THE BYTE ECONOMICS OF TRILLION-PARAMETER MOE INFERENCE ON CONSUMER HARDWARE

Anonymous authors
Paper under double-blind review

ABSTRACT

Mixture-of-experts language models now exceed a trillion parameters, while the largest consumer GPU holds 32 GB. We show that serving such a model on a desktop is governed by byte economics: decode time is the bytes each token moves across the memory hierarchy divided by the bandwidth of the links, and prefill time is set by how many times the expert store is read. We build OMOE, a tiered serving system that runs the native 2.8-trillion-parameter Kimi-K3 checkpoint (1.4 TB of MXFP4 experts, unmodified) on two RTX 5090s, 192 GB of DRAM and a 3.2 TB storage tier, with outputs gated byte-exact or within 6×10^{-7} of a full-VRAM reference. A bandwidth cost model with no fitted constants guided four system generations from 4.62 to 1.65 s/token at short context and places the software ceiling for this model on this machine at 0.58 s/token. A layer-major prefill schedule reads each expert at most once per prefill, so prefill throughput falls only from 69 to 47 tokens/s between 96K and one million tokens of context, where needle retrieval passes with 6.1 h to first token and 3.53 s/token decode. The instrumented deployment measures what a production MoE asks of a memory hierarchy: an LRU hit rate of 0.34 at 3% of the store, 3.0 TB written per million-token prefill, and the bytes a model designer could remove. Concurrent consumer-scale systems report contexts up to 65K; the system, telemetry and acceptance harnesses will be released.

1 INTRODUCTION

We present OMOE, a serving system that runs the 2.8-trillion-parameter Kimi-K3 checkpoint — 1.4 TB of MXFP4 expert weights, with weights and routing unmodified — on a single consumer desktop with two RTX 5090s, 192 GB of DRAM, and a 3.2 TB storage tier, at context lengths up to one million tokens. Outputs are gated byte-exact, or within a measured bound, against a full-VRAM reference, and GLM-5.2 is used for the prefill-schedule measurement. The same deployment doubles as a measurement instrument: it produces routing-locality and storage-traffic data for a production-scale MoE, and from these measurements the paper derives design guidance for both serving systems and future models.

Models of this class are otherwise served on multi-node datacenter pods — four 8×H100 nodes in the published recipes for K3[1] — while local deployments remain at dense models of a few to a few tens of billions of parameters, and an ongoing memory-supply crunch has pushed retail DDR5 to roughly five times its year-earlier price (3DCenter.org, 2026; IDC, 2025; TrendForce, 2026), closing off the large-DRAM path. Local serving nonetheless remains the setting where private data need not leave the machine and access depends on no provider. Concurrent systems — FreeToken (Yang et al., 2026), SSD-LLaMA (Liang et al., 2026), and Edge0 (Lin et al., 2026) — confirm the demand: all three serve MoE models from DRAM or SSD on single consumer GPUs, and all three operate at short context, with none reporting a native trillion-scale checkpoint served with verified numerics at long context. Figure 1 places these systems and ours on a common map.

Serving a model whose weights exceed host DRAM raises three coupled problems. Decode must move each token's routed experts across the memory hierarchy, so throughput is set by bytes and link

[1] Per the model's serving recipes in mainline frameworks; see §7.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

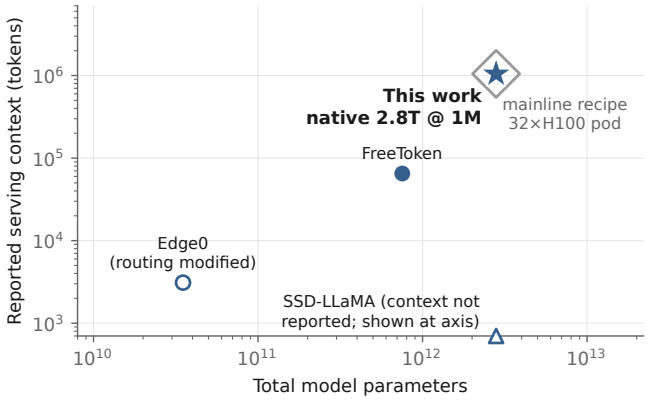

Figure 1: The consumer serving landscape at submission time: largest reported model vs. longest reported serving context per system on a single consumer node (filled: checkpoint served as released; open: routing modified; gray: the datacenter recipe for the same model). Concurrent systems cluster below 65K context; this work serves the native 2.8T checkpoint at one million tokens on two consumer GPUs.

bandwidths rather than by compute. Chunked prefill re-reads each layer's experts once per chunk, so at long context the store is re-read many times over. And the KV state and scratch of a million-token request exceed host memory, so a write path to storage becomes unavoidable, its volume deciding which media survive repeated use.

We address the three in order: a cost model with no fitted constants — bytes per token, measured hit rates and link bandwidths, a speculative multiplier — selects each system generation (§2); a *layer-major* schedule runs all chunks through one layer before the next, reading each expert at most once per prefill, bounded by the store size rather than the context length (§3); and the few full-attention layers' KV lives in pinned host memory, streamed through GPU tile rings inside the idle windows of expert traffic, while bounded scratch spills to an endurance-class tier (§5).

Results. Four system generations lowered short-context decode from 4.62 to 1.65 s/token, each targeting the term the accounting identified as dominant; the model's $H \to 1$ asymptote places the software ceiling at 0.58 s/token, so the remaining gap is an engineering budget and everything past it is model design (§6). Layer-major scheduling shortens a 15,360-token GLM-5.2 prefill by $12.1\times$ and holds prefill between 69 and 47 tokens/s from 96K to one million tokens, where retrieval passes with numerics bit-exact or within 6×10^{-7} of reference. At 3.53 s/token over a million tokens of context, the deployment serves as a verifier and planner over repository-scale context rather than an interactive assistant. The instrument measures an LRU hit rate of 0.337 at 3% of the store, a 79.5% dispatch-wave reduction from popularity-ordered admission, and 19.3 TB read with 3.0 TB written per million-token prefill.

Contributions. We *formalize* offloaded MoE serving as a two-part byte-economics cost model and validate it across system generations; we *build* a byte-exact tiered serving system that doubles as a routing- and storage-measurement instrument at production scale; we *demonstrate* million-token serving of a native trillion-parameter checkpoint on consumer hardware; and we *derive* a quantified ladder of model-design changes for memory-hierarchy-friendly MoEs. The system, telemetry, and acceptance harnesses will be released.

2 A BYTE-ECONOMICS COST MODEL FOR OFFLOADED MOE SERVING

2.1 SETUP AND NOTATION

Consider a MoE decoder with L MoE layers, top-k routing over E experts per layer, and quantized expert tensors of b bytes per expert per tensor-parallel rank. At batch size 1 (the interactive regime),

Gen	Change	H_{dram}	Measured s/tok
1	threaded pread, QD8	—	4.616
2	async O_DIRECT engine	—	3.258
3	pinned-slot DRAM cache (16 GiB, overlap off)	0.289	1.890
4	20 GiB cache + H2D/compute overlap	0.337	1.649

Table 1: Measured decode across system generations (K3, TP2, batch 1, greedy); the full measurement grid is Appendix E.

each decode step must materialize on each GPU the $k \cdot L$ routed experts selected by that token:

$$B_{\mathrm{tok}} \;=\; k\,L\,b \qquad \text{(per rank)}. \tag{1}$$

For Kimi-K3 (L=92, k=16, b=8,773,632 B in MXFP4 with block scales, TP2): $B_{\mathrm{tok}} = 12.9$ GB per rank per token — three orders of magnitude larger than the KV and activation traffic of the same step. The size of this number drives every design decision that follows.

Experts are served from a VRAM slot cache (hit fraction H_{gpu}, free), a pinned-DRAM cache (H_{dram} of the remainder, at PCIe speed $\mathrm{BW}_{\mathrm{pcie}}$), and the storage tier (the rest, at $\mathrm{BW}_{\mathrm{ssd}}$). Perfect overlap bounds the decode period by the slowest stream; no overlap, by the sum. Both share one core term:

$$t_{\mathrm{io}} \;=\; B_{\mathrm{tok}}\,(1 - H_{\mathrm{gpu}}) \left[\frac{H_{\mathrm{dram}}}{\mathrm{BW}_{\mathrm{pcie}}} \;+\; \frac{1 - H_{\mathrm{dram}}}{\mathrm{BW}_{\mathrm{ssd}}}\right], \qquad \mathrm{tok/s} \;\le\; \frac{\lambda}{t_{\mathrm{io}} + c}, \tag{2}$$

where c aggregates per-step compute and dispatch overhead (attention, kernel launches, router, combine) and $\lambda \geq 1$ is the expected number of tokens committed per weight-materialization (1 without speculation; the acceptance length with it). Equation 2 has no free parameters: B_{tok} is model geometry, $H_{\mathrm{gpu}}, H_{\mathrm{dram}}$ are measured cache telemetry, $\mathrm{BW}_{\mathrm{pcie}}, \mathrm{BW}_{\mathrm{ssd}}$ are measured link speeds, and c is measured once at full cache hit. The equation assumes host-to-device copies of SSD-missed bytes hide under the slower SSD reads while cache-hit copies are exposed; the fully-pipelined and fully-serial cases bound it from either side.

Three consequences follow. *Compute is not the binding constraint*: in the generation-1 per-token decomposition, expert and attention compute account for 0.13 s of a 4.62 s step (Appendix E), and the GPU's economic roles are capacity, format, and link count, not FLOPs. *The only free variables are H, B_{tok} and λ*: hardware fixes the denominators; systems work raises H and overlap; model design sets the rest (§6). *Routing skew substitutes for bandwidth*: with $\mathrm{BW}_{\mathrm{pcie}}/\mathrm{BW}_{\mathrm{ssd}} \approx 1.5$ and free VRAM hits, routing mass concentrated on a cacheable head is exactly equivalent to buying faster links (§4).

2.2 Validation across system generations

We validate Eq. 2 not on one operating point but on the *trajectory* of our system: each generation changed one term, and the accounting named the dominant term before each generation was built. Table 1 summarizes: generation 1 (threaded `pread`, QD8), generation 2 (asynchronous `O_DIRECT` engine, 14.7 GB/s aggregate; later gated pending lifecycle hardening, so production serves with threaded `preadv`, §3), generation 3 (pinned-slot DRAM cache: a cache hit *is* a zero-copy DMA source at 22.4 GB/s, removing a copy from the miss path as well), and generation 4 (wave-pipelined H2D/compute overlap). Measured decode moved $4.616 \to 3.258 \to 1.649$ s/token on K3, and the priced terms are auditable: at generation 1's measured 5.63 GB/s per-rank engine, reads alone cost 2.29 s of the 4.62 s step; at generation 2's 7.35 GB/s, 1.76 s of 3.26. The shrinking remainder is the software that later generations removed. These are development-trajectory measurements, not held-out predictions; single-variable re-runs on a second medium are ongoing.

2.3 Interpreting the residual

At $H_{\mathrm{dram}} \to 1$ the model's asymptote is $B_{\mathrm{tok}}/\mathrm{BW}_{\mathrm{pcie}} = 0.58$ s/token: the software-independent floor for this model on this machine, which no cache or overlap improvement can pass; only model-side changes to B_{tok}, H, or λ can (§6). Against the measured 1.649 s/token, the floor separates what engineering can still win from what requires a different model.

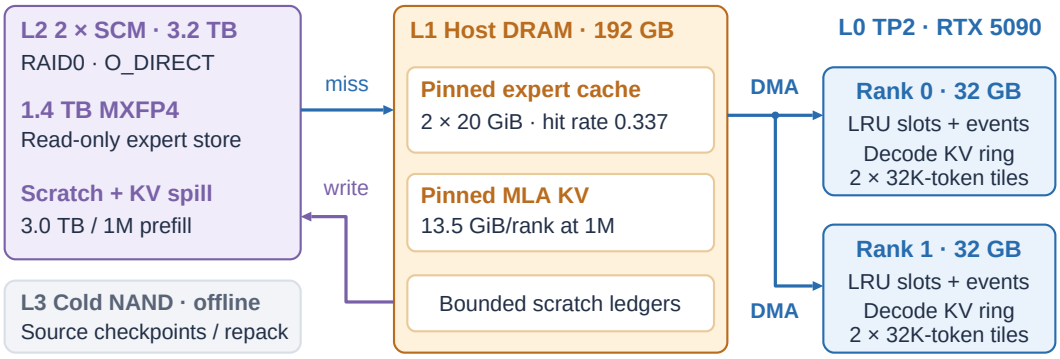

Figure 2: TP2 desktop, decode KV shown. The quoted rates are separate operating points: 22.4 GB/s/rank DMA (16 GiB cache, overlap off); hit rate 0.337 (20 GiB cache, overlap on); 14.87 GB/s array (`io_uring` probe). Production reads use threaded `preadv` at QD16.

3 The instrument: a byte-exact four-tier serving system

This section describes the system only to the depth needed to trust the measurements; full engineering detail ships with the artifact (see the Reproducibility statement).

Tier ownership. Figure 2 shows the hierarchy: VRAM expert slots under LRU with CUDA-event ownership; pinned-DRAM slots registered once, so a cache hit is itself a zero-copy DMA source; a storage tier read with `O_DIRECT`; and an offline NAND tier holding source checkpoints. The `io_uring` engine that saturates the array remains gated pending lifecycle hardening; the engine variable is controlled in Table 1. Four independent attempts to treat `mmap`-ed store files as memory each lost to explicit DMA staging (Appendix H): paging abstractions add per-byte cost that explicit ownership pays once, a lesson that recurs in the KV design (§5).

Byte-exact store. Expert tensors are repacked offline into the serving kernels' native ABI (MXFP4 weights, 128-row-interleaved block scales, four planes per expert per rank), so any kernel sharing this ABI upgrades without rebuilding the 1.45 TB store. Seeded spot checks decode stored experts against the source checkpoint at relative L_2 exactly zero, and acceptance gates hash every artifact.

Layer-major prefill. Chunked prefill re-fetches each layer's experts once per chunk. Running prefill *layer-major* (all chunks through layer ℓ before layer $\ell{+}1$) is a legal reassociation of the same computation that reads each expert from the store at most once per prefill, bounded by the store size rather than the context length; it took a 15,360-token GLM-5.2 TTFT from 15.5 minutes to 77 s. Decode is bandwidth-bound (Eq. 2); long prefill is the one regime where GPU compute returns to the critical path (§5).

Fail-closed operation. Unknown checkpoint tensors halt the loader; I/O errors poison affected cache slots and drain in-flight work before reuse; acceptance harnesses gate on byte identity and $\mathrm{rel}\, L_2 \leq 10^{-3}$ numerics. All throughput numbers were taken with every guard enabled.

Models. Kimi-K3: 2.8T parameters, 93 layers (92 MoE), 896 routed experts per layer with top-16 routing plus 2 shared experts, hybrid attention (69 KDA linear-attention layers; 24 MLA layers with 576 B/token latent KV in FP8), native MXFP4 experts. GLM-5.2: FP8 weights, used for the layer-major prefill measurement (0.262 tokens/s on generation-1 I/O). Both serve with all routed experts external to VRAM (loader-verified), TP2, greedy decoding.

4 Routing statistics: what the instrument already shows

Every expert fetch in OMOE crosses an instrumented boundary, so the deployment doubles as a measurement platform for routing locality in a 896-expert, 92-layer production MoE — a scale at

which routing-locality measurements have not previously been reported. This section reports what the production telemetry already establishes and states the full measurement agenda the platform is built to run.

4.1 Three measured facts

(1) LRU captures a third of routed bytes with 3% of the store. Under real decode workloads, a 20 GiB-per-rank LRU cache (3.0% of the measured 723.2 GB per-rank store) serves $H_{\mathrm{dram}} = 0.337$ of routed-expert bytes — routing is measurably skewed and temporally local, but far from cacheable. **(2) Within-batch expert unions saturate (an arithmetic fact).** A 2048-token prefill chunk makes $32{,}768$ top-16 draws over the 896 experts of a layer — a $36.6\times$ oversubscription at which the per-chunk expert union approaches the full layer, so chunk-level expert caching cannot help prefill; this is why layer-major scheduling (§3) is worth a constant-factor rewrite of the loop. **(3) Grouping by popularity beats grouping by identity.** Ordering expert admissions by descending assignment count reduced dispatch waves by 79.5% at 96K relative to ID-order batching — a pure scheduling win available only because routing mass is concentrated.

4.2 Implication and hypothesis: balance is a training artifact

MoE training flattens expert usage with auxiliary losses whose purpose is training-cluster utilization, not model quality. Under Eq. 2 flatness is the adversary: a uniform router at K3's geometry makes every cache tier useless below hundreds of GiB, while a skewed router of equal quality converts the same DRAM into bandwidth. The measured $H_{\mathrm{dram}} = 0.337$ bounds the skew that survives balance-trained routing; whether a *cache-aware objective* can steepen it at equal quality is a falsifiable training-time hypothesis that a frozen-model deployment cannot decide. The platform contributes the measurement half: replaying captured routing traces through LRU, static-pinning, and Belady-oracle policies yields the achievable-H frontier any training-side intervention should be scored against.

5 Million-token context on consumer hardware

5.1 Why hybrid models are structurally cheap at long context

K3 interleaves 69 KDA layers, whose per-request state is constant in sequence length, with 24 MLA layers costing 576 B/token/layer of FP8 latent KV. Long context grows only 13.5 KiB per token per rank — 1.27/3.38/13.5 GiB at 96K/256K/1M — trivial for host DRAM and pointless in VRAM; the question is whether streaming it back per decode step is affordable.

5.2 KV tiles hidden inside expert windows

The 24 MLA layers' KV lives in registered pinned host memory; each decode step streams it through a two-slot 32K-token tile ring on GPU, merging partial attention by online softmax, gated at $\mathrm{rel}\, L_2 \le 10^{-3}$ against a full-VRAM reference (measured 5.59×10^{-7}; extend exactly 0), with CUDA-event tile ownership and fault-injected poison-on-error semantics.

The scheduling opportunity comes from the architecture: MLA layers are *interleaved among* KDA and MoE computation whose expert fetches occupy PCIe in bursts with idle windows between waves. KV tile transfers are scheduled into exactly those windows. Measured decode against the 1.65 s/token short-context baseline: $+17\%$ at 96K (1.92 s/token), $+24\%$ at 256K (2.05), and $+114\%$ at 1M (3.53) — the million-token tier pays a real but bounded streaming tax, still within the interactive-batch regime the deployment targets (a duty-cycle generalization of this observation is given in Appendix H).

5.3 Acceptance and headline

Long-context acceptance ran twice. The initial build passed needle-retrieval gates with needles planted at 2%/50%/98% depth (covering first, middle, and the partial final KV tile) — 3/3 at 96K, 3/3 at 256K, and the approved two-depth partial (2%/98%) at 1M — under exact-substring match with greedy decoding and clean-teardown telemetry (zero leaked leases, tiles drained). An optimization round then eliminated the dominant scratch-ledger traffic (Figure 3) and paged prefill-resident weights;

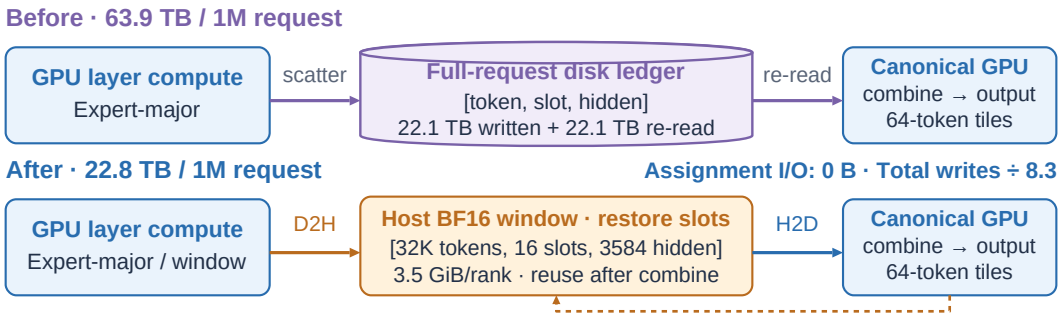

Figure 3: Both builds are layer-major: each expert is read at most once per prefill. Window reuse preserves canonical reduction (512-token logits: $\mathrm{rel}\, L_2 = 0$). The 1M-request totals include other optimizations: final prefill reads/writes 19.3/3.0 TB; decode/tail adds 0.44 TB.

Context	KV/rank	TTFT	Prefill tok/s	Decode s/tok	Needle	rel-L_2
96K	1.27 GiB	23.8 min	69	1.92	pass	0
256K	3.38 GiB	1.19 h	61	2.05	pass	0
1M	13.5 GiB	6.14 h	47	3.53	pass	0

Table 2: Final-build long-context results on 2×RTX 5090: needle retrieval under greedy decoding, 512-token logits gates byte-exact against a full-VRAM reference; prefill tok/s = prompt tokens / TTFT. Exact values for both builds: Appendix D.

the final build re-passed byte-exact 512-token logits gates against the same references at every step, and re-verified retrieval at each tier. Table 2 and Figure 4 report the final build:

Prefill cost. Prefill throughput degrades only $1.5\times$ from 96K to 1M — the layer-major signature (Figure 4). A 1M prefill reads 19.3 TB and writes 3.0 TB (breakdown: Appendix F), while the array averages $\sim$1 GB/s against its 14.87 GB/s capability: the medium is not the bottleneck. The remaining cost is $O(n^2)$ attention on the 24 MLA layers — hours on two consumer GPUs, the one regime where compute rather than bytes binds. We report it as a one-time indexing cost (the $\sim$27 GB KV state is checkpointable and resumable by design, though not yet implemented), not an interactive latency; the sizing corollary is that compute-dense accelerators matter for context *construction*, bandwidth and capacity for *use*.

5.4 Storage endurance: why the tier must be storage-class

The measured write budget determines which media can run this workload *repeatedly*. A million-token prefill writes 3.0 TB (KV spill, prefix persistence, bounded scratch) alongside its 19.3 TB of reads. Reads do not wear NAND; the writes do. Against the $\sim$1,200 TBW warranty of a high-end 2 TB consumer TLC drive (Samsung Electronics, 2022), nightly million-token sessions exhaust rated endurance in roughly 400 sessions — about a year — before counting development iterations. Storage-class media rated in the 100-drive-writes-per-day class (Intel Corporation, 2023) sustain the same schedule for the life of the platform (order 10^5 sessions), hold bandwidth through hours of mixed I/O where consumer drives exhaust their SLC caches, and provide the $\sim$10 μs QD1 reads the memory-layer end of §6 needs. Concurrent work prices SSD lifetime per served prompt (Liang et al., 2026); at million-token context the *session* is the right unit, and it is three orders of magnitude heavier. Against DRAM: consumer-socket platforms cap at 192–256 GB, under a fifth of the 1.4 TB pool, at prices the 2026 crunch has multiplied (3DCenter.org, 2026; IDC, 2025), while storage capacity remains at cents per GB (Liang et al., 2026).

Two clarifications apply. First, the medium is not the source of the headline numbers: the array averages 7% of its bandwidth during a 1M prefill, the expert-read path is multi-megabyte sequential streaming, and current consumer PCIe-5.0 drives exceed our discontinued PCIe-4.0 parts in peak sequential bandwidth — the tier was chosen for endurance and quality-of-service, not speed. The throughput results would largely survive on NAND; the endurance economics would not, and read-only designs avoid the write budget by giving up long-context serving. Second, we state the tier as a requirements profile — DWPD-class write endurance, sustained mixed I/O without cache cliffs,

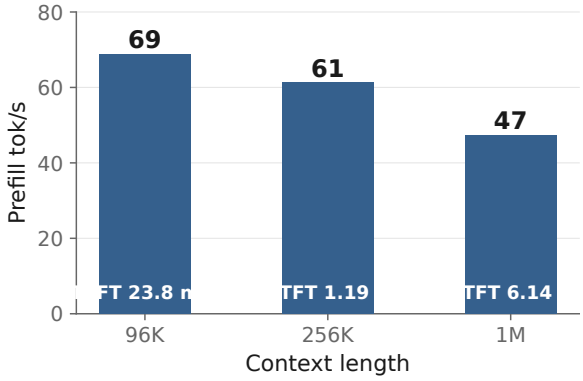

Figure 4: Prefill throughput across context lengths (tok/s = prompt tokens / TTFT; time to first token inside each bar). Layer-major scheduling reads each expert at most once per prefill, bounded by the store size, so throughput falls only $1.5\times$ from 96K to 1M.

Lever	Mechanism	B_{tok}/rank	Est. tok/s	IQ-parity evidence
(today)	—	12.9 GB	0.61 (meas.)	—
L1 skew-trained router	$H\uparrow$	12.9 GB	$\sim$1	routing degeneracy; §4
L2 resident-shifted actives	$B_{\text{tok}}\downarrow$	$\sim$6.5 GB	$\sim$1.8	shared/routed ablations
L3 base+delta experts	$B_{\text{tok}}\downarrow$	$\sim$1.9 GB	4–5	2-bit QAT / factorized experts
L4 native MTP + self-draft	$\lambda\uparrow$	$\sim$1.1 GB/acc	$\sim$10	exact by construction
L5 memory-layer capacity	$B_{\text{tok}}\Downarrow$	0.3–0.9 GB	20–30	mid-scale parity results

Table 3: The design ladder, cumulative, on fixed hardware (2×5090, measured links). "Est. tok/s" applies Eq. 2 with measured bandwidths and conservative overheads. The asymptote — the point where the machine becomes compute-bound again — is $\sim$50 tok/s at batch 1.

$\sim 10\,\mu$s QD1 reads — met by currently shipping datacenter-class NVMe drives, and the economics are robust to any procurement path: even at launch list price, the most expensive storage-class device in our tier costs several times less per gigabyte than the cheapest retail DDR5 kit does today (Intel Corporation, 2021; Webster, 2022; 3DCenter.org, 2026).

6 Designing models for memory hierarchies: a quantified ladder

Equation 2 leaves exactly three model-side variables: the bytes a token touches (B_{tok}), the cacheability of those bytes (H, set by routing statistics), and the tokens committed per fetch (λ). We organize the design space as five cumulative levers, each evaluated at equal intelligence — either provably (speculative decoding is distribution-preserving) or with the strongest available training-time evidence, graded per lever. Table 3 gives the arithmetic on our machine.

Three inversions of current practice. (1) *Load balancing.* Balance losses serve expert-parallel training utilization; on hierarchical-memory inference, skew of equal quality is strictly better, and only the training objective can create more of it (§4.2). (2) *Expert granularity.* Finer experts at fixed active bytes steepen the cacheable head and shrink verification unions (§4.1). (3) *Single-token decode.* Once B_{tok} has crossed the bus, verifying k drafted tokens is nearly free: under offload, speculation converts idle FLOPs into bandwidth at ratio λ — multi-token heads are *bandwidth features* here, not latency features.

Inference-time halves, measured. Two levers have frozen-model halves that we evaluate directly on K3: (a) *dynamic top-k*: truncating routed experts by router probability mass, bytes scaling with mean accepted k (quality calibration in progress); (b) *speculative drafting*: a checkpoint audit shows K3 ships *no* MTP head (`num_nextn_predict_layers`=0, zero matching tensors), so the frozen-model half is limited to training-free self-drafting and the L4 rung reads as a prescription for future checkpoints, not a measured K3 capability.

Memory-layer capacity. L5 replaces bulk FFN capacity with fine-grained retrieved memory (product-key / million-expert / ultra-sparse memory designs), turning per-token parameter access from 16×8.8 MB streams into kilobyte-granular random reads. This is precisely the access pattern where low-latency storage-class tiers are categorically superior to NAND — QD1 4K reads at $\sim$10 μs — and where, we note, our current workload *under*-uses the hardware: sequential multi-megabyte expert streams are NAND-feasible, while a memory-layer model would be Optane-native. If the community ships a trillion-scale memory-layer model, the analysis of §2 predicts 20–30 tokens/s on the desktop measured here, with the GPU finally compute-bound. We offer the prediction as a pre-registered test of the byte-economics framing.

Deployment pattern. The measured profile — hours of one-time ingestion, then seconds per token — suits a division of labor in which this model plans and audits over million-token context while a fast local dense model executes interactively, persistent KV making each consultation pay only its incremental suffix. We report the pattern, not an evaluated system.

7 Related work

Offloaded LLM inference. FlexGen (Sheng et al., 2023) established throughput-oriented offloading with a linear-programming placement model for dense models; LLM-in-a-flash (Alizadeh et al., 2024) read sparse FFN rows directly from flash with windowed caching; PowerInfer (Song et al., 2024) exploited activation sparsity on consumer GPUs. Our cost model is closest to FlexGen's but targets expert-routed traffic at 2.8T scale, adds cache-hit and speculative terms, and is used predictively across system generations.

MoE-specific offloading. Expert caching and prefetching are studied in Mixtral-offloading (Eliseev & Mazur, 2023), MoE-Infinity (Xue et al., 2024), Pre-gated MoE (Hwang et al., 2024), SiDA (Du et al., 2024), Fiddler (Kamahori et al., 2024), PIPO (Liu et al., 2025), MoEpic (Yan et al., 2025), and FineMoE (Yu et al., 2025); KTransformers (Chen et al., 2025) and llama.cpp made CPU+GPU MoE serving common practice. Three concurrent systems define the current consumer frontier (Appendix C and Figure 5 there). FreeToken (Yang et al., 2026) serves DRAM-resident pools with a bandwidth-adaptive split between cache fills and CPU-side execution: 22–25 tok/s for a 284B model on an RTX 5090, largest 753B on a 512 GB-class host, contexts to 65K. The pool must fit in DRAM, which our 2.8T store exceeds by $\sim$7×; its reported decode-time LRU miss rates (16–39%) corroborate the locality we quantify in §4. SSD-LLaMA (Liang et al., 2026) serves trillion-scale models from one NVMe device with per-token expert fetches at decode (K3: 0.465 tok/s on one RTX 5090) and expert-grouped prefill at 1.2 tok/s — a week-scale extrapolation for a million-token prompt. Its serving is read-only, so single-pass prefill is capped by what host memory holds in flight; our scratch-and-spill write path amortizes one store read over the entire prefill, whatever its length — 47 tok/s at 1M on 1.65× its array bandwidth. Edge0 (Lin et al., 2026) trains a prerouter that consumes its own one-layer-ahead prediction: 20 tok/s for a 35B MoE in 3 GiB of active memory, at the cost of altered routing and a measured gap to its teacher — a different fidelity contract from serving the checkpoint unmodified. Earlier K3-capable systems (AirLLM (Li & contributors, 2026) at a reported 292 s/token; llama.cpp's $\sim$1-bit-class RAM-resident variants) trade latency or fidelity for residency (Appendix C). Long-context correctness has been hard ecosystem-wide: community reports through August 2026 documented mainline engines collapsing beyond $\sim$240K context (llama.cpp community, 2026; vLLM issue tracker, 2026), and KDA state-handling fixes landed upstream only weeks before this submission — our deployment carried byte-exact gates from the first tier because long-context state corruption is silent. The recipes that serve K3 at 1M provision multi-node H100-class pods.

Speculative decoding under memory constraints. Speculative decoding (Leviathan et al., 2023; Chen et al., 2023) preserves the target distribution; trained-draft variants extend it with learned heads (EAGLE (Li et al., 2024) under exact verification, Medusa (Cai et al., 2024) under relaxed acceptance). SpecExec (Svirschevski et al., 2024) and SpecMoEOff (Wang et al., 2025) apply speculation to offloaded serving. We add the byte-economics account: under offload, acceptance length multiplies *bandwidth*, and MoE verification cost is governed by expert-union overlap; the target checkpoint ships no draft head (§6).

MoE design. Fine-grained and shared experts (Dai et al., 2024), expert-choice routing (Zhou et al., 2022), auxiliary-loss-free balancing (DeepSeek-AI, 2024b), hybrid linear attention (KDA; Kimi Team, 2025), and MLA (DeepSeek-AI, 2024a) define the design space; memory-layer capacity (Lample et al., 2019; He, 2024; Berges et al., 2024; Huang et al., 2025) and low-bit training (Ma et al., 2024) point at the L3/L5 rungs of our ladder; balancing objectives trace to Switch (Fedus et al., 2022). Our contribution here is not a new architecture but the quantified inference-side objective: what routing statistics and parameter layouts are worth, in tokens per second, on hardware most users own.

8 LIMITATIONS

Our measurements come from one machine and two models; the telemetry findings (§4) should be replicated on other routers before being treated as universal. Output-quality verification beyond retrieval is in progress; claims are scoped to serving correctness and measured performance, not end-task quality. Million-token prefill costs 6.1 h, a one-time cost whose residual is $O(n^2)$ MLA attention (array $\sim$1 GB/s against 14.87); interactive long-context use assumes checkpointable sessions, not yet implemented. Needle retrieval at the final build ran one depth per tier (the initial build passed 2%/50%/98%), and each tier is a single multi-hour run: we report exact values with full artifacts rather than variance. Baseline comparisons are limited for a concrete reason: no second system serves this model at native precision and long context (§7); we therefore validate against our own single-variable-controlled generations and publish the full harness.

9 CONCLUSION

Trillion-parameter MoE decode on consumer hardware is well described by a parameter-free cost model that guided each generation of our system, culminating in 0.61 tokens/s short-context decode and verified million-token retrieval (6.1 h one-time prefill, 3.53 s/token decode) for a 2.8T model on two consumer GPUs. The larger gains are model-side: balanced routers, coarse experts, and single-token heads are each worth multiples of throughput if inverted at equal quality (§6). We release the system, telemetry, and acceptance harnesses so the next such model can be measured directly.

AI USE STATEMENT

Large language models were used in two capacities in this work. First, as engineering collaborators: substantial parts of the serving system, tests, and telemetry tooling were implemented by AI coding assistants working under author-specified designs, and every change was admitted only through the acceptance gates of Appendix A (byte-exact or bounded-error numerics against reference paths, full test suites, and artifact-hashed run evidence). All measurements were produced by instrumented runs and verified by the authors against raw artifacts. Second, as writing assistants: the manuscript was drafted and revised with LLM assistance under author direction, and every quantitative claim was audited against a claim-to-artifact ledger maintained alongside the paper source. The authors take full responsibility for all content.

REPRODUCIBILITY STATEMENT

All measurements were taken on a single fixed machine (Ryzen 9950X, 192 GB DDR5-5600, two RTX 5090 32 GB on PCIe Gen5 x16, two Optane P5800X 1.6 TB in RAID0). Software pins: SGLang v0.5.17, FlashInfer v0.6.17, CUDA 13.0.2. The serving system, store builder, telemetry, and all acceptance harnesses will be released as open source; the repository link will accompany the camera-ready/arXiv listing. Store integrity has two distinct audits, reported separately: source correspondence is established by frozen-seed spot checks (8/8 sampled (layer, expert, rank) tensors decode byte-exactly against the source checkpoint, relative L_2 exactly 0), and whole-store self-consistency by full SHA-256 manifests; the manifests do not by themselves prove source correspondence.

REFERENCES

3DCenter.org. DDR5 retail price index: August 2026 survey. 3DCenter retail price survey, August 2026. URL https://www.3dcenter.org/artikel/speicherkrise-preisindex-august-2026. Reports the tracked DDR5 index at 486% of its July 2025 baseline; accessed September 18, 2026.

Keivan Alizadeh, Iman Mirzadeh, Dmitry Belenko, Karen Khatamifard, Minsik Cho, Carlo C Del Mundo, Mohammad Rastegari, and Mehrdad Farajtabar. LLM in a flash: Efficient large language model inference with limited memory. In *ACL*, 2024.

Vincent-Pierre Berges, Barlas Oğuz, et al. Memory layers at scale. *arXiv preprint arXiv:2412.09764*, 2024.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D Lee, Deming Chen, and Tri Dao. Medusa: Simple LLM inference acceleration framework with multiple decoding heads. In *ICML*, 2024.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Hongtao Chen, Weiyu Xie, Boxin Zhang, Jingqi Tang, Jiahao Wang, Jianwei Dong, Shaoyuan Chen, Ziwei Yuan, Chen Lin, Chengyu Qiu, Yuening Zhu, Qingliang Ou, Jiaqi Liao, Xianglin Chen, Zhiyuan Ai, Yongwei Wu, and Mingxing Zhang. KTransformers: Unleashing the full potential of CPU/GPU hybrid inference for MoE models. In *Proceedings of the ACM SIGOPS 31st Symposium on Operating Systems Principles (SOSP)*, 2025. doi: 10.1145/3731569.3764843.

Damai Dai et al. DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

DeepSeek-AI. DeepSeek-V2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024a.

DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024b.

Zhixu Du, Shiyu Li, Yuhao Wu, Xiangyu Jiang, Jingwei Sun, Qilin Zheng, Yongkai Wu, Ang Li, Hai Li, and Yiran Chen. SiDA: Sparsity-inspired data-aware serving for efficient and scalable large mixture-of-experts models. In *MLSys*, 2024.

Artyom Eliseev and Denis Mazur. Fast inference of mixture-of-experts language models with offloading. *arXiv preprint arXiv:2312.17238*, 2023.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *JMLR*, 2022.

Xu Owen He. Mixture of a million experts. *arXiv preprint arXiv:2407.04153*, 2024.

Zihao Huang, Qiyang Min, Hongzhi Huang, Defa Zhu, Yutao Zeng, Ran Guo, and Xun Zhou. Ultra-sparse memory network. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=zjeHLSiNv1.

Ranggi Hwang, Jianyu Wei, Shijie Cao, Changho Hwang, Xiaohu Tang, Ting Cao, and Mao Yang. Pre-gated MoE: An algorithm-system co-design for fast and scalable mixture-of-expert inference. In *ISCA*, 2024.

IDC. Global memory shortage crisis: Market analysis and the potential impact on the smartphone and PC markets in 2026. IDC Resource Center blog, 2025. URL https://www.idc.com/resource-center/blog/global-memory-shortage-crisis-market-analysis-and-the-potential-impact-on-the-smar Accessed September 26, 2026.

Intel Corporation. Intel optane SSD DC P5800X 1.6 TB: product specifications. Intel ARK product page, 2021. URL https://www.intel.com/content/www/us/en/products/sku/201859/intel-optane-ssd-dc-p5800x-series-1-6tb-2-5in-pcie-x4-3d-xpoint/specifications.html. Official 1.6 TB product record; the current discontinued-product page no longer exposes its historical recommended customer price; accessed September 18, 2026.

Intel Corporation. Intel optane SSD P5800X series product brief. Product brief, 2023. URL https://cdrdv2-public.intel.com/772801/Intel-Optane-SSD-P5800X-Product-Brief.pdf. Endurance rated at 100 drive writes per day; accessed September 18, 2026.

Keisuke Kamahori, Yile Gu, Kan Zhu, and Baris Kasikci. Fiddler: CPU-GPU orchestration for fast inference of mixture-of-experts models. *arXiv preprint arXiv:2402.07033*, 2024.

Kimi Team. Kimi linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025.

Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Large memory layers with product keys. In *NeurIPS*, 2019.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *ICML*, 2023.

Gavin Li and contributors. AirLLM: inference of very large models with layer-wise and per-expert streaming. https://github.com/lyogavin/airllm, 2026. Kimi-K3 per-expert streaming added July 2026.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE: Speculative sampling requires rethinking feature uncertainty. In *ICML*, 2024.

Fangzhou Liang, Yibin Shen, Jianmin Hu, Jiayang Xu, Hanchi Gao, Minxian Xu, and Zili Meng. SSD-LLaMA: SSD-native inference for trillion-parameter MoE at 1+ token/s on a consumer PC. *arXiv preprint arXiv:2609.18110*, 2026.

Yu Lin, Yiming Wang, Runyuan Cai, Hanze Liu, and Xiaodong Zeng. The other half of the memory wall: Serving 35B MoEs from SSD with trained routing prediction. *arXiv preprint arXiv:2609.18063*, 2026.

Yangyijian Liu, Jun Li, and Wu-Jun Li. PIPO: Pipelined offloading for efficient inference on consumer devices. *arXiv preprint arXiv:2504.03664*, 2025.

llama.cpp community. Kimi K3: pre-release analysis and groundwork for llama.cpp support. GitHub discussion #26041, ggml-org/llama.cpp, 2026. URL https://github.com/ggml-org/llama.cpp/discussions/26041. Accessed September 26, 2026.

Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. The era of 1-bit LLMs: All large language models are in 1.58 bits. *arXiv preprint arXiv:2402.17764*, 2024.

Samsung Electronics. Samsung SSD 990 PRO data sheet. Product data sheet, 2022. URL https://download.semiconductor.samsung.com/resources/data-sheet/Samsung_NVMe_SSD_990_PRO_Datasheet_Rev.1.0_10129514072296.pdf. 2 TB model rated 1,200 TBW; accessed September 18, 2026.

SGLang contributors. Kimi-k3 serving cookbook, 2026. URL https://github.com/sgl-project/sglang/blob/main/docs/cookbook/autoregressive/Moonshotai/Kimi-K3.mdx. Accessed September 26, 2026; H100 recipe: four eight-GPU nodes.

Ying Sheng, Lianmin Zheng, Binhang Yuan, Zhuohan Li, Max Ryabinin, Beidi Chen, Percy Liang, Christopher Ré, Ion Stoica, and Ce Zhang. FlexGen: High-throughput generative inference of large language models with a single GPU. In *ICML*, 2023.

Yixin Song, Zeyu Mi, Haotong Xie, and Haibo Chen. PowerInfer: Fast large language model serving with a consumer-grade GPU. In *SOSP*, 2024.

Ruslan Svirschevski, Avner May, Zhuoming Chen, Beidi Chen, Zhihao Jia, and Max Ryabinin. SpecExec: Massively parallel speculative decoding for interactive LLM inference on consumer devices. In *NeurIPS*, 2024.

TrendForce. AI server demand to drive memory contract price increases in 2Q26 as CSPs secure supply via long-term agreements. TrendForce Press Center, March 2026. URL https://www.trendforce.com/presscenter/news/20260331-12995.html. Published March 31, 2026; accessed September 18, 2026.

Unsloth. Kimi k3: Dynamic quantization and hardware requirements, 2026. URL https://unsloth.ai/docs/models/kimi-k3. Accessed September 26, 2026; UD-IQ1_S weights 594.0 GB.

vLLM issue tracker. KDA chunked prefill uses wrong recurrent state layout and breaks Kimi-linear long-context retrieval. GitHub issue #41292, vllm-project/vllm, 2026. URL https://github.com/vllm-project/vllm/issues/41292. Accessed September 26, 2026.

Zhibin Wang, Zhonghui Zhang, Yuhang Zhou, Zibo Wang, Mo Zhou, Peng Jiang, Weilin Cai, Chengying Huan, Rong Gu, Sheng Zhong, and Chen Tian. Accelerating mixture-of-experts inference by hiding offloading latency with speculative decoding. *arXiv preprint arXiv:2508.21706*, 2025.

Sean Webster. Intel optane SSD DC P5800X review: The fastest SSD ever made. Tom's Hardware, 2022. URL https://www.tomshardware.com/reviews/intel-optane-ssd-dc-p5800x-review. Contemporaneous specification table lists the 1.6 TB model at US$3,724; accessed September 18, 2026.

Leyang Xue, Yao Fu, Zhan Lu, Luo Mai, and Mahesh Marina. MoE-Infinity: Activation-aware expert offloading for efficient MoE serving. *arXiv preprint arXiv:2401.14361*, 2024.

Jiaming Yan, Jianchun Liu, Hongli Xu, and Liusheng Huang. Accelerating mixture-of-expert inference with adaptive expert split mechanism. *arXiv preprint arXiv:2509.08342*, 2025.

Shuo Yang, Xiaoze Fan, Melissa Pan, Haocheng Xi, Zhe Wang, Shanlin Sun, Kurt Keutzer, Song Han, Matei Zaharia, Chenfeng Xu, and Ion Stoica. FreeToken: Efficient edge-native MoE serving with bandwidth-adaptive execution. *arXiv preprint arXiv:2608.16157*, 2026.

Hanfei Yu, Xingqi Cui, Hong Zhang, Hao Wang, and Hao Wang. Taming latency-memory trade-off in MoE-based LLM serving via fine-grained expert offloading (FineMoE). *arXiv preprint arXiv:2502.05370*, 2025.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, et al. Mixture-of-experts with expert choice routing. In *NeurIPS*, 2022.

A Acceptance protocol summary

Every tier passes: (i) numerics gates against full-VRAM references (decode rel-$L_2 \leq 10^{-3}$, achieved 5.6×10^{-7}; extend exactly 0); (ii) needle retrieval at 2%/50%/98% planted depths (chosen to cover the first, middle, and partial-final KV tiles), single-occurrence asserted, greedy decoding, exact-substring pass; (iii) clean-teardown evidence (zero leaked leases/borrows, tile rings drained, GPUs returned to idle); and (iv) fail-closed lifecycle behavior under injected faults. The full G0–G5 ignition ladder and per-tier runbooks ship with the artifact.

B SELF-CONSISTENCY CROSS-CHECKS

The headline numbers over-constrain one another, and the redundancy is itself evidence. (i) *Traffic vs. time*: the 1M run's 22.8 TB of measured array traffic over 22,108 s implies 1.03 GB/s average throughput, consistent with the phase-sampled I/O telemetry and far under the 14.87 GB/s ceiling — prefill is attention-bound, not I/O-bound, exactly the layer-major design intent. (ii) *Decode tax vs. KV bytes*: decode rises from 1.92 s/token at 96K to 3.53 at 1M while streamed MLA KV grows by 12.2 GiB/rank; the implied effective KV stream rate ($\sim$8 GB/s) sits plausibly between the array and pinned-DMA ceilings. (iii) *Prefill rate vs. context*: per-tier throughput (69/61/47 tok/s) falls by the growth of the $O(n^2)$ MLA term over a bounded store read — prefill store reads are capped by the 1.45 TB store itself ($\sim$97 s at measured bandwidth), and the 1.86 TB whole-request store bucket additionally contains decode-time cache re-reads (factor 1.28, attributed in telemetry). Each check combines independently logged quantities (wall clocks, I/O phase samples, store manifests).

C MECHANISM-LEVEL COMPARISON WITH CONCURRENT SYSTEMS

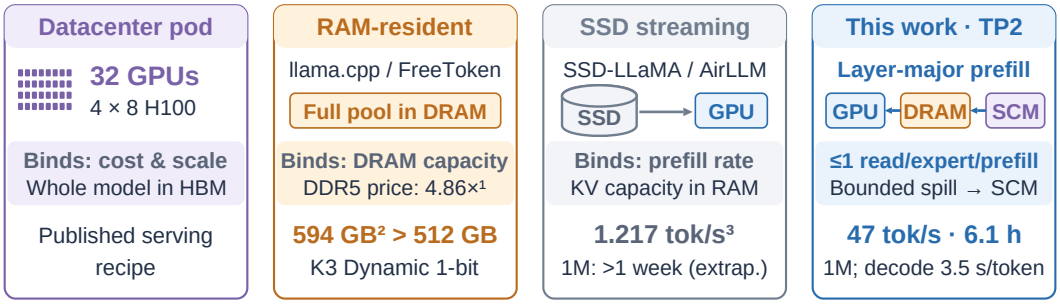

Figure 5: Different weights/workloads. H100: (SGLang contributors, 2026); [1]DDR5 Aug. 2026/July 2025: (3DCenter.org, 2026); [2]K3 weights: (Unsloth, 2026). FreeToken: 753B/512 GB-class max. (Yang et al., 2026). [3]Short-context (Liang et al., 2026); 1M rate-extrapolated; ours measured with scratch spill.

Values are as reported by each paper; "n/r" = not reported.

	FreeToken	Edge0	SSD-LLaMA	This work
Weight residence	host DRAM	SSD	SSD (1 NVMe)	storage tier (2 SCM drives)
Largest model	753B (NVFP4)	35B (int4-g64)	2.8T	2.8T (native MXFP4)
Longest context	65K (agent traj.)	3.1K (measured)	n/r	1,048,512 (verified)
Prefill amortization	full-layer double buffer, DRAM-resident	predicted routing, one layer ahead	within-layer expert grouping, memory-bounded	one store pass per prefill at any length
Serving-time writes	none	none	none	3.0 TB per 1M session (KV spill, prefix, scratch)
Fidelity contract	native blocks / BF16 tiers	modified routing + recovery LoRA	transport-lossless; K3 serving format n/r	checkpoint and routing unmodified
Output verification	n/r	benchmark vs. teacher (-3.9 pts avg.)	transport byte-identity	logits byte-exact / $\leq 6 \times 10^{-7}$ vs. reference
Endurance account	n/r	n/r	TBW estimate per prompt	measured read/write phase split per session

Table 4: Mechanism map across concurrent consumer-scale MoE serving systems, as reported by each paper.

Earlier K3-capable systems. For completeness beyond the concurrent trio: AirLLM added per-expert streaming for Kimi-K3 in July 2026, running the unmodified 2.8T model in under 4 GB of VRAM at a reported 292 s/token — an existence proof two orders of magnitude from interactive

use, without long-context support. llama.cpp merged text-model support for this architecture in mid-August 2026; published community runs reach $\sim$5 tokens/s on K3 variants compressed to $\sim$1-bit-class dynamic formats (594 GB of weights alone (Unsloth, 2026)) so that weights become RAM-resident on 600 GB-class or multi-A100 hosts — trading fidelity for residency, where we trade a storage tier for both. KTransformers serves smaller MoE families (through K2.5/K2.6 and GLM-5.3-flash at 1M context) via CPU/GPU hybrid execution with AMX kernels, but does not support K3 at the time of writing.

D EXACT LONG-CONTEXT RESULTS, BOTH BUILDS

Table 5 gives unrounded values for both acceptance campaigns. The initial build established multi-depth retrieval; the optimization round (assignment-ledger elimination, prefill-resident paging, scratch-release ordering) was admitted through byte-exact 512-token logits gates at every step and re-verified retrieval at each tier. Peak memory *rises* at 256K/1M in the final build by design: prefill-resident paging trades host memory for a fully GPU-resident MLA KV during prefill.

Build	Tier	TTFT (s)	Decode (s/tok)	Peak (GiB)	Needle
initial	96K	1426.909	1.933	—	3/3 (2/50/98%)
initial	256K	7719.003	2.057	85.3	3/3 (2/50/98%)
initial	1M D02	35469.603	3.533	159.6	pass (2%)
initial	1M D98	9.863 h	3.556	159.6	pass (98%)
final	96K	1425.114	1.924	153.9	pass (2%)
final	256K	4270.478	2.048	125.2	pass (2%)
final	1M	22108.112	3.530	165.0	pass (2%)

Table 5: Both acceptance campaigns, exact values. Prompt lengths: 98,240 / 262,081 / 1,048,518 tokens. The final-build 256K TTFT improvement is $1.81\times$; 1M is $1.60\times$. The 1M final run rode its 165 GiB cgroup ceiling with 4 GiB of swap and zero OOM events — the memory-admission model treats this as the bound to re-pin, not headroom.

Needle responses (greedy decoding, exact-substring criterion; trailing serialization tokens elided; full raw outputs ship in the artifact):

```
K3-F96K-7Q9M2X-D02 / I found the authoritative needle.
K3-F256K-4V8R6P-D02 / I found the authoritative needle.
K3-F1M-9C2W7N-D02 / I found the authoritative needle.
```

E DECODE GENERATIONS: THE FULL MEASUREMENT GRID

The four-generation trajectory of Table 1 compresses the following controlled grid (TP2, batch 1, greedy; per-token bytes are critical-rank values; hit-path H2D measured with `staging_cache_copy_bytes`=0, i.e., a cache hit is consumed as a zero-copy DMA source).

Read engine	Cache / overlap	H_{dram}	miss GB/tok	s/token
threaded `preadv`, QD8	off / off	0	12.915	4.616
`io_uring` `O_DIRECT`, QD16	off / off	0	12.915	3.258
`io_uring` `O_DIRECT`, QD32	off / off	0	12.915	3.241
production	16 GiB / off	0.289	7.222	1.890
production	16 GiB / on	0.290	7.213	1.767
production	20 GiB / on	0.337	6.459	1.649

Table 6: Decode measurement grid. The QD16$\rightarrow$QD32 step gains only 0.5%, locating the knee of the array; cache steps then trade DRAM for array traffic exactly as Eq. 2 prices them.

Two methodological notes. Equation 2's two overlap regimes (fully pipelined, fully serial) bound the operating point; the measured overlap fraction of a given engine build selects where between them a configuration lands, which is how the same equation prices both the overlap-off and overlap-on rows above. And the GPU's three economic roles in this regime are worth spelling out: *capacity* (the

fraction of bytes it can pin resident), *format* (which quantized types its copy and compute engines accept natively), and *link count* (how many independent host links feed it) — FLOPs appear nowhere in the decode denominator. Under offload, speculation's exchange rate is also strictly better than on HBM machines: there, acceptance length only shortens latency; here it multiplies effective bandwidth, because verifying k tokens reuses weights that already crossed the bus.

F STORAGE TRAFFIC ACCOUNTING, BEFORE AND AFTER

Dual-rank decimal TB per 1M-token needle. The optimization round eliminated the assignment ledger entirely and moved residual-ledger reclaim to precise windows; the remaining write budget is the structural cost of long-context prefill.

Initial build	read	write	Final build	read	write
assignment ledger	22.126	22.126	assignment ledger	0	0
MoE prefix	2.766	2.766	prefill (all)	19.314	3.017
attention residual	12.025	0.241	decode + teardown	0.441	$<$0.001
expert store	1.855	—			
total	38.772	25.133	total	19.755	3.017

Table 7: Per-needle storage traffic at 1M context. Total traffic fell $2.8\times$ (63.9 to 22.8 TB); writes fell $8.3\times$. The expert-store bucket totals 1.86 TB per request in both builds: prefill reads each expert at most once (bounded by the 1.45 TB store); the remainder is decode-time cache re-reads.

G MEMORY ADMISSION AND FAIL-CLOSED LIFECYCLE

Launches are gated by a closed-form worst-case memory model (per-tier scratch, pinned arenas, KV, and a service floor) that is re-pinned against measured `memory.peak` after every accepted run; bounds for mutually exclusive phases take the maximum, not the sum. The model under-predicted twice during development (a KV arena term hidden in a calibrated floor; a transient overlap of prefill scratch with decode-layout restore); both incidents are recorded in the artifact with their fixes, and the shipped bound is the measured ceiling. Failure injection verifies the lifecycle contract: I/O errors poison affected cache slots and drain in-flight work before reuse; teardown asserts zero leaked leases and borrows, drained tile rings, and idle GPUs; an orphaned-scratch reaper takes an exclusive advisory lock and validates process identity via start-time lease before deleting anything.

H ENGINEERING NOTES

Findings that shaped the system but did not fit the main text; each is documented with artifacts in the release.

Paging abstractions lose to explicit DMA. Four independent attempts to treat `mmap`-ed store files as memory — with and without `MAP_POPULATE`, huge pages, and `madvise` prefetch hints — each lost to explicit `O_DIRECT` staging into registered pinned buffers: a page-cache hit still pays a kernel copy, and page-fault scheduling is invisible to CUDA streams, so faults land inside compute windows the scheduler believes are free.

Per-op dispatch tax on many-core hosts. Scratch-ledger maintenance originally issued many small tensor-library ops per wave; on a many-core host each paid a parallel-region setup cost three orders of magnitude above the equivalent `numpy` call. Rebuilding the append path on `numpy` with a per-bucket coalescing buffer cut ledger maintenance from 42.7 to 3.53 s per layer ($12\times$) with byte-identical output: on CPU-adjacent paths, dispatch overhead rather than arithmetic dominates.

What the byte-exact gates caught. During the optimization round alone, the byte-exact gates caught (i) a cross-stream ordering bug in which auxiliary CUDA streams consumed paged-in weights before the H2D event was CPU-synchronized (observed rel-$L_2 = 0.186$; fixed by publishing

parameter views only after event synchronization), and (ii) two tempting fast paths — a pinned-bounce decode-weight restore and a per-process page-store hash cache — each of which produced rel-$L_2 \approx 0.19$ under the forced file-backed tier and was fully reverted. Long-context state corruption is silent at the output level until it is catastrophic; logits-level gates make it loud immediately.

The idle-PCIe dividend, generalized. KV tile transfers hide inside the duty-cycle gaps that expert fetches leave on the host link. The observation generalizes: any MoE whose expert traffic leaves link-idle windows ships long context "for free" up to the gap budget — a second, unadvertised dividend of hybrid attention after the KV capacity itself, and a scheduling opportunity any offloaded serving system can exploit.