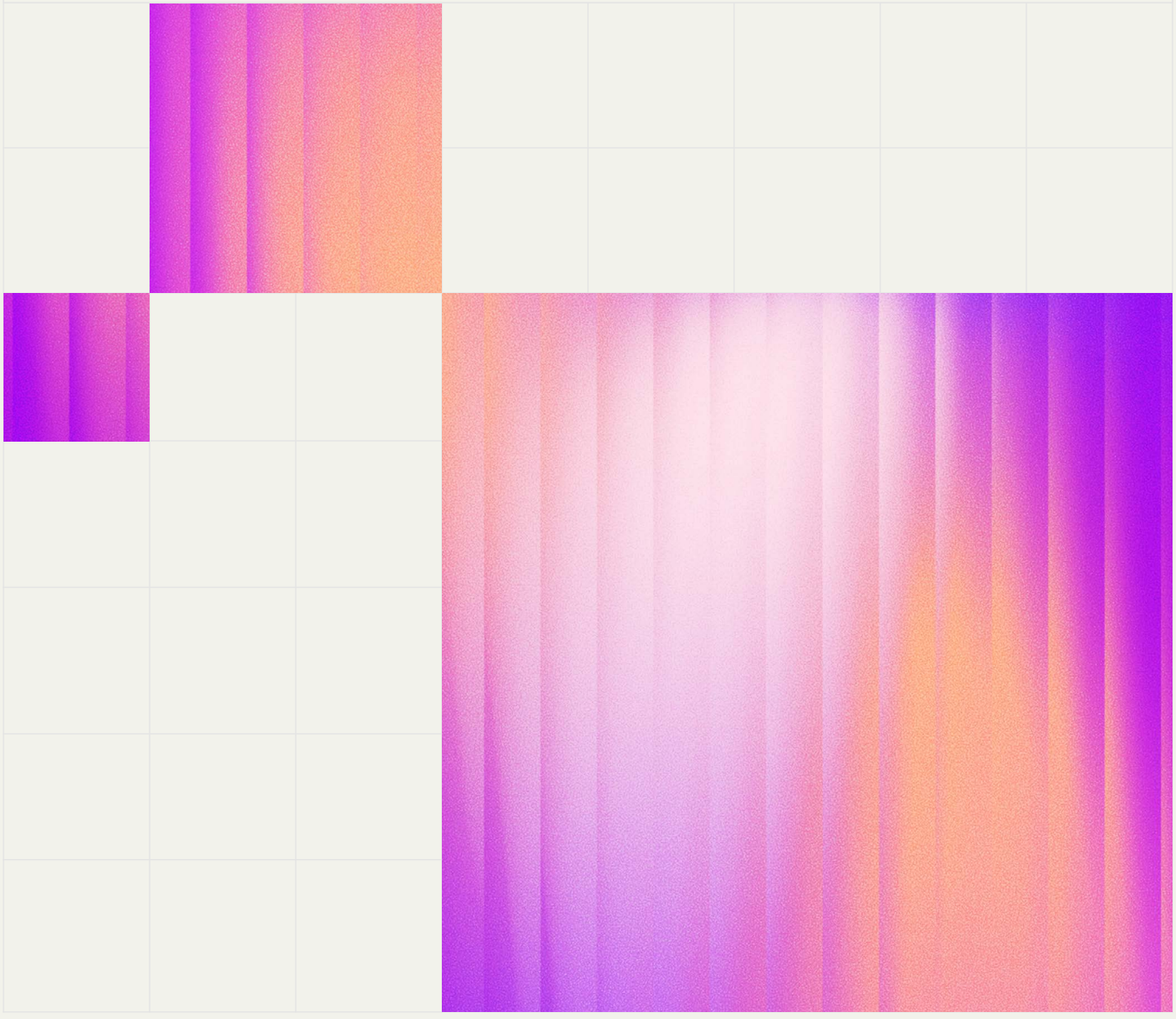
WEKA®

The Memory Wall and the New Tokenomics

How WEKA's Augmented Memory Grid Unlocks the Business Value of Agentic AI



Contents

Click the WEKA logo to return here

Executive Summary

The transition from single-shot generative AI to sophisticated, stateful AI agent swarms is creating an infrastructure crisis. The operational model of these swarms—characterized by long-lived sessions and massive context windows—is colliding with the fundamental memory limitations of current GPU-based inference architectures. This "memory wall," specifically the bottleneck created by the Large Language Model's (LLM) Key-Value (KV) cache, renders large-scale agentic AI economically unviable and performance-constrained. Current state-of-the-art solutions, such as disaggregated prefill-decode architectures, mitigate resource contention but fall short of addressing fundamental GPU constraints, specifically limited KV cache capacity and the inability to universally share cached data across GPUs and nodes. While prefix caching provides some level of optimization, these architectures still face intrinsic limitations, perpetuating high latency—measured as time to first token (TTFT)— and driving up costs due to continuous redundant recomputation of context during the compute-intensive prefill phase.

This report provides a rigorous technical and economic analysis of WEKA's Augmented Memory Grid™. This analysis posits that Augmented Memory Grid is not merely an incremental improvement; it's a new architectural paradigm. By creating a petabyte-scale, persistent "token warehouse" on a high-performance, NVMe-based data platform, Augmented Memory Grid externalizes the KV cache from scarce GPU memory. In our labs, we've leveraged NVIDIA Magnum IO GPUDirect Storage (GDS) and RDMA over high-bandwidth compute fabrics, enabling Augmented Memory Grid to deliver KV cache directly into GPU memory at near-memory speeds of ~300GB/s per host—effectively establishing a powerful, expansive new memory tier optimized specifically for AI workloads.

Our analysis validates that Augmented Memory Grid directly addresses the infrastructure crisis with profound business implications across the entire AI value chain:

- **For GPU Providers**: Augmented Memory Grid enables an estimated 80% reduction in GPUs allocated to prefill workloads, allowing for significantly higher tenant density, GPU oversubscription, and increased revenue per kilowatt-hour.
- **For Model Providers**: Augmented Memory Grid unlocks the full potential of cached token pricing, delivering 75-90% cost reductions on input tokens by transforming ephemeral sessions into persistent ones. This fundamentally alters unit economics, enabling profitable long-context services and new, high-margin Service Level Agreement (SLA)-based offerings.
- **For AI Agent Independent Software Vendors (ISVs)**: Augmented Memory Grid removes the primary performance and cost throttles imposed by model APIs, enabling the development of more powerful, higher-quality agentic applications with 10-100x more token processing capacity for the same cost.

Overall Impact: Augmented Memory Grid shatters the existing cost curve for inference, reducing TTFT by up to 41x and lowering the total system cost of token throughput by up to 24%, making the future of scalable, persistent-context AI economically feasible.

The Dawn of Agentic AI and the Imminent Infrastructure Crisis

1.1 The Paradigm Shift: From Generative AI to Agentic Swarms

The field of artificial intelligence (AI) is undergoing a significant evolution, moving beyond the single-turn, stateless queries that characterized early generative AI (e.g., “write a poem about the sea”) toward multi-turn, stateful, and goal-oriented tasks performed by autonomous AI agents and swarms (e.g., deploy a swarm of agents to analyze procurement inefficiencies, recommend vendor changes, and simulate cost savings). An AI agent swarm is a system where multiple specialized AI agents collaborate to achieve a complex objective that would be difficult or impossible for a single agent to handle.[9] This paradigm mirrors a team of human specialists, where tasks are decomposed and distributed among agents with specific expertise, such as planning, coding, data analysis, or user interaction.

The viability of these swarms hinges on their ability to maintain a persistent, shared context—a long-term memory of the overarching goal, previous actions, environmental feedback, and intermediate results. This context is not static; it grows with every interaction, forming a detailed history of the task. Real-world applications in software development, where agents collaboratively write, debug, and test code, or in supply chain management, where agents monitor inventory, predict demand, and optimize logistics in real-time, all depend on these long-lived, high-context sessions. The fundamental value proposition of agentic AI is directly proportional to the richness and persistence of this shared context.

1.2 The Anatomy of Large Language Model (LLM) Inference: A Tale of Two Phases

To understand the infrastructure challenge posed by agentic AI, it is essential to first dissect the mechanics of LLM inference, which is universally divided into two distinct phases:

- **Prefill Phase**: This is the initial stage where the model processes the entire input prompt in parallel. It is a compute-intensive operation that can fully saturate the processing power of a GPU, involving large-scale matrix operations. The primary output of this phase is not a generated token, but a populated data structure known as the KV cache. The duration of the prefill phase directly determines the **TTFT**, a critical latency metric that heavily influences the perceived responsiveness of an AI application (e.g., the delay between hitting "Enter" on a prompt and seeing the first character of the response appear on screen).
- **Decode Phase**: Following prefill, the model enters the auto-regressive decode phase, generating the response one token at a time. For each new token, the model must reference the system prompt—which defines the fundamental behavior, persona, and constraints of the AI—as well as the initial user prompt, associated context, and any additional prompts or context incorporated in previous interactions. This is achieved by reading the KV cache created during the prefill phase. This phase is not limited by compute power, but by memory bandwidth, as the speed of generating each token is constrained by how quickly the massive KV cache can be read from memory. The performance of this phase is measured by **time per output token (TPOT) or its inverse, tokens per second (TPS), and inter-token latency (ITL).**

Crucially, the effectiveness of prefill and decode phases in aggregated inference architectures is also significantly influenced by continuous batching. When multiple users or prompts are aggregated into the same batch, large prefill tasks intermixed with decode operations can create substantial spikes in ITL. These spikes occur because generating the KV cache for a new prompt during batching temporarily disrupts token generation for ongoing sessions. Consequently, KV cache hit rates are pivotal, impacting not only TTFT but also the sustained decode performance measured by TPOT/ITL.

The table below summarizes these critical performance metrics and their relevance to different AI applications:

Metric	Definition	Phase	Impact on User Experience
Time to first token (TTFT)	Time elapsed from request submission to the generation of the first output token.	Prefill	Determines the initial responsiveness. High TTFT leads to a perception of lag, critical for interactive applications like chatbots and agents.
Time per output token (TPOT)	Average time taken to generate each subsequent token after the first. Also known as Inter-Token Latency (ITL).	Decode	Affects the fluency and smoothness of the streaming response. Low TPOT is essential for real-time applications to keep pace with human reading speed.
Tokens per second (TPS)	The number of output tokens generated per second during the decode phase. The inverse of TPOT.	Decode	A measure of generation speed and overall system throughput.
Total Latency (E2EL)	Total time from request submission to receipt of the final token. E2EL=TTFT+Token Generation Time.	Both	Represents the complete wait time for the user and is the ultimate measure of performance for non-streaming tasks.

1.3 The KV Cache Bottleneck: AI's Memory Wall

KV cache is the fundamental optimization that makes modern LLMs practical. It is a mechanism that stores the intermediate "key" and "value" attention computations from previous tokens, allowing the model to reuse this information instead of recomputing it for every new token. This changes the computational complexity of attention from scaling quadratically with sequence length, $O(n^2)$, which increases the computation time fourfold, to scaling linearly, $O(n)$. With quadratic scaling, if the sequence length doubles, the computation time increases fourfold ($2^2 = 4$), becoming a bottleneck when dealing with long sequences, like entire documents, long code snippets, or multi-turn conversations. With linear scaling, if the sequence length doubles, the computation time also roughly doubles ($2^1 = 2$). This represents a significant efficiency gain. In essence, the KV cache is the model's working memory.

However, scaling linearly introduces a severe bottleneck. Because the size of the KV cache grows linearly with both the sequence length (the number of tokens in the context) and the batch size (the number of concurrent users), it can lead to excessive memory consumption. For a relatively small 7-billion-parameter model, the KV cache consumes approximately 0.5 MB per token; for a large 176-billion-parameter model, this figure balloons to 4 MB per token. A request with a 100,000-token context window—increasingly common for agentic workflows—could thus require a KV cache of 50 GB or more, just for a single user.

This massive memory footprint collides with the most scarce and expensive resource in an AI server: GPU High-Bandwidth Memory (HBM). HBM is ephemeral by design, lacking the ability to persistently store the large volumes of KV cache data required over extended inference sessions. For context, a state-of-the-art NVIDIA H100 GPU has only 80 GB of HBM, while the newer H200 has 141 GB. A substantial portion of this is already allocated to storing the model's weights, leaving minimal space for the dynamically growing KV caches of multiple concurrent users. This collision between the escalating memory demands of the KV cache and the ephemeral capacity of HBM creates what is known as the "memory wall": the single greatest constraint on LLM inference performance and scalability.

1.4 The Agentic Dilemma: When Swarms Hit the Memory Wall

The operational model of AI agent swarms directly and catastrophically exacerbates the KV cache bottleneck. The very characteristic that makes agents powerful—their reliance on long, persistent context—is fundamentally misaligned with the memory architecture of today's inference systems. This creates a debilitating dilemma:

1. Effective AI agents and swarms require long-term, stateful context to perform complex, multi-step tasks efficiently.
2. This context is stored in the KV cache, which must reside in GPU HBM for fast access during the decode phase.
3. Due to extreme pressure on the limited HBM, the KV cache for any given user session is ephemeral. It must be evicted (usually to a slower, higher capacity memory tier like DRAM) after a short period—often just minutes—to make room for other incoming requests.
4. When an agent in a swarm is ready to perform its next action (which could be seconds or minutes after its last action), it finds that its KV cache has been evicted. The entire session history and context are lost.
5. To proceed, the application must resubmit the entire context to the model, triggering another full, expensive, and high-latency prefill phase to rebuild the KV cache from scratch.

This vicious cycle results in chronically low KV cache hit rates. The outcome is an economically unsustainable model for scaling agentic AI, characterized by high TTFT, immense wasted GPU computation on redundant prefilling, and punishingly high costs from paying premium prices for uncached input tokens.

Evaluating Architectural Responses

2.1 The Rise of Disaggregated Inference

Traditionally, aggregated serving manages prefill and decode on the same GPU. This can lead to inefficiencies—during decode, GPU utilization drops, and batched requests can complicate meeting TTFT and ITL.

Conversely, disaggregated serving separates prefill and decode on different GPUs. This separation provides several advantages:

- **Optimized Resource Use**: GPUs excel at prefill's parallel tasks, while decode nodes can be memory-optimized. By tailoring resources to specific computational needs and leveraging specific techniques, you can reduce infrastructure costs and improve efficiency.
- **Improved Performance Management**: Separate handling ensures consistent performance and better control of ITL, directly enhancing the end-user experience by providing predictably fast and reliable inference.
- **Scalability and Flexibility**: Resources can scale independently to meet specific demands dynamically, allowing organizations to rapidly adapt to changing workloads, whether dealing with sudden traffic spikes or evolving use cases.
- **Enhanced Operational Simplicity**: By clearly delineating roles between prefill and decode, operational teams benefit from simplified resource planning and troubleshooting, leading to faster resolution times and less downtime.

This approach was popularized with DeepSeek's architecture, highlighting the ability to achieve efficient and cost-effective LLM inference, even with models of massive scale. Since then, it's been championed by frameworks like NVIDIA Dynamo and llm-d, highlighting the effectiveness of the solution.

Disaggregated architectures offer substantial flexibility by enabling independent scaling of the prefill and decode phases based on demand. However, there is still a premium on how KV cache is retained and delivered as it grows to support today's contexts and agentic swarms: Because the GPU HBM is ephemeral and capacity is constrained, the inference system must rely on repeated cache recomputation and incur network latencies that undermine TTFT gains.

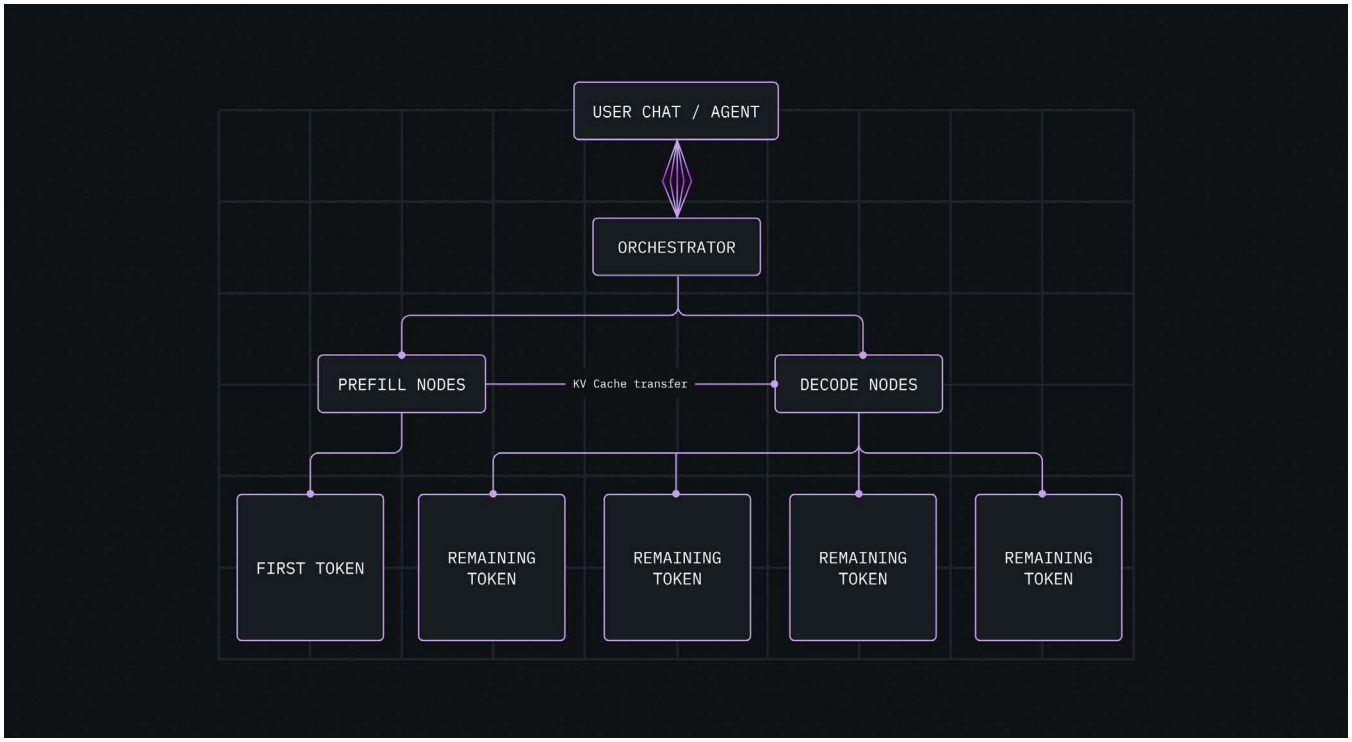


Figure 1: A simplified diagram of a disaggregated inference architecture. User requests are routed to a dedicated prefill cluster that computes the KV cache. This cache is then transferred to a separate, independently scalable decode cluster for token generation.

2.1.2 Prefix Caching: A Foundational Optimization for LLM Inference

To overcome the persistent challenges of LLM inference, particularly the redundant computations and transfer overheads associated with the KV cache, prefix caching has emerged as a foundational optimization. The core idea is simple: To avoid redundant prompt computations, the KV-cache blocks of previously processed requests are cached and subsequently reused when a new request shares the same prefix. This technique represents a natural evolution of KV caching, which was initially developed to capture intermediate states for a single generation; prefix caching extends this by storing these states for reuse across different responses to prompts containing identical prefixes.

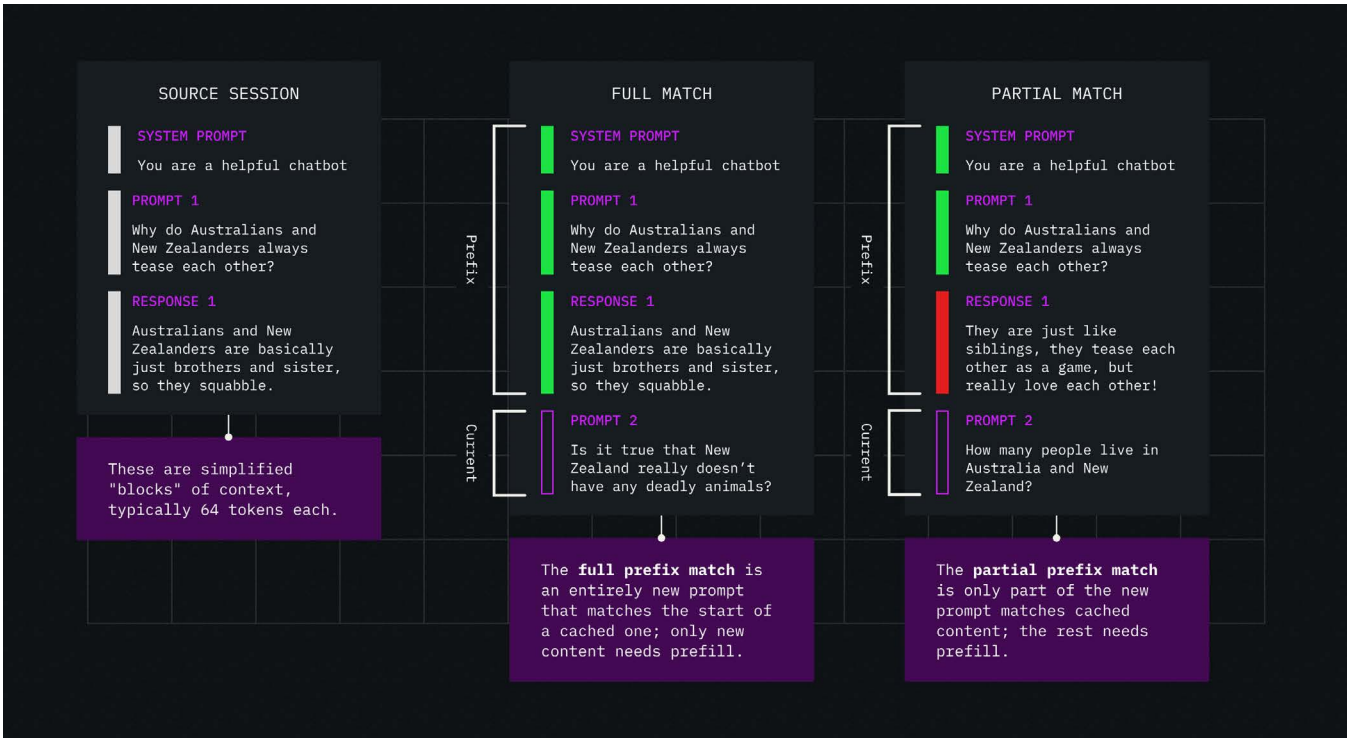


Figure 2: Prefix caching improves LLM inference efficiency by reusing previously computed key-value (KV) cache states. When a new request shares the same initial tokens as an earlier one, the model can skip redundant computations and build only on the cached prefix. A full prefix match reuses the entire cached sequence, while a partial match reuses the overlapping portion, reducing compute overhead. This optimization minimizes latency, avoids wasted GPU cycles, and enables faster, more cost-efficient scaling of inference workloads.

Without caching, these computations would be re-executed every time the model processes a sequence. Prefix caching intervenes by storing these pre-computed KV caches. Implementations often use hash-based approaches to uniquely identify and retrieve cached blocks based on the sequence of tokens and other unique identifiers like LoRA IDs or multi-modality inputs (e.g., image embeddings). When a new request arrives, the system checks if its prefix matches an existing cached entry. If a match is found, the prefill phase, which is a sequence of expensive matrix computations, can be almost entirely bypassed.

The impact of prefix caching is profound across several critical metrics:

- **Reduced Prefill Computations**: By reusing cached KV-cache blocks, the computationally intensive prefill phase is significantly reduced or entirely avoided for common prefixes. This is particularly beneficial for agentic workflows where a shared context or system prompt forms a consistent prefix across multiple turns or agents.
- **Improved TTFT**: As the prefill phase directly determines TTFT, bypassing it through prefix caching drastically reduces the initial latency, leading to a perception of immediate responsiveness for interactive applications.
- **Substantial Cost Savings**: Prefix caching can reduce LLM inference costs by transforming expensive "uncached" input token processing into significantly cheaper "cached" operations.

Many public LLM providers leverage prompt caching to lower inference costs. Furthermore, most open-source LLM inference frameworks, such as vLLM, NVIDIA's TensorRT-LLM, and SGLang, feature automatic prefix caching. This widespread adoption underscores its effectiveness in not altering model outputs while providing significant performance and cost benefits.

Prefix caching, while a powerful optimization for LLM inference, faces significant challenges due to the limitations of current GPU memory architectures. The KV cache, which stores the pre-computed attention states necessary for prefix caching, must reside in the GPU's High-Bandwidth Memory (HBM) for fast access during the decode phase. However, HBM is a scarce and expensive resource. The size of the KV cache grows linearly with the number of tokens and concurrent users, quickly consuming the finite HBM capacity of modern GPUs—meaning the KV cache for any given user session is often ephemeral and must be evicted to make room for new requests. As a result, when an AI agent needs its context again, the pre-computed cache for that prefix is gone, forcing a full and costly re-computation, which leads to chronically low KV cache hit rates and undermines the very purpose of prefix caching.

This implies that the performance benefits are only truly maximized when supported by sophisticated cache management and persistent infrastructure. Moreover, the economic benefits of prefix caching are directly tied to the ability to achieve high cache hit rates. This transforms a technical optimization into a fundamental shift in unit economics for LLM inference. Reduced computation means lower resource utilization, which directly translates to lower operational costs. This is not just about saving money; it fundamentally alters the profitability of long-context services and enables new business models, making previously unviable agentic applications feasible.

2.1.3 Synergies: Prefix Caching, Disaggregation, and Persistent Memory

Prefix caching enhances the effectiveness of disaggregated architectures, particularly when combined with persistent, high-speed solutions that solve the memory bottlenecks for prefill and decode:

- **Complementary Roles with Disaggregation**: Prefix caching directly addresses the "Hidden Bottleneck" of KV cache transfer in disaggregated inference systems. By reducing or eliminating the need for redundant prefill computations, prefix caching inherently minimizes the volume of KV cache data that needs to be transferred between the prefill and decode clusters. If a request's prefix is cached, the prefill cluster's role is diminished, making disaggregation a more viable and efficient architectural choice by mitigating a primary bottleneck.
- **The Critical Role of Persistent Storage**: While prefix caching offers benefits, its effectiveness is often limited by the ephemeral nature of GPU HBM. KV caches are frequently evicted after short periods—often just minutes—to make room for new requests, leading to chronically low cache hit rates (e.g., 50-70% for AI agents). This continuous eviction necessitates resubmitting the entire context and triggering another full, expensive prefill phase to rebuild the KV cache from scratch. This is where solutions like WEKA's Augmented Memory Grid become indispensable for maximizing the benefits of prefix caching.

WEKA Augmented Memory Grid: A New Architectural Paradigm

3.1 Introducing the Token Warehouse™: A Persistent Memory Tier for AI

WEKA's Augmented Memory Grid reframes the problem not as one of faster storage, but of memory extension. Augmented Memory Grid creates a new, persistent, and massively scalable memory tier designed specifically for the demands of AI inference workloads.

The core of this paradigm is the token warehouse: a petabyte-scale repository for KV caches, built upon NeuralMesh™ by WEKA®. This architecture externalizes the KV cache from the confines of the GPU cluster, breaking the dependency on ephemeral HBM and system DRAM for context persistence. By moving the KV cache to a persistent layer built on high-performance NVMe flash, Augmented Memory Grid expands the effective memory capacity available for inference by three orders of magnitude, from the single-digit terabytes of collective DRAM in a server rack to petabytes of capacity on the NeuralMesh system. This is a 1000x improvement for the KV cache size.

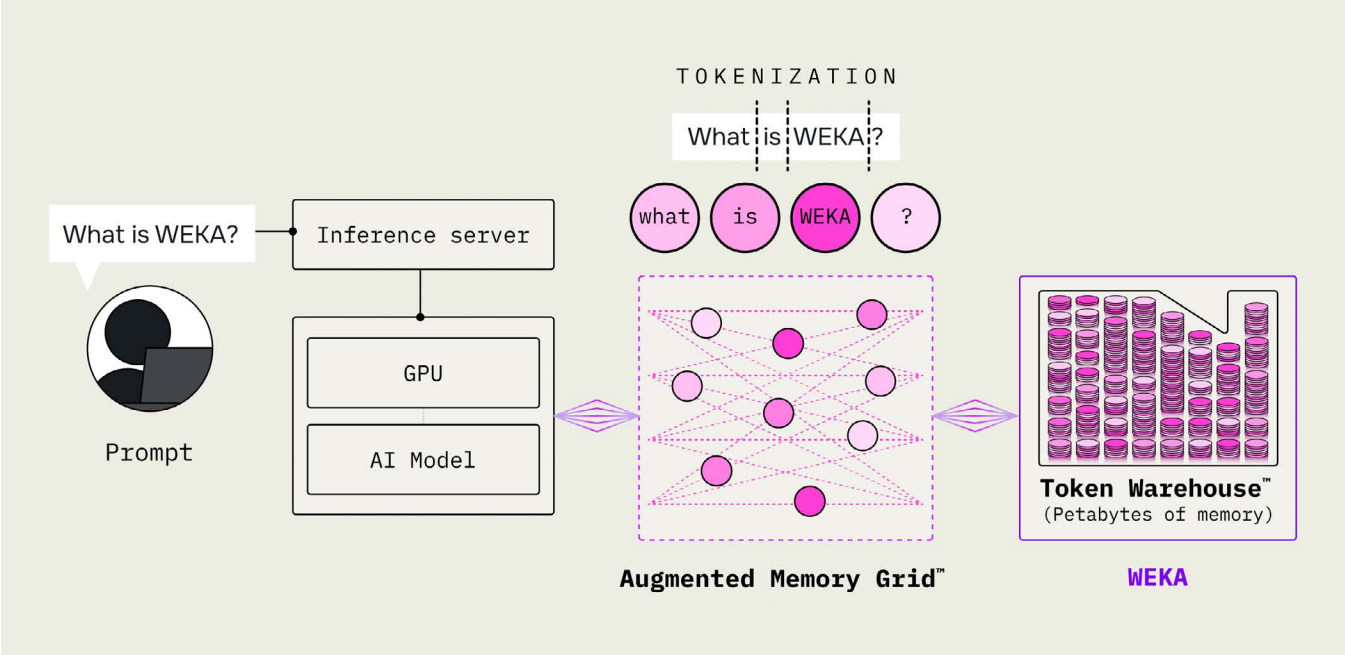


Figure 3: The token warehouse externalizes KV caches from GPU memory into a persistent, petabyte-scale layer on NeuralMesh by WEKA. By moving cached token states from volatile HBM and DRAM into the Augmented Memory Grid, inference systems gain a massively expanded memory tier—scaling from terabytes to petabytes. This architecture delivers up to 1000x more effective cache capacity, reducing dependency on limited GPU memory and enabling higher throughput, lower latency, and more efficient scaling of AI inference workloads.

3.2 Technical Deep Dive: The Mechanics of Microsecond-Latency Memory Augmentation

The viability of the token warehouse concept depends entirely on the ability to move KV cache data between this new persistent tier and the GPU's HBM at speeds that do not leave the GPU waiting. WEKA Augmented Memory Grid achieves this through a combination of software-defined infrastructure and tight integration with NVIDIA technologies:

- **Foundation**: The system is built on NeuralMesh, a software-defined, parallel file system designed for high-performance computing. NeuralMesh bypasses the operating system kernel for both storage and networking, eliminating resource contention and minimizing latency. It also distributes data across a cluster of NVMe SSDs to deliver massive parallel throughput.
- **GPU-Embedded Storage Fabric**: Augmented Memory Grid works with capabilities like NeuralMesh Axon, which embeds NeuralMesh directly inside a GPU server by harnessing the local NVMe, spare CPU cores, and existing network infrastructure. This reduces the required rack space, power, and cooling requirements in on-premises data centers, lowering infrastructure costs and complexity.
- **High-Speed Data Path**: Critically, Augmented Memory Grid can leverage the high-bandwidth East-West compute fabric (e.g., NVIDIA Spectrum-X or InfiniBand), which is the same network GPUs use for high-speed internode communication during training.
- **NVIDIA GPUDirect Storage (GDS) Integration**: Augmented Memory Grid utilizes GDS to establish a direct data path between the NVMe-based token warehouse and the GPU's HBM. As illustrated in the diagram below, this technology enables the GPU's Direct Memory Access (DMA) engine to pull data directly from the WEKA platform, completely bypassing the host CPU and its system DRAM. This zero-copy transfer eliminates multiple sources of latency, including CPU interrupts, context switching, and unnecessary data copies in system memory.

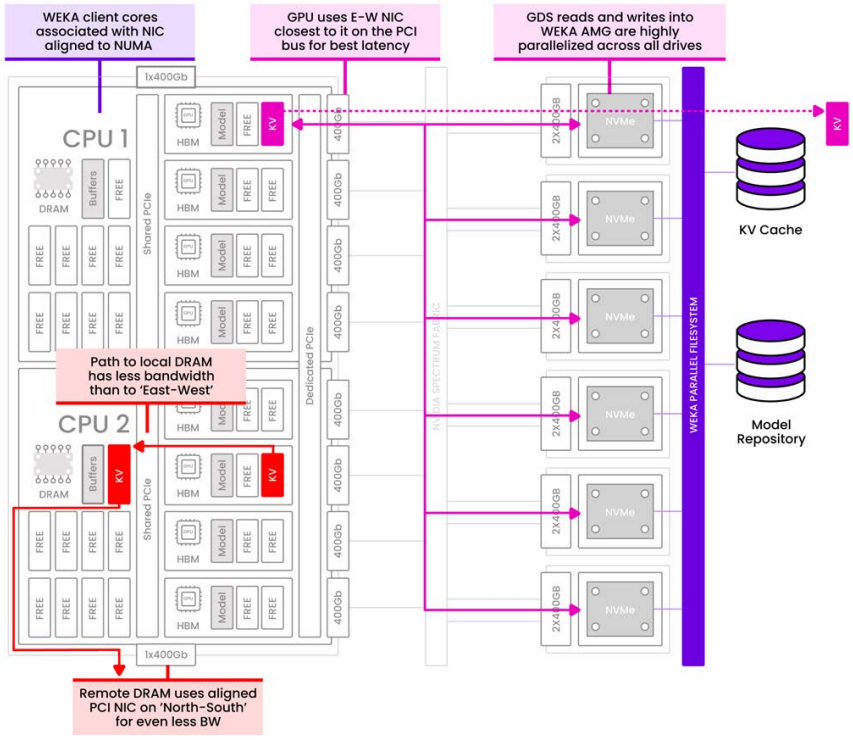


Figure 4: The WEKA Augmented Memory Grid architecture leverages the high-bandwidth East-West compute fabric and NVIDIA GPUDirect Storage (GDS) to create a direct, zero-copy data path between the NVMe-based token warehouse and GPU HBM. This bypasses CPU and system DRAM bottlenecks, enabling data transfer at microsecond latencies.

This unique architecture allows data to be retrieved from the persistent token warehouse and loaded into GPU memory with microsecond-scale latency at an aggregate bandwidth of up to 300 GB/s per host. This performance is sufficient to feed the GPUs without stalling them, effectively solving the KV cache transfer problem and making the token warehouse a viable extension of the GPU's own memory.

3.3 Performance Validation: Deconstructing the 41x TTFT Gain

Augmented Memory Grid can drive a **41x reduction in TTFT** by eliminating prefill computation entirely in cases of cache hit.

The process unfolds as follows, based on a demonstration involving a 105,000-token context where a user submits a request with a large, persistent context (e.g., an entire book or a long history project).

Scenario A Without Augmented Memory Grid	Scenario B With Augmented Memory Grid
The inference system has no persistent memory of this context. It must perform a full prefill computation on all 105,000 tokens to build the KV cache.	The system first queries the token warehouse to see if a KV cache for this exact 105,000-token context already exists. Upon finding a match (a cache hit), KV cache uses GDS to load the pre-computed KV cache directly from NeuralMesh into GPU HBM.
Result In WEKA's test, the TTFT process took **23.97 seconds**.	**Result** In the test, the TTFT process took only **0.58 seconds**. The computationally expensive prefill phase is completely bypassed. The system can immediately begin the decode phase.

The resulting 41x performance improvement (23.97s÷0.58s≈41.3) is the ratio of the time required to compute the prefill versus the time required to load the pre-computed KV cache. This highlights the transformative potential for workloads with high context reuse, which is the defining characteristic of AI agent swarms.

Deconstructing the Business Value Across the AI Infrastructure Stack

The architectural innovations of WEKA's Augmented Memory Grid translate into direct and quantifiable business value for every stakeholder in the AI ecosystem. This section analyzes the impact on each layer of the "SaaS AI Agents Cloud Bill of Materials."

4.1 For GPU Providers (e.g., CoreWeave, Nebius, Ori Cloud)

GPU providers, whose business models are predicated on selling GPU-hours, face a constant challenge of maximizing the utilization of their expensive hardware assets. Idle or underutilized GPUs represent lost revenue:

- **GPU Reallocation and Efficiency**: In a disaggregated architecture, a significant portion of the GPU fleet must be dedicated to handling the compute-intensive prefill phase. By making redundant prefill operations obsolete, Augmented Memory Grid allows these GPUs to be freed up. The claim that this could liberate up to **80% of GPUs** in a prefill cluster is plausible; in a system with a 4:1 ratio of prefill-to-decode resources, eliminating the need for prefill frees up four-fifths of the hardware. These GPUs can be reallocated to serve more revenue-generating decode tasks, dramatically increasing the overall efficiency of the cluster.
- **Increased Tenant Density and Oversubscription**: With the massive KV cache offloaded to the WEKA token warehouse, the memory pressure on individual GPUs is substantially relieved. This enables providers to safely increase tenant density, hosting more user sessions on the same physical hardware. It also unlocks the ability to oversubscribe GPU memory, a common practice in cloud computing that is difficult with memory-constrained AI workloads. This leads directly to higher revenue per GPU and per kilowatt-hour, with claims of **10x-100x higher token throughput per GPU**.
- **Simplified Operations**: The token warehouse acts as a shared, centralized state store for all sessions. This eliminates the need for complex and fragile session-stickiness logic, where load balancers must try to route a user back to the specific GPU that holds their ephemeral KV cache. With Augmented Memory Grid, any GPU in the cluster can serve any user at any time and still achieve a full cache hit, simplifying operations and improving load balancing and elasticity.

4.2 For Model Providers (e.g., OpenAI, Anthropic, Together.ai)

Model providers face immense costs in serving long-context models, driven primarily by the compute and memory required for the prefill phase. This forces them to create pricing tiers that heavily penalize uncached inputs, creating an economic barrier for developers of stateful applications:

- **Flipping the Economic Model**: Augmented Memory Grid enables model providers to achieve KV cache hit rates approaching 100%. This means nearly all input tokens for ongoing sessions can be processed at the significantly lower "cached" or "prompt caching" price tier. As the following table demonstrates, this is not an incremental saving; it represents a fundamental shift in the cost structure:

Model Provider	Model	Uncached Input Price ($/M tokens)	Cached Input Price ($/M tokens)	Cost Reduction with Cache
OpenAI	GPT-4.1	$2.00	$0.50	**75.0%**
OpenAI	GPT-4.1 mini	$0.40	$0.10	**75.0%**
Anthropic	Claude Opus 4	$15.00	$1.50	**90.0%**
Anthropic	Claude Sonnet 4	$3.00	$0.30	**90.0%**
Anthropic	Claude Haiku 3.5	$0.80	$0.08	**90.0%**

Table based on publicly available API pricing. Data sourced from Open AI[1] and Anthropic[2]

This table provides the core economic argument for Augmented Memory Grid from the model provider's perspective. A technical capability—achieving a high cache hit rate—translates directly into a **75-90% reduction** in the cost of processing input tokens, a major component of their cost of goods sold (COGS).

- **Enabling New Business Models**: By making long-context, stateful inference profitable, Augmented Memory Grid allows providers to move beyond simple pay-per-token models. They can introduce new premium enterprise tiers offering guaranteed context persistence over days or weeks, SLAs for low TTFT, and higher overall throughput, creating new, high-margin revenue streams.

4.3 For AI Agent Swarm ISVs (e.g., Manus, Cursor, Cognition)

Independent Software Vendors (ISVs) building agentic applications are caught in the middle of the infrastructure dilemma. Their products, such as AI-powered coding assistants or autonomous task managers, require long, persistent context to be effective, but they are throttled by the high costs and performance penalties imposed by model provider APIs for long, uncached prompts.

- **Massive Cost Reduction**: ISVs are the direct beneficiaries of the 75-90% input token cost savings enabled by Augmented Memory Grid. This allows them to process **10 to 100 times more tokens for the same cost**, a transformative economic shift that enables them to build more capable, intelligent, and context-aware agents without passing prohibitive costs on to their customers.
- **Unlocking Performance and New Applications**: By virtually eliminating the TTFT penalty associated with re-prefilling context, ISVs can build applications that are dramatically more responsive. This opens the door to a new class of real-time agentic systems for applications like live voice interaction, collaborative video analysis, and interactive coding environments, where low latency is non-negotiable.
- **Sustainable Competitive Advantage**: An ISV building on Augmented Memory Grid-powered infrastructure can offer a product that is simultaneously more powerful (more context, more features) and faster (lower latency) at a fundamentally lower cost base than a competitor using a traditional inference stack. This creates a durable competitive advantage in the rapidly growing market for agentic AI.

4.4 For AI Agent Users (Developers & Enterprises)

Ultimately, the benefits of the underlying infrastructure accrue to the end-user, whether a developer using a coding agent or an enterprise deploying a swarm for business process automation. Users experience the pain of infrastructure limitations directly as slow response times, frustrating context limits, and high subscription costs for powerful features.

- **Superior User Experience**: The technical benefits of Augmented Memory Grid translate into a fluid, real-time user experience. The long pauses associated with the AI “thinking” (i.e., re-prefilling) disappear, leading to seamless interactions even when recalling information from much earlier in a long conversation or project.
- **Effectively Unbounded Context**: Users are no longer constrained by artificial context window limits imposed by cost and performance. A session can persist for days or weeks, with the AI agent retaining a perfect, complete memory of the entire interaction. This leads to significantly higher-quality and more contextually relevant outcomes.
- **Enabling the Future of Work**: This architecture unlocks the full potential of next-generation applications. It makes it feasible to build and use AI agents that can truly function as long-term collaborators, possessing a persistent understanding of complex projects and workflows over extended periods.

The Economic Imperative: A Quantitative Model of the WEKA Augmented Memory Grid Advantage

To crystallize the financial impact of WEKA's Augmented Memory Grid, this section presents a quantitative model comparing the total cost of ownership (TCO) for a representative AI agent swarm workload running on a traditional disaggregated stack vs. an Augmented Memory Grid-powered stack.

5.1 Modeling an AI Agent Swarm Workload

The model is based on a software development agent swarm tasked with a complex coding project over a standard workday:

- **Workload Parameters:**
  - **Swarm Composition**: 5 specialized agents (e.g., Planner, Coder, Debugger, Tester, Reviewer).
  - **Session Duration**: 8 hours.
  - **Interaction Frequency**: 20 turns (LLM calls) per agent per hour.
  - **Context Size (Input)**: Average of 50,000 tokens per turn.
  - **Generated Size (Output)**: Average of 2,000 tokens per turn.
- **Total Tokens per Session:**
  - **Total Input Tokens**: 5 agents * 20 turns/hr * 8 hrs * 50,000 tokens/turn = 40,000,000 tokens.
  - **Total Output Tokens**: 5 agents * 20 turns/hr * 8 hrs * 2,000 tokens/turn = 1,600,000 tokens.

5.2 Comparative Tokenomics Analysis (TCO per Session)

This analysis compares two scenarios using pricing for a capable model like OpenAI's GPT-4.1 ($2.00/M uncached input, $0.50/M cached input, $8.00/M output)1 and estimated GPU infrastructure costs:

- **Scenario A**: Traditional Disaggregated Stack:
  Assumes a **65% KV Cache hit rate**, a midpoint based on reports from agent developers.
- **Scenario B**: Augmented Memory Grid-Powered Stack:
  Assumes a **99% KV Cache hit rate**, as enabled by a persistent token warehouse.

Cost Component	Calculation Details (per Session)	Scenario A: Traditional Stack	Scenario B: Augmented Memory Grid-Powered Stack	% Savings with Augmented Memory Grid
Input Token Cost	(Total Input Tokens * % Uncached * Price_Uncached) + (Total Input Tokens * % Cached * Price_Cached)	$38.00	$20.60	**45.8%**
Output Token Cost	Total Output Tokens * Price_Output	$12.80	$12.80	**0.0%**
GPU Infrastructure Cost (Prefill)	Assumes a 4:1 prefill:decode GPU ratio. Prefill GPUs are active 35% of the time (1 - 65% hit rate)	$17.92	$0.51	**97.2%**
GPU Infrastructure Cost (Decode)	Decode GPUs are active for the full session.	$12.80	$12.80	**0.0%**
Total Session Cost	**Sum of all costs**	**$81.52**	**$51.71**	**36.6%**
Cost per 1M Output Tokens	Total Session Cost / 1.6M Output Tokens	**$50.95**	**$32.32**	**36.6%**

Note: GPU costs are illustrative, based on an estimated $2.00/hr/GPU. Prefill cost in Scenario B is near-zero, but not absolute, reflecting the 1% of requests that may still require prefill. The analysis demonstrates a significant overall TCO reduction, aligning with claims of up to 24% lower cost for total token throughput, with this model showing even higher savings due to the workload's high input-to-output ratio.

5.3

The New Pareto Frontier for Inference

The economic model reveals a crucial dynamic: With traditional AI infrastructure, the cost of an agentic session scales punishingly with its duration and context size due to the compounding cost of re-prefilling. Augmented Memory Grid fundamentally alters this relationship by decoupling cost from session length—**creating a 10X improvement, even as session length and context increase**. This can be visualized as a shift in the Pareto frontier of inference, which plots cost against context persistence.

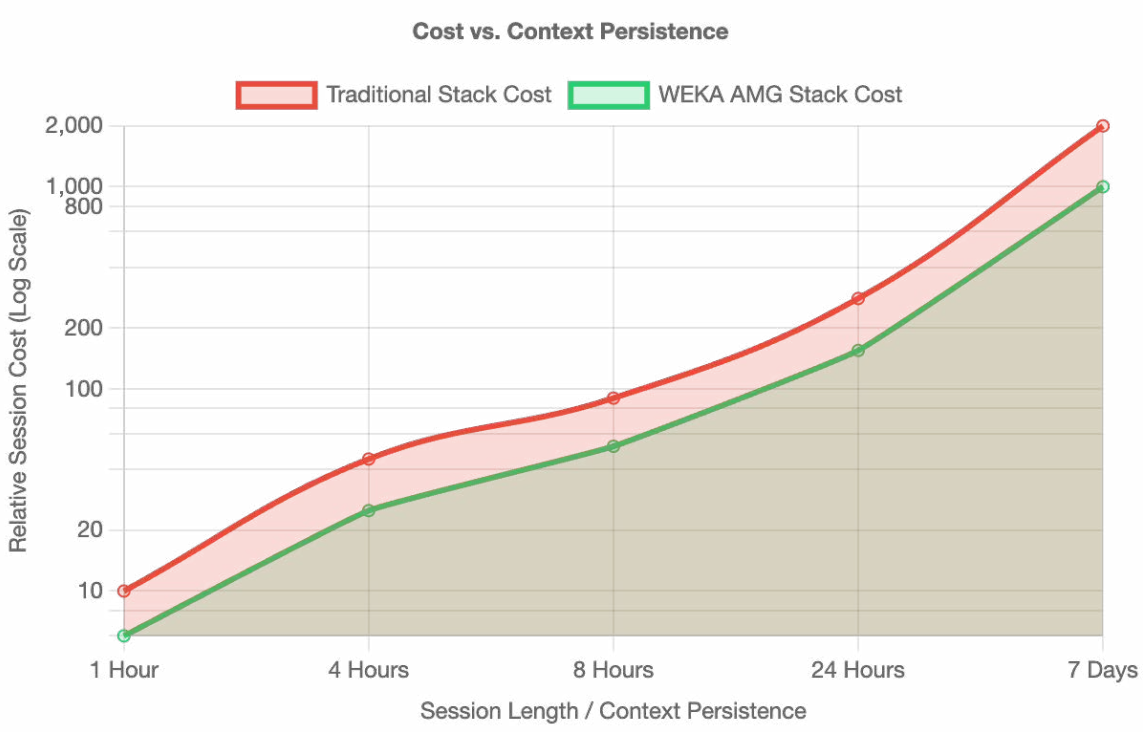


Figure 5: A conceptual chart illustrating how WEKA Augmented Memory Grid shifts the Pareto frontier for AI inference. The traditional architecture (red curve) shows costs rising exponentially with session length and context size, making large-scale agentic AI economically unviable. The Augmented Memory Grid-powered architecture (green line) decouples cost from session length, creating a nearly flat cost curve that makes persistent, long-context AI sustainable at scale.

This chart illustrates that Augmented Memory Grid does not just make long-context AI incrementally cheaper—it moves it from a niche, prohibitively expensive capability to a mainstream, economically viable one. It makes the future of collaborative, persistent AI possible.

Conclusion: The Future of AI is Persistent and Augmented

6.0

The evolution from simple generative models to complex, collaborative AI agent swarms represents a fundamental shift in the capabilities of artificial intelligence. However, this evolution has created a severe infrastructure crisis. The very foundation of agentic AI—its reliance on vast, persistent context—is incompatible with the ephemeral, memory-constrained nature of current GPU-based inference infrastructure. The "memory wall" created by the LLM's KV cache has become the primary barrier to realizing the full potential of this new AI paradigm.

Efforts to address this issue—even with specialized architectural optimizations—continue to struggle with challenges like limited GPU KV cache capacity, resource contention, and redundant prefill. While these architectures optimize resource allocation, they fail to solve the underlying data gravity problem of the massive KV cache, creating new bottlenecks in network transfer that perpetuate high latency and inefficiency.

WEKA's Augmented Memory Grid provides the necessary architectural leap forward. By creating a token warehouse—a new, persistent memory tier external to the GPU—Augmented Memory Grid resolves the memory wall bottleneck at its source. Its integration with high-speed compute fabrics and NVIDIA GPUDirect Storage addresses the data transfer problem, enabling the delivery of context to GPUs at near-memory speeds.

This innovation fundamentally realigns the unit economics of inference with the demands of agentic AI. It transforms the cost structure for every stakeholder, from the GPU providers who can now achieve unprecedented hardware utilization, to the model providers who can profitably offer long-context services, to the ISVs who are finally free from the economic and performance throttles that have constrained their innovation. For the end user, it unlocks a new generation of powerful, responsive, and truly collaborative AI applications. The future of AI is not just about larger models; it is about persistent, stateful, and context-aware systems. WEKA's Augmented Memory Grid provides the enabling infrastructure not for the AI of today, but for the AI of tomorrow.

Works Cited

1. API Pricing - OpenAI, accessed July 22, 2025: https://openai.com/api/pricing/
2. Pricing - Anthropic, accessed July 21, 2025: https://docs.anthropic.com/en/docs/about-claude/pricing

 WKA436-02 09/25