HYPIC

Accelerating Hybrid-Attention LLM Serving with Position-Independent Caching

Paper authors: Yifei Liu, Juntong Wu, Yang Liu, Junhao Hu,
Minghao Li, Xiaoxu Chen, and Weihang Chen

arXiv:2607.01299v2 | revised July 12, 2026
Research discussion | October 16, 2026

Cache how a segment transforms memory, then repair its boundary.

Source: arxiv.org/abs/2607.01299v2. Reading presentation; not an author presentation.

Repeated documents do not imply repeated prefixes

A typical RAG reuse opportunity

Request 1: System → Document A → Document B → Query 1

Request 2: System → Document C → Document B → Query 2

- Prefix caching stops at the first mismatch.
- Position-independent caching (PIC) reuses B behind a new prefix.
- B's position and contextual hidden states both change.

Hybrid models complicate reuse: most layers retain a **recurrent state**, while a minority retain per-token KV.

Paper, Secs. 1–2, PDF pp. 1–3. Request example is a teaching illustration.

Three obstacles require three different mechanisms

Obstacle	HYPIC mechanism
End-state addition ignores decay and erasure	Cache each segment's transition T_C and zero-start state $S_{C\|0}$
Linear layers block arbitrary-token correction	Recompute a short seam at segment beginnings through the entire stack
Cold requests still need prefill	Scatter independent segments across instances, then combine

The reusable unit is a segment with an explicit composition rule.

Paper, Sec. 3, PDF pp. 3–5; Sec. 4, PDF pp. 6–9.

Linear attention separates memory transport from writing

$$\boxed{S_i = T_i S_{i-1} + u_i}, \qquad o_i = q_i^\top S_i$$

- $S_i \in \mathbb{R}^{d_k \times d_v}$: memory.
- $T_i \in \mathbb{R}^{d_k \times d_k}$: transport, decay, or erasure.
- $u_i \in \mathbb{R}^{d_k \times d_v}$: new write.
- $q_i \in \mathbb{R}^{d_k}$: read query.

Representative transitions

Scalar: $T_i = \gamma I$

Diagonal: $T_i = \mathrm{diag}(g_i)$

GDN: $T_i = g_i(I - \beta_i k_i k_i^\top)$

T_i, u_i depend on the layer's input, not on its recurrent state.

Paper, Eq. (2) and Table 1, PDF p. 3. g_i is scalar for GDN and a vector for diagonal gating.

Adding segment states retains memory that should decay

$$S_{C_1 C_2|0} = \underbrace{T_{C_2} S_{C_1|0}}_{\text{transport old memory}} + S_{C_2|0}$$

$$\Delta = S_{\text{true}} - S_{\text{addition}} = (T_{C_2} - I) S_{C_1|0}$$

Constant decay makes the failure explicit

$T_i = \gamma I \quad \Rightarrow \quad \|\Delta\| / \|S_{C_1|0}\| = 1 - \gamma^{|C_2|}$

Suffix length	256	512	1024
Error, $\gamma = 1 - 2^{-10}$	0.221	0.394	0.632

Paper, Eqs. (4)–(5), Table 2, PDF p. 4. Error is structural, not a floating-point artifact.

A cached segment is an affine map on incoming memory

$$F_C(S) = T_C S + B_C, \qquad B_C \equiv S_{C|0}$$

$$(T_B, B_B) \circ (T_A, B_A) = \big(T_B T_A,\ T_B B_A + B_B\big)$$

- Compute each tuple once; apply it behind a new prefix.
- Composition is associative; segment order still matters.

Teaching example: two steps of $S \leftarrow 0.5S + 1$

$T_C = 0.25$, $B_C = 1.5$. From prefix state 10:
correct composition $= 4$; naive addition $= 11.5$.

Derived from paper Eq. (6), PDF p. 6. Tuple notation and numerical example are teaching additions.

Reuse is constant in segment length, not in all dimensions

Transition	Extra T_C storage	Build	Apply to S
Scalar	2 B	$O(\|C\|)$	$O(d_k d_v)$
Diagonal	256 B	$O(\|C\| d_k)$	$O(d_k d_v)$
Dense	32 KiB	$O(\|C\| d_k^2)$	$O(d_k^2 d_v)$

- Storage: FP16, $d_k = d_v = 128$, per segment/head/layer.
- The end-state $S_{C|0}$ is additional to the table's storage.
- n segments still require $O(n)$ compositions.
- Full-attention KV movement and seam work remain.

Paper, Table 3, PDF p. 7. Paper labels 32 KiB as 32 KB.

Layer-exact algebra coexists with deep contextual drift

The essential condition

Composition is exact for the same layer input hidden states.
Independent segment prefill changes those inputs at deeper layers.

Layer 0: Qwen3.5-35B

1096 tokens / 4 segments

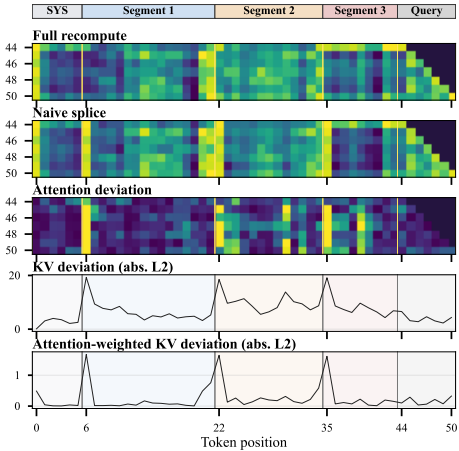
Relative error: 6×10^{-5}

Angular error: 0.003°

Deepest linear layer	Relative L_2	Angle
Qwen3.5-35B	8.92%	5.11°
Ring-flash	8.69%	4.98°

Paper, Sec. 4.2, p. 7; Sec. 6.3 and Table 4, p. 12. Deep probe: 512 tokens, two chunks, all full-attention layers disabled. These are measurements, not a general error bound.

The largest observed splice deviations occur at boundaries

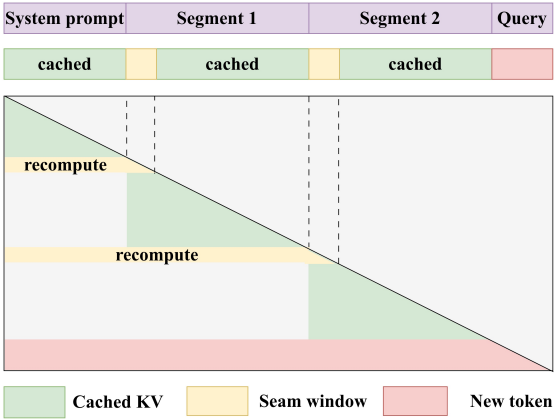


- Token position on the horizontal axis.
- Compare full recompute, naive splice, and their deviations.
- KV and attention-weighted KV errors peak at segment starts.

A boundary offers both a useful repair target and an available prefix state.

Paper, Fig. 3 and Sec. 3.2, PDF p. 5. Diagnostic: Qwen3.5-35B-A3B, layer 7, head 0.

An eight-token seam repairs cross-segment attention

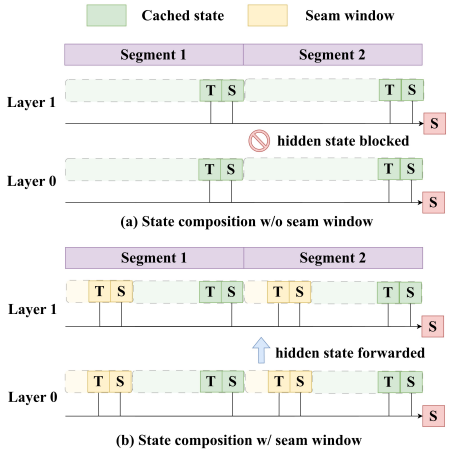


- System prompt: cache all.
- Interior segment: recompute first w tokens; reuse the body.
- Query: compute end to end.
- Default $w = 8$.

Seam queries attend to the assembled prefix under the causal mask.

Paper, Fig. 7 and Sec. 4.3, PDF pp. 7–8. Green: reused KV; yellow: seam; pink: query.

Seam tokens must also traverse every linear layer



At each linear layer:

1. Start from the assembled prefix state.
2. Recur over seam tokens and emit their per-token outputs.
3. Apply the cached body tuple to jump to the segment end.

The body tuple excludes the seam to avoid counting it twice.

Paper, Fig. 8 and Sec. 4.3, PDF p. 8. The reused body still reflects independent-prefill hidden states.

Shared segment caches become request-specific running state

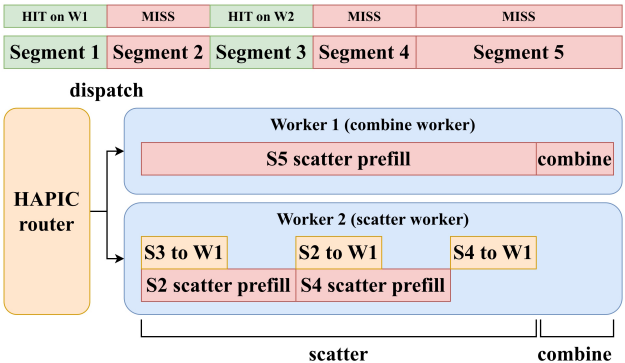
Public pool	Private pool
Segment tuples $(T_C, S_{C\|0})$	Assembled running state
Local-position K and unchanged V	Globally positioned K copies
Trailing causal-convolution state	Retained for later prefix hits

$$K_b = R(b-a)K_a$$

- Prefix lookup first, PIC lookup for remaining segments second.
- Separate capacities and LRU policies for the two pools.
- Linear-state re-rotation needs the stated scalar-transition condition.

Paper, Secs. 4.2–4.3, Eqs. (7)–(8), PDF pp. 7–8. Convolution warm-up is covered when $w \geq k - 1$.

Independent cold segments can prefill on different instances



- Scatter only cache misses.
- Fetch hit caches from peers.
- Combine tuples and repair seams.
- Then start normal decoding.

$$O(n|C|) \longrightarrow O(\lceil n/m \rceil |C| + c)$$

Paper, Fig. 9 and Sec. 4.4, PDF pp. 8–9. Simplified scheduling model: n equal-length segments, m workers, combine overhead c.

Three evaluation regimes answer different serving questions

Regime	Workload and measurement
Fully warmed	HotpotQA / TriviaQA: F1; MultiNews / GovReport: ROUGE-L. Compare p50 TTFT.
Production load	Prod-RAG: mean 12k input tokens, 10-minute warm-up, remaining misses. Sweep request rate.
Fully cold	32k-token prompt, eight segments, TP-1 instances. Scale worker count.

Four models: Ring-mini, Ring-flash, Qwen3.5-35B, Qwen3.5-122B.
Baselines: Full Recompute, Prefix Cache, Naive Addition.

Paper, Secs. 6.1–6.5, PDF pp. 10–12. Hardware: one 8×H20-3e node, 141 GB/GPU. Figs. 10–11 use TP=1/4/2/4 respectively.

Warm reuse improves TTFT with model-dependent quality

Model	Speedup / Full	Speedup / Prefix	Score delta / Full
Ring-mini	2.94×	2.77×	−3.44 points
Ring-flash	2.97×	2.85×	−3.29 points
Qwen3.5-35B	3.79×	3.32×	+0.47 points
Qwen3.5-122B	4.67×	4.05×	−0.56 points

Reported aggregate across 16 model–task cells

3.25× mean speedup over Prefix Cache; **1.71-point** mean loss.

Naive Addition loses 66.9% of Full Recompute's score, but omits both transitions and seam repair.

Paper, Fig. 10 and Sec. 6.2, PDF pp. 10–11. All segments pre-warmed; p50 TTFT. Points are absolute score differences on a 100-point scale, aggregated across F1 and ROUGE-L.

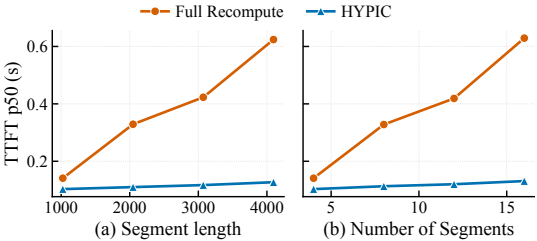
More requests fit under the same median-latency target

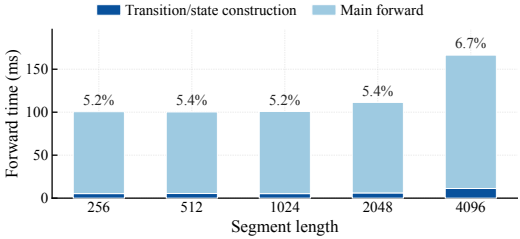
Model	QPS / Prefix at 1 s	Peak token throughput / Prefix
Ring-mini	1.85×	1.50×
Ring-flash	1.49×	1.46×
Qwen3.5-35B	1.58×	1.32×
Qwen3.5-122B	1.71×	1.30×

- Mean sustainable QPS gain: **1.66×** over Prefix Cache.
- The 1-second threshold is applied to p50 TTFT.
- Peak token throughput and SLO-constrained QPS are different metrics.

Paper, Fig. 11 and Sec. 6.2, PDF p. 11. Prod-RAG after 10-minute warm-up. No p95/p99 guarantee follows from these median curves.

Longer segments make reuse more attractive





Four segments, 1k $\rightarrow$ 4k each:

Full: 141 $\rightarrow$ 624 ms

HYPIC: 103 $\rightarrow$ 127 ms

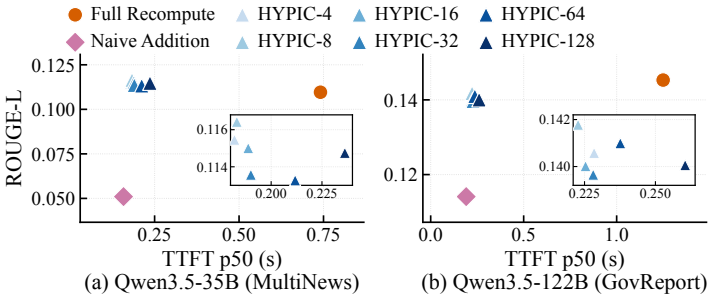
Dense-transition construction:

5.2–6.7% over main forward for 256–4096-token segments.

At fixed 1k-token segments, each extra segment adds about 2.3 ms for HYPIC versus 40.7 ms for Full Recompute.

Paper, Figs. 12–13 and Sec. 6.3, PDF p. 11. Construction probe uses Qwen3.5-35B-A3B; cost is paid on first segment prefill.

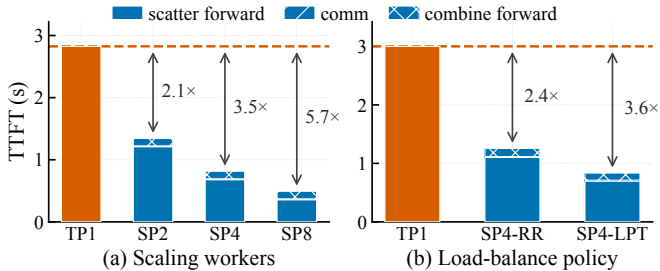
Eight seam tokens are a useful setting in the two tested cells



- At $w = 8$: ROUGE-L 0.1164 (35B/MultiNews), 0.1418 (122B/GovReport).
- Wider seams add latency without consistent quality gains.
- Text and plot disagree on the sweep; no explicit HYPIC-0 point appears.

Paper, Fig. 14 and Sec. 6.4, PDF p. 12. Text: 0/4/8/16/32. Legend: 4/8/16/32/64/128. Details in backup.

Eight workers cut the measured cold-request TTFT to 0.49 s



Configuration	Full, 1 worker	HYPIC, 2	HYPIC, 4	HYPIC, 8
TTFT (s)	2.83	1.34	0.82	0.49

Communication: 12–15 ms. Combine: about 120 ms.
Most latency reduction comes from parallel scatter prefill.

Paper, Fig. 15(a), Sec. 6.5, PDF p. 12. 32k tokens, eight equal segments, TP-1 instances. Reported 5.7× uses more workers than the baseline.

LPT reduces stragglers; immediate transfers limit stalls

Scheduling rule

Sort cold segments by decreasing token count. Assign each one to the worker with the smallest accumulated token load.

Eight uneven segments, four workers	Round-robin	LPT
Scatter forward	1.10 s	0.69 s
Total TTFT	1.26 s	0.84 s

Each worker sends a completed segment immediately, overlapping transfer with its next segment's computation.

Paper, Eq. (9), Sec. 4.4, PDF p. 9; Fig. 15(b), Sec. 6.5, PDF p. 12. LPT is about 1.50× faster than round-robin using displayed TTFT values.

Contextual accuracy and resource costs shape the tradeoff

- **Accuracy:** deep cached inputs differ from full-context inputs; Ring loses more task score than Qwen in these experiments.
- **Memory:** store extra transitions and request-specific rotated K; cache capacity affects realized hit rate.
- **Resources:** cold-request speedup uses additional instances on the tested single-node system.
- **Evidence:** median load curves and a short isolated drift probe leave tail latency and long-context drift open.

Most informative next experiment

Keep transitions, remove only seam repair; then measure quality and p95/p99 latency under a fixed cluster and cache budget.

Analysis based on Secs. 4.2–4.4 and 6.2–6.5, PDF pp. 6–12. Next experiment is a discussion proposal.

Segment reuse connects algebra to serving execution

1. **Cache a transformation:** $(T_C, S_{C|0})$ preserves how a segment changes incoming memory.
2. **Repair a reachable region:** seam tokens carry corrected hidden states through the hybrid stack.
3. **Schedule the same unit:** independent segments allow parallel cold prefill followed by composition.

The central distinction: exact composition for fixed layer inputs; approximate reuse across contextualized deep inputs.

Yifei Liu et al. HYPIC: Accelerating Hybrid-Attention LLM Serving with Position-Independent Caching. arXiv:2607.01299v2, July 2026. Versioned paper link.

Backup: unrolling the recurrence explains the cached tuple

For a segment of m tokens, with fixed layer inputs:

$$S_m = (T_m \cdots T_1) S_0 + \sum_{t=1}^{m} (T_m \cdots T_{t+1}) u_t.$$

Cache T_C and $S_{C|0}$, which do not depend on S_0.

For suffix segments $C_1, \ldots, C_n$ behind prefix state S_S:

$$S_{S\,C_1 \cdots C_n} = \left(\prod_{i=n}^{1} T_{C_i} \right) S_S + \sum_{i=1}^{n} \left(\prod_{j=n}^{i+1} T_{C_j} \right) S_{C_i|0}.$$

Later transitions multiply on the left. Empty products equal I.
Sequential application needs n tuple operations.

Paper, Eq. (6), PDF p. 6. First equation is an explanatory expansion of Eq. (2).

Backup: two recurrence calls construct the reusable state

Content accumulation

Initialize $S_0 = 0$.
Keep normal writes u_i.

Result: $S_{C|0}$.

Transition accumulation

Initialize the accumulator to I.
Set all writes to zero.

Result: $T_m \cdots T_1 = T_C$.

- SGLang implementation uses FLA kernels, Python, and Triton.
- Application markers define segments; token-ID hashes find hits.
- A width-k causal convolution needs $k - 1$ boundary inputs.
- Cache trailing convolution state; $w \geq k - 1$ lets the seam warm it up.

Paper, Secs. 4.2–4.3, PDF pp. 7–8; Sec. 5, PDF p. 9. Transition accumulator has shape $d_k \times d_k$.

Backup: KV and state shifts have different conditions

Full-attention keys

$R(a+b) = R(a)R(b) \quad \Rightarrow \quad K_b = R(b-a)K_a.$

Rotate cached K; retain V; regenerate Q and seam tokens.

The paper's scalar-transition linear-state case

$T_i = \gamma I$ commutes with rotations, giving

$S_{C|b} = R(b-a)S_{C|a}.$

The second identity relies on the stated transition structure.

A general dense transition need not commute with RoPE.

Paper, Eqs. (7)–(8), PDF pp. 7–8. In the re-rotation equation, a, b denote starting positions.

Backup: the seam sweep does not isolate a zero-seam baseline

Evidence in v2	What it contains
Sec. 6.4 text	Widths $\{0, 4, 8, 16, 32\}$
Fig. 14 legend	HYPIC-4/8/16/32/64/128
Naive Addition	No cached transitions and no seam repair

- The plot supports a useful $w = 8$ operating point on two cells.
- It does not expose a clear "correct transitions, $w = 0$" control.
- The aggregate Naive Addition gap cannot quantify the seam's isolated contribution.

Paper, Secs. 6.1 and 6.4, Fig. 14, PDF pp. 10 and 12. This discrepancy is present in both the submitted TeX and rendered PDF.