STORAGE

Storage as a Performance Tier for LLM Serving



Joohee Lee *Research Engineer, SK hynix*

Donguk Moon *Director, SK hynix*

OCP PLATINUM Member

Prompt Caching: Saving Computation

- Caches KV states of shared prompt prefixes (e.g., system instructions, context, examples), to avoid redundant prefill computation.
- Reuses cached KV across requests and conversation turns. → Lower GPU compute & TTFT

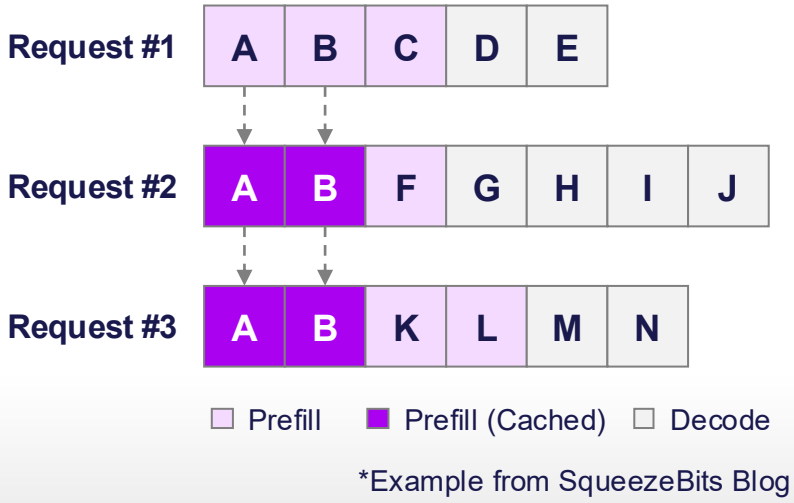
Prompt Caching Example*

Request #1

A chat between a curious user and an artificial intelligence assistant. The assistant gives helpful, detailed, and polite answers to the user's questions.
User: Hello!

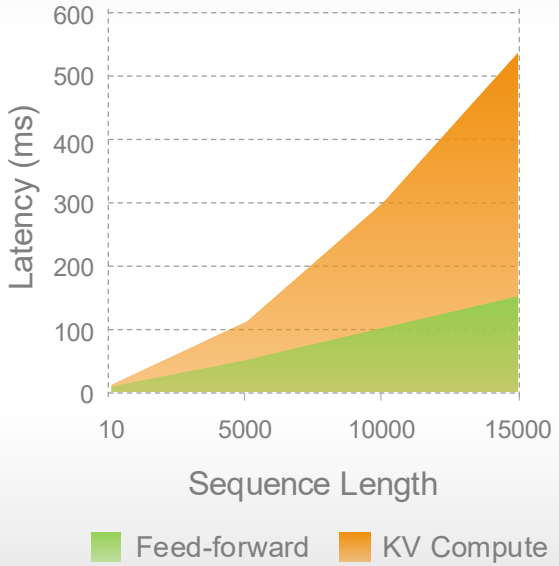
Request #2

A chat between a curious user and an artificial intelligence assistant.
The assistant gives helpful, detailed, and polite answers to the user's questions.
User: How are you?



*Example from SqueezeBits Blog

KV Compute Overhead

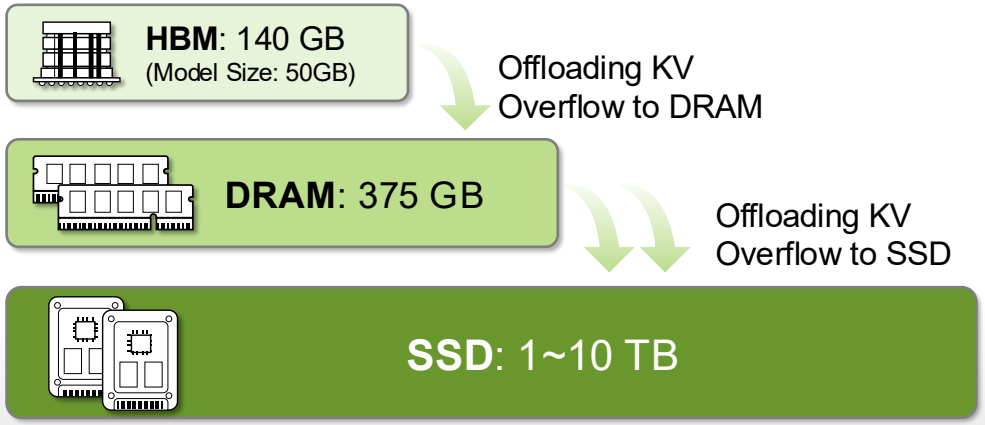


The Growing KV Cache Working Set

- Production LLM services retain cached prefixes to maximize reuse.
  - Anthropic provides 5-minute and 1-hour cache Time-to-Live (TTL) options.
  - OpenAI supports extended prompt caching with a TTL of up to 24 hours.
  - Longer TTL → More reusable KV states → Larger KV working set.

Hierarchical Prompt Cache

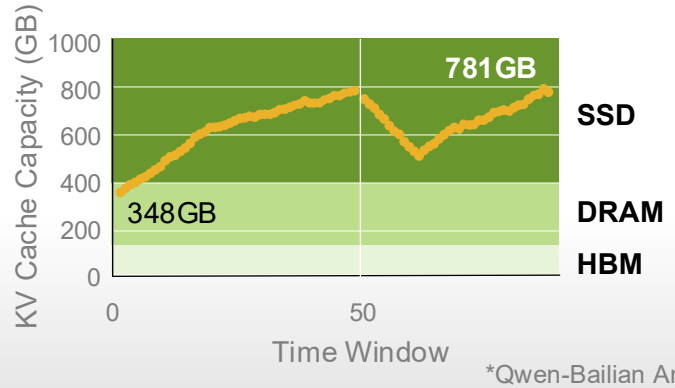
- KV caches are primarily stored in GPU HBM and host DRAM.
- Excess KV caches are offloaded to SSDs.



*NVIDIA B200 Server

Working Set*

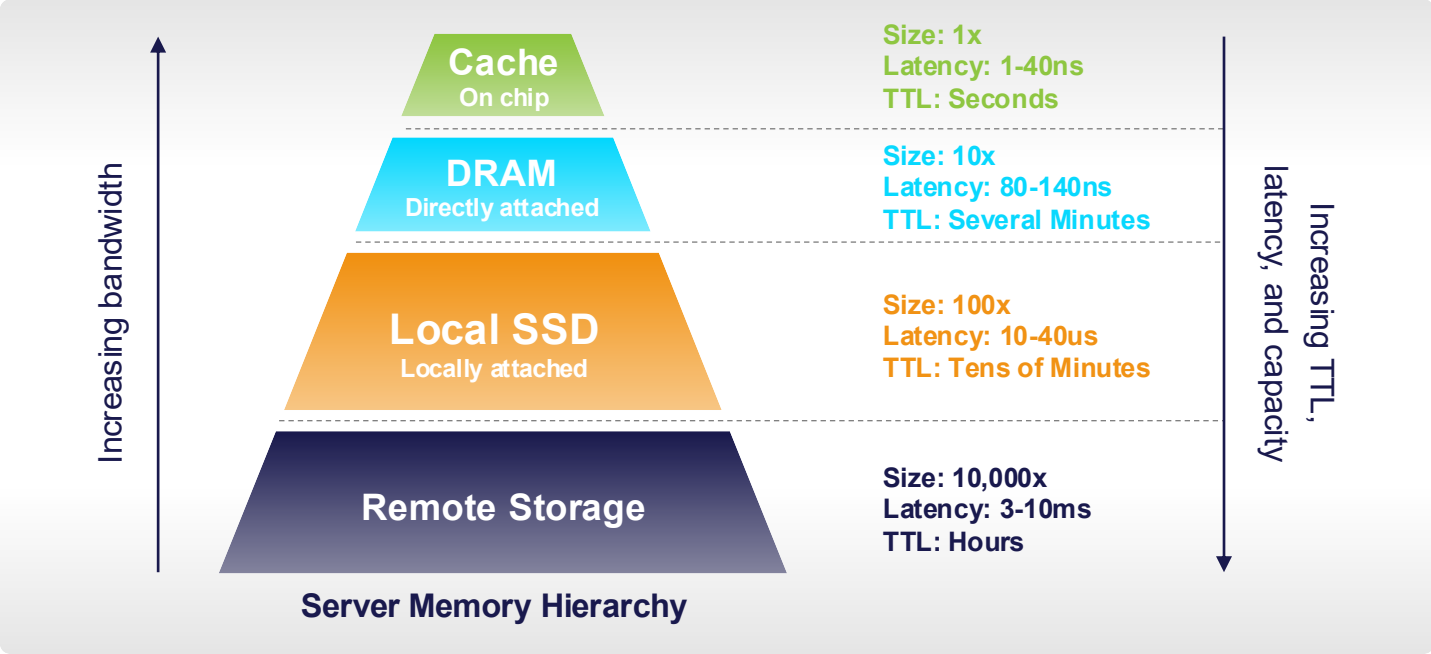
- HBM/DRAM capacity is insufficient for 10-minute KV cache retention in chat workloads.
- For Qwen3-14B (BF16), a 10-minute TTL requires up to ~781GB of KV cache capacity per GPU.



*Qwen-Bailian Anonymous Dataset

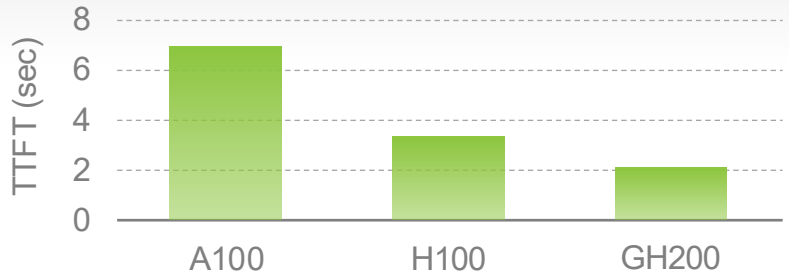
Scaling Prompt Caching Beyond Memory

- Longer cache retention improves reuse opportunities, but rapidly expands the KV working set beyond HBM/DRAM capacity.
- Production LLM services are moving KV caches to storage.
  - OpenAI: GPU + CPU RAM + Storage
  - DeepSeek: GPU + Storage
  - Google Cloud: GPU + CPU RAM + Storage

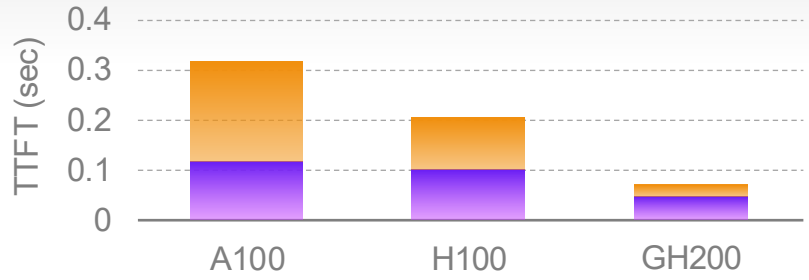


Prefill Performance

- Without prompt cache reuse, prefill is GPU compute-bound due to full KV re-computation. Thus, TTFT scales almost linearly with GPU compute performance.
- With prompt cache reuse, prefill retrieves KV data from outside the GPU and decodes only one token, making TTFT dependent on link bandwidth (for KV transfer) and GPU memory bandwidth (for decoding). Thus, improving these bandwidths yields nearly inversely proportional TTFT gains.



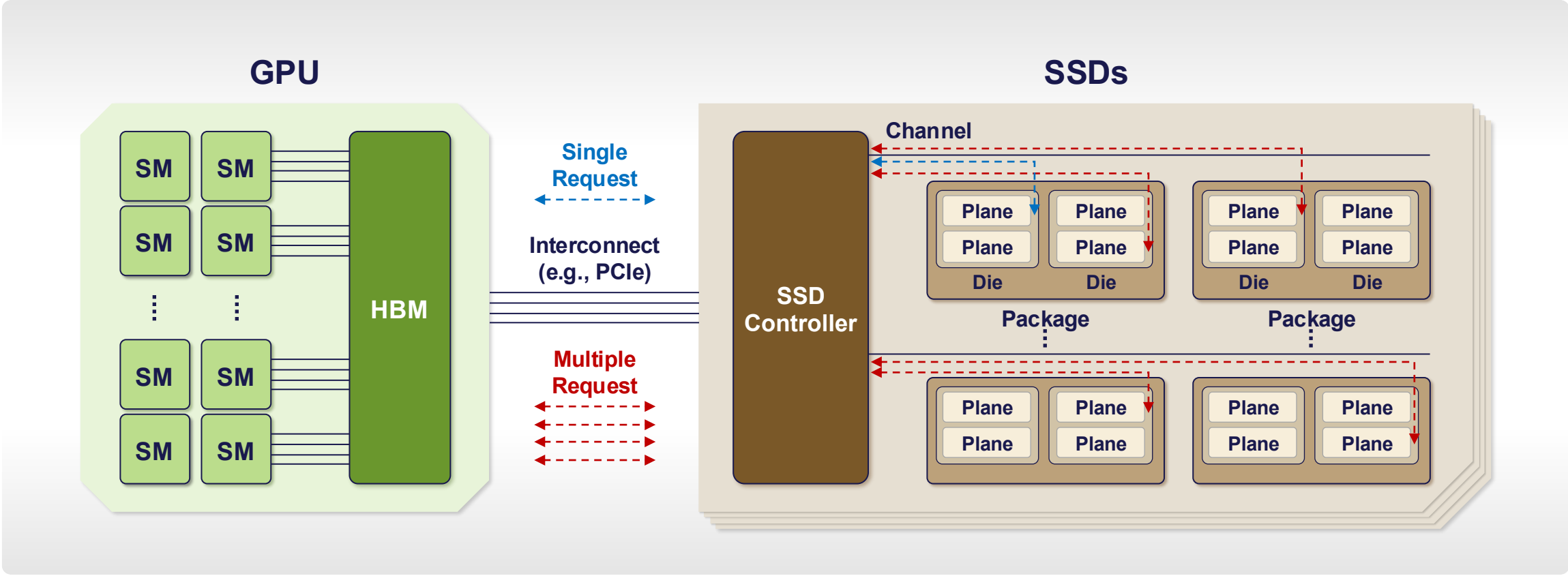
GPUs	A100	H100	GH200
FP16 (TFLOPS)	624	1,513	1,979



GPUs	A100	H100	GH200
Link BW (GB/s)	32	64	450
Mem BW (TB/s)	1.9	2	4

SSD Parallelism

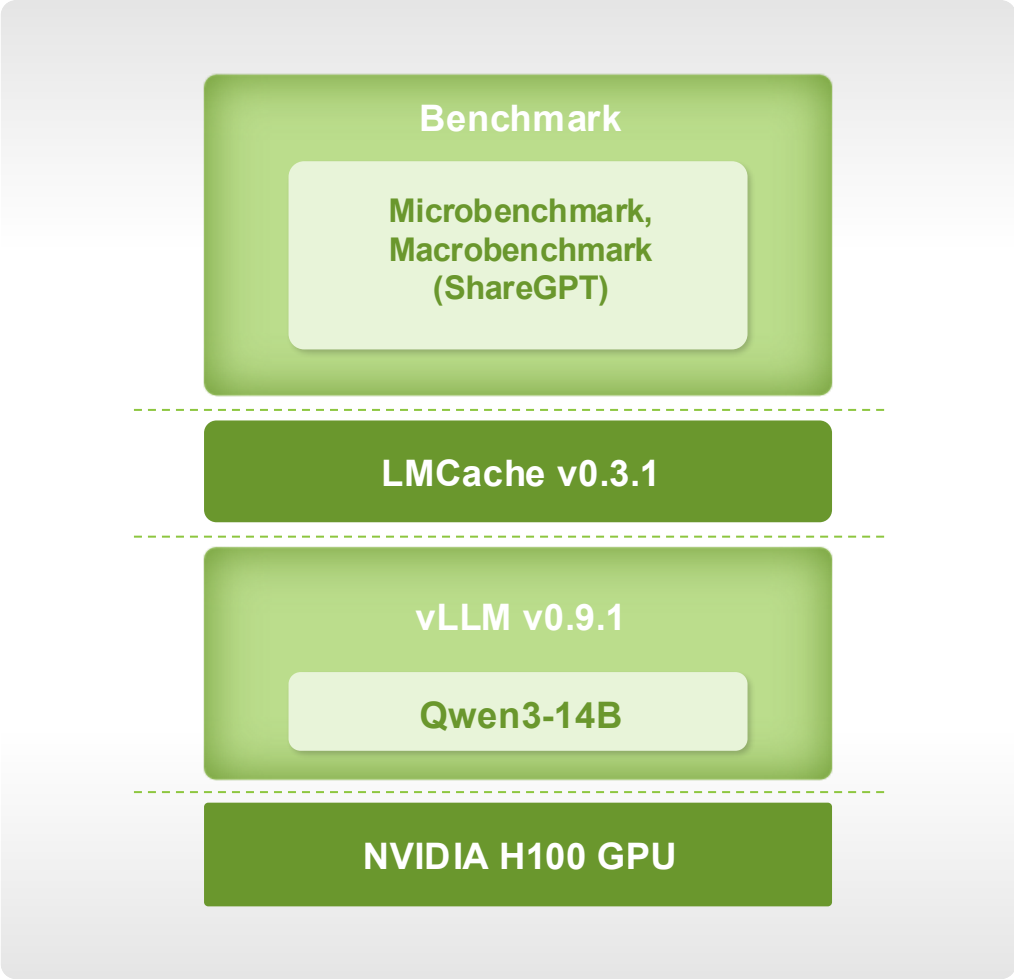
- When using SSDs, achieving peak performance requires exploiting their internal parallelism – across channels, packages, dies, and planes – so that KV traffic can be served with maximal bandwidth.



Evaluation Setup

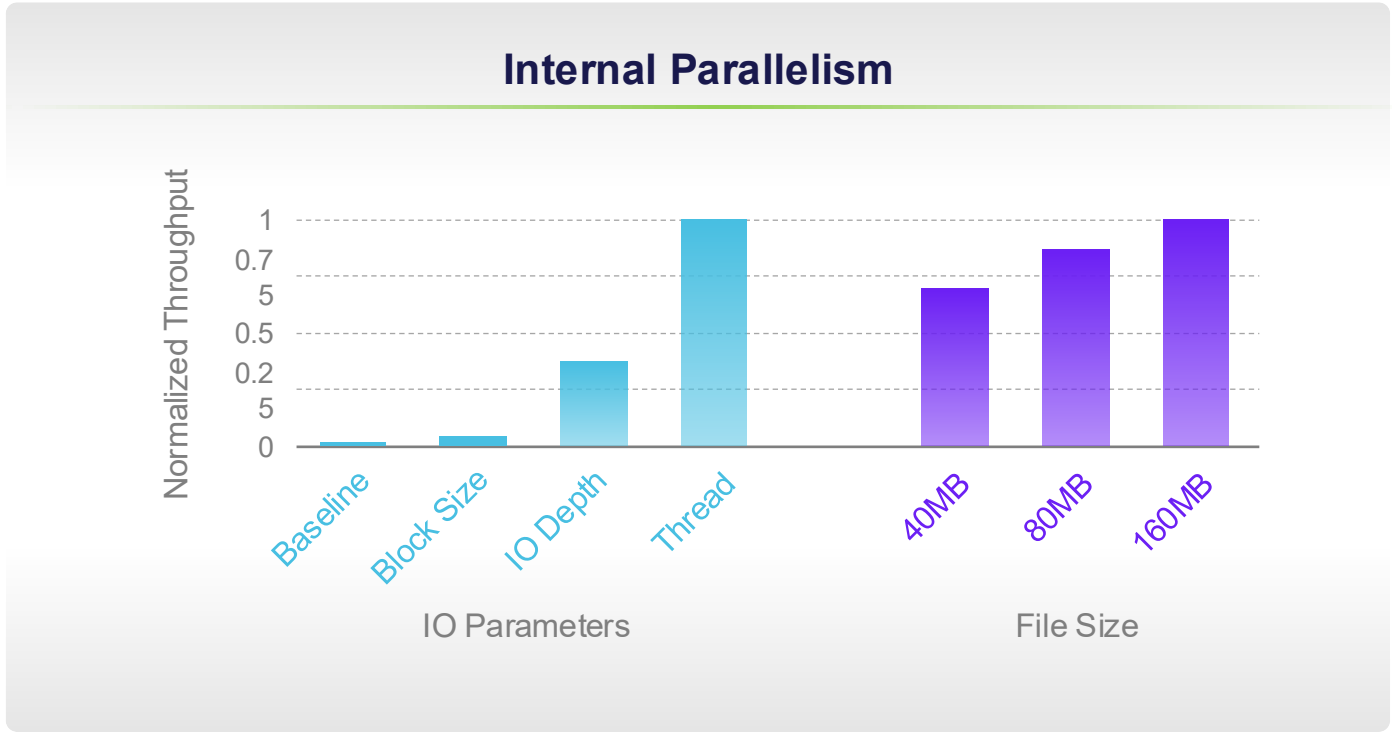
- **System Configuration**
  - Serving Framework: vLLM v0.9.1 + LMCache v0.3.1
  - Model: Qwen3-14B (BF16)
  - GPU: NVIDIA H100 PCIe (PCIe Gen5 x16)
  - Storage: NVMe SSD (PCIe Gen5)
  - SSD I/O Optimization: Linux io_uring-based asynchronous, multi-threaded I/O

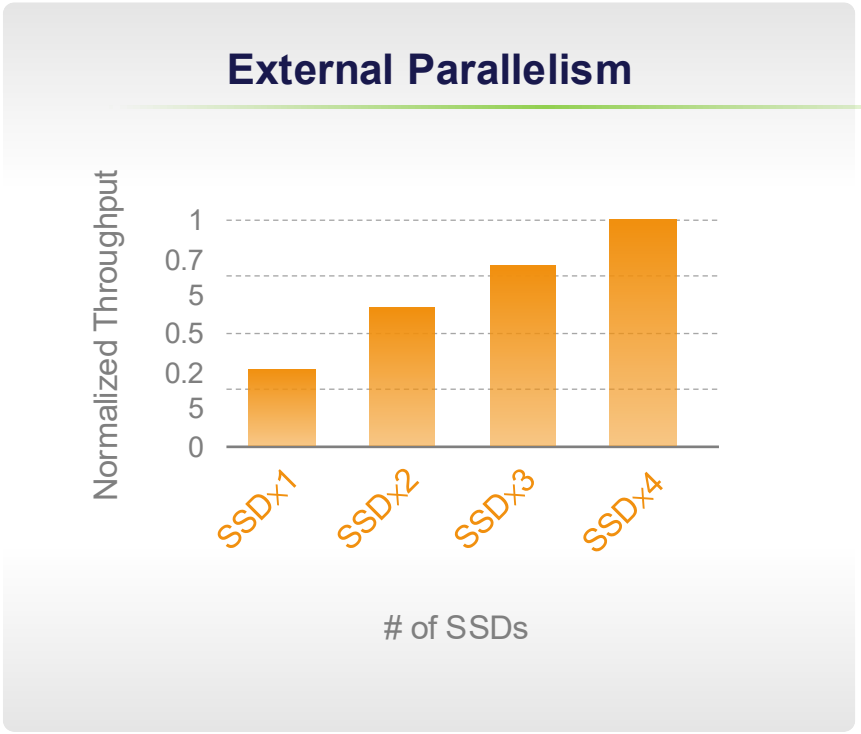
- **Benchmark Configuration**
  - **Microbenchmark**
  - Synthetic prompt cache traffic with 100% cache hit
  - Isolates the impact of SSD internal & external parallelism
  - Measures the best achievable TTFT for cached prefill
  - **Macrobenchmark**
  - ShareGPT-based realistic multi-turn workload
  - Request rate (QPS) sweep to evaluate serving performance under varying loads
  - Equal cache capacity for DRAM and SSD



SSD Performance

- **[Internal Parallelism]** Key I/O parameters that impact SSD performance include request size (block size), I/O depth, and the number of issuing threads.
- **[External Parallelism]** GPUs operate over a PCIe x16 interface while each SSD occupies a PCIe x4 link.

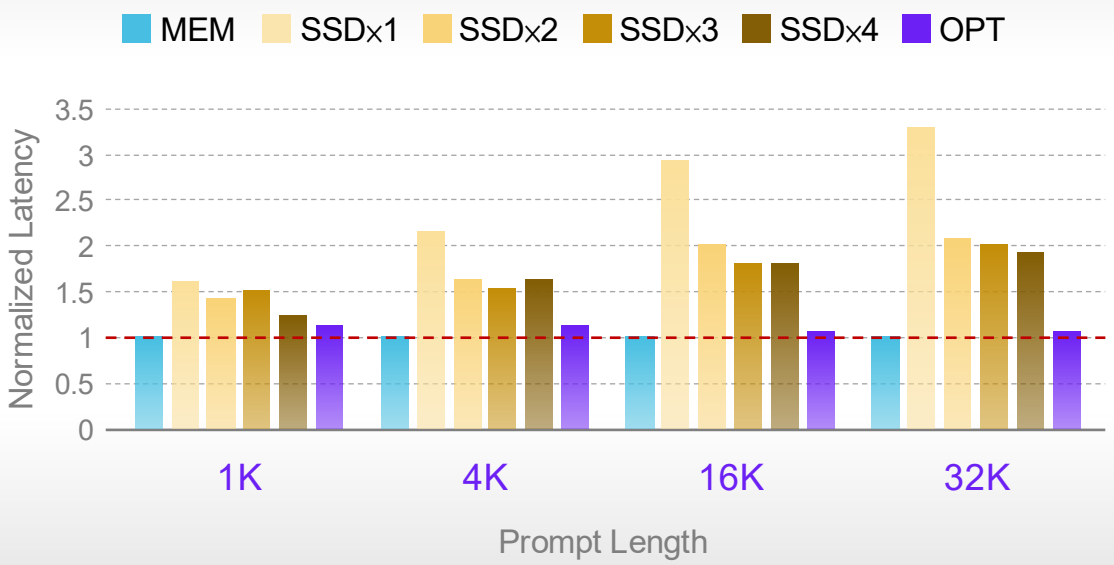




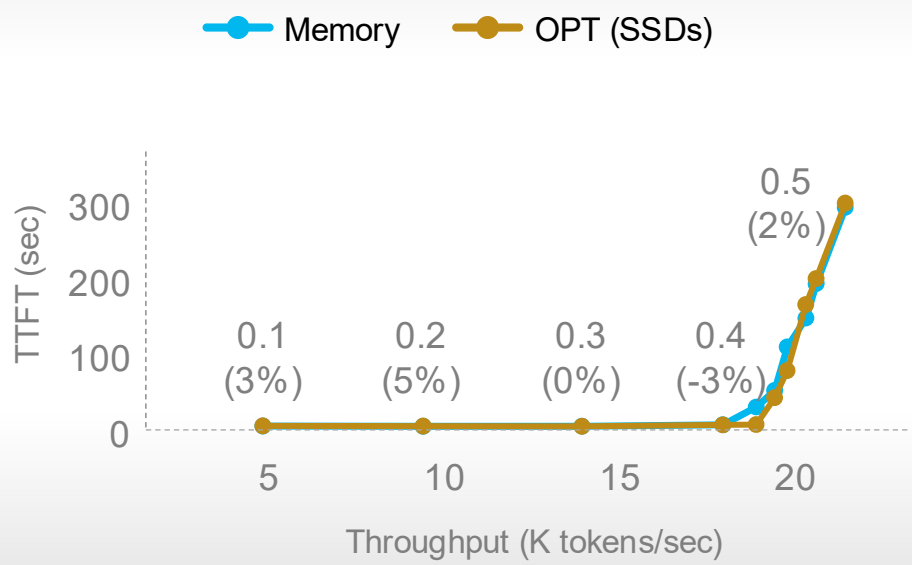
Prompt Caching Performance

- Without any additional optimizations, scaling from 1 → 4 SSDs (MD4) resulted in 1.26x~1.92x higher TTFT than the host-memory baseline.
- With LMCache chunk size tuning and io_uring optimizations applied, TTFT approached the host-memory baseline, reaching 1.07x~1.14x of the memory performance.

Microbenchmark



Macrobenchmark (ShareGPT)



SSD Endurance Considerations

- We define the endurance-bound capacity as the minimum SSD capacity required to sustain the expected daily write amount under given DWPD (Daily Writes Per Day) constraint.

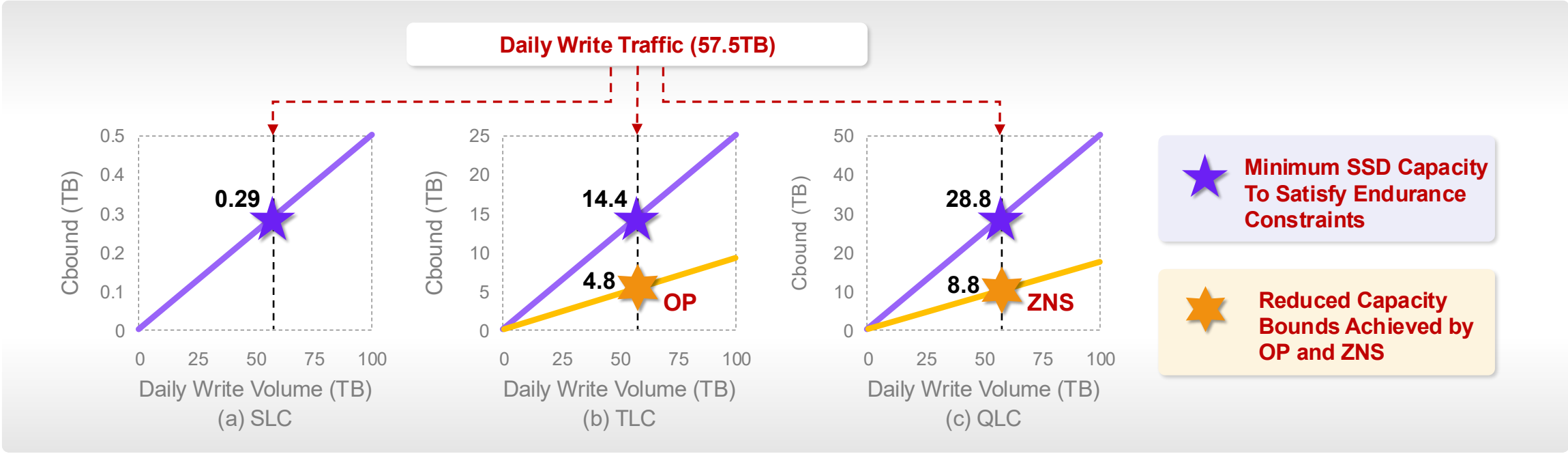
$$C_{bound} = \frac{total\ writes\ per\ day}{the\ number\ of\ SSDs \times DWPD}$$

$$TBW\,(Terabytes\ Written) = \frac{P/E\ Cycles \times Capacity}{WAF(Write\ Amplification\ Factor)}$$

$$DWPD = \frac{TBW}{Capacity \times Warranty\ Days}$$

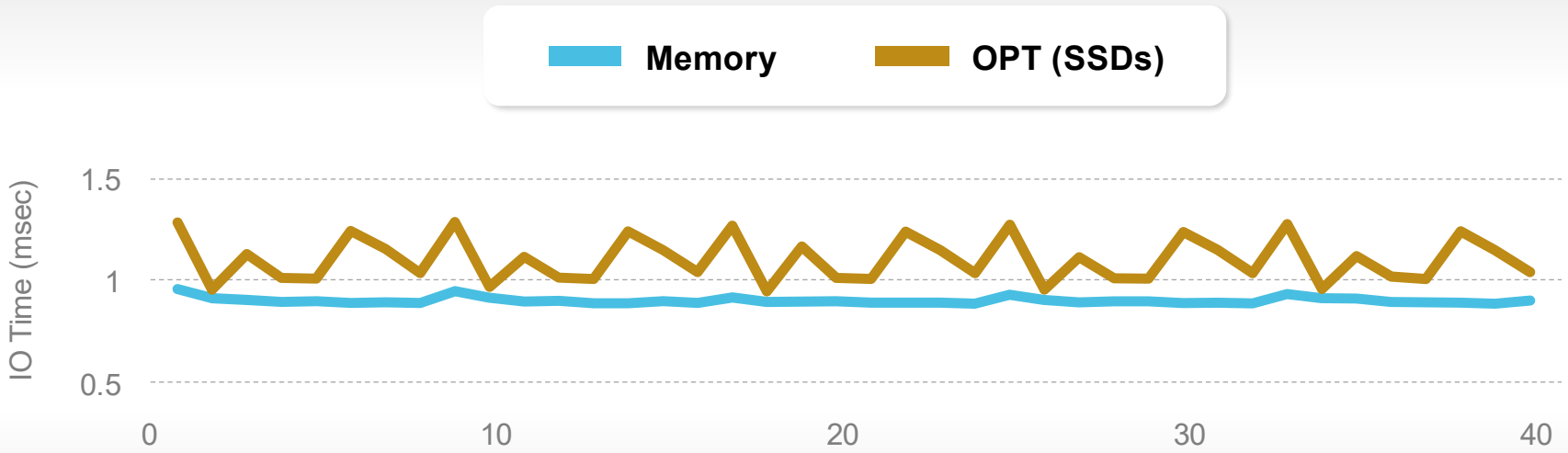
SSD Endurance Considerations

- SSD-backed prompt caching generates heavy write traffic from continuous KV cache writes.
- With Qwen3-14B, write traffic can reach up to ~57.5TB/day per GPU.
- Some enterprise SSDs cannot meet the required Drive Writes Per Day.
- Endurance-aware SSD selection and design are essential for sustainable deployment.



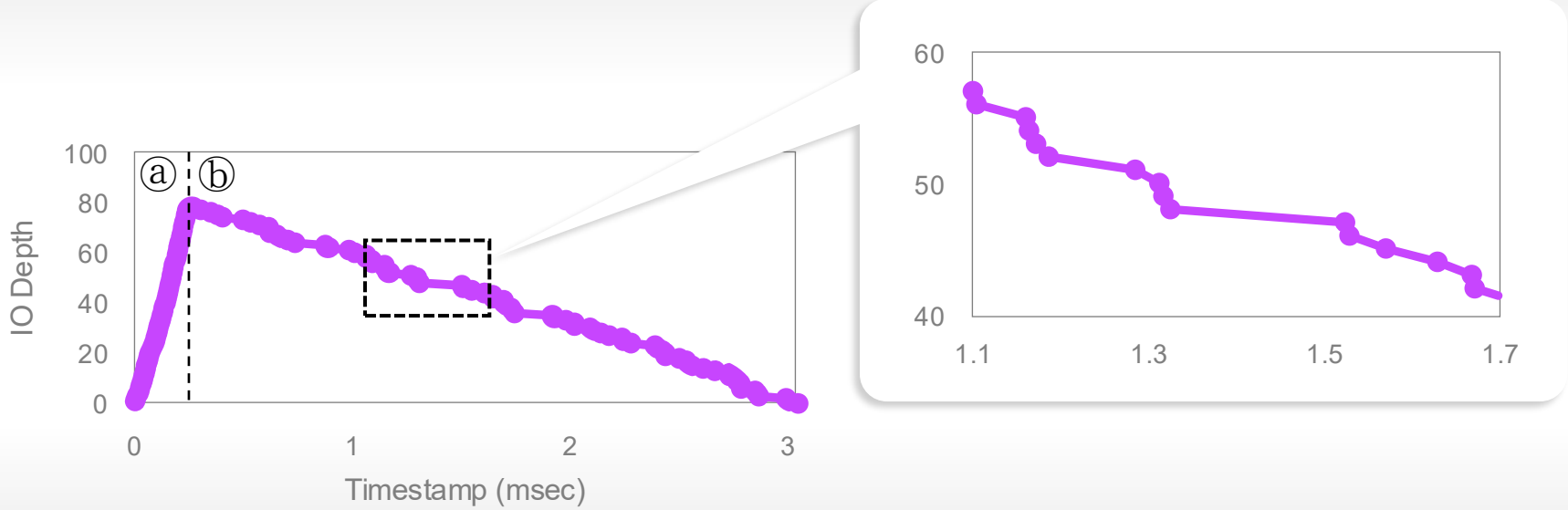
Remaining Challenge: Non-uniform SSD I/O Latency

- Even after SSD-aware optimization, per-chunk I/O latency remains variable.
  - DRAM retrieval shows consistent per-chunk latency.
  - SSD retrieval exhibits latency fluctuations even for same-sized KV chunks.
  - Variability remains despite fully optimized SSD I/O



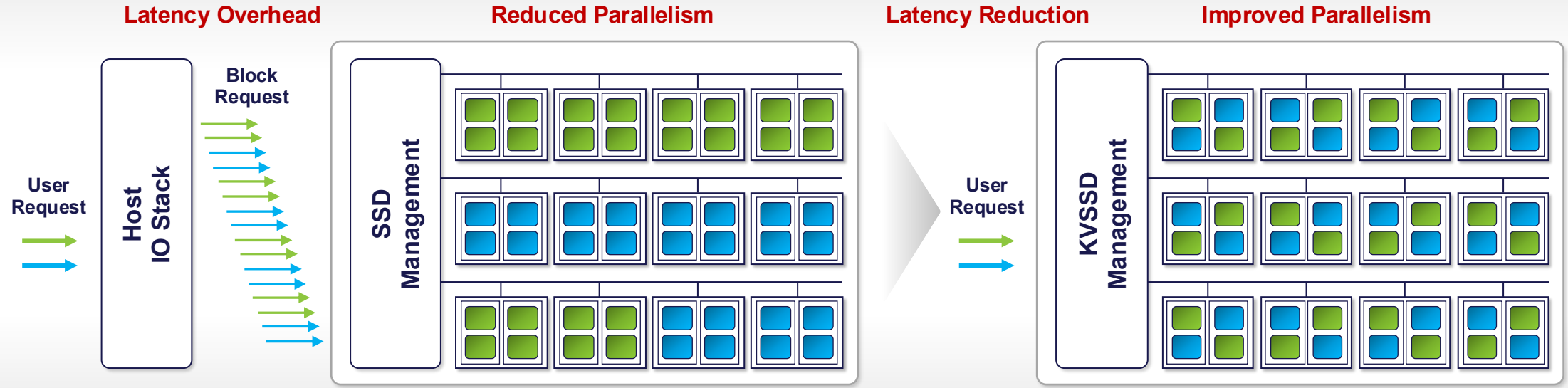
Remaining Challenge: Irregular I/O Completion

- The SSD I/O pipeline exhibit initial queue buildup and irregular request completions.
  - ⓐ Initial Queue Buildup: I/O requests accumulate before the first completions arrive.
  - ⓑ Irregular Request Completion: Queue drains at non-uniform intervals, even for same-sized requests.



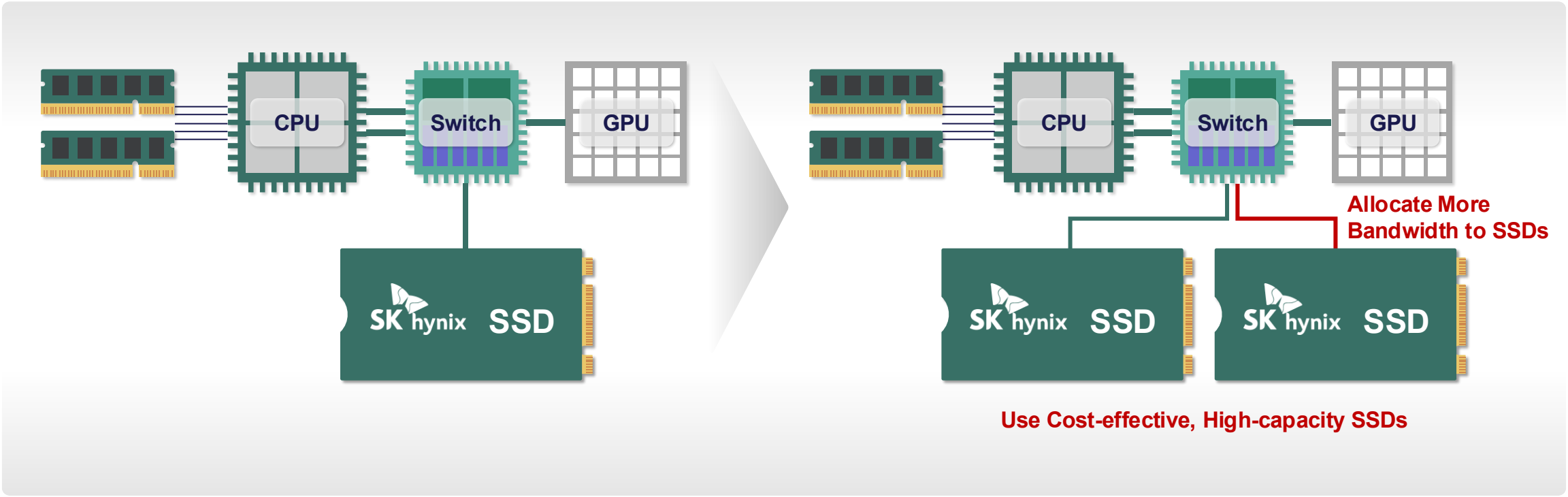
Toward KV-aware Storage

- Block-oriented operation and software I/O processing can introduce service-time variability.
- KV-aware storage exposes key-value semantics to storage (e.g., KV SSDs), avoiding conventional block-address translation and I/O processing.
- This can reduce retrieval overhead and improve service-time consistency.



Toward Higher Storage Bandwidth

- Limited system-level PCIe allocation to NVMe storage can constrain SSD bandwidth below the GPU interconnect capacity.
- Scaling aggregate storage bandwidth requires sufficient PCIe resources, denser fabrics, and faster interfaces to keep pace with future GPUs.



Key Takeaways

- Prompt caching shifts LLM prefill from compute-bound to data movement bound, making efficient KV transfer critical to TTFT.
- SSD internal and external parallelism must be fully exploited to maximize effective SSD-to-GPU bandwidth.
- With SSD-aware I/O optimization, SSD-backed prompt caching can achieve near-DRAM-class TTFT, even under realistic multi-turn workloads.
- SSD endurance is a first-class deployment consideration; OP, ZNS, and FDP can substantially improve write endurance and reduce capacity requirements.
- Remaining I/O variability reveals further optimization opportunities through tighter HW/SW co-design and KV-aware storage (e.g., KV SSDs).
- SSD is not just a capacity tier – it can become a high-performance KV cache tier for scalable LLM serving.

Call to Action

- Problem to solve
  - Design AI serving systems with SSDs as a performance-critical cache tier, not just passive capacity.
  - Provision sufficient system-level I/O bandwidth for storage aggregate SSD bandwidth aligned with rapidly growing GPU interconnect bandwidth.
  - Explore KV-aware storage interfaces (e.g., SNIA Key Value Storage API) and HW/SW co-design for low-overhead, more predictable KV retrieval.
- Potential OCP Collaboration
  - Evaluate storage technologies and interfaces such as FDP, ZNS, and KV SSDs for LLM KV cache workloads.
- Feel free to contact us for any questions, discussion or collaboration
  - Project Wiki: https://github.com/skhynix/LMCache/wiki/Storage-as-a-Performance-Tier-for-LLM-Serving
  - E-mail: joohee.lee@sk.com, donguk.moon@sk.com

Thank You!

