READ DEPTH AND BROADCAST DEPTH: WHAT SHARING KEYS AND VALUES ACROSS LAYERS COSTS A TRANSFORMER

Anonymous authors
Paper under double-blind review

ABSTRACT

Language models increasingly share keys and values across layers, so many layers query caches that few layers compute. We separate the two roles of attention: the *read depth* ρ counts layers that query, or select from, the whole context afresh, and the *broadcast depth* β counts the key–value generations that global attention reads, each computed below its readers. Sharing preserves the first and not the second. The composition construction of Chen, Peng and Wu (FOCS 2025) works with one generation, and reusing or re-ranking selections within a generation adds no read depth: for a fixed architecture at logarithmic precision, read depth ρ cannot compose ρ functions on long enough prompts whose blocks lie beyond the sliding windows' reach. With one generation of per-token keys and selections and the windows below it, the hierarchy holds at linear length: following $\rho+1$ pointers between two m-entry tables farther apart than the windows' reach needs read bandwidth (the bits one side of the prompt sends over all reads) $\Omega(m) - O(\rho \log m)$, less one hidden state, at read depth ρ, but $O(\rho \log n)$ at read depth $\rho+1$. Without sliding windows, layers reading one such generation need read bandwidth $N/2 - \log_2(1/(2\varepsilon))$ to compute the inner product of two N-bit blocks with advantage ε over uniform inputs, whereas two generations compute it with two layers of logarithmic size. In these terms YOCO with a sliding-window self-decoder has broadcast depth 1 and read depth $L/2$, its sparse variant YOIO has read depth 1, and DeepSeek-V4.1-Flash, as specified, has broadcast and read depth 4 among 38 global layers; at published widths and $p = 16$ the bounds for YOCO's joins and YOIO's two-pointer chasing apply from prompts of order 10^7 tokens, but DeepSeek-V4.1-Flash's composition bound needs over 10^{100000} tokens. Bounds concern one forward pass in p-bit arithmetic with exact attention partial sums, p possibly growing with the prompt length n.

1 INTRODUCTION

In a standard decoder-only transformer every attention layer does two things. It *reads*: each position issues a fresh query against the keys and values of the earlier positions. And it *writes*: each position publishes fresh keys and values, computed from its current hidden state, for later positions to read. The written keys and values—the KV cache—dominate the memory of long-context inference, and architectures increasingly pull the two roles apart. Beyond sharing or compressing keys and values within a layer (Shazeer, 2019; Ainslie et al., 2023; DeepSeek-AI, 2024), cross-layer attention shares one KV cache between adjacent layers (Brandon et al., 2024; Wu et al., 2025); YOCO computes a single global KV cache from the output of a "self-decoder" of efficient layers (gated retention by default, or sliding-window attention) and lets every upper layer read it (Sun et al., 2024), and YOIO adds a single top-k selection that all upper layers reuse (Sun et al., 2026); and in sparse attention (Yuan et al., 2025; DeepSeek-AI, 2025) and sparse decoding, later layers may reuse the positions that an earlier layer selected (Yang et al., 2025; Bai et al., 2026; Gao et al., 2026). DeepSeek-V4.1-Flash (DeepSeek-AI, 2026b) combines a causal encoder–decoder split of its layers with compressed sparse attention whose layers either compute fresh keys and values together with a fresh top-k selection (*Full*), re-select with a fresh indexer query over shared keys and values (*Reindex*; in the post-trained model, within a candidate pool), or reuse both (*Reuse*). In that model 38 of the 40 layers attend

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.

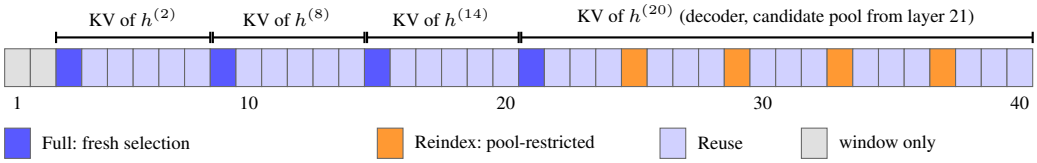


Figure 1: The post-trained DeepSeek-V4.1-Flash through the depth ledger. All 40 layers have sliding-window attention ($w = 128$) and 38 attend globally, but only four global KV generations are ever published (brackets: $\beta = 4$) and only four layers select fresh entries from the whole context (dark: $\rho = 4$); the Reindex layers re-select inside the candidate pool of layer 21, and the other 30 global layers reuse earlier selections of the same generation (Corollary G.1).

globally, but only 4 of them read a KV generation of their own and select from the whole context (Figure 1).

Which of these layers count as depth? Depth is the resource that the known unconditional lower bounds for multi-layer transformers measure. Chen et al. (2025) proved that r-step sequential function composition requires size polynomial in the context length for a standard transformer with r layers but is solved with $r + 1$ layers of polylogarithmic size. In a standard transformer every layer reads and writes, so this hierarchy cannot tell which of the two roles it counts—and it is exactly this distinction that KV sharing exploits.

We separate the two roles. The *read depth* ρ of a model counts the layers that issue a fresh dense query or a fresh top-k selection over the whole context, and its *broadcast depth* β counts the distinct KV sources of its global layers, the KV generations it reads (Definition 2.2). Lookup chains, as in function composition and pointer chasing, need only a frozen KV cache that the last position consults often enough; a join, such as the inner product $\mathrm{IP}(x, y) = \bigoplus_i x_i y_i$ of two blocks, is naturally computed by pairing y_i with x_i at the position of y_i and aggregating the products, which must then be written into a fresh KV generation.

Contributions. All results concern *typed KV-sharing transformers* (Section 2), with arbitrary token-wise maps, one forward pass over the prompt, and p-bit arithmetic with exact attention partial sums, where p may grow with the prompt length n (Definition 2.3).

1. **Read depth survives KV sharing; index reuse within a generation adds none.** The $(r + 1)$-layer construction of Chen et al. (2025) for r-step function composition reads a single KV generation, with one head of dimension 3 and $O(\log n)$-bit precision (Theorem 3.1). Conversely, for a fixed architecture at logarithmic precision, read depth ρ cannot compose ρ functions on long enough prompts whose blocks lie beyond the sliding windows' reach, however many layers reuse or re-select within a generation (Theorem 3.2); reusing a selection across generations can add read depth (Proposition F.2).
2. **Within one generation the read hierarchy holds at linear length.** If all global layers read one KV generation of per-token keys and values and the sliding windows sit below it, as in YOCO, then following $\rho + 1$ pointers between two tables of m entries that lie farther apart than the windows' reach needs read bandwidth $\Omega(m) - O(\rho \log m)$, less one hidden state, at read depth ρ; read depth $\rho + 1$ suffices at logarithmic size (Theorem 3.3 and Lemma 3.4). With a second generation, ρ reads follow $2\rho - 1$ pointers from the fixed start (Proposition F.3).
3. **Broadcast depth does not survive KV sharing** (Theorems 4.1 and 4.2). Without sliding windows, layers that read one generation of per-token keys and values need read bandwidth $N/2 - \log_2(1/(2\varepsilon))$ to compute the inner product of two N-bit blocks with advantage ε over uniform inputs, whatever their number and types; with sliding windows the same holds, for blocks longer than the windows' total reach, for a cost that also charges the reads of the positions whose windows reach the last position. Two generations compute it with two layers of logarithmic size. Whether every further level of aggregation needs its own generation is Conjecture 4.3.
4. **A depth ledger** (Section 5). YOCO with a sliding-window self-decoder and L layers has broadcast depth 1 and read depth $L/2$, YOIO has read depth 1, and DeepSeek-V4.1-Flash, as specified, has broadcast and read depth 4 among 38 global layers.

Two simulations underlie the lower bounds. A single KV generation is a table fixed before any global communication, so a cut between two blocks gives a two-party protocol in which every read costs one exact partial sum or local top-k list per side (plus, with sliding windows, the states of positions whose windows reach the reader); across many generations, Lemma B.2 charges one epoch (round of messages) of the communication model of Chen et al. (2025) per fresh read.

Related work. Chen et al. (2025) introduced the autoregressive communication model; their simulation already covers dense layers whose keys and values come from any earlier layer, and Theorem 3.1 is their construction. We add re-selection, reuse and sliding-window layers to the simulation and use their epoch lower bound as a black box. Ye et al. (2026) separate hybrids of full and linear attention, an incomparable class. Wang (2026) studies cache-size trade-offs for pointer chasing when every layer writes its own cache and names as open the translation of pointer-chasing round lower bounds (Nisan & Wigderson, 1993; Yehudayoff, 2020) into depth lower bounds, which Theorem 3.3 carries out for models whose global layers read one KV generation with the windows below it, a class different from Wang's. Our lower bounds extend one-layer communication arguments (Sanford et al., 2023; Peng et al., 2024) to many layers that share one KV generation; Section I discusses depth separations that count layers that read and write, and KV-cache size lower bounds, which measure memory rather than depth.

2 Typed KV-sharing transformers

A model reads tokens $x_1, \dots, x_n$ from a finite alphabet Σ and computes hidden states $h_i^{(\ell)}$ with at most D coordinates (the *width*) for positions $i \in [n]$ and layers $0 \le \ell \le L$, starting from $h_i^{(0)} = \mathrm{emb}(x_i, i)$. As in Chen et al. (2025), token-wise maps are arbitrary functions and may depend on n, on the task parameters and on the position of the token (as rotary embeddings do); lower bounds therefore hold for every choice of weights, and upper bounds are constructions in the same model.

Definition 2.1 (Typed KV-sharing transformer)**.** Each layer $\ell \in [L]$ consists of token-wise maps and at most two attention blocks, both reading the states $h_\cdot^{(\ell-1)}$ for their queries.

- *Local attention* with window $w \ge 1$: position i attends to the positions $(i-w, i]$, with keys and values computed from $h_\cdot^{(\ell-1)}$. ($w = 1$ means no local mixing; w denotes the largest window over the layers.)
- *Global attention* with H heads whose query, key and value vectors have at most d coordinates, and a *KV source* $\sigma(\ell) \in \{0, \dots, \ell-1\}$: the key and value of position j are fixed functions of $h_j^{(\sigma(\ell))}$, and the query of position i is a function of $h_i^{(\ell-1)}$. Global attention is *dense* (position i attends to all $j \le i$) or *sparse*: position i attends to a set $S_i^{(\ell)} \subseteq [i]$ of at most k positions, produced in one of three ways.
  - (a) *Fresh selection*: the top k positions $j \le i$ by an indexer score, a fixed function of (q_i^I, κ_j) rounded to p bits (for instance $\langle q_i^I, \kappa_j \rangle$, or a sum of gated inner products over indexer heads as in DeepSeek-AI, 2025; 2026a), where q_i^I is a function of $h_i^{(\ell-1)}$ and κ_j of $h_j^{(\sigma(\ell))}$. Such a layer may also produce a *candidate pool*, the top K_{pool} positions by a second indexer score of the same form.
  - (b) *Pool-restricted re-selection*: the top k positions, by a fresh indexer score of the same form, inside the pool produced at the same position by an earlier layer ℓ' of type (a) with $\sigma(\ell') = \sigma(\ell)$.
  - (c) *Reuse*: $S_i^{(\ell)} = S_i^{(\ell')}$ for an earlier layer ℓ' of type (a) or (b) with $\sigma(\ell') = \sigma(\ell)$.

The widths H, d, D, the sparsity k and the pool size are maxima over the layers; every global layer has at least one head, every sparse layer selects at least one position, and all global heads of a layer share its access type. A head may carry a fixed attention sink (a constant added to its denominator). A standard decoder-only transformer has dense global attention with $\sigma(\ell) = \ell - 1$ in every layer and no local attention.

Every KV source is computed below every layer that reads it, and layers that share a KV source may apply different key and value maps to it. Four optional features, defined in Section A, cover the

Table 1: Depth ledger: KV sharing lowers broadcast depth, index reuse within a KV generation lowers read depth, and sliding windows contribute local depth only. For DeepSeek-V4.1-Flash, $(+4)$ counts the pool-restricted re-selections, which are not fresh reads; in the pre-trained Base checkpoint they score the whole context and are fresh reads. Entries for DeepSeek-V4.1-Flash, YOCO and YOIO are Corollaries G.1, G.2 and G.4; cross-layer attention is discussed in Section G.4.

Architecture	β	ρ	λ	mechanism
standard transformer, L layers	L	L	0	every layer reads and writes
cross-layer attention, factor 2 (Brandon et al., 2024)	$L/2$	L	0	pairs of layers share one KV cache
YOCO, sliding-window self-decoder (Sun et al., 2024)	1	$L/2$	$L/2$	one KV cache read by the upper half
YOIO / cross-layer sparse attention (Sun et al., 2026)	1	1	$L/2$	one selection reused by the upper half
DeepSeek-V4.1-Flash (DeepSeek-AI, 2026b)	4	4 $(+4)$	40	Full / Reindex / Reuse, encoder–decoder
DeepSeek-V4.1-Flash-Base	4	8	40	re-selections score the whole context

remaining mechanisms of the architectures in Section 5: (d) compressed entries, whose key and value are functions of the states of an aligned block of c consecutive positions; (e) block-max candidate pools; (f) joint heads, which apply one softmax to the concatenation of their window items and their global items; and (g) raw-token windows, which let the token-wise maps of layers $\ell \geq 1$ read the last $a_{\text{raw}} \leq w$ tokens. The *span* of an entry or pool block is the number of positions whose states it depends on, and $c_{\max}$ denotes the largest span ($c_{\max} = 1$ without (d) and (e)). We say that a model has *per-token keys and values* (and selections) when it uses neither feature (d) nor feature (e). Unless a statement says otherwise, a typed KV-sharing transformer uses none of the features (d)–(g).

Definition 2.2 (Three depths)**.** The *broadcast depth* β of a model is the number of distinct KV sources $\sigma(\ell)$ among its global layers: the number of *KV generations*. Its *read depth* ρ is the number of global layers that are dense or of type (a); we call them *fresh-read layers*. Its *local depth* λ is the number of layers with local attention of window $w \geq 2$.

A standard transformer with L layers has $\beta = \rho = L$ and $\lambda = 0$. Broadcast depth counts KV sources, not the information in them (Section A), and local depth is bookkeeping: none of our bounds depends on λ, and sliding windows enter them only through w and L. Table 1 lists the three depths of the architectures that motivate this paper.

Definition 2.3 (Precision and the exact-sum convention)**.** Fix an integer $p \geq 2$. A *p-bit number* is a dyadic rational $a/2^f$ with integers $|a| < 2^p$ and $0 \leq f \leq p$; rounding is to the nearest p-bit number with a fixed tie rule, saturating at the extremes. There are $(p+2)2^p - 1 < 2^{2p}$ such numbers, so each has a fixed-length code of $2p$ bits, and a hidden state of width D has a code of $s \coloneqq 2Dp$ bits. Every stored quantity is a p-bit number. Inside a head with query q_i and attended set S, each score $\langle q_i, k_j \rangle$ and each weight $a_{ij} = e^{\langle q_i, k_j \rangle}$ is rounded to p bits, the numerator $\text{Num}_i = \sum_{j \in S} a_{ij} v_j$ and the denominator $\text{Den}_i = \sum_{j \in S} a_{ij}$ are computed *exactly*, and the output $\text{Num}_i / \text{Den}_i$ is rounded to p bits (the output is 0 if $\text{Den}_i = 0$). Top-k selections and candidate pools sort by rounded indexer score and break ties by the larger position.

Each coordinate of an exact partial sum over at most n items has a fixed-length code of $4p + \lceil \log_2 n \rceil + 4$ bits, and for every partition $S = S_1 \cup S_2$ of the attended items (i) the output of a head is a fixed function of (Num, Den) over S_1 and over S_2, and (ii) a top-k selection over S is a fixed function of the top-k lists (positions and scores) over S_1 and S_2; the simulation of Chen et al. (2025, Lemma 3.1) uses the same convention. It idealizes the accumulator: with rounded accumulation, or a kernel that subtracts a running maximum before exponentiating (Milakov & Gimelshein, 2018; Dao et al., 2022), the simulations do not apply literally (Section A). A model has *logarithmic size* if H, d, D, the sparsity k, the pool sizes and the spans are constants and $p = O(\log n)$; each of its reads then has $O(\log n)$ bits.

Definition 2.4 (Read bandwidth)**.** The *per-read bandwidth* of a global layer is the number of bits in which one side of a partition of the attended set can describe its part of one read:

$$B_{\text{dense}} \coloneqq H(d+1)\big(4p + \lceil \log_2 n \rceil + 4\big) \quad \text{(dense)}, \qquad B_{\text{sparse}} \coloneqq k\big(\lceil \log_2 n \rceil + 2p + c\,s\big) \quad \text{(type (a))},$$

plus $B_{\text{pool}} \coloneqq K_{\text{blk}}(\lceil \log_2 n \rceil + 2p + c'c\,s)$ for a pool-producing layer (with feature (e), K_{blk} pool blocks of c' entries; without it, $K_{\text{blk}} = K_{\text{pool}}$ and $c' = 1$), and 0 for layers of type (b) and (c); here c is the span of the layer's entries ($c = 1$ without feature (d)). A dense part is its exact partial numerators and denominator; a sparse part is its local top-k list, with positions, rounded scores and the KV-source states of the listed entries. We write B_ℓ for the per-read bandwidth of layer ℓ, $B_{\text{read}} \coloneqq \max_\ell B_\ell$ over the fresh-read layers, and $\mathrm{RB}(M) \coloneqq \sum_{\ell \text{ global}} B_\ell$ for the *read bandwidth* of the model M. Read bandwidth is a communication measure, not the memory traffic of an attention kernel: a dense read that streams the whole cache has bandwidth $O(Hd(p + \log n))$.

Tasks. *Join parity* JP_N: the prompt is $x_1 \cdots x_N\, y_1 \cdots y_N\, q$ over $\Sigma = \{0, 1, q\}$ ($n = 2N + 1$), and the output at position n is $\mathrm{IP}(x, y) = \bigoplus_{i=1}^{N} x_i y_i$. *Sequential function composition* r-FuncComp (Chen et al., 2025, Definition 2.1): functions $z_0 \in [m]$ and $z_j : [N_{j-1}] \to [N_{j-1}]$ for $j \in [r]$, a query $\omega = (\omega_1, \ldots, \omega_{r-1})$, and the output i_r of the chain $i_0 = z_0$, $i_1 = z_1(i_0)$, $i_{j+1} = z_{j+1}(\omega_j, i_j)$, where the pair (ω_j, i_j) is identified with an element of $[N_j]$ by a fixed bijection. The prompt lists the blocks $E_r, \ldots, E_1$ (one token per table entry of $z_r, \ldots, z_1$), then E_0 (the token z_0), then E_{-1} (the token ω, last); the parameters K, m, n_j, N_j are defined from a bandwidth B as in Chen et al. (2025) (Section A). For a model with L layers, window w and maximum span $c_{\max}$, the *padded* instance inserts $G \coloneqq L(w - 1) + c_{\max} - 1$ blank tokens after every block E_i, $i \geq 0$. *Layered path parity* LPP_t: vertex sets $V_0, \ldots, V_t$, maps $f_s : V_{s-1} \to V_s$, bits $b \in \{0, 1\}^{V_0}$ and a target $e \in V_t$; block $s - 1$ lists the pairs $(u, f_s(u))$, $u \in V_{s-1}$ (block 0 also lists $b(u)$), the last token holds e, and the output is the parity of b over $(f_t \circ \cdots \circ f_1)^{-1}(e)$.

3 Read depth survives sharing; index reuse within a generation adds none

A lookup chain is the natural task for a model whose keys and values are frozen: the answer of each lookup is already written somewhere in the cache, and the reader only has to ask the right questions in the right order. Such a model can ask one question per reading layer, and layers that reuse or re-rank earlier selections within a generation ask no new ones. The constructions of Theorem 3.1, Theorem 3.3(b) and Propositions F.3 and F.6 read the cache with one *retrieval head* (Lemma F.1) at precision

$$p_{\text{ret}}(n) \coloneqq 3\lceil \log_2 n \rceil + \lceil \log_2 \tau_n \rceil + 6, \qquad \tau_n \coloneqq \lceil (\lceil \log_2 n \rceil + 3) \ln 2 + \ln n \rceil, \tag{3.1}$$

so $p_{\text{ret}}(n) = 3 \log_2 n + O(\log \log n)$.

Theorem 3.1 (Read depth survives KV sharing)**.** *Let $r \geq 1$ and consider r-FuncComp, unpadded or with $G \geq 0$ blanks between consecutive blocks E_i, with prompt length n and table entries at most n. There is a typed transformer with $r + 1$ layers of dense global attention, all of KV source 0, no local attention, $H = 1$, $d = 3$, $D = r + 7$ and precision $p_{\text{ret}}(n)$ that computes r-FuncComp at the last position on every input. It has broadcast depth 1 and read depth $r + 1$, and its keys and values are the same in every layer. The same holds with a fresh top-1 selection of KV source 0 in place of dense attention in every layer.*

Proof idea. The construction is that of Chen et al. (2025, proof of Corollary 1.2); the point is that it reads a single cache. One retrieval head does all the work: with keys $(t, t^2, 1)$ and values the scaled token content, the same in every layer, the query $\tau(2\pi, -1, -\pi^2)$ scores position t by $-\tau(t - \pi)^2$, which vanishes only at the target π, so with exact partial sums the head returns the content of π. The last position fetches z_0, then $z_1(i_0)$, and so on, each time from the same cache (Section F). □

By Chen et al. (2025, Theorem 1.1), in their arithmetic model no standard transformer with r layers and $Hdp \leq n^{2^{-4r}}$ solves r-FuncComp for $r \leq \tilde{O}(\log \log n)$, whereas the model above has $Hdp = 3p = O(\log n)$ and broadcast depth 1. So the extra depth that composition needs is *read* depth. The matching lower bound uses the autoregressive communication model of Chen et al. (2025): players hold the blocks of the instance, and in each epoch every player sends its information to the holders of earlier blocks, who answer with $2B$ bits per token of the sender's block; for $\rho \geq 2$, no protocol with ρ epochs solves ρ-FuncComp (Fact B.1). Their simulation spends one epoch per layer; Lemma B.2 (Section B) spends one per *fresh-read* layer, with messages of at most $(1 + 2G)B_{\text{read}}$

bits per token, by confining sliding windows to the padding (Figure 2b) and letting re-selection and reuse layers revisit only candidates fetched when the selection was made.

Theorem 3.2 (Index reuse within a generation adds no read depth)**.** *Let M be a typed KV-sharing transformer, possibly with compressed entries, block-max pools, joint heads and raw-token windows (features (d)–(g)), of read depth $\rho \geq 1$, with L layers, window $w \geq 1$, maximum span $c_{\max}$, and any number of re-selection, reuse and local-attention layers and of KV generations. Let $G = L(w-1) + c_{\max} - 1$, and for $B \geq 1$ let I_B be the G-padded instance of ρ-FuncComp with the parameters defined from B, of length n_B.*

(a) If $(1+2G)\, B_{\mathrm{read}} \leq B$, with B_{read} evaluated at length n_B, then M errs on some input of I_B.

(b) The length is polynomial in B: $n_B \leq C_\rho B^{e_\rho} + (\rho+1)G$ with C_ρ, e_ρ depending only on ρ. Hence a fixed architecture at logarithmic precision errs on I_B for all large B: if L, w, the spans, H, d, D, k and the pool parameters are fixed while the weights and the precision $p \leq c_p \log_2 n_B$ may depend on B, then M errs on I_B for all $B \geq B_1(\rho, c_p, \text{parameters})$.

(c) The hierarchy is tight: for every B and every padding, the model of Theorem 3.1 with $r = \rho$ (broadcast depth 1, read depth $\rho+1$) computes ρ-FuncComp on every input of I_B, with precision at most $4 \log_2 n_B$ for $B \geq B_0(\rho)$.

Part (a) follows from Lemma B.2 and Fact B.1 (for $\rho = 1$, from a direct injectivity argument); Section C proves a slightly stronger form (Theorem C.1). For a fixed precision, the error statement of part (b) already follows by counting, since the last state takes at most 2^{2Dp} values while the answer ranges over at least K values; its content is the regime $p = \Theta(\log n)$ of part (c), which makes the hierarchy tight in ρ. Some padding is necessary: unpadded, z_0 sits next to the last token, so one window layer brings it to the last position and one dense read returns $z_1(z_0)$; and G grows with $L(w-1)$ because L window layers bridge any gap of fewer than $L(w-1)$ positions. The instances inherit the sizes of Chen et al. (2025), with e_ρ growing like 8^ρ, so for a fixed architecture the theorem is about very long prompts (Section 5). Reuse must stay within a generation: with cross-generation reuse, as in IndexCache and TidalDecode (Bai et al., 2026; Yang et al., 2025), a two-layer model of read depth 1 and logarithmic size computes padded 1-FuncComp on the instances of Theorem 3.2 (Proposition F.2).

3.1 One generation, windows below it: a linear hierarchy

When all global layers read one generation and the sliding windows sit below it, every question goes to two fixed tables, and pointer chasing separates read depth ρ from $\rho+1$ on instances of linear size. In *alternating pointer chasing* $\mathsf{APC}_r(m)$ the prompt lists a table $f_A : [m] \to [m]$ (one token per entry, the entry for argument i at position i), then $G_0 \geq 0$ blanks, then a table $f_B : [m] \to [m]$, then a fixed query token, so $n = 2m + G_0 + 1$; the output is v_r, where $v_0 = 1$, $v_j = f_A(v_{j-1})$ for odd j and $v_j = f_B(v_{j-1})$ for even j. For $r \geq 1$ let $\mathrm{PC}(m, r)$ be the least cost of a deterministic protocol in which Alice holds f_A, Bob holds f_B, exactly r messages are sent alternately with Bob sending the first (the order in which round hierarchies for pointer chasing are hard, Nisan & Wigderson, 1993), and the receiver of the last message outputs v_{r+1}.

Theorem 3.3 (One generation, windows below it: a linear read hierarchy)**.** *Let M be a typed KV-sharing transformer with per-token keys and values (joint heads and raw-token windows, features (f) and (g), allowed) whose global layers all have the same KV source σ_0 and whose layers above σ_0 have window 1 and read raw tokens of the current position only. Let $\rho \geq 1$ be its read depth and w its largest window, and evaluate $\mathrm{RB}(M)$ at $n = 2m + G_0 + 1$.*

(a) If $G_0 \geq \sigma_0(w-1)$ and M outputs $\mathsf{APC}_{\rho+1}(m)$ at the last position on every input, then

$$s' + 2\,\mathrm{RB}(M) \;\geq\; \mathrm{PC}(m, \rho),$$

where $s' = 2Dp$ (one hidden state) if $w \geq 2$ and $s' = 0$ if $w = 1$.

(b) Conversely, for all $r \geq 1$, $m \geq 2$ and $G_0 \geq 0$, there is a typed transformer with r layers of dense global attention, all of KV source 0 (broadcast depth 1, read depth r), no local attention, $H = 1$, $d = 3$, $D = 8$ and precision $p_{\mathrm{ret}}(n)$ that outputs $\mathsf{APC}_r(m)$ on every input, and one that selects one position afresh in every layer.

Proof idea. Split the prompt after the blanks. The windows sit below the shared KV source and the blanks are wide enough, so each player knows the KV-source states of its own positions, and above the source the state of the last position changes only through its own global reads. Both players follow that state in lockstep, each sending its part of every fresh read (exact partial sums, or a local top-k list with the listed states); grouping Alice's and Bob's parts P_e^A, P_e^B as $(S, P_1^B), (P_1^A, P_2^A), (P_2^B, P_3^B), \ldots$, where S is the last position's state at the KV source, gives ρ alternating messages, Bob first (Lemma D.1). The receiver of the last message outputs $v_{\rho+1}$, which gives (a); for (b), the retrieval head follows the pointer (Section D). □

The bandwidth that part (a) demands is the round complexity of pointer chasing with the wrong player first, which Lemma 3.4 bounds below linearly in the table size.

Lemma 3.4 (Round lower bounds for pointer chasing)**.** *There is an absolute constant $c_1 > 0$ such that for all $m \geq 2$ and $\rho \geq 1$*

$$\mathrm{PC}(m, \rho) \;\geq\; \max\Big\{ \frac{m}{1000(\rho+1)} - (\rho+1)\log_2 m - 1,\;\; c_1 m - (\rho+2)\lceil \log_2 m \rceil \Big\},$$

and $\mathrm{PC}(m, 1) \geq m \log_2 m - 1$. The first bound follows from Yehudayoff (2020, Theorem 1.1), the second from Nisan & Wigderson (1993, Theorem 2.4).

Proof idea. Let the receiver of the last message of a protocol for $\mathrm{PC}(m, \rho)$ send $v_{\rho+1}$ (for the first bound, its parity) as one more message. The result has $\rho + 1$ alternating messages, Bob first, and the round lower bounds apply (Section D). □

By Theorem 3.3(a) and Lemma 3.4, within one generation with the windows below it, read depth ρ needs read bandwidth $\Omega(m) - O(\rho \log m + Dp)$ to follow $\rho + 1$ pointers between tables beyond the windows' reach, while $\rho + 1$ reads of logarithmic size suffice; the bound is linear in m, and in the prompt length when $G_0 = O(m)$. The two hierarchies are offset by one: r-FuncComp needs $r + 1$ reads because its first lookup, of z_0, is a read, whereas the chase starts at the fixed vertex $v_0 = 1$, so APC_r needs r. A second generation changes the count, because positions other than the last can follow part of the chain: with layer 1 of KV source 0 and layers $2, \ldots, \rho$ of KV source 1, ρ dense layers of logarithmic size without sliding windows follow $2\rho - 1$ pointers, since in layer 1 the entries of f_B store $f_A \circ f_B$ (Proposition F.3). For $\rho \geq 2$ they compute $\mathsf{APC}_{\rho+1}(m)$ with $2\mathrm{RB} = 8\rho(4p + \lceil \log_2 n \rceil + 4)$, whereas by part (a) one generation needs $s' + 2\,\mathrm{RB}(M) \geq \mathrm{PC}(m, \rho)$, which is larger for fixed ρ, $G_0 = O(m)$ and m large. Function composition, whose steps depend on a query that only the last position holds, admits no such shortcut: Theorem 3.2 holds for any number of generations.

4 Broadcast Depth Does Not Survive Sharing

A join is the natural task for a model that can publish intermediate results. With one KV generation, a cut between the two blocks turns the forward pass into a two-party protocol (Lemma E.1). After Alice sends the $\sigma_0(w-1)$ tokens just left of the cut, each player knows the keys and values of its own positions. Bob simulates the *window cone* of the last position, the positions $C_\ell = [n - (L-\ell)(w-1), n]$ whose windows reach it from layer ℓ (Figure 2a); every fresh read of a position of C_ℓ costs its state (s bits) and Alice's exact partial sums or local top-k list (B_ℓ bits). Reuse and re-selection layers cost nothing, and without sliding windows both players follow the last position in lockstep. Charging $\gamma_\ell := s + B_\ell$ bits ($s = 2Dp$) for each fresh read of a cone position in a fresh-read layer ℓ, and $\gamma_\ell := 0$ in re-selection and reuse layers, the protocol costs at most

$$C(M) := \sigma_0(w-1) + \sum_{\ell \text{ global}} \big((L-\ell)(w-1) + 1\big)\gamma_\ell + 1 \tag{4.1}$$

bits: the boundary tokens (one bit each, as Alice's tokens in JP_N are bits), the reads of the $|C_\ell| = (L-\ell)(w-1) + 1$ cone positions, and one output bit. The inner product needs communication about N (Chor & Goldreich, 1988; Kushilevitz & Nisan, 1997), in the form of Lemma E.2.

Theorem 4.1 (One KV generation cannot join)**.** *Let M be a typed KV-sharing transformer with per-token keys and values (joint heads and raw-token windows, features (f) and (g), allowed) whose*

global layers all read the same KV generation σ_0 ($\sigma_0 := 0$ if there is none); M may have any number L of layers, any mixture of dense, fresh-selection, pool-producing, re-selection and reuse layers, local attention of window $w \geq 1$ in any layers, and any H, d, D, precision $p \geq 2$, sparsity and pool size. Let $N \geq \max\{1, L(w-1)\}$, and evaluate $C(M)$ of (4.1) and $\mathrm{RB}(M)$ at the length $n = 2N+1$ of JP_N.

(a) If M outputs $\mathrm{IP}(x, y)$ at the last position for every $(x, y) \in \{0,1\}^N \times \{0,1\}^N$, then $C(M) \geq N$.

(b) If M outputs $\mathrm{IP}(x, y)$ for at least a $(\frac{1}{2} + \varepsilon)$ fraction of uniformly random (x, y), then $C(M) \geq N - 2\log_2(1/(2\varepsilon))$.

(c) If $w = 1$ (no sliding windows, and raw-token windows of the current token only), then $\mathrm{RB}(M) \geq N/2$ under the hypothesis of (a), and $\mathrm{RB}(M) \geq N/2 - \log_2(1/(2\varepsilon))$ under that of (b).

Proof. Split the prompt after x: $a = N$, so $n - a = N + 1 \geq L(w-1) + 1$. Alice's tokens are bits ($\Sigma_A = \{0,1\}$), and only one output bit must be known to both players: Bob sends whether the output of M at position n is 1. Lemma E.1 gives a deterministic protocol of cost at most $C(M)$, and for $w = 1$ its lockstep version gives cost at most $2\,\mathrm{RB}(M)$ (the last token is the fixed query token). The protocol outputs $\mathrm{IP}(x, y)$ whenever M does, so Lemma E.2(a) and (b) apply. □

Part (c) says that without sliding windows (of attention or of raw tokens) the layers that read a shared cache need read bandwidth $\mathrm{RB}(M) \geq N/2$ in total, whatever their number and types. A dense read returns a sum over the whole context, but only $O(Hd(p + \log n))$ bits of it are information about Alice's part; with sliding windows the cost also charges the fresh reads in the window cone, whose positions can carry information to the last position without a global read (Section E evaluates the bound at deployed widths). The bound is tight: two generations escape it, and without sliding windows one generation meets it up to a constant factor.

Theorem 4.2 (Two KV generations join with two layers)**.** *For every $N \geq 1$ ($n = 2N+1$) there is a typed transformer with 2 layers of dense global attention, KV sources $\sigma(1) = 0$ and $\sigma(2) = 1$ (broadcast depth 2, read depth 2), no local attention, $H = 1$, $d = 3$, $D = 8$ and precision $p = 2\lceil\log_2 n\rceil + \lceil\log_2 \tau\rceil + 4$ with $\tau = \lceil\ln(8n)\rceil$, whose output at position n equals $\mathrm{IP}(x, y)$ for every input of JP_N.*

Proof idea. In layer 1, position $N+i$ retrieves x_i with the head of Theorem 3.1 and stores $u_i = x_i y_i$; in layer 2, which reads the new generation, the last position averages u over the block of y, and N times the output rounds to $\sum_i u_i$ (Section F). □

Conversely, $\lceil N/b\rceil$ dense layers of KV source 0 without sliding windows, with $H = 2$, $d = 3$, $D = 8$ and $p = b + O(\log n)$, read the two blocks in chunks of b bits and compute IP with $\mathrm{RB}(M) = \lceil N/b\rceil \cdot 8(4p + \lceil\log_2 n\rceil + 4)$, which is $O(N + \log n)$ for $b = \min\{N, \lceil\log_2 n\rceil\}$ (Proposition F.4). So without sliding windows the price of one KV generation is exactly linear, $\mathrm{RB}(M) = \Theta(N)$, against two layers of logarithmic size with two generations.

Beyond one generation. The join is the first level of layered path parity: up to the input format, JP_N is LPP_2 with f_1 a bijection, $V_2 = \{1, 2\}$, $f_2(i) = 2 - y_i$ and $e = 1$. On the LPP_2 instances of Proposition F.5, whose fibers $f_2^{-1}(v)$ have size m, a model whose global layers share one KV source needs $2\,\mathrm{RB}(M) \geq m$ without sliding windows, and with them (for $2m \geq L(w-1)$) the cost bound of Proposition F.5(a), while two dense layers with KV sources 0 and 1 and logarithmic size suffice.

Conjecture 4.3 (Broadcast hierarchy)**.** *For every $\beta \geq 1$ there are instance families of $\mathsf{LPP}_{\beta+1}$ with fibers of size m at every level such that every typed transformer of broadcast depth β without sliding windows that computes $\mathsf{LPP}_{\beta+1}$ on all of them has read bandwidth $\mathrm{RB}(M) \geq m^{\Omega(1)}/\mathrm{polylog}(n)$.*

Proposition F.5 is the case $\beta = 1$. Conversely, reads can replace one broadcast at a cost linear in the fan-in: on instances with canonical vertex labels, $|V_{\beta+1}| = 2$, level-$(\beta+1)$ fibers of size m and fixed fiber sizes below, broadcast depth β and $(\beta - 1) + 2m$ dense layers with $H = 2$, $d = 3$ and $p = O(\log n)$ compute $\mathsf{LPP}_{\beta+1}$ (Proposition F.6), so read bandwidth $O((\beta + m)\log n)$ suffices for

every β. A direct two-party cut does not reach $\beta \geq 2$: with two generations, the positions of a later block can spend their first-generation reads on *hints* that the second generation publishes, and the total capacity of these hints exceeds what a cut of LPP_3 needs. A proof must show that such hints are nearly uncorrelated with the level-two parities rather than that they are short.

5 A DEPTH LEDGER FOR KV-SHARING ARCHITECTURES

The measures of Definition 2.2 are read off an architecture's layer map (Table 1); the formal statements, with their modelling assumptions, are in Section G.

DeepSeek-V4.1-Flash. As specified in its technical report, and under the modelling assumptions of Section G (among them a cold prefill, per-token expert routing, the tie rule of Definition 2.3, aligned pool blocks, one precision p, and idealized tokens: one token per table entry, the tokenizer not modelled), the text backbone of the post-trained model is a typed KV-sharing transformer with features (d)–(g), 38 global layers, KV sources $\{2, 8, 14, 20\}$ and fresh-read layers $\{3, 9, 15, 21\}$ (Corollary G.1). By Theorem 3.2, for every choice of weights and every precision $p \leq c_p \log_2 n$ its single forward pass errs on some padded 4-FuncComp instances, which a model of broadcast depth 1 and read depth 5 solves. This is an accounting statement: with $D \geq 4 \cdot 5120$ (four residual streams of width 5120) and $p = 16$, the instances have more than 10^{100000} tokens. The pre-trained checkpoint, whose Reindex layers score the whole context, has read depth 8.

YOCO and YOIO: where the bounds bite. YOCO with a sliding-window self-decoder of $L/2$ layers and a cross-decoder of $L/2$ dense layers that read one global cache has broadcast depth 1 and read depth $L/2$, and since all its windows sit below the cache it computes JP_N only if $\frac{L}{2}(w-1) + L\,B_{\text{dense}} \geq N$ (Corollary G.2); YOIO replaces the dense reads of its 16 upper layers by one top-2048 selection that all of them reuse, so its read depth is 1 (Corollary G.4). At the published widths and precision $p = 16$ (in the format of Definition 2.3), and whatever their weights, the YOCO baseline of Sun et al. (2026) (hidden width 2560, 20 heads of dimension 128, $L = 32$, $w = 512$) fails JP_N for every $N \geq 7{,}603{,}697$ (Corollary G.2(ii)), that is, on prompts of about $1.52 \cdot 10^7$ tokens and more, and YOIO ($D = 2560$) fails $\mathsf{APC}_2(m)$ on tables of at least $1.42 \cdot 10^7$ entries $16 \cdot 512$ blanks apart (Corollary G.4), on prompts of about $2.83 \cdot 10^7$ tokens and more; both tasks are solved by two dense layers of logarithmic size (Theorem 4.2, Theorem 3.3(b)). With the tables $\frac{L}{2}(w-1)$ blanks apart, no weights let the YOCO baseline (token-wise maps reading only the current position) follow $L/2 + 1$ pointers at $p = 16$ on tables of $1.52 \cdot 10^{11}$ entries or more, whereas at logarithmic precision a YOCO model of this form follows $L/2$ (Corollary G.3); Section G.2 gives the thresholds at $p = 133$, which represents every BF16 value.

6 DISCUSSION

Read and broadcast depth measure computation, not memory: lower bounds on KV-cache size (Section I) apply unchanged at broadcast depth 1. Our lower bounds hold for every trained instance of the architectures in Section 5 under the stated modelling assumptions, for one forward pass without generated tokens, in the arithmetic of Definition 2.3 and at the stated lengths. Open, with the obstacles discussed in Section I, are the broadcast hierarchy (Conjecture 4.3), a read hierarchy for many generations at polynomial length with an exponent independent of ρ, and aggregation by sparse reads.

Takeaway. Count fresh reads and KV generations, not just layers. In our model:

- a fresh read buys one sequential lookup, which a single KV generation serves (Theorems 3.1 to 3.3, under their conditions);
- a further KV generation buys publication: intermediate positions can follow parts of a chain (Proposition F.3), and later positions can read one level of aggregation (Theorems 4.1 and 4.2);
- reuse or re-selection within a generation buys neither.

AI USE STATEMENT

The original research ideas and initial manuscript draft were developed by the authors. AI tools were used to assist with checking mathematical details and the correctness of arguments, filling in routine proof steps, drawing TikZ figures, coding assistance and language polishing. The authors take full responsibility for the content and correctness of the final manuscript.

ETHICS STATEMENT

This is a theoretical paper. It involves no human subjects, no data sets and no experiments beyond exact-arithmetic checks of the constructions; its statements about deployed architectures concern their published specifications.

REPRODUCIBILITY STATEMENT

All definitions are in Section 2 and Section A. Complete proofs of every result are in Section 4 (for Theorem 4.1) and Sections B to G, and the modelling assumptions for each architecture are stated with its corollary in Section G. The numbers in Section 5 and Sections E and G are computed by the supplementary script `ledger_numbers.py`, and the constructions were executed in exact arithmetic by the supplementary scripts described in Section H; `experiments/README.md` lists the commands.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. GQA: Training generalized multi-query Transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 4895–4901. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.298.

Yushi Bai, Qian Dong, Ting Jiang, Xin Lv, Zhengxiao Du, Aohan Zeng, Jie Tang, and Juanzi Li. IndexCache: Accelerating sparse attention via cross-layer index reuse. In *Third Conference on Language Modeling (COLM 2026)*, 2026. URL https://openreview.net/forum?id=74886M5htU. Section, figure and appendix numbers cited from arXiv:2603.12201v1.

William Brandon, Mayank Mishra, Aniruddha Nrusimha, Rameswar Panda, and Jonathan Ragan-Kelley. Reducing Transformer key-value cache size with cross-layer attention. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, pp. 86927–86957. Curran Associates, Inc., 2024. doi: 10.52202/079017-2758.

Justin Y. Chen, Ying Feng, Piotr Indyk, Michael Kapralov, Ekaterina Kochetkova, and Boris Prokhorov. Towards tight bounds for streaming attention. *arXiv preprint arXiv:2606.07205*, 2026.

Lijie Chen, Binghui Peng, and Hongxun Wu. Theoretical limitations of multi-layer Transformer. In *2025 IEEE 66th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 2631–2653. IEEE, 2025. doi: 10.1109/FOCS63196.2025.00136. Theorem, lemma, section and equation numbers cited from arXiv:2412.02975v1.

Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988. doi: 10.1137/0217015.

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 35 (NeurIPS 2022)*, pp. 16344–16359. Curran Associates, Inc., 2022. doi: 10.52202/068431-1189.

DeepSeek-AI. DeepSeek-V2: A strong, economical, and efficient mixture-of-experts language model. arXiv preprint arXiv:2405.04434, 2024.

DeepSeek-AI. DeepSeek-V3.2: Pushing the frontier of open large language models. arXiv preprint arXiv:2512.02556, 2025.

DeepSeek-AI. DeepSeek-V4: Towards highly efficient million-token context intelligence. arXiv preprint arXiv:2606.19348, 2026a.

DeepSeek-AI. DeepSeek-V4.1-Flash: Pushing the limits of KV cache compression. arXiv preprint arXiv:2609.19969, 2026b.

Angela Fan, Thibaut Lavril, Edouard Grave, Armand Joulin, and Sainbayar Sukhbaatar. Addressing some limitations of Transformers with feedback memory. *arXiv preprint arXiv:2002.09402*, 2020.

Yizhao Gao, Jianyu Wei, Qihao Zhang, Yu Cheng, Shimao Chen, Zhengju Tang, Zihan Jiang, Yifan Song, Hailin Zhang, Liang Zhao, Bo Yang, Gang Wang, Shijie Cao, and Fuli Luo. HySparse: A hybrid sparse attention architecture with oracle token selection and KV cache sharing. *arXiv preprint arXiv:2602.03560*, 2026.

Themistoklis Haris and Krzysztof Onak. Compression barriers in autoregressive transformers. In *Proceedings of Thirty Eighth Conference on Learning Theory (COLT)*, volume 291 of *Proceedings of Machine Learning Research*, pp. 2757–2785. PMLR, 2025. URL https://proceedings.mlr.press/v291/haris25a.html.

Ekaterina Kochetkova, Kshiteej Sheth, Insu Han, Amir Zandieh, and Michael Kapralov. Streaming attention approximation via discrepancy theory. In *Advances in Neural Information Processing Systems 38 (NeurIPS 2025)*. Curran Associates, Inc., 2025. doi: 10.52202/085713-0705.

Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997. doi: 10.1017/CBO9780511574948.

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of thought empowers Transformers to solve inherently serial problems. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*, pp. 11911–11943, 2024. URL https://openreview.net/forum?id=3EWTEy9MTM.

Edo Liberty, Alexandr Andoni, and Eldar Kleiner. Nearly optimal attention coresets. *arXiv preprint arXiv:2605.05602*, 2026.

William Merrill and Ashish Sabharwal. The parallelism tradeoff: Limitations of log-precision Transformers. *Transactions of the Association for Computational Linguistics*, 11:531–545, 2023. doi: 10.1162/tacl_a_00562.

William Merrill and Ashish Sabharwal. The expressive power of Transformers with chain of thought. In *The Twelfth International Conference on Learning Representations (ICLR 2024)*, pp. 7690–7706, 2024. URL https://openreview.net/forum?id=NjNGlPh8Wh.

Maxim Milakov and Natalia Gimelshein. Online normalizer calculation for softmax. *arXiv preprint arXiv:1805.02867*, 2018.

Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. *SIAM Journal on Computing*, 22(1):211–219, 1993. doi: 10.1137/0222016.

Binghui Peng, Srini Narayanan, and Christos Papadimitriou. On limitations of the Transformer architecture. In *Conference on Language Modeling (COLM)*, 2024. URL https://openreview.net/forum?id=KidynPuLNW.

Clayton Sanford, Daniel Hsu, and Matus Telgarsky. Representational strengths and limitations of transformers. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*, pp. 36677–36707. Curran Associates, Inc., 2023. doi: 10.52202/075280-1596.

Clayton Sanford, Daniel Hsu, and Matus Telgarsky. Transformers, parallel computation, and logarithmic depth. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 43276–43327. PMLR, 2024. URL https://proceedings.mlr.press/v235/sanford24a.html. Theorem numbers cited from arXiv:2402.09268v1.

Noam Shazeer. Fast Transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*, 2019.

Lena Strobl, William Merrill, Gail Weiss, David Chiang, and Dana Angluin. What formal languages can Transformers express? A survey. *Transactions of the Association for Computational Linguistics*, 12:543–561, 2024. doi: 10.1162/tacl_a_00663.

Yutao Sun, Li Dong, Yi Zhu, Shaohan Huang, Wenhui Wang, Shuming Ma, Quanlu Zhang, Jianyong Wang, and Furu Wei. You only cache once: Decoder-decoder architectures for language models. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024)*, pp. 7339–7361. Curran Associates, Inc., 2024. doi: 10.52202/079017-0235.

Yutao Sun, Yanqi Zhang, Li Dong, Jianyong Wang, and Furu Wei. You only index once: Cross-layer sparse attention with shared routing. *arXiv preprint arXiv:2606.06467*, 2026.

Xiao Wang. How much cache does reasoning need? Depth-Cache tradeoffs in KV-compressed transformers. *arXiv preprint arXiv:2604.17935*, 2026.

Haoyi Wu and Kewei Tu. Layer-condensed KV cache for efficient inference of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11175–11188. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.602.

You Wu, Haoyi Wu, and Kewei Tu. A systematic study of cross-layer KV sharing for efficient LLM inference. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pp. 396–403. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.naacl-short.34.

Lijie Yang, Zhihao Zhang, Zhuofu Chen, Zikun Li, and Zhihao Jia. TidalDecode: Fast and accurate LLM decoding with position persistent sparse attention. In *The Thirteenth International Conference on Learning Representations (ICLR 2025)*, pp. 5823–5844, 2025. URL https://openreview.net/forum?id=EkfLaCJ7bk.

Xiaowei Ye, Xiaoyu He, Chao Liao, Chen Wu, and Pinyan Lu. A provable expressiveness hierarchy in hybrid linear-full attention. *arXiv preprint arXiv:2602.01763*, 2026.

Amir Yehudayoff. Pointer chasing via triangular discrimination. *Combinatorics, Probability and Computing*, 29(4):485–494, 2020. doi: 10.1017/S0963548320000085. Preprint: ECCC TR16-151 (2016).

Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, Yuxing Wei, Lean Wang, Zhiping Xiao, Yuqing Wang, Chong Ruan, Ming Zhang, Wenfeng Liang, and Wangding Zeng. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23078–23097. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.1126.

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 33 (NeurIPS 2020)*, pp. 17283–17297. Curran Associates, Inc., 2020.

A Model details

Optional features. The following features extend Definition 2.1; each states how it interacts with the layer types.

(d) *Compressed entries.* The global attention of layer ℓ ranges over entries indexed by the aligned blocks $[(q-1)c_\ell + 1,\ qc_\ell]$ of length c_ℓ; the key, value and indexer key of block q are functions of the KV-source states $h_u^{(\sigma(\ell))}$ of the positions u of the block, and position i may attend only to blocks contained in $[1, i]$.

(e) *Block-max pools.* A pool-producing layer may select the top K_{blk} aligned blocks of c'_ℓ consecutive entries (among blocks whose entries lie in $[1, i]$) by the maximum of the pool indexer score over the block's entries; the pool is the union of the selected blocks' entries.

(f) *Joint heads.* A head may apply one softmax to the concatenation (disjoint union) of its window items (keys and values from $h_\cdot^{(\ell-1)}$ at the positions $(i-w, i]$) and its global items; a position may occur in both parts. By consequence (i) of Definition 2.3, its output is a fixed function of the $(\mathrm{Num}, \mathrm{Den})$ pairs of the two parts.

(g) *Raw-token windows.* The token-wise maps producing $h_i^{(\ell)}$, $1 \le \ell \le L$, may read the raw tokens x_j for $j \in (i - a_{\mathrm{raw}}, i]$, with $a_{\mathrm{raw}} \le w$. The embedding reads only its own token, and query, key, value and indexer maps remain functions of hidden states.

The span of an entry is c_ℓ positions and the span of a pool block $c'_\ell c_\ell$ positions; $c_{\max}$ is the largest span. A layer of type (b) or (c) uses the entries, and for (b) the pool, of its producing layer: same entry partition and block lengths; only its queries, its indexer query and the maps applied to the entries' source states may differ. All maps may depend on the position of the token they are applied to (as rotary embeddings do); positions are public in every simulation below. Tokens come from a finite alphabet that may depend on n, as in Chen et al. (2025), where one token holds one table entry.

Local receptive field. If layers $1, \dots, \ell$ have no global attention, then $h_i^{(\ell)}$ depends only on the tokens x_j with $j \in [i - \ell(w-1), i]$: each window and each raw-token window reaches at most $w-1$ positions back, and induction on ℓ adds the reaches. More generally, through local attention alone the state $h_n^{(L)}$ depends on the layer-ℓ states of the positions in the *cone*

$$C_\ell := [\, n - (L-\ell)(w-1),\ n \,], \qquad |C_\ell| = (L-\ell)(w-1) + 1. \tag{A.1}$$

Broadcast depth. Broadcast depth counts KV sources, not the information in them: two sources with no global layer between them are counted twice, although the second adds only local mixing.

Arithmetic. For a sum over at most n items, $2^{2p}\mathrm{Num}$ and $2^p\mathrm{Den}$ are integers of absolute value below $n2^{4p}$, so each coordinate of an exact partial sum has a fixed-length code of

$$4p + \lceil \log_2 n \rceil + 4 \text{ bits};$$

the simulations below transmit only partial sums over global items. We say that a model has *logarithmic size* if H, d, D, the sparsity k, the pool sizes and the spans are constants and $p = O(\log n)$; each of its reads then has $O(\log n)$ bits. Two consequences are used throughout: for every partition $S = S_1 \cup S_2$ of the attended items,

(i) the output of a head is a fixed function of $(\mathrm{Num}, \mathrm{Den})$ over S_1 and over S_2; and

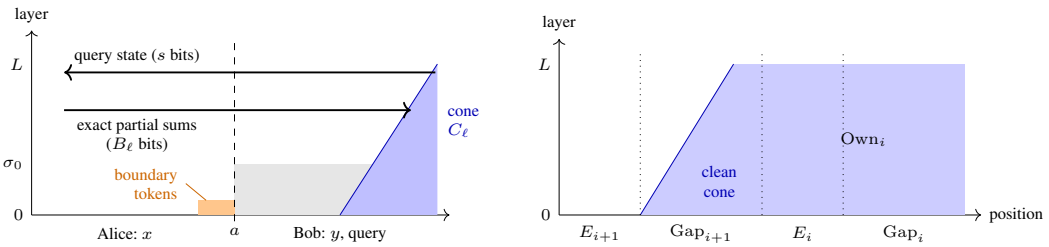
(ii) a top-k selection over S is a fixed function of the top-k lists (positions and scores) over S_1 and S_2.

The convention models an accumulator that is wide enough for the sums it forms, and it is the convention implicit in the simulation of Chen et al. (2025, Lemma 3.1), which also transmits partial sums of weights and weighted values. With rounded accumulation the simulations would need a canonical summation order, and a kernel that subtracts a running maximum before exponentiating (Milakov & Gimelshein, 2018; Dao et al., 2022) is not covered literally.

Composition parameters. The parameters K, m, n_j, N_j of r-FuncComp are defined from a bandwidth B by (Chen et al., 2025, eq. (4),(5)), with B in place of Hdp:

$$K = (Br)^8\, 8^{2r^2}, \qquad m = K^{1+\sum_{j=0}^{r-1} 8^j}, \qquad n_j = K^{4\cdot 8^{r-j-1}}, \qquad N_j = m \prod_{j' \le j} n_{j'}. \tag{A.2}$$

We write $m_{(i)} := |E_i|$ for the number of tokens of block E_i.



(a) one KV generation: a two-party protocol

(b) typed simulation: Sim_i (shaded) for player i

Figure 2: The two simulations. (a) With one KV generation (Lemma E.1), Bob computes the states of the positions whose windows reach the last position (the cone) and all KV-source states of his part (grey), after Alice sends the $\sigma_0(w-1)$ tokens just left of the cut; each fresh read of a cone position costs one query state and one exact partial sum. (b) In the typed simulation (Lemma B.2), player i owns its block and the padding after it, and also simulates the part of the preceding padding that no window path from the preceding block reaches within ℓ layers; the padding $G = L(w-1) + c_{\max} - 1$ keeps the preceding block out of every window cone and lets owners of entries that straddle a boundary compute them.

B TYPED SIMULATION

Communication. Two-party protocols are deterministic protocol trees: the speaker and the decision to halt are functions of the transcript, a message bit is a function of the speaker's input and the transcript, and the output is a function of the final transcript or is computed by one designated party. The cost is the worst-case number of bits. The autoregressive communication model of Chen et al. (2025, Section 3) has players $-1, 0, \ldots, r$, player i holding the block E_i of a prompt, so that a larger index means an earlier block. Initially the information $X_i^{(0)}$ of player i is its block. In each epoch every player i sends its whole information to every player $j > i$, and each such j answers with a message $\Pi_{j,i}$ of exactly $2B\, m_{(i)}$ bits computed from its own and player i's information; each player then adds the answers it received to its information $X_i^{(e)}$, and player -1 outputs after the last epoch. We allow the number of epochs to differ from the number of players minus two; for r epochs and players $-1, \ldots, r$ it is their model.

Fact B.1 (Chen et al., 2025, Theorem 4.1)**.** *Let $r \geq 2$ and $B \geq 1$, and let r-FuncComp have the parameters defined from B. No deterministic autoregressive communication protocol with r epochs and message bits B solves r-FuncComp.*

Chen et al. (2025) state the theorem with $B = Hdp$; taking $H = B$ and $d = p = 1$ gives every integer $B \geq 1$. Their Lemma 3.1 simulates a standard r-layer transformer by such a protocol. That simulation already applies, verbatim, to dense layers whose keys and values come from any earlier layer: a responder computes its partial sums from its own states. What Lemma B.2 adds is the treatment of re-selection and reuse layers and of sliding windows.

Lemma B.2 (Epochs are fresh-read layers)**.** *Let M be a typed KV-sharing transformer, possibly with features (d)–(g), with L layers, window $w \geq 1$, maximum span $c_{\max}$ and read depth $\rho \geq 1$. Let $r \geq 1$, let $m, n_1, \ldots, n_{r-1}$ be any positive integers, $N_0 = m$ and $N_j = n_j N_{j-1}$, defining r-FuncComp, and let $G = L(w-1) + c_{\max} - 1$. Consider the instance laid out as in Section 2 (blocks $E_r, \ldots, E_1, E_0, E_{-1}$, $m_{(i)} = |E_i|$), padded with G blank tokens after every block E_i, $i \geq 0$; let n be its length and B_{read} the per-read bandwidth of M at length n. Then there is a deterministic protocol, depending only on M and the layout, in the autoregressive communication model with players $-1, \ldots, r$ and exactly ρ epochs, in which every message $\Pi_{j,i}$ has at most $B^\star m_{(i)}$ bits, where*

$$B^\star := (1 + 2G)\, B_{\mathrm{read}},$$

and after which the information of player -1 determines the hidden state $h_n^{(L)}$ of M at the last position. For every integer $B \geq B^\star$, padding each message u to $u10^q$ of length exactly $2Bm_{(i)}$ gives the message length required by Chen et al. (2025).

Proof overview. Player i owns its block E_i and the gap Gap_i of G blanks after it; an entry or pool block is owned by the owner of its last position. Player i simulates the pairs (t, ℓ) with t in its own

part, and in addition the pairs with $t \in \mathrm{Gap}_{i+1}$ whose distance from the end of E_{i+1} exceeds $\ell(w-1)$ (Figure 2b). This *clean cone* is closed under the windows of all layers (Claim B.3); its states depend only on blanks, on the player's own tokens and on global reads. Because $G \geq L(w-1) + c_{\max} - 1$, the owner of an entry whose span straddles a boundary simulates every position of the span at every KV source (Claim B.5), so it can answer reads for its entries. In epoch e each player i sends its information to the later players, who evaluate the queries of player i's simulated positions at the e-th fresh-read layer and return their exact partial sums or local top-k lists together with the KV-source states of the listed entries; there are at most $|E_i| + 2G \leq (1+2G)m_{(i)}$ such positions, and each answer has at most B_{read} bits. Between two fresh-read layers all global layers are of type (b) or (c). Following the chain of producing layers back to its type-(a) root, which lies at or before the current fresh-read layer, the player recomputes each re-selection from the stored pool and each reuse from the stored selection, without communication.

Proof of Lemma B.2. We proceed in six steps.

Step 1: ownership and simulated sets. Let Gap_i be the G blank positions right after E_i ($i \geq 0$), so that $\min E_i = \max E_{i+1} + G + 1$. Player i owns

$$\mathrm{Own}_i \coloneqq E_i \cup \mathrm{Gap}_i \qquad (\text{player } -1 \text{ owns } E_{-1});$$

these sets partition $[1, n]$ into consecutive intervals in the order $\mathrm{Own}_r, \ldots, \mathrm{Own}_0, \mathrm{Own}_{-1}$. An entry and a pool block are owned by the owner of their last position. For $t \in \mathrm{Gap}_{i+1}$ let $\mathrm{dist}(t) \coloneqq t - \max E_{i+1} \in [1, G]$ be its distance from the preceding block. The set of pairs that player i simulates is

$$\mathrm{Sim}_i \coloneqq \{(t,\ell) : t \in \mathrm{Own}_i,\ 0 \leq \ell \leq L\} \cup \{(t,\ell) : t \in \mathrm{Gap}_{i+1},\ 0 \leq \ell \leq L,\ \mathrm{dist}(t) > \ell(w-1)\}, \tag{B.1}$$

the second part being empty for $i = r$. Sim_i is closed downward in ℓ.

Step 2: what a player can compute on its own. The simulated sets have four properties, used in every epoch.

Claim B.3 (Closure under windows). If $(t,\ell) \in \mathrm{Sim}_i$, $\ell \geq 1$ and $t' \in (t-w, t]$, then $(t', \ell-1) \in \mathrm{Sim}_i$.

Proof. If $t' \in \mathrm{Own}_i$ there is nothing to show. Otherwise $t' < \min E_i$ and $t' \geq t - (w-1)$.

- If $t \in \mathrm{Gap}_{i+1}$, then $t' - \max E_{i+1} \geq \mathrm{dist}(t) - (w-1) > (\ell-1)(w-1) \geq 0$ because $\mathrm{dist}(t) > \ell(w-1)$, so $t' \in \mathrm{Gap}_{i+1}$ and $\mathrm{dist}(t') > (\ell-1)(w-1)$.
- If $t \in \mathrm{Own}_i$, then $t \geq \min E_i$, so

$$\begin{aligned} t' - \max E_{i+1} \ &\geq\ \min E_i - (w-1) - \max E_{i+1} \ =\ G - (w-2) \\ &=\ (L-1)(w-1) + c_{\max} \ >\ (\ell-1)(w-1), \end{aligned}$$

  hence again $t' \in \mathrm{Gap}_{i+1}$ and $\mathrm{dist}(t') > (\ell-1)(w-1)$. □

Claim B.4 (Raw tokens and joint heads). If $(t,\ell) \in \mathrm{Sim}_i$, $\ell \geq 1$, and the token-wise maps producing $h_t^{(\ell)}$ read $x_{t'}$ with $t - a_{\mathrm{raw}} < t' \leq t$, then $(t', 0) \in \mathrm{Sim}_i$. For a joint head at (t,ℓ), the window items are computed from states at pairs $(t', \ell-1)$, $t' \in (t-w, t]$, all of which lie in Sim_i.

Proof. Since $a_{\mathrm{raw}} \leq w$, Claim B.3 gives $(t', \ell-1) \in \mathrm{Sim}_i$, and downward closure gives $(t', 0) \in \mathrm{Sim}_i$; the second statement is Claim B.3 itself. □

Claim B.5 (Owners can form their entries). Let ℓ be a global layer, let an entry or pool block be owned by player j, and let t be a position of its span. Then $(t, \sigma(\ell)) \in \mathrm{Sim}_j$.

Proof. Suppose $t \notin \mathrm{Own}_j$. The span is an interval of at most $c_{\max}$ positions ending in Own_j, so $\min E_j - (c_{\max} - 1) \leq t < \min E_j$. As $G \geq c_{\max} - 1$, $t \in \mathrm{Gap}_{j+1}$ and, because $\sigma(\ell) \leq L - 1$,

$$\mathrm{dist}(t) \ \geq\ G - (c_{\max} - 2) \ =\ L(w-1) + 1 \ >\ \sigma(\ell)(w-1).$$ □

Claim B.6 (Local computation). For $(t, \ell) \in \mathrm{Sim}_i$, the state $h_t^{(\ell)}$ is a function of the tokens at the positions t' with $(t', 0) \in \mathrm{Sim}_i$ (player i's input and fixed blanks) and of the global parts of the heads at pairs of Sim_i: the exact $(\mathrm{Num}, \mathrm{Den})$ of the global items, or the selected entries with the KV-source states of their spans.

Proof. By Claims B.3 and B.4 and induction on ℓ. □

Step 3: the invariant. Let $\ell_1 < \cdots < \ell_\rho$ be the fresh-read layers, $\ell_0 := 0$ and $\ell_{\rho+1} := L+1$; epoch e handles layer ℓ_e. For $e = 0, \ldots, \rho$, the invariant $I(e)$ states that from $X_i^{(e)}$ player i can compute

(I1) $h_t^{(\ell)}$ for all $(t, \ell) \in \mathrm{Sim}_i$ with $\ell < \ell_{e+1}$, and

(I2) for every $(t, \ell') \in \mathrm{Sim}_i$ with $\ell' \le \ell_e$ of type (a), the selected entries and pool blocks of position t at layer ℓ' together with the KV-source states of their spans.

Base case. Layers before ℓ_1 have no dense or type-(a) global attention, hence no global attention at all (a layer of type (b) or (c) needs an earlier type-(a) layer at the same position). By Claim B.6, player i computes $h_t^{(\ell)}$ for $(t, \ell) \in \mathrm{Sim}_i$, $\ell < \ell_1$, from $X_i^{(0)}$; (I2) is vacuous. So $I(0)$ holds.

Step 4: one epoch. Assume $I(e-1)$. Player i sends $X_i^{(e-1)}$ to every $j > i$. Let R_i be the set of positions t with $(t, \ell_e) \in \mathrm{Sim}_i$, and fix $t \in R_i$. The entries and pool blocks that t may use at layer ℓ_e are those whose span lies in $[1, t]$; each is owned by a player $j \ge i$, and by a player $j > i$ if $t \in \mathrm{Gap}_{i+1}$.

- *What a responder knows.* A responder $j > i$ computes, by $I(e-1)$ for itself and Claim B.5 (as $\sigma(\ell_e) < \ell_e$), the KV-source states of the spans of all entries and pool blocks it owns, hence their keys, values and indexer keys; by $I(e-1)$ for player i, evaluated on $X_i^{(e-1)}$, it knows $h_t^{(\ell_e - 1)}$ and hence all queries of t.
- *What it sends for t.* For a dense layer, the exact partial numerators and denominators of the global items it owns in $[1, t]$ (at most B_{dense} bits); for a type-(a) layer, its local top-k owned entries in $[1, t]$ with position, rounded indexer score and the KV-source states of the span, and for a pool-producing layer also its local top-K_{blk} owned pool blocks with scores and span states (at most $B_{\mathrm{sparse}} + B_{\mathrm{pool}}$ bits). All fields use the fixed-length codes of Definition 2.3, and the number of listed entries is determined by public data, so messages are parsed without delimiters. Player i computes the same over the entries and pool blocks it owns in $[1, t]$.
- *How player i combines the answers.* By consequences (i) and (ii) of Definition 2.3—a top-k selection over a disjoint union is determined by the parts' top-k lists, and a block-max score is determined by the block's owner, who holds the whole span—player i obtains the exact global part at (t, ℓ_e) and stores the selected entries and pool blocks with their span states. For a joint head it computes the $(\mathrm{Num}, \mathrm{Den})$ of the window items itself (Claim B.4) and combines the two parts.

Since $|R_i| \le |\mathrm{Own}_i| + |\mathrm{Gap}_{i+1}| \le |E_i| + 2G \le (1+2G) m_{(i)}$, every message has at most

$$(1+2G)\, m_{(i)}\, B_{\mathrm{read}} \;=\; B^\star m_{(i)} \text{ bits.}$$

Step 5: restoring the invariant. Player i computes layer ℓ_e at every $(t, \ell_e) \in \mathrm{Sim}_i$: global parts as in Step 4, local parts and token-wise maps by Claims B.3 and B.4. Every layer ℓ with $\ell_e < \ell < \ell_{e+1}$ has only local attention, token-wise maps, and global attention of type (b) or (c). Let $(t, \ell) \in \mathrm{Sim}_i$ with layer ℓ of type (b) or (c), and follow the chain of producing layers from ℓ back to its type-(a) root ℓ_{root} (every layer of type (b) or (c) is produced by an earlier layer of type (a) or (b) at the same position with the same KV source).

- No fresh-read layer lies in (ℓ_e, ℓ), so $\ell_{\mathrm{root}} \le \ell_e$ and $\sigma(\ell_{\mathrm{root}}) = \sigma(\ell)$, and $(t, \ell'') \in \mathrm{Sim}_i$ for every layer $\ell'' \le \ell$ of the chain by downward closure.

- By (I2)—for $\ell_{\text{root}} = \ell_e$ just stored, for $\ell_{\text{root}} < \ell_e$ by $I(e-1)$—player i holds the selection and the pool of (t, ℓ_{root}) with the KV-source states of their spans.
- Proceeding in increasing layer order, it recomputes each re-selection of the chain, at a layer ℓ_b of type (b), as the top-k over the stored pool, from the pool's span states, which determine the indexer keys, and from the indexer query, a function of the already computed $h_t^{(\ell_b-1)}$; a reuse layer copies its producer's selection.
- The keys and values of the selected entries are functions of the stored span states, so these heads, including the window parts of joint heads, need no communication.

With Claim B.3 this yields $I(e)$.

Step 6: conclusion. By $I(\rho)$, player -1 computes $h_t^{(\ell)}$ for all $(t, \ell) \in \text{Sim}_{-1}$, including (n, L). For the last sentence of the lemma: $B_{\text{read}} \geq 1$, so $Bm_{(i)} \geq B^\star m_{(i)} \geq |u|$ and $|u| + 1 \leq 2Bm_{(i)}$; the map $u \mapsto u10^q$ is injective and fixed, so receivers decode it. □

C Proof of Theorem 3.2

We prove the following form of Theorem 3.2. It adds the explicit condition (b)(ii) and states the tightness in (c) for classes of model families; Theorem 3.2 consists of its parts (a), (b)(i) with the length bound, and the statement about the model of Theorem 3.1 in (c).

Theorem C.1 (Full form of Theorem 3.2)**.** *Let $\rho \geq 1$ and let M be a typed KV-sharing transformer, possibly with features (d)–(g), with read depth ρ, L layers, window $w \geq 1$ and maximum span $c_{\max}$; M may contain any number of pool-restricted re-selection, reuse and local-attention layers and any number of KV generations. Let $G = L(w-1) + c_{\max} - 1$. For an integer $B \geq 1$ let I_B be the instance of ρ-FuncComp with the parameters defined from B, padded with G blanks after every block E_i, $i \geq 0$, and let n_B be its length.*

(a) If the per-read bandwidth of M at length n_B satisfies $(1+2G)\, B_{\text{read}} \leq B$, then the output of M at the last position differs from ρ-FuncComp on some input of I_B.

(b) There are constants $C_\rho, e_\rho \geq 1$ depending only on ρ such that $n_B \leq C_\rho B^{e_\rho} + (\rho+1)G$ for every $B \geq 1$. Consequently, (i) for every family $(M_B)_{B\geq 1}$ of such models, one for each instance, whose numeric parameters other than the precision (L, w, the spans, H, d, D, k and the pool parameters) are fixed and whose precisions satisfy $p(M_B) \leq c_p \log_2 n_B$ for a constant c_p, there is $B_1 = B_1(\rho, c_p, \text{parameters})$ such that M_B errs on I_B for all $B \geq B_1$; and (ii) in general M errs on I_B whenever $(1+2G)\, B_{\text{read}} \leq \big((n_B - (\rho+1)G)/C_\rho\big)^{1/e_\rho}$.

(c) For $c_p > 0$ let $\mathcal{F}_{c_p}$ be the class of families $(M_B)_{B\geq B_*}$ of typed models of any read depth, one for each instance from some B_* on, whose numeric parameters other than the precision are fixed and whose precisions satisfy $p(M_B) \leq c_p \log_2 n_B$. For every $B \geq 1$ and every padding $G \geq 0$, the model of Theorem 3.1 for ρ-FuncComp (broadcast depth 1, read depth $\rho+1$, $H=1$, $d=3$, $D=\rho+7$) computes ρ-FuncComp on every input of I_B, and its precision is at most $4\log_2 n_B$ for $B \geq B_0(\rho)$. So this family, taken from B_0 on, lies in $\mathcal{F}_4$ and is correct on every I_B, whereas every family of read depth ρ in $\mathcal{F}_4$ errs on I_B for every $B \geq B_1$ in its range, by (b)(i).

Proof. **Part (a).** Suppose M is correct on I_B. The padding of I_B is the one Lemma B.2 requires, and the blanks carry no input, so the players hold exactly the inputs of the communication version of ρ-FuncComp. Lemma B.2 with $r = \rho$ gives a protocol with ρ epochs and messages of at most

$$(1+2G)\, B_{\text{read}}\, m_{(i)} \;\leq\; B\, m_{(i)} \text{ bits},$$

which its last sentence turns into messages of exactly $2Bm_{(i)}$ bits. For $\rho \geq 2$ this contradicts Fact B.1. For $\rho = 1$ the parameters are $K = 64B^8$ and $m = K^2 = N_0$, and the task is $i_1 = z_1(z_0)$ for $z_1 : [m] \to [m]$ and $z_0 \in [m]$. With one epoch, the only information that player -1 receives about z_1 is the message $\Pi_{1,-1}(z_1, \omega)$ of $2B$ bits, a function of (z_1, ω) alone. If two tables $z_1 \neq z_1'$ gave the

same message, player -1 would answer identically on (z_1, a) and (z_1', a) for an argument a where they differ. So $z_1 \mapsto \Pi_{1,-1}(z_1, \omega)$ is injective on the m^m tables, and

$$2B \;\geq\; m \log_2 m \;\geq\; 64B^8,$$

which is false.

Instance length. By (A.2) with $r = \rho$, $K = (B\rho)^8\, 8^{2\rho^2}$, and m, every n_j and every N_j are powers K^e with exponents depending only on ρ (for $\rho = 4$: $m = K^{586}$ and $N_3 = K^{878}$). The unpadded prompt has $\sum_\ell N_{\ell-1} + 2$ tokens, which is at most $C_\rho B^{e_\rho}$ for constants depending only on ρ, and the padding adds $(\rho+1)G$ tokens:

$$n_B \;\leq\; C_\rho B^{e_\rho} + (\rho+1)G.$$

Part (b)(i). For a fixed p the claim follows by counting: $h_n^{(L)}$ takes fewer than 2^{2Dp} values, while ρ-FuncComp takes $N_{\rho-1} \geq K$ values on I_B. In general, every per-read quantity of Definition 2.4 is affine in $(\lceil \log_2 n \rceil, p)$ with nonnegative coefficients determined by the numeric parameters:

$$H(d+1)\big(4p + \lceil \log_2 n \rceil + 4\big), \qquad k\big(\lceil \log_2 n \rceil + 2p + 2cDp\big), \qquad K_{\mathrm{blk}}\big(\lceil \log_2 n \rceil + 2p + 2c'cDp\big)$$

(the last for pool-producing layers). With p fixed or $p \leq c_p \log_2 n$, therefore, for constants a, b',

$$\begin{aligned}(1+2G)\, B_{\mathrm{read}} \;&\leq\; a \log_2 n_B + b',\\ \log_2 n_B \;&\leq\; \log_2\big(C_\rho B^{e_\rho} + (\rho+1)G\big) \;=\; O(\log B).\end{aligned}$$

Hence $(1+2G)B_{\mathrm{read}} \leq B$ for all large B, and part (a) applies.

Part (b)(ii). The bound on n_B gives $B \geq ((n_B - (\rho+1)G)/C_\rho)^{1/e_\rho}$, so the displayed condition implies $(1+2G)B_{\mathrm{read}} \leq B$, and part (a) applies.

Part (c). Apply Theorem 3.1 with $r = \rho$ to ρ-FuncComp padded with G blanks between consecutive blocks E_i, which is the padding of I_B; the table entries are at most n_B. Its precision is $p_{\mathrm{ret}}(n)$ of (3.1) with $\tau_n \leq 2\log_2 n + 4$, so for $n = n_B$ and all large B

$$p_{\mathrm{ret}}(n) \;\leq\; 3\log_2 n + \log_2 \log_2 n + 12 \;\leq\; 4 \log_2 n. \qquad \square$$

Remarks.

- Without sliding windows ($w = 1$) the factor $1 + 2G = 2c_{\max} - 1$ does not depend on the number of layers: a model with ρ fresh selections and arbitrarily many reuse and re-selection layers is, on these instances, no stronger than ρ fresh reads of the same bandwidth.
- Sliding windows enter only through $G = L(w-1) + c_{\max} - 1$, and this is necessary in general, since $L(w-1) \geq n$ lets windows alone mix the whole context. The padding itself is essential: in the unpadded layout, z_0 sits next to the last token, so one window layer brings it to the last position and a single dense read then returns $z_1(z_0)$.
- The instances inherit the sizes of Chen et al. (2025): the exponent e_ρ grows like 8^ρ, so for a fixed architecture the theorem is about very long prompts (Corollary G.1).

D Proofs for Section 3.1

We first isolate the protocol behind Theorem 3.3(a), then prove the theorem and the round lower bounds of Lemma 3.4; the form of Theorem 3.3 for YOCO is Corollary G.3 in Section G.2.

D.1 The lockstep protocol

Lemma D.1 (Lockstep protocol). *Let M be as in Theorem 3.3, with read depth $\rho \geq 1$, and let $G_0 \geq \sigma_0(w-1)$. Give Alice the first $a := m + G_0$ tokens of the prompt of $\mathsf{APC}_{\rho+1}(m)$ (her table and the blanks) and Bob the rest (his table and the query token). There is a deterministic protocol with exactly ρ messages, sent alternately with Bob first, of total length at most $s' + 2\,\mathrm{RB}(M)$, after which the receiver of the last message knows the output of M at the last position.*

Proof. Positions are public. The proof has four steps.

Step 1: source states. A global layer ℓ has $\sigma_0 = \sigma(\ell) \le \ell - 1$, so no layer at or below σ_0 has global attention, and by the local receptive field (Section A; windows and raw-token windows reach at most $w-1$ positions back) $h_j^{(\sigma_0)}$ depends only on the tokens in $[j - \sigma_0(w-1), j]$. For $j \le a$ these are Alice's. For $j > a$ they lie in $[m+1, n]$, since $G_0 \ge \sigma_0(w-1)$, and are blanks or Bob's. So each player knows the keys, values and indexer keys of its own positions in every global layer.

Step 2: the two parts of a read. Above σ_0 every layer has window 1 and reads raw tokens of the current position only, so $h_n^{(\ell)}$ is a function of $h_n^{(\ell-1)}$, x_n and the outputs at n of the global heads of layer ℓ (the window item of a joint head is position n itself). Let $\ell_1 < \cdots < \ell_\rho$ be the fresh-read layers; a type-(b) or type-(c) layer refers to an earlier type-(a) layer, so every global layer that is not a fresh-read layer lies after ℓ_1. For $e \le \rho$ let P_e^A be Alice's part of the read of position n in layer ℓ_e: her exact partial sums for every head of a dense layer; for a type-(a) layer, which makes one selection for all its heads, her local top-k list and, if the layer produces a pool, her local top-K_{pool} list, with positions, rounded scores and KV-source states. Let P_e^B be Bob's. Both players know a and $n-a$, so each part has a fixed-length code of at most B_{ℓ_e} bits. Let

$$S := h_n^{(\sigma_0)} \ (s \text{ bits}) \text{ if } w \ge 2, \qquad S := \emptyset \text{ if } w = 1,$$

in which case $h_n^{(\sigma_0)}$ is a function of the query token.

Step 3: invariant. We show by strong induction on $\ell > \sigma_0$:

- $h_n^{(\ell)}$, and every selection or candidate pool made at position n by a layer $\ell' \le \ell$ together with the KV-source states of its members, are functions of S and of the parts P_e^A, P_e^B with $\ell_e \le \ell$;
- P_e^A (resp. P_e^B) is a function of $h_n^{(\ell_e - 1)}$ and Alice's (resp. Bob's) input.

Layers without global attention act token-wise at n. At ℓ_e the selection and the pool are the top k and the top K_{pool} of the union of the lists in P_e^A and P_e^B, and the heads are functions of $h_n^{(\ell_e-1)}$, P_e^A and P_e^B by consequences (i) and (ii) of Definition 2.3. A type-(b) layer re-selects, with a fresh query that is a function of $h_n^{(\ell-1)}$, inside a pool made at n by an earlier type-(a) layer, and a type-(c) layer copies a selection made at n by an earlier type-(a) or type-(b) layer; by the induction hypothesis these pools and selections and the states of their members are functions of S and the parts with $\ell_e \le \ell$, and the keys and values of the members are functions of their states because the KV source is σ_0.

Step 4: messages. The messages are

$$M_1 = (S, P_1^B) \text{ (Bob)}, \qquad M_j = \begin{cases} (P_{j-1}^A, P_j^A) & \text{(Alice) for even } j, \\ (P_{j-1}^B, P_j^B) & \text{(Bob) for odd } j, \end{cases} \qquad 2 \le j \le \rho.$$

Before sending M_j, its sender has sent its own parts of index at most $j-2$ and has received S and the other player's parts of index at most $j-1$; by the invariant it computes $h_n^{(\ell_{j-1}-1)}$, its own part of index $j-1$, $h_n^{(\ell_{j-1})}$, $h_n^{(\ell_j - 1)}$ and its own part of index j. After M_ρ, the receiver knows S, all of the sender's parts and its own parts of index at most $\rho - 1$; it computes its part of index ρ and then $h_n^{(L)}$ and the output of M. The ρ messages alternate, Bob sends the first, and their total length is at most

$$|S| + 2\sum_{e=1}^{\rho} B_{\ell_e} \ \le \ s' + 2\,\mathrm{RB}(M). \qquad \square$$

D.2 Proof of Theorem 3.3

Proof of Theorem 3.3. Part (a). If M outputs $v_{\rho+1}$ on every input, so does the receiver of the last message of the protocol of Lemma D.1, whose cost is therefore at least $\mathrm{PC}(m, \rho)$.

Part (b). Use the retrieval head of Lemma F.1 with the integer content $P(t) \le m$ of the token at t (0 at blanks and at the query token). All positions other than the last use zero queries. The last position

stores $v_0 = 1$; in layer $j \le r$ it targets

position v_{j-1} (the entry of f_A) if j is odd, position $m+G_0+v_{j-1}$ (the entry of f_B) if j is even,

and stores v_j. After layer r it outputs v_r. For the sparse variant the same scores serve as indexer scores, with a unique maximum at the target. The width $D = 8$ holds P, t, t^2, 1, the current pointer, the head output and two spare coordinates. □

D.3 Proof of Lemma 3.4

Proof of Lemma 3.4. **The first bound.** We use the round lower bound of Yehudayoff (2020, Theorem 1.1). Let $V_A = \{1, \dots, m\}$ and $V_B = \{m+1, \dots, 2m\}$; with Alice holding $f_A : V_A \to V_B$ and Bob holding $f_B : V_B \to V_A$, and with $z_0 = 1$, $z_1 = f_A(z_0)$, $z_2 = f_B(z_1), \dots$, every ρ'-round protocol in which Bob speaks first and which computes $z_{\rho'} \bmod 2$ with error at most $1/3$ under the uniform distribution has length at least

$$\frac{m}{1000\rho'} - \rho' \log m;$$

the output is determined by the transcript. Given a protocol for $\mathrm{PC}(m, \rho)$ of cost c, relabel $f_A(i) \mapsto m + f_A(i)$ and $f_B(m+j) := f_B(j)$, a bijection of input pairs under which $z_t = v_t$ for even t and $z_t = m + v_t$ for odd t, and let the receiver of the ρ-th message send one more bit, $z_{\rho+1} \bmod 2$, which is a function of $v_{\rho+1}$. The result has $\rho + 1$ alternating messages, Bob first, error 0 and length $c + 1$, so

$$c + 1 \;\ge\; \frac{m}{1000(\rho+1)} - (\rho+1)\log m$$

(with $\log = \log_2$, or a fortiori if log is natural). For $m < 1000$ the stated bound is negative.

The second bound. It comes from Nisan & Wigderson (1993, Theorem 2.4) (their n is our m): in their model a K-round protocol sends K messages alternately such that at the end both players know $g_K = f^{(K)}(v_0)$ for v_0 in Alice's vertex set, and the deterministic complexity with Bob first is $\Omega(n - K \log n)$ for every K. Their proof adds the current vertex to every message, at most $K\lceil \log_2 n \rceil$ bits in all, and shows that the augmented protocol needs more than $\epsilon n/2$ bits for an absolute $\epsilon > 0$; so the complexity is at least

$$c_1 n - K\lceil \log_2 n \rceil$$

for an absolute $c_1 > 0$ and all $n \ge 2$, after shrinking c_1 to cover small n (for $K \ge 2$ every protocol costs at least one bit, and for $K = 1$ none exists). Appending $v_{\rho+1}$ as a message of $\lceil \log_2 m \rceil$ bits turns a protocol for $\mathrm{PC}(m, \rho)$ into such a protocol for $g_{\rho+1}$ with $n = m$, so

$$c + \lceil \log_2 m \rceil \;\ge\; c_1 m - (\rho+1)\lceil \log_2 m \rceil,$$

which is the second bound.

One message. For $\rho = 1$, Bob sends one message $M(f_B)$ and Alice outputs $f_B(f_A(1))$. If $M(f_B) = M(f'_B)$ for $f_B \ne f'_B$, choose a with $f_B(a) \ne f'_B(a)$ and $f_A(1) = a$; Alice errs on one of the two inputs. So M is injective on the m^m functions f_B, and fewer than 2^{c+1} strings have length at most c, so $c > m \log_2 m - 1$. □

Remark. The two hierarchies are offset by one: r-FuncComp needs $r + 1$ reads because its first lookup, of z_0, is a read, whereas APC_r needs r.

E Proofs for Section 4

Lemma E.1 (Two-party simulation of one KV generation)**.** *Let M be a typed KV-sharing transformer, possibly with features (f) and (g) but without (d) and (e), with L layers, alphabet Σ, precision $p \ge 2$, width D and window $w \ge 1$, whose global layers all have the same KV source σ_0 ($\sigma_0 := 0$ if there is none). For every global layer ℓ let $\gamma_\ell := s + B_\ell$ if ℓ is dense or of type (a), and $\gamma_\ell := 0$ if it is of type (b) or (c). Let $1 \le a < n$ with $n - a \ge L(w-1) + 1$, and let Alice hold $x_1 \cdots x_a$ and Bob hold $x_{a+1} \cdots x_n$.*

(a) *There is a deterministic protocol, depending only on* (M, n, a)*, after which both players know* $h_n^{(L)}$*, of cost at most*

$$\sigma_0(w-1)\lceil \log_2 |\Sigma| \rceil + \sum_{\ell \text{ global}} \big((L-\ell)(w-1)+1\big)\gamma_\ell + s;$$

if only c_{out} *output bits must be known to both, the final* s *may be replaced by* c_{out}*.*

(b) *In the boundary term* $\sigma_0(w-1)\lceil \log_2 |\Sigma| \rceil$*,* $|\Sigma|$ *may be replaced by* $|\Sigma_A|$ *for a fixed nonempty set* $\Sigma_A \subseteq \Sigma$ *known to both players; the protocol then depends on* Σ_A *and works for every input whose first* a *tokens lie in* Σ_A*.*

(c) *If* $w = 1$ *(and raw-token windows consist of the current token only), there is such a protocol of cost at most* $\lceil \log_2 |\Sigma| \rceil + 2\sum_\ell B_\ell$*, whose first term (Bob sends* x_n*) can be dropped when* x_n *is fixed.*

Proof overview. Since no global layer lies at or below σ_0, the states $h_j^{(\sigma_0)}$ depend only on the tokens in $[j - \sigma_0(w-1), j]$. After Alice sends her last $\sigma_0(w-1)$ tokens, each player knows the KV-source states, hence all keys, values and indexer keys, of its own positions. Bob simulates the cone $C_\ell = [n - (L-\ell)(w-1), n]$ of the last position (Figure 2a), which lies in his part. For each global read of a cone position he sends its state (s bits) and Alice answers with her exact partial sums or her local top-k list with the KV-source states of the listed positions (B_ℓ bits); by the exact-sum convention Bob completes the read. Reuse and re-selection layers revisit stored candidates and cost nothing. With $w = 1$ the cone is $\{n\}$, both players can follow $h_n^{(\ell)}$ in lockstep, and each read costs one answer from each side.

Proof of Lemma E.1. By the local receptive field (Section A), since no layer at or below σ_0 has global attention (a global layer ℓ has $\sigma_0 = \sigma(\ell) \le \ell - 1$), the states $h_j^{(\ell)}$ with $\ell \le \sigma_0$ depend only on the tokens in $[j - \ell(w-1), j]$.

Step 0: boundary. Alice sends x_t for $t \in [\max(1, a+1-\sigma_0(w-1)), a]$ (nothing if $w = 1$ or $\sigma_0 = 0$), each in $\lceil \log_2 |\Sigma_A| \rceil$ bits: at most $\sigma_0(w-1)\lceil \log_2 |\Sigma_A| \rceil$ bits. Now Bob knows every token x_t with $t > a - \sigma_0(w-1)$, hence $h_j^{(\sigma_0)}$ for every $j > a$, and Alice knows $h_j^{(\sigma_0)}$ for every $j \le a$. So Alice can compute the key, value and indexer key of every position $j \le a$ in every global layer, and Bob of every $j > a$.

Step 1: what Bob simulates. Let C_ℓ be the cone of (A.1). Since

$$n - (L-\ell)(w-1) \;\ge\; n - L(w-1) \;\ge\; a+1,$$

every C_ℓ lies in Bob's part. Bob computes $h_j^{(\ell)}$ for $j \in C_\ell$ in increasing order of ℓ. For $\ell \le \sigma_0$ these depend only on tokens in $[n - L(w-1), n]$, which are Bob's. For $\ell > \sigma_0$ he needs, for $j \in C_\ell$:

- the states $h_t^{(\ell-1)}$ for $t \in (j-w, j]$, which lie in $C_{\ell-1}$ because $t \ge n - (L-(\ell-1))(w-1)$; and
- if layer ℓ is global, the global part of the heads of position j.

Everything else is token-wise; raw-token windows read tokens in $[j - a_{\text{raw}} + 1, j] \subseteq [n - L(w-1), n]$, and the window part of a joint head is a function of the states in the first item.

Step 2: one global read. Fix a global layer ℓ and $j \in C_\ell$.

- *Dense layer.* Bob sends $h_j^{(\ell-1)}$ (s bits), from which Alice computes all queries of j. Alice returns, per head, the exact numerator and denominator over $\{t \le a\}$ (at most B_{dense} bits); Bob computes the same over $\{a < t \le j\}$ and combines them by consequence (i).
- *Type-(a) layer.* Bob sends $h_j^{(\ell-1)}$. Alice returns her local top-k list over $\{t \le a\}$ with positions, rounded scores and the KV-source states of the listed positions, and for a pool-producing layer also her local top-K_{pool} list (at most $B_{\text{sparse}} + B_{\text{pool}}$ bits); Bob merges

them with his lists by consequence (ii), obtains the selection (and pool) together with the KV-source state of every member, computes the heads, and stores the members with their states.

- *Type-(b) or type-(c) layer.* Nothing is sent. The producing layer $\ell' < \ell$ at the same position satisfies $j \in C_\ell \subseteq C_{\ell'}$, so Bob simulated j at layer ℓ' and stored its pool or selection with the KV-source states of the members; since $\sigma(\ell) = \sigma(\ell') = \sigma_0$, the keys, values and indexer keys that layer ℓ uses are functions of the stored states. For a re-selection Bob takes the top k of the stored pool and stores the result for later reuse layers. Bob computes the heads without communication.

Step 3: count. Since $|C_\ell| = (L-\ell)(w-1)+1$, the reads cost at most

$$\sum_{\ell \text{ global}} \big((L-\ell)(w-1)+1\big)\gamma_\ell \text{ bits.}$$

After layer L, Bob knows $h_n^{(L)}$ ($C_L = \{n\}$) and sends it, or the c_{out} output bits. Every quantity Bob uses equals the one M computes, by induction on ℓ and consequences (i),(ii), and the protocol is deterministic. This proves (a) and (b).

Step 4: the lockstep variant ($w = 1$). Then $C_\ell = \{n\}$ and there is no local attention, so $h_n^{(\ell)}$ is a function of x_n and of the global head outputs at position n in layers up to ℓ. Bob sends x_n (unless it is fixed), so both players know $h_n^{(0)}$. At a global layer ℓ both know $h_n^{(\ell-1)}$ and hence all queries of n. Alice sends her part over $\{t \le a\}$ and Bob his part over $\{a < t \le n\}$, in the formats above, each at most B_ℓ bits; both players then know the head outputs and store the merged lists, so later layers of type (b) or (c) cost nothing. After layer L both know $h_n^{(L)}$. This proves (c). □

Lemma E.2 (Inner product)**.** *Let $N \ge 1$.*

(a) Every protocol computing IP_N on all inputs has cost at least N.

(b) Every protocol of cost c whose output equals $\mathrm{IP}_N(x,y)$ for at least a $(\frac{1}{2}+\varepsilon)$ fraction of uniformly random (x,y) satisfies $c \ge N - 2\log_2(1/(2\varepsilon))$.

(c) For $m \ge 1$, let $x \in \{0,1\}^{2m}$ satisfy $x_{m+1} = \cdots = x_{2m} = 0$ and let $y \in \{0,1\}^{2m}$ have Hamming weight m. Every protocol computing $\mathrm{IP}_{2m}(x,y)$ for all such x and y has cost at least m.

Proof of Lemma E.2. **Setup.** Call a leaf of the protocol tree *reached* if some input ends there. The transcripts of reached leaves are prefix-free strings of length at most c, so there are $T \le 2^c$ of them, and the inputs reaching a leaf form a rectangle $A_t \times B_t$ (a message bit depends on the speaker's input only). Let $G[x,y] = (-1)^{\mathrm{IP}_N(x,y)}$; then

$$GG^\top = 2^N I, \qquad \text{so} \qquad \operatorname{rank} G = 2^N \quad \text{and} \quad \|Gv\|_2 = 2^{N/2}\|v\|_2.$$

Part (a). On a leaf, the output is constant (transcript output), a function of y alone (Bob outputs) or of x alone (Alice outputs). If the protocol is correct, the block $G[A_t, B_t]$ is therefore constant, has constant columns, or has constant rows: its rank is at most 1. Summing the blocks, $2^N \le T \le 2^c$.

Part (b). Without loss of generality the output is a bit and Bob outputs (if Alice outputs, apply the argument to $G^\top$). For a leaf let $s_t \in \{\pm 1\}^{B_t}$ be the sign vector of the output as a function of y. By Cauchy–Schwarz, each $|\mathbf{1}_{A_t}^\top G s_t| \le \|\mathbf{1}_{A_t}\|_2 \, \|Gs_t\|_2 = 2^{N/2}\sqrt{|A_t||B_t|}$, and the rectangles partition the 4^N inputs, so

$$\begin{aligned} 2\varepsilon \;\le\; \Pr[\text{correct}] - \Pr[\text{wrong}] \;&=\; 4^{-N}\sum_t \mathbf{1}_{A_t}^\top G s_t \\ &\le\; 4^{-N}\,2^{N/2}\sum_t \sqrt{|A_t||B_t|} \\ &\le\; 4^{-N}\,2^{N/2}\sqrt{T\textstyle\sum_t |A_t||B_t|} \;=\; \sqrt{T}\,2^{-N/2}. \end{aligned}$$

Hence $c \geq \log_2 T \geq N - 2\log_2(1/(2\varepsilon))$.

Part (c). For $x' \in \{0,1\}^m$ and $y' \in \{0,1\}^m$ set $x = (x', 0^m)$ and $y = (y', 1^{m-|y'|}, 0^{|y'|})$. Then $|y| = m$ and $\mathrm{IP}_{2m}(x, y) = \mathrm{IP}_m(x', y')$, and each party computes its string locally, so a protocol of cost c as in (c) yields a protocol of cost at most c computing IP_m on all inputs; apply (a). □

The bound (a) and the discrepancy bound behind (b) are classical (Chor & Goldreich, 1988; Kushilevitz & Nisan, 1997); we include the proof because the output may be computed by one party, which is the form our simulations produce.

Deployed widths. At deployed widths the bound concerns long joins. Take $H = 32$ query heads of dimension $d = 128$, $p = 16$ (in the format of Definition 2.3) and contexts of at most 2^{26} tokens. Then one dense read has $B_{\mathrm{dense}} = 388{,}032$ bits, so a model without sliding windows whose 24 reading layers share one KV generation has

$$\mathrm{RB}(M) = 9{,}312{,}768 \text{ bits},$$

and by Theorem 4.1(c) it errs on JP_N for every N with

$$2\mathrm{RB}(M) \;<\; N \;\leq\; 2^{25} - 1 \qquad \big(\text{so that } n = 2N + 1 \leq 2^{26}\big),$$

that is, from about $1.9 \cdot 10^7$ bits on. At $p \geq 133$, which represents every BF16 value, $2\mathrm{RB}(M)$ exceeds 2^{25} and this range of N is empty. For the sliding-window form of YOCO, where all windows sit below the shared cache, a sharper count removes the state term and the cone factor (Corollary G.2).

F Constructions

All constructions use the following facts about the arithmetic of Definition 2.3.

- (F1) Integers of absolute value below 2^p are p-bit numbers and are not rounded.
- (F2) $e^0 = 1$ is exact.
- (F3) A score at most $-\tau$ gives a rounded weight in $[0, e^{-\tau} + 2^{-p}]$.
- (F4) An output in $[0, 2)$ is changed by at most 2^{-p} by rounding; above 1 only the relative bound $|\mathrm{round}(x) - x| \leq x2^{-p}$ holds in general.

A *retrieval query* for position π is $\tau(2\pi, -1, -\pi^2)$ against keys $(t, t^2, 1)$: its score

$$\tau\big(2\pi t - t^2 - \pi^2\big) \;=\; -\tau(t - \pi)^2$$

is an integer, 0 at $t = \pi$ and at most $-\tau$ elsewhere. For JP_N we write $A = [1, N]$ and $B = [N + 1, 2N]$ for the blocks of x and of y.

F.1 The retrieval head

Lemma F.1 (Retrieval head). *Let $n \geq 2$, $Q := \lceil \log_2 n \rceil + 1$, and let $\tau = \tau_n$ and $p = p_{\mathrm{ret}}(n)$ be as in (3.1). Let every position $t \leq n$ hold an integer content $0 \leq P(t) \leq n$, and let a head have keys $k_t = (t, t^2, 1)$ and values $v_t = P(t)/2^Q \in [0, 1/2]$. If a position i uses the retrieval query for a target $\pi \leq i$ over the positions $t \leq i$, the rounded output o satisfies*

$$|o - v_\pi| \;\leq\; ne^{-\tau} + (n + 1)2^{-p} \;<\; 2^{-Q-1},$$

so rounding $2^Q o$ returns $P(\pi)$; the same holds coordinate by coordinate for values with several such coordinates. All key and query coordinates (t, t^2, 1, $2\tau\pi$, τ, $\tau\pi^2$) and all scores are integers of absolute value at most $\tau n^2 < 2^p$. Used as indexer scores with $k = 1$, the same scores have the unique maximum 0 at π, and attention over $\{\pi\}$ returns v_π exactly.

Proof. By (F1)–(F3) the weight is 1 at π and lies in $[0, e^{-\tau} + 2^{-p}]$ elsewhere. Let $\delta = n(e^{-\tau} + 2^{-p})$. Then

$$\mathrm{Num} = v_\pi + E_1, \qquad \mathrm{Den} = 1 + E_2, \qquad 0 \leq E_1 \leq \delta/2, \qquad 0 \leq E_2 \leq \delta,$$

so $|\mathrm{Num}/\mathrm{Den} - v_\pi| \leq \delta/2$, and by (F4) the rounded output satisfies the displayed first inequality. By the choice of τ, $ne^{-\tau} \leq 2^{-Q-2}$, and by the choice of p, $(n+1)2^{-p} \leq 2^{-Q-3}$; so $|o - v_\pi| < 2^{-Q-1}$. The scores are integers with $|\tau(t - \pi)^2| \leq \tau n^2 < 2^p$, so they are not rounded, and the maximum 0 is attained only at $t = \pi$. □

F.2 READ DEPTH

Theorem 3.1 (Read depth survives KV sharing)**.** *Let $r \geq 1$ and consider r-FuncComp, unpadded or with $G \geq 0$ blanks between consecutive blocks E_i, with prompt length n and table entries at most n. There is a typed transformer with $r+1$ layers of dense global attention, all of KV source 0, no local attention, $H = 1$, $d = 3$, $D = r + 7$ and precision $p_{\mathrm{ret}}(n)$ that computes r-FuncComp at the last position on every input. It has broadcast depth 1 and read depth $r + 1$, and its keys and values are the same in every layer. The same holds with a fresh top-1 selection of KV source 0 in place of dense attention in every layer.*

Proof. **Layout.** Let $P(t)$ be the integer content of the token at position t (0 at blanks and at the last token). The positions of z_0 and of the entry of z_ℓ at every argument are fixed functions of the parameters (blocks $E_r, \ldots, E_1$ of lengths $N_{r-1}, \ldots, N_0$ separated by G blanks, then z_0, a gap, and ω); the model hard-codes them, as well as a fixed bijection between arguments $(\omega_j, i_j) \in [n_j] \times [N_{j-1}]$ and $[N_j]$ (Chen et al. (2025) use this identification implicitly in their eq. (6); any fixed choice works). Keys and values are those of Lemma F.1; they are functions of $h_t^{(0)}$ and the same in every layer.

Computation. All positions other than the last use zero queries. The last token ω is decoded by the embedding into $\omega_1, \ldots, \omega_{r-1}$. With the retrieval head of Lemma F.1:

- layer 1 targets the position of z_0 and stores i_0;
- layer 2 targets the entry of z_1 at i_0 and stores i_1;
- layer $\ell + 1$ ($2 \leq \ell \leq r$) targets the entry of z_ℓ at $(\omega_{\ell-1}, i_{\ell-1})$ and stores i_ℓ.

By induction the last position stores i_ℓ after layer $\ell + 1$, and outputs i_r after layer $r + 1$. For the sparse variant, the same scores serve as indexer scores with $k = 1$ (Lemma F.1). The width is $D = (r - 1) + 8$ (the components of ω, the current index, position data, the head output and a spare coordinate). □

Proposition F.2 (Reuse across generations can add read depth)**.** *Extend Definition 2.1 by* cross-generation reuse*: a global layer ℓ may take $S_i^{(\ell)} = S_i^{(\ell')}$ for an earlier type-(a) layer ℓ' with $\sigma(\ell') \neq \sigma(\ell)$, its keys and values being functions of $h_j^{(\sigma(\ell))}$; read depth still counts the dense and type-(a) layers, and such a layer has per-read bandwidth 0, like a reuse layer. For every $B \geq 1$ and every padding $G \geq 0$ there is a model in this extension with two layers and no local attention, whose layer 1 selects one position afresh with KV source 0 and whose layer 2 reuses that selection with KV source 1, with $H = 1$, $d = 3$, $D = 8$ and precision $p = 2\lceil \log_2 n \rceil + 2$, that computes 1-FuncComp on every input of the instance I_B of Theorem 3.2 with $\rho = 1$. For each fixed G this family has read depth 1 and meets every condition of the class $\mathcal{F}_4$ of Theorem C.1(c) except that it is not typed. With $G = 0$, the padding that Lemma B.2 and Theorem 3.2 prescribe for its parameters, its per-read bandwidth is $37\lceil \log_2 n_B \rceil + 36$, at most B for every $B \geq 8213$; so the conclusion of Lemma B.2 fails for every B, and that of Theorem 3.2(a) for every $B \geq 8213$, in the extension.*

Proof. **Layout.** Block E_1 lists the table of $z_1 : [m] \to [m]$, the entry for argument a at position a; then come G blanks, the token z_0 at position $\pi_0 = m + G + 1$, G blanks, and the last token at $n = m + 2G + 2$. Let $P(t)$ be the integer content of the token at t (0 at blanks and at the last token), $Q = \lceil \log_2 n \rceil + 1$, and let $h_t^{(0)}$ contain $P(t)$, t, t^2 and 1. With $p = 2\lceil \log_2 n \rceil + 2$, integers of absolute value at most n^2 are p-bit numbers, and so is every $P(t)/2^Q$.

Layer 1 (type (a), one position, KV source 0). Indexer keys are $\kappa_t = (t, t^2, 1)$, and the indexer query of position i is $(2\pi(i), -1, -\pi(i)^2)$ with

$$\pi(\pi_0) = z_0 \ \text{(computed from the token } z_0), \qquad \pi(n) = \pi_0, \qquad \pi(i) = i \ \text{otherwise.}$$

The indexer score of $j \leq i$ is $-(j - \pi(i))^2$, an exact integer whose unique maximum over $j \leq i$ is 0 at $j = \pi(i)$, so $S_i^{(1)} = \{\pi(i)\}$. With attention query 0 the single weight is $e^0 = 1$, and the output is the value $P(\pi(i))/2^Q$ exactly. The token-wise map stores $c(t) = 2^Q$ times the output; in particular $c(\pi_0) = P(z_0) = z_1(z_0) = i_1$.

Layer 2 (cross-generation reuse, KV source 1). $S_n^{(2)} = S_n^{(1)} = \{\pi_0\}$, and the value of position t is $c(t)/2^Q$, a function of $h_t^{(1)}$. The output at n is $i_1/2^Q$ exactly, and the model outputs i_1, the value of 1-FuncComp. On I_B we have $m = K^2 \geq 64^2$, so $n \geq 4$ and $p \leq 4\log_2 n$; the other parameters are fixed. □

F.3 Two generations and pointer chasing

Proposition F.3 (Two generations: ρ reads follow $2\rho - 1$ pointers)**.** *For all $\rho \geq 1$, $1 \leq r \leq 2\rho - 1$, $m \geq 2$ and $G_0 \geq 0$ there is a typed transformer with ρ layers of dense global attention, layer 1 of KV source 0 and layers $2, \ldots, \rho$ of KV source 1 (broadcast depth $\min\{\rho, 2\}$, read depth ρ), no local attention, $H = 1$, $d = 3$, $D = 8$ and the precision of Theorem 3.1, that outputs $\mathsf{APC}_r(m)$ on every input: $\lfloor r/2 \rfloor + 1$ reads suffice for r pointers, against the r reads of Theorem 3.3(b). The model has no sliding windows and read bandwidth $\rho\, B_{\text{dense}} = 4\rho(4p + \lceil \log_2 n \rceil + 4)$. In particular, for $\rho \geq 2$ two generations compute $\mathsf{APC}_{\rho+1}(m)$ at read depth ρ with $2\mathrm{RB} = 8\rho(4p + \lceil \log_2 n \rceil + 4)$, whereas one generation of per-token entries with the windows below it needs $s' + 2\,\mathrm{RB}(M) \geq \mathrm{PC}(m, \rho)$ when the tables are at least $\sigma_0(w - 1)$ blanks apart (Theorem 3.3(a)), which is larger for fixed ρ, $G_0 = O(m)$ and m large.*

Proof. Let $P(t)$ be the integer content of the token at t (0 at blanks and at the query token), $Q = \lceil \log_2 n \rceil + 1$, and let Bob's entry for argument j sit at $b(j) = m + G_0 + j$ and Alice's at j. Let $u(j) = f_A(f_B(j))$; then

$$v_{2t+1} = u(v_{2t-1}), \qquad v_{2t} = f_B(v_{2t-1}).$$

We use the retrieval head of Lemma F.1 with values of up to three coordinates in $[0, 1/2]$.

Layer 1 (KV source 0). Keys $(t, t^2, 1)$ and values $(P(t)/2^Q, 0, 0)$. Position $b(j)$ targets position $f_B(j) = P(b(j)) < b(j)$ and stores $c(b(j)) = u(j)$; the last position targets position 1 and stores the pointer $v_1 = f_A(1)$; every other position stores $c(t) = 0$.

Layers $2, \ldots, \rho$ (KV source 1). Keys $(t, t^2, 1)$ and values $(P(t)/2^Q, c(t)/2^Q, 0)$. The last position holds v_{2t-1} after layer t.

- If r is odd, layer $t + 1$ targets $b(v_{2t-1})$ and takes the second coordinate, $u(v_{2t-1}) = v_{2t+1}$, until v_r is reached after layer $(r + 1)/2 \leq \rho$.
- If r is even, it does so until v_{r-1} is reached after layer $r/2$, and then targets $b(v_{r-1})$ and takes the first coordinate, $f_B(v_{r-1}) = v_r$, after layer $r/2 + 1 \leq \rho$.

Remaining layers use the zero query. The width $D = 8$ holds P, t, t^2, 1, c, the pointer and two output coordinates. □

F.4 Joins

Theorem 4.2 (Two KV generations join with two layers)**.** *For every $N \geq 1$ ($n = 2N + 1$) there is a typed transformer with 2 layers of dense global attention, KV sources $\sigma(1) = 0$ and $\sigma(2) = 1$ (broadcast depth 2, read depth 2), no local attention, $H = 1$, $d = 3$, $D = 8$ and precision $p = 2\lceil \log_2 n \rceil + \lceil \log_2 \tau \rceil + 4$ with $\tau = \lceil \ln(8n) \rceil$, whose output at position n equals $\mathrm{IP}(x, y)$ for every input of JP_N.*

Proof. The embedding stores the token bit χ_t (x_t for $t \leq N$, y_{t-N} for $N < t \leq 2N$, 0 at n), t, t^2 and the indicator of block B. The parameters give $2^{-p} \leq 1/(16n^2\tau) \leq 1/(8n)$ and $2^p > 4N$.

Layer 1 (KV source 0): position $N + i$ fetches x_i. Keys $(t, t^2, 1)$, values χ_t for $t \leq N$ and 0 otherwise. Position $N + i$ uses the retrieval query for i. With $\delta = n(1/(8n) + 2^{-p}) \leq 1/4$, $\mathrm{Num} = x_i + E_1$ and $\mathrm{Den} = 1 + E_2$ with $0 \leq E_1, E_2 \leq \delta$: the ratio is at least $4/5$ if $x_i = 1$ and at most $1/4$ otherwise, so after rounding the output exceeds $1/2$ iff $x_i = 1$. The token-wise map stores $u_i = x_i y_i$.

Layer 2 (KV source 1): the last position sums the products. Keys $([N < t \leq 2N], 0, 0)$ and values u_{t-N} on block B, and 0 elsewhere. The last position uses the query $(\tau', 0, 0)$ with

$\tau' = \lceil \ln(8(N+1)) \rceil$. With $A = \mathrm{round}(e^{\tau'})$, the relative bound gives $A \geq e^{\tau'}(1-2^{-p}) > 4(N+1)$; the other $N+1$ positions have weight 1 and value 0. So $\mathrm{Num} = AS$ with $S = \sum_i u_i$ and $\mathrm{Den} = NA + N + 1$, and

$$N \frac{\mathrm{Num}}{\mathrm{Den}} = S\Big(1 - \frac{N+1}{NA+N+1}\Big) \in \big(S - \tfrac{1}{4}, S\big].$$

The rounded output o' has $|No' - N\mathrm{Num}/\mathrm{Den}| \leq N2^{-p} < 1/4$, so $\mathrm{round}(No') = S$ and the output $S \bmod 2$ equals $\mathrm{IP}(x, y)$. □

Proposition F.4 (One generation joins at linear read cost)**.** *For all $N \geq 1$ and $1 \leq b \leq N$ ($n = 2N+1$) there is a typed transformer with $L = \lceil N/b \rceil$ layers of dense global attention, all of KV source 0 (broadcast depth 1), no local attention, $H = 2$, $d = 3$, $D = 8$ and precision $p = b + 2\lceil \log_2 n \rceil + \lceil \log_2 \tau \rceil + 6$ with $\tau = \lceil \ln n + (b+3) \ln 2 \rceil$, whose output at position n equals $\mathrm{IP}(x, y)$ for every input of JP_N. Its read bandwidth is $\mathrm{RB}(M) = \lceil N/b \rceil \cdot 8(4p + \lceil \log_2 n \rceil + 4)$ and $C(M) = \lceil N/b \rceil (48p + 8\lceil \log_2 n \rceil + 32) + 1$; for $b = \min\{N, \lceil \log_2 n \rceil\}$ both are $O(N + \log n)$.*

Proof. **Chunks.** For $t \leq N$ let $c(t) = \lceil t/b \rceil$ and for $N < t \leq 2N$ let $c(t) = \lceil (t-N)/b \rceil$; chunk ℓ of block A is $J^A_\ell = \{t \leq N : c(t) = \ell\}$, of size $b_\ell \leq b$, and similarly J^B_ℓ.

Heads. Head 1 has keys $(g(t), g(t)^2, 1)$ with $g(t) = c(t)$ on block A and $g(t) = -n$ elsewhere, and values $x_t 2^{((t-1) \bmod b) - b}$ on block A and 0 elsewhere; head 2 is the same for block B. Both are functions of $h^{(0)}_t$. At layer ℓ the last position uses the retrieval query for ℓ in both heads. The score is 0 exactly on the chunk and at most $-\tau$ elsewhere.

Decoding a chunk. Let $V = \sum_{t \in J^A_\ell} v_t$, a multiple of 2^{-b} in $[0, 1)$ whose binary digits are the chunk's bits. With $\delta = n(e^{-\tau} + 2^{-p})$, $\mathrm{Num} = V + E_1$ and $\mathrm{Den} = b_\ell + E_2$ with $0 \leq E_1, E_2 \leq \delta$, so

$$|b_\ell o - V| \leq ne^{-\tau} + n2^{-p} + b2^{-p} \leq 2^{-b-2}$$

by the choice of τ and p. Rounding $b_\ell o$ to a multiple of 2^{-b} returns V, i.e. the bits of the chunk; head 2 returns the chunk of y. The token-wise map updates a stored parity by $\bigoplus_r x_{(\ell-1)b+r} y_{(\ell-1)b+r}$; after $\lceil N/b \rceil$ layers it holds $\mathrm{IP}(x, y)$. Scores and keys are integers of absolute value at most $4\tau n^2 < 2^p$, and values have $b \leq p$ fractional bits.

Cost. With $w = 1$ every layer has

$$B_\ell = B_{\mathrm{dense}} = 8\big(4p + \lceil \log_2 n \rceil + 4\big) \qquad \text{and} \qquad s = 16p,$$

which gives the stated $\mathrm{RB}(M)$ and $C(M)$. □

F.5 Layered path parity

Proposition F.5 (Layered path parity, first level)**.** *Let $m \geq 2$ and $m_1 \geq 1$, and let $\mathsf{LPP}_2(m; m_1)$ be the instances with $V_1 = \{1, \ldots, 2m\}$, $V_2 = \{1, 2\}$, fibers of f_1 of size m_1 and fibers of f_2 of size m.*

(a) If a typed transformer M whose global layers share one KV source, with $2m \geq L(w-1)$, computes LPP_2 on all these instances, then

$$\sigma_0(w-1)\lceil \log_2 |\Sigma| \rceil + \sum_{\ell \text{ global}} \big((L-\ell)(w-1)+1\big)\gamma_\ell + 1 \geq m,$$

with γ_ℓ as in Lemma E.1, and if $w = 1$ then $2\,\mathrm{RB}(M) \geq m$.

(b) There is a typed transformer with two dense global layers of KV sources 0 and 1, no local attention, $H = 1$, $d = 3$, $D = 8$ and $p = 2\lceil \log_2 n \rceil + \lceil \log_2 \tau \rceil + 4$, $\tau = \lceil \ln(8n) \rceil$, that computes LPP_2 on all these instances.

Proof. **Part (a): a reduction from the inner product.** The prompt lists block 0, whose tokens are the triples $(u, f_1(u), b(u))$; block 1, whose tokens are the pairs $(v, f_2(v))$; and the target e. Given $x \in \{0,1\}^{2m}$ with $x_{m+1} = \cdots = x_{2m} = 0$, Alice sets $f_1(u) = \lceil u/m_1 \rceil$, $b(u) = x_j$ if

$u = (j-1)m_1 + 1$ and $b(u) = 0$ otherwise, so the parity of b over $f_1^{-1}(j)$ is x_j. Given y of weight m, Bob sets $f_2(j) = 1$ if $y_j = 1$ and $f_2(j) = 2$ otherwise (both fibers have size m) and $e = 1$. Then

$$\mathsf{LPP}_2 \;=\; \bigoplus_{j:\, y_j = 1} x_j \;=\; \mathrm{IP}_{2m}(x, y).$$

The split after block 0 has $n - a = 2m + 1 \geq L(w-1) + 1$, so Lemma E.1 (with one output bit, or its lockstep variant with the fixed last token) gives a protocol of the stated cost for the problem of Lemma E.2(c), whose cost is at least m.

Part (b): two generations.

- *Layer* 1 *(KV source* 0*).* Keys are $(f_1(u), f_1(u)^2, 1)$ on block 0 and $(-n, n^2, 1)$ elsewhere, and values are $b(u)$ on block 0. The position of vertex v uses the retrieval query for v. With $c = \sum_{u \in f_1^{-1}(v)} b(u)$,
$$|m_1 o - c| \;\leq\; \delta + m_1 2^{-p} \;<\; \tfrac{1}{2},$$
and the token-wise map stores $s_1(v) = \mathrm{round}(m_1 o) \bmod 2$.
- *Layer* 2 *(KV source* 1*).* It does the same over block 1, with keys $(f_2(v), f_2(v)^2, 1)$, values $s_1(v)$ and the retrieval query for e at the last position, and outputs the parity of the returned count.

□

Proposition F.6 (Reads replace one broadcast at linear cost)**.** *Let $\beta \geq 1$ and consider $\mathsf{LPP}_{\beta+1}$ with canonical vertex labels $V_t = \{1, \ldots, |V_t|\}$, $|V_{\beta+1}| = 2$, fibers of $f_{\beta+1}$ of size m (so $|V_\beta| = 2m$), and fibers of every f_t, $t \leq \beta$, of a fixed size m_t known to the model. There is a typed transformer with broadcast depth β, $L = (\beta - 1) + 2m$ dense global layers, no local attention, $H = 2$, $d = 3$, $D = 8$ and precision $p_{\mathrm{ret}}(n)$ of (3.1) that computes $\mathsf{LPP}_{\beta+1}$ on all such inputs. For $\beta = 1$ its read bandwidth is $\mathrm{RB}(M) = O(m \log n)$, matching Proposition F.5(a) up to the logarithmic factor.*

Proof. **Fiber aggregation.** If every position of a block Q stores a label $g(u)$ and a bit $z(u)$ and every other position stores the label $-n$, a head with keys $(g, g^2, 1)$, values z on Q, and the retrieval query for c returns o with

$$\Big|\mu o - \sum_{u \in Q:\, g(u) = c} z(u)\Big| \;<\; \frac{1}{2}$$

whenever the number $\mu \geq 1$ of matches is a fixed parameter, as in Proposition F.5(b).

Levels. Layer $t \leq \beta - 1$ (KV source $t - 1$) lets every position of block t, holding $v \in V_t$, aggregate s_{t-1} ($s_0 = b$) over $f_t^{-1}(v)$ and store $s_t(v)$.

Retrieval. As in Lemma F.1: keys $(t', t'^2, 1)$ on block β indexed by the vertex t' and $(-n, n^2, 1)$ elsewhere, values $f_{\beta+1}(t')/2^Q$ on block β and 0 elsewhere, and the retrieval query for v with parameter τ_n return $f_{\beta+1}(v)$; the scores are 0 at the position of v and at most $-\tau_n$ elsewhere.

Catch-up. Since $V_\beta = \{1, \ldots, 2m\}$, the layers can enumerate it. Layer $\beta - 1 + v$, $v = 1, \ldots, 2m$, has KV source $\beta - 1$; at the last position, head 1 retrieves $f_{\beta+1}(v)$ and head 2 aggregates $s_{\beta-1}$ over $f_\beta^{-1}(v)$, giving $s_\beta(v)$. The token-wise map adds $s_\beta(v)$ to a stored parity if $f_{\beta+1}(v) = e$. After these $2m$ layers the parity is $\mathsf{LPP}_{\beta+1}$. The KV sources are $\{0, \ldots, \beta - 1\}$. All scores are integers and all coordinates are integers or dyadic rationals with denominator at most 2^Q, of absolute value at most $\tau_n(2n)^2 < 2^p$. For $\beta = 1$,

$$\mathrm{RB}(M) \;=\; 2m \cdot H(d+1)\big(4p + \lceil \log_2 n \rceil + 4\big) \;=\; O(m \log n).$$ □

G Architecture accounting

This appendix proves the ledger statements of Section 5. Each of the first three subsections describes a model as its report specifies it and then states a corollary; the last one treats cross-layer attention.

G.1 DEEPSEEK-V4.1-FLASH

The model. We model the text backbone of the post-trained DeepSeek-V4.1-Flash as specified in its technical report:

- 40 layers, with sliding-window attention of window $w = 128$ in every layer, in one softmax with the selected global entries;
- compressed sparse attention with top-$k = 512$ in layers 3–40;
- encoder groups 3–8, 9–14, 15–20, each a Full layer followed by five Reuse layers, with entries compressing 2 positions;
- decoder groups of four layers from layer 21: Full, then Reindex in layers $25, 29, 33, 37$, the rest Reuse, with entries of one position; the decoder's keys and values come from the encoder output, and the decoder Full layer builds a candidate pool of at most 2048 blocks of 8 entries within which the Reindex layers select;
- Engram lookups that read the current and at most three previous tokens; mHC mixing, the mixture-of-experts layers and the normalizations are token-wise.

Modelling assumptions.

(A1) We model a single forward pass over the prompt, a cold prefill: prefix caching, under which the encoder's bounded replay makes outputs depend on the cache-hit position, is not modelled.

(A2) Mixture-of-experts routing is per token (no capacity-based dropping).

(A3) Top-k ties and pool-block ties are broken by the larger position, as in Definition 2.3.

(A4) Pool blocks are aligned blocks of 8 consecutive entries, and only complete blocks are selectable.

(A5) Mixed precision is modelled by one precision $p \geq 2$ (an exact representation of BF16 would need $p \geq 133$, which only increases $(1 + 2G)B_{\text{read}}$ and the instance length).

(A6) The decoder's bounded replay of its sliding windows only truncates windows at the replay boundary, which keys with saturated scores (rounded weight 0) represent.

(A7) Tokens and the embedding are idealized as in Chen et al. (2025), i.e. the tokenizer is not modelled.

Layer map. Table 2 lists the layers of the backbone with their names in the report and, as derived in the proof of Corollary G.1, their KV sources and the types of their global attention in the sense of Definition 2.1.

Table 2: Layer map of the DeepSeek-V4.1-Flash backbone. The last two columns are derived in the proof of Corollary G.1; (a), (b) and (c) are the types of Definition 2.1.

Layers	Name in the report	KV source	Global attention
1–2	–	–	none (sliding window only)
3, 9, 15	Full (encoder)	2, 8, 14	(a) fresh top-512 selection
4–8, 10–14, 16–20	Reuse (encoder)	2, 8, 14	(c) reuse of the group's selection
21	Full (decoder)	20	(a) fresh selection and candidate pool
25, 29, 33, 37	Reindex	20	(b) re-selection within the pool of layer 21
22–24, 26–28, 30–32, 34–36, 38–40	Reuse (decoder)	20	(c) reuse

Corollary G.1 (DeepSeek-V4.1-Flash)**.** *Under (A1)–(A7), the backbone described above is a typed KV-sharing transformer with features (d)–(g) whose layers are as in Table 2. Moreover:*

(i) Depths. *It has* 38 *global layers and* $c_{\max} = 8$. *Its KV sources are* $\{2, 8, 14, 20\}$, *so its broadcast depth is* 4; *its fresh-read layers are* $\{3, 9, 15, 21\}$, *so its read depth is* 4; *the other* 34 *global layers are four pool-restricted re-selections and* 30 *reuse layers; its local depth is* 40.

(ii) Lower bound. *For every choice of weights and every precision $p \le c_p \log_2 n$ (any constant c_p; in particular any fixed precision), there are padded 4-FuncComp instances I_B on which its single forward pass errs, although 38 of its layers attend globally and 8 select entries (Theorem 3.2(b)).*

(iii) Upper bound. *A model of broadcast depth 1, read depth 5 and precision at most $4 \log_2 n_B$ solves every such instance, for all large B (Theorem 3.1).*

(iv) Pre-trained checkpoint. *In the pre-trained checkpoint the Reindex layers score the whole context, and the same accounting gives read depth 8.*

(v) Size of the hard instances. *With $D \ge 4 \cdot 5120$ and $p = 16$, $(1 + 2G)B_{\mathrm{read}}$ exceeds 10^{14} bits, so Theorem 3.2(a) applies only for $B > 10^{14}$; then the instances I_B have more than 10^{100000} tokens.*

Proof. **Layer types.** The Full layers 3, 9, 15 (first layers of the encoder groups, after two window-only layers) and 21 (first decoder layer) compute fresh keys and values from their input states: KV sources 2, 8, 14, and, for layer 21, the encoder output $h^{(20)}$. Reuse layers reuse the most recent keys, values and top-k indices, so they have the KV source of their group's Full layer and are of type (c). The decoder has a single Full layer, so every decoder global layer has KV source 20. The Reindex layers $25, 29, 33, 37$ reuse the decoder's keys, values and indexer keys and select the top 512 within the candidate pool of layer 21 at the same position: type (b). The encoder has no Reindex layers. Hence the KV sources are $\{2, 8, 14, 20\}$ ($\beta = 4$), the fresh-read layers are $\{3, 9, 15, 21\}$ ($\rho = 4$), there are four pool-restricted re-selections, and $38 - 8 = 30$ reuse layers.

Features. Encoder entries compress 2 positions and decoder entries 1; decoder pool blocks consist of 8 entries; so $c_{\max} = 8$. Indexer keys are projections of the entries' keys, hence functions of the entries' source states (feature (d)), and the candidate pool is a block-max selection of 2048 blocks (feature (e)). Sliding-window attention with $w = 128$ occurs in all 40 layers ($\lambda = 40$), and in the compressed-sparse layers the window's keys and values and the selected entries enter one softmax (feature (f)). Engram lookups depend on the current token and at most three previous tokens and are gated by the current state; we fold each Engram module into the last token-wise map of the preceding layer if it precedes attention in its block, and into the same layer's post-attention map otherwise, so every KV source remains a hidden state, the KV sources are unchanged, and the module reads a raw-token window of size $4 \le 128$ (feature (g)). Expert routing, the mixing of the four residual streams of one token, and normalizations are token-wise.

Pre-trained checkpoint. The candidate pool is introduced in post-training, so in the pre-trained checkpoint the Reindex layers score all causally visible entries and are of type (a): the fresh-read layers are $\{3, 9, 15, 21, 25, 29, 33, 37\}$.

Consequence. The numeric parameters are fixed: $L = 40$, $w = 128$, $c_{\max} = 8$, $k = 512$, $K_{\mathrm{blk}} = 2048$, $c' = 8$ and the widths H, d, D. The precision is at most $c_p \log_2 n$, so Theorem 3.2(b) applies with $\rho = 4$, and part (c) gives the model of broadcast depth 1, read depth 5 and precision at most $4 \log_2 n_B$.

Magnitudes. The supplementary script `ledger_numbers.py` computes the following numbers. First, $G = 40 \cdot 127 + 8 - 1 = 5087$ and $1 + 2G = 10175$. With $D \ge 4 \cdot 5120$ (four residual streams of width 5120) and $p = 16$, $s = 2Dp \ge 655360$. The decoder Full layer is pool-producing, so

$$B_{\mathrm{read}} \;\ge\; 512s + 2048 \cdot 8s \;>\; 1.07 \cdot 10^{10} \qquad \text{and} \qquad (1 + 2G)\, B_{\mathrm{read}} \;>\; 10^{14}.$$

Hence Theorem 3.2(a) needs $B > 10^{14}$, so $K = (4B)^8\, 8^{32} > 2^{484}$, and the instance contains the block of z_4, with $N_3 = K^{878}$ tokens:

$$n_B \;>\; 2^{424000} \;>\; 10^{100000}. \qquad \square$$

G.2 YOCO WITH A SLIDING-WINDOW SELF-DECODER

The model. A *YOCO model with a sliding-window self-decoder*, in the typed form with arbitrary, possibly position-dependent maps, has

- $L \ge 2$ layers, L even;

- a *self-decoder*: layers $\ell \le L/2$ have local attention of window $w \ge 1$ and token-wise maps only;
- a single global cache: the keys and values at position j are functions of $h_j^{(L/2)}$;
- a *cross-decoder*: every layer $\ell > L/2$ has dense causal attention over this cache with $H \ge 1$ query heads of dimension $d \ge 1$, and no other attention.

Corollary G.2 (YOCO with a sliding-window self-decoder)**.** *Let M be a YOCO model with a sliding-window self-decoder.*

(i) Depths. *M has broadcast depth 1, read depth $L/2$, and local depth $L/2$ if $w \ge 2$.*

(ii) Joins. *If M computes JP_N on every input, then*

$$\tfrac{L}{2}(w-1) + L\,B_{\text{dense}} \;\ge\; N;$$

if it does so on a $(\frac{1}{2}+\varepsilon)$ fraction of uniformly random inputs, then

$$\tfrac{L}{2}(w-1) + L\,B_{\text{dense}} \;\ge\; N - 2\log_2\big(1/(2\varepsilon)\big).$$

In particular, with L, w, H, d, p at most $\mathrm{polylog}(N)$, M fails JP_N for all large N.

(iii) Upper bounds. *A standard transformer with two layers, one head of dimension 3 and $O(\log n)$-bit precision computes JP_N (Theorem 4.2); and for every $r \ge 1$ a YOCO model of this form with $L = 2(r+1)$ layers solves r-FuncComp with $H = 1$, $d = 3$, $D = r+7$ and $O(\log n)$-bit precision, reading a single cache.*

Proof. Part (i). Every cross-decoder layer reads keys and values that are functions of $h_j^{(L/2)}$, and the self-decoder has no global attention, so the only KV source is $L/2$; the $L/2$ cross-decoder layers are dense.

Part (ii). Split after x ($a = N$) and let $\sigma_0 = L/2$. Positions are public, so position-dependent maps are computable by both players. Alice sends her bits x_t for $t \in [\max(1, a+1-\sigma_0(w-1)), a]$, at most $\frac{L}{2}(w-1)$ bits. By the local receptive field, Alice then knows $h_j^{(\sigma_0)}$ for all $j \le a$ and Bob for all $j > a$, in particular $h_n^{(\sigma_0)}$. For $\ell = \sigma_0 + 1, \dots, L$, Bob knows $h_n^{(\ell-1)}$ and sends the H queries of position n (Hd numbers, $2pHd \le B_{\text{dense}}$ bits); Alice returns her exact partial sums over $\{t \le a\}$ (B_{dense} bits); Bob adds his own and obtains the head outputs by consequence (i), then $h_n^{(\ell)}$. After layer L, Bob outputs. The cost is at most

$$\tfrac{L}{2}(w-1) \;+\; \tfrac{L}{2}\cdot\big(2pHd + B_{\text{dense}}\big) \;\le\; \tfrac{L}{2}(w-1) + L\,B_{\text{dense}},$$

and Lemma E.2(a),(b) give the bounds. With L, w, H, d, p at most $\mathrm{polylog}(N)$ the left-hand side is $\mathrm{polylog}(N)$.

Part (iii). The first part is Theorem 4.2. For the second, prepend to the model of Theorem 3.1 (for r-FuncComp) $r+1$ layers whose local attention has zero output and whose token-wise maps are the identity, so that $h^{(r+1)} = h^{(0)}$; the shifted layers read the same keys and values as before, now as functions of $h^{(r+1)}$, and these are the same in every layer. □

The read side. Theorem 3.3 specializes to YOCO as follows.

Corollary G.3 (Pointer chasing in YOCO)**.** *Let M be a YOCO model with a sliding-window self-decoder whose token-wise maps read only the current position's state, and let $s' = 2Dp$ if $w \ge 2$ and $s' = 0$ if $w = 1$.*

(a) Lower bound. *If $G_0 \ge \frac{L}{2}(w-1)$ and M outputs $\mathsf{APC}_{L/2+1}(m)$ on every input, then*

$$s' + L\,B_{\text{dense}} \;\ge\; \mathrm{PC}(m, L/2) \;\ge\; \frac{m}{500(L+2)} - \frac{L+2}{2}\log_2 m - 1.$$

(b) Upper bound. *For all even $L \ge 2$, $w \ge 1$, $m \ge 2$ and $G_0 \ge 0$, some YOCO model of this form with window w, $H = 1$, $d = 3$, $D = 8$ and precision $p_{\text{ret}}(n)$ outputs $\mathsf{APC}_{L/2}(m)$ on every input.*

Proof. **Part (a).** By Corollary G.2(i) and the form of the model, M satisfies the hypotheses of Theorem 3.3(a) with $\sigma_0 = L/2$, read depth $L/2$ and $\mathrm{RB}(M) = \frac{L}{2} B_{\mathrm{dense}}$: above $L/2$ it has no local attention, and its token-wise maps read only the current state. So

$$s' + L\, B_{\mathrm{dense}} \;=\; s' + 2\,\mathrm{RB}(M) \;\geq\; \mathrm{PC}(m, L/2),$$

and Lemma 3.4 with $\rho = L/2$ gives the explicit bound, since $1000(L/2+1) = 500(L+2)$.

Part (b). Prepend to the model of Theorem 3.3(b) with $r = L/2$ a stack of $L/2$ layers whose local attention (of window w) has zero output and whose token-wise maps are the identity, as in the proof of Corollary G.2(iii) above. □

Thus such a model follows $L/2$ pointers between two tables at logarithmic size, but following $L/2+1$ pointers between tables of m entries that are $\frac{L}{2}(w-1)$ blanks apart needs

$$L\, B_{\mathrm{dense}} \;\geq\; \frac{m}{500(L+2)} - O(L \log m + Dp).$$

On both tasks, with the blocks or tables beyond the windows' reach, the self-decoder enters only through boundary terms: $\frac{L}{2}(w-1)$ bits for the join and one hidden state for pointer chasing.

Deployed widths. At the width of the YOCO baseline of Sun et al. (2026) (hidden width 2560, 20 heads of dimension 128, $L = 32$, $w = 512$):

- with the tables $\frac{L}{2}(w-1)$ blanks apart, Corollary G.3(a) rules the model out on tables of at least $1.52 \cdot 10^{11}$ entries at $p = 16$ (in the format of Definition 2.3), and of at least $8.22 \cdot 10^{11}$ entries at $p = 133$, which represents every BF16 value;
- Corollary G.2(ii) rules out JP_N at $p = 16$ for every $N \geq 7{,}603{,}697$, and at $p = 133$ for every $N \geq 4.65 \cdot 10^7$. Since B_{dense} grows only with $\lceil \log_2 n \rceil$, the condition $\frac{L}{2}(w-1) + L\, B_{\mathrm{dense}} < N$, once it holds, holds for every larger N.

G.3 YOIO

The model. We model the cross-layer sparse attention model of Sun et al. (2026) as specified there, with the self-decoder and cross-decoder blocks of Sun et al. (2024):

- 32 layers; layers 1–16 have sliding-window attention with window 512 and token-wise maps only;
- the single KV cache is computed from $h^{(16)}$;
- a single-head indexer computes index queries and keys from $h^{(16)}$ and selects, at every position, the top $k = 2048$ positions at or before it, once;
- every layer 17–32 attends with its own query to the selected entries and has no other attention.

Corollary G.4 (YOIO: sixteen global layers, read depth one)**.** *Under our tie rule, one precision p and idealized tokens, the model above is a typed KV-sharing transformer with 16 global layers, broadcast depth 1, read depth 1 (layer 17 selects afresh, layers 18–32 reuse its selection) and local depth 16, with no local mixing above its KV source. Consequently, for every $m \geq 2$ and $G_0 \geq 16 \cdot 512$, if it outputs $\mathsf{APC}_2(m)$ on every input, then*

$$s + 2B_{\mathrm{sparse}} \;\geq\; m \log_2 m - 1, \qquad \textit{where}\;\; B_{\mathrm{sparse}} = 2048\big(\lceil \log_2 n \rceil + 2p + s\big) \;\textit{and}\; s = 2Dp.$$

Proof. The layer types are read off the description; Theorem 3.3(a) and Lemma 3.4 with $\rho = 1$ give the bound, since the reuse layers have bandwidth 0 and the largest window is at least 2. □

So, whatever its weights, following two pointers requires its single selection to carry about $\frac{1}{2} m \log_2 m$ bits: its sixteen global layers ask one question. With the width $D = 2560$ of Sun et al. (2026), $p = 16$ and the tables $16 \cdot 512$ blanks apart, the bound rules out tables of at least $1.42 \cdot 10^7$ entries, and at $p = 133$ tables of at least $1.05 \cdot 10^8$ entries.

G.4 Cross-layer attention

With sharing factor two, cross-layer attention computes keys and values in one layer of each consecutive pair and reuses them in the other (Brandon et al., 2024); every layer still issues fresh dense queries. It therefore has broadcast depth $L/2$ and read depth L: sharing halves the number of KV generations and leaves the read depth, which Theorem 3.2 measures, untouched.

H Exact-arithmetic checks

All constructions of Section F except that of Theorem 3.1, whose retrieval head is the one run for Theorem 3.3(b), were also executed in exact arithmetic under Definition 2.3: p-bit rounding with saturation, exact numerators and denominators as fractions, and exponentials computed in double precision before rounding to the p bits of the construction (the largest p used is 28). The supplementary scripts (commands in `experiments/README.md`) check the following.

- `check_constructions.py`: Theorem 4.2 on all inputs for $N \le 5$; Proposition F.4 on all inputs for five (N, b) configurations with $N \le 6$; Proposition F.5(b) on random instances; and Proposition F.6 for $\beta = 2$ on 160 random instances.
- `check_pointer.py`: Theorem 3.3(b) in both variants, on all inputs for $r \le 4$ and the four pairs $(m, G_0) = (2,0), (2,3), (3,0), (3,2)$ (5960 inputs each); Proposition F.3 on all inputs with the same (m, G_0), $\rho \le 3$ and $r \le 2\rho - 1$ (13410 inputs); and Proposition F.2 on all inputs with $(m, G) \in \{(2,0), (3,1), (4,3)\}$ (1113 inputs).

There were no errors. These checks guard against constant-level slips; the proofs above do not depend on them.

I Further related work and open problems

Communication lower bounds for transformers. Chen et al. (2025) introduced the autoregressive communication model and proved unconditional lower bounds for multi-layer transformers. Theorem 3.1 follows their upper-bound construction (Chen et al., 2025, proof of Corollary 1.2), in which every earlier token exposes a key and a value determined by its own token, and observes that it reads a single cache. Their simulation already covers dense layers whose keys and values come from any earlier layer; Lemma B.2 adds re-selection, reuse and sliding-window layers, so that in typed models their hierarchy counts fresh reads rather than layers, and we use their Theorem 4.1 as a black box. Theorems 3.3 and 4.1 extend the one-layer communication arguments of Sanford et al. (2023) and Peng et al. (2024) to any number of layers that share one KV generation of per-token keys and values, and Theorem 3.3 turns the round hierarchy for pointer chasing (Nisan & Wigderson, 1993) into a read hierarchy.

Hybrid models and cache size. Ye et al. (2026) prove that $L-1$ full-attention layers, each followed by 2^{3L^2} linear-attention layers, cannot perform L-step composition when the product of the number of heads, the head dimension and the precision is at most $n^{2^{-4L-2}}$, whereas $L+1$ full layers suffice; their cheap layers are linear-attention layers and ours reuse or re-select within a KV generation, so the two results concern incomparable classes. Wang (2026) studies depth–cache-size trade-offs for pointer chasing when every layer keeps a subset of positions in its cache; there every layer writes its own cache, whereas our measures separate the number of caches written from the number of reads. Wang names as open the translation of unconditional pointer-chasing lower bounds (Nisan & Wigderson, 1993; Yehudayoff, 2020) into depth lower bounds; Theorem 3.3 carries it out for models whose global layers read one KV generation with the windows below it, a class different from Wang's.

Depth separations and sparse attention. Depth separations for transformers have also been obtained through circuit complexity, simulations of massively parallel computation and chain of thought (Merrill & Sabharwal, 2023; 2024; Li et al., 2024; Sanford et al., 2024; Strobl et al., 2024); those results count layers that read and write. In particular, Sanford et al. (2024) show that multi-hop induction heads need and admit depth logarithmic in the number of hops in standard transformers

(the lower bound assumes a conjecture on massively parallel computation); within one KV generation with the windows below it, Theorem 3.3 and Lemma 3.4 show unconditionally that alternating pointer chasing needs one fresh read per hop unless the reads, together with one hidden state, carry $\Omega(m)$ bits, for tables of m entries beyond the windows' reach. For sparse attention, Zaheer et al. (2020) show, assuming the Orthogonal Vectors conjecture, that a task that one layer of full attention solves needs $\tilde{\Omega}(n^{1-o(1)})$ layers of any sparse attention with $\tilde{O}(n)$ edges; our bounds are unconditional and count which layers read afresh.

Memory rather than depth. Lower bounds on the *size* of KV caches for exact or approximate attention (Haris & Onak, 2025; Kochetkova et al., 2025; Chen et al., 2026; Liberty et al., 2026) measure memory, not depth, and are orthogonal to our measures.

Index reuse in practice. IndexCache (Bai et al., 2026) and, in its decoding phase, TidalDecode (Yang et al., 2025) let a layer attend to its *own* fresh keys and values at the positions that an earlier layer selected (in TidalDecode the selecting layers are dense reads that also emit a selection). Each of their layers writes its own KV generation, so this is the cross-generation reuse of Proposition F.2, and the number of their fresh selections does not play the role of read depth: such reuse can add read depth. HySparse (Gao et al., 2026) reuses both the selection and the keys and values of the preceding full-attention layer, which is reuse within a generation, but its selections are emitted by dense reads, a combination that Definition 2.1 does not include. Layer-condensed KV caches (Wu & Tu, 2024) and feedback memory (Fan et al., 2020) let a layer read keys and values computed at or above it for earlier tokens; such caches are outside our model, in which every KV source lies below the layers that read it.

Open problems. Three questions remain open.

- *The broadcast hierarchy.* Conjecture 4.3 asks whether every level of aggregation needs its own generation. Its first level is Proposition F.5; a direct cut fails at the second level because first-generation reads can prepare hints that the second generation publishes, so a proof must show that the hints are nearly uncorrelated with the level-two parities rather than that they are short.
- *A read hierarchy for many generations at moderate length.* Within one generation of per-token keys with the windows below it, read depth ρ fails $(\rho+1)$-step pointer chasing at read bandwidth $o(m)$ and hidden states of $o(m)$ bits on instances of linear length whose tables lie farther apart than the windows' reach (Theorem 3.3 and Lemma 3.4); across many generations the bound is Theorem 3.2, whose instance length is polynomial in the bandwidth with an exponent exponential in ρ. With two generations the $(\rho+1)$-step chase no longer separates read depth ρ from $\rho+1$, since two generations follow it with $\lfloor(\rho+1)/2\rfloor+1 \le \rho$ reads for $\rho \ge 2$ (Proposition F.3); once positions can publish, they can compose parts of the input before the last position reads it, and the two-party round argument no longer applies. A read hierarchy on instances of polynomial length, with an exponent independent of ρ and moderate constants, would turn the read depth of deployed models (Table 1) into a statement about realistic context lengths.
- *Aggregation by sparse reads.* A dense read sums a whole block, but a sparse read aggregates at most k entries. Whether computing the parity of N bits needs broadcast depth about $\log_k N$ in models whose global layers are all sparse with sparsity k and that have no sliding windows, unless the read bandwidth is polynomial in N, is a question about aggregation trees; a single two-party cut sees only one level of such a tree.