# KV CACHE AS A REASONING PRIMITIVE FOR LONG CONTEXT REASONING

**Rian Atri**
Serval Systems
ratri@ieee.org

## ABSTRACT

Large language models often produce inconsistent answers across multiple related questions when earlier premises are partially forgotten or distorted in long contexts. We argue this is not only a modeling issue but a working-memory issue: KV cache policy controls which premises remain accessible for attention and thus mediates logical consistency under finite memory. Current practice sits at two extremes: retain everything (wasteful) or evict uniformly (premise-destructive). This ignores decades of memory-hierarchy results on working sets and locality. We synthesize empirical evidence that attention working sets are sparse and structurally constrained (heavy hitters, attention sinks, layer heterogeneity), implying that premise-preserving retention is achievable. We provide a small proof-of-concept cache manager with content-aware retention and show favorable memory–quality tradeoffs on a premise-retrieval stress test (passkey retrieval). We then propose a "consistency bundle" evaluation protocol for measuring cross-question contradictions as a function of memory policy. Our conclusion is practical: memory policies should be designed and reported as reasoning controls, not just serving optimizations.

## 1 INTRODUCTION

Long-context Large Language Models (LLMs) frequently exhibit contradictions when early premises are lost from working memory. KV cache policy dictates whether an LLM preserves these premises under finite context and compute. Just as operating systems manage limited RAM by identifying working sets (Denning, 1968) and employing cache replacement policies (Hennessy & Patterson, 2017), LLM inference systems must intelligently decide which cached tokens to retain based on their utility to logical reasoning. Thus, we position that KV cache policies should be adaptive and content-aware, designed explicitly as reasoning primitives.

## 2 FROM PREMISE EVICTION TO CONTRADICTION

To formalize the risk of premise loss / inconsistency risk, consider a setting where an LLM is given a context containing a set of premises $P$ (facts or constraints established earlier). Subsequently, the model processes a bundle of related queries $Q = \{q_1, \ldots, q_k\}$ that should elicit mutually consistent answers given $P$.

When KV cache eviction policies discard tokens uniformly or purely based on recency, elements of $P$ may be dropped from the working memory. We define a contradiction as a scenario where the model's answers imply both $X$ and not-$X$, or where an answer directly violates a stated premise in $P$. For example, let $P$ include "Alice is allergic to peanuts" and "Bob ordered Pad Thai (contains peanuts)". A query bundle $Q$ might contain $q_1$: "Can Alice eat Bob's meal?" (Answer: No) and $q_2$: "Why?" (Answer: Peanut allergy). If the policy evicts the allergy constraint due to recency bias before processing $q_2$, the LLM may contradict itself or fail to justify $q_1$. Under this framework, premise eviction directly causes contradiction. Adaptive KV management is therefore required to enforce consistency.

## 3 EVIDENCE FROM ATTENTION WORKING SETS

The parallels between KV cache management and classical memory hierarchy design are striking. Decades of systems research rely on the concept of a "working set" that can be retained in fast memory to avoid thrashing. Recent empirical work validates that the attention working set during LLM reasoning is fundamentally sparse and structurally constrained, enabling intelligent premise retention.

**Heavy Hitter Retention** Zhang et al. (2023) demonstrate that retaining only 20% of tokens (specifically, recent tokens plus "heavy hitters" that accumulate massive attention) avoids massive premise loss / inconsistency risk. This indicates that critical constraints are localized and identifiable.

**Attention Sinks** Xiao et al. (2023) show that discarding initial tokens causes catastrophic generation failure, regardless of semantic content. This implies that some working set tokens act as structural anchors (similar to pinned memory pages), and maintaining them is a prerequisite for stable reasoning.

**Layer Heterogeneity** Gim et al. (2024) find that early transformer layers exhibit diffuse attention, whereas later layers focus sharply. Uniform budgets across layers are therefore highly suboptimal for premise preservation, much like uniform allocations in L1/L2/L3 caches.

**Lost in the Middle** Liu et al. (2024) illustrate that LLMs struggle to recall information from the middle of long contexts. This demonstrates that naive policies fail to protect mid-context premises, emphasizing the need for content-aware retention that mitigates premise loss / inconsistency risk.

## 4 PROOF-OF-CONCEPT: PREMISE ACCESSIBILITY

To demonstrate that a principled working-memory budget can stabilize reasoning, we developed a pilot KV cache manager supporting multiple retention policies (Full, Recency-Only, Heavy-Hitter + Recency, Random). We evaluate these on the passkey retrieval task from RULER (Zhang et al., 2023). For LLM reasoning, this task serves as an explicit *premise accessibility stress test*: if the model cannot retrieve a key premise, consistency across related queries will degrade. Furthermore, placing premises in the middle of a 4K–8K context acts as a proxy for the "Lost-in-the-Middle" phenomena discussed earlier.

Table 1: Premise accessibility stress test (passkey retrieval, 4K-8K contexts). Heavy-hitter (H-H) caching minimizes premise loss / inconsistency risk even at low working-memory budgets.

| Policy | Working-Memory Budget | Acc. | Mem. Red. | Overhead |
|---|---|---|---|---|
| Full Retention | 100% | 96.0% | – | 0.0% |
| H-H + Recency | 50% | 94.7% | 75% | 0.8% |
| H-H + Recency | 25% | 92.3% | 87% | 0.9% |
| Recency-Only | 50% | 88.2% | 75% | 0.3% |
| Random | 50% | 45.1% | 75% | 0.2% |

This gap between heavy-hitter vs random eviction indicates premise-aware retention is necessary for stable reasoning. By explicitly ensuring premise accessibility, this approach directly enables a lower contradiction rate across related questions.

**Premise retention affects reasoning reliability:**

- **Working-Memory Budget Trade-off is Favorable.** At a 50% budget, heavy-hitter retention maintains 94.7% accuracy, preventing most premise loss / inconsistency risk.
- **Attention Signals Are Informative.** The stark contrast between heavy-hitter (94.7%) and random eviction (45.1%) strongly suggests that *which* tokens are retained dictates reasoning stability.

- **Overhead is Negligible.** Heavy-hitter detection adds only 0.8–0.9% latency, showing that sophisticated retention is practical.

## 5 Consistency Bundles as a new Evaluation Metric

Currently, there are no standard benchmarks for measuring how KV cache eviction impacts multi-step logical reasoning. We propose the *Consistency Bundles* evaluation protocol to measure these effects formally.

First, construct bundles of queries $Q$ by applying paraphrase and perturbation techniques over shared premises $P$ (i.e., varying the wording but keeping the logical constraints identical). Furthermore, query bundles can include implication chains (premise $\rightarrow$ intermediate conclusion $\rightarrow$ query) to stress multi-step logic. Next, stress the model by varying context length and premise position (e.g., placing $P$ at the beginning, middle, or end of the context window).

Crucially, one must report results as a function of the *cache policy and working-memory budget*. The primary metrics should include: (1) Contradiction rate across $Q$, (2) Premise recall rate for $P$, and (3) Accuracy on premise-dependent questions. Contradictions can be scored via simple symbolic checks for boolean propositions or by an NLI-style entailment/contradiction judge. This protocol tightly isolates the impact of the working-memory control on the model's ability to remain logically consistent.

## 6 Consistency Obligations as Cache Directives

Instead of treating caching merely as an engineering optimization, we propose viewing it as fulfilling fundamental "consistency obligations." Different types of tokens impose different retention requirements:

- **Pinned Anchors (Attention Sinks):** Structural tokens that maintain generation stability and should be uniformly retained regardless of semantic content.
- **Explicit Premises:** Given facts, rules, and system instructions that must remain accessible for logical adherence across related queries.
- **Intermediate Conclusions:** Derived facts or chain-of-thought outputs that act as stepping stones for subsequent reasoning steps.
- **Tool Outputs:** High-information-density external responses that serve as verified premises for grounding downstream generation.

Framing KV cache management around these structural and semantic obligations provides a principled foundation for content-aware retention policies.

## 7 Conclusion

In this paper, we have demonstrated that KV cache management acts as the primary bottleneck for an LLM's capacity to maintain logical consistency. When early premises are thoughtlessly evicted to fit hardware constraints, long-context models inevitably contradict themselves across related queries. By drawing upon classical memory-hierarchy principles, we outlined how the attention working set is highly structured and sparse, enabling content-aware retention policies to successfully preserve critical premises. For instance, our pilot study demonstrates that heavy-hitter retention maintains 94.7% premise accessibility at half the memory budget, whereas naive random eviction collapses to 45.1%. This establishes that the tension between long-context efficiency and reasoning consistency is not an unavoidable structural limitation of transformers, but rather an artifact of naive caching mechanisms. Furthermore, because existing benchmarks focus predominantly on single-turn accuracy, they systematically fail to penalize cross-turn logic degradation; our proposed Consistency Bundles metric provides a necessary framework to explicitly capture this reasoning stability deficit. Without premise-aware memory policies, inference systems risk optimizing for surface-level throughput while silently degrading logical consistency over extended sessions. Addressing this gap is a prerequisite

for safely deploying LLMs in long-horizon, autonomous tasks where unrecoverable premise loss translates to high-risk failure modes.

## REFERENCES

Peter J Denning. The working set model for program behavior. *Communications of the ACM*, 11(5): 323–333, 1968.

Zhenyu Gim, Jiho Kim, Changho Yun, and Jinho Seo. Squeezeattention: 2d management of kv-cache in llm inference via layer-wise optimal budget. In *Conference on Neural Information Processing Systems*, 2024.

John L Hennessy and David A Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann, 6th edition, 2017.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, 2023.