SDC | StorageAI™

A SNIA Event

April 29, 2026 • Denver, Colorado

Scaling Inference with KV Cache Storage Offload and RDMA-Accelerated Architecture

Ugur Kaynar
Dell Technologies
Distinguished Engineer– Office of Storage CTO

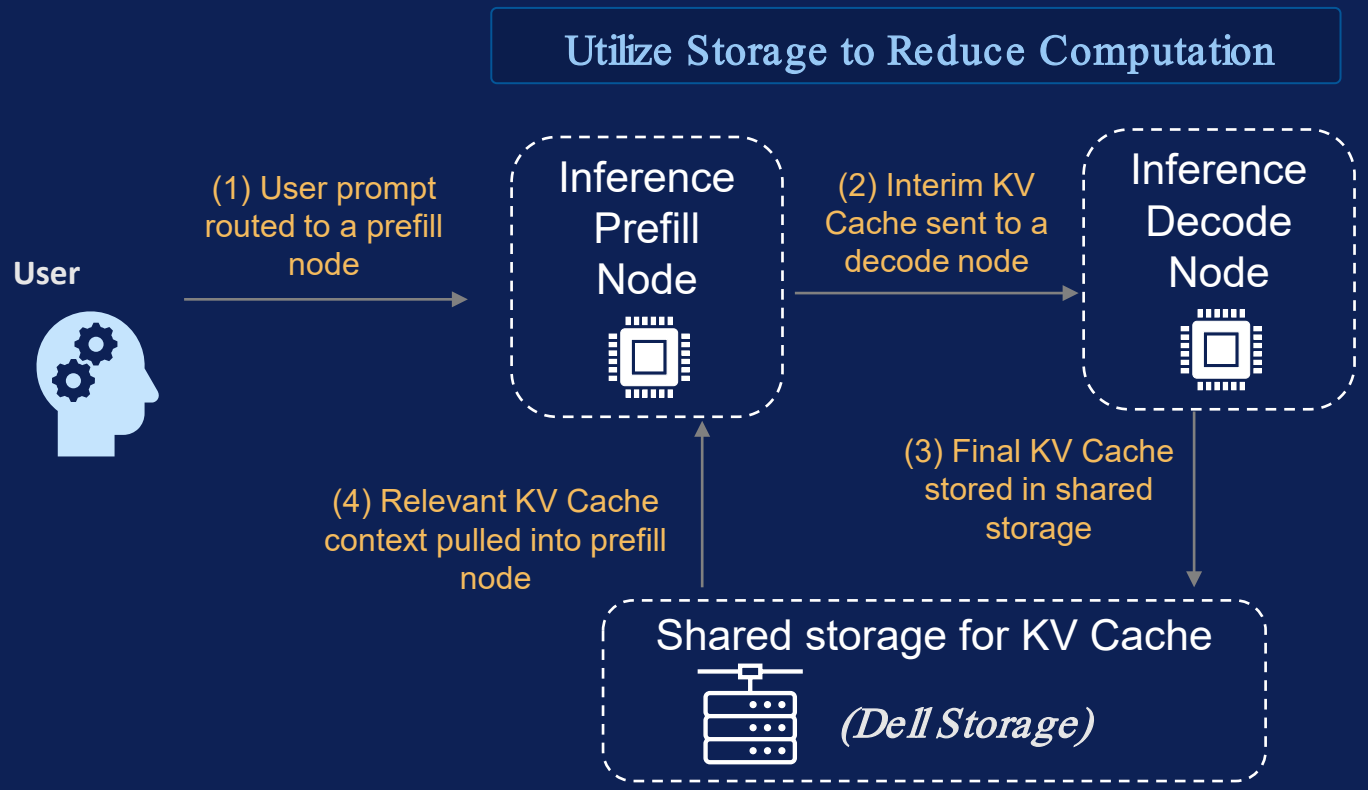
KV Cache Accelerates AI Inferencing

What is KV Cache?

- KV = Key-Value tensors used by transformer models to process token semantics.
- Acts like a context memory avoids recomputing attention for previous tokens.
- KV Cache size scales linearly with longer inputs, larger models, and more users.

Benefits of KV Cache Shared Storage

- Reduces GPU compute cost
- Scalable and flexible deployment
- Faster response time and longer contexts
- Higher cache hit rates
- Reuse of KV cache across requests.



- Prefill Phase: Compute-bound task where input tokens (prompt and context) are processed to build KV cache.
- Decode Phase: Memory-bound task where new tokens are generated one at a time using KV cache.
- Storage throughput and latency requirements increase with faster GPUs.

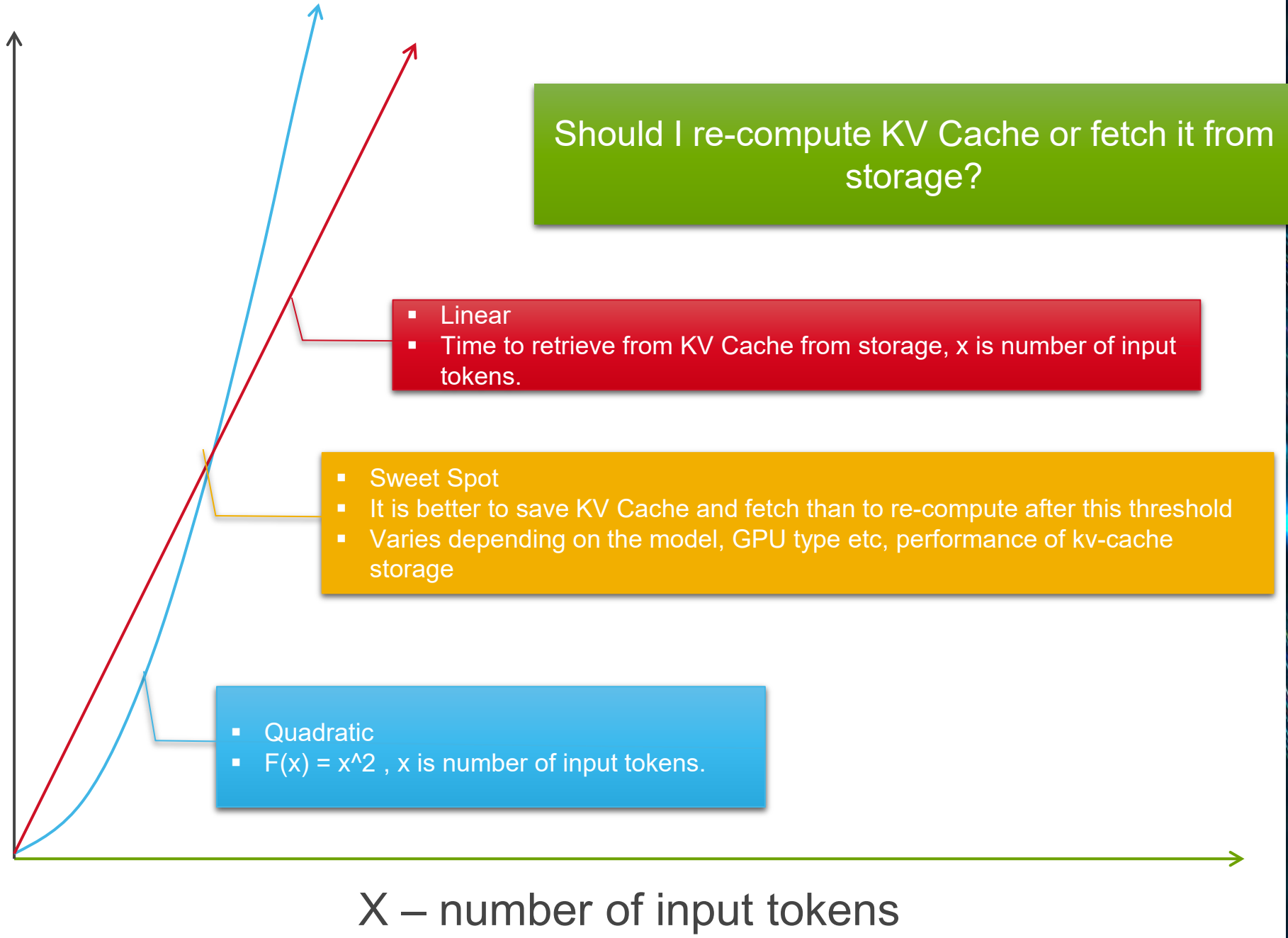
How Big Is the KV Cache, Really?

- In high-concurrency environments, the aggregate KV Cache footprint across users and sessions can easily reach **terabytes**, far exceeding the capacity of GPU memory. This is where storage offload becomes critical not just for high performance, but for reasonable query response times.

Model (FP16)	KV Size per Token	Short Sequence (4K tokens)	Long Sequence (128K tokens)	1M Active Sessions
Llama3-8B	128 KB	500 MB	15.6 GB	15.6 PB
Llama3-70B	327 KB	1,282 MB	40 GB	40 PB
Llama3-405B	516 KB	2,114 MB	62 GB	62 PB
GPT-OSS-20B	48 KB	196 MB	6.2 GB	6.2 PB
GPT-OSS-120B	72 KB	295 MB	9.2 GB	9.2 PB
DeepSeek-R1-685B	70 KB	286 MB	9 GB	9 PB

Time to Compute KV-cache

Time to Fetch From Storage



Should I re-compute KV Cache or fetch it from storage?

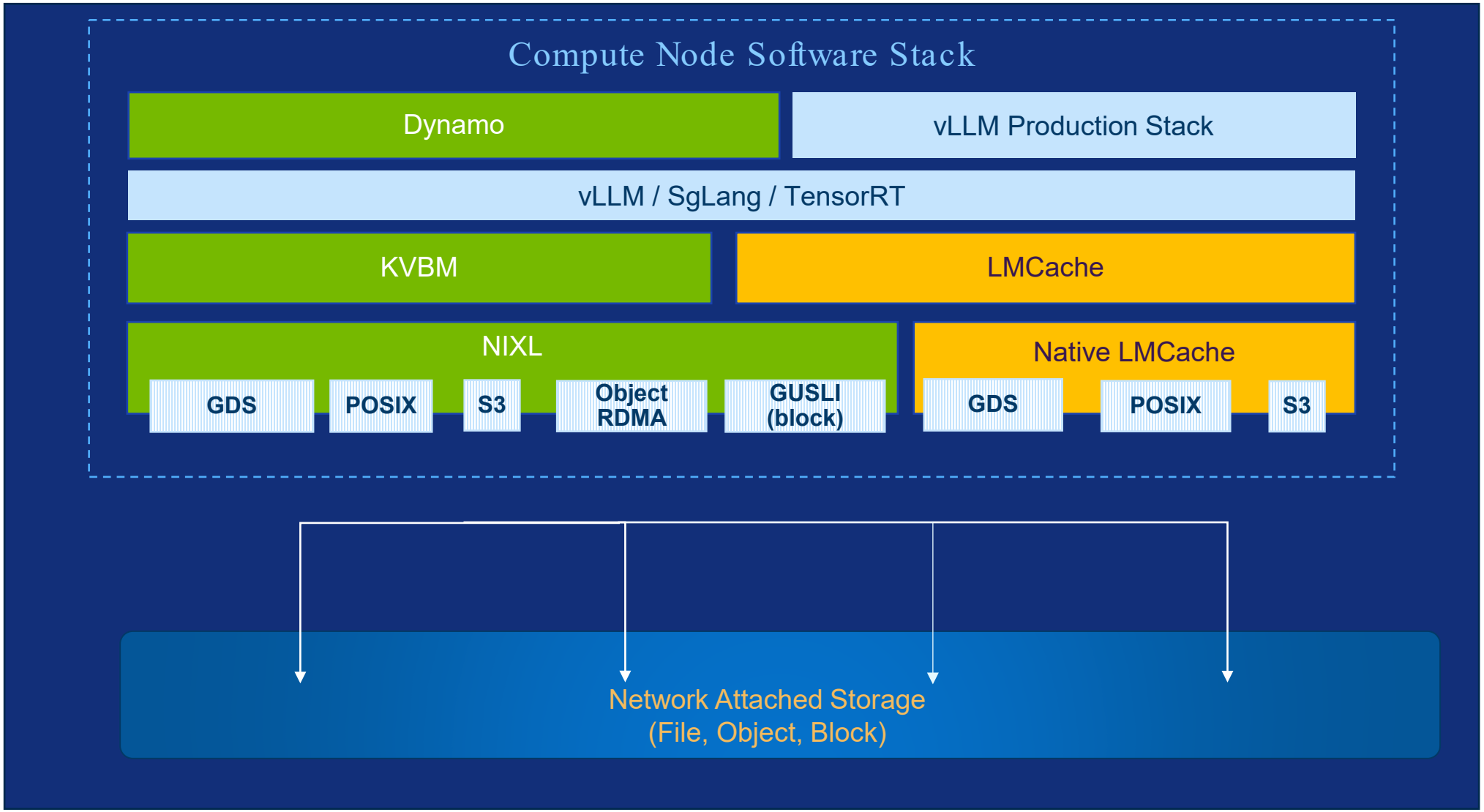
- Linear
- Time to retrieve from KV Cache from storage, x is number of input tokens.

- Sweet Spot
- It is better to save KV Cache and fetch than to re-compute after this threshold
- Varies depending on the model, GPU type etc, performance of kv-cache storage

- Quadratic
- F(x) = x^2 , x is number of input tokens.

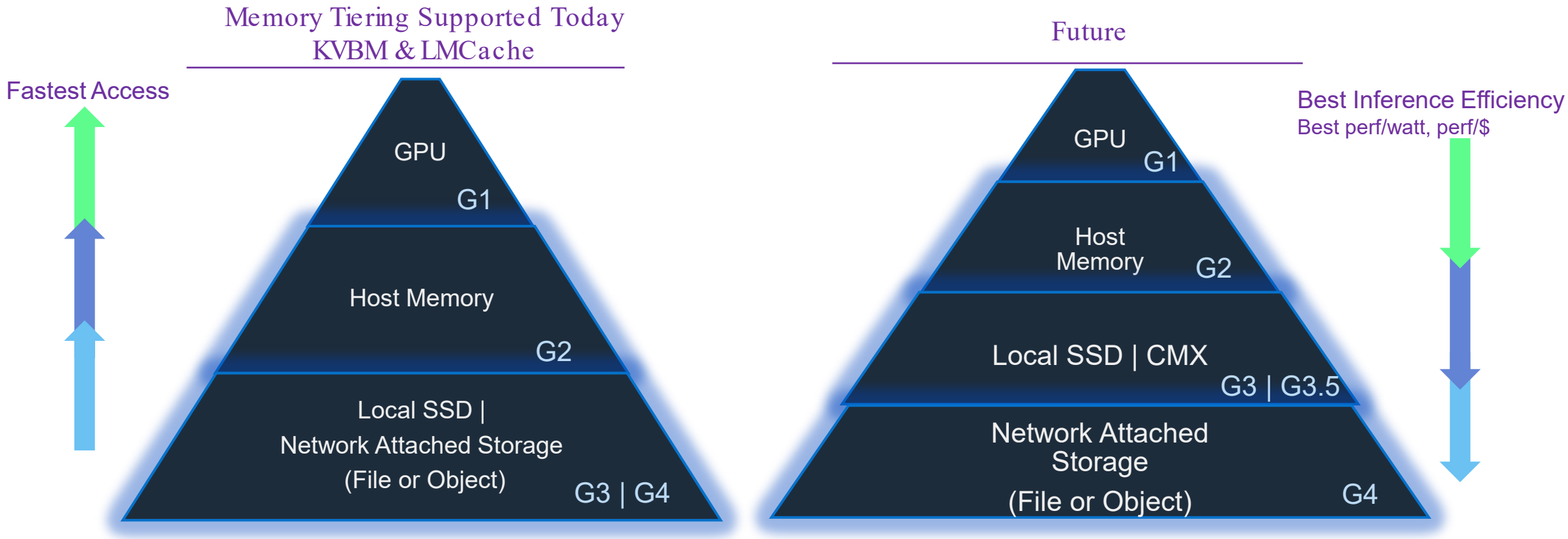
X – number of input tokens

AI Inference Storage Stack: From Orchestration to Storage



KV Cache Memory Tiers from GPU (G1) to Shared Storage (G4)

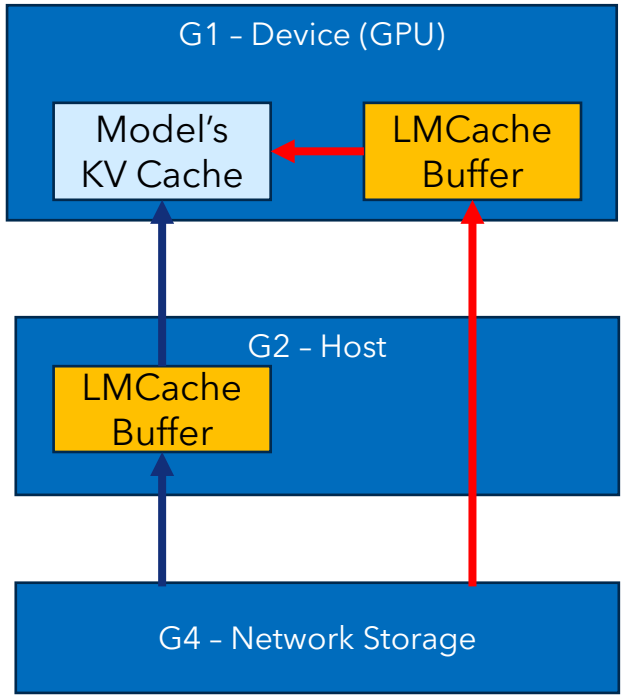
Distributed KV-Cache managers (LMCache, KVBM) offload infrequently accessed KV cache to more cost-efficient memory tiers.

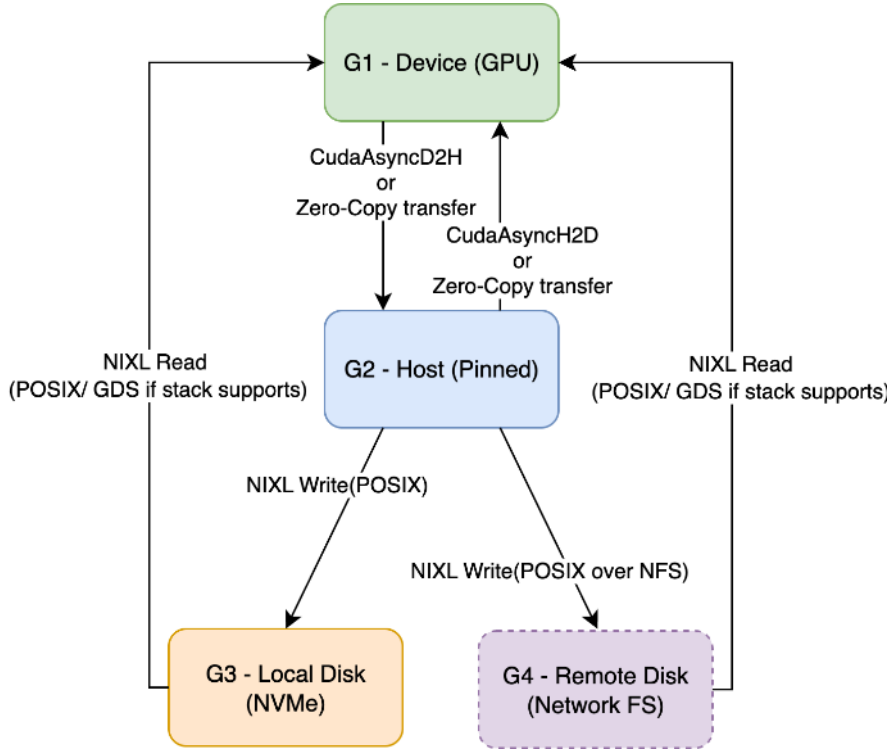


The inference engine directly manages GPU Memory (G1) with optional Host Memory (G2) offloading. LMCache and KV Block Manager (KVBM) manage G2 and lower tiers.

KV Cache IO Paths of LMCache and KVBM

RDMA READ
path for remote storage

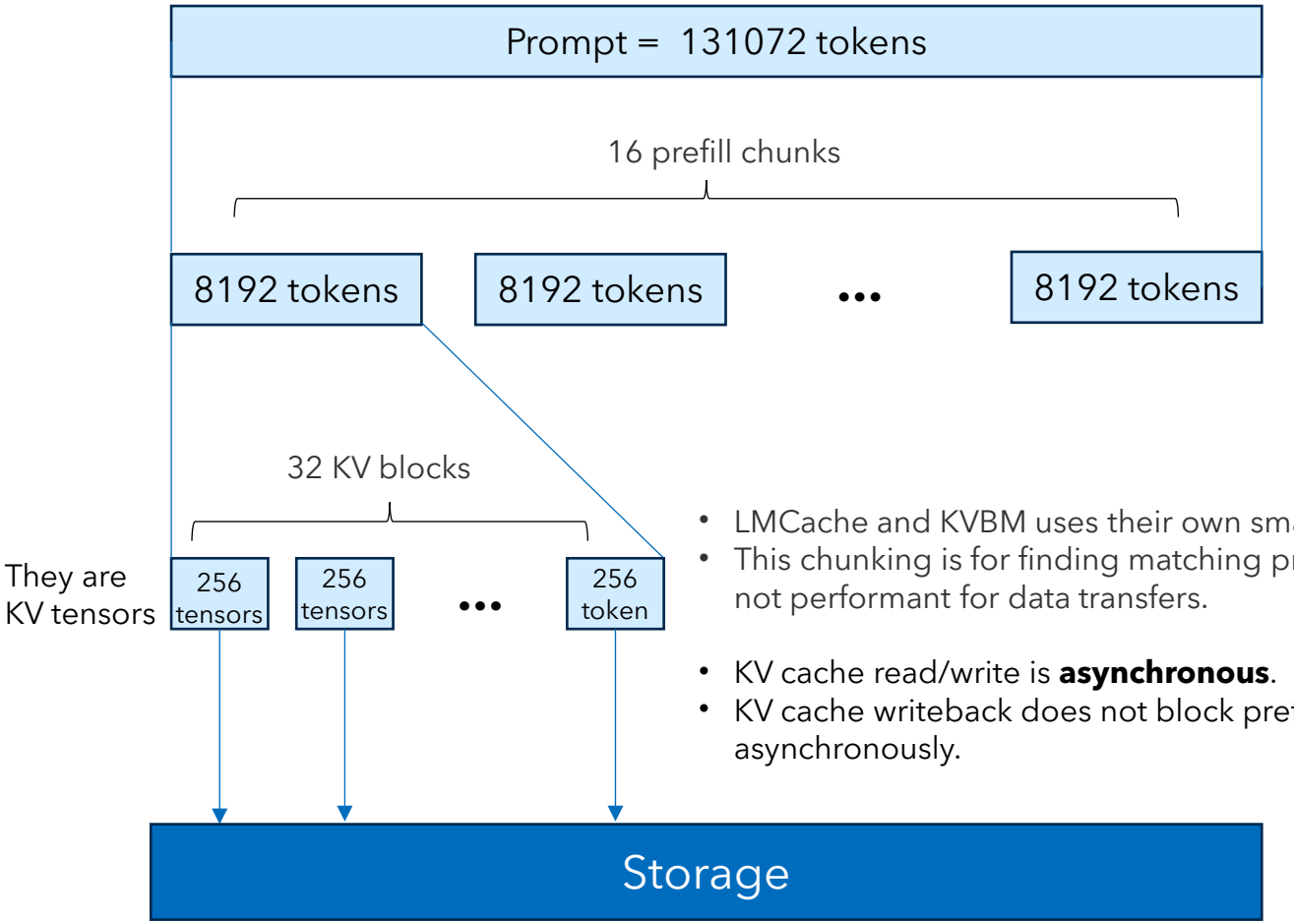




Reads: LMCache returns data directly from storage to the requester using GDS, **without populating the Host Memory cache.**
Writes: Tiered. Data is first written to the Host Memory and is then asynchronously persisted to storage backends.

KVBM recommends using host-cache when paired with remote storage.
IO Behavior: IO always go through a host cache. Direct storage-to-device access is **experimental**.

How the Inference Engine Processes Prompts and Persists KV Cache



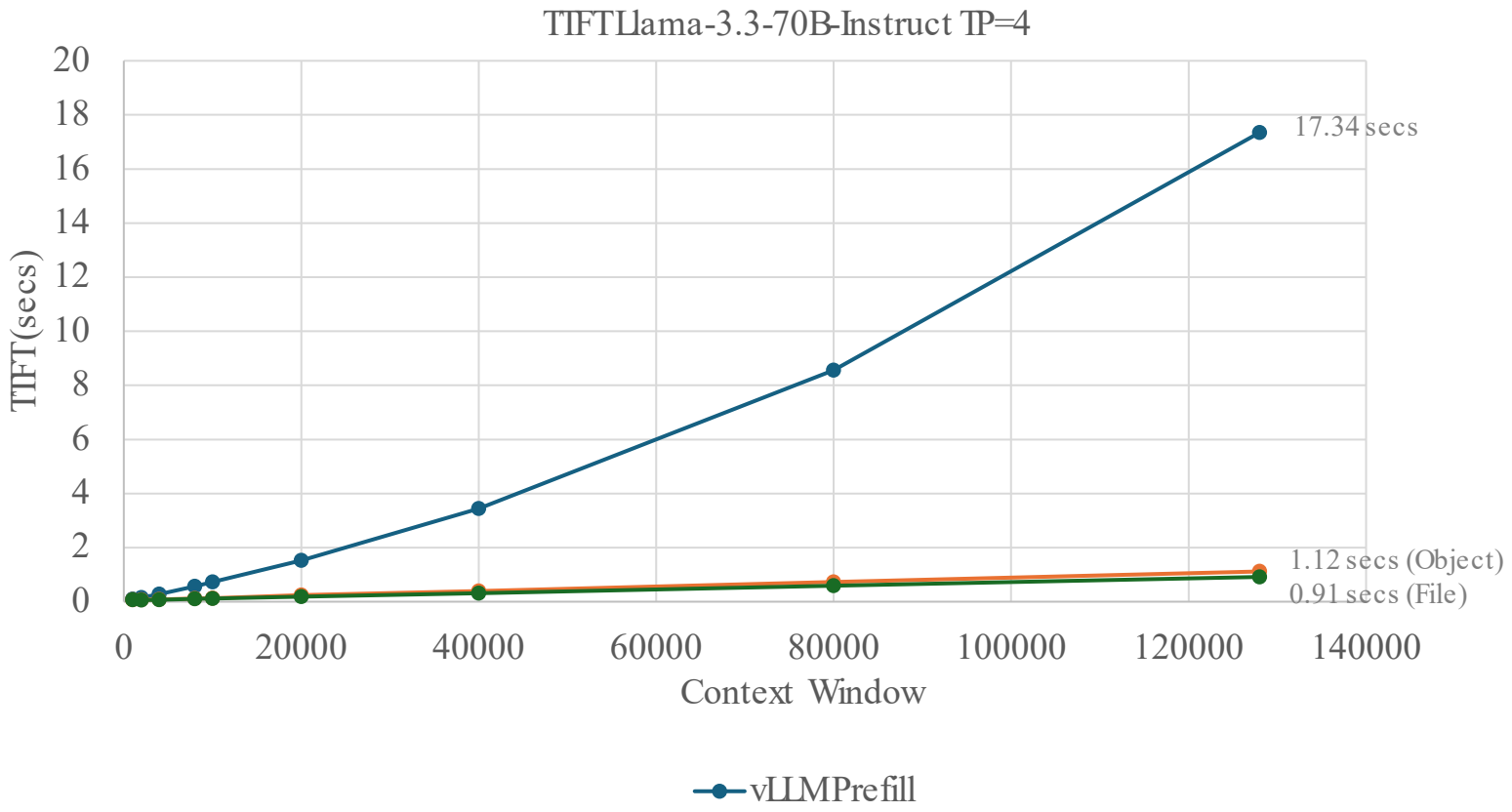
- Modern Inference engine processes large prompts through **chunked prefill,** in iterations (e.g., 8192 tokens per iteration).
- This is done to balance prefill and decode operation and size is configurable.

Prefill execution in the inference engine is **synchronous**

- LMCache and KVBM uses their own smaller chunk size (e.g. 256 tokens) for caching.
- This chunking is for finding matching prefixes as doing per token is expensive and not performant for data transfers.

- KV cache read/write is **asynchronous**.
- KV cache writeback does not block prefill execution; persistence can occur asynchronously.

Single Shot Benchmarking (TTFT)

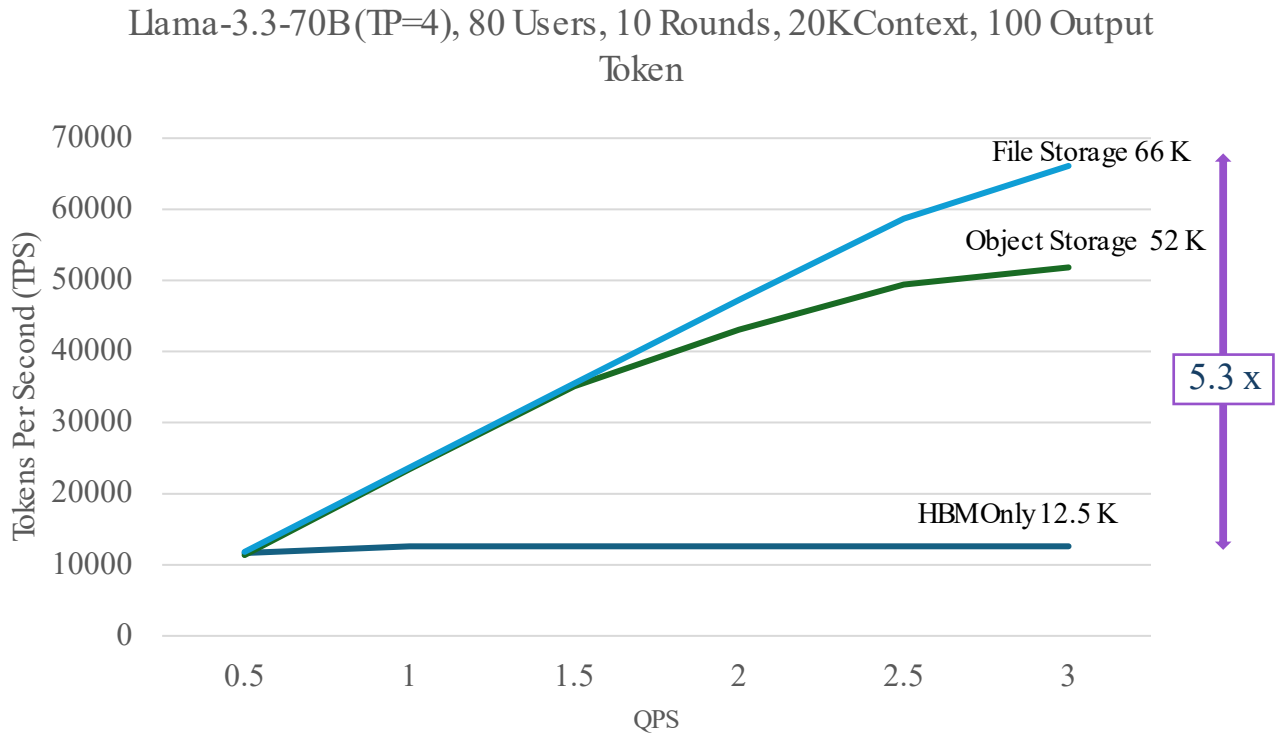


[1] From Bottleneck to Breakthrough: Scalable KV Cache Offloading with Dell AI Storage Engines | Dell Technologies Info Hub

Multi-Turn Benchmarking (Token Throughput)

Benchmark: LMBenchmark[1]

Concurrent Users: 80
Number of Rounds: 10 per user
System Prompt: 1,000 tokens (shared across sessions)
Chat History: 4,000 tokens per user (unique per session)
Response Length: 200 tokens per turn (unique per session)
QPS (Queries per Second): Varied from 1 to 7 to simulate concurrency



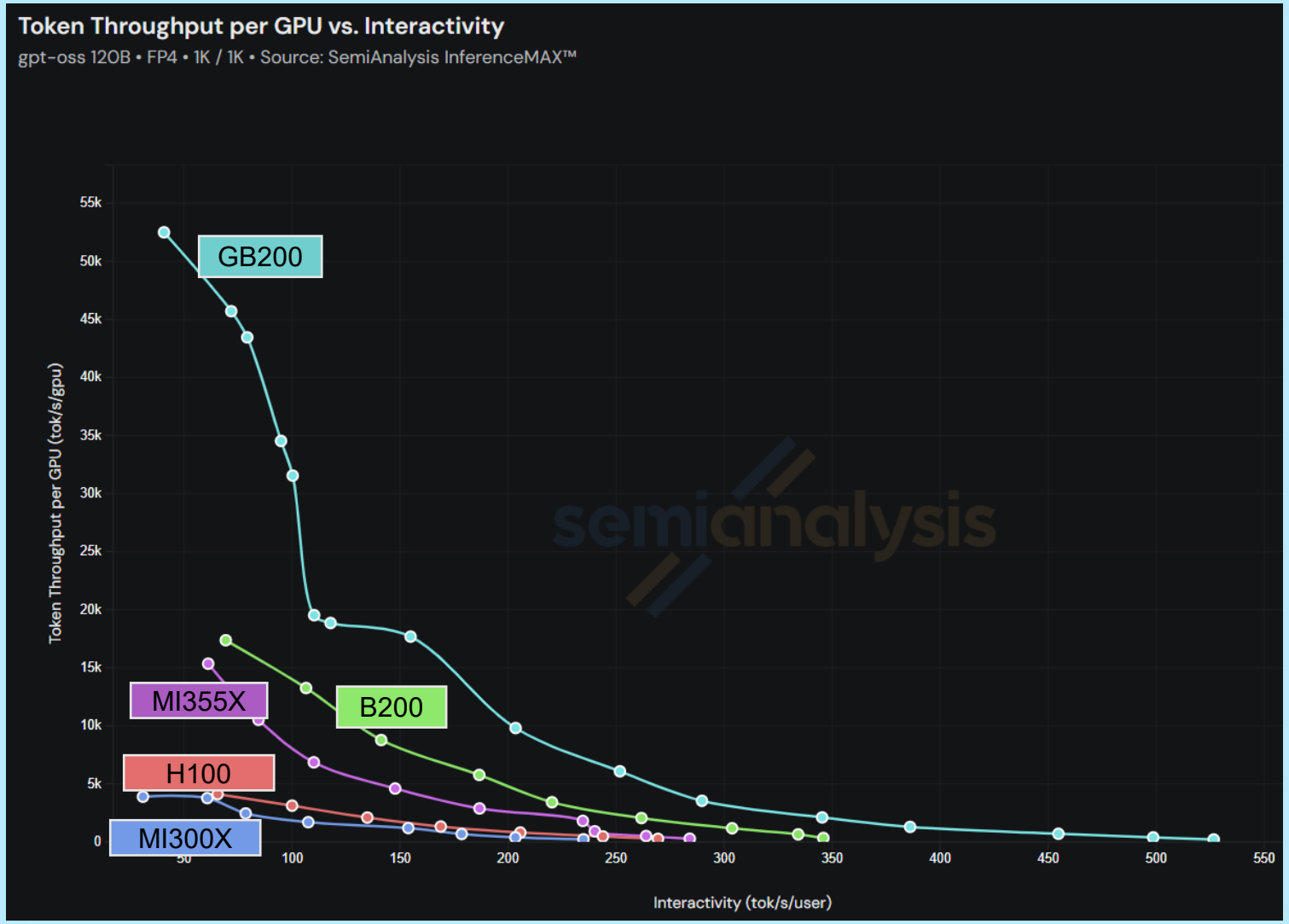
[1] LMCache/LMBenchmark: Systematic and comprehensive benchmarks for LLM systems.
[2] Scaling Multi-Turn LLM Inference with KV Cache Storage Offload and Dell RDMA-Accelerated Architecture | Dell Technologies Info Hub

Key Lessons from Benchmarking

- The real value of benchmarking emerges only through **end-to-end evaluation**, incorporating the **full software stack and GPUs**, rather than isolated micro-testing.
- Benchmark results can vary significantly based on **GPU type**, **model configuration**, and the **software stack**, including storage connectors.
  - Achieving expected storage performance requires **optimizing the layers above storage** in the software stack.
  - Blocking, synchronization, and waiting within the software stack can prevent storage from reaching its full potential.
- **Single-shot benchmarking:**
  - Highlights the *best-case* (hero) TTFT for a single request,
  - Does not reflect the true capabilities of the storage system, as the access pattern primarily produces a short **burst read**, masking sustained or concurrent storage behavior.
- **Increasing concurrency (batch size) or using multi-turn workloads:**
  - Provides a more realistic representation of production inference behavior.
  - However, these scenarios introduce additional factors—including **vLLM scheduling effects, decode overhead, and GPU bottlenecks**—which can obscure isolated storage performance.

Hardware Trends in AI Inference

[1] InferenceMAX by SemiAnalysis



- Rapid Growth in Per-GPU Inference Throughput
  - Each new generation of GPU delivers a significant increase in tokens per second per device.[1]
- Emergence of Inference-Specific Accelerators
  - Beyond general-purpose GPUs, there is a growing class of specialized hardware designed specifically for inference workloads
- Increasing Hardware Heterogeneity in Inference Deployments.

Implication: As inference hardware becomes faster and more diverse, storage and software orchestration play an increasingly critical role in end-to-end inference performance and efficiency.

Evolution of LLM Architectures and KV Cache Optimizations

- **Modern LLMs are no longer homogeneous**
  State-of-the-art models go beyond stacks of identical full-attention Transformer layers, resulting in heterogeneous architectures.
  - **Mixture of Experts (MoE):** Although overall model sizes are large, only a small subset of experts is activated per token, reducing compute cost per token.
  - **Sparse / Sliding-Window Attention:** Limits the attention context, reducing KV cache growth and attention compute, especially at long sequence lengths.
  - **Hybrid Models (Transformer + State Space Models):** Models such as Mamba and linear-attention variants reduce the cost of KV cache.
- **KV Cache Compression**
- **Context sizes are getting larger 1M, 10M:** Storage Capacity requirements getting larger.


SDC | StorageAI™
A SNIA Event
Thank You