When to Wait Before Fan-Out: Warm-Up, Verification, and Residency in Cache-Aware LLM Evaluation

Anonymous authors
Paper under double-blind review

Abstract

Long-context LLM evaluation often issues many requests that share a long prefix. When the provider does not coordinate cold requests, dispatching them concurrently can create the same prefix repeatedly, whereas serial release sacrifices parallelism. We study this release decision from the client side, where cache state is hidden and reuse is observed only through completed-response telemetry. Cache-Aware Execution Planning (CAEP) changes only execution order and release time, keeping request payloads and scoring fixed; its stable-canary policy (CAEP-SC) releases a prefix family after a leader and a probe verify reuse, and a break-even rule relates avoided prefix creation to the value of added latency. In development studies, CAEP-SC reduces telemetry-priced cost relative to work-conserving scheduling by 49.9% on DeepSeek at a 3.7% makespan increase, and creation-confirmed release reduces it by 63.4% on Alibaba's explicit cache at 1.50× makespan. Matched controls locate these savings in warm-up rather than verification: in an eight-block hosted ablation, leader-only and blind two-request warm-up match the gate's cost in every block, and across 37 replicated matched comparisons gating never resolves a cost or computation saving. Under constrained vLLM capacity, CAEP-SC processes 3.18× the work-conserving input with Llama-3.1-8B, 50.7% more than blind warm-up; Qwen3.6-27B shows a 41.2% gate penalty at an 8 GiB budget, and competing prefixes can erode reuse after a passing probe. Limiting unfinished families removes excess synthetic computation; on a fixed Qwen QA workload, a four-family cap cuts computed input by 51.2% and makespan by 32.1% relative to uncapped blind warm-up, although work-conserving release remains faster and the effect is unresolved on new document banks. Waiting before fan-out pays off when concurrent cold requests are billed separately, and one completed warm-up request captured the full measured hosted saving.

1 Introduction

Long-context large language model (LLM) evaluation repeatedly submits documents, demonstrations, retrieval context, or system instructions shared by many requests. Prompt caching can reduce prefill work and full-rate uncached input without dropping examples or shortening model-visible content (Gim et al., 2024; Juravsky et al., 2024; Zheng et al., 2024; Srivatsa et al., 2025). Hosted providers have also begun exposing caching and cache-token accounting through their APIs (DeepSeek, 2026; MiniMax, 2026a;b; Volcano Engine, 2026).

The central scheduling decision is when to release requests that share a prefix. Concurrent cold requests can each incur a creation charge if the provider handles them independently. Waiting for one request to finish can avoid this duplication. Waiting for a subsequent cache hit adds a different intervention: release now depends on observed reuse. Comparing a gated scheduler only with cold concurrent release combines these two effects. Shared-prefix and prefix-aware serving systems address related scheduling trade-offs (Srivatsa et al., 2025; Dexter et al., 2025; Xie et al., 2026); here the client makes release decisions from completed-response telemetry.

We study this distinction with Cache-Aware Execution Planning (CAEP), a client framework that keeps model-visible requests and scoring coordinates fixed. Its stable-canary policy (CAEP-SC)

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.

executes an ordinary leader and probe before releasing followers when a registered cache-token gate passes. Leader-only and blind-two controls remove the telemetry decision while retaining one or two ordinary warm-up requests. These controls separate the economic value of warm-up from the behavior induced by verification. They make leader-only release a necessary baseline when assessing a gate (Figure 1).

The matched controls identify two sources of scheduling gains. One completed warm-up request captures the hosted workload's full measured billing reduction. In the engine, bounding active families removes extra computation introduced by uncapped warm-up. Four independent Qwen launches confirm 51.2% less computed input and 32.1% shorter makespan on the existing QA bank, relative to uncapped blind-two. New document banks instead show point estimates of less computation with longer completion, with aggregate intervals including one. The work-conserving baseline remains faster in both QA studies (Figure 2).

Verification adds no measured saving. Across 37 replicated matched comparisons, gated release never resolved lower cost or computation than warm-up without telemetry, and at reduced cache budgets it computed 50.7% (Llama) and 41.2% (Qwen) more QA input than blind-two. The cache observations explain why. Completed responses report past reuse: a passing probe cannot establish future residency, while a failed fractional threshold can coexist with substantial block-aligned reuse. Budget and serializer controls show that the Qwen penalty depends on the cache budget and persists when zero hits are reported explicitly (Section 5.3).

Provider reuse also depends on serialized message roles, boundaries and cache scope (Gu et al., 2025; Lumer et al., 2026). Changing those fields can change outputs (Sclar et al., 2024; Mizrahi et al., 2024; Zhuo et al., 2024; Hua et al., 2025). Our validity contract therefore binds payloads across scheduling policies and audits serialization changes separately. Same-condition repeats (A/A controls) calibrate behavioral disagreement against hosted-serving variability (He & Thinking Machines Lab, 2025; Gond et al., 2026).

Our contributions are threefold:

- **Matched attribution of scheduling gains.** Client-side controls separate warm-up, telemetry-dependent release and admission on identical requests; gate and blind-two share the queue and fan-out implementation.
- **Identification of the hosted saving mechanism.** One completed warm-up request captures the full measured cost reduction, establishing the importance of warm-up-matched baselines.
- **Admission effects and cache-observation limits.** Two model configurations and independent QA repeats measure when admission removes extra computation; budget, telemetry and displacement controls explain how release decisions differ from cache reuse.

2 Evaluation setup and measurements

Scheduling policies execute the same model-visible messages, model and decoding settings, and scorer coordinates under a shared worker limit. Only order, release time and cache scope vary. We verify exactly-once coverage and preserve the original scorer. Serialization changes are separate conditions, evaluated with task scores and same-condition repeats (Appendix F).

Measurements. Computed input counts prompt tokens actually processed by the engine after cache reuse. We reconcile response usage with completed-input counters and require zero preemption for the reported local comparisons. Makespan is wall time from first dispatch to final completion; hosted timing includes endpoint and transport variation. Cached-token counts determine gate decisions but do not directly measure retained cache contents.

For implicit caches, hit, miss and output token counts (H_i, M_i, O_i) and dated per-million-token prices (p_h, p_m, p_o) give telemetry-priced cost

$$\widehat{C}(P) = 10^{-6} \sum_i (p_h H_i + p_m M_i + p_o O_i). \tag{1}$$

Explicit-cache billing separates read, creation, ordinary input and output. Historical retries with unavailable usage receive an all-miss upper bound. Requests share hidden cache state within a scope,

so policy comparisons use paired complete scopes; independent launch or document-bank units supply the follow-up intervals (Section 4.2).

3 RELEASE POLICIES AND COST DECOMPOSITION

3.1 CALIBRATE BEFORE OPTIMIZING

Fresh-scope cold, exact-replay and suffix probes calibrate each provider/model pair; APIs with multiple layouts or explicit caches also receive placement probes. A cache-responsive layout passes the hit threshold in every registered matched case and is bound in a shared request manifest. Recalibration tracks provider drift. A newly selected serialization receives a separate prediction and task-score audit against the original layout (Section 2).

3.2 CONSTRUCT IMMUTABLE PREFIX FAMILIES

For scheduling, a planning-only representation groups requests that share a registered prefix of at least $L_{\min}$ tokens or bytes. Family membership is bound before execution and never modifies the payload. Cache-Aware Prefix Reordering (CAPR) is the flat baseline: it emits a prefix-contiguous order but remains work conserving, so multiple members of a cold family can start together.

3.3 WARM-UP AND VERIFICATION CONTROLS

Stable-canary CAEP (CAEP-SC) sends an original request as leader and then one as probe. Passing the registered cached-token and consecutive-hit requirements enqueues all remaining family members, with hot families promoted to the queue front and concurrency bounded by the global worker cap. This base policy has no separate family-admission cap or fixed family width. A below-threshold cache observation resets the hit streak and triggers another ordinary probe; it may still contain substantial prefix reuse. Missing telemetry keeps the family serial. After release, cache regressions are logged but do not close the gate. Leaders and probes belong to the evaluation population. Appendices A and B specify the local and hosted policy configurations, respectively.

Leader-only releases followers after the leader completes. **Blind-two** executes a leader and one subsequent ordinary request serially, then releases followers regardless of cache telemetry. It shares the native gate's queue and fan-out implementation. Both controls retain actual telemetry. The historical **fixed-delay** control waits 0.6884 s after the leader; its warm-up request count differs from blind-two.

The **admission intervention** limits families from first dispatch until their final request completes. The scheduler selects the first eligible ready task, preserving priority among eligible tasks, the global worker bound and the gate. Enqueuing followers does not release an admission slot.

Alibaba uses **creation-confirmed release** (CAEP-CC): the leader must report at least 1,024 creation tokens, zero read tokens and no error, then all followers are released immediately. Families execute sequentially, with within-family workers equal to the remaining request count. This provider-specific policy has no serial read probe.

3.4 A RETROSPECTIVE BREAK-EVEN DECOMPOSITION

For family f, m_f requests share L_f chargeable prefix tokens. Let $K_f(P)$ be newly created or recomputed prefix tokens divided by L_f, including fractional recreation. Estimation from aggregate usage requires separating prefix and suffix input. With per-token creation/miss and read prices $p_m > p_h$ and policy-invariant suffix/output charges B_f, family cost is

$$C_f(P) = B_f + L_f\{p_h m_f + (p_m - p_h)K_f(P)\}. \tag{2}$$

The policy's modeled saving relative to CAPR is $S_f = L_f(p_m - p_h)(K_f(CAPR) - K_f(CAEP - SC))$. With makespan difference $\Delta T = T(CAEP - SC) - T(CAPR)$ and time valued at λ currency units per second, the model favors CAEP-SC when

$$\sum_f S_f > \lambda \Delta T. \tag{3}$$

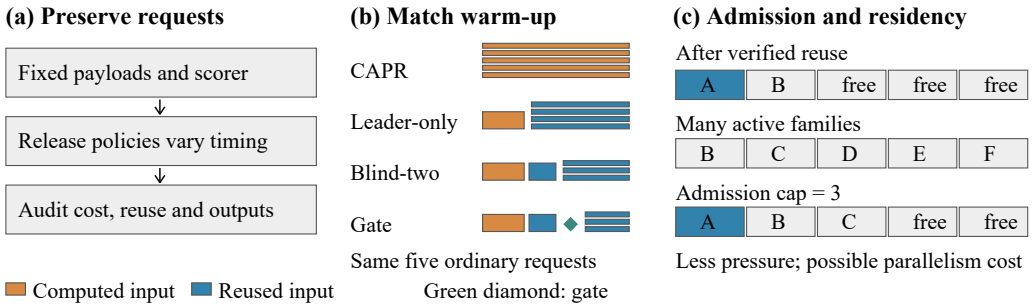


Figure 1: **Method overview.** (a) Request-preserving execution and audit. (b) Each row schedules the same five ordinary requests; the green diamond marks the telemetry-dependent release decision. Timelines illustrate reuse after warm-up. (c) Illustrative key–value (KV) cache states show competing prefixes displacing A and a three-family admission cap reducing concurrent demand. Section 5.2 measures the computation and latency effects of admission.

The full ledger includes policy-dependent suffix/output residuals; makespan includes overlap among families. Warm-up can reduce K_f when concurrent creations are processed separately, and residency preserves reuse through completion. Equation 3 interprets measured outcomes; prospective use requires creation and latency estimates (Appendix F.4).

4 Experimental Methodology

We compare complete cache scopes for scheduling, prefix families for Alibaba, and matched cases with same-layout A/A controls for serialization audits.

4.1 Matched hosted release policies

The hosted ablation uses DeepSeek V4.1 Flash as documented on 22 September 2026 (returned alias `deepseek-flash`). Eight blocks each contain four synthetic record-lookup families with eight requests per family, about 1,200 prompt tokens, and concurrency eight. Four policies process identical inputs in isolated scopes: CAPR, leader-only, blind-two and CAEP-SC. The gate requires one probe meeting a frozen prefix-length proxy threshold. The 1,024 evaluation requests span two same-day windows separated by 120 s. Prior content exposure and transport checks are documented in Appendix B.

4.2 Native admission and QA controls

The matched local study uses Llama-3.1-8B-Instruct (Grattafiori et al., 2024), vLLM 0.19.1 (Kwon et al., 2023) and one GH200, with default (482,496-token) and 65,536-token key–value (KV) cache capacity. Each family has eight requests and an approximately 4K-token prefix. We vary family count/concurrency over $(4, 8)$, $(16, 8)$ and $(16, 32)$, with eight new-content blocks per capacity split across two server launches. The formal study contains 192 scopes and 18,432 requests. Ratios are means of paired block ratios with 10,000-draw bootstrap intervals, conditional on the collected content and launches. Appendix C.1 records gates, controls and deviations.

A follow-up fixes native admission and real-QA controls before either pilot. Synthetic admission uses 13 fresh-content blocks per capacity at $(16, 32)$, testing each warm-up policy with caps 4, 8, 16 and no cap, plus CAPR. Question answering (QA) repeats 27 Qasper and 42 NarrativeQA questions in 20 families at eight workers, with eight reset blocks per capacity and four uncapped policies. Randomized Latin squares balance execution positions; each capacity spans two launches. The 402 formal scopes contain 47,680 requests. Primary contrasts are gate@4/gate and QA gate/blind-two; other contrasts are secondary (Appendix C.2).

Qwen replication. A Qwen3.6-27B replication (Qwen Team, 2026) repeats synthetic admission and QA controls at 16 GiB and automatic cache budgets (164 scopes; Appendix C.3). Follow-ups add

8/16 GiB budget and serializer repeats, four-launch admission confirmation, and document transfer (Appendix D). The admission confirmation repeats the existing 69 questions in 20 families across four independent launches, with a 17,408-token context limit. Document transfer uses four disjoint QASPER banks comprising 64 papers and 248 questions, evaluated at both 8 and 16 GiB with a 32,768-token context limit.

Follow-up estimation. Protocols fix policies, orders, independent units and primary comparisons before confirmation responses, excluding preceding pilots. We average paired log ratios across the two blocks of each launch or bank, then equally across four units. Exponentiated Student-$t(3)$ intervals use Bonferroni correction within each study's four, six or eight primary effects. Secondary intervals are pointwise. All QA arms retain eight workers and report task scores; Appendix D gives the complete effect families.

4.3 Historical hosted studies and serialization audits

Historical DeepSeek development compares eight policies on 192 MWE-ECL inputs in two scopes per policy; the frozen source grid has 900 requests (He et al., 2026). Alibaba evaluates creation-confirmed release on 29 QA requests in eight LongBench document families, balanced across two accounts. Its cache-marked system-prefix layout changes the original message roles; a separate utility repeat and same-request A/A stage measure output variation. Provider-contract probes and interleaved same-layout repetitions calibrate cache telemetry and cross-layout disagreement. Appendix F gives the historical comparisons, billing ledger, task-score analysis and A/A estimator; Appendix A.3 records data and scorer provenance.

5 Results

5.1 A single warm-up captures the hosted saving

A single completed request captures the hosted workload's full measured caching benefit. Leader-only, blind-two and CAEP-SC have identical cost in every block, at 31.1% of CAPR's cost. Leader-only has the lowest mean makespan among the warm-up policies, at 1.075× CAPR (Table 1, Figure 2a). All four policies return the same answers and score 256/256; all 1,024 requests complete without errors or retries. Threshold and timing diagnostics are in Appendix B.

5.2 Family admission controls extra computation

In the controlled engine, no reported warm-up policy computed resolvably less input than work-conserving CAPR, and all four QA policies computed identical input at Llama's default capacity and at both original Qwen budgets (Tables 10 and 11). Local warm-up therefore had no duplicated cold prefill to remove; admission instead bounds the extra computation that uncapped warm-up can introduce under cache pressure.

Independent QA confirmation. Limiting unfinished families reduces the overhead of uncapped warm-up on the fixed Qwen QA workload. At 8 GiB, a four-family cap cuts computed input by 51.2% and makespan by 32.1% relative to uncapped blind-two across four independent launches (Table 2). The cap bounds active families; the worker limit stays eight. Relative to CAPR, capped blind-two has a computed-input ratio of 1.0369 [0.9417, 1.1416] and longer makespan (1.2647 [1.1467, 1.3949]).

New document banks. All four banks show less computed input but longer completion under the same cap. Aggregate cap-four/uncapped ratios are 0.8623 [0.5971, 1.2454] for computation and 1.1616 [0.7952, 1.6969] for time; both intervals include one. CAPR remains faster (Figure 2b,c). Uncapped blind-two/CAPR computation is 1.176× on these banks versus 2.124× on the existing bank, consistent with less extra computation to remove. Content and context limits both change; Appendix D reports these secondary intervals, per-bank effects and task scores.

Synthetic workloads. The follow-up changes admission inside the same native queue and gate. At constrained capacity, limiting unfinished families to four reduces gated input computation by

Table 1: **Matched hosted release policies relative to CAPR.** Means of eight paired block ratios; conditional 95% bootstrap intervals (10,000 draws). Gate/CAPR retains the registered estimates; the two additional time contrasts are post-hoc. Cost uses the frozen peak tariff.

Policy	Cost / CAPR [95%]	Makespan / CAPR [95%]
CAPR	1.0000	1.0000
Leader-only	0.3107 [0.2986, 0.3241]	1.0754 [0.9975, 1.1550]
Blind-two	0.3107 [0.2986, 0.3241]	1.1037 [1.0373, 1.1793]
CAEP-SC	0.3107 [0.2986, 0.3241]	1.2934 [1.1186, 1.5017]

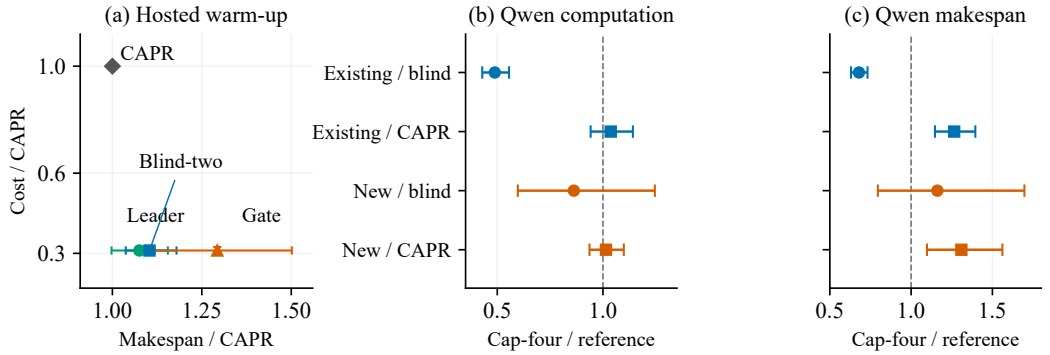


Figure 2: **Warm-up attribution and admission across workloads.** (a) Hosted policy/CAPR mean paired ratios with 95% bootstrap intervals; leader and blind-two time contrasts are post-hoc. (b,c) Qwen at 8 GiB: cap-four blind-two relative to uncapped blind-two or CAPR, on existing QA and new document banks. Bars show study-wise simultaneous 95% intervals from the primary contrasts; dashed lines mark equal computation/time. Lower ratios favor the numerator.

Table 2: **Admission across QA workloads at 8 GiB.** Cap-four blind-two divided by the reference; four independent launch units (existing QA) or document banks (new documents). Simultaneous 95% intervals within each six/eight-effect primary family; Section 4.2 defines the estimator.

Workload	Reference	Computed input ratio	Makespan ratio
Existing QA	Uncapped blind-two	0.4882 [0.4289, 0.5556]	0.6793 [0.6305, 0.7319]
Existing QA	CAPR	1.0369 [0.9417, 1.1416]	1.2647 [1.1467, 1.3949]
New documents	Uncapped blind-two	0.8623 [0.5971, 1.2454]	1.1616 [0.7952, 1.6969]
New documents	CAPR	1.0142 [0.9363, 1.0986]	1.3090 [1.0975, 1.5613]

12.6%; the makespan ratio is 1.0528 [0.8887, 1.2352] (Table 3). Caps four and eight match CAPR computation exactly in all 13 blocks for each of leader-only, blind-two and gated warm-up. Uncapped gating computes 1.1452× as much as CAPR. At default capacity, cap four leaves computation unchanged and increases makespan by 13.8%.

The secondary cap-eight contrast reduces constrained gated makespan to 0.8781 [0.7912, 0.9424] of the uncapped policy, while remaining 1.2701× as slow as CAPR. Admission thus controls extra computation but also restricts execution parallelism. Appendix C.2 shows all caps and the nonbinding cap-sixteen control, including its small timing differences.

5.3 Cache observations and release decisions can disagree

The six-condition Llama matched study finds identical gate/blind-two computation in five conditions and ratio 0.9974 [0.9728, 1.0229] in the sixth; all six makespan intervals include one (Appendix C.1). Across all 37 replicated gate/blind-two and fixed-delay cost or computation comparisons, no upper

Table 3: **Admission and QA: registered primary contrasts in both models.** Paired-block mean ratios with conditional 95% intervals (10,000 draws). Synthetic: 13 Llama or 10 Qwen blocks per budget; QA: eight. Each budget spans two launches. Computation counts completed prompt input tokens; lower ratios favor the numerator.

Contrast	Cache budget	Compute ratio [95%]	Time ratio [95%]
Llama-3.1-8B			
Gate@4/gate	Lower	0.8737 [0.8618, 0.8844]	1.0528 [0.8887, 1.2352]
Gate@4/gate	Default	1.0000 [1.0000, 1.0000]	1.1378 [1.0717, 1.2530]
QA gate/blind-two	Lower	1.5075 [1.3527, 1.6414]	1.4411 [1.3136, 1.5402]
QA gate/blind-two	Default	1.0000 [1.0000, 1.0000]	1.0329 [0.9433, 1.1337]
Qwen3.6-27B			
Gate@4/gate	16 GiB	0.7832 [0.7662, 0.8035]	0.8931 [0.8263, 0.9733]
Gate@4/gate	Automatic	0.9343 [0.9112, 0.9574]	1.0938 [1.0505, 1.1308]
QA gate/blind-two	16 GiB	1.0000 [1.0000, 1.0000]	1.0089 [0.9933, 1.0261]
QA gate/blind-two	Automatic	1.0000 [1.0000, 1.0000]	1.0019 [0.9746, 1.0275]

95% bound lies below one; the estimate most favorable to the gate is 0.9768 [0.9445, 1.0089] (Appendix E.3).

Gating adds computation and delay on real QA under pressure. With the same 69 QA requests and balanced policy positions, the gate computes 50.7% more input and takes 44.1% longer than blind-two at constrained capacity (Table 3); both ratios exceed one in all eight blocks. At default capacity their computation matches in every block and the makespan ratio is 1.0329 [0.9433, 1.1337]. Figure 3 summarizes the three warm-up policies relative to work-conserving release. Relative to CAPR, constrained leader-only, blind-two and gated warm-up compute 1.8006, 2.1135 and 3.1774 times as much input, respectively (Appendix C.2).

Numeric below-threshold observations trigger another ordinary probe. Eight probes lack counts and trigger fallback decisions, but each ends its family, so no fallback request executes. Neither observation identifies whether a prefix was never cached or was evicted; Appendix C.2 reports the full phase counts.

Competing prefixes displace verified reuse. In five separate Llama reset diagnostics, a passing probe reuses 4,096 of 4,113 tokens. Fifteen intervening prefixes reduce follower reuse to 3,648 tokens; waiting ten seconds without competing requests preserves 4,096. With Qwen3.6-27B at 16 GiB, fifteen intervening prefixes preserve the same 3,920 of 4,110 follower tokens as immediate follow-up (Appendix C.3). Displacement depends on the configuration. Engine-source audits reconcile completed prompt computation with response accounting and require zero preemption; repeated lookup misses are recorded separately (Appendix E).

Qwen gate overhead persists with explicit zero counts. The original Qwen study ties QA computation at 16 GiB and automatic budgets, with sampled allocation below 46% (Appendix C.3). Four independent 8/16 GiB launch pairs resolve an 8 GiB gate/blind-two computation ratio of 1.4117 [1.3414, 1.4857]. The default serializer omits zero cache counts, triggering 28 serial-fallback requests per gate scope. Four further serializer pairs remove this fallback by reporting zeros explicitly; gate/blind-two computation remains 1.3622 [1.2612, 1.4713]. Missing zero counts therefore do not fully explain the gate penalty. The serializer interaction interval includes one (Appendix D).

Gate failure can coexist with substantial reuse. All 112 failed Qwen QA probes returned the block-aligned family-common prefix, yet fell below the fixed fractional threshold (Appendix C.3). Gate failure is a release decision; computed input counts tokens processed without reuse, and displacement is a subsequent loss of reuse. Calibrate release thresholds to attainable cache counts, alongside nominal prefix length.

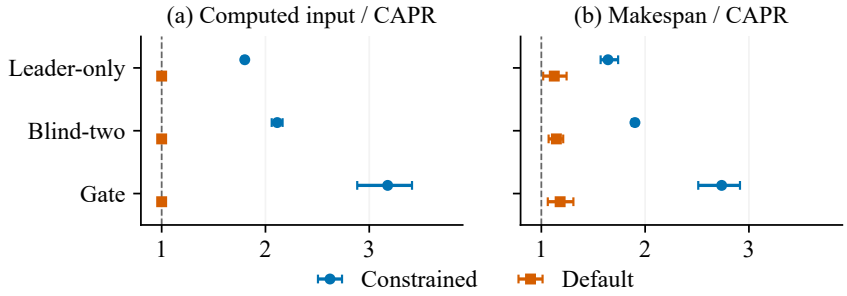


Figure 3: **Matched warm-up on real QA with Llama-3.1-8B.** Constrained and default KV pools; the dashed reference is work-conserving CAPR = 1. Points are arithmetic means of eight paired reset-block ratios; bars are conditional 95% paired bootstrap intervals (10,000 draws). Each block repeats the same 69 questions in 20 document families. These are secondary baseline comparisons; Table 3 gives the primary gate/blind-two contrast.

5.4 Additional hosted settings

Historical DeepSeek development runs observe 49.9% lower cost for CAEP-SC than CAPR, combining warm-up and verification. On Alibaba, creation-confirmed release (CAEP-CC) reduces reconstructed cost by 63.43% at 1.4975× makespan on eight QA families; it verifies creation rather than a subsequent read. Its role-changing layout has inconclusive QA-F1 noninferiority. Separate provider and serialization controls show that cache reuse depends on message layout and that output variation can occur within a fixed layout. Appendix F reports the historical policy frontier, retry accounting, task-score results and A/A comparisons.

6 Related work

Validity and hosted-cache measurement. Prompt formatting changes behavior (Sclar et al., 2024; Mizrahi et al., 2024; Zhuo et al., 2024; Hua et al., 2025), while cache audits and provider evaluations study side channels and reuse (Gu et al., 2025; Lumer et al., 2026). We combine reuse telemetry with A/A controls that compare serialization effects against endpoint variation (He & Thinking Machines Lab, 2025; Gond et al., 2026).

Serving and scheduling. Systems optimize batching and prefill/decode scheduling (Yu et al., 2022; Kwon et al., 2023; Agrawal et al., 2024; Zhong et al., 2024), shared prefixes (Gim et al., 2024; Juravsky et al., 2024; Zheng et al., 2024; Srivatsa et al., 2025; Pan et al., 2025), and prefix-aware queues (Dexter et al., 2025; Xie et al., 2026). BatchLLM combines global prefix sharing with throughput-oriented batching (Zheng et al., 2026); CacheBlend reuses non-prefix RAG caches with selective recomputation (Yao et al., 2025). CONCUR regulates agent admission under cache pressure (Chen et al., 2026). Our client operates through completed response telemetry. We measure how cache capacity affects release decisions based on this past-reuse signal.

Efficient evaluation. Liu et al. (2025) and Tan et al. (2026) improve cache reuse by reordering relational fields or retrieved evidence; such reordering changes payloads. Anthropic recommends warming caches before parallel requests and exposes entries when the first response begins.[1] Our matched controls isolate verification after warm-up. Decoding variability affects evaluation (Song et al., 2025). Example selection, batch prompting, and model routing change the sample, payload, or model (Maia Polo et al., 2024; Li et al., 2025; Cheng et al., 2023; Chen et al., 2024); our execution-only branch preserves all three and the original scorer.

[1] `https://platform.claude.com/docs/en/build-with-claude/prompt-caching`, accessed 23 September 2026.

7 IMPLICATIONS FOR CACHE-AWARE EVALUATION

Leader-only has the lowest mean makespan among the three warm-up policies in the original hosted and QA matched studies, with cost or computation no greater than the gate. Compare any wait or gate with both leader-only and work-conserving release: warm-up can still add computation relative to the latter. Assess admission jointly on computation and time, and calibrate thresholds against returned cache counts before execution: hosted and Qwen probes can report substantial reuse below a fixed-fraction threshold. Report zero counts explicitly to distinguish no reuse from missing telemetry.

Completed responses expose past reuse while routing and eviction remain hidden. Measure residency under competing traffic before using a hit as a release guarantee.

8 SCOPE AND LIMITATIONS

Sampling. Hosted evidence covers two development scopes, eight filtered Alibaba families and eight lookup blocks in two same-day windows. Local studies use Llama-3.1-8B and Qwen3.6-27B on GH200/vLLM. Initial studies repeat 69 exposed QA questions; independent confirmations use four launch units, and transfer studies use four disjoint document banks. Existing-bank Qwen confirmations use a 17,408 context limit; document transfer uses 32,768. Cross-study differences combine workload and context settings (Appendix D). The matched hosted ablation uses one provider and short synthetic record-lookup prompts; providers with different cache visibility, routing or creation charges may change the warm-up benefit.

Utility and selection. Alibaba's role-changing layout has inconclusive utility noninferiority. QA results include response variation and equal-family F1 differences. Development evidence informed scheduler and capacity settings; confirmation plans were frozen before confirmation responses and exclude pilot observations. Appendix F documents the historical prediction-gate failure; Appendix B gives the transport amendment and frozen threshold trace.

9 CONCLUSION

Matched controls locate the hosted saving in warm-up, and no replicated matched comparison resolves a saving from telemetry verification; at reduced cache budgets, gating added input computation in both models. In the engine, bounding unfinished families removed extra computation introduced by uncapped warm-up. On the fixed Qwen QA workload, a four-family cap improved both computed input and completion time relative to uncapped blind warm-up; on new document banks both effects remained unresolved, and work-conserving release stayed faster in both studies. Cache counts describe past reuse; threshold alignment, zero reporting and competing traffic determine how that signal supports release decisions. Compare any gate with leader-only, blind and work-conserving baselines, and measure completion time alongside reuse.

AI USE STATEMENT

In this work, we used generative AI tools to implement the scheduler, experiment runners, analyzers and figure code; generate synthetic evaluation inputs through deterministic generator code; provide feedback on experimental protocols; propose and refine hypotheses; suggest the distributional-distance form of the A/A estimand (Equation 6); and interpret results. We also used them for retrieval and discovery of related work, for reference formatting, and to aid or polish writing. We have not used generative AI tools to originate the research questions or to develop the initial method or conceptual framework. We have reviewed all AI-assisted work; tests, hash-bound plans, payload and token-conservation audits, and primary-source citation checks support its verification. The tools were not treated as authors or as sources of experimental measurements. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics Statement

We use author-controlled cache scopes and do not probe cross-user sharing. Alibaba inputs pass a frozen safety filter without prompt rewriting or obfuscation; credentials and sensitive local paths are excluded from artifacts.

Reproducibility Statement

Appendix A specifies policies, budgets, workloads, request preservation and scorer provenance. Appendices B–D and F give each study's design, sample sizes, amendments and exclusions, and Appendix E defines token accounting and validation checks. The code and study records will be released: frozen protocols, request and result manifests, response records, analyzers and tests, which regenerate the reported tables and figures without model or paid-provider calls and retain failed launches, transport amendments and excluded pilots. Historical runs that lack complete execution source trees will be marked separately from the source-complete confirmation studies. MWE-ECL data and documentation are designated CC BY 4.0, subject to third-party and provider terms. Credentials and private local paths will be excluded.

References

Amey Agrawal, Nitin Kedia, Ashish Panwar, Jayashree Mohan, Nipun Kwatra, Bhargav Gulavani, Alexey Tumanov, and Ramachandran Ramjee. Taming throughput–latency tradeoff in LLM inference with Sarathi-Serve. In *18th USENIX Symposium on Operating Systems Design and Implementation*, pp. 117–134. USENIX Association, 2024. URL https://www.usenix.org/conference/osdi24/presentation/agrawal.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A bilingual, multitask benchmark for long context understanding. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3119–3137, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.172. URL https://aclanthology.org/2024.acl-long.172/.

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, Anthony DiPofi, Julen Etxaniz, Benjamin Fattori, Jessica Zosa Forde, Charles Foster, Jeffrey Hsu, Mimansa Jaiswal, Wilson Y. Lee, Haonan Li, Charles Lovering, Niklas Muennighoff, Ellie Pavlick, Jason Phang, Aviya Skowron, Samson Tan, Xiangru Tang, Kevin A. Wang, Genta Indra Winata, François Yvon, and Andy Zou. Lessons from the trenches on reproducible evaluation of language models, 2024. URL https://arxiv.org/abs/2405.14782.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=cSimKw5p6R.

Qiaoling Chen, Zhisheng Ye, Tian Tang, Peng Sun, Boyu Tian, Guoteng Wang, Shenggui Li, Yonggang Wen, Zhenhua Han, and Tianwei Zhang. CONCUR: High-Throughput Agentic Batch Inference of LLM via Congestion-Based Concurrency Control, 2026. URL https://arxiv.org/abs/2601.22705.

Zhoujun Cheng, Jungo Kasai, and Tao Yu. Batch prompting: Efficient inference with large language model APIs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 792–810. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-industry.74. URL https://aclanthology.org/2023.emnlp-industry.74/.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. In Kristina Toutanova,

Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4599–4610, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.365. URL https://aclanthology.org/2021.naacl-main.365/.

DeepSeek. Context caching. DeepSeek API Documentation, 2026. URL https://api-docs.deepseek.com/guides/kv_cache/. Accessed 2026-08-03.

Gregory Dexter, Shao Tang, Ata Fatahi Baarzi, Qingquan Song, Tejas Dharamsi, and Aman Gupta. LLM query scheduling with prefix reuse and latency constraints. In *Advances in Neural Information Processing Systems*, 2025. URL https://papers.nips.cc/paper_files/paper/2025/hash/878bbdf6227315995c207561211ddb53-Abstract-Conference.html.

In Gim, Guojun Chen, Seung-seob Lee, Nikhil Sarda, Anurag Khandelwal, and Lin Zhong. Prompt cache: Modular attention reuse for low-latency inference. In *Proceedings of Machine Learning and Systems*, volume 6, 2024. URL https://proceedings.mlsys.org/paper_files/paper/2024/hash/a66caa1703fe34705a4368c3014c1966-Abstract-Conference.html.

Raja Gond, Aditya K. Kamath, Ramachandran Ramjee, and Ashish Panwar. LLM-42: Enabling determinism in LLM inference with verified speculation. *arXiv preprint arXiv:2601.17768*, 2026. URL https://arxiv.org/abs/2601.17768v2.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012. URL https://www.jmlr.org/papers/v13/gretton12a.html.

Chenchen Gu, Xiang Lisa Li, Rohith Kuditipudi, Percy Liang, and Tatsunori Hashimoto. Auditing prompt caching in language model APIs. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 20477–20496. PMLR, 2025. URL https://proceedings.mlr.press/v267/gu25b.html.

Horace He and Thinking Machines Lab. Defeating nondeterminism in LLM inference. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20250910. URL https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/.

Wei He, Aline Villavicencio, Rodrigo Wilkens, and Zhenyun Deng. MWE-ECL: Recoverable long-range context does not always override local lexical priors, 2026. URL https://arxiv.org/abs/2609.27590.

Andong Hua, Kenan Tang, Chenhe Gu, Jindong Gu, Eric Wong, and Yao Qin. Flaw or artifact? Rethinking prompt sensitivity in evaluating LLMs. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 19889–19899. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.emnlp-main.1006. URL https://aclanthology.org/2025.emnlp-main.1006/.

Jordan Juravsky, Bradley Brown, Ryan Ehrlich, Daniel Y. Fu, Christopher Ré, and Azalia Mirhoseini. Hydragen: High-throughput LLM inference with shared prefixes, 2024. URL https://arxiv.org/abs/2402.05099.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2018. doi: 10.1162/tacl_a_00023. URL https://aclanthology.org/Q18-1023/.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626. Association for Computing Machinery, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Yang Li, Jie Ma, Miguel Ballesteros, Yassine Benajiba, and Graham Horwood. Active evaluation acquisition for efficient LLM benchmarking. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 35581–35602. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/li25bp.html.

Shu Liu, Asim Biswal, Amog Kamsetty, Audrey Cheng, Luis Gaspar Schroeder, Liana Patel, Shiyi Cao, Xiangxi Mo, Ion Stoica, Joseph E. Gonzalez, and Matei Zaharia. Optimizing LLM queries in relational data analytics workloads. In M. Zaharia, G. Joshi, and Y. Lin (eds.), *Proceedings of Machine Learning and Systems*, volume 7. MLSys, 2025. URL https://proceedings.mlsys.org/paper_files/paper/2025/file/b5dc49f44db2fadc5c4d717c57f4a424-Paper-Conference.pdf.

Elias Lumer, Faheem Nizar, Akshaya Jangiti, Kevin Frank, Anmol Gulati, Mandar Phadate, and Vamse Kumar Subbiah. Don't break the cache: An evaluation of prompt caching for long-horizon agentic tasks, 2026. URL https://arxiv.org/abs/2601.06007.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. tinyBenchmarks: evaluating LLMs with fewer examples. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 34303–34326. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/maia-polo24a.html.

MiniMax. Anthropic-compatible explicit cache. MiniMax API Documentation, 2026a. URL https://platform.minimaxi.com/docs/api-reference/anthropic-api-compatible-cache. Accessed 2026-08-03.

MiniMax. Text prompt caching. MiniMax API Documentation, 2026b. URL https://platform.minimaxi.com/docs/api-reference/text-prompt-caching. Accessed 2026-08-03.

Moran Mizrahi, Guy Kaplan, Dan Malkin, Rotem Dror, Dafna Shahaf, and Gabriel Stanovsky. State of what art? A call for multi-prompt LLM evaluation. *Transactions of the Association for Computational Linguistics*, 12:933–949, 2024. doi: 10.1162/tacl_a_00681. URL https://aclanthology.org/2024.tacl-1.52/.

Rui Pan, Zhuang Wang, Zhen Jia, Can Karakus, Luca Zancato, Tri Dao, Yida Wang, and Ravi Netravali. Marconi: Prefix caching for the era of hybrid LLMs. In *Proceedings of Machine Learning and Systems*, volume 7, 2025. URL https://proceedings.mlsys.org/paper_files/paper/2025/hash/7c180af017258d239bac6248d1eb26ac-Abstract-Conference.html.

Qwen Team. Qwen3.6-27B model card, 2026. URL https://huggingface.co/Qwen/Qwen3.6-27B. Accessed 23 September 2026.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. Quantifying language models' sensitivity to spurious features in prompt design or: How I learned to start worrying about prompt formatting. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=RIu5lyNXjT.

Yifan Song, Guoyin Wang, Sujian Li, and Bill Yuchen Lin. The good, the bad, and the greedy: Evaluation of LLMs should not ignore non-determinism. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long*

Papers), pp. 4195–4206, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.211. URL https://aclanthology.org/2025.naacl-long.211/.

Vikranth Srivatsa, Zijian He, Reyna Abhyankar, Dongming Li, and Yiying Zhang. Preble: Efficient distributed prompt scheduling for LLM serving. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5bc342f48de8264779952fac378f96dc-Abstract-Conference.html.

Gábor J. Székely and Maria L. Rizzo. Energy statistics: A class of statistics based on distances. *Journal of Statistical Planning and Inference*, 143(8):1249–1272, 2013. doi: 10.1016/j.jspi.2013.03.018. URL https://doi.org/10.1016/j.jspi.2013.03.018.

Kaizhen Tan, Rong Gu, and Mingyuan Li. CacheWeaver: Cache-aware evidence ordering for efficient grounded RAG inference, 2026. URL https://arxiv.org/abs/2606.19667.

Volcano Engine. Context caching API. Volcano Engine Ark Documentation, 2026. URL https://docs.volcengine.com/docs/ark/context-caching-api?lang=zh. Accessed 2026-08-04 via the original documentation URL; redirect target re-verified 2026-09-24.

Bing Xie, Zhipeng Wang, Masahiro Tanaka, and Zheng Zhen. PEEK: Predictive Queue-Informed KV Cache Management for LLM Serving, 2026. URL https://arxiv.org/abs/2607.02525.

Jiayi Yao, Hanchen Li, Yuhan Liu, Siddhant Ray, Yihua Cheng, Qizheng Zhang, Kuntai Du, Shan Lu, and Junchen Jiang. CacheBlend: Fast large language model serving for RAG with cached knowledge fusion. In *Proceedings of the Twentieth European Conference on Computer Systems*, 2025. doi: 10.1145/3689031.3696098. URL https://doi.org/10.1145/3689031.3696098.

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. Orca: A distributed serving system for Transformer-Based generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation*, pp. 521–538. USENIX Association, 2022. URL https://www.usenix.org/conference/osdi22/presentation/yu.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-2000. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/724be4472168f31ba1c9ac630f15dec8-Abstract-Conference.html.

Zhen Zheng, Xin Ji, Taosong Fang, Fanghao Zhou, Chuanjie Liu, and Gang Peng. BatchLLM: Optimizing large batched LLM inference with global prefix sharing and throughput-oriented token batching. In *Proceedings of Machine Learning and Systems*, volume 8, 2026. URL https://proceedings.mlsys.org/paper_files/paper/2026/hash/5b7ae1758452854dee4e962207d38304-Abstract-Conference.html.

Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. DistServe: Disaggregating prefill and decoding for goodput-optimized large language model serving. In *18th USENIX Symposium on Operating Systems Design and Implementation*. USENIX Association, 2024. URL https://www.usenix.org/conference/osdi24/presentation/zhong-yinmin.

Jingming Zhuo, Songyang Zhang, Xinyu Fang, Haodong Duan, Dahua Lin, and Kai Chen. ProSA: Assessing and understanding the prompt sensitivity of LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024. doi: 10.18653/v1/2024.findings-emnlp.108. URL https://aclanthology.org/2024.findings-emnlp.108/.

Table 4: **Evidence guide.** Where each reader question is answered.

Reader question	Evidence
Which requests, gates, models and budgets are compared?	Appendix A
How is warm-up separated from verification?	Appendix B
How do admission and QA effects vary by engine setting?	Appendix C
Does admission help across launches and new documents?	Appendix D
How are computed tokens and loss of reuse measured?	Appendix E
What do the other hosted and behavioral studies show?	Appendix F

A Experimental configurations and workloads

A.1 Policy configurations

The controlled-engine studies use one GH200 per server, vLLM 0.19.1, bfloat16 (BF16) eager execution and prefix caching. Frozen model revisions and study-specific controls appear in Appendix C. Table 5 summarizes the study-specific settings. For native policies, one observed count at least

Table 5: **Local policy configurations.** Every gate requires one observation $\geq \lceil 0.9L_{\text{actual}} \rceil$ cached tokens. Released followers join the native queue front. Base policies have no family-admission cap. The first five rows describe the original studies; follow-up context limits are given in tokens.

Study	Cache budgets	Workers / cap	Comparators
Llama matched	65,536 / 482,496 nominal tokens	8 or 32 workers; uncapped	Leader-only, blind-two, CAPR
Llama admission	Same two budgets	32 workers; caps 4, 8, 16, none	Each warm-up policy; CAPR
Llama QA	Same two budgets	8 workers; uncapped	Leader-only, blind-two, CAPR
Qwen synthetic	65,072 / 98,784 nominal tokens	32 workers; gate cap 4 or none	Leader-only, blind-two, CAPR
Qwen QA	Same two Qwen budgets	8 workers; uncapped	Leader-only, blind-two, CAPR
Qwen budget	8 / 16 GiB; context 17,408	8 workers; uncapped	Leader-only, blind-two, CAPR
Qwen telemetry	8 GiB; context 17,408	8 workers; uncapped	Original / explicit-zero; blind-two, CAPR
Qwen admission	8 GiB; context 17,408	8 workers; caps 2, 4, none	Same-cap blind-two; CAPR
Qwen transfer	8 / 16 GiB; context 32,768	8 workers; caps 2, 4, none	Same-cap blind-two; CAPR
Llama transfer	65,536 / 482,496 nominal tokens; context 32,768	8 workers; caps 2, 4, none; 75% input budget	Budgeted blind-two/gate; CAPR orders

$\lceil 0.9L_{\text{actual}} \rceil$ passes the gate, using the tokenizer-derived family-common prefix. Numeric failures trigger another ordinary probe; missing evidence keeps release serial. Released followers enter the native queue front. A family holds an admission slot from first dispatch to final completion. The cap limits unfinished families while preserving the worker bound.

A.2 Request preservation

An evaluation request contains the model-visible messages, model identifier, decoding configuration, and the coordinate used by the scorer. We serialize these fields canonically and bind each request to

$$h_i = \text{SHA-256}(\text{canon}(q_i)). \tag{4}$$

Request preservation fixes these bound fields while policies vary order, release times and logical cache scopes. The analyzer requires exactly-once coverage and matching request, payload and coordinate digests. End-to-end benchmark preservation additionally requires the original harness scorer.

Because hosted inference can remain nondeterministic at temperature zero, the audit enforces payload equivalence and measures prediction and score agreement separately. After execution, coordinates restore scorer inputs; source, plan, manifest, price and completion digests bind results to their runs.

A.3 WORKLOAD CONSTRUCTION AND SCORER PROVENANCE

MWE-ECL. The multiword-expression Effective Context Length workload (He et al., 2026) contains constructed diagnostic contexts and questions. Its original English grid contains 50 manually curated cases (25 literal and 25 figurative), grouped into 25 expression families. He et al. (2026) construct anchors, triggers and answer options to test whether a distant semantic cue changes the selected interpretation, and cross-check the expression inventory against earlier inventories. The diagnostic contexts and questions are newly constructed. Interpretation, anchor retrieval and no-anchor prior-control tasks are crossed with approximate filler-token distances 0, 512, 2,048, 8,192, 32,768 and 65,536, producing $50 \times 3 \times 6 = 900$ requests. The development frontier uses 192 of these inputs; repeated executions do not add new cases.

The benchmark is described by He et al. (2026), whose primary English evaluation uses seven distances through 128K. Our frozen historical grid uses the six distances above; its input records define the evaluated version. The importer invokes the source prompt builder and preserves exact message arrays, task coordinates and reference labels in an immutable manifest. The construction procedure above defines the workload used here; the scheduling comparison operates on the bound requests.

LongBench QA scoring. The Qasper and NarrativeQA inputs use their LongBench task versions (Bai et al., 2024; Dasigi et al., 2021; Kočiský et al., 2018). The Alibaba analyzer uses the LongBench QA-F1 implementation in lm-evaluation-harness (Biderman et al., 2024). The English scorer lowercases text, removes punctuation and articles, normalizes whitespace, and computes token-overlap F1. It takes the maximum over the example's reference answers. The registered utility estimand first averages scores within a prefix family, then gives the eight family effects equal weight. The native scorer and frozen artifact bindings determine the reported values; the citation identifies the benchmark and implementation lineage.

Native QA eligibility and document transfer. Input-only eligibility retains 69 of 155 project questions: 27 Qasper and 42 NarrativeQA in 20 document families. Inputs have fewer than 70,000 characters and 2,048–24,000 rendered tokens; output quality is not a filter. Original single-user messages and reference answers are preserved. The native English token-F1 scorer matches LongBench on 1,000 comparison cases. The later Qwen confirmations use a 17,408-token context limit on this bank. Transfer uses four disjoint QASPER banks with 64 papers, 248 questions and 439 reference answers, at 8 and 16 GiB with a 32,768-token context limit (Appendix D).

B MATCHED HOSTED EVALUATION

B.1 RELEASE RULES AND SCOPE ISOLATION

Table 6: **Hosted policy configurations.** Thresholds use cached input tokens except for the Alibaba leader-creation check.

Study	Gate	Release/admission	Comparator
DeepSeek development	$\lceil 0.9L_f \rceil$; two consecutive hits for $L_f \leq 4096$, otherwise one	All remaining members queued; 8 global workers	Work-conserving CAPR and static windows
Historical hosted ablation	Same short-prefix consecutive-hit rule	Native queue; 8 global workers	Leader-only; leader plus 0.6884 s wait
New hosted ablation	One hit $\geq \lceil 0.9(P_{\text{cal}} - 64) \rceil$	Native queue; 8 global workers	Leader-only; blind-two; CAPR
Alibaba explicit cache	Leader: creation ≥ 1024, read $= 0$, no error	Immediate followers; $m_f - 1$ workers; sequential families	CAPR with m_f workers

The new hosted threshold uses $P_{\text{cal}} - 64$: the calibration request's prompt count minus a fixed suffix allowance, rather than a tokenizer-derived common prefix. Its short-prefix cutoff is zero, so one

passing probe suffices. All arms use eight workers. Blind-two shares the gate implementation and releases after the leader and one ordinary probe regardless of cache counts.

B.2 New hosted protocol and transport amendment

The protocol uses synthetic record lookup with 160 key/value records per family, approximately 1,190–1,202 prompt tokens, temperature zero, disabled thinking and an eight-token output cap. Every arm has 32 scientific requests; the complete run has eight blocks, 32 arms and 1,024 scientific responses, plus 36 calibration and smoke responses. Equal-width `user_id` values differ by arm while model-visible payloads remain identical. The official isolation contract and observed cold/replay controls jointly support this scope separation.[2] The official model mapping identifies V4.1 Flash; the response returns `deepseek-flash`, not an immutable revision identifier.

A first launch completed one block, then stopped after transport errors in the next two; it is excluded from all estimates. A transport-only amendment switched to a pooled HTTPX client without retries and reran the frozen design with identical payload bytes in a fresh scope namespace. The first three content banks were partly exposed in the stopped launch, so the complete run is not wholly held-out content.

Peak-standardized USD rates per million tokens are 0.006 for cache hits, 0.30 for misses and 1.20 for output. This standardization removes tariff-window variation from policy comparisons; reconstructed amounts are not invoices. Observed usage is USD 0.17 for the complete run and USD 0.20 including the stopped launch. Intervals use 10,000 paired block-bootstrap draws.

The near-threshold family has calibration prompt count 1,202, hence a gate of $\lceil .9(1202-64) \rceil = 1025$. Returned reuse is 1,024 throughout. The affected scope contains four leaders, ten probes and eighteen followers; the other scopes each contain four, four and twenty-four. The trace diagnoses this parameterization. The shared-length proxy and attainable provider counts must be calibrated together: lowering 1,025 to 1,024 after observation would relax the registered 90% rule. The trace does not establish a universal cache-block granularity or justify a post-hoc threshold correction.

Hosted order and sensitivity. Table 7 gives every execution order and paired makespan ratio. The frozen design shuffles arms independently within each block; it does not balance arm positions. The gate precedes blind-two in six of eight blocks. Consequently the contrast includes order-dependent endpoint variation and is not an isolated estimate of the CPU cost of checking telemetry. All three warm-up arms have exactly equal reconstructed cost within each block.

Table 7: **Hosted execution order.** G: gated; B: blind-two; L: leader-only; C: CAPR. Blocks are zero-indexed.

Block	Order	G/B makespan
0	L → B → G → C	1.2701
1	B → L → C → G	1.4517
2	G → L → C → B	1.3057
3	G → B → C → L	1.0498
4	G → C → B → L	1.0311
5	L → C → G → B	1.0781
6	C → G → L → B	1.0963
7	G → L → B → C	1.0027

The primary eight-block result retains block 1 and its original 10,000-draw interval. Excluding block 1 is an exploratory sensitivity analysis prompted by the threshold trace, not a revised eligibility criterion. Its mean ratio is 1.1191 [1.0432, 1.2040], so that family does not account for the whole observed contrast. Neither bootstrap interval removes possible order or time drift. Direct leader-only/CAPR and blind-two/CAPR ratios in Table 1 are calculated within each block before averaging; they are not quotients of aggregate ratios. The exploratory block-exclusion analysis uses 100,000 bootstrap draws.

[2] DeepSeek API cache isolation and pricing documentation, accessed 22 September 2026: `https://api-docs.deepseek.com/quick_start/rate_limit/` and `https://api-docs.deepseek.com/quick_start/pricing/`.

Post-hoc phase timing. Both arms use the same non-streaming HTTPX transport, waiting for the complete response body and parsing usage before returning. A passing gate and blind-two both enqueue the remaining followers at the queue front. The diagnostic in Table 8 pairs 31 families, excluding block 1 family 2 from both arms because its release path differs. The primary estimate retains that family and all eight blocks. Mean release-gap differences are below a millisecond; larger differences occur inside observed request durations. The largest phase-mean difference is 130 ms in leader duration, before that family's probe decision. Thus the whole-arm time contrast combines request-duration variation and policy execution; it does not isolate decision overhead. Other families can execute concurrently with that leader. Those durations include provider execution, transport and response parsing, which the recorded timestamps cannot separate. Concurrent family windows do not sum to scope makespan. The two blind-first blocks have the largest gate/blind-two ratios; shuffled arm order does not by itself explain their direction or eliminate time variation.

Table 8: **Post-hoc hosted phase timing.** Client-observed milliseconds, averaged over 31 paired families in eight blocks. The near-threshold family in block 1 is excluded from both arms only in this structural diagnostic. Differences are gate minus blind-two; concurrent phase windows do not decompose scope makespan.

Phase	Blind-two (ms)	Gate (ms)	Difference (ms)
Leader response	518.79	648.33	+129.54
Leader done to probe dispatch	2.09	2.18	+0.09
Probe response	520.80	561.67	+40.87
Probe done to first follower	1.89	1.84	-0.05
Fan-out window	1501.20	1538.44	+37.24

C Controlled-engine comparisons

Table 3 in Section 5.2 gives the registered synthetic admission and QA contrasts for both models; this appendix adds the baseline, nonbinding-cap and budget controls.

C.1 Matched local controls

Table 9 reports all six matched local conditions.

Table 9: Matched local ablation: CAEP-SC/blind-two. Each row has eight paired content blocks split across two server launches. Computation is engine-reconciled completed prompt input tokens, not FLOPs or energy.

KV capacity	Families	Workers	Compute ratio [95%]	Time ratio [95%]
Constrained	4	8	1.0000 [1.0000, 1.0000]	1.1255 [0.8117, 1.5245]
Constrained	16	8	1.0000 [1.0000, 1.0000]	0.9997 [0.9817, 1.0176]
Constrained	16	32	0.9974 [0.9728, 1.0229]	1.0026 [0.8667, 1.1592]
Default	4	8	1.0000 [1.0000, 1.0000]	1.0055 [0.9828, 1.0330]
Default	16	8	1.0000 [1.0000, 1.0000]	0.9848 [0.9302, 1.0192]
Default	16	32	1.0000 [1.0000, 1.0000]	0.9975 [0.9880, 1.0087]

The model snapshot is `d10aef7999a2b5ba950ab3974312feeedbfe0b77`; runs use BF16 eager execution, a 32,768-token model limit and vLLM 0.19.1 on one GH200. The constrained pool has 4,096 blocks of 16 tokens. Each capacity has eight new-content blocks, with four in each of two server launches. All four policies share content, scoring, native queue priority and global concurrency. The isolated blind-two patch releases after a leader and one ordinary probe, without altering their returned telemetry. The six conditions produce 192 formal scopes and 18,432 ordinary requests; the separate pilot adds 24 scopes and 2,304 requests.

All scopes pass coverage, payload/scorer binding, cold-reset and response/engine token-conservation checks, with zero errors, retries or preemption. Controls with automatic prefix caching disabled (APC-off) pass at both capacities.

At constrained capacity with sixteen families and 32 workers, CAEP-SC uses 208 probes across eight blocks, compared with blind-two's 128. Leader, probe and follower computed-token totals are respectively 526,304/61,684/38,924 for CAEP-SC and 526,304/56,528/46,144 for blind-two. At default capacity both have 128 probes and totals 526,304/2,192/13,152. Phase attribution changes with policy, so these are descriptive process measurements rather than a causal mediation decomposition. Per-request service times overlap and must not be summed as makespan.

C.2 Native admission intervention and matched real-QA controls

C.2.1 Frozen design and implementation

The frozen Llama-3.1-8B-Instruct snapshot and shared engine settings are those of the matched local study above. An isolated in-memory patch adds the family cap defined in Appendix A. Dispatch selects the first eligible ready task, preserving order among eligible tasks; unfinished admitted families can dispatch while the cap is full.

Synthetic scopes contain 16 families of eight requests, approximately 4,096 shared-prefix tokens per family, and 32 global workers. The 13 arms are CAPR and leader-only, blind-two and gated warm-up, each with no cap or cap 4, 8 or 16. Cap 16 is nonbinding. Thirteen fresh-content blocks per capacity are split seven/six across two independent server launches. A randomized Latin square places each arm in every execution position once. Both capacities use the same content banks and orders, with distinct allocations and reset scopes. The formal synthetic study contains 338 scopes and 43,264 requests.

QA scopes preserve 27 Qasper and 42 NarrativeQA questions in 20 document families from the earlier native QA bank, including exact payload bytes, scoring coordinates and a 128-token output cap. The four uncapped arms are CAPR, leader-only, blind-two and the gate, at eight workers. Eight reset blocks per capacity split four/four across two launches; two randomized four-arm Latin-square cycles balance positions. QA follows synthetic scopes within each allocation. The 64 QA scopes contain 4,416 requests. These previously exposed questions are repeated across blocks; the repeats do not enlarge the task population.

The protocol, content banks, arm orders, analysis and all formal plans were frozen before the two pilots. Each pilot uses a separate synthetic bank and one QA block, totaling 34 scopes and 3,880 requests across both capacities. Pilots validate execution and remain outside all formal estimates. The follow-up configuration was selected using earlier boundary evidence; its formal sample size and thresholds were unchanged after observing outcomes.

C.2.2 Endpoints, controls and inference

The registered primary contrasts are gate@4/uncapped gate for synthetic admission, and gate/blind-two for QA, separately by capacity. Endpoints are engine-reconciled completed input tokens and whole-scope wall-clock makespan. We report the arithmetic mean of paired block ratios and 95% paired bootstrap intervals using 10,000 draws. Other caps, comparisons with CAPR and geometric-mean sensitivities are secondary. Intervals condition on the selected workloads and observed launches; they are not multiplicity-adjusted comparisons for selecting an optimal cap.

Every scope starts after a cold cache reset. Each allocation passes APC-on cold/warm/reset checks, and both pilots additionally pass APC-off cold/replay. All 402 formal scopes pass exact coverage, payload/scorer hashes and engine prompt-source conservation, with zero errors, retries or preemption. Raw dispatch/completion intervals and admission events independently verify the cap.

C.2.3 Admission results and diagnostic scope

Under constrained capacity, native gate@4/gate computation is 0.8737 [0.8618, 0.8844] and makespan is 1.0528 [0.8887, 1.2352]. All three warm-up policies at caps four and eight have exactly CAPR's completed input computation in every block at both capacities: 156 independently checked scope comparisons. Cap eight also reduces constrained gated makespan to 0.8781 [0.7912, 0.9424] of the uncapped gate, while its ratio to CAPR is 1.2701 [1.2398, 1.2981]. At default capacity, cap four leaves computation unchanged and increases gated makespan to 1.1378 [1.0717, 1.2530]. These results expose the trade-off between family admission and execution parallelism.

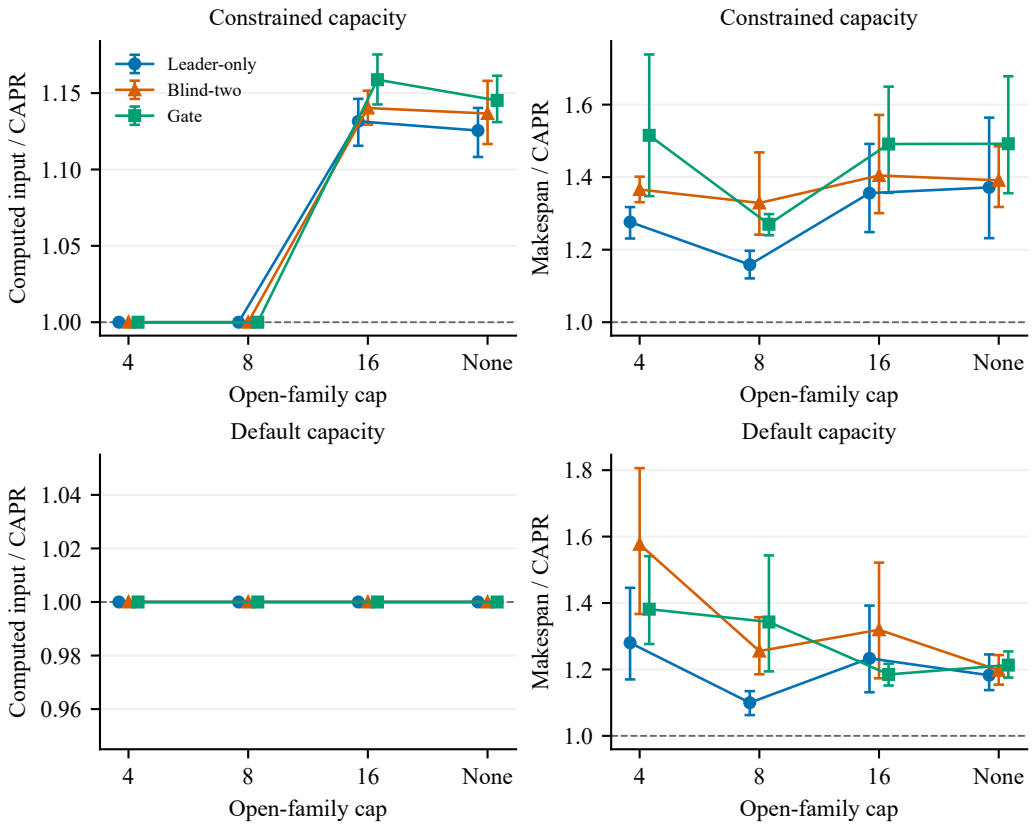


Figure 4: **Admission inside the native scheduler.** Ratios to work-conserving CAPR for each warm-up policy and cap across 13 paired synthetic blocks per capacity. Points are arithmetic means; intervals are 95% conditional paired bootstrap intervals. These are secondary baseline comparisons; Table 3 gives the primary gate@4/uncapped-gate contrast. Cap sixteen is nonbinding.

Nonbinding cap-sixteen compute-ratio intervals cover one. Timing variation persists: default gate@16/gate has ratio 0.9776 [0.9632, 0.9923]. This control qualifies attribution of small timing changes to admission itself. Figure 4 reports all caps relative to CAPR, including the nonbinding and uncapped controls.

The controller samples engine metrics every 100 ms without exposing them to the policies. Allocation gauges and sums of open-family prefix lengths are pressure diagnostics, not measurements of retained cache-block identities. Total actual shared-prefix length across the 16 synthetic families ranges from 65,534 to 65,584 tokens; one of the 13 banks lies at or below 65,536. The intervention therefore does not validate an exact eviction threshold based solely on the sum of prefix lengths.

C.2.4 QA RESULTS, PHASE COUNTS AND UTILITY

At constrained capacity, gate/blind-two computation is 1.5075 [1.3527, 1.6414] and makespan is 1.4411 [1.3136, 1.5402]; all eight paired ratios exceed one for both endpoints. At default capacity, computation matches block by block and the makespan ratio is 1.0329 [0.9433, 1.1337]. Table 10 retains the direct CAPR comparisons for all three warm-up policies.

Figure 3 in Section 5.3 shows these baseline comparisons.

Constrained blind-two uses 160 leaders, 160 probes and 232 fan-out requests; the gate uses 160, 346 and 46. Of its 346 probe attempts, 338 return numeric counts: 98 pass and 240 fall below threshold. Eight lack cache counts and trigger missing-telemetry fallback decisions. Each is member 1 of the two-request family 7, already its last request, so zero serial-fallback requests execute. Seventeen

Table 10: **QA policies relative to work-conserving CAPR.** Eight paired reset blocks per capacity; 95% conditional bootstrap intervals. All rows are secondary contrasts on the same fixed QA population.

KV pool	Policy	Compute ratio [95%]	Time ratio [95%]
Constrained	Leader-only	1.8006 [1.7788, 1.8150]	1.6420 [1.5712, 1.7398]
Constrained	Blind-two	2.1135 [2.0602, 2.1650]	1.9019 [1.8731, 1.9235]
Constrained	Gate	3.1774 [2.8842, 3.4121]	2.7374 [2.5117, 2.9138]
Default	Leader-only	1.0000 [1.0000, 1.0000]	1.1247 [1.0177, 1.2432]
Default	Blind-two	1.0000 [1.0000, 1.0000]	1.1450 [1.0701, 1.2107]
Default	Gate	1.0000 [1.0000, 1.0000]	1.1811 [1.0621, 1.3088]

leaders also lack counts. At default capacity, 16 leaders lack counts, but every probe has a numeric count. The audit of all 120 formal native-admission gate scopes finds no executed serial-fallback requests, including the synthetic cap arms; their missing counts occur in leaders. Compact exports of older matched and historical runs omit per-response phase records, so this audit does not cover them. At default capacity both policies use 160 leaders, 160 probes and 232 fan-out requests, and all 160 gated probes pass. These are ordinary requests redistributed across execution phases, with 552 requests in each policy. Phase counts describe the changed release path, without a causal mediation decomposition of the 50.7% computation difference.

Gate/blind-two response-text disagreements are 31/552 under constrained capacity and 33/552 at default capacity. Mean equal-document-family F1 differences are -0.00038 and $+0.00039$, respectively. The scorer uses normalized token-overlap QA-F1 and the same reference answers. These are descriptive differences over repeated execution pairs, with no registered noninferiority test or behavioral-equivalence claim.

C.3 QWEN3.6-27B: ADMISSION AND VERIFICATION IN A HYBRID MODEL

Model and qualification. We repeat the native policy comparisons with Qwen3.6-27B (Qwen Team, 2026), snapshot `6a9e13bd6fc8f0983b9b99948120bc37f49c13e9`, in BF16 on one GH200. The installed vLLM 0.19.1 engine uses eager execution, Triton/FLA gated-delta prefill, a 32,768-token context limit, and hybrid prefix caching in `align` mode. Thinking is disabled through the model's chat template; the original QA messages, answers and scoring coordinates are preserved. Cold, replay, changed-suffix, cache-reset and APC-off controls pass prompt/cached/computed-token reconciliation with zero preemption. Three earlier environment qualification attempts failed before usable inference, and the backend was amended before any policy outcome.

Budgets and frozen design. The lower cache budget is 16 GiB; the automatic budget uses 0.80 GPU-memory utilization and reports 24.25–24.26 GiB available for caching. The engine reports 65,072 and 98,784 nominal cache tokens, respectively. Hybrid recurrent state, checkpoints and attention pages share these model-specific budgets. Budget selection used allocation metadata and input sizes before the pilots.

The plans freeze 164 formal scopes and 17,216 requests. Synthetic scopes use 16 families of eight requests, approximately 4K shared tokens, 32 workers and an eight-token output limit. Ten fresh-content blocks per budget compare CAPR, leader-only, blind-two, the gate and gate@4 in two randomized five-arm Latin-square cycles. QA repeats all 69 questions in 20 families, at eight workers and a 128-token output limit, under the four uncapped policies. Eight reset blocks per budget form two randomized four-arm Latin cycles. Each cycle uses a separate server launch, and each policy occupies every position twice. Two pilots contribute 1,832 additional requests outside the formal estimates. All planned scopes pass coverage, payload, source-accounting, reset and zero-preemption checks; gate@4 also passes actual-interval admission checks. We use paired block ratios and 10,000 bootstrap draws, conditional on the fixed inputs and observed launches.

Admission improves both endpoints at the lower budget. Gate@4 reduces gated computation by 21.7% and makespan by 10.7% at 16 GiB. At the automatic budget its compute ratio is 0.934 and time ratio is 1.094 (Table 3). Gate@4 matches CAPR computation in every synthetic block at both

budgets. Thus the computation benefit extends to this hybrid configuration, while its latency effect changes with budget.

QA computation matches across policies. All four QA policies compute exactly the same input in every block at both budgets. Gate/blind-two makespan ratios are 1.009 [0.993, 1.026] and 1.002 [0.975, 1.028]. The QA penalty occurs in the Llama pressure configuration, while Qwen computation ties at both tested budgets. Table 11 reports the secondary CAPR comparisons. Across each budget's eight blocks, blind-two executes 160 leaders, 160 probes and 232 fan-out requests; the gate executes 160, 168 and 224, with 112 passing probes. Output text differs in 29/552 gate/blind-two execution pairs at 16 GiB and 27/552 at the automatic budget; mean equal-family F1 differences are -0.0030 and -0.0124. These descriptive utility measurements accompany the efficiency endpoints.

Table 11: **Qwen real-QA policies relative to CAPR.** Secondary contrasts on the same eight paired reset blocks per budget; conditional 95% bootstrap intervals.

Contrast	Budget	Compute ratio [95%]	Time ratio [95%]
Leader/CAPR	16 GiB	1.0000 [1.0000, 1.0000]	1.0246 [0.9937, 1.0595]
Blind-two/CAPR	16 GiB	1.0000 [1.0000, 1.0000]	1.0580 [1.0290, 1.0881]
Gate/CAPR	16 GiB	1.0000 [1.0000, 1.0000]	1.0674 [1.0336, 1.1038]
Leader/CAPR	Automatic	1.0000 [1.0000, 1.0000]	1.0304 [1.0060, 1.0581]
Blind-two/CAPR	Automatic	1.0000 [1.0000, 1.0000]	1.0691 [1.0426, 1.0923]
Gate/CAPR	Automatic	1.0000 [1.0000, 1.0000]	1.0707 [1.0389, 1.1045]

Sampled allocation occupancy. A post-hoc comparison of the recorded 100-ms metric samples finds maximum gate-scope occupancy of 45.9% for Qwen at 16 GiB and 99.6% for constrained Llama QA (Table 12). Qwen therefore has substantial sampled allocation headroom alongside stable observed reuse. The gauge measures allocated blocks; the free queue also contains reusable cached blocks eligible for eviction.[3] It measures a different quantity from retained-prefix residency, and the 100-ms sampling can miss short allocation peaks.

Alignment and residency controls. The engine sets its attention block size to 784 tokens. In qualification, a 2,045-token common prefix yields 1,568 cached tokens and fails the fixed 90% gate despite observed reuse; prefixes of 3,957–4,209 tokens yield 3,920 cached tokens and pass. All formal inputs retain the same threshold, so the policy includes this alignment effect. A post-hoc audit of every formal QA probe finds that all 112/112 failed probes (56 per budget) return exactly $784\lfloor L_{\mathrm{actual}}/784 \rfloor$ cached tokens, the block-aligned family-common prefix, while falling below $\lceil 0.9 L_{\mathrm{actual}} \rceil$. This explains the failed threshold checks through common-prefix alignment; longer matches shared by a subset of requests could exceed that family-wide prefix. In a separate five-reset diagnostic at 16 GiB, immediate follow-up, ten seconds of idle time, and fifteen intervening prefixes each preserve 3,920 cached follower tokens out of 4,110 prompt tokens. These conditions show stable reuse in the tested hybrid configuration, alongside the displacement observed in the Llama diagnostic.

[3] vLLM 0.19.1 `BlockPool.get_usage` and free-queue implementation: `https://github.com/vllm-project/vllm/blob/v0.19.1/vllm/v1/core/block_pool.py`.

Table 12: **Sampled QA allocation occupancy.** Each cell is the mean [minimum, maximum] of eight per-scope peaks, in percent. Polling is nominally every 100 ms. The gauge measures allocated blocks; free-queue blocks can retain reusable cached content.

Model	Budget	Blind-two peak (%)	Gate peak (%)
Llama	Lower	96.4 [90.0, 99.4]	98.1 [95.2, 99.6]
Llama	Default	17.2 [14.6, 18.6]	17.3 [16.2, 18.6]
Qwen	Lower	43.2 [39.6, 45.6]	44.1 [41.7, 45.9]
Qwen	Default	29.2 [28.5, 30.3]	29.4 [27.9, 30.3]

D INDEPENDENT BUDGET, TELEMETRY, ADMISSION AND DOCUMENT REPLICATIONS

Design and analysis units. The follow-ups separate budget, cache-telemetry representation and family admission. Every confirmatory protocol fixes its policies, orders, sample size and primary contrasts before new responses; preceding pilots are excluded. Qwen3.6-27B runs on GH200/vLLM with BF16, eager execution, prefix caching, hybrid-cache alignment, thinking disabled, eight request workers and a 128-token output limit. The existing bank contains 69 questions in 20 families. Budget replication uses four independent 8/16 GiB launch pairs; telemetry replication uses four original/explicit-zero serializer pairs at 8 GiB. Admission confirmation uses four independent explicit-zero launches, each with two blocks and seven policies: CAPR, blind-two and gate, with the latter two uncapped or capped at two/four open families. Request content and scoring coordinates are fixed across policies.

Within each block we compute a paired log ratio, average the two blocks of each launch, and then average four independent launch units. Interactions subtract paired log effects before launch averaging. Exponentiated Student t intervals use three degrees of freedom; Bonferroni correction covers each separately frozen primary family (four, four, six or eight effects). Repeated requests are not treated as independent samples. The tables report every primary endpoint; secondary contrasts use pointwise intervals.

Budget and telemetry findings. At 16 GiB, all policies compute 186,823 prompt tokens per existing-bank scope. At 8 GiB, the gate increases computation and time relative to blind-two. The explicit-zero serializer emits an observed zero cache count instead of omitting that field; unknown values remain unknown. It removes the 28 serial-fallback requests seen in each original-serializer gate scope, but the gate penalty persists. The serializer interaction intervals include one. Every existing-bank scope submits 659,575 prompt tokens; additional computed tokens exactly match fewer engine cache-hit tokens, with zero preemptions and zero recomputed-token counters. Sampled allocation is not a census of retained reusable prefixes: the installed block-pool implementation can evict cached contents from its free queue. Gate penalties therefore also occur below 90% sampled allocation.

Admission confirmation. Cap-four blind-two reduces computation and makespan relative to uncapped blind-two. CAPR remains faster; equal-cap gate/blind-two intervals include one. The cap bounds open prefix families while retaining eight workers. Raw request timestamps independently verify both bounds. Cap-two blind-two matches CAPR's 186,823 computed tokens throughout; the complete secondary comparisons retain its longer completion times.

Document transfer and context settings. The four fixed QASPER banks contain 64 disjoint papers, 248 questions (54/74/54/66 per bank) and 439 reference answers (Dasigi et al., 2021). These banks were previously evaluated on Llama; this is their first Qwen evaluation. Messages, labels, questions and bank membership remain fixed; only tokenizer-derived fields change. The longest Qwen input has 19,650 tokens, so both transfer budgets use a 32,768 context limit. Existing-bank confirmations use 17,408. Each bank runs two blocks of all seven policies at each budget; paired log effects are averaged within bank and then equally across four banks. Cross-study differences jointly change document content and context settings.

At 8 GiB, cap-four/uncapped blind-two computation ratios across banks are 0.9143, 0.7368, 0.9270 and 0.8855; makespan ratios are 1.3201, 1.0097, 1.1574 and 1.1802. Each bank shows less computation with longer completion, while both aggregate intervals include one. The cap-four/CAPR time penalty is resolved; equal-cap gate increments and budget interactions are not. The earlier Llama transfer completes 7,936 scientific requests across the same four banks. All eight primary intervals include one; budgeted/uncapped blind-two computation is identical, with makespan ratio 1.0213 [0.8636, 1.2077]. Its estimates remain separate from Qwen.

Llama transfer policies and order interaction. Llama transfer uses the same four document banks, nominal cache capacities $K = 65{,}536$ and $482{,}496$ tokens, a 32,768-token context limit, eight workers and a 128-token output cap. Each bank has a separate server launch per capacity and two paired blocks. The *budgeted* policies admit families against $B = \lfloor 0.75K \rfloor$ estimated tokens. With $\lceil x \rceil_{16} = 16\lceil x/16 \rceil$, the frozen family weight is

$$w_f = \lceil L_f \rceil_{16} + \min(8, m_f)\lceil P_f^{\max} - L_f + 128 \rceil_{16},$$

where L_f is the serialized common-prefix length (zero for a singleton), m_f is family size and $P_f^{\max}$ is its longest serialized prompt. New families open in frozen family order when the sum of open-family weights plus w_f fits B; the reservation lasts until the last member completes. Already open families can continue dispatching. An oversized family runs alone, with the exception logged. This input-derived estimate allocates room for shared prefixes, private tails and output; it is not a measured KV-occupancy bound. Fixed cap-four instead limits the number of open families. Budgeted blind-two and budgeted gate share this admission rule.

Grouped order exhausts each family's input list before the next; round-robin takes one next request from each nonempty family in turn. Both preserve within-family order and use work-conserving CAPR. For each endpoint the primary interaction is $(\text{round-robin}/\text{grouped})_{\text{constrained}}/(\text{round-robin}/\text{grouped})_{\text{default}}$; a ratio above one means a larger round-robin penalty at constrained capacity. Seeded random interleaving is a separate secondary CAPR condition.

Secondary computation headroom. Uncapped blind-two/CAPR computation is 2.1240 [2.0775, 2.1716] on the existing QA bank and 1.1761 [0.9998, 1.3835] on new banks at 8 GiB; at 16 GiB the latter is 0.9986 [0.9940, 1.0032]. These secondary contrasts use pointwise 95% intervals. The new-bank point estimate leaves less extra computation to remove, while its interval includes one. Document content and context limits both change across studies. On the existing bank CAPR computes 186,823 tokens per scope at both budgets.

Answer quality. All responses are independently rescored using the frozen normalization and reference labels. Family F1 and request EM are averaged within each scope; cross-bank summaries weight banks equally. Capped minus uncapped blind-two F1 differences are -0.366 percentage points on the existing bank and -0.340 on new banks at 8 GiB; EM differences are -0.181 and zero. These are descriptive differences, without an equivalence claim. None of the 10,808 admission-confirmation and Qwen-transfer responses reaches the 128-token output limit.

Request accounting. Scientific/control requests number 4,416/432 for budget replication, 3,312/368 for telemetry, 3,864/312 for admission confirmation and 6,944/624 for Qwen transfer. Three diagnostic pilots are excluded from confirmation intervals: a one-launch-per-serializer diagnostic (414/68), a two-launch admission pilot (966/100) and a 16/8/6/4 GiB pressure pilot (1,104/152). No pressure-pilot budget, including the preempting 6 and 4 GiB conditions, met its original eligibility rule, so that confirmation branch stopped. With the Llama transfer, the eight extension studies total 28,956 scientific and 2,224 control requests, with no new hosted API calls.

Table 13: **Qwen budget replication.** All 4 primary effects; simultaneous 95% intervals within this study, four independent units, $t(3)$.

Contrast	Endpoint	Ratio	Simultaneous 95% CI
Gate / blind-two, 8 GiB	Computation	1.4117	[1.3414, 1.4857]
Gate effect: 8 / 16 GiB	Computation	1.4117	[1.3414, 1.4857]
Gate / blind-two, 8 GiB	Makespan	1.3481	[1.2894, 1.4095]
Gate effect: 8 / 16 GiB	Makespan	1.3525	[1.2858, 1.4226]

Table 14: **Qwen telemetry replication.** All 4 primary effects; simultaneous 95% intervals within this study, four independent units, $t(3)$.

Contrast	Endpoint	Ratio	Simultaneous 95% CI
Gate / blind-two, explicit zero	Computation	1.3622	[1.2612, 1.4713]
Gate effect: explicit / original	Computation	0.9655	[0.8798, 1.0596]
Gate / blind-two, explicit zero	Makespan	1.3102	[1.2429, 1.3812]
Gate effect: explicit / original	Makespan	0.9758	[0.8915, 1.0681]

Table 15: **Qwen admission confirmation.** All 6 primary effects; simultaneous 95% intervals within this study, four independent units, $t(3)$.

Contrast	Endpoint	Ratio	Simultaneous 95% CI
Blind-two cap 4 / uncapped	Computation	0.4882	[0.4289, 0.5556]
Blind-two cap 4 / CAPR	Computation	1.0369	[0.9417, 1.1416]
Gate / blind-two at cap 4	Computation	1.0474	[0.9513, 1.1533]
Blind-two cap 4 / uncapped	Makespan	0.6793	[0.6305, 0.7319]
Blind-two cap 4 / CAPR	Makespan	1.2647	[1.1467, 1.3949]
Gate / blind-two at cap 4	Makespan	1.0321	[0.9523, 1.1187]

Table 16: **Qwen document transfer.** All 8 primary effects; simultaneous 95% intervals within this study, four independent units, $t(3)$. Admission contrasts are at 8 GiB; the interaction compares both budgets.

Contrast	Endpoint	Ratio	Simultaneous 95% CI
Blind-two cap 4 / uncapped	Computation	0.8623	[0.5971, 1.2454]
Blind-two cap 4 / CAPR	Computation	1.0142	[0.9363, 1.0986]
Gate / blind-two at cap 4	Computation	0.9904	[0.9266, 1.0586]
Cap effect: 8 / 16 GiB	Computation	0.8668	[0.6216, 1.2087]
Blind-two cap 4 / uncapped	Makespan	1.1616	[0.7952, 1.6969]
Blind-two cap 4 / CAPR	Makespan	1.3090	[1.0975, 1.5613]
Gate / blind-two at cap 4	Makespan	1.0437	[0.7680, 1.4183]
Cap effect: 8 / 16 GiB	Makespan	0.9347	[0.7646, 1.1428]

Table 17: **Llama document transfer.** All 8 primary effects; simultaneous 95% intervals within this study, four independent units, $t(3)$. Admission contrasts use constrained capacity; Section D defines budgeted admission and the round-robin/grouped capacity interaction.

Contrast	Endpoint	Ratio	Simultaneous 95% CI
Order / capacity interaction	Computation	1.8381	[0.3708, 9.1115]
Budgeted / uncapped blind-two	Computation	1.0000	[1.0000, 1.0000]
Budgeted / cap-four blind-two	Computation	1.0000	[1.0000, 1.0000]
Gate / blind-two, budgeted	Computation	1.0000	[1.0000, 1.0000]
Order / capacity interaction	Makespan	1.3332	[0.7273, 2.4437]
Budgeted / uncapped blind-two	Makespan	1.0213	[0.8636, 1.2077]
Budgeted / cap-four blind-two	Makespan	0.7855	[0.4360, 1.4151]
Gate / blind-two, budgeted	Makespan	1.0673	[0.8217, 1.3864]

E MEASUREMENT AND MECHANISM DIAGNOSTICS

E.1 COMPLETED-TOKEN ACCOUNTING

Counter semantics and validation checks. Frozen vLLM source records prefix lookup statistics in `KVCacheManager.get_computed_blocks` before allocation succeeds. A waiting request may therefore be looked up repeatedly. The initial lookup-equals-completed-prompt gate stopped two pressure runs; one recorded 486,136 lookup-miss tokens versus 300,125 completed computed tokens. We retain these failures as diagnostics and exclude them from policy estimates. Fresh runs use the engine's `prompt_tokens_by_source_total` with source `local_compute`. Let C, H, E, R, P denote computed, local cached, externally transferred, recomputed and total prompt tokens. Checks enforce $C + H + E - R = P$ and $H + E - R$ equal to completed-response cached tokens, matching total response prompt tokens and requiring zero engine preemption. Missing response evidence remains unknown to the policy; controller aggregates validate measurements without entering scheduling decisions.

Each scope additionally requires exactly-once request coverage, immutable payload/scorer hashes, no HTTP errors or retries, cold-reset checks and recorded engine/model metadata.

E.2 DISPLACEMENT AND IDLE CONTROLS

The first displacement diagnostic uses zero, four or fifteen competing families, reserving one of the sixteen bank families for the target. A separate five-block control randomizes immediate, ten-second idle and fifteen-prefix displacement arms. Every arm requires a positive probe before intervention. Immediate and idle followers reuse 4,096/4,113 tokens; displaced followers reuse 3,648/4,113 in all five blocks. Displacement takes 2.18–3.10 seconds, shorter than the successful idle control. Follower computed input is 17 versus 465 tokens. With content held fixed across reset repeats, these controls isolate displacement from elapsed time. The policy traces also contain losses *before* a positive probe, alongside the post-probe displacement mechanism.

Qwen's corresponding 16 GiB controls and the threshold-alignment audit are reported in Appendix C.3. Llama phase counts and missing telemetry decisions appear in Appendix C.2; explicit-zero Qwen replications appear in Appendix D.

E.3 COMPLETE GATE-COMPARISON CENSUS

Across all 37 replicated gate/blind-two and fixed-delay cost/computation comparisons, no upper 95% bound lies below one. The estimate most favorable to the gate is 0.9768 [0.9445, 1.0089] for Qwen synthetic scopes at the automatic budget. Tables 18 and 19 list all 22 original and 15 follow-up contrasts, their admission settings and interval conventions. Several rows share observations; this is an index of comparisons, not a pooled test or a count of independent studies. The intervals do not establish equivalence. Thirty-one earlier exploratory comparisons are outside the census: one single-pair exploration has ratio 0.9813 with a degenerate interval, and replicated exploratory comparisons have upper bounds of at least one.

Table 18: **Gate versus blind warm-up: comparison census.** Original study-specific 95% intervals are retained. The six same-cap native contrasts are additional post-hoc paired-block estimates (10,000 draws). Rows are not independent experiments and are not pooled.

Study / condition	Budget	n	Cost or compute ratio [95%]
Llama matched 4/8	Lower	8	1.0000 [1.0000, 1.0000]
Llama matched 16/8	Lower	8	1.0000 [1.0000, 1.0000]
Llama matched 16/32	Lower	8	0.9974 [0.9728, 1.0229]
Llama matched 4/8	Default	8	1.0000 [1.0000, 1.0000]
Llama matched 16/8	Default	8	1.0000 [1.0000, 1.0000]
Llama matched 16/32	Default	8	1.0000 [1.0000, 1.0000]
Hosted blind-two	Hosted	8	1.0000 [1.0000, 1.0000]
Llama native synthetic	Lower	13	1.0089 [0.9848, 1.0359]
Llama native QA	Lower	8	1.5075 [1.3527, 1.6414]
Llama native synthetic	Default	13	1.0000 [1.0000, 1.0000]
Llama native QA	Default	8	1.0000 [1.0000, 1.0000]
Qwen synthetic	Lower	10	0.9868 [0.9558, 1.0144]
Qwen QA	Lower	8	1.0000 [1.0000, 1.0000]
Qwen synthetic	Default	10	0.9768 [0.9445, 1.0089]
Qwen QA	Default	8	1.0000 [1.0000, 1.0000]
Hosted fixed-delay	Hosted	8	1.0000 [0.9868, 1.0114]
Llama synthetic cap 4	Lower	13	1.0000 [1.0000, 1.0000]
Llama synthetic cap 8	Lower	13	1.0000 [1.0000, 1.0000]
Llama synthetic cap 16	Lower	13	1.0166 [0.9990, 1.0328]
Llama synthetic cap 4	Default	13	1.0000 [1.0000, 1.0000]
Llama synthetic cap 8	Default	13	1.0000 [1.0000, 1.0000]
Llama synthetic cap 16	Default	13	1.0000 [1.0000, 1.0000]

Table 19: **Gate comparison census: all 15 follow-up contrasts.** Together with Table 18, the census contains 37 cost/computation contrasts. All rows have four independent launch or bank units and Student-$t(3)$ intervals. P denotes primary Bonferroni simultaneous intervals within the registered family; S denotes secondary pointwise 95% intervals. Pilots, interactions and duplicate primary/secondary estimates are excluded. Admission settings match within each ratio. Rows are not pooled or independent studies.

Study	Budget / serializer	Cap	Type	Compute ratio [95%]
Budget	8 GiB	none	P	1.4117 [1.3414, 1.4857]
Budget	16 GiB	none	S	1.0000 [1.0000, 1.0000]
Telemetry	8 GiB / zero	none	P	1.3622 [1.2612, 1.4713]
Telemetry	8 GiB / original	none	S	1.4108 [1.3703, 1.4526]
Admission	8 GiB / zero	none	S	1.3556 [1.2485, 1.4719]
Admission	8 GiB / zero	2	S	1.0000 [1.0000, 1.0000]
Admission	8 GiB / zero	4	P	1.0474 [0.9513, 1.1533]
Qwen transfer	8 GiB	none	S	0.9969 [0.9227, 1.0772]
Qwen transfer	8 GiB	2	S	1.0013 [0.9971, 1.0056]
Qwen transfer	8 GiB	4	P	0.9904 [0.9266, 1.0586]
Qwen transfer	16 GiB	none	S	1.0034 [0.9926, 1.0144]
Qwen transfer	16 GiB	2	S	1.0027 [0.9942, 1.0112]
Qwen transfer	16 GiB	4	S	1.0148 [0.9684, 1.0634]
Llama transfer	Lower	budgeted	P	1.0000 [1.0000, 1.0000]
Llama transfer	Default	budgeted	S	1.0000 [1.0000, 1.0000]

F Additional hosted and behavioral analyses

F.1 Historical DeepSeek development and matched controls

The DeepSeek frontier completes eight policies in two scopes each: 3,072 executions of 192 inputs, with six successful retries. Predictions and correctness agree across all policies at all 384 scope-aligned coordinates.

Relative to CAPR, CAEP-SC saves 49.9% in cost and 51.1% in miss input, at a +3.7% makespan change. Its mean makespan is 0.758 of family-serial window one, which is marginally cheaper. This contrast combines warm-up and verification; Table 20 reports all arms and retry-sensitive comparisons. Native input order is itself cheaper than CAPR in these scopes (cost ratio 0.8223); the gate is cheaper and faster than native order under both returned-usage and retry-upper-cost ledgers.

Table 20: **Registered frontier means over two cache scopes.** Savings and makespan changes are paired geometric means relative to CAPR. "Pareto" marks frontier membership under retry-inclusive cost upper bounds, not significance or the existence of a cost saving.

Policy	Cost (USD)↓	Hit↑	Time (s)↓	Save↑	ΔT ↓	Pareto
Native	0.05298	31.8%	24.14	17.8%	+36.9%	–
CAPR	0.06443	16.6%	17.64	0.0%	0.0%	✓
Window 1	0.03177	59.9%	24.16	50.7%	+32.0%	✓
Window 2	0.05667	26.9%	17.93	12.0%	+1.6%	✓
Window 4	0.05939	23.3%	18.49	7.8%	+4.8%	–
Window 8	0.06358	17.7%	30.46	1.3%	+72.7%	–
CAEP-SC	0.03230	59.2%	18.30	49.9%	+3.7%	✓
CAEP-QA	0.03193	59.7%	20.65	50.4%	+15.2%	–

Native order has paired geometric cost ratio 0.8223 to CAPR (0.8243 under retry-cost upper bounds) and mean makespan 24.14 s, versus 18.30 s for the gate. Gate/native ratios are 0.6097 for cost, 0.6082 for retry-upper cost and 0.7579 for time. These are two development scopes. Neither the queue-aware arm nor an earlier belief-controller branch enters the current gate/blind estimates.

F.2 Historical ablation and inference audit

In the historical eight-block DeepSeek ablation, paired geometric cost ratios are 0.9826 [0.9691, 0.9964] for CAEP-SC/leader-only and 1.0000 [0.9868, 1.0114] for CAEP-SC/fixed-delay. The newer blind-two control additionally matches the initial ordinary-request count; the studies use different models, gates and workloads.

Re-executing the historical hosted analyzer reproduces its saved JSON exactly. The 32 arms contain 6,144 ordinary requests from 192 reused MWE-ECL inputs. CAEP-SC and CAPR agree on all 1,536 aligned coordinates, but fixed-delay changes one label in block 2 and leader-only changes one in block 6. The all-policy gate therefore remains failed. The 0.6884 s delay was frozen before this ablation as the median of 38 leader-to-verified-release intervals from the earlier development frontier; it is a prospective time-matched control, not a delay fitted to these outcomes. Retry counts are 2, 7, 2 and 1 for CAPR, leader-only, fixed-delay and CAEP-SC, respectively. An unobserved previous attempt is conservatively bounded using that request's all-miss reservation. Mean returned-response costs are USD 0.0637, 0.0332, 0.0326 and 0.0326, and mean makespans are 21.89, 24.41, 21.96 and 19.66 seconds, in that order. The historical geometric paired ratios differ from ratios of these arithmetic means; their intervals use the original 20,000-draw paired bootstrap over the eight observed blocks.

F.3 Alibaba creation confirmation and utility

The Alibaba study uses `qwen3.7-plus-2026-05-26` and the documented explicit-cache interface.[4] Its frozen, input-filtered subset contains four Qasper (Dasigi et al., 2021) and four NarrativeQA (Kočiský et al., 2018) prefix families from LongBench (Bai et al., 2024), totaling 29 QA requests per policy. Within every block the provider payload is identical across policies; assignments are balanced across two independently administered Alibaba Cloud accounts. The explicit cache marks a dedicated system-prefix message followed by a user suffix. This preserves concatenated visible text while changing the original single-user-message role structure.

[4] Alibaba Model Studio context-cache documentation, accessed 5 August 2026: `https://help.aliyun.com/en/model-studio/context-cache`.

Table 21: **Alibaba formal economic result.** Cost is reconstructed from provider read, creation, ordinary input, and output classes. Frozen CNY rates per million are 0.2, 2.5, 2 and 8, respectively. Output tokens are 254/264; the 766 ordinary input tokens contribute CNY 0.001532 in each arm.

Policy	Prompt	Cache read	Cache creation	Ordinary	Cost (CNY)↓
CAEP-CC	501,091	404,084	96,241	766	0.3249833
CAPR	501,091	159,050	341,275	766	0.8886415

The eight family-level cache-read-ratio effects are all positive, with mean 0.5520966. The archived conditional sign-flip statistic yields $p = 1/256$; its interpretation requires the assumptions in this section. Telemetry-priced cost falls by 63.43%, with a descriptive makespan ratio of 1.4975.

The prospective utility protocol generated new outputs from the same eight families. Its primary equal-family QA-F1 difference is -0.006381 against a frozen -0.02 noninferiority margin. Six of eight shifted family effects are positive, and the archived conditional sign-flip calculation over 256 patterns gives $p = 0.265625$, leaving noninferiority inconclusive. After a 372-second expiration gap, the same-request A/A stage measures 31.94% cross-policy response disagreement and a 35.69% same-policy A/A floor (descriptive excess -3.75 percentage points). Both stages complete 58/58 calls with zero errors and retries. The A/A comparison measures ordinary endpoint variation in exact response text.

Two earlier zero-retry utility protocols stopped prospectively after one transport disconnection each; no replacement families or partial scores entered the final analysis. The final protocol allowed one transport retry but used none. The two completed stages contain 116 responses at telemetry-priced cost CNY 2.28.

The allocation assigns four families to each account orientation, whereas the archived sign-flip calculation enumerates all 256 independent sign patterns. Its p-values are therefore exact only under independent, sign-symmetric family effects about the tested center, which is unverified for the utility margin; we report the paired estimates descriptively and leave noninferiority unestablished.

F.4 Provider contracts and economic interpretation

Table 22: **Provider-contract calibration and diagnostic controls.** DeepSeek and MiniMax-M3 use 80 requests each. The MiniMax prior-user-turn layout changes message structure; output checks compare the observed paired cases. The uncalibrated MiniMax output pilot is superseded by the 240-call grid (Section F.5); the cache calibration remains separate.

Provider/model	Condition	Cache gate	Output check	Use
DeepSeek V4 Flash	exact replay	20/20	20/20 agree	contract
DeepSeek V4 Flash	changed suffix	20/20	38/40 primary	contract
MiniMax-M3	original system block	0/20	n/a	negative
MiniMax-M3	prior-user-turn layout	20/20	3/40 differ (pilot)	transform
MiniMax-M2.7	auto/explicit paths	mixed	mixed	diagnostic
Fireworks	native/affinity paths	no reuse	matched	diagnostic

A common tariff multiplier preserves percentage saving within one window but scales absolute saving and the latency-value threshold. The dated DeepSeek snapshot records a $2\times$ multiplier for all billed items at 09:00–12:00 and 14:00–18:00 Beijing time, pending formal start notice at collection.[5]

Alibaba's documentation specifies Batch pricing at 50% of real-time rates and excludes cache discounts for Batch file-input calls.[6] Using the frozen 5 August price snapshot and holding the observed CAPR prompt/output ledger fixed, the resulting no-cache-discount counterfactual is CNY

[5]DeepSeek pricing, accessed 5 August 2026: `https://api-docs.deepseek.com/zh-cn/quick_start/pricing/`.

[6]Alibaba Model Studio Batch and context-cache documentation, verified 24 September 2026: `https://help.aliyun.com/zh/model-studio/batch-interfaces-compatible-with-openai/` and `https://help.aliyun.com/zh/model-studio/context-cache`.

0.50215: 43.5% below CAPR and 1.55× CAEP-CC's cost. The first reduction is less than 50% because CAPR already pays discounted cache-read rates on part of its prompt ledger, whereas Batch reprices the full prompt and output at 50% of standard rates. The calculation reprices the observed ledger; Batch execution and asynchronous makespan remain unmeasured. This differs from DeepSeek's announced same-channel peak multiplier, which applies to all billed items and therefore composes with automatic cache billing.

Within the DeepSeek ledger, Equation 3 yields an aggregate latency-value threshold of USD 0.0487/s. Independently, the Alibaba ledger yields CNY 0.0475/s. Below the setting-specific threshold, measured cost saving exceeds the valued delay. Each threshold uses its provider's billing currency. Post-hoc classification at hit-ratio cutoffs 0.80, 0.90 and 0.95 retains all 112 observed DeepSeek positive verification rows and 21 Alibaba follower reads. Alibaba's release rule checks creation, not these follower ratios. Reclassifying fixed traces does not measure performance under a changed gate, which would alter request timing and cache state.

A post-hoc zero-call check predicts the DeepSeek repeats within 3.2%/2.6% but overpredicts Alibaba by 51.9%. There, 7/8 families are within 1%; the nine-request outlier has 4 CAPR creations versus the nominal 8. Ordering and completion timing determine how many same-family requests are dispatched before the first cache-ready completion. A descriptive leave-one-family-out check reduces aggregate error to 0.03%, locating the mismatch in that family; the primary estimate retains all eight families.

F.5 A/A-CALIBRATED BEHAVIORAL AUDIT

Table 23: **Complete behavioral-disagreement audit.** Intervals resample matched cases conditional on each frozen execution. All-zero case contributions yield degenerate intervals. Doubao cross-layout counts include two matched replicate comparisons for each of 40 case–task cells.

Setting	Quantity	Observed	Case-resampling 95% interval
MiniMax	original A/A	10.0%	[0.0%, 22.5%]
MiniMax	prior-user A/A	12.5%	[2.5%, 25.0%]
MiniMax	prefix-system A/A	15.0%	[5.0%, 25.0%]
MiniMax	original–prior raw	13.75%	[5.0%, 23.75%]
MiniMax	original–prior excess	2.5%	[0.0%, 7.5%]
MiniMax	original–prefix raw	12.5%	[3.75%, 22.5%]
MiniMax	original–prefix excess	0.0%	[0.0%, 0.0%]
DeepSeek	byte-exact warm A/A	0/40 pairs	not estimated
Doubao 32K	original warm A/A	0/40 pairs	mechanically $[0, 0]$
Doubao 32K	prior-user warm A/A	0/40 pairs	mechanically $[0, 0]$
Doubao 32K	original–prior raw	0/80 comparisons	mechanically $[0, 0]$
Doubao 32K	original–prior excess	0.0%	mechanically $[0, 0]$
Doubao English	original warm A/A	0/40 pairs	mechanically $[0, 0]$
Doubao English	prior-user warm A/A	0/40 pairs	mechanically $[0, 0]$
Doubao English	original–prior raw	4/80 comparisons	[0.0%, 12.5%]
Doubao English	original–prior excess	5.0%	[0.0%, 12.5%]

For case c, let A_1, A_2 and B_1, B_2 be two executions of layouts a and b. Define $\widehat{D}_{aa,c} = \mathbf{1}[A_1 \neq A_2]$, $\widehat{D}_{bb,c} = \mathbf{1}[B_1 \neq B_2]$ and $\widehat{D}_{ab,c} = \frac{1}{2}\sum_{r=1}^{2} \mathbf{1}[A_r \neq B_r]$. The signed, unclipped estimator is

$$\widehat{\Delta}_{a,b} = |\mathcal{C}|^{-1} \sum_{c} \{\widehat{D}_{ab,c} - \tfrac{1}{2}(\widehat{D}_{aa,c} + \widehat{D}_{bb,c})\}. \tag{5}$$

Independent stationary layout-conditioned draws give

$$\mathbb{E}[\widehat{\Delta}_{a,b}] = |\mathcal{C}|^{-1} \sum_{c} \tfrac{1}{2}\|p_{a,c} - p_{b,c}\|_2^2. \tag{6}$$

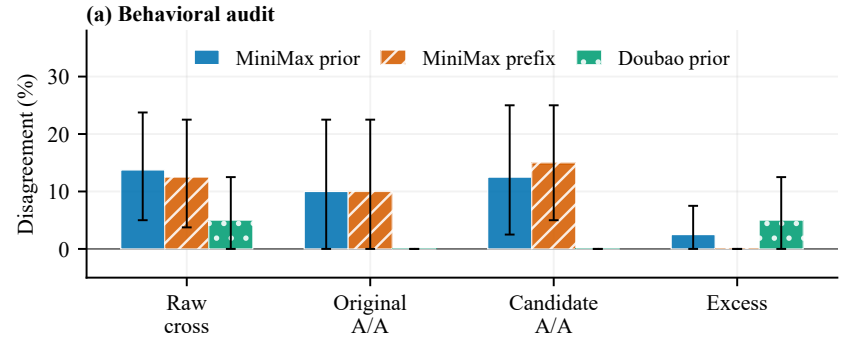
For each case, substitute $D_{ab} = 1 - \sum_y p_a(y)p_b(y)$ and $D_{aa} = 1 - \sum_y p_a(y)^2$ (likewise D_{bb}) to obtain the identity, then average cases. This is half the energy distance for the discrete metric, or half the squared MMD for the Kronecker kernel (Székely & Rizzo, 2013; Gretton et al., 2012). Finite estimates may be negative. Matched replicate indices are interleaved execution waves. Case-bootstrap

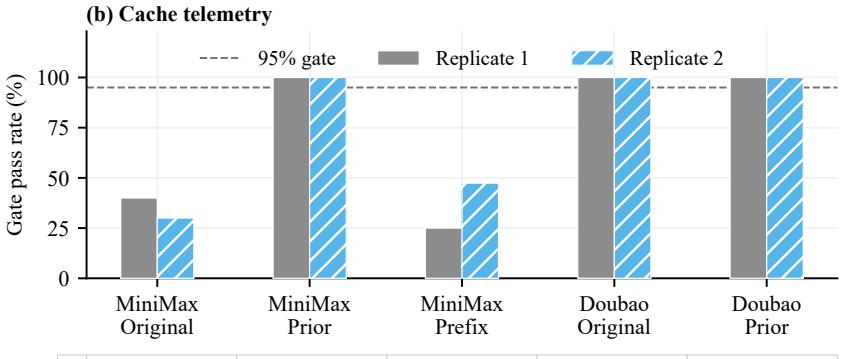
intervals describe frozen-case composition conditional on observed responses; dependence or drift can invalidate the independent stationary-draw identity. Equivalence requires a prospective margin and a test designed for that claim.

MiniMax's grid contains 20 matched Chinese cases, two tasks, three layouts and two measured repetitions (240 calls). The parsed alphabet is $\{\mathrm{A}, \mathrm{B}, \mathrm{PARSE_FAILURE}\}$; no parse failures occur. The completed Doubao Chinese and prospectively selected English grids use 32K contexts and paired refresh–repeat–repeat blocks. Cache-only repair rules were frozen before repair calls: the first passing measurement replaces a failed slot, without using response text or labels. Chinese/English repairs cover 15/14 slots; all original and repair calls are retained. Stopped 64K and initial English attempts remain excluded from completed-stage estimates. One MiniMax prefix-system repetition lacks cache telemetry but retains its parsed prediction in behavioral analysis.

The MiniMax prefix-system excess contribution is exactly zero for every case, so its case-resampling interval is algebraically degenerate without clipping. The DeepSeek and Chinese Doubao grids also have zero observed contributions, producing $[0, 0]$ under case resampling. The English Doubao grid has nonzero cross-layout disagreement and an excess interval of [0.0%, 12.5%]. Uncertainty over future hosted-serving repeats requires independent execution repetitions.

Sensitivity. An all-cross-pairs sensitivity analysis and declared categorical simulations, including temporal-block drift, illustrate why case resampling conditional on one execution does not quantify uncertainty over future serving windows.





R1	8/20	20/20	5/20	40/40	40/40
R2	6/20	20/20	9/19	40/40	40/40

Figure 5: **Behavioral calibration and cache telemetry across the MiniMax 128K and prospective Doubao English 32K audits.** Error bars resample matched cases conditional on each frozen execution. Zero-width intervals result from identical observed case contributions. The lower panel reports gate pass rates; the aligned count rows give passing/observed responses for each replicate (R1/R2), with Doubao denominators fixed by its expected measured slots.