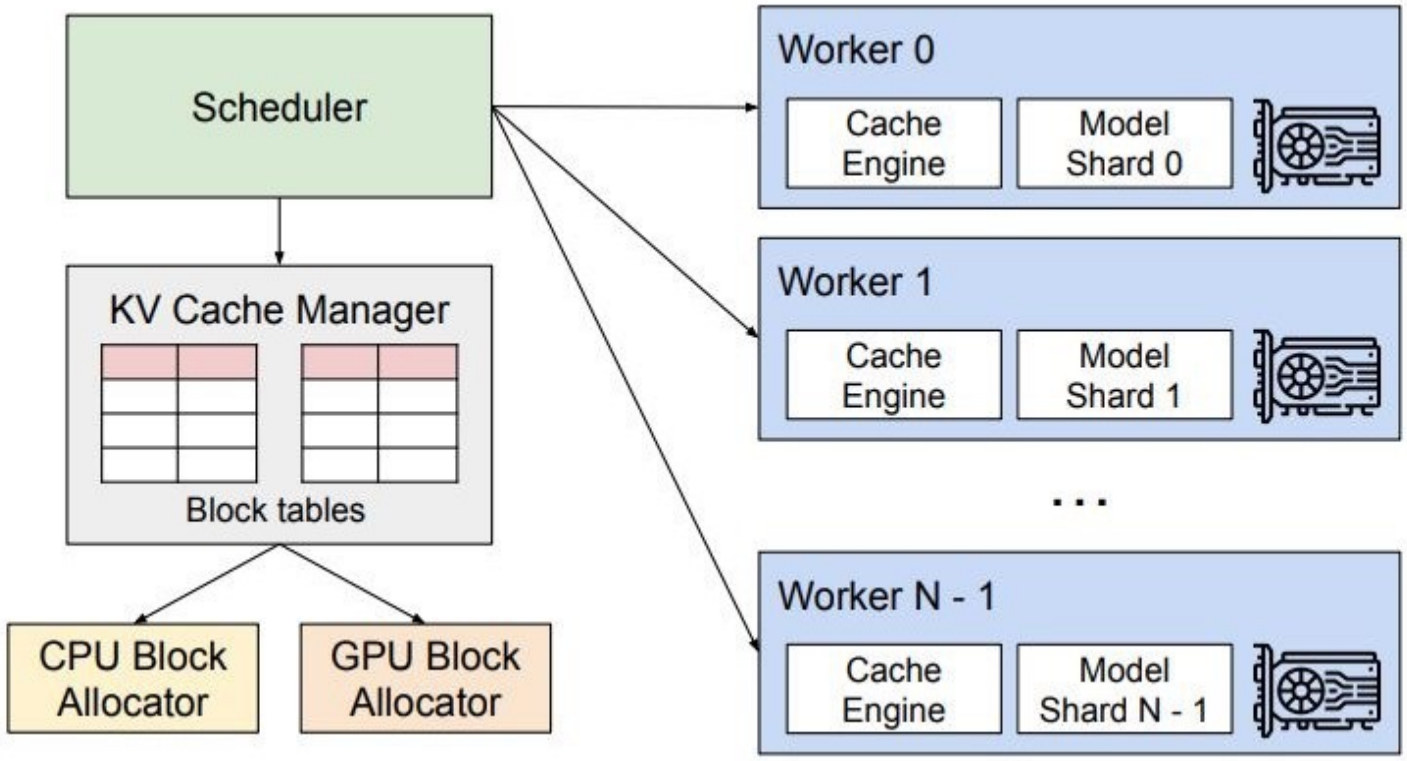


Memory Sharing: Beam Search

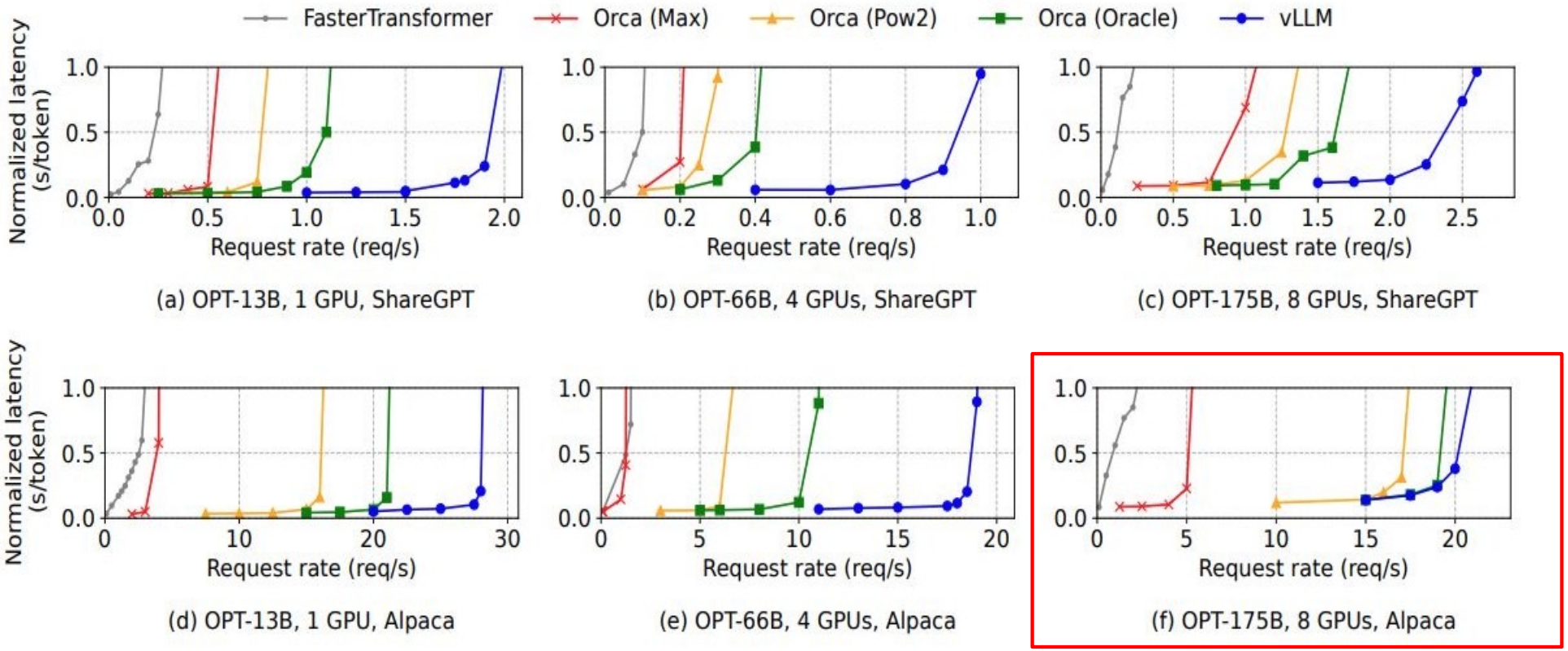
- Efficiently supported by dynamic block mapping and copy-on-write mechanism



PagedAttention vLLM - Design

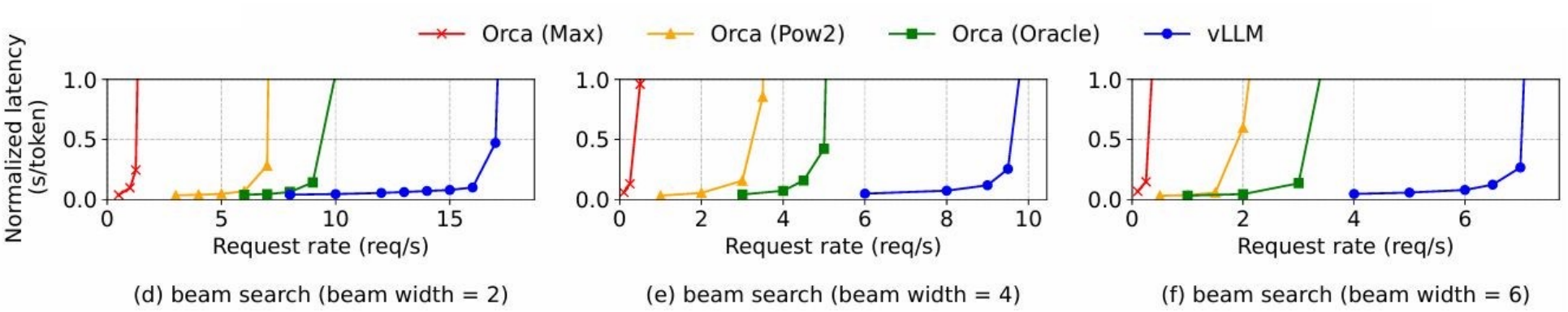


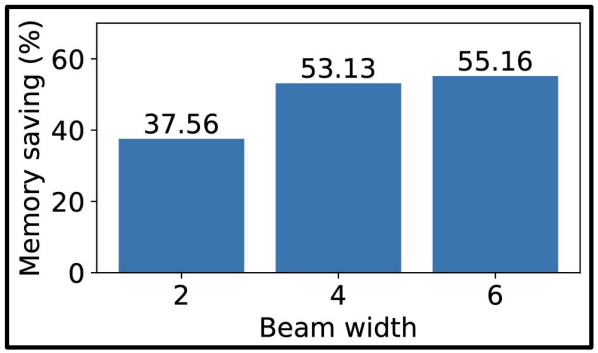
Evaluation



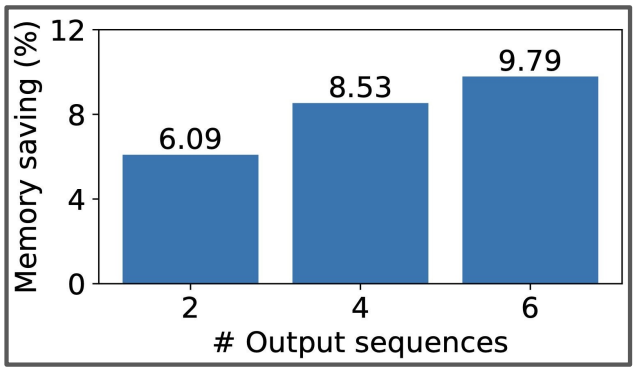
- **Setup** - A100 GPUs single node
- Normalized latency
- x-axis

Beam Search with PagedAttention





Parallel sampling savings →



Main Takeaways

- Reduces memory fragmentation with paging
- Reduces memory usage with KV block sharing
- Enables batching of more requests
- Increases the throughput of LLM inference

Limitations

- Choice of block size can have a substantial impact on the performance
- Increased kernel complexity and overhead
- Limited effectiveness in short-sequenced workloads
- Doesn’t natively support various models/GPU architectures

Thank You + Q&A

Peer Review

Summary:

This paper addresses a critical bottleneck in large language model (LLM) serving: inefficient memory management of the KV cache. The authors identify that existing systems suffer from significant memory waste due to internal fragmentation, external fragmentation, and an inability to share memory effectively across requests. This waste limits the batch size, thereby constraining the overall throughput of the serving system.
The core contribution is PagedAttention, a novel attention algorithm inspired by classic operating systems concepts of virtual memory and paging. PagedAttention allows for the KV cache to be stored in non-contiguous memory blocks. Building on this, the authors present vLLM, an LLM serving engine that leverages PagedAttention to achieve near-zero memory waste. The authors demonstrate through extensive experiments that vLLM improves throughput by 2-4x over state-of-the-art systems like FasterTransformer and Orca across a variety of models, sequence lengths, and complex decoding algorithms.

Peer Review

Strengths And Weaknesses:

Strengths:
- Novel and Elegant Analogy: The core idea of applying the well-understood concepts of virtual memory and paging from operating systems to the problem of KV cache management is both novel and highly effective. This analogy provides a strong conceptual framework that elegantly solves the issues of fragmentation and memory sharing.
- Significant Performance Gains: The empirical results are impressive. A 2-4x improvement in throughput over highly optimized, state-of-the-art serving systems is a substantial contribution that has immediate practical implications for reducing the cost and improving the scalability of LLM services.
- Practical Impact and Open Source: The authors have made vLLM publicly available and it has already been widely adopted by the community.

Peer Review

Weaknesses:

- Performance Overhead of PagedAttention: The paper acknowledges a 20-26% increase in latency for the attention kernel itself due to the overhead of managing non-contiguous memory blocks (e.g., block table lookups). While the end-to-end throughput benefits clearly outweigh this kernel-level overhead, a more in-depth analysis of this trade-off would be beneficial. For instance, are there scenarios (e.g., very short sequences, compute-bound configurations) where this overhead could negate the benefits of better memory management?
- Limited Exploration of Scheduling Policies: The paper states that vLLM uses a simple First-Come-First-Serve (FCFS) scheduling policy. While FCFS is fair, it is often suboptimal for throughput and latency in serving systems. Given the dynamic nature of LLM requests (variable prompt lengths, unknown output lengths), it would be valuable to discuss how PagedAttention could enable more sophisticated scheduling policies. For example, could the system prioritize shorter requests or those that can share KV cache blocks with already running requests?

Peer Review

Questions:
- What is the biggest difference between PagedAttention and paged memory in operating systems?
- How does PagedAttention interact with other memory-saving techniques like quantization or sparsity? Can these methods be combined, and what would be the expected interplay between them? For example, would smaller quantized KV cache entries favor smaller block sizes?
- The copy-on-write mechanism is triggered at the block level. Have you analyzed the performance implications of the block size on the frequency and cost of these copy operations, especially in highly dynamic scenarios like beam search? Is there an adaptive block size strategy that could optimize this?

Peer Review

Limitations:
Yes, the paper appropriately discuss limitations and social impact.

Others:
Ethics Flag: No
Ethics Review Area: I don’t know
Soundness: 3 good
Presentation: 3 good
Contribution: 4 excellent
Rating: 8: Strong Accept: Technically strong paper, with novel ideas, excellent impact on at least one area, or high-to-excellent impact on multiple areas, with excellent evaluation, resources, and reproducibility, and no unaddressed ethical considerations.
Confidence: 4: You are confident in your assessment, but not absolutely certain. It is unlikely, but not impossible, that you did not understand some parts of the submission or that you are unfamiliar with some pieces of related work.