TEMPORAL WORKING SETS FOR SPARSE ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse attention algorithms output the attention support over a small subset of input tokens, but finding this subset can require full cache search at each step in the decoder. We prove that this search has further structure not captured by the sparsity of attention support compared to previous attention supports. Larger working sets in two pretrained models are useful for finding subsequent attention sets. Query permutation against fixed memory reduces working set lifetimes from 22.9 and 19.7 to 4.3 and 3.9 steps for the Llama-3.1-8B and Qwen3-8B models respectively, without affecting each query's answer. Recurrent Attention Routing (RAR) maintains working sets and updates bounds from previous page visits instead of learning these bounds. Each query ranks the best set of current candidates, the best scores for new keys, and expands attention search to pages that may change its top-k. We prove that this algorithm exhaustively selects all relevant tokens, derive sufficient conditions to reuse working sets by margins in query ranks and query displacements, and show a RAR speedup in routing with 128K context for $3.9\times$ and $3.6\times$, and a $1.4\times$ decoding speedup for both models. All 65,069,056 selected head routes coincide with exhaustive selection. Temporal working sets lower the cost of computing each new attention support without training.

1 INTRODUCTION

Sparse attention reduces the cost of long-context language modeling by restricting the positions used in value aggregation (Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020). For query-dependent selection, locating those positions can still require scoring the entire key cache. In Llama-3.1-8B at a 128K sequence budget, exhaustive top-k routing takes 13.17 ms of a 31.73 ms decoding step. Much of the cost remains in determining which values to read.

Query-aware page selection and asymmetric indexing reduce this cost by directing each query toward competitive keys (Tang et al., 2024; Mazaré et al., 2025). Decoding also produces a sequence of related retrieval problems. Successive tokens in a code explanation may select different definitions and call sites from the same passages. Earlier queries have already identified many competitive keys and measured the scores of surrounding pages. Reusing those measurements offers a way to compute each new selection with less access to the cache.

We study this possibility through *temporal working sets*. Each working set contains m candidate keys from which a head selects its current top k, with $m > k$. Across book, report, question-answering, and code workloads, sets of size $4k$ remain sufficient for 11.7 to 29.3 steps on average. Even among queries in the lowest quartile of consecutive-support overlap, 42.7% of routes require no additional old-key scoring. Changes in the selected tokens often remain contained within a longer-lived candidate range.

To measure the contribution of query order, we permute recorded queries against a fixed memory. Every query retains its scores and answer. Working-set lifetime falls from 22.9 to 4.3 steps in Llama-3.1-8B and from 19.7 to 3.9 steps in Qwen3-8B. Routing time increases by 96.8% under recurrent execution, compared with 1.6% for static page selection. At an unchanged candidate budget, discarding previous page measurements increases routing time by 31.4%. Both candidate retention and information about the remaining cache contribute to temporal reuse.

We develop *Recurrent Attention Routing* (RAR) to use this structure in pretrained language models (Figure 1). Each head retains a working set and score maxima from earlier visits to other key pages. Current queries rerank the candidates and newly appended keys. Stored maxima and query

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.

displacement provide upper bounds for the remaining pages, and only pages that can affect the current top-k are revisited. Extensive expansions trigger a new working set. Every step returns the support and scores obtained by exhaustive selection, with no additional model training.

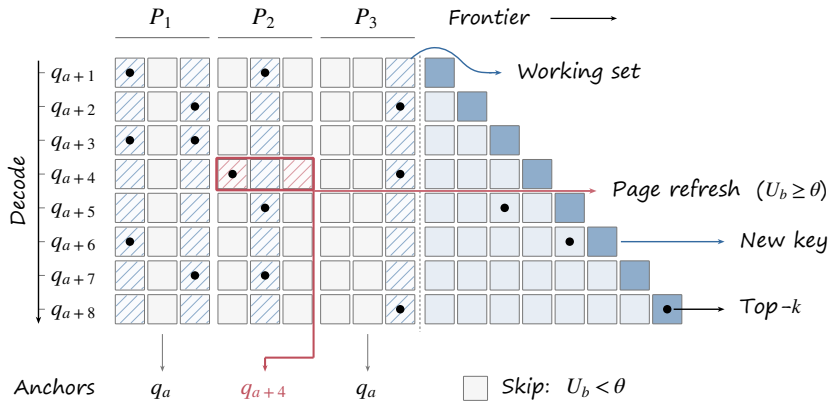

Figure 1: Working-set reuse and selective page refresh in RAR. Top-k selections change while working-set membership remains fixed. A page revisit updates its anchor and leaves other page anchors unchanged.

Rank margins describe how long the candidate range can remain sufficient as scores change. Page-level bounds relate the age of a stored maximum to its usefulness for a later query. The resulting work bound accounts for local scoring, page expansions, and working-set renewals over a decoding trajectory. On Llama-3.1-8B and Qwen3-8B, the resulting implementation reduces 128K routing latency to 3.38 ms and 4.19 ms and accelerates decoding by $1.4\times$. Supports agree across all 65,069,056 evaluated head routes, as do the paired greedy and stochastic completions.

Our contributions are threefold: (i) We characterize temporal working-set persistence in language-model attention through workload measurements and a fixed-memory query-order intervention. (ii) We derive conditions for candidate reuse and page refresh, and construct a head-local routing algorithm that preserves the exhaustive top-k under causal cache growth. (iii) We establish the computational benefit on two pretrained model families through matched retrieval baselines, component ablations, and complete-request measurements.

2 Temporal Structure in Attention Retrieval

2.1 The Retrieval Problem

For one attention head, let $q_t \in \mathbb{R}^d$ denote the query and $\{(z_i, v_i)\}_{i=1}^{N_t}$ its visible keys and values. Queries and keys include the model's normalization and positional encoding (Su et al., 2021). Suppressing layer and head indices, token-sparse attention restricts scaled dot-product attention (Vaswani et al., 2017) to

$$s_t(i) = q_t^\top z_i, \qquad S_t = \mathrm{TopK}_{i \leq N_t}(s_t(i), k), \qquad o_t = \sum_{i \in S_t} \frac{e^{s_t(i)/\sqrt{d}}}{\sum_{j \in S_t} e^{s_t(j)/\sqrt{d}}} v_i. \tag{1}$$

Logical token indices resolve score ties in increasing order. Decoding appends one key per step and leaves existing keys unchanged. We optimize the computation of S_t with the scores and value aggregation in Equation (1) fixed.

At renewal step a, the router selects E_a, the highest-scoring m keys of its indexed prefix. Newly appended positions form F_t. The current route can be computed from these candidates whenever

$$S_t \subseteq E_a \cup F_t. \tag{2}$$

Selections can change within this candidate range. On a fixed memory, we measure consecutive-support overlap $|S_t \cap S_{t-1}|/k$ and working-set survival

$$P_m(h) = \Pr[S_{a+j} \subseteq E_a \text{ for every } 1 \leq j \leq h]\,. \tag{3}$$

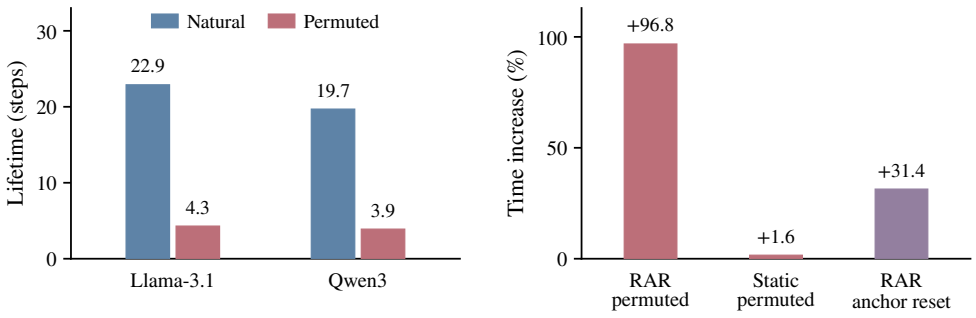

Figure 2: Temporal order sustains reuse. Permutation shortens working-set lifetimes (left) and nearly doubles RAR routing time (right).

Reported lifetime is the restricted mean of this survival curve over a 64-step horizon. Local-only steps score the working set and frontier without visiting excluded pages.

Table 1: Temporal working-set statistics. Lifetimes are measured in decoding steps.

Workload	Support overlap (%)	Working-set lifetime $m = 2k$	Working-set lifetime $m = 4k$	Local-only (%)	Renewals per 100 steps
Book continuation	75.8	6.7	20.8	69.4	4.8
Report continuation	77.1	8.2	24.6	73.8	3.7
Question answering	68.9	4.1	11.7	56.2	8.9
Code completion	79.6	9.5	29.3	77.6	3.1

2.2 Persistence across decoding steps

Consecutive routes share 68.9–79.6% of their selected keys (Table 1). Working sets of $4k$ candidates support an average of 11.7 new selections in question answering and 29.3 in code completion. Increasing the width from $2k$ to $4k$ extends lifetime by approximately a factor of three across workloads. The added candidates accommodate changes in the rankings before retrieval must leave the working set.

Queries in the lowest quartile of support overlap have mean candidate coverage $|S_t \cap (E_a \cup F_t)|/k = 85.4\%$. Of these routes, 42.7% require no additional old-key scores. Low overlap with the preceding answer is compatible with sufficient coverage by a larger candidate set. In held-out cost prediction, past support overlap gives $R^2 = 0.19$. Including the anchor margin, query displacement, epoch age, and prior expansion fraction increases R^2 to 0.64. Appendix C specifies the prediction task.

2.3 Query-order intervention

We hold the prefill memory and query multiset fixed, preserving every query's scores and top-k support. Random permutation reduces working-set lifetime from 22.9 to 4.3 steps for Llama-3.1 and from 19.7 to 3.9 for Qwen3 (Figure 2). Under permutation, RAR scores an additional 43.7 percentage points of the keys and takes 96.8% longer. Static page selection changes by 0.3 percentage points and 1.6%, respectively.

Resetting page anchors after each query increases routing time by 31.4% at the same candidate budget. Previous page measurements contribute beyond candidate retention. The ordering and anchor-reset interventions attribute reuse to the retained candidates and page-specific score information.

3 RECURRENT ATTENTION ROUTING

3.1 WORKING SETS OVER A GROWING CACHE

RAR maintains an independent route for each query head. Keys occupy chronological pages of capacity B. At renewal step a, the indexed prefix, working set, and append frontier are

$$n_a = B\lfloor N_a/B \rfloor, \qquad E_a = \mathrm{TopK}_{i \le n_a}(q_a^\top z_i, m), \qquad F_t = \{n_a + 1, \dots, N_t\}. \tag{4}$$

Both the prefix and E_a stay fixed until the next renewal. Frontier F_t contains the partial page at renewal and all subsequent keys. For an old page P_b, define $D_b = P_b \setminus E_a$. Empty subsets are skipped. The router uses exhaustive selection when the completed prefix contains at most m keys. Heads renew independently. Completed-page geometry remains shared across heads with different renewal times because their stored keys do not change.

Each page stores a center μ_b, radius $r_b \ge \max_{i \in P_b} \|z_i - \mu_b\|_2$, and coordinate extrema $l_{bj} \le z_{ij} \le u_{bj}$. Query heads in the same KV group share these summaries (Ainslie et al., 2023). Each head also records a page anchor p_b and the maximum residual score obtained on its last visit,

$$v_b = \max_{i \in D_b} p_b^\top (z_i - \mu_b). \tag{5}$$

Renewal initializes $p_b = q_a$. Subsequent page visits refresh both p_b and v_b, allowing different pages to retain measurements from different steps.

Frontier keys are scored until renewal, including keys that complete a physical page. Local expansion leaves E_a unchanged. Renewal rebuilds all excluded-page maxima under the new membership mask (Appendix B.3).

3.2 UPDATING THE SEARCH

Candidate scoring over $C_t = E_a \cup F_t$ gives a kth score θ_t. At renewal, the router records two additional quantities,

$$\beta_a = \max_{i \le n_a,\, i \notin E_a} q_a^\top z_i, \qquad R_a = \max_{i \le n_a} \|z_i\|_2 . \tag{6}$$

For each old key outside E_a, the current score has upper bound

$$G_t = \beta_a + \|q_t - q_a\|_2 R_a. \tag{7}$$

When $G_t < \theta_t$, candidates already contain the current top-k. This test requires one query displacement and the stored maximum and norm bound.

Otherwise, the router evaluates page-specific bounds. Expanding $q_t^\top z_i$ about the stored page center and anchor gives

$$U_b^{\mathrm{temp}}(q_t) = q_t^\top \mu_b + v_b + \|q_t - p_b\|_2 r_b, \tag{8}$$

$$U_b^{\mathrm{box}}(q_t) = \sum_{j=1}^{d} \max(q_{tj} l_{bj}, q_{tj} u_{bj}), \qquad U_b(q_t) = \min\{G_t, U_b^{\mathrm{temp}}, U_b^{\mathrm{box}}\}. \tag{9}$$

Both expressions upper-bound every score in D_b. The temporal expression uses the maximum observed on a preceding visit. Coordinate extrema supply a second bound (Tang et al., 2024). Pages still capable of changing the selection form

$$\mathcal{U}_t = \{b : D_b \neq \varnothing,\ U_b(q_t) \ge \theta_t\}, \qquad C_t^+ = C_t \cup \bigcup_{b \in \mathcal{U}_t} D_b. \tag{10}$$

Scoring C_t^+ determines the current top-k, which alone enters attention. Each visited page is refreshed using the current query.

3.3 RENEWAL AND EXECUTION

We use $k = 2048$, $m = 4096$, $B = 128$, and maximum epoch length $H = 64$. A head renews at age H or when unresolved pages contain at least half of the excluded prefix. Renewal scores the

remaining keys, reuses current scores, and constructs a new E_a. Applying its membership mask precedes recomputation of all page maxima. Each continuation reranks candidates and tests the bounds; unresolved pages trigger local expansion or renewal (Algorithm 1, Appendix B.3).

Centers, radii, and coordinate extrema are shared within each KV group. Each query head stores its last $H + 1$ queries in a ring buffer. Pages contain indices into this buffer, allowing query distances to be computed once per distinct anchor. Candidate lists, page masks, and renewal decisions remain on the GPU. Requests from heads sharing a KV head are coalesced when they access the same key tiles, following the memory-traffic considerations of attention kernels and paged caches (Dao et al., 2022; Dao, 2024; Kwon et al., 2023).

Candidate scores feed a common selected-softmax/value kernel. All support-preserving implementations use the same dot-product reduction, tie rule, and ordering of selected positions. Bounds are outward-rounded before comparison. Appendix B specifies the numerical evaluation and state layout.

4 Working-set stability

4.1 Selection consistency

Theorem 1. *Assume that the bounds upper-bound all associated excluded scores and every frontier key is scored. RAR returns the exhaustive top-k for any query sequence and append-only cache.*

Proof. Renewal and short-context steps score all visible keys. Otherwise, candidate scores establish threshold θ_t. Every skipped key scores strictly below this threshold. Adding unresolved pages can only raise the kth candidate score, so skipped keys cannot enter the final selection. Equality triggers a page visit and preserves the tie rule. Page membership stays fixed within each epoch, and frontier scoring covers every new key. Applying this argument at each step proves the claim. □

One expansion suffices because the initial candidate threshold lower-bounds the final threshold. After scoring unresolved pages, the router can return the top-k without repeating the metadata pass.

4.2 Working-set margins

Let $s_{(j)}(q_a)$ denote the jth largest score in the prefix at renewal. Define the working-set margin

$$\gamma_{k,m}(q_a) = s_{(k)}(q_a) - s_{(m+1)}(q_a). \tag{11}$$

For $m = k$, this is the immediate selection-boundary gap. For $m > k$, it measures the gap between the kth score and the highest score outside the working set.

Proposition 1. *Assume $\|z_i\|_2 \le R$ for old keys. If*

$$2R \|q_t - q_a\|_2 < \gamma_{k,m}(q_a), \tag{12}$$

then $S_t \subseteq E_a \cup F_t$ after any number of causal key insertions.

Proof. Write $\delta = \|q_t - q_a\|_2$. At t, every anchor top-k key scores at least $s_{(k)}(q_a) - R\delta$, and every old key outside E_a scores at most $s_{(m+1)}(q_a) + R\delta$. Equation (12) places at least k keys in E_a above every old excluded key. Adding F_t can replace members of that top-k, but cannot improve an old excluded key's rank. □

Scores near rank k can exchange order within the margin $\gamma_{k,m}$. Widening E_a increases this tolerance and candidate-scoring cost. Section 5.4 measures both effects.

4.3 Persistence of page maxima

For an excluded page, let $M_b(q) = \max_{i \in D_b} q^\top z_i$ denote its true maximum and $e_b(q) = U_b^{\text{temp}}(q) - M_b(q)$ the slack of its temporal bound.

Proposition 2. *For a page last visited at query p_b,*

$$0 \le e_b(q) \le 2r_b \left\| q - p_b \right\|_2 . \tag{13}$$

In particular, the temporal bound excludes that page whenever

$$\theta_t - M_b(q_t) > 2r_b \left\| q_t - p_b \right\|_2 . \tag{14}$$

Proof. The residual maximum $f_b(q) = \max_{i \in D_b} q^\top (z_i - \mu_b)$ is r_b-Lipschitz. Substituting $M_b(q) = q^\top \mu_b + f_b(q)$ yields $e_b(q) = f_b(p_b) - f_b(q) + r_b \left\| q - p_b \right\|_2$ and the stated bounds. Under Equation (14), $U_b^{\text{temp}}(q_t) < \theta_t$. □

Pages remain excluded when their score gaps exceed the displacement-induced slack. Visiting a page resets this slack to zero in real arithmetic. Each page can consequently retain its own anchor until the current query requires another visit.

Table 4 evaluates candidate width and page refresh with fixed page geometry.

4.4 Trajectory-dependent work

For a head whose indexed prefix exceeds m, let $\mathcal{J}$ contain its renewal steps, $f_t = |F_t|$, and r_t count old excluded keys scored during local expansion. Reusing scores at renewal gives the total key-score count

$$W = \sum_{t \in \mathcal{J}} N_t + \sum_{t \notin \mathcal{J}} (m + f_t + r_t), \qquad f_t < B + H. \tag{15}$$

Steps requiring page bounds incur an additional $O(C_t d + Hd)$ cost, where $C_t = n_a / B$. Equation (15) and this metadata cost jointly account for routing execution.

Let $V_T = \sum_{t=2}^{T} \left\| q_t - q_{t-1} \right\|_2$. If renewal margins are at least $\Gamma > 0$ and all old key norms are at most R, then in real arithmetic

$$|\mathcal{J}| \le 1 + \frac{2RV_T}{\Gamma} + \left\lfloor \frac{T-1}{H} \right\rfloor . \tag{16}$$

A non-scheduled renewal requires displacement of at least $\Gamma/(2R)$ from its anchor. Disjoint epoch paths consume at most V_T in total, and scheduled renewals contribute the final term. Appendix A gives the derivation. This bound permits different routing costs for different orderings of the same query collection.

5 Experiments

5.1 Models, workloads, and comparisons

We evaluate frozen Llama-3.1-8B-Instruct and Qwen3-8B checkpoints (Grattafiori et al., 2024; Yang et al., 2025a). Every query head uses token-sparse attention during decoding, with dense FlashAttention-2 prefill (Dao, 2024). Experiments run on one H100-SXM 80GB with BF16 weights and KV caches. Qwen3 uses non-thinking mode and static YaRN beyond 32K (Peng et al., 2024). Primary comparisons fix $k = 2048$.

LongBench provides Qasper, HotpotQA, 2WikiMultihopQA, GovReport, LCC, and RepoBench-P (Bai et al., 2024). Each task contributes 32 fixed examples, evaluated under a 32K total sequence budget. QA uses F1, summarization uses ROUGE-L, and code completion uses edit similarity. Six RULER configurations cover single-needle, multi-key, multi-value, variable-tracking, and word-aggregation tasks at 32K and 128K, with 16 examples per configuration (Hsieh et al., 2024). Timing uses natural PG-19 continuations at 8K, 32K, 64K, and 128K with 128 or 512 output tokens (Rae et al., 2020). Sequence budgets include generated tokens. Appendix C gives sampling and execution details.

Routing baselines share the score operator and selected-attention kernel. *Exhaustive Top-k* scores every key using the calibrated radix or framework selection kernel (Li et al., 2024a). *Static-Page*

Table 2: Routing at 128K with 512 generated tokens and batch size one. Latencies are ms/token and traffic is GiB/token.

Model	Method	Route ↓	Decode ↓	Keys (%) ↓	Traffic ↓
Llama-3.1 8B	Exhaustive Top-k	13.17	31.73	100.0	8.00
	Static-Page	8.84	27.41	45.3	5.18
	Buffered-MIPS	6.39	24.96	28.7	3.46
	EMA-Proposal	5.22	23.78	22.4	2.83
	RAR	**3.38**	**21.96**	**14.6**	**2.05**
Qwen3-8B	Exhaustive Top-k	15.08	35.84	100.0	9.00
	Static-Page	10.31	31.08	48.1	6.22
	Buffered-MIPS	7.55	28.31	32.1	4.42
	EMA-Proposal	6.29	27.05	25.3	3.61
	RAR	**4.19**	**24.91**	**17.1**	**2.71**

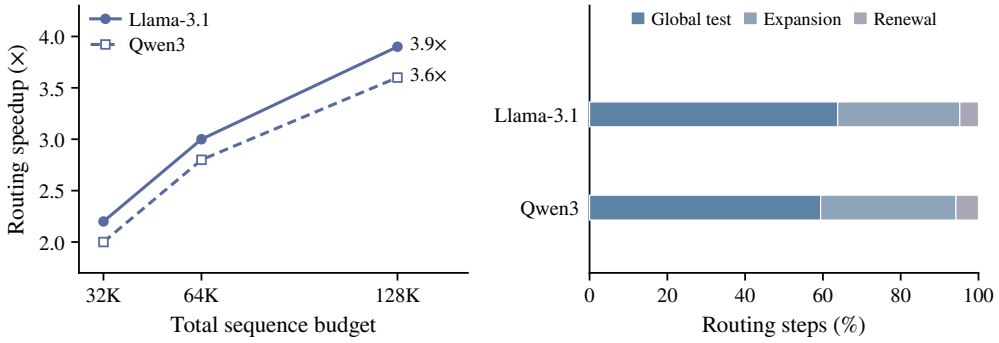

Figure 3: Routing gains grow with context. Speedup over exhaustive top-k (left) and route-resolution frequencies at 128K (right).

uses coordinate bounds and visits all pages that could alter the current top-k. *Buffered-MIPS* retains m candidates and renews globally when Equation (7) fails. *EMA-Proposal* draws candidates from a decay-0.9 average of previous supports and completes selection using static page bounds. Task comparisons additionally include full attention and Quest. Appendix C.5 defines each implementation.

Timing uses teacher-forced natural continuations and matched reader kernels. Task evaluation uses autoregressive generation. Request time includes initialization and maintenance.

5.2 Faster routing and decoding

At 128K, RAR reduces routing latency from 13.17 to 3.38 ms for Llama and from 15.08 to 4.19 ms for Qwen3 (Table 2). These correspond to speedups of $3.9\times$ and $3.6\times$. Relative to EMA-Proposal, routing takes 35.2% and 33.4% less time. Updating page-specific maxima reduces the work beyond candidate retention.

Decoding latency falls from 31.73 to 21.96 ms for Llama and from 35.84 to 24.91 ms for Qwen3, a $1.4\times$ speedup on each model. RAR scores 14.6% and 17.1% of cached keys and reads 2.05 and 2.71 GiB of key-cache data per output token. Grouped-query coalescing reduces traffic by 27.9% and 25.6% relative to separate head requests.

Routing speedup increases with context length (Figure 3). Llama achieves $2.2\times$, $3.0\times$, and $3.9\times$ at 32K, 64K, and 128K. Qwen3 achieves $2.0\times$, $2.8\times$, and $3.6\times$. At 128K, the global test resolves 63.8% and 59.4% of routes. Local expansion handles 31.4% and 34.8%, and renewal accounts for the remaining 4.8% and 5.8%.

Including prefill and state construction, request latency decreases by 20.3% for Llama and 18.1% for Qwen3. At batch size two and 64K, throughput is 96.8 and 87.4 generated tokens per second.

Table 3: Long-context task performance. Sparse methods use $k = 2048$.

Model	Attention	RULER 32K	RULER 128K	QA F1	Report ROUGE-L	Code Similarity
Llama-3.1 8B	Full	93.4	63.2	49.7	34.8	51.5
	Quest	90.8	55.1	47.3	34.1	49.8
	Exhaustive Top-k	92.1	58.7	47.9	34.6	50.6
	RAR	92.1	58.7	47.9	34.6	50.6
Qwen3-8B	Full	95.1	70.8	46.2	33.1	67.9
	Quest	92.7	63.2	44.4	32.2	65.7
	Exhaustive Top-k	94.0	66.5	45.1	32.5	66.6
	RAR	94.0	66.5	45.1	32.5	66.6

Table 4: Routing ablations at 64K. Time is relative to RAR, metadata is MiB/token, and state is MiB.

Variant	Time	Keys (%)	Metadata	State
RAR	**1.00**	17.6	68.9	239.7
No global continuation test	1.23	17.6	193.4	239.7
Static page bounds only	1.57	33.2	193.4	217.3
No page-local refresh	1.34	25.6	101.7	222.6
No query-history encoding	1.08	17.6	68.9	348.9
Renewal on every insertion	2.91	100.0	8.1	218.4

Additional state occupies 239.7 and 269.6 MiB per request. Appendix D reports request-length and layer-stratified measurements.

5.3 Task performance and route agreement

RAR and exhaustive top-k produce identical benchmark scores on both checkpoints (Table 3). QA reports mean F1 across three datasets, Report gives GovReport ROUGE-L, and Code averages two edit-similarity scores. Supports agree on all 65,069,056 evaluated head routes, and every paired greedy and stochastic completion agrees. With common scores and value-reduction ordering, the per-head result of Theorem 1 extends through the layers and decoding steps.

Attention support accounts for the differences across the other task conditions. Llama scores 63.2 on 128K RULER with full attention, 55.1 with Quest, and 58.7 with token-level top-k. RAR returns the token-level support and retains the 58.7 score. Throughout the cost comparisons, k fixes the attention operator and m sets the candidate budget used to compute it.

5.4 Component analysis

Removing the global test leaves the scored-key fraction unchanged at 17.6%, yet increases metadata traffic from 68.9 to 193.4 MiB per token and routing time by 23% (Table 4). The test saves a metadata pass on routes whose candidate threshold already exceeds the global bound.

Static page bounds require scoring 33.2% of keys, compared with 17.6% under temporal bounds, and increase routing time by 57%. Keeping temporal bounds without page-local refresh raises routing time by 34%. Both comparisons use the same page geometry. Previously measured maxima reduce expansion, and refreshing them preserves that reduction as queries change.

Query-history encoding reduces additional state from 348.9 to 239.7 MiB. Storing a separate anchor vector per page increases routing time by 8%. Renewing on every insertion scores the full cache and takes $2.91\times$ the time of RAR. Explicit frontier scoring preserves the usefulness of old-prefix summaries as new keys arrive.

Increasing m/k from one to two reduces routing time from 3.27 to 2.31 ms and renewal rate from 13.6% to 5.1% (Figure 4). At widths four and eight, renewal rates fall to 2.1% and 0.8%, but routing time rises to 2.47 and 3.08 ms. Additional candidate scoring eventually exceeds the saving from fewer renewals. Width two gives the lowest combined cost at this setting.

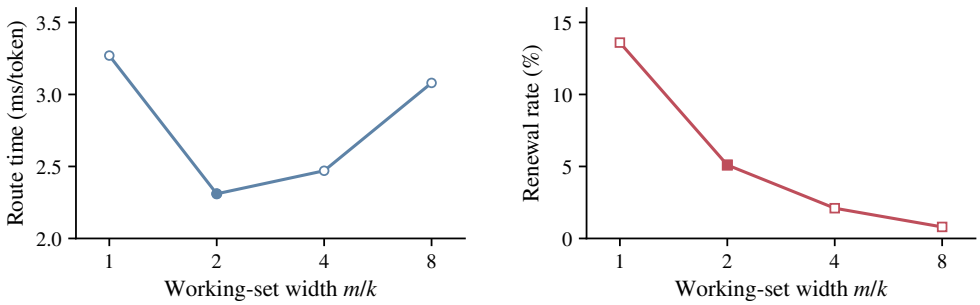

Figure 4: Wider sets reduce renewals but need not route faster. Routing time (left) and renewal rate (right) at 64K; filled markers denote the default $m = 2k$.

Page capacities 64, 128, and 256 give routing times of 2.42, 2.31, and 2.57 ms. Smaller pages require more metadata, and larger pages retrieve more keys per expansion. We use the default capacity of 128 and width of two throughout the primary evaluation.

6 Related work

Selecting attention positions. Sparse Transformers, Longformer, and Big Bird impose structured access patterns that reduce the number of query–key interactions (Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020). Reformer uses locality-sensitive hashing to form query–key groups, and Routing Transformers use online clustering to restrict content-dependent attention (Kitaev et al., 2020; Roy et al., 2021). For pretrained models, Quest ranks cache pages with coordinate-wise key extrema (Tang et al., 2024). SAAP clusters keys offline and trains a query classifier to select the clusters to search, accommodating the difference between query and key distributions (Mazaré et al., 2025).

Retaining information during generation. H_2O retains recent tokens and keys with high accumulated attention scores, StreamingLLM retains attention sinks and recent tokens, and SnapKV derives head-specific cache selections from a prompt observation window (Zhang et al., 2023; Xiao et al., 2024; Li et al., 2024b). TidalDecode propagates selected positions across attention layers (Yang et al., 2025b). PRR predicts blocks from previous selections and overlaps speculative attention with the current selector, then incorporates the additional selected blocks (Wang et al., 2026). Transformer-XL carries hidden states across segments, and linear attention maintains recurrent feature statistics for its kernelized attention computation (Dai et al., 2019; Katharopoulos et al., 2020).

Locality and memory access. Denning's working-set model describes a process through the pages accessed within a recent execution interval (Denning, 1968). Continuous nearest-neighbor algorithms maintain geometric regions within which query answers remain valid and update them as queries or data change (Hasan et al., 2009). At the accelerator level, FlashAttention tiles attention to reduce transfers between high-bandwidth memory and on-chip storage, FlashAttention-2 improves work partitioning, and PagedAttention allocates KV storage in noncontiguous blocks (Dao et al., 2022; Dao, 2024; Kwon et al., 2023). Grouped-query attention shares key and value heads to reduce decode-time memory access (Ainslie et al., 2023).

7 Conclusion

Temporal working sets allow successive attention supports to be computed from a persistent candidate range. RAR maintains these sets and updates page-score bounds, returning the same top-k as exhaustive selection. Rank margins relate their lifetime to query displacement. Query-order interventions and controlled ablations attribute the measured savings to temporal reuse. On Llama-3.1-8B and Qwen3-8B, RAR accelerates 128K routing by $3.9\times$ and $3.6\times$ and decoding by $1.4\times$. Working-set persistence reduces repeated retrieval across a decoding trajectory without changing its attention supports.

AI USE STATEMENT

We used large language models (LLMs) to improve grammar and correct spelling errors in the manuscript. The authors take full responsibility for the final content of this work.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. GQA: Training Generalized Multi-Query Transformer Models from Multi-Head Checkpoints. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901, 2023.

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 *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 3119–3137, 2024.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The Long-Document Transformer. *arXiv:2004.05150*, 2020.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating Long Sequences with Sparse Transformers. *arXiv:1904.10509*, 2019.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988, 2019.

Tri Dao. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. In *International Conference on Learning Representations*, 2024.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. In *Advances in Neural Information Processing Systems*, 2022.

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 *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pp. 4599–4610, 2021.

Peter J. Denning. The Working Set Model for Program Behavior. *Communications of the ACM*, 11(5):323–333, 1968.

Aaron Grattafiori et al. The Llama 3 Herd of Models. *arXiv:2407.21783*, 2024.

Daya Guo, Canwen Xu, Nan Duan, Jian Yin, and Julian McAuley. LongCoder: A Long-Range Pre-trained Language Model for Code Completion. In *Proceedings of the International Conference on Machine Learning*, pp. 12098–12107, 2023.

Mahady Hasan, Muhammad Aamir Cheema, Xuemin Lin, and Ying Zhang. Efficient Construction of Safe Regions for Moving kNN Queries over Dynamic Datasets. In *Advances in Spatial and Temporal Databases*, pp. 373–379, 2009.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. In *Proceedings of the International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. RULER: What's the Real Context Size of Your Long-Context Language Models? In *Conference on Language Modeling*, 2024.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. Efficient Attentions for Long Document Summarization. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pp. 1419–1436, 2021.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. In *Proceedings of the International Conference on Machine Learning*, pp. 5156–5165, 2020.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The Efficient Transformer. In *International Conference on Learning Representations*, 2020.

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 ACM Symposium on Operating Systems Principles*, pp. 611–626, 2023.

Yifei Li, Bole Zhou, Jiejing Zhang, Xuechao Wei, Yinghan Li, and Yingda Chen. RadiK: Scalable and Optimized GPU-Parallel Radix Top-K Selection. In *Proceedings of the International Conference on Supercomputing*, pp. 537–548, 2024a.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM Knows What You are Looking for Before Generation. In *Advances in Neural Information Processing Systems*, 2024b.

Tianyang Liu, Canwen Xu, and Julian McAuley. RepoBench: Benchmarking Repository-Level Code Auto-Completion Systems. In *International Conference on Learning Representations*, 2024.

Pierre-Emmanuel Mazaré, Gergely Szilvasy, María Lomelí, Francisco Massa, Naila Murray, Hervé Jégou, and Matthijs Douze. Inference-Time Sparse Attention with Asymmetric Indexing. *arXiv:2502.08246*, 2025.

NVIDIA. Floating Point and IEEE 754 Compliance for NVIDIA GPUs. *CUDA Documentation*. Accessed September 14, 2026.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. YaRN: Efficient Context Window Extension of Large Language Models. In *International Conference on Learning Representations*, 2024.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive Transformers for Long-Range Sequence Modelling. In *International Conference on Learning Representations*, 2020.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient Content-Based Sparse Attention with Routing Transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *arXiv:2104.09864*, 2021.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-Aware Sparsity for Efficient Long-Context LLM Inference. In *Proceedings of the International Conference on Machine Learning*, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, 2017.

Tianyu Wang, Gourav Rattihalli, Aditya Dhakal, Junbo Li, Zhiwei Ren, Dejan Milojicic, and Longfei Shangguan. Predict, Reuse, and Repair: Accelerating Dynamic Sparse Attention for Long-Context LLM Decoding. *arXiv:2606.30389*, 2026.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient Streaming Language Models with Attention Sinks. In *International Conference on Learning Representations*, 2024.

An Yang et al. Qwen3 Technical Report. *arXiv:2505.09388*, 2025a.

Lijie Yang, Zhihao Zhang, Zhuofu Chen, Zikun Li, and Zhihao Jia. TidalDecode: Fast and Accurate LLM Decoding with Position Persistent Sparse Attention. In *International Conference on Learning Representations*, 2025b.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, 2018.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big Bird: Transformers for Longer Sequences. In *Advances in Neural Information Processing Systems*, 2020.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H_2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models. In *Advances in Neural Information Processing Systems*, 2023.

A PROOFS AND STATE INVARIANTS

A.1 NOTATION AND INDEXED SUBSETS

All quantities below refer to one layer and query head unless a KV-group index is shown. A decoding step appends its current key before selecting the attention support. Stored keys include the checkpoint's normalization and positional transformation. The indexed prefix consists of complete chronological pages at the most recent renewal.

Table 5: Notation for one recurrent route. Symbols for the working set and indexed prefix retain their renewal index because their membership is unchanged within an epoch. Page anchors can be refreshed independently.

Symbol	Definition
q_t, z_i, v_i	Current query, cached key, and cached value
N_t, d, k	Visible token count, head dimension, and selected-token budget
a, n_a	Renewal step and completed-prefix boundary $B\lfloor N_a/B \rfloor$
E_a, m	Top-m working set selected from positions $1, \dots, n_a$
F_t	Positions $n_a + 1, \dots, N_t$, including the partial page and new keys
P_b, D_b	Physical key page and its excluded subset $P_b \setminus E_a$
μ_b, r_b, l_b, u_b	Page center, radius, and coordinate extrema over P_b
p_b, v_b	Page's stored query and maximum residual score over D_b
C_t, C_t^+	Initial and expanded candidate sets
θ_t	kth score in the initial candidate set
$G_t, U_b(q_t)$	Global complement bound and combined page bound
$\mathcal{U}_t$	Pages requiring additional key scoring at step t
$H, \mathcal{J}$	Maximum epoch length and renewal steps

For $n_a > m$, the visible keys have the disjoint partition

$$\{1, \dots, N_t\} = E_a \,\dot\cup\, F_t \,\dot\cup \bigdot_{b=1}^{n_a/B} D_b. \tag{17}$$

Each score belongs to one part of this partition. Initial candidate scoring covers $E_a \cup F_t$. Page expansion covers subsets D_b selected by the bound test. Renewal covers all parts and constructs a new partition. Empty D_b subsets need neither a maximum nor a page visit. The short-context branch covers $n_a \le m$ directly.

A.2 VALIDITY OF PAGE BOUNDS

For $i \in D_b$, expand the score around the stored anchor and page center,

$$q_t^\top z_i = q_t^\top \mu_b + p_b^\top (z_i - \mu_b) + (q_t - p_b)^\top (z_i - \mu_b) \tag{18}$$

$$\le q_t^\top \mu_b + v_b + \|q_t - p_b\|_2 \, \|z_i - \mu_b\|_2 \tag{19}$$

$$\le q_t^\top \mu_b + v_b + \|q_t - p_b\|_2 \, r_b. \tag{20}$$

This proves Equation (8). The coordinate inequality

$$q_{tj} z_{ij} \le \max(q_{tj} l_{bj}, q_{tj} u_{bj}) \tag{21}$$

holds for either sign of q_{tj} and yields Equation (9) on summation. Finally,

$$q_t^\top z_i = q_a^\top z_i + (q_t - q_a)^\top z_i \le \beta_a + \|q_t - q_a\|_2 \, R_a \tag{22}$$

for old excluded keys. Each bound covers D_b, so their minimum is valid.

Within an epoch, neither D_b nor its keys change. Updating a page's anchor to q_t after scoring D_b preserves the defining maximum in Equation (5). At renewal, the new membership mask is applied before recomputing maxima. Keys removed from a working set are consequently included in the new excluded-page summaries.

A.3 SELECTION AND CAUSAL INSERTION

The initial candidates contain at least k visible keys. Let their kth score be θ_t. If the expanded candidates have kth score θ_t^+, then $\theta_t^+ \geq \theta_t$. Every unvisited key scores strictly below θ_t, and hence below θ_t^+. A page touching the threshold is visited. This also handles equal-score keys under the fixed logical-index tie rule.

New keys enter the frontier even if they complete a physical page. The current head's indexed prefix remains unchanged until renewal. A new key can immediately enter the top-k, but cannot invalidate a bound whose domain is the fixed old prefix. If at least k old working-set keys outrank an old excluded key, adding further candidates cannot improve that excluded key's rank. This proves the insertion statement in Proposition 1.

The same argument gives full-model equivalence. At the first sparse decoding layer, identical inputs yield identical selected supports, scores, and value-reduction order. The output is the same, and induction gives identical inputs to the next layer. Induction over decoding steps then gives the same logits and, with common random numbers, the same sampled sequence. The operator throughout is Equation (1).

A.4 RENEWAL COUNT

Consider an epoch anchored at a. Under $\|z_i\|_2 \leq R$, its candidate threshold at t obeys

$$\theta_t \geq s_{(k)}(q_a) - R\delta, \qquad \delta = \|q_t - q_a\|_2 . \tag{23}$$

The global complement bound obeys

$$G_t \leq s_{(m+1)}(q_a) + R\delta. \tag{24}$$

If $2R\delta < \Gamma \leq \gamma_{k,m}(q_a)$, then $G_t < \theta_t$, so no non-scheduled renewal occurs. Each such renewal consumes at least $\Gamma/(2R)$ of query-path length from its anchor. Epoch intervals are disjoint, giving at most $2RV_T/\Gamma$ non-scheduled renewals. A scheduled renewal consumes H steps. Adding initialization yields Equation (16).

Equation (16) assumes a positive uniform working-set margin and a uniform key-norm bound. Outward-rounded numerical evaluation preserves the selection rule by including pages whose bounds reach the threshold.

A.5 QUERY REGIONS SUPPORTED BY A WORKING SET

Fix the old prefix at renewal step a. Let $T_a \subseteq E_a$ be its anchor top-k, and let $O_a = \{1, \ldots, n_a\} \setminus E_a$. Every query in the region

$$\Omega_a = \bigcap_{i \in T_a} \bigcap_{j \in O_a} \{q \in \mathbb{R}^d : q^\top (z_i - z_j) > 0\} \tag{25}$$

places at least k working-set keys above every old excluded key. This intersection of half-spaces gives a sufficient region for continued candidate coverage. Keys in $E_a \setminus T_a$ can move above members of T_a without leaving the region. The resulting support can differ from the anchor support.

The working-set margin supplies a ball contained in Ω_a. For $i \in T_a$ and $j \in O_a$,

$$q^\top (z_i - z_j) = q_a^\top (z_i - z_j) + (q - q_a)^\top (z_i - z_j) \tag{26}$$

$$\geq \gamma_{k,m}(q_a) - \|q - q_a\|_2 \, \|z_i - z_j\|_2 \tag{27}$$

$$\geq \gamma_{k,m}(q_a) - 2R \, \|q - q_a\|_2 . \tag{28}$$

When $R > 0$ and the margin is positive, the open ball of radius $\gamma_{k,m}(q_a)/(2R)$ around q_a lies in Ω_a. Proposition 1 follows from this inclusion. Equations (25) and (12) describe sufficient conditions using, respectively, pairwise score differences and a common norm bound.

For fixed q_a and increasing m, $s_{(m+1)}(q_a)$ is nonincreasing. Hence $\gamma_{k,m}(q_a)$ is nondecreasing. Larger working sets admit at least as large a margin-based neighborhood, at the cost of scoring additional candidates on every continuation step. The measured width sweep evaluates this recurring cost together with renewal frequency.

A.6 CANDIDATE THRESHOLDS AND PAGE REFRESH

For a fixed candidate set C containing at least k keys, write $\theta_C(q)$ for its kth score and $R_C = \max_{i \in C} \|z_i\|_2$. Then

$$|\theta_C(q) - \theta_C(p)| \le R_C \|q - p\|_2 . \tag{29}$$

To prove the lower inequality, retain the k keys attaining the highest scores at p. Every one loses at most $R_C \|q - p\|_2$ under the new query, so at least k scores at q remain above $\theta_C(p) - R_C \|q - p\|_2$. Interchanging q and p gives the upper inequality.

Suppose page b is visited at p_b, and let $M_b(p_b)$ be its observed maximum. For a fixed candidate subset C,

$$U_b^{\text{temp}}(q) \le M_b(p_b) + (\|\mu_b\|_2 + r_b) \|q - p_b\|_2 , \tag{30}$$

$$\theta_C(q) \ge \theta_C(p_b) - R_C \|q - p_b\|_2 . \tag{31}$$

Consequently, the page remains excluded whenever

$$(R_C + \|\mu_b\|_2 + r_b) \|q - p_b\|_2 < \theta_C(p_b) - M_b(p_b). \tag{32}$$

Adding new candidates can only raise the current kth score. The same sufficient condition applies when C is retained as a subset of the growing candidate set. Equation (32) connects the duration of a page measurement to its score separation at the visit. It uses quantities associated with that page and candidate set, rather than a common age for the whole index.

At a visit, the temporal bound attains the page maximum in real arithmetic. For later queries, define $f_b(q) = \max_{i \in D_b} q^\top (z_i - \mu_b)$. Its r_b-Lipschitz property yields

$$U_b^{\text{temp}}(q) - M_b(q) = f_b(p_b) - f_b(q) + r_b \|q - p_b\|_2 \tag{33}$$

$$\in [0, 2r_b \|q - p_b\|_2]. \tag{34}$$

This supplies the full derivation of Proposition 2. Refreshing an inspected page sets its displacement term to zero at the current query. Other pages keep their own anchors.

A.7 UPPER-BOUND COMPOSITION AND THRESHOLD TIES

Each term in $U_b(q) = \min\{G_t, U_b^{\text{temp}}(q), U_b^{\text{box}}(q)\}$ covers the same excluded subset D_b. Taking their minimum retains the upper-bound property. A change in which term attains the minimum requires no change to the page contents or its stored anchor.

The comparison is strict when deciding to skip a page. If $U_b(q) = \theta_t$, the page is visited because a key with an equal score can enter the support under the logical-index ordering. This rule includes the zero-query case. All key scores then tie, and selection returns the smallest visible logical indices. Pages whose bounds touch zero are visited until those indices are recovered by the common selection operation.

A locally retrieved key can appear in the current support and remain outside E_a. Its residual maximum is refreshed with the rest of D_b. This keeps page domains fixed until renewal. Reassigning the working set occurs only after the full current search, when the excluded-page maxima are rebuilt under the new membership mask.

A.8 PER-HEAD AND MODEL-LEVEL WORK

The terms in Equation (15) count original-key scores for one head. At renewal, already computed candidate scores are retained, so the total is N_t rather than $N_t + m + f_t$. At continuation, the partition in Equation (17) gives $m + f_t + r_t$ without duplicate candidate scores.

Writing $u_a = N_a - n_a$, the frontier size after h single-token insertions is $f_{a+h} = u_a + h$. Since $0 \le u_a < B$ and scheduled renewal occurs at age H, continuation satisfies $f_t < B + H$. This bound holds even when inserted keys complete a physical page. The head's indexed boundary changes only at renewal.

Layer and model arithmetic is obtained by summing over heads and layers,

$$W_{\text{model}} = \sum_{\ell=1}^{L} \sum_{h=1}^{H_q^\ell} W_{\ell,h}. \tag{35}$$

Geometric metadata is shared within KV groups. Its evaluation depends on which heads require page tests, and physical key traffic depends on their union of requested addresses. These costs are reported separately from W_{model}.

B Numerical evaluation and GPU state

B.1 Reference arithmetic

Let $\widehat{s}(q,z) = \rho(q,z)$ denote the reference dot-product kernel. Queries and keys are stored in BF16 and accumulated with a fixed FP32 reduction. The kernel is shared by exhaustive selection and candidate scoring. Selection orders pairs as $(-\widehat{s}(q,z_i), i)$. Before the value reduction, selected entries are placed in ascending logical order.

Floating-point reduction order affects boundary comparisons (NVIDIA, 2026). For finite, normal-range arithmetic, use the conservative bound

$$|\widehat{s}(q,z) - q^\top z| \le \gamma_{2d} \sum_j |q_j z_j|, \qquad \gamma_n = \frac{nu}{1-nu}, \tag{36}$$

where u is FP32 unit roundoff. With $M_b = \max_{i \in P_b} \|z_i\|_\infty$, the page-wide error bound is

$$\epsilon_b(q) = \gamma_{2d} \|q\|_1 M_b. \tag{37}$$

The implementation records the residual maximum as an upper endpoint,

$$v_b^+ = \max_{i \in D_b} \left(\widehat{s}(p_b, z_i) + \epsilon_b(p_b)\right) - \mathrm{Lower}(p_b^\top \mu_b), \tag{38}$$

with outward-rounded subtraction. Stored radii, norm differences, and the global maximum are bounded in the same direction. The final page bound includes $\epsilon_b(q_t)$ before comparison with the actual candidate threshold. Subnormal-sensitive operations use directed interval arithmetic in the bound path. A non-finite bound makes its page unresolved. These rules preserve the strict inequality required by Theorem 1.

B.2 Head-local state and grouped-query sharing

The page center, extrema, radius, and norm bound are stored once per KV head under grouped-query attention (Ainslie et al., 2023). Query heads maintain their own working sets, residual maxima, and anchor identifiers. A ring of $H+1$ query vectors stores all anchors that can be active before the next scheduled renewal. The current query's distance to each occupied slot is evaluated once and reused across pages.

Each head also has a working-set membership bitmap. Expansion kernels mask these positions when scoring D_b, avoiding duplicate score calculations. A locally retrieved key can enter the final selection for the current query. Membership in E_a remains fixed until renewal, when page maxima are rebuilt for the new excluded subsets.

For H_q query heads, H_{kv} KV heads, C completed pages, and head dimension d, the additional state per layer is

$$O(H_{kv} C d + H_q(C + Hd + m + N/w)), \tag{39}$$

where w is the bitmap word size. Page records contain indices into a head-specific query history. At 64K, the query-history representation uses 239.7 MiB of additional state instead of 348.9 MiB. Separate page anchors increase route time by 8% at the same scored-key fraction.

If $\mathcal{I}_t^h$ is the set of keys scored by head h, arithmetic work counts $\sum_h |\mathcal{I}_t^h|$. Physical access within KV group g is governed by

$$\mathcal{I}_t^g = \bigcup_{h:g(h)=g} \mathcal{I}_t^h. \tag{40}$$

The scheduling kernel groups compatible page requests and candidate gathers. Main results report both the arithmetic fraction and measured key-cache traffic.

B.3 LOGICAL STATE TRANSITIONS

At initialization, the current query scores the visible memory and selects E_a from its completed prefix. The same scores initialize β_a and the excluded-page residual maxima, while the partial page at the prefix boundary forms the initial frontier. Subsequent insertions extend this frontier without changing the indexed boundary, working set, or page anchors. Completing a physical page makes its shared geometry available, but its keys remain in the frontier until the head's next renewal.

Each continuation step updates the head's query history and recomputes candidate scores. If the global test succeeds, the route completes with all old-page summaries and working-set membership intact. During local expansion, the current query refreshes the residual maxima and anchor identifiers of the visited pages (Algorithm 3); unvisited page summaries and membership in E_a remain unchanged.

Renewal, whether scheduled or triggered by extensive expansion, advances the indexed boundary to the current completed prefix, selects a new working set, and resets the frontier to the remaining partial page (Algorithm 2). The new membership mask is applied before rebuilding the global summary and all excluded-page residual maxima. This ordering ensures that any key removed from the working set enters its page's next excluded subset. Each head performs these updates independently while retaining the stored keys and shared geometry of completed pages.

Algorithm 1 Recurrent Attention Routing for one head

Require: Query q_t, visible keys Z_t, state h and budgets k, m, B, H
Append all newly visible keys to the frontier F_t
if $B\lfloor N_t/B \rfloor \le m$ **then**
    return exhaustive top-k and empty state
end if
if state is empty or epoch age reaches H **then**
    Score all keys, renew h, and **return** top-k
end if
Score $C_t = E_a \cup F_t$ and obtain its kth score θ_t
if $G_t < \theta_t$ **then**
    return top-k over C_t and state h
end if
Form $\mathcal{U}_t$ using Equation (10)
if $\sum_{b \in \mathcal{U}_t} |D_b| \ge (n_a - m)/2$ **then**
    Score remaining keys, reuse current scores, renew h, and **return** top-k
end if
Score D_b for $b \in \mathcal{U}_t$ and refresh their anchors and maxima
return top-k over C_t^+ and updated state h

Algorithm 2 Working-set renewal from current scores

Require: Query q, visible key count N, current scores $s_1, \ldots, s_N$, budgets m, B
$n \leftarrow B\lfloor N/B \rfloor$
$E \leftarrow \mathrm{TopK}_{i \le n}(s_i, m)$ using the common logical-index tie rule
$F \leftarrow \{n+1, \ldots, N\}$
Rebuild the membership bitmap for E
$\beta \leftarrow \max_{i \le n,\ i \notin E} s_i$
$R \leftarrow \max_{i \le n} \|z_i\|_2$
Store q in the query-history ring and record its slot a
for each complete page P_b in the indexed prefix **do**
    $D_b \leftarrow P_b \setminus E$
    if $D_b \neq \varnothing$ **then**
        $v_b \leftarrow \max_{i \in D_b} s_i - q^\top \mu_b$
        Store anchor identifier $a_b \leftarrow a$
    else
        Mark the excluded page inactive
    end if
end for
Reset epoch age and retain the current top-k and its scores
return the updated routing state

Algorithm 2 is written in real arithmetic. The upper-endpoint evaluation in Section B supplies its floating-point implementation. Current full scores serve both selection and page-summary construction. No additional query–key product is needed to form a residual maximum from these scores.

Algorithm 3 Page refresh after local expansion

Require: Query q_t, unresolved pages $\mathcal{U}_t$, membership bitmap for E_a
Store the current query once and record its history slot j_t
for each $b \in \mathcal{U}_t$ **do**
    Load keys in P_b and mask indices belonging to E_a
    Compute the common score operator on D_b
    Add these scores and logical indices to the current candidate reduction
    $v_b \leftarrow \max_{i \in D_b} q_t^\top z_i - q_t^\top \mu_b$
    $a_b \leftarrow j_t$
end for
Merge candidate reductions and select the current top-k
Return selected scores and values in increasing logical-index order

B.4 Query-history addressing

Each history slot stores a query vector and its logical decoding step. Page metadata records the slot containing its last anchor. The scheduled renewal happens before a live anchor could be overwritten. With $H + 1$ slots, queries from the current epoch remain addressable throughout continuation.

For occupied slots j, evaluate

$$d_{tj} = \|q_t - q_j\|_2 . \tag{41}$$

All pages referring to slot j reuse this distance. The global test uses the renewal slot. Page refresh changes a scalar slot identifier and residual maximum, without copying the full current query into every page entry. This representation preserves the page-specific bound in Equation (8).

Heads sharing a KV head use the same stored key vectors and geometric page summaries. Their working sets, thresholds, and anchor identifiers remain independent. Completing a physical page makes its static geometry available to every head. A head incorporates it into its indexed prefix only at its own next renewal.

B.5 Device-side execution dependencies

Candidate scores establish θ_t before page tests begin. A successful global comparison directly supplies the selected-score reduction to the attention reader. Otherwise, bound evaluation produces an unresolved-page mask and a device-side work count. The expansion kernel consumes that mask, skips working-set positions, and merges the new scores with the retained candidate reduction.

When the unresolved-key count reaches the renewal threshold, the remaining cache is scored once. Selection and summary construction use the resulting full-score stream. Candidate scores already computed at this step are retained. All phases use logical token identifiers when resolving ties and ordering reader inputs.

Within a KV group, requests for the same key tile can be serviced together. The loaded tile participates in each requesting head's score calculation. Different query vectors still produce different scores, and a shared memory transaction does not imply shared attention support. Per-head score counts and grouped key-cache traffic characterize these two parts of execution.

B.6 Storage and request accounting

Let $C = n_a/B$. Shared static storage is $O(H_{kv}Cd)$ per layer. Head-specific storage consists of $O(H_qC)$ residual maxima and anchor identifiers, $O(H_qHd)$ query histories, $O(H_qm)$ candidate indices, and $O(H_qN/w)$ membership words. Frontier indices are contiguous and are represented by their endpoints.

Persistent-state measurements include allocated metadata buffers, candidate indices, and query histories. Key-cache traffic measures accesses to the existing key tensors. Metadata traffic measures the separate summary and index buffers. The traffic columns are per output token, aggregated across the evaluated network. Working-set membership never duplicates a value tensor.

For a paired request, write its elapsed time as

$$T_{\mathrm{req}} = T_{\mathrm{prefill}} + T_{\mathrm{init}} + \sum_{t=1}^{T} \left(T_{\mathrm{route,t}} + T_{\mathrm{reader,t}} + T_{\mathrm{rest,t}} \right). \tag{42}$$

Renewal is included in the routing time of the step that performs it. Initial construction belongs to T_{init}. This decomposition retains both the first-token cost and the amortized cost of maintaining the working set over the continuation.

C Experimental details

C.1 Models and positional configuration

Table 6 lists model geometry. Llama uses its released Llama-3.1 positional configuration. Qwen3 uses its native configuration through 32,768 tokens and static YaRN with factor four and original length 32,768 beyond that budget. YaRN scaling (Peng et al., 2024) is selected before prefill and remains fixed throughout the request. Qwen3's chat template has `enable_thinking=False`. Query/key normalization, positional transforms, and pretrained projections are unchanged.

Table 6: Model and routing settings. Both models are frozen.

Property	Llama-3.1-8B-Instruct	Qwen3-8B
Layers	32	36
Query heads per layer	32	32
KV heads per layer	8	8
Head dimension	128	128
Weight and KV precision	BF16	BF16
Selected tokens k	2048	2048
Working-set size m	4096	4096
Page capacity B	128	128
Maximum epoch age H	64	64

Dense prefill constructs each layer's KV memory. During decoding, every query head computes the token-sparse operator in Equation (1). The selected scores are retained from routing and supplied to a fused softmax/value kernel. Full-attention and Quest task evaluations use their respective attention operators.

C.2 Data, sampling, and decoding

LongBench contains Qasper, HotpotQA, 2WikiMultihopQA, GovReport, LCC, and RepoBench-P in our evaluation. Each task contributes 32 examples chosen by ascending SHA-256 hash of the example identifier. The examples are fixed across methods and models. QA evaluation uses Qasper (Dasigi et al., 2021), HotpotQA (Yang et al., 2018), and 2WikiMultihopQA (Ho et al., 2020), averaged by F1. Code similarity averages LCC (Guo et al., 2023) and RepoBench-P (Liu et al., 2024) under the official edit-similarity metric. GovReport uses ROUGE-L.

Each LongBench request has a 32,768-token total budget. Answering and code completion allow 128 generated tokens, and GovReport allows 512. The instruction and question are retained in full. An overlength context is shortened by taking equal prefix and suffix portions. LCC and RepoBench-P use their completion prompts, and the remaining tasks use the model's chat wrapper around the benchmark prompt.

The RULER configurations are `niah_single_1`, `niah_multikey_1`, `niah_multivalue`, `vt`, `cwe`, and `fwe`. Each contributes 16 examples at each of the 32K and 128K total budgets, using seed 17. A 128-token response allowance is reserved inside each budget. We retain official scoring and macro-average the six task scores. Each model and attention condition receives 384 task requests, comprising 192 LongBench and 192 RULER examples.

Greedy decoding is used for task evaluation. The paired stochastic comparison uses 16 QA requests per model, temperature 0.6, top-$p = 0.95$, and seeds 17 and 29. Selection comparisons cover 65,069,056 head routes and give 100.0% support, greedy-completion, and stochastic-completion agreement.

C.3 Calibration and timing

Calibration uses 16 PG-19 training books (Rae et al., 2020) and 16 GovReport training documents (Huang et al., 2021), ordered by document hash. Each contributes an 8,192-token prefix and a 256-token natural continuation. The calibration set determines the top-k backend and launch configuration. No language-model or router parameters are trained.

The hardware is one H100-SXM 80GB with BF16 model weights and caches. The execution stack uses PyTorch 2.6, CUDA 12.6, Triton 3.2, and FlashAttention-2 in a common decoding loop. Grouped-head scheduling and candidate/expansion buffers remain on device. Support-preserving baselines share score arithmetic, the selected-attention kernel, and tie-breaking.

Controlled timing uses four PG-19 test books at total budgets 8,192, 32,768, 65,536, and 131,072. Natural continuations have length 128 or 512, and the corresponding prefix fills the remaining budget. Batch-one runs cover all budgets. Batch-two runs use 32K and 64K with 512-token continuations. Compilation and a separate short warm-up precede two paired timed repetitions. Method order is randomized within a repetition block. Route timing includes candidate preparation, metadata, expansion, selection, and renewal. Complete-request timing also includes prefill and initial state construction.

All speedups use ratios of paired elapsed times. Source documents are the aggregation unit for trajectory statistics. Runtime profiles record original-key traffic and metadata traffic separately. We report point estimates.

C.4 Temporal analysis

The recorded analysis set comprises eight held-out PG-19 books and eight GovReport test documents, removing any source overlap with the task subset. Eight documents use 16K prefixes and eight use 64K prefixes, followed by 512-token natural continuations. Eight evenly spaced layers and one query head per KV group provide 64 analyzed routes per model.

The causal replay retains the original key insertions and queries. Fixed-memory replay scores each recorded query against its unchanged prefill cache. It computes the reference route for that fixed cache before permuting the query order, retaining the first query as the common anchor. Every individual retrieval answer remains fixed under this intervention.

Working-set lifetime is measured from regularly spaced analysis anchors, independently of algorithmic renewals. The event is failure of Equation (2) during the next 64 steps. End-of-trace observations are right-censored. Local-only steps are completed without scoring any old excluded key. Candidate coverage is $|S_t \cap (E_a \cup F_t)|/k$.

For the cost-prediction analysis, a ridge model predicts routing time over the next eight steps. Inputs comprise past support overlap, the current query's displacement from its anchor, the stored renewal margin, epoch age, and previous expansion fraction. Responses contain routing times over the following eight steps. Features are standardized on calibration data, and regularization is chosen from 10^{-3} through 10^{3} by document-grouped cross-validation. Document-held-out R^2 is 0.19 for overlap alone and 0.64 for the state feature set.

C.5 Baseline specifications

Exhaustive Top-k. The baseline scores every causal key. A disjoint calibration set selects the faster radix or framework top-k kernel for each shape. Boundary ties are resolved by logical index. Selected scores feed the same reader as RAR.

Static-Page. Pages are processed in descending coordinate-bound order until at least m keys have been scored. Their kth score supplies a threshold, and all unvisited pages with bounds at or above it are scored. Visiting every qualifying page recovers the exhaustive support.

Buffered-MIPS. The baseline retains a top-m candidate set and the global maximum and norm bound in Equation (7). A failed test performs a full search and renewal. It uses the same append frontier and maximum age as RAR.

EMA-Proposal. A lazy exponential average of selection indicators has decay 0.9. Its largest m weights propose candidates, with a pool of at most $8k$ logical indices. Newly selected indices are inserted and the smallest weights evicted, breaking ties by index. New keys are explicitly visible. Coordinate bounds complete the route to the exhaustive top-k.

Quest. The page-sparse task comparison uses a token budget of 2048 and page capacity 16. Attention is normalized over selected pages. Routing comparisons retain the token-level support in Equation (1).

C.6 Computational resources

Experiments use one H100-SXM 80GB. Model weights and pretrained routing projections remain frozen. Parameter sweeps reuse recorded queries and causal key sequences, evaluating the routing computation for each configuration.

C.7 Benchmark units and aggregation

Table 7 summarizes the evaluation units underlying the main quality table. LongBench task averages assign equal weight to datasets. RULER averages assign equal weight to the six selected configurations at each context length. All method comparisons use the same retained examples.

Table 7: Evaluation subsets and reported metrics. LongBench uses a 32K total sequence budget. Each RULER configuration contributes 16 examples at both 32K and 128K. Output allowances are included in the total sequence budget.

Dataset or group	Examples	Output allowance	Reported metric
Qasper	32	128	Token F1
HotpotQA	32	128	Token F1
2WikiMultihopQA	32	128	Token F1
GovReport	32	512	ROUGE-L
LCC	32	128	Edit similarity
RepoBench-P	32	128	Edit similarity
RULER, six configurations per length	96	128	Mean task score

Let M_D be the benchmark score averaged over retained examples of dataset D. The reported LongBench aggregates are

$$M_{\mathrm{QA}} = \frac{M_{\mathrm{Qasper}} + M_{\mathrm{HotpotQA}} + M_{\mathrm{2WikiMultihopQA}}}{3}, \tag{43}$$

$$M_{\mathrm{Code}} = \frac{M_{\mathrm{LCC}} + M_{\mathrm{RepoBench\text{-}P}}}{2}. \tag{44}$$

GovReport is reported separately. Full attention, Quest, exhaustive token selection, and RAR retain their respective attention computations across the same benchmark examples.

With six LongBench datasets and 32 examples per dataset, LongBench contributes 192 requests. Six RULER configurations, two lengths, and 16 examples contribute another 192. Each checkpoint and attention condition has 384 quality requests. The paired stochastic comparison uses the stated 16 QA requests, two seeds, and two checkpoints. These counts describe the fixed evaluation design independently of the number of attention heads visited during generation.

C.8 Survival, overlap, and candidate coverage

For fixed memory and analysis anchor a, let $E_a^{(m)}$ be the anchor top-m. Define the number of consecutive supported queries

$$L_a^{(m)} = \max\left\{h \geq 0 : S_{a+j} \subseteq E_a^{(m)} \text{ for all } 1 \leq j \leq h\right\}. \tag{45}$$

At a 64-step horizon, restricted mean lifetime is

$$\mathbb{E}\left[\min\{L_a^{(m)}, 64\}\right] = \sum_{h=1}^{64} \Pr[L_a^{(m)} \geq h] = \sum_{h=1}^{64} P_m(h). \tag{46}$$

Only the observed portion contributes for an anchor near the end of a sequence. Right-censored observations contribute to the survival curve up to their last observed query. A failure at the first subsequent query gives zero supported continuation steps under this convention.

Consecutive overlap is $o_t = |S_t \cap S_{t-1}|/k$. Candidate coverage during causal execution is

$$c_t = \frac{|S_t \cap (E_a \cup F_t)|}{k}. \tag{47}$$

Local-only execution is the event that no old excluded key is scored at the current step. It implies $c_t = 1$ under Theorem 1. Candidate coverage of one can also occur on steps that inspect additional pages. These quantities are recorded separately because they describe selection membership and the work used to recover it.

Renewals per 100 steps count both age-triggered and expansion-triggered renewals. Page-local expansion includes every non-renewal route that reaches the page-bound stage. A page-bound stage can establish that all remaining pages are below the threshold without reading their keys. The route-path percentages in Appendix D classify control flow and sum to 100%. Local-only percentages classify old-key scoring.

C.9 Ordering and feature controls

Each fixed-memory query is scored against the same prefill key set before either ordering is applied. Its reference support remains attached to the query when the sequence is permuted. The first query is retained so both runs begin with the same anchor. Subsequent candidate sets and page anchors evolve under each ordering according to Algorithm 1.

The anchor-reset comparison retains the candidate budget and physical page layout. Its temporal page information is reset according to the comparison condition. Static-page selection evaluates current-query coordinate bounds without page-specific historical maxima. Neither intervention modifies a stored key vector, the current query vector, or the selected-token budget.

The cost predictor uses information available when the current query reaches the router. Historical overlap, epoch age, and the preceding expansion fraction have already been observed. The current displacement is computed from the current query and its stored anchor. The margin is recorded at renewal. The response is the routing time over the following eight steps. Source-document grouping keeps each continuation sequence within one evaluation unit.

C.10 Controlled changes in routing parameters

Changing m rebuilds working-set membership and excluded-page maxima. The final attention budget remains k. Changing page capacity B rebuilds the chronological partition and all associated geometric summaries. Query vectors, key vectors, candidate score arithmetic, and logical-index ordering are retained for these comparisons.

Removing the global test preserves the same candidates and evaluates page bounds directly. Removing the temporal term retains coordinate bounds. Removing local refresh keeps the renewal anchor for each page until the next renewal. Query-history ablation represents each page anchor by a separate vector rather than an index into shared history. Renewal on insertion incorporates every new key into a new globally scored state. These interventions specify which component changes without altering the support selected by the common score operator.

D Additional measurements

D.1 Serving and route resolution

Table 8 collects serving measurements. Full-request ratios are rounded to one decimal place. The 20.3% and 18.1% reductions correspond to ratios of approximately 1.25 and 1.22, respectively, at the reported 128K operating point.

Table 8: Serving measurements. Throughput uses batch size two at 64K. Additional state is per request.

Measurement	Llama-3.1-8B	Qwen3-8B
Throughput, generated tokens/s	96.8	87.4
Additional state, MiB	239.7	269.6
128K request-latency reduction, %	20.3	18.1
Request speedup, 128 generated tokens	1.1×	1.1×
Request speedup, 512 generated tokens	1.3×	1.2×
State-construction share of request time, %	2.1	2.5
GQA coalescing traffic reduction, %	27.9	25.6

At 128K, the global test, local expansion, and renewal paths account for 63.8%, 31.4%, and 4.8% of Llama routes, and 59.4%, 34.8%, and 5.8% of Qwen3 routes. The route-resolution percentages sum to 100% in each model.

D.2 Layer and page measurements

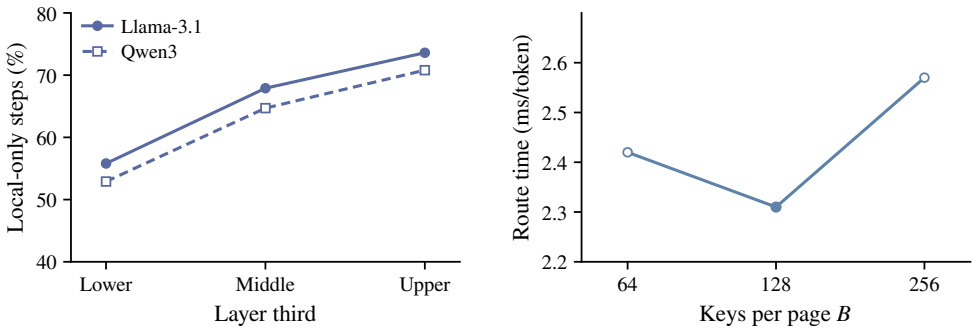

Figure 5: More routes stay local in deeper layers (left). Page-size sensitivity at 64K (right); the filled marker denotes the default $B = 128$. Lower, middle, and upper denote layer thirds.

Local-only rates in the lower, middle, and upper thirds of Llama are 55.8%, 67.9%, and 73.6%. Qwen3 gives 52.9%, 64.7%, and 70.8%. The same head-local algorithm is applied throughout the network, with renewal decisions determined by the current query and stored bounds.

The page-size sweep takes 2.42 ms at $B = 64$, 2.31 ms at $B = 128$, and 2.57 ms at $B = 256$. The width sweep and renewal rates are listed in Table 9.

Table 9: Working-set sweep at 64K.

m/k	Route time, ms	Renewal rate, %
1	3.27	13.6
2	**2.31**	5.1
4	2.47	2.1
8	3.08	0.8

D.3 Ordering intervention

Table 10 gives both work and timing changes under query permutation. Anchor reset increases RAR route time by 31.4% at the same candidate budget. In the fixed-memory replay, working-set lifetime changes from 22.9 to 4.3 steps in Llama and from 19.7 to 3.9 in Qwen3.

Table 10: Natural order to random permutation on fixed-memory queries.

Method	Scored-key change, percentage points	Route-time change, %
RAR	+43.7	+96.8
Static-Page	+0.3	+1.6

D.4 Latency ratios and traffic measures

For a paired workload with baseline elapsed time T_b and method time T_r, the speedup and latency reduction are

$$\mathcal{S} = \frac{T_b}{T_r}, \qquad \mathcal{D} = 1 - \frac{T_r}{T_b}. \tag{48}$$

The two quantities use different denominators. A 20.3% request-latency reduction corresponds to $1/(1 - 0.203) \approx 1.25$ times the baseline throughput for that fixed request, and an 18.1% reduction corresponds to approximately 1.22. The one-decimal request ratios in Table 8 follow the same conversion.

Table 11: Ratios computed from the 128K measurements in Table 2. Selection budgets and model weights are fixed. Relative reductions compare RAR with the stated baseline at batch size one and 512 generated tokens.

Quantity	Llama-3.1-8B	Qwen3-8B
Routing speedup over exhaustive	3.90×	3.60×
Decoding speedup over exhaustive	1.44×	1.44×
Routing latency reduction over EMA-Proposal	35.2%	33.4%
Original-key score reduction over exhaustive	85.4%	82.9%
Key-cache traffic reduction over exhaustive	74.4%	69.9%

Per-head scoring and physical memory access have different aggregation rules under GQA. Scoring counts each query–key pair evaluated by a head. Key-cache traffic records the movement of key tiles requested across the KV group. Candidate indices, query histories, and page summaries belong to metadata traffic. The ablation without the global test leaves scored keys unchanged at 17.6% but increases metadata traffic from 68.9 to 193.4 MiB per token.

D.5 Working-set width and page capacity

The width sweep in Table 9 holds k fixed. Its configurations alter the number of candidates reranked unconditionally and the rate of renewal. At $m/k = 1$, the working set initially equals the selected support. At larger widths, additional high-ranking keys can enter a later support without leaving the candidate range. Measured route time includes both this repeated candidate scoring and subsequent expansions.

The minimum reported route time, 2.31 ms, occurs at $m/k = 2$. Doubling this width reduces renewal rate from 5.1% to 2.1% and increases route time to 2.47 ms. Doubling again reduces renewal rate to 0.8% and increases route time to 3.08 ms. The timing objective accounts for every step, including the more frequent steps that do not renew.

Page capacity changes the granularity of bound evaluation and key loading. Smaller pages require more metadata entries for a fixed prefix. Larger pages associate each unresolved bound with more keys. The reported times of 2.42, 2.31, and 2.57 ms for capacities 64, 128, and 256 include both costs. Geometric and temporal summaries are rebuilt for each capacity, preserving the page-domain definitions in Section 3.