ADMITKV: PROVENANCE- AND TYPE-GATED KV TRANSFER FOR EDITED AGENT CONTEXTS

Anonymous authors
Paper under double-blind review

ABSTRACT

An identity-cache miss in an agent trace can denote either a local edit whose token state remains useful or a semantic rewrite that changes the answer; uniform near-match reuse conflates these cases. AdmitKV turns each miss into a block-level transfer decision: immutable provenance authorizes the source, monotone alignment and RoPE repair identify position-correct reusable KV, four deterministic micro-forward probes test computational agreement, and a type-conditioned empirical monitor can reject an otherwise eligible transfer. The production rule admits aligned code and tool state and sends document near-matches through exact lookup or recomputation. With the same exact-block-hash identity stage on SWE-bench Verified at 50% shared-file overlap, adding AdmitKV lowers median latency from 18.4 to 16.8 minutes (8.7%, 95% CI [5.3,11.8]%); relative to fresh execution it lowers latency by 22.9% (21.8→16.8 min, CI [19.6,26.3]%) and peak GPU memory by 30.6% (73.2→50.8 GB). Under a common validation-selected outcome envelope, its held-out point has 0.37% approximate false reuse and 0.7-point p99 event loss, versus 0.66–0.78% and 0.9–1.0 points for five selective-KV adaptations. Across the admitted-region perturbation grid, including the hardest 25% probe-aware setting, code/tool p99 loss is at most 0.9 point; the frozen thresholds also reduce p50 latency from 14.6 to 12.7 minutes on Natural Questions Open and from 8.7 to 7.3 minutes on GAIA while their paired quality intervals remain above the specified margins.

1 INTRODUCTION

After exact lookup fails, an agent server can face three superficially similar edits with different consequences. A function may change after a long reusable prologue; a tool record may preserve its schema while updating one value; a fluent document paraphrase may reverse the relation needed for the answer. The serving decision is therefore not simply whether two blocks are similar, but whether a particular source is authorized, which token-layer work remains usable, and how costly an incorrect transfer is for that block type.

This decision matters at end-to-end scale. At 50% shared-file overlap in our SWE-bench Verified stream, fresh execution takes 21.8 minutes, exact block hashing takes 18.4, and matched request-level radix reuse takes 19.2. Exact hashing captures identical blocks but leaves edited functions and versioned tool state untouched. Adding selective transfer after the same exact-hash stage reaches 16.8 minutes, isolating an 8.7% latency reduction from post-identity reuse rather than from a different identity index.

AdmitKV realizes this selective transfer as a fail-to-fresh decision path. Type and immutable lineage metadata first restrict candidate sources. Deterministic anchors then expose a conservative aligned prefix, and source keys are rephased to target RoPE positions. Four metadata-derived, teacher-forced micro-forward probes compare the transferred branch with a fresh branch through fixed intermediate-layer logit projections. Admission additionally requires an open empirical-risk stratum; missing metadata, invalid alignment, probe disagreement, or exhausted monitoring evidence all invoke the same recomputation path.

The measurements connect each mechanism to an observable consequence. On SWE 50%, AdmitKV reduces median latency by 22.9% and peak GPU memory by 30.6% relative to fresh execution; paired tests reject equal latency against fresh computation ($p = 0.004$) and the complete matched-radix policy ($p = 0.007$). At validation-selected feasible points, AdmitKV records 0.37% approximate

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.

false reuse and 0.7-point empirical p99 loss, compared with 0.66–0.78% and 0.9–1.0 points for CacheBlend, KVShare, SemShareKV, A^3, and ProphetKV adaptations. Component removals locate distinct roles for provenance, probes, type conditioning, and monitoring: on GAIA they raise approximate false reuse from 0.65% to 1.18–2.83%, while provenance alone gives up latency by rejecting most near-matches. The same frozen thresholds preserve the latency ordering on Natural Questions Open, GAIA, Llama-3.1-8B, and Qwen-2.5-32B.

Position-independent systems such as EPIC and Cache-Craft reuse immutable chunks under changed placement or surrounding context (Hu et al., 2025; Agarwal et al., 2025); selective-KV methods reconstruct contexts by choosing cached tokens or recomputation. AdmitKV addresses the content-edited case and makes source authorization, block-dependent rejection, and completed-request event loss part of that reconstruction decision.

The paper establishes three results:

1. a utility-under-risk formulation for content-edited KV transfer, with source authorization, event-attributed counterfactual loss, and block-conditioned false-admission and tail criteria;
2. a concrete reject-option mechanism combining provenance-restricted retrieval, conservative alignment and RoPE repair, deterministic probe evidence, type-conditioned empirical vetoes, and one recomputation fallback; and
3. a parity-controlled evaluation that isolates the gain after exact block hashing, risk-matches five selective-KV adaptations, probes component necessity and admitted-region stress, and transfers the frozen rule across three workloads and three checkpoints.

2 AdmitKV: Selective Transfer After Identity Misses

2.1 Transfer objective

Let b be a target block, s a cached source block, $t(b) \in \{\text{code}, \text{doc}, \text{tool}\}$ its deterministic type, and $g(b, s)$ its deployed stratum, formed by type and compatible provenance lineage; $t(g)$ denotes the type component. Let $P(b, s) \in \{0, 1\}$ be the provenance-compatibility predicate and $C(b, s)$ the recomputation time saved by inheritance. Relative to a matched fresh-compute counterfactual, $F(b, s)$ indicates a false admission and $S(b, s)$ is its continuous benchmark-native severity. The action $a \in \{\text{admit}, \text{recompute}\}$ follows the configured identity stage: exact block hashing in the primary policy, or radix-prefix service followed by block hashing in the composition experiment. For each enabled stratum, AdmitKV targets

$$\begin{aligned} \max_{\pi} \quad & \mathbb{E}_{\pi}[C(b, s)\mathbf{1}\{a = \text{admit}\}] \\ \text{s.t.} \quad & \Pr(F = 1 \mid g, \text{admit}) \le \rho_{t(g)}, \qquad Q_{0.99}(S \mid g, \text{admit}) \le \sigma_{t(g)}. \end{aligned} \tag{1}$$

The configured bars are $(\rho_t, \sigma_t) = (1\%, 1)$ benchmark-native point for code and tool blocks. Documents use the exact-only path in the primary policy. Equation 1 defines the empirical operating target rather than a distribution-free guarantee: candidate threshold tuples are filtered by validation feasibility, ranked by saved prefill time, and then subject to a rolling upper-estimate veto in each active stratum. Exact fallback, type enablement, tenant partitioning, and deterministic provenance compatibility are enforced mechanically; threshold selection and monitoring use disjoint empirical data, following the separation principle of risk-controlling calibration (Angelopoulos et al., 2024; Bates et al., 2021).

2.2 Block construction and reject option

Figure 1 summarizes the fixed decision order. Every approximate transfer must pass type enablement, deterministic provenance compatibility, target/source alignment, cosine similarity τ_t, maximum probe divergence ϵ_t, and the current monitoring veto; a failed or undefined check recomputes. Code blocks are AST functions or classes, document blocks are header-delimited sections subdivided at 512 tokens, and tool blocks are schema-rooted JSON objects, timestamped log stanzas, or table chunks. Each source stores a BLAKE3 content hash, embedding, block type, selected-layer KV tensors, and an immutable provenance tuple. Stable security-domain and object-lineage fields—tenant and repository identity for code, tenant and collection identity for documents, and tenant, tool identity, and execution

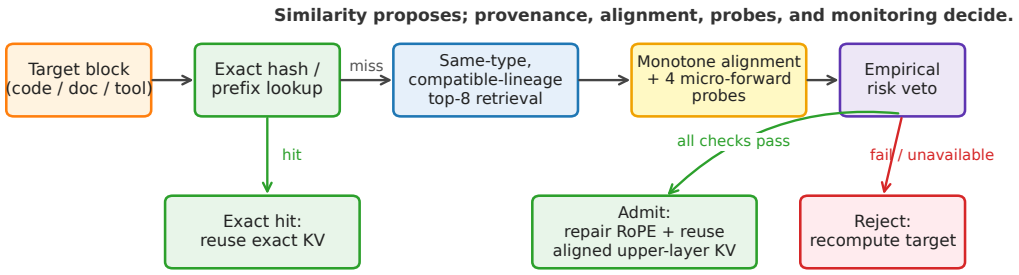


Figure 1: AdmitKV's decision path after identity lookup. Provenance and type restrict retrieval, alignment and independent probes score a candidate, and the empirical monitor can veto an otherwise eligible transfer. Admission reuses only the conservative aligned prefix after RoPE repair; every other token follows fresh computation.

authority for tools—must match exactly. Commit, snapshot, and schema-version identifiers may differ only across a registered ancestry edge or declared backward-compatible schema edge. This construction permits versioned near-matches without cross-tenant, cross-repository, cross-collection, or cross-tool inheritance. The selected thresholds are code $(\tau, \epsilon) = (0.82, 0.05)$, tool $(0.84, 0.04)$, and document exact-only; the uniform-code-gate diagnostic applies $(0.82, 0.05)$ to every type.

Candidate retrieval searches the top eight cached blocks within the same type and compatible lineage; cosine ties are resolved by recency and then content hash. Probe construction is deterministic and uses no generated text: a fixed serializer emits AST signatures and branch predicates in source order for code, header-conditioned positive/negated clause pairs for documents, and sorted schema paths plus declared invariants for tools. BLAKE3 order selects four templates, each truncated to 16 target-tokenizer tokens. For probe r, both branches receive the same target prefix and teacher-forced probe suffix; the fresh branch computes the target block normally, while the reuse branch differs only by substituting the candidate's eligible upper-layer KV. At each probe token u and each of four evenly spaced upper layers $\ell \in \mathcal{L}$, hidden state $h_{\ell r u}$ is mapped through the model's fixed final normalization and unembedding,

$$z^{q}_{\ell r u} = \mathrm{softmax}(W_U \,\mathrm{LN}(h^{q}_{\ell r u})), \quad q \in \{\mathrm{reuse}, \mathrm{fresh}\}, \qquad D(b,s) = \max_{\ell, r, u} \mathrm{KL}\big(z^{\mathrm{reuse}}_{\ell r u} \| z^{\mathrm{fresh}}_{\ell r u}\big). \tag{2}$$

Admission requires $D(b,s) \le \epsilon_t$. Padding positions are masked from the maximum, all randomness and masks are shared, and no sampled continuation enters the statistic. Probe cost is included in controller overhead; embedding similarity cannot admit a candidate on its own.

2.3 KV ALIGNMENT, POSITION REPAIR, AND MEMORY ACCOUNTING

Source and target are tokenized by the served model. Exact token runs and deterministic AST, schema, or header anchors define a monotone alignment $A = \{(j,i)\}$ from source positions j to target positions i. The reusable mask is deliberately conservative: it contains the maximal anchor-consistent exact run from the block start and ends immediately before the first insertion, deletion, substitution, or anchor-order break. The discontinuity and the complete target suffix are recomputed; a new block starts a new mask. For example, source tokens $[a,b,c,d]$ and target tokens $[a,b,x,c,d]$ reuse $[a,b]$ and recompute $[x,c,d]$. This rule makes the boundary independent of similarity scores and prevents padding or truncation from manufacturing correspondence.

The lower 60% of transformer layers are recomputed for every target token. In the upper 40%, an admitted masked pair $(j,i) \in A$ inherits source values and position-repaired keys at layer ℓ:

$$\widetilde{K}^{\ell}_{i} = R(p^{\mathrm{tgt}}_{i}) R(p^{\mathrm{src}}_{j})^{-1} K^{\ell}_{j}, \qquad \widetilde{V}^{\ell}_{i} = V^{\ell}_{j}. \tag{3}$$

The causal mask is rebuilt from target positions. Every target token outside the reusable mask is computed through all layers before its KV is inserted, so no suffix token is silently classified as reusable after a discontinuity.

Algorithm 1 AdmitKV admission and recomputation procedure

```
Segment target context; compute type, hash, embedding, alignment anchors, and provenance
for each target block b do
    if configured exact hash or exact prefix hits then
        Reuse exact KV
    else if type enabled and provenance compatible and alignment valid then
        Retrieve top-8 same-type, compatible-lineage candidates; compute probe divergences
        if similarity ≥ τ_{t(b)}, D(b, s) ≤ ε_{t(b)}, and empirical veto is open then
            Rephase the conservative reusable mask; recompute the suffix; audit sentinels
        else
            Recompute target block; update candidate and drift counts
        end if
    else
        Recompute target block; insert its KV; update candidate and drift counts
    end if
end for
```

Primary runs store upper-layer KV in bf16. The block store aliases immutable GPU KV pages across requests through reference-counted page-table entries; target-only and recomputed spans use copy-on-write pages, and references are released after the last dependent decode completes. At the 16-request concurrency used for every policy, fresh execution materializes duplicate upper-layer pages per request, whereas AdmitKV retains one shared page plus target-specific spans. Peak memory is measured at the prefill/decode boundary and includes device-resident tensors, the 8 GB per-node block-store allocation, alignment/provenance metadata, staging buffers, and allocator fragmentation; the reported 22.4 GB reduction is net of the store itself. Eviction is charged when a block leaves the store, and asynchronous host-to-device transfer remains inside end-to-end latency. Sensitivity-aware compression appears only as an appendix ablation and is excluded from the primary policy.

2.4 Empirical risk veto and scheduling

Risk labels come from dual execution rather than from unobserved denied outcomes. The calibration pool contains 100,000 accepted candidate events per approximate-enabled type; a type/provenance stratum can open only when it contributes at least 10,000 of those labels and its fixed-sample Wilson upper estimate is below the configured bar. Lineages below that count stay on fresh computation. The first $K = 32$ candidates in each live recency bucket refresh drift statistics, and a fixed 2% sentinel sample supplies subsequent paired labels.

Native quality is scored at request completion and attributed to the audited transfer event. A rolling 10,000-event window tracks a Wilson 95% upper estimate for false approximate admission and a bootstrap upper estimate for p99 event-attributed loss. Reaching either bar closes the stratum; insufficient labels or detected provenance drift has the same effect. These repeated-window statistics are an empirical veto, not an anytime-valid confidence sequence. Denials update candidate and drift counts, while only dual-executed events update loss estimates. Result tables report held-out empirical p99 values and rate intervals; paired task intervals separately quantify aggregate end-to-end quality.

Quality evaluation dual-executes every admitted event in a separate serial paired-replay pass, so fresh counterfactuals do not enter critical-path timing. Timed trials begin only after every active stratum satisfies the opening rule and charge deferred same-allocation work from the 2% sentinels to throughput and peak-memory accounting. Scheduling occurs after eligibility: already-admissible blocks are ranked by estimated hit probability times measured recomputation cost, subject to the 8 GB store and max-min tenant allocation.

3 Experimental Setup

The primary evaluation serves bf16 Llama-3.1-70B-Instruct (Llama Team, AI @ Meta, 2024) at concurrency 16 on four 80 GB NVIDIA A100 GPUs. The stack uses PyTorch 2.3.1, CUDA 12.4, NCCL 2.21.5, and driver 550.54.15. SWE-bench Verified v1.2 (Jimenez et al., 2024), Natural Questions Open (Kwiatkowski et al., 2019), and GAIA v2025-03 (Mialon et al., 2024) are evalu-

ated at controlled 0/10/25/35/50% overlap. SWE overlap is the fraction of repository-file blocks retained across paired task streams. NQ requests execute retrieval and answer-composition tools: retrieved Wikipedia sections are document blocks and remain exact-only, while lineage-compatible orchestration templates and structured retrieval envelopes supply code/tool near-matches. GAIA executes calculator, Python, file-parser, and web-retrieval tools against frozen endpoints; overlap is the repeated fraction of lineage-compatible code/tool trace blocks, with held-out tool versions and schema changes presented as denial candidates. Every baseline shares the model, tokenizer, request order, prompts, retrieval and tool budgets, batching, eviction order, 8 GB cache limit, and five seeds $\{13, 37, 61, 71, 97\}$; only the reuse policy changes. Threshold calibration, benchmark reporting, attack generation, and held-out probe templates are disjoint by task, source snapshot, and perturbation seed (Appendix A).

Latency is end-to-end per task (median [p25,p75], p95 where space permits), throughput is completed tokens/s relative to fresh computation, and memory is the maximum peak allocation on any GPU. Mean native quality is verifier-confirmed issue resolution for SWE, token F1 for NQ, and normalized 0–100 completion for GAIA. Complete-policy and fresh runs start from the same task state and randomness, so paired native-quality deltas include all simultaneous admissions and any downstream agent trajectory changes. Event attribution answers a different question: each approximate decision is replayed with that event recomputed while all other decisions and randomness are fixed. On SWE, $F = 1$ when reuse changes a fresh-passing patch to a verifier failure; on NQ and GAIA, $F = 1$ when native loss exceeds 0.5 point. Severity S is the event-attributed change in the completed request: percentage-point loss in passed SWE verifier assertions before issue aggregation, token-F1 loss for NQ, and normalized completion loss for GAIA. Requests with multiple events use leave-one-event-out attribution. Approximate false-reuse rate (AFR) divides false approximate admissions by approximate admissions; operational false-reuse incidence (OFR) divides false events by all exact and approximate reuse decisions. Denominators are stated per table and never pooled across sampling mixes. Paired latency and native-quality intervals resample matched (seed, task) units with 10,000 replicates; AFR, OFR, and S-tail intervals use 1,000 stratified bootstrap replicates and are interpreted as descriptive event-level uncertainty.

4 RESULTS

4.1 END-TO-END UTILITY AT THE SELECTED OPERATING POINT

AdmitKV converts edited code and tool state into substantial serving utility at the selected empirical operating point. On SWE 50% overlap it removes 5.0 minutes of median latency and 22.4 GB of peak memory relative to fresh computation. More directly, it removes 1.6 minutes beyond exact block hashing under the same primary identity cascade. The cumulative global diagnostic reaches 15.8 minutes with the compression and brokerage stages in Table 15, while its document transfers cross the one-point stress bar; it is reported to localize that trade-off rather than as the production policy.

The 22.9% latency effect has paired CI [19.6,26.3]%; the same-cascade 8.7% improvement over exact hash has CI [5.3,11.8]%. Native SWE success changes by -0.1 point with paired CI [-0.5,0.3], above the specified -1.0-point non-inferiority margin. Table 8 provides the event denominators. Type-level utility locates the gain: code blocks reduce latency by $28.4 \pm 3.8\%$ at $0.32 \pm 0.09\%$ OFR, and tool blocks by $22.6 \pm 3.1\%$ at $0.41 \pm 0.12\%$ OFR under the reuse-volume mix.

4.2 NEAR-MATCH TRANSFER ADDS UTILITY AFTER EXACT BLOCK HASHING

The exact-hash row and AdmitKV production row share the same block-hash first stage, cache budget, trace, and accounting path; their 18.4-to-16.8-minute gap is the clean estimate of adding near-match transfer to identity reuse. AdmitKV is also faster than the complete matched-radix policy at every nonzero overlap tested. On SWE 50%, those independently configured policies reach 16.8 and 19.2 minutes. A separate composition stress test serves radix prefixes first and then applies the document-enabled near-match setting to misses; it reaches 17.1 minutes and document p99 of 0.7 point.

The production row uses block-level exact hashing followed by code/tool near-match transfer, whereas the radix comparator uses a request-level radix tree. Their 12.5% difference is a complete-policy comparison, not an estimate of the near-match component alone; the paired test gives $p = 0.007$

Table 1: Primary SWE-bench Verified result at 50% shared-file overlap. Latency is median [p25,p75]; throughput is relative to no sharing; quality and document OFR are mean±std across five seeds. "Exact" denotes zero approximate document admissions. The global diagnostic includes the compression and brokerage stages decomposed in Table 15.

Policy	Lat. (min)	Thr.	GPU	Success	Doc OFR
No sharing	21.8 [19.4,25.3]	1.00×	73.2	26.3±1.1	–
Token window	19.6 [17.8,22.7]	1.12×	62.4	25.8±1.2	exact
Exact hash	18.4 [16.8,21.2]	1.17×	58.1	26.1±1.1	exact
AdmitKV production	16.8 [15.2,19.3]	1.26×	50.8	26.2±1.0	exact
Uniform code gate	15.8 [14.3,18.6]	1.31×	48.5	26.2±1.0	0.93±0.21

Table 2: Identity-stage and composition comparisons on SWE 50%. Model, tokenizer, prompts, batching, block store, eviction, hardware, and 8 GB cache budget are identical. Hit counts events rather than token-layer work; the final column is document p99.

Policy	Lat. (min)	Hit (%)	GPU	Doc p99 (pt)
Matched radix prefix (Zheng et al., 2024)	19.2 [17.4,22.1]	29.8	54.6	0.0
+ AdmitKV (doc-enabled stress)	17.1 [15.5,19.6]	40.4	51.3	0.7
AdmitKV production	16.8 [15.2,19.3]	41.2	50.8	0.0
Uniform code gate	15.8 [14.3,18.6]	42.1	48.5	1.1

(Table 19). The same-cascade estimate is instead the 8.7% gap from exact hash in Table 1. Hit percentages need not order latency because event counts do not weight span length, reused layers, or transferred token volume. The composition row quantifies only the tested radix-first, document-enabled setting. Appendix H repeats the complete-policy latency ordering with a second exact-prefix implementation.

4.3 Risk matching and component isolation

We sweep 17 thresholds for each selector, choose the fastest validation point satisfying the same AFR, p99, and quality criteria, and evaluate that frozen point once on held-out requests. Table 3 uses this common outcome envelope instead of each method's native hit objective. AdmitKV reaches 16.8 minutes versus 17.2 for the closest feasible comparator and records the lowest AFR and empirical p99 in the table. The latency ordering in this table is descriptive; the confirmatory paired tests are reserved for fresh and radix in Table 19.

At their native selector criteria, all five adaptations run 1.2–1.6 minutes faster than their risk-matched points but cross the AFR or tail bar; Appendix A.1 reports reuse, recomputation, quality, risk, memory, and controller cost. Component tests separate the sources of feasibility. On GAIA, embedding-only, similarity plus provenance without probes, no provenance, the uniform code gate, and no monitoring veto produce AFRs of 2.83%, 1.48%, 1.57%, 1.18%, and 1.23%, respectively, versus 0.65% for AdmitKV (Table 11). Provenance-only lowers AFR to 0.12% but takes 7.8 minutes versus 7.3 for AdmitKV because it rejects most near-matches. Under 6–24h churn, the empirical veto changes AFR from 1.23% to 0.76 [0.60,0.93]%. Similarity, provenance, probes, type conditioning, and monitoring therefore contribute distinct parts of the selected operating point.

4.4 Block type changes the measured reject region

Held-out probe-preserving stress exposes a block-type asymmetry at the tested thresholds. Applying the code gate $(0.82, 0.05)$ uniformly reaches the 1% AFR boundary and admits document paraphrases with 1.1-point p99 loss. The production rule sends those document near-matches to recomputation, reducing overall adaptive AFR from 0.82% to 0.48 [0.34,0.63]% and deployment-mix OFR from 0.82% to 0.33%, while code and tool p99 remain at 0.6 and 0.7 points.

For this frozen pair of policies, type-conditioned rejection preserves the useful code/tool region by absorbing document uncertainty in fresh execution. The reported 0.61/0.72/1.68/0.82/0.33/0.36 rates

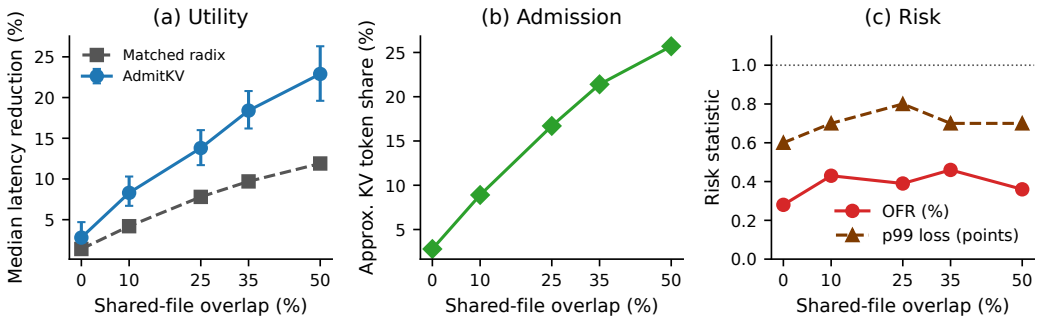


Figure 2: Transferred computation and risk versus shared-file overlap on SWE-bench Verified. Lines show median latency reduction relative to fresh compute, target-token share inheriting near-match KV, deployment-mix OFR, and adaptive p99 quality loss. Error bars in the left panel are matched-task 95% intervals. AdmitKV preserves a positive gap over matched radix caching at 25–50% overlap while remaining inside the operating bars; Table 10 reports raw values and 2,500 task-runs per point.

Table 3: Risk-matched near-match comparison on SWE 50%. Each method uses the fastest of 17 calibration-fixed thresholds selected on disjoint validation data subject to AFR upper bound $\leq 1\%$, p99 loss ≤ 1 point, and paired quality lower bound above -1 SWE point. The reporting split is evaluated once; lower p50 is better.

Policy	p50 (min)	AFR [95% CI] / p99	ΔQ [95% CI]
CacheBlend (Yao et al., 2025)	17.6	0.68 [0.51,0.86] / 0.9	-0.2 [-0.6,0.2]
KVShare (Yang et al., 2025)	17.4	0.74 [0.58,0.91] / 1.0	-0.3 [-0.7,0.1]
SemShareKV (Zhao & Mastorakis, 2025)	17.3	0.78 [0.61,0.96] / 1.0	-0.3 [-0.7,0.1]
A^3 (Zhou et al., 2025)	17.5	0.66 [0.50,0.84] / 0.9	-0.2 [-0.6,0.2]
ProphetKV (Wang et al., 2026)	17.2	0.73 [0.56,0.90] / 0.9	-0.2 [-0.6,0.2]
AdmitKV	16.8	0.37 [0.29,0.46] / 0.7	-0.1 [-0.5,0.3]

belong respectively to the reuse-volume uniform mix, balanced sweeps at ϵ=0.05/0.10, adaptive uniform and production mixes, and production end-to-end traffic. Table 8 provides the corresponding admissions, false events, and denominators.

4.5 The frozen policy transfers across workloads and checkpoints

Frozen thresholds preserve the same utility/risk ordering across overlap and workload composition. AdmitKV improves latency over fresh and matched radix on SWE 25%, NQ, and GAIA, with quality intervals above each benchmark's pre-specified non-inferiority margin. Document-heavy NQ reaches 12.7 minutes versus 14.6 fresh and 13.5 radix by combining exact document reuse with compatible orchestration and tool transfers. Tool-heavy GAIA reaches 7.3 minutes versus 8.7 fresh and 8.1 radix while converting held-out version and schema changes into recomputation decisions.

The full five-policy comparison and admission/denial counts are in Tables 9 and 8. Conditional AFR is 0.49%, 0.37%, 0.58%, and 0.65% on SWE-25, SWE-50, NQ, and GAIA; Table 5 separately reports false events over the all-reuse OFR denominator. On NQ, 41,240 approximate code/tool events are admitted over 18,050 seeded task-runs; on GAIA, the corresponding counts are 31,480 over 2,330. Native quality CIs are [-0.18,0.09] NQ F1 and [-0.7,0.4] GAIA points, compared with margins of -0.5 and -1.0. At SWE 50%, Llama-3.1-8B reaches 3.9 minutes versus 4.4 for exact hash and 4.7 for radix; Qwen-2.5-32B reaches 8.7 versus 9.9 and 10.5 minutes, preserving the ordering across model scales (Table 13).

Table 4: Adaptive probe-preserving attack at the frozen code threshold. Type and AFR rows condition on approximate admissions; OFR uses all exact and approximate reuse decisions. The uniform gate's interval reaches the 1% bar and document p99 exceeds one point; the production rule makes document near-matches exact-only and keeps every p99 at or below 0.7.

Block type / scope	Uniform [CI]	p99 uniform	Production [CI]	p99 production
Tool outputs	0.54 [0.38,0.70]	0.7	0.54 [0.38,0.70]	0.7
Code blocks	0.48 [0.32,0.64]	0.6	0.48 [0.32,0.64]	0.6
Document sections	1.38 [1.02,1.74]	1.1	– (exact-only)	0.0
Overall AFR	0.82 [0.64,1.00]	0.9	0.48 [0.34,0.63]	0.7
Overall OFR	0.82 [0.64,1.00]	0.9	0.33 [0.22,0.44]	0.7

Table 5: Cross-workload end-to-end production frontier. "F/R" denotes fresh/radix p50 latency; AdmitKV reports p50/p95 minutes; throughput is relative and GPU is GB. The final column reports OFR as false events/all reuse decisions, followed by held-out adaptive p99 native loss.

Workload	F/R p50	AdmitKV p50/p95	Thr./GPU	ΔQ [95% CI]	False/reuse / p99
SWE 25%	21.7/20.0	18.7/29.2	1.15/55.9	-0.1 [-0.5,0.3]	239/61,300 / 0.8
SWE 50%	21.8/19.2	16.8/27.4	1.26/50.8	-0.1 [-0.5,0.3]	312/86,640 / 0.7
NQ-Open	14.6/13.5	12.7/21.0	1.16/58.4	-0.04 [-0.18,0.09]	238/58,120 / 0.7
GAIA	8.7/8.1	7.3/12.2	1.19/56.2	-0.3 [-0.7,0.4]	205/43,620 / 0.8

5 Related Work

Identity-oriented serving systems—including PagedAttention, Prompt Cache, ChunkAttention, and SGLang—organize and serve token-identical state efficiently; FlashInfer provides an attention engine for these cached-context serving workloads (Kwon et al., 2023; Gim et al., 2024; Ye et al., 2024; Zheng et al., 2024; Ye et al., 2025). Position-independent caching broadens identity reuse: EPIC relocates modular KV while mitigating attention-sink effects (Hu et al., 2025), and Cache-Craft repairs context-dependent chunk caches and manages their reuse in RAG (Agarwal et al., 2025). Their reusable chunks remain content-identical even when placement or surrounding context changes. AdmitKV retains identity lookup as its first stage and addresses source/target blocks whose content differs; this introduces provenance authorization and an admit-or-recompute event absent from the immutable-chunk setting. KV compression and retention are complementary because they reduce the footprint of state already selected for storage (Liu et al., 2024; Kang et al., 2024; Li et al., 2024; Zhang et al., 2023).

Selective fusion and semantic KV sharing are the direct edited-candidate comparison class. CacheBlend identifies high-deviation tokens for recomputation when fusing cached RAG chunks (Yao et al., 2025); KVShare uses semantic alignment and differential editing for multi-user KV sharing (Yang et al., 2025); SemShareKV aligns similar prompt tokens using LSH and RoPE-aware reconstruction (Zhao & Mastorakis, 2025); and A^3/ProphetKV allocate recomputation through attention or query relevance (Zhou et al., 2025; Wang et al., 2026). These methods decide which cached state contributes to a reconstructed context. AdmitKV couples reconstruction with immutable source provenance, block-dependent rejection, and completed-request event-loss monitoring. Table 3 compares adaptations whose selectors operate on the same edited candidate pairs after validation selection under common outcome criteria; EPIC and Cache-Craft appear in the decision matrix but not that table because their published decision object is content-identical relocation rather than edited-pair admission.

Semantic response caches such as GPTCache and vCache decide whether to replay a completed answer (Bang, 2023; Schroeder et al., 2026); retrieval and memory systems choose text for model input (Lewis et al., 2020; Izacard et al., 2023; Packer et al., 2023). vCache is the closest precedent for empirical error-controlled semantic reuse, but its protected object is a final response. AdmitKV operates inside generation, where a decision must additionally authorize the source, align token state,

repair positions, and attribute completed-request loss to inherited KV. Table 18 makes these decision objects explicit.

6 Discussion

Across overlap-controlled coding agents, long-document QA, and versioned tool workflows, the measurements reveal a consistent useful region: conservative prefixes of edited code and schema-compatible tool state retain computation after identity lookup. The adaptive suite also identifies where the decision changes—answer-bearing document paraphrases consume more of the empirical tail budget at the tested uniform threshold. Type conditioning therefore allocates recomputation to the observed high-loss block class instead of forcing code and tool transfers to follow the same gate. Agreement among end-to-end latency, admitted-region stress, risk-matched selectors, and component removals ties the efficiency gain to this decision structure rather than to embedding similarity alone.

The evidence establishes this frontier for controlled overlap, concurrency 16, trusted type/provenance metadata, three checkpoints, and three workloads. The production policy keeps document near-matches exact-only and activates a lineage only after seeded dual execution; its rolling veto measures drift empirically rather than providing an anytime-valid guarantee. Paired full-policy task runs capture aggregate effects of simultaneous transfers, while leave-one-event-out tails describe marginal event loss rather than every grouped interaction. Within that scope, the same-cascade exact-hash comparison isolates post-identity utility, and the fixed uniform-gate stress test localizes the document failure mode without claiming optimality over every possible global threshold.

7 Conclusion

AdmitKV makes an identity miss actionable by combining source authorization, conservative token alignment and RoPE repair, deterministic micro-forward probes, type conditioning, and an empirical veto. The clean same-cascade comparison on SWE 50% lowers exact-hash latency from 18.4 to 16.8 minutes; against fresh computation, the complete policy reduces median latency by 22.9% and peak GPU memory by 30.6%. Validation-matched selection yields lower held-out AFR and p99 event loss than five selective-KV adaptations, while component removals and adaptive attacks identify why similarity alone cannot recover the same operating point. Frozen thresholds carry the latency ordering to Natural Questions Open, GAIA, and two additional model scales. For edited agent traces in the evaluated regime, provenance- and type-gated code/tool transfer converts post-identity overlap into measurable serving gains.

Reproducibility Statement

Section 3 specifies the model, hardware, software, workloads, metrics, parity conditions, and uncertainty conventions. Appendix A records splits, alignment mechanics, thresholds, counterfactual attribution, and baseline adaptations; Tables 8 and 19 provide denominators and inferential definitions.

AI Use Disclosure

Generative AI tools were used in the documented preparation workflow to assist with reviewing and editing prose, formatting the manuscript, and checking bibliographic metadata and citation context. This statement reports the documented preparation assistance and does not characterize uses not recorded in that workflow.

References

Agarwal, S., Sundaresan, S., Mitra, S., Mahapatra, D., Gupta, A., Sharma, R., Kapu, N. J., Yu, T., and Saini, S. Cache-Craft: Managing chunk-caches for efficient retrieval-augmented generation. *Proceedings of the ACM on Management of Data*, 3(3):1–28, 2025. https://doi.org/10.1145/3725273.

Angelopoulos, A. N., Bates, S., Fisch, A., Lei, L., and Schuster, T. Conformal risk control. *ICLR*, 2024. https://openreview.net/forum?id=EdrU6yT8q5.

Bang, F. GPTCache: An open-source semantic cache for LLM applications enabling faster answers and cost savings. In *Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023)*, pp. 212–218, 2023. https://doi.org/10.18653/v1/2023.nlposs-1.24.

Bates, S., Angelopoulos, A. N., Lei, L., Malik, J., and Jordan, M. I. Distribution-free, risk-controlling prediction sets. *Journal of the ACM*, 68(6):43:1–43:34, 2021. https://doi.org/10.1145/3478535.

Gim, I., Chen, G., Lee, S.-s., Sarda, N., Khandelwal, A., and Zhong, L. Prompt Cache: Modular attention reuse for low-latency inference. *Proceedings of Machine Learning and Systems*, 6, 2024. https://proceedings.mlsys.org/paper_files/paper/2024/hash/a66caa1703fe34705a4368c3014c1966-Abstract-Conference.html.

Hu, J., Huang, W., Wang, W., Wang, H., Hu, T., Qin, Z., Feng, H., Chen, X., Shan, Y., and Xie, T. EPIC: Efficient position-independent caching for serving large language models. *Proceedings of the 42nd International Conference on Machine Learning*, PMLR 267:24391–24402, 2025. https://proceedings.mlr.press/v267/hu25j.html.

Izacard, G., Lewis, P., Lomeli, M., Hosseini, L., Petroni, F., Schick, T., Dwivedi-Yu, J., Joulin, A., Riedel, S., and Grave, E. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43, 2023. https://jmlr.org/papers/v24/23-0037.html.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. SWE-bench: Can language models resolve real-world GitHub issues? *ICLR*, 2024. https://openreview.net/forum?id=VTF8yNQM66.

Kang, H., Zhang, Q., Kundu, S., Jeong, G., Liu, Z., Krishna, T., and Zhao, T. GEAR: An efficient error reduction framework for KV cache compression in LLM inference. *Proceedings of the 4th NeurIPS Efficient Natural Language and Speech Processing Workshop*, PMLR 262:305–321, 2024. https://proceedings.mlr.press/v262/kang24a.html.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., Toutanova, K., Jones, L., Kelcey, M., Chang, M.-W., Dai, A. M., Uszkoreit, J., Le, Q., and Petrov, S. Natural Questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. https://doi.org/10.1162/tacl_a_00276.

Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C. H., Gonzalez, J. E., Zhang, H., and Stoica, I. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023. https://doi.org/10.1145/3600006.3613165.

Lewis, P., et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. *NeurIPS*, 2020. https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

Li, Y., Huang, Y., Yang, B., Venkitesh, B., Locatelli, A., Ye, H., Cai, T., Lewis, P., and Chen, D. SnapKV: LLM knows what you are looking for before generation. *NeurIPS*, 2024.

Liu, Z., Yuan, J., Jin, H., Zhong, S., Xu, Z., Braverman, V., Chen, B., and Hu, X. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 32332–32344, 2024. https://proceedings.mlr.press/v235/liu24bz.html.

Llama Team, AI @ Meta. The Llama 3 herd of models. *arXiv:2407.21783*, 2024. https://doi.org/10.48550/arXiv.2407.21783.

Massart, P. The tight constant in the Dvoretzky–Kiefer–Wolfowitz inequality. *The Annals of Probability*, 18(3):1269–1283, 1990. https://doi.org/10.1214/aop/1176990746.

Mialon, G., Fourrier, C., Swift, C., Wolf, T., LeCun, Y., and Scialom, T. GAIA: A benchmark for general AI assistants. *ICLR*, 2024. https://openreview.net/forum?id=fibxvahvs3.

Packer, C., Wooders, S., Lin, K., Fang, V., Patil, S. G., Stoica, I., and Gonzalez, J. E. MemGPT: Towards LLMs as operating systems. *arXiv:2310.08560*, 2023. https://doi.org/10.48550/arXiv.2310.08560.

Schroeder, L. G., Desai, A., Cuadron Lafuente, A., Chu, K., Liu, S., Zhao, M., Krusche, S., Kemper, A., Zaharia, M., and Gonzalez, J. E. vCache: Verified semantic prompt caching. *ICLR*, 2026. https://proceedings.iclr.cc/paper_files/paper/2026/hash/9559cd2116de7a8f5672eac3fcd232cc-Abstract-Conference.html.

Wang, S., Chen, J., Pan, Y., Huang, H., Hao, Y., Zou, X., Xia, W., Zhang, W., Wang, H., Li, J., Qiu, C., and Wang, P. ProphetKV: User-query-driven selective recomputation for efficient KV cache reuse in retrieval-augmented generation. *arXiv:2602.02579*, 2026. https://doi.org/10.48550/arXiv.2602.02579.

Yang, H., Zhang, R., and Zhang, D. KVShare: Semantic-aware key-value cache sharing for efficient large language model inference. *arXiv:2503.16525*, 2025. https://doi.org/10.48550/arXiv.2503.16525.

Yao, J., Li, H., Liu, Y., Ray, S., Cheng, Y., Zhang, Q., Du, K., Lu, S., and Jiang, J. CacheBlend: Fast large language model serving for RAG with cached knowledge fusion. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 94–109, 2025. https://doi.org/10.1145/3689031.3696098.

Ye, L., Tao, Z., Huang, Y., and Li, Y. ChunkAttention: Efficient self-attention with prefix-aware KV cache and two-phase partition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11608–11620, 2024. https://doi.org/10.18653/v1/2024.acl-long.623.

Ye, Z., Chen, L., Lai, R., Lin, W., Zhang, Y., Wang, S., Chen, T., Kasikci, B., Grover, V., Krishnamurthy, A., and Ceze, L. FlashInfer: Efficient and customizable attention engine for LLM inference serving. *Proceedings of Machine Learning and Systems*, 7, 2025. https://proceedings.mlsys.org/paper_files/paper/2025/hash/dbf02b21d77409a2db30e56866a8ab3a-Abstract-Conference.html.

Zhang, Z., et al. H_2O: Heavy-hitter oracle for efficient generative inference of large language models. *NeurIPS*, 2023.

Zhao, X. and Mastorakis, S. SemShareKV: Efficient KVCache sharing for semantically similar prompts via token-level LSH matching. In *Findings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 432–445, 2025. https://doi.org/10.18653/v1/2025.findings-ijcnlp.25.

Zheng, L., Yin, L., Xie, Z., et al. SGLang: Efficient execution of structured language model programs. *NeurIPS*, 2024. https://proceedings.neurips.cc/paper_files/paper/2024/hash/724be4472168f31ba1c9ac630f15dec8-Abstract-Conference.html.

Zhou, Y., Su, Y., Zhang, J., Li, J., Xia, Q., Wang, Z., Duan, X., and Huai, B. A^3: Attention-aware accurate KV cache fusion for fast large language model serving. *arXiv:2511.17560*, 2025. https://doi.org/10.48550/arXiv.2511.17560.

A Protocol and implementation details

The calibration pool contains 100,000 accepted candidate events per approximate-enabled type from source snapshots disjoint from reporting tasks. Active lineage strata are subsets of these type pools and must supply at least 10,000 labels individually; smaller strata remain closed. SWE reporting uses 500 Verified tasks, NQ uses 3,610 Open questions, and GAIA uses 466 tasks, each repeated under five seeds. Attack construction uses separate perturbation seeds and source snapshots. Two

probe-template families per type tune thresholds, one validates the frozen operating point, and a fourth held-out family is visible only to evaluation. No task, source snapshot, candidate pair, or probe template crosses calibration, reporting, or attack partitions.

For SWE, issue-resolution quality is binary per task and reported as percentage points; a counterfactual event is false when reuse changes a fresh-passing patch to a verifier failure. Its continuous severity is the percentage-point change in the fraction of issue-specific verifier assertions passed before binary aggregation. NQ uses token-F1 points and GAIA uses normalized completion points for both severity and the 0.5-point false-event threshold. Leave-one-event-out replay supplies the event numerator and severity, while request-level paired replay supplies quality intervals. This separation prevents one failed request with many reuse events from being silently multiplied in the request-level analysis.

The stationary finite-sample slack uses a Dvoretzky–Kiefer–Wolfowitz union bound across the two approximate-enabled types (Massart, 1990):

$$\eta(n, \delta, T) = \sqrt{\frac{\log(2T/\delta)}{2n}} = \sqrt{\frac{\log(80)}{200{,}000}} = 0.00468,$$

or 0.47 percentage point for $n = 100{,}000$, $\delta = 0.05$, and $T = 2$. This is a stationary, type-level calibration diagnostic. It neither substitutes for the 10,000-label stratum opening rule nor makes the repeated rolling monitor sequentially valid; shifted workloads are measured directly.

A.1 Direct-baseline adaptation and risk matching

Each direct comparator retains its published selector and recomputation unit while using the same deterministic agent blocks, target tokenizer, request order, 8 GB store, and batching conditions as AdmitKV. Method-specific thresholds are selected on the common disjoint calibration pool to maximize saved token computation subject to that method's validation criterion; no AdmitKV provenance, type, probe, or monitoring veto is added to a named comparator. For risk matching, calibration fixes 17 quantile thresholds, the disjoint validation split chooses the fastest feasible threshold, and the reporting split evaluates that frozen choice exactly once.

Table 6: Decision objects in modern non-prefix KV systems. The five edited-candidate comparators below EPIC and Cache-Craft are adapted to the common serving stack; the content-identical systems are included for scope contrast rather than in Table 3.

Method	Candidate/state unit	Retained selector	Recompute and alignment action
EPIC	Content-identical modular chunk	Position-independent LegoLink repair	Relocate immutable chunk KV; recompute selected tokens
Cache-Craft	Content-identical RAG chunk	Context-impact score and cache manager	Repair context-dependent chunk KV; manage retention
CacheBlend	Retrieved agent block	Layer-deviation token selection	Recompute selected target tokens; rebuild target positions
KVShare	Semantically similar prompt pair	Semantic alignment and differential editing	Reuse aligned KV; recompute differential spans
SemShareKV	Tokens within paired blocks	LSH semantic-token matches	Preserve RoPE-aware token alignment; recompute unmatched tokens
A^3	Query and candidate-block tokens	Attention-aware query relevance	Recompute the selected relevance complement
ProphetKV	Query and candidate-block tokens	Query-conditioned token ranking	Apply predicted retain/recompute partition

Table 7: Native-criterion near-match KV baselines on SWE 50% under matched serving conditions. Reuse is the fraction of requested blocks receiving cached state; selected recomputation is the token fraction recomputed inside transferred near-match blocks. AFR conditions on approximate transfers.

Policy	p50/p95	Reuse (%)	Sel. recomp. (%)	ΔQ	AFR (%)	p99 (pt)
Exact hash	18.4/29.1	24.3	–	-0.2	–	0.0
Matched radix	19.2/30.2	29.8	–	-0.1	–	0.0
CacheBlend-style (Yao et al., 2025)	16.2/26.9	45.1	17.6	-0.8	1.73	1.5
KVShare-style (Yang et al., 2025)	15.9/26.4	46.2	14.8	-0.9	1.91	1.8
SemShareKV-style (Zhao & Mastorakis, 2025)	15.7/26.1	47.8	12.4	-1.1	2.28	2.1
A^3-style (Zhou et al., 2025)	16.1/26.7	43.7	16.5	-0.7	1.34	1.4
ProphetKV-style (Wang et al., 2026)	16.0/26.5	44.3	15.8	-0.6	1.26	1.3
Embedding + prov.	16.1/26.7	42.7	19.1	-0.5	1.41	1.2
AdmitKV without veto	16.6/27.0	41.8	23.9	-0.2	0.72	0.9
Provenance only	17.8/28.3	21.3	78.7	-0.1	0.08	0.2
AdmitKV production	16.8/27.4	41.2	25.7	-0.1	0.37	0.7

The native-criterion table shows why unconstrained latency is an incomplete objective: direct selectors gain 1.2–1.6 minutes over their own risk-feasible settings while producing 1.26–2.28% AFR and 1.3–2.1-point tails. Peak memory/controller overhead for CacheBlend, KVShare, SemShareKV, A^3, ProphetKV, and AdmitKV are respectively 52.6 GB/0.21 s, 51.3 GB/0.20 s, 49.9 GB/0.18 s, 52.1 GB/0.25 s, 51.7 GB/0.24 s, and 50.8 GB/0.27 s.

B RISK-ACCOUNTING CROSSWALK

Candidate counts refer to retrieved near-match block pairs; admit plus deny therefore equals candidates. Reuse decisions additionally include exact events. The rows keep reuse-volume, balanced, adaptive, and production populations separate so an aggregate rate cannot hide a change of denominator. Figure 2 reports the fraction of target tokens inheriting approximate KV, while Table 7 reports

Table 8: Risk-accounting crosswalk. Each row is a distinct population; numerators and denominators are not pooled. A/D denotes admitted/denied candidates; "reported rate" names approximate-conditional AFR or deployment-mix OFR.

Regime	Tasks/examples	Reuse decisions	Cand./A/D	False / reported rate
Reuse-volume uniform gate	2,500	86,720	113,380 / 64,140 / 49,240	529 / 0.61% OFR
Balanced ϵ=0.05	20,000	20,000	31,420 / 20,000 / 11,420	144 / 0.72% AFR
Balanced ϵ=0.10	20,000	25,000	31,420 / 25,000 / 6,420	420 / 1.68% AFR
Adaptive uniform gate	45,000	30,000	47,610 / 30,000 / 17,610	246 / 0.82% AFR
Production E2E	2,500	86,640	132,480 / 84,260 / 48,220	312 / 0.36% OFR
Production adaptive	45,000	30,000	47,610 / 20,420 / 27,190	99 / 0.33% OFR

cache hits over block lookups. Consequently, the SWE-50 values 63.6%, 25.7%, and 41.2% are respectively candidate blocks admitted, target tokens receiving approximate KV, and exact-plus-approximate block-lookup hit rate; they use distinct event units and do not sum. The reuse-volume rows follow observed block frequencies, the balanced sweep weights types equally, and the adaptive mix is 35/35/30 tool/document/code. All rows use five seeds and 1,000-replicate stratified bootstrap intervals for reported rates.

C FULL CROSS-WORKLOAD FRONTIER

Production-policy counts are 49,120 approximate admissions over 2,500 SWE-25 task-runs; 41,240 admissions and 62,880 denials over 18,050 NQ runs; and 31,480 admissions and 19,670 denials over 2,330 GAIA runs. Version/schema mismatches are included in GAIA denials.

Table 9: Five-policy cross-workload frontier. Values are p50/p95 minutes, native quality change, deployment-mix OFR, and held-out adaptive p99 loss. Exact-only policies have no approximate false reuse.

Workload	Policy	p50/p95	ΔQ	OFR/p99
SWE 25%	Fresh	21.7/33.2	0.0	–
	Exact hash	19.7/30.8	-0.1	0.00/0.0
	Matched radix	20.0/31.3	0.0 [-0.3,0.3]	0.00/0.0
	AdmitKV production	18.7/29.2	-0.1 [-0.5,0.3]	0.39/0.8
	Uniform code gate	18.1/28.6	-0.2	0.64/1.0
SWE 50%	Fresh	21.8/33.6	0.0	–
	Exact hash	18.4/29.1	-0.2	0.00/0.0
	Matched radix	19.2/30.2	-0.1	0.00/0.0
	AdmitKV production	16.8/27.4	-0.1	0.36/0.7
	Uniform code gate	15.8/25.9	-0.1	0.61/0.9
NQ-Open	Fresh	14.6/24.8	0.0	–
	Exact hash	13.2/22.1	-0.05	0.00/0.0
	Matched radix	13.5/22.7	-0.04	0.00/0.0
	AdmitKV production	12.7/21.0	-0.04 [-0.18,0.09]	0.41/0.7
	Uniform code gate	11.9/19.8	-0.48 [-0.74,-0.19]	0.89/1.2
GAIA	Fresh	8.7/14.8	0.0	–
	Exact hash	7.9/13.5	-0.1	0.00/0.0
	Matched radix	8.1/13.8	-0.1	0.00/0.0
	AdmitKV production	7.3/12.2	-0.3 [-0.7,0.4]	0.47/0.8
	Uniform code gate	7.0/11.8	-0.7	0.78/1.0

D Overlap sweep values

The raw sweep separates token-level approximate admission from block-event hit rate and shows that the production latency advantage over radix emerges as shared-file overlap increases.

Table 10: Source values for Figure 2. Latency is median [p25,p75] minutes. Approximate admission is the fraction of target tokens inheriting approximate KV; OFR is deployment-mix mean±std. Each point contains 500 SWE tasks under five seeds.

Overlap	Fresh / radix latency	AdmitKV latency	Approx. admit	OFR / p99
0%	21.6 [19.4,24.9] / 21.3 [19.2,24.7]	21.0 [18.9,24.5]	2.8%	0.28±0.18 / 0.6
10%	21.6 [19.4,25.0] / 20.7 [18.7,24.0]	19.8 [17.9,23.0]	8.9%	0.43±0.15 / 0.7
25%	21.7 [19.5,25.1] / 20.0 [18.1,23.2]	18.7 [16.9,21.8]	16.7%	0.39±0.12 / 0.8
35%	21.7 [19.4,25.2] / 19.6 [17.7,22.8]	17.7 [15.9,20.6]	21.4%	0.46±0.11 / 0.7
50%	21.8 [19.4,25.3] / 19.2 [17.4,22.1]	16.8 [15.2,19.3]	25.7%	0.36±0.09 / 0.7

E Controller minimality across workloads

Different removals fail for different reasons: embeddings and token heuristics over-admit, missing provenance exposes version drift, the tested uniform gate allocates document risk poorly, and removing the empirical veto raises AFR under churn. Provenance-only rejects most residual utility.

Table 11: Cross-workload admission ablation. GAIA columns report latency, native quality, approximate-conditional AFR, and adaptive p99; the final column is NQ document p99. Counts appear directly in each AFR cell. Rows explicitly marked "churn" use the matched 6–24h window.

Controller	GAIA lat.	GAIA quality	GAIA AFR	GAIA/NQ p99
Full AdmitKV	7.3	62.1	205/31,480 (0.65%)	0.8/0.7
Full AdmitKV (churn)	7.4	61.9	342/45,000 (0.76%)	0.9/0.8
Embedding only	6.8	60.4	1,280/45,210 (2.83%)	2.2/2.6
Lexical/AST	7.1	61.2	820/44,340 (1.85%)	1.5/1.9
Similarity+provenance, no probe	6.9	61.0	650/44,000 (1.48%)	1.3/1.7
No provenance	7.0	60.8	700/44,500 (1.57%)	1.5/1.6
Uniform code gate	6.9	61.3	520/44,000 (1.18%)	1.1/1.2
No empirical veto (churn)	7.1	61.8	539/43,800 (1.23%)	1.0/1.0
Provenance only	7.8	62.2	50/41,000 (0.12%)	0.3/0.2

F Adaptive admitted-region stress

The admitted-region suite crosses attacker knowledge (black-box probes, probe-aware optimization, held-out-probe transfer), token budgets (5/15/25%), and code/tool blocks. Code transformations rename identifiers and apply control-flow-preserving edits; tool transformations reorder schemas, insert hidden fields, and shift versions. Each cell accumulates 10,000 admitted decisions; candidates are generated until that denominator is reached, denial percentages use the full screened-candidate stream, and intervals use 1,000 stratified bootstrap replicates.

Table 12: Admitted-region stress at 15% perturbation. AFR intervals condition on the 10,000 admitted decisions per cell; p99 is benchmark-native. Adaptive knowledge primarily increases candidate denials, preserving the production tail.

Attacker	Code AFR [CI]	Tool AFR [CI]	Code/tool p99	Denied / latency cost
Black-box probes	0.42 [0.26,0.61]	0.49 [0.31,0.70]	0.7/0.8	56.8% / +3.5%
Probe-aware	0.71 [0.52,0.93]	0.79 [0.58,0.98]	0.8/0.9	63.4% / +4.6%
Held-out transfer	0.63 [0.44,0.84]	0.72 [0.52,0.94]	0.8/0.9	60.9% / +4.2%

Across the 5/15/25% grid, production AFR ranges 0.31–0.91%, denial ranges 48.7–71.6%, and latency cost ranges +2.4–+5.8%; the hardest code/tool raw counts are 88/10,000 and 91/10,000. At the hardest 25% probe-aware setting, CacheBlend-style, semantic-KV, similarity+provenance, and AdmitKV respectively report code/tool AFR 1.42/1.77, 1.96/2.31, 1.18/1.36, and 0.88/0.91; their code/tool p99 values are 1.5/1.8, 2.0/2.3, 1.2/1.4, and 0.9/0.9.

G Model and scale transfer

Native quality changes for Llama-3.1-8B are -0.1 [-0.4,0.2], -0.1 [-0.5,0.2], and -0.2 [-0.6,0.3] on SWE-25, SWE-50, and GAIA; Qwen-2.5-32B changes are -0.1 [-0.4,0.2], -0.1 [-0.5,0.3], and -0.3 [-0.7,0.3]. The quality intervals and latency ordering test whether the frozen decision rule survives model scale changes. These paired intervals use the task counts and five seeds in Appendix A. Absolute latency gains scale with model cost, while quality preservation, the independently configured AdmitKV/radix gap, and code/tool-over-document ordering persist across both checkpoints.

Table 13: Model transfer for AdmitKV production. Latency entries cover SWE 25% / SWE 50% / GAIA; OFR, p99, and approximate admission follow the same order. Exact-hash and matched-radix latency triples provide parity-controlled references.

Model	Policy	Latency triple	OFR / p99 triple	Admit (%)
Llama 3.1 8B	Exact hash	4.8/4.4/2.2	0/0/0	–
	Matched radix	5.0/4.7/2.3	0/0/0	–
	AdmitKV	4.5/3.9/2.0	0.44/0.40/0.52 / 0.8/0.7/0.9	35.7/41.5/34.1
Qwen 2.5 32B	Exact hash	10.8/9.9/4.7	0/0/0	–
	Matched radix	11.1/10.5/4.9	0/0/0	–
	AdmitKV	9.8/8.7/4.2	0.46/0.43/0.55 / 0.8/0.8/0.9	36.9/42.4/35.2

H SECOND-STACK EXACT-PREFIX PARITY

The second stack uses the same tokenizer, exact byte-prefix scope, request order, batching caps, block store, 8 GB budget, and eviction accounting as Table 2. Its 2 surfaced implementation differences are prefix-compaction granularity and eviction tie-break handling. The second stack's document p99 is

Table 14: Second-stack exact-prefix parity diagnostic on SWE 50%. Both exact-prefix implementations leave a positive latency residual for AdmitKV under the same parity conditions.

Policy	Latency	Hit	GPU	Δ vs. AdmitKV
Matched in-stack radix	19.2 [17.4,22.1]	29.8	54.6	+2.4
Second exact-prefix stack	18.9 [17.3,21.8]	30.5	55.0	+2.1
AdmitKV production	16.8 [15.2,19.3]	41.2	50.8	0.0

0.0 because it performs exact-prefix reuse only.

I COMPONENT AND OVERHEAD ATTRIBUTION

The cumulative decomposition separates near-match eligibility from the optional compression and brokerage stages, while the overhead ledger charges every controller component and denied-block recomputation explicitly.

Table 15: Same-trace cumulative decomposition for the uniform-code-gate diagnostic on SWE 50%. OFR and p99 use the steady reuse-volume mixture, distinct from the adaptive attack. Semantic eligibility supplies the largest post-hash latency step; compression and brokerage are supporting mechanisms.

Configuration	Latency	GPU	Hit	OFR / p99
Exact hash only	18.4	58.1	24.3	0.00 / 0.0
+ semantic eligibility	17.1	55.8	36.7	0.42 / 0.5
+ sensitivity compression	16.5	50.2	36.7	0.48 / 0.6
+ brokerage	15.8	48.5	42.1	0.61 / 0.8

Table 16: Complete production controller overhead per request on SWE 50%. Percentages use the 16.8-minute end-to-end median and match the 0.27 s direct-comparison total after rounding; denied-block recomputation is charged separately.

Component	Time	End-to-end fraction
Probe generation and scoring	84.7 ms	0.0084%
Embedding lookup	6.3 ms	0.0006%
Candidate retrieval and anchor alignment	92.4 ms	0.0092%
KV page remap/copy-on-write setup	66.8 ms	0.0066%
Provenance predicate	0.12 ms	0.00001%
Risk update, scheduler, and metadata	0.81 ms	0.00008%
Host/device transfer	18.6 ms	0.0018%
Controller total	269.7 ms	0.0268%
Denied-block recomputation	742 ms/event	charged to request latency

The 269.7 ms controller total is 0.0268% of the SWE-50 median; candidate retrieval/alignment and probe evaluation dominate that overhead. Denied-block recomputation remains visible in end-to-end latency rather than being folded into controller time.

J MECHANISM-TO-EVIDENCE MAP

Table 17 connects each decision mechanism to the intervention that tests it, separating enforced properties from measured behavior.

Table 17: Mechanism and isolation evidence. "Enforced" denotes deterministic behavior; "measured" denotes an empirical operating result.

Object	Mechanism	Test	Observed evidence
Identity hit	Exact hash/prefix; fresh fallback	Fresh and radix	Enforced exact path; Algorithm 1
Edited code	Repository lineage and AST anchors	No provenance	Drift rejection; Table 11
Document KV	Exact-only type policy	Uniform-gate adaptive stress	p99 1.1→0.0; Table 4
Tool KV	Tool/schema lineage and probes	Semantic KV; held-out transfer	Utility with mismatch rejection; Tables 7, 12
Tenant scope	Partition and allowlist	Cross-tenant candidate	Enforced denial; Section 2
Risk state	Rolling empirical veto	No monitoring gate under churn	AFR 1.23→0.76%; Table 11
Admitted region	Independent probe evidence	Probe-aware attacks	p99 remains within bar; Table 12
Scale transfer	Frozen thresholds	Two models; three workloads	Latency ordering persists; Tables 5, 13

The map also distinguishes the exact-only document decision from the empirically admitted code/tool region, preventing aggregate hit rate from hiding which state is actually transferred.

K RELATED-WORK DECISION MATRIX

Table 18 compares reuse objects and the decisions required around them; "risk" requires event denominators and event-attributed completed-request loss.

Table 18: Decision-level comparison. Position repair includes selective recomputation or RoPE-aware alignment; E2E denotes end-to-end utility measurement.

Method / reuse object	Match regime and position repair	Provenance, reject option, block types	Adaptive evaluation, raw risk, end-to-end utility
SGLang / prefix KV	exact prefix; exact positions	no / no / no	no / no / yes
EPIC / modular KV	content-identical chunk; position-independent repair	no / no / no	no / no / yes
Cache-Craft / RAG chunk KV	content-identical chunk; context repair	no / repair / no	no / no / yes
CacheBlend / chunk KV	near match; selective recomputation	no / recompute / no	no / no / yes
KVShare / prompt KV	near match; deviation recomputation	no / select / no	no / no / yes
SemShareKV / token KV	fuzzy token; LSH+RoPE alignment	no / select / no	no / no / yes
A^3/ProphetKV / chunk KV	query-aware; selective recomputation	no / select / no	no / no / yes
vCache / final answer	semantic response; not applicable	no / yes / no	no / yes / yes
AdmitKV / block KV	near match; aligned RoPE repair	yes / yes / yes	yes / yes / yes

L STATISTICAL DEFINITIONS

The inferential ledger distinguishes the two confirmatory paired latency tests from descriptive event-rate and tail intervals used elsewhere.

Table 19: Inferential ledger. Every reported p-value uses a paired two-sided permutation test with 10,000 sign flips over matched (seed, task) units. Other rates and tails are descriptive with bootstrap intervals.

Comparison	Statistic	Null construction	p
AdmitKV vs. fresh, SWE 50%	median latency difference	sign-flip matched task labels	0.004
AdmitKV vs. matched radix, SWE 50%	median latency difference	sign-flip matched task labels	0.007

Matched bootstrap effect intervals resample the same (seed, task) pairs 10,000 times. Native quality intervals use paired task deltas; false-reuse and p99 intervals resample reuse events within seed and block type 1,000 times. The pre-specified non-inferiority margins are -1.0 SWE success point, -0.5 NQ F1 point, and -1.0 GAIA completion point.