MIND THE MASK: A SUCCINCTNESS HIERARCHY FOR ATTENTION AND RECURRENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

How large must a recurrent model become to reproduce an attention model's decisions? We study exact replacement on every finite nonempty sequence for unique hard-attention transformers without positional encodings and with exact rational evaluation. The recurrent target has fixed finite state and explicitly represented Boolean update and readout circuits. We show that the replacement cost depends sharply on the attention masks. For fully unmasked attention, we prove an exact characterization by the first- and last-occurrence orders of input symbols, with polynomial conversions to profile predicate circuits and polynomial-size recurrent replacement. For mixed masks, we construct dictionary recognizers over a four-symbol alphabet, of description size polynomial in n, for which every exact recurrent replacement requires at least 2^{2^n} persistent bits. One unmasked retrieval between causal stages makes dictionary representations depend on the later query; causal operations then compare complete blocks. Together with inherited bounds, these results give polynomial, single-exponential, and double-exponential replacement regimes for unmasked, causal, and mixed attention. A direct causal compiler further quantifies the role of retrieval depth, and classical information bounds show that, under a specified dense dictionary distribution, any fixed average error below one half still requires a constant fraction of these bits.

1 INTRODUCTION

A recurrent recognizer must retain in its state whatever information from the prefix a later query may need. Attention provides a different interface: depending on the masks, earlier positions can retrieve information from later ones before the final decision is made. When both models recognize a given language, how much larger must the recurrent implementation be? We study the cost of replacing an attention model with a single fixed recurrent recognizer that preserves its accept/reject decisions on every finite nonempty sequence.

For unique hard-attention transformers (UHATs), this cost depends sharply on the masks: polynomial in the transformer's description length for fully unmasked attention, singly exponential for causal attention, and doubly exponential for mixed masks. These worst-case regimes within the three mask classes form the succinctness hierarchy of the title. Without positional encodings, the decision of a fully unmasked UHAT of any depth and width depends only on the first- and last-occurrence orders of the input symbols, and it admits polynomial-size recurrent replacement. In contrast, a polynomial-size dictionary recognizer with one unmasked retrieval between causal stages requires doubly exponentially many recurrent state bits.

To illustrate this cost, consider a dictionary lookup: deciding whether a query matches any previously presented record. A recurrent reader must summarize the dictionary before seeing the query, whereas attention can use the later query to compare records and aggregate the results. Our construction does not merely search the past dictionary from the final query. One unmasked retrieval makes the representation of every dictionary position depend on the later query, and subsequent causal operations aggregate these representations block by block, while a format guard rejects every malformed input. On the same dictionary languages, every causal UHAT recognizer has size exponential in the address length (Corollary 4.5); the choice of masks thus changes source succinctness on a single task.

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.

Table 1: Worst-case cost of exact recurrent replacement within each source class, for source description size s. Lower bounds count persistent bits on witness families; upper bounds bound recurrent description size for every source in the indicated class.

Source	Lower: $b(R)$	Upper: $\lvert R\rvert$	Status
UHAT_0	—*	$\mathrm{poly}(s)$	new characterization and compiler (Thm. 4.1)
UHAT_C	$2^{s^{\Omega(1)}}$	$2^{\mathrm{poly}(s)}$	inherited (Prop. 4.2); direct compiler (Thm. 4.3)
UHAT_M	$2^{2^{s^{\Omega(1)}}}$	$2^{2^{\mathrm{poly}(s)}}$	new lower bound (Thm. 3.2); inherited upper bound

*Because of the polynomial upper bound, no source family forces superpolynomially many bits; Proposition D.3 gives polynomial-size sources that require m persistent bits.

1.1 Related work

Bergsträßer et al. (2026) study succinctness: how compactly a model can describe a language. They show that UHATs can be exponentially more succinct than recurrent networks and linear temporal logic, and doubly exponentially more succinct than finite automata. Their causal construction already gives the single-exponential recurrent lower bound. Their logical translation also implies single-exponential causal and double-exponential general recurrent upper bounds through standard conversions (Appendix C). Our mixed-mask construction closes this recurrent gap in tower height.

Yang et al. (2024) characterize strict causal hard attention with rightmost tie-breaking and no positional encodings by linear temporal logic and the star-free languages. Their activation-count argument also supplies the recurrence used in our depth analysis. Related prefix summaries under fixed-precision, left-to-right floating-point semantics appear in Nowak et al. (2026, Section 3, Proposition 9). In a finite-precision setting, restricting all operations to leftmost tie-breaking strictly weakens masked hard attention (Jerad et al., 2025); our model allows either rule in each operation. For unmasked attention, the relevant equivalence identifies words with the same first- and last-occurrence orders; this quotient is classical (Kimura, 1958; Polák, 2006).

Recurrent state-space models (SSMs), including selective architectures such as Mamba, also summarize the prefix in a fixed-size state (Gu & Dao, 2023); copying and associative recall expose the consequences (Jelassi et al., 2024; Arora et al., 2024). Prior work establishes lower bounds on the amount of memory recurrent models need to solve tasks such as copying, indexing, and associative recall (Sanford et al., 2023; Bhattamishra et al., 2024; Jelassi et al., 2024), and circuit-complexity and formal-language analyses compare transformers with state-space models at the expressivity level (Merrill et al., 2024; Sarrof et al., 2024). In our dictionary task, the recurrent state retained before the query acts as a one-way communication message (Kremer et al., 1999). However, the memory lower bound in input length alone does not establish a succinctness advantage: this also requires a compact attention model that recognizes the complete language, including rejecting malformed inputs. Any recurrent implementation whose complete stored configuration advances by a finite circuit inherits our persistent-bit lower bounds (Appendix B).

1.2 Summary of results

We use exact rational UHATs without positional encodings and finite-state recurrence with explicitly represented Boolean update and readout circuits. We write s for source description size and m for alphabet size.

Table 1 summarizes the mask-dependent replacement bounds as a function of source description size $s = |T|$, distinct from input length L and the dictionary family parameter n. It reports lower bounds on the persistent state bits $b(R)$ of a recurrent recognizer R and upper bounds on its description size $|R|$ (Section 2). Since $|R| \geq b(R)$, the persistent-bit lower bounds also imply description-size lower bounds. The hierarchy compares worst-case costs within each source class, not a common language: the dictionary languages, for instance, have no unmasked recognizer (Section 4.1).

The two main results are the following.

- **Doubly exponential mixed-mask separation (Section 3).** Theorem 3.2 gives sources of size $\mathrm{poly}(n)$ for which every equivalent recurrent recognizer requires at least 2^{2^n} persistent bits. The compact dictionary realization in Proposition F.2 supplies the strengthened lower bound with one unmasked retrieval between causal stages; the double-exponential general upper bound follows from the prior logical translation. On the same languages, every causal UHAT recognizer has size $2^{\Omega(n)}$ (Corollary 4.5).
- **Unmasked characterization and polynomial replacement (Section 4.1).** Theorem 4.1 characterizes unmasked UHAT languages exactly by occurrence profiles, with polynomial conversions between UHAT descriptions and profile predicate circuits. Every such source admits recurrent replacement of description size $\mathrm{poly}(s)$ using $O(m \log(m+1))$ persistent bits. The UHAT characterization and compiler extend the classical profile equivalence to this model.

Two further results refine and extend these endpoints.

- **Depth-sensitive recurrent compilation (Section 4.2).** Theorems 4.3 and 4.4 give a direct compiler and a depth refinement. For k causal retrievals followed by a possibly empty unmasked suffix, replacement is polynomial when $2^k \log(m+1) = O(\log s)$. The refinement and explicit circuit accounting supplement the single-exponential causal upper bound already implied by the prior translation.
- **Constant-error consequence (Section 3.3).** Theorem 3.3 and Proposition G.2 apply classical bounds from information theory to quantify how much memory recurrent models need for approximate retrieval under specified dictionary and query distributions. For independent uniform membership bits and an independent uniform query, achieving any fixed average error below one half still requires the number of persistent bits to be at least a constant fraction of the number of membership bits.

2 Models and replacement cost

An alphabet Σ is a nonempty finite set listed explicitly in the source description. A word $w = w_1 \cdots w_L$ has length L. We write Σ^* for all finite words, ε for the empty word, and $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$. A language is a set of words. Throughout, recognized languages are subsets of Σ^+; acceptance is read after a nonempty input. Thus exact equivalence means the same accept/reject decision on every finite nonempty sequence. All logarithms are base two, $[m] = \{1, \ldots, m\}$, and $\mathbb{Q}$ denotes the rationals.

Table 2: Resources and parameters used throughout. Description and state capacity are fixed; the state contents change as input symbols are processed.

Symbol	Meaning
$m = \lvert\Sigma\rvert$, $L = \lvert w\rvert$	Alphabet cardinality and input length
$s = \lvert T\rvert, \lvert R\rvert$	Source and recurrent binary description lengths
$b(R) = pD$	Persistent recurrent bits; at most $2^{b(R)}$ configurations
k	Retrieval depth, excluding position-wise operations

2.1 Unique hard-attention transformers

A unique hard-attention transformer (UHAT) has rational symbol embeddings and a finite sequence of attention and position-wise operations. At query position i, attention scores eligible positions j by $\langle A(v_i), B(v_j)\rangle$ for affine maps A, B, and chooses the leftmost or rightmost maximizer as specified for that operation. Its output is $C(v_i, v_j)$ for an affine map C. If no position is eligible, the selected vector is zero. Affine maps and coordinate-wise $\mathrm{ReLU}(x) = \max\{x, 0\}$ may be interleaved. A threshold at the last input position determines acceptance. UHAT selects one key after tie-breaking; its numerical maximizer need not be unique.

Each attention operation ℓ has a fixed mask $\mu_\ell \in \{\mathsf{Past}, \mathsf{All}, \mathsf{Future}\}$, with eligible keys at position i given by

$$E_i^{(\ell)} = \begin{cases} \{j : 1 \le j < i\}, & \mu_\ell = \mathsf{Past}, \\ \{1, \dots, L\}, & \mu_\ell = \mathsf{All}, \\ \{j : i < j \le L\}, & \mu_\ell = \mathsf{Future}. \end{cases}$$

The mask sequence is architectural: each operation applies the same rule at every position, while content-dependent scores select within the resulting eligible set. The class UHAT_0 uses only All, including $j = i$; the strict causal class UHAT_C uses only Past, which hides the present and later positions; and the mixed class UHAT_M permits arbitrary compositions, including sources using only one mask type. There are no positional encodings. Retrieval depth counts operations that use a selected value; position-wise computation is charged to model size but not retrieval depth. Appendix A fixes the complete semantics.

Evaluation is exact on a length-independent finite representation of reachable rational values. The polynomial bit bound of Bergsträßer et al. (2026, Proposition 12) justifies this representation. Arbitrary floating-point rounding is a different semantics: it can change a comparison, tie, or threshold, and needs a separate correctness argument.

2.2 Recurrent recognizers

Definition 2.1 (Finite-precision recurrent recognizer)**.** An RNN recognizer is $R = (\Sigma, g, h_0, f)$ with fixed hidden dimension D, finite format $\mathcal{F}_p \subset \mathbb{Q}$ of at most 2^p values, initial state $h_0 \in \mathcal{F}_p^D$, transition $g : \mathcal{F}_p^D \times \Sigma \to \mathcal{F}_p^D$, and acceptance map $f : \mathcal{F}_p^D \to \{0, 1\}$. It computes $h_t = g(h_{t-1}, w_t)$ and accepts iff $f(h_L) = 1$. Both D and p are independent of input length. Its persistent state has $b(R) = pD$ bits.

This is the general transition-function semantics of Bergsträßer et al. (2026, Section 2.4). A Boolean circuit is a finite network of AND, OR, and NOT gates acting on bits. The update circuit C_g takes the encoded current state and input symbol and outputs the encoded next state; the readout circuit C_f maps the state bits to one acceptance bit. We explicitly include both circuits in the model description. Every vector, buffer, and flag retained between tokens counts toward $b(R)$.

A b-bit state has at most 2^b configurations, and the same fixed circuit can update it for an arbitrarily long input. Input length counts these updates, state width counts retained bits, and circuit description counts the fixed implementation. Flattening a finite-precision state into bits preserves this accounting; conversely, a Boolean-circuit recurrence is an RNN using Boolean coordinates (Appendix B). Throughout, an RNN is this general circuit-defined recognizer, not a particular neural cell.

Definition 2.2 (Explicit model size)**.** Fix self-delimiting binary encodings of finite lists, integers, rationals, and circuits. We write $\mathrm{enc}(x)$ for the binary string representing x and $|\,\mathrm{enc}(x)|$ for its bit length. For tuples, $\mathrm{enc}(x_1, \dots, x_k)$ denotes the encoding of the ordered tuple $(x_1, \dots, x_k)$. The encoding explicitly lists the architecture and numerical coefficients, including zero entries in dense maps. For source architecture $\mathcal{A}_T$ and dense parameter list Θ_T, let

$$|T| = |\,\mathrm{enc}(\Sigma, \mathcal{A}_T)| + \sum_{\theta \in \Theta_T} |\,\mathrm{enc}(\theta)|. \tag{1}$$

The architecture includes dimensions, masks, tie rules, and operations; parameters include embeddings, affine maps, biases, and readout. For the recurrent target, let

$$|R| = |\,\mathrm{enc}(\Sigma, D, 1^p, \mathcal{F}_p)| + pD + |\,\mathrm{enc}(C_g)| + |\,\mathrm{enc}(C_f)|. \tag{2}$$

Here 1^p is a string of p ones, charging precision in unary. The pD term explicitly stores the initial state, so $|R| \ge b(R)$. Circuit encodings charge gates, wires, and constants; the finite format is fixed and specified.

These sizes describe the supplied representations, rather than arbitrarily compressed programs generating them. The predecessor's binary-size viewpoint and unary precision convention motivate this accounting (Bergsträßer et al., 2026, Section 2.5); the explicit circuit and dense-parameter representations here are additional conventions. At uniform parameter precision q, the parameter contribution to $|T|$ is qP_T for P_T listed scalars, plus architecture overhead. Persistent bits exclude fixed parameters and temporary workspace. Temporary circuit evaluation is allowed, but any information retained between tokens is charged to the state.

2.3 EXACT REPLACEMENT

Exact replacement of T asks for R satisfying $L(R) = L(T)$ on every finite nonempty input. The intermediate attention representations need not be reproduced; the final accept/reject answer must agree. We compare $|R|$ and $b(R)$ with $s = |T|$.

Definition 2.3 (Succinctness)**.** A representation class $\mathcal{A}$ is u-more succinct than $\mathcal{B}$ if there is a family $A_n \in \mathcal{A}$ of unbounded sizes, whose languages are representable in $\mathcal{B}$, such that every equivalent $B \in \mathcal{B}$ satisfies $|B| \geq u(|A_n|)$. Exponential and doubly exponential gaps mean $u(s) = 2^{s^{\Omega(1)}}$ and $u(s) = 2^{2^{s^{\Omega(1)}}}$, respectively. Here, $s^{\Omega(1)}$ denotes a function bounded below by cs^d for some constants $c, d > 0$ and all sufficiently large s.

Definition 2.4 (Bounded expansion)**.** The target class $\mathcal{B}$ has v-bounded expansion over $\mathcal{A}$ if every $A \in \mathcal{A}$ has an equivalent $B \in \mathcal{B}$ with $|B| \leq v(|A|)$.

Following Bergsträßer et al. (2026), a succinctness lower bound exhibits difficult source families, while a bounded-expansion upper bound supplies a replacement for every source in the stated class. We say that UHATs are more succinct than RNNs, and that RNNs have bounded expansion over UHATs.

For upper bounds, $2^{\mathrm{poly}(s)}$ denotes a bound of the form 2^{Cs^d} for fixed $C, d > 0$ and all sufficiently large s. The positive powers in these upper and lower bounds need not be integers: for example, $2^{\sqrt{s}}$ fits the exponential convention but is not $2^{\Theta(s)}$. Double-exponential expressions nest two exponentials. Matching tower heights need not match the powers or constants in the exponents.

3 A DOUBLY EXPONENTIAL GAP FROM ONE UNMASKED RETRIEVAL

Definition 3.1 (Addressed dictionary language)**.** Fix $n \geq 1$ and $\Sigma = \{0, 1, \#, \$\}$. A cell is $av\#$ with an n-bit address a and one payload bit v. For $x \in \{0,1\}^{2^n}$, its block B_x lists all addresses in increasing order, placing $x[a]$ at address a. Define

$$D_n = \{B_{x_1} \cdots B_{x_r} \,\$\, B_q : r \geq 0,\ q \in \{x_1, \ldots, x_r\}\}.$$

All other words are rejected. There are $N_n = 2^{2^n}$ blocks, each of length $\ell_n = 2^n(n+2)$.

Each block is a character sequence, not one token; its addresses are ordinary input symbols, not positional encodings. For example, at $n = 2$,

$$B_{1010} = \texttt{001\#010\#101\#110\#},$$

and $B_{1000}B_{1010}\$B_{1010}$ is accepted. In contrast, $B_{1000}B_{0010}\$B_{1010}$ is rejected: at each address some dictionary block agrees with the query, but no single block equals it. Empty dictionaries have valid syntax but cannot satisfy membership. The following family T_n recognizes D_n for arbitrarily many dictionary blocks with the same source, rejecting all malformed inputs.

Theorem 3.2 (Mixed-mask succinctness and bounded expansion)**.** UHAT_M *is doubly exponentially more succinct than the recurrent class of Definitions 2.1–2.2. The family* T_n *has size* $\mathrm{poly}(n)$ *and forces*

$$|R| \geq b(R) \geq 2^{2^n}$$

for every equivalent RNN. Conversely, that recurrent class has $2^{2^{\mathrm{poly}(s)}}$*-bounded expansion over* UHAT_M*. The bounds match in tower height.*

3.1 COMPACT REALIZATION IN THREE STAGES

Proposition F.2 realizes T_n with the mask sequence

$$\mathsf{Past}^{\,n+4} \to \mathsf{All} \to \mathsf{Past}^{\,3}.$$

For fixed n, it uses exactly $n + 8$ retrievals regardless of the number of dictionary blocks, and $O(n^2)$ interleaved position-wise operations in the conservative implementation; Table 4 in Appendix F.5 lists the retrievals. Figure 1 traces the three stages on the accepted example above.

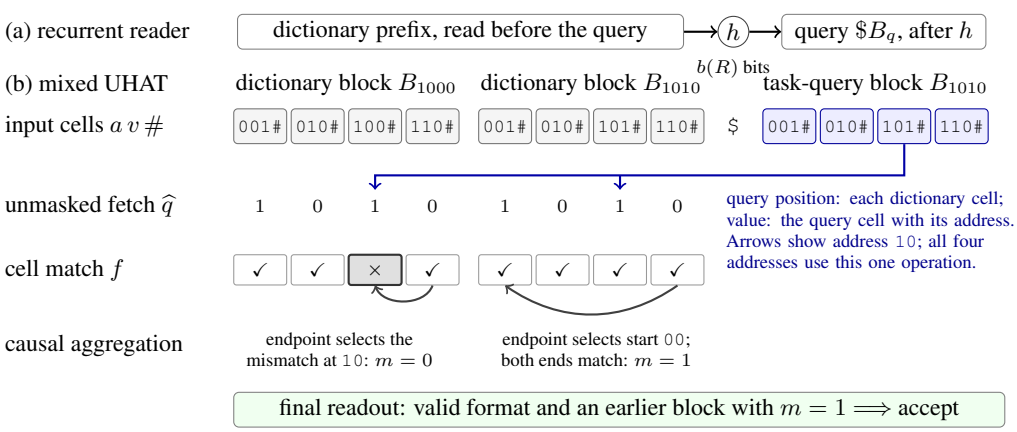


Figure 1: The dictionary construction on the accepted input $B_{1000}B_{1010}\$B_{1010}$ ($n = 2$). (a) A recurrent reader must compress the dictionary into its state h before the query arrives. (b) After $n + 4$ causal operations prepare cells, addresses, payloads, and format flags, one unmasked operation lets every dictionary cell fetch the query bit at its own address. Three causal operations then aggregate the flags f: each block endpoint selects its latest mismatch or its start, and the block passes ($m = 1$) only if the start is selected and both ends match.

Local preparation (causal). Constant scores and rightmost strict-past selection retrieve the immediately preceding position. Repeating this operation recovers the local window needed to identify completed cells, addresses, and payloads, with validity bits handling the input boundary. The next two operations identify the region and preceding completed cell. Shared scoring maps compare the n address coordinates and identify query cells; they enumerate neither the 2^n addresses nor the N_n possible blocks in their parameters. The same local fields feed a format guard that checks cell delimiters, a single separator at a cell boundary, consecutive addresses, complete dictionary blocks, and exactly one query block.

Query-dependent dictionary representations (unmasked). One unmasked operation then performs separate lookups across dictionary positions, copying the corresponding query bit into each dictionary cell. It is a single attention operation applied at every position: each dictionary cell is an attention query, and its selected value comes from the query cell with the same address in the later task-query block. A position-wise comparison then gives every completed dictionary cell a flag f recording whether its payload equals the query bit at its address.

Complete-block aggregation (causal). Acceptance requires whole-block agreement,

$$\exists B \text{ in the dictionary} \quad \forall a \in \{0,1\}^n, \quad B[a] = B_q[a],$$

not matches at different addresses in different blocks, as in the rejected example above. The aggregation must preserve this quantifier order. At a block endpoint, a rightmost mismatch-or-block-start retrieval detects an interior mismatch; the current block's start excludes all earlier blocks. The block passes exactly when the selected cell is its start and both the start and the current endpoint match the query. The final two retrievals detect a matching block and a format violation. Proposition F.2 and Appendix F supply the full guard, selector defaults, boundary cases, and conservative bound $|T_n| = O(n^6 \log(n + 2))$.

3.2 The recurrent lower bound

The lower bound counts the dictionaries that the recurrent state must distinguish at the separator:

$$n\text{-bit addresses} \to 2^n \text{ bits per block} \to N_n = 2^{2^n} \text{ blocks} \to 2^{N_n} \text{ dictionaries} \to b(R) \geq N_n.$$

For each subset, list its blocks once in a fixed order. Different subsets must give different answers to a common suffix $\$B_q$ with q in exactly one subset, while two prefixes leaving the same recurrent state give the same answer to every common suffix. The state at the separator therefore distinguishes the

$2^{N_n} - 1$ nonempty subsets, which forces $b(R) \geq N_n$. This classical prefix argument (Sipser, 2013; Kremer et al., 1999), made precise in Lemmas B.3 and F.1, combines with the compact source to prove the lower tower. The upper tower follows from Proposition 4.2(2). Section 4.1 shows that no unmasked UHAT recognizes D_n, and Corollary 4.5 shows that every causal UHAT recognizing D_n has size exponential in n.

3.3 Robustness to constant average error

Enumerate the N_n blocks and draw their membership table $X \in \{0,1\}^{N_n}$ with independent fair bits. Listing present blocks once in this order gives a canonical dictionary prefix. This distribution is *dense*: the expected dictionary contains $N_n/2$ blocks; it is not a uniform distribution over encoded strings. Let M be the recurrent state after the prefix, before the separator. An independent uniform index $I \in [N_n]$, supplied as the suffix $\$B_I$ for the enumerated block B_I, asks for X_I. Error is averaged over the table, query, and allowed randomness. Write $h_2(u) = -u \log u - (1-u)\log(1-u)$, with continuous endpoint values.

Theorem 3.3 (Constant-error consequence under a dense dictionary distribution)**.** *Let $X \in \{0,1\}^{N_n}$ and $I \in [N_n]$ follow the independent uniform distribution above. If a b-bit recurrent boundary state answers X_I with average error at most $\varepsilon \leq 1/2$, then*

$$b \geq N_n(1 - h_2(\varepsilon)).$$

The bound allows private or input-independent public randomness.

The state carries at most b bits of information about X, and Fano's inequality bounds the remaining uncertainty of each queried bit by h_2 of its error. These are classical random-access information bounds (Nayak, 1999) applied to the dictionary family; Appendix G gives the proof. Zero error recovers $b \geq N_n$; at error one half, guessing needs no information. Every fixed $\varepsilon < 1/2$ still requires a positive fraction of $N_n = 2^{2^n}$ bits, while the compact mixed source answers exactly. Thus the separation survives constant average error under this distribution. The density assumption matters: for dictionaries of r blocks and a uniform query, always answering "absent" already has error r/N_n. Finite-answer and sparse variants remain in Appendix G.

4 The polynomial and single-exponential regimes

The unmasked and causal classes admit cheaper replacement than the mixed class: unmasked attention reduces to occurrence profiles, and causal attention compiles through summaries of possible activation values.

4.1 Unmasked attention is determined by occurrence profiles

Let σ_w list symbols by first occurrence and π_w by last occurrence, and write $\mathrm{prof}(w) = (\sigma_w, \pi_w)$. For example,

$$\mathrm{prof}(\texttt{abaca}) = \mathrm{prof}(\texttt{aabbccaa}) = (\texttt{abc}, \texttt{bca}).$$

Every unmasked recognizer gives these words the same answer, despite their different repetition counts. It can still use the order exposed by extreme ties and last-position readout. A profile predicate circuit reads both lists, padded to $m = |\Sigma|$ slots. Equality of the orders is the classical regular-band equivalence (Kimura, 1958, Section 6, Theorems 10–11); see also Polák (2006, Result 1(ix)).

Theorem 4.1 (Profile characterization and polynomial expansion)**.** *Unmasked UHATs recognize exactly the profile-determined languages. Their descriptions and profile predicate circuits convert into each other with polynomial overhead. Every $T \in \mathsf{UHAT}_0$ has an equivalent RNN with*

$$|R| = \mathrm{poly}(|T|), \qquad b(R) \leq 2m\lceil \log(m+1) \rceil.$$

Thus RNNs have polynomial-bounded expansion over UHAT_0.

Equal input symbols retain equal vectors. For a fixed query symbol, maximizing keys form a union of symbol classes; σ_w determines its leftmost winner and π_w its rightmost winner. The lists thus determine every retrieval, and the last entry of π_w determines the readout symbol. However deep or wide the network, the input therefore affects the decision only through these two orders.

A canonical representative. Every word w has the representative $\sigma_w \pi_w$, the concatenation of its two lists. Its length is at most $2m$ and its profile is $\mathrm{prof}(w)$, so every unmasked UHAT gives it the same answer as w. Recurrence maintains the lists by appending a symbol to the first list only on its first appearance, and by moving it to the end of the last-occurrence list whenever it appears. Two m-slot arrays give the bit bound, and the readout circuit evaluates the source on the representative using exact polynomial-bit arithmetic.

For a fixed alphabet there are only finitely many profiles, independently of network width or depth, so each such language has a constant-size finite-state recognizer. A table indexed by profiles, however, has at least $(m!)^2$ entries; the uniform polynomial guarantee controls description size as the explicitly listed alphabet grows. Conversely, successive extreme retrievals recover both lists, after which position-wise Boolean gates evaluate any profile circuit. Appendix D proves both conversions and their padding conventions, and gives the verification consequences in Proposition D.2.

Consequence for dictionaries. For fully unmasked attention, the inputs

$$\texttt{00\#10\#\$00\#10\#} \text{ (accept)}, \qquad \texttt{00\#11\#\$00\#10\#} \text{ (reject)}$$

have the same first- and last-occurrence orders $(0, \#, 1, \$)$ and $(\$, 1, 0, \#)$. Theorem 4.1 rules out unmasked recognition of D_1; Appendix D extends this to every n.

4.2 Causal compilation and retrieval depth

The following proposition collects the consequences of Bergsträßer et al. (2026) used in Table 1. Their derivations under our explicit recurrent representation are given in Appendix C.

Proposition 4.2 (Consequences of prior work)**.** *Under Definitions 2.1–2.2, with $s = |T|$:*

1. *Every UHAT T has polynomial-bit reachable vectors, uniformly in input length, and an equivalent linear temporal logic (LTL) formula of size $2^{\mathrm{poly}(s)}$ (Bergsträßer et al., 2026, Propositions 12–13).*
2. *Every $T \in \mathsf{UHAT}_M$ has an equivalent RNN R with $|R| \le 2^{2^{\mathrm{poly}(s)}}$.*
3. *Every $T \in \mathsf{UHAT}_C$ has an equivalent RNN R with $|R| \le 2^{\mathrm{poly}(s)}$.*
4. *For the causal witness family of Bergsträßer et al. (2026, Corollary 11, Theorem 15, Corollary 18), every equivalent RNN R satisfies $b(R) \ge 2^{s^{\Omega(1)}}$, and hence the same lower bound holds for $|R|$.*

Item 3 bounds causal replacement through temporal logic. A direct compiler instead exposes the parameter that governs the recurrent cost: the number of possible activation values. An activation value here is a whole hidden vector; its possible-value universe ranges over all inputs for a fixed source. Extending a causal input leaves its past values unchanged. For each retrieval operation, ordered summaries retain the first and last occurrences of its possible input vectors: only these occurrences can win the prescribed extreme ties. Querying before inserting the current key preserves the strict mask, and circuits polynomial in summary capacity implement scores and updates (Appendix E).

Theorem 4.3 (Direct activation-summary compiler)**.** *Every $T \in \mathsf{UHAT}_C$ has an equivalent RNN whose persistent-bit count and description size are polynomial in $|T|$ and in the total number of possible input vectors of its retrieval operations. The compiler retains first- and last-occurrence summaries of these vectors and evaluates each query before inserting its current key. The polynomial-bit bound makes this number at most $2^{\mathrm{poly}(|T|)}$, so the RNN has size $2^{\mathrm{poly}(|T|)}$; together with Proposition 4.2(4), the bounds agree at a single-exponential tower.*

The inherited lower tower uses polynomial-size causal sources whose shortest accepted words have length at least 2^{2^n} (Bergsträßer et al., 2026, Corollary 11, Theorem 15, Corollary 18). A b-bit recognizer with a nonempty language accepts a word of length at most 2^b, so these sources force $b \ge 2^n$ (Appendix C).

Retrieval depth. The depth refinement restricts the mask sequence to $\mathsf{Past}^{\,k}\mathsf{All}^{\,r}$, $r \ge 0$, where superscripts count repetitions. Here k counts the causal prefix and the total number of retrievals is

$k + r$; position-wise operations are excluded. A later causal operation falls outside this order: its inputs may already depend on future positions.

Theorem 4.4 (Depth-sensitive bounded expansion)**.** *If T has k causal retrieval operations followed by zero or more unmasked retrievals, with position-wise operations interleaved, it has an equivalent RNN of size*

$$\min\{(m+1)^{O(2^k)},\ 2^{\mathrm{poly}(|T|)}\}\,\mathrm{poly}(|T|).$$

This is polynomial when $2^k \log(m+1) = O(\log |T|)$, in particular for constant k.

For a fixed alphabet, polynomial replacement thus remains possible through $k \le \log\log|T| + O(1)$ causal retrievals, followed by any unmasked suffix. After j causal retrievals, let v_j bound the number of possible whole vectors. A new value depends on a current vector and either a selected vector or the null result; hence the inherited recurrence (Yang et al., 2024, Appendix C.2, Lemma 22) gives

$$v_{j+1} \le v_j(v_j+1), \quad v_0 \le m, \qquad v_j \le (m+1)^{2^j} - 1.$$

Position-wise maps cannot increase this cardinality. We use the bound to size the summaries of Theorem 4.3, then treat the realized sequence of causal-output vectors as a word over an effective alphabet. Its occurrence profile suffices to evaluate the unmasked suffix in the acceptance circuit. An empty suffix covers pure causal networks (Appendix E). Appendix H gives fixed-alphabet depth witnesses whose inputs carry explicit clock fields and whose source descriptions grow as $2^{O(k)}$.

Causal versus mixed masks on one language. A fully causal final query can attend to the preceding dictionary; earlier positions' inability to see the query alone proves no lower bound. The compiler nevertheless separates the two mask classes on the dictionary languages.

Corollary 4.5 (Causal versus mixed recognition of the same language)**.** *The dictionary language D_n has a mixed-mask recognizer of size $\mathrm{poly}(n)$ with one unmasked retrieval, whereas every causal UHAT recognizing D_n has size $2^{\Omega(n)}$.*

For a causal source of size s_C recognizing D_n, the compiler in Theorem 4.3 and the dictionary lower bound imply, for fixed $K, d > 0$,

$$2^{2^n} \le 2^{K s_C^d} \quad \Longrightarrow \quad s_C \ge 2^{\Omega(n)}.$$

This excludes every polynomial-size causal strategy, including searches initiated from the final query. In the mixed construction, later causal operations aggregate representations already made query-dependent by the unmasked retrieval, outside the order of Theorem 4.4.

5 Conclusion

Attention masks determine the cost of exact recurrent replacement. Fully unmasked models reduce to occurrence profiles and admit polynomial compilation, while a single unmasked retrieval between causal stages suffices to realize polynomial-size dictionary recognizers that require doubly exponentially many recurrent state bits. The direct compiler further shows how causal retrieval depth bounds replacement cost before an unmasked suffix; a depth-sensitive classification of arbitrary mask interleavings remains open (Appendix **??**). The hierarchy separates the ability to reproduce attention's decisions from the memory and description required to do so.

AI use statement

Generative AI tools assisted with proof review, literature lookup, organization, and prose editing. The authors are responsible for the mathematical claims and references.

Reproducibility statement

Sections 2–4 specify the models, resource measures, and theorem statements. Appendices A–H give the constructions and complete proofs, including the encoding conventions and resource accounting. The self-contained source includes the style, bibliography, and mechanism figure. All reported results are theoretical.

REFERENCES

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. Zoology: Measuring and improving recall in efficient language models. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

Pascal Bergsträßer, Ryan Cotterell, and Anthony Widjaja Lin. Transformers are inherently succinct. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2510.19315v3. arXiv:2510.19315v3, 15 May 2026.

Satwik Bhattamishra, Michael Hahn, Phil Blunsom, and Varun Kanade. Separations in the representational capabilities of transformers and recurrent architectures. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, 2024.

Giuseppe De Giacomo and Moshe Y. Vardi. Linear temporal logic and linear dynamic logic on finite traces. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*, 2013.

Hana Galperin and Avi Wigderson. Succinct representations of graphs. *Information and Control*, 56 (3):183–198, 1983. doi: 10.1016/S0019-9958(83)80004-7.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. arXiv:2312.00752, 2023. URL https://arxiv.org/abs/2312.00752.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. Repeat after me: Transformers are better than state space models at copying. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 21502–21521, 2024.

Selim Jerad, Anej Svete, Jiaoda Li, and Ryan Cotterell. Unique hard attention: A tale of two sides. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 977–996. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-short.76.

Naoki Kimura. The structure of idempotent semigroups. I. *Pacific Journal of Mathematics*, 8(2): 257–275, 1958. doi: 10.2140/pjm.1958.8.257.

Ilan Kremer, Noam Nisan, and Dana Ron. On randomized one-round communication complexity. *Computational Complexity*, 8(1):21–49, 1999. doi: 10.1007/s000370050018.

William Merrill, Jackson Petty, and Ashish Sabharwal. The illusion of state in state-space models. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 35492–35506, 2024.

Ashwin Nayak. Optimal lower bounds for quantum automata and random access codes. arXiv:quant-ph/9904093v3, 1999. URL https://arxiv.org/abs/quant-ph/9904093v3.

Franz Nowak, Ryan Cotterell, and Reda Boumasmoud. A compositional theory of causally masked transformers. arXiv:2607.26988v1, 2026. URL https://arxiv.org/abs/2607.26988v1.

Christos H. Papadimitriou and Mihalis Yannakakis. A note on succinct representations of graphs. *Information and Control*, 71(3):181–185, 1986. doi: 10.1016/S0019-9958(86)80009-2.

Libor Polák. Small conjunctive varieties of regular languages. *Acta Cybernetica*, 17(4):825–841, 2006. URL https://acta.bibl.u-szeged.hu/12798/.

Clayton Sanford, Daniel Hsu, and Matus Telgarsky. Representational strengths and limitations of transformers. In *Advances in Neural Information Processing Systems 36 (NeurIPS)*, 2023.

Yash Sarrof, Yana Veitsman, and Michael Hahn. The expressive capacity of state space models: A formal language perspective. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, 2024.

Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970. doi: 10.1016/S0022-0000(70)80006-X.

Claude E. Shannon. Coding theorems for a discrete source with a fidelity criterion. *IRE National Convention Record*, 7(4):142–163, 1959.

Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, third edition, 2013.

A. Prasad Sistla and Edmund M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, 1985. doi: 10.1145/3828.3837.

Andy Yang, David Chiang, and Dana Angluin. Masked hard-attention transformers recognize exactly the star-free languages. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, 2024. URL https://arxiv.org/abs/2310.13897v4. arXiv:2310.13897v4.

A CONVENTIONS AND A GUIDE TO THE PROOFS

The main-text statements use one source model and one size measure throughout. This appendix makes implementation conventions explicit; the subsequent appendices organize the proofs by construction and imported ingredient. The organization is summarized in Table 3.

Table 3: Where each main claim is justified, in the order of the main text. Prior theorem numbers refer to Bergsträßer et al. (2026), arXiv version 3.

Main statement	Proof location	Role of earlier results
RNN model / prefix interface	App. B	Classical finite-state and communication facts
Thm. 3.2; Prop. F.2; Lem. F.1	App. F	New dictionary realization; restricted B-RASP compiler (Lemma 9) imported
Thm. 4.1; Props. D.2, D.3	App. D	Classical profile quotient; UHAT/circuit correspondence; imported finite-value bound
Prop. 4.2: inherited bounds	App. C	Props. 12–13; Cor. 11; Thm. 15; LTL-to-NFA translation
Thm. 4.3: direct compiler	App. E	Direct proof of an already implied bound, with value counts
Thm. 4.4: depth refinement	App. E	Known value recursion; profile evaluation of the unmasked suffix
Cors. 4.5, F.3	App. F	Dictionary rows with the direct compiler and automata translations
Thm. 3.3; Props. G.1, G.2	App. G	Classical information bounds applied to the witness
Thm. H.1; Cor. H.2	App. H	Depth circuit bound, clocked witness, binary simulation

Source semantics. For an input $w_1 \cdots w_t$, the initial value at position i is $\mathrm{emb}(w_i) \in \mathbb{Q}^{d_0}$. An attention layer has affine maps A, B, C, an eligibility mask $M(i,j)$, and a tie rule $\tau \in \{\min, \max\}$. Let

$$E_i = \{j : M(i,j) = 1\}, \qquad J_i = \arg\max_{j \in E_i} \langle A(v_i), B(v_j) \rangle.$$

If $E_i \neq \varnothing$, the selected vector is $a_i = v_{\tau(J_i)}$; otherwise $a_i = 0$. The output is $C(v_i, a_i)$. Position-wise affine maps and ReLUs may be interleaved, and the final test is $\langle u, v_t \rangle > 0$, with an affine bias represented by a constant coordinate. All coordinates and parameters have explicit rational encodings. An unmasked layer has $M \equiv 1$, including $i = j$.

Position-wise affine operations can be implemented as degenerate attention layers whose update ignores the selected vector. They make no retrieval and do not increase a value-set cardinality. In the depth results we count the attention operations that use a selected value; such affine-only operations are included with the interleaved position-wise computation. If one counts all degenerate syntactic layers as well, the statement should be read in this retrieval-depth parameter rather than as an unchanged raw layer count.

Arithmetic and encoding. The finite-value conclusion of Bergsträßer et al. (2026, Proposition 12) bounds reachable-vector encodings polynomially in source size, independently of input length. We give a self-contained justification that tracks coefficient numerators as well as denominators. A common denominator alone cannot bound scaled coefficient magnitudes: even when it is 1, an integer coefficient may have a large encoded numerator. This supplies the needed detail in the cited proof, without changing its conclusion.

Put $s = |T| \geq 2$. Under the explicit dense encoding, the number of operations, maximum width, and number of encoded rational parameters are $O(s)$, and every parameter has $O(s)$ numerator and denominator bits. Choose bounds $K, W, B = O(s)$ for the number of affine updates, maximum width, and parameter bit length, respectively. Include embeddings, biases, score maps, and the

final readout among the parameters. Let D be the product of all their positive denominators, so $\log D = O(s^2)$ and every parameter denominator divides D. Each coefficient and bias has absolute value at most 2^B.

After t affine updates to forwarded coordinates, use the common denominator $q_t = D^{t+1}$. Initially the embedding denominators divide D. An affine update uses at most $2W$ previous coordinates (the current and selected vectors) and a bias; its output denominator divides $Dq_t = q_{t+1}$, including the bias since its denominator divides D. Selected-value copying, the null zero vector, and ReLU preserve this divisibility and do not increase magnitudes. If $A_t \geq 1$ bounds the absolute values of forwarded coordinates, then

$$A_0 \leq 2^B, \qquad A_{t+1} \leq (2W+1)2^B A_t, \qquad \log A_t \leq B + t\big(B + \log(2W+1)\big).$$

For $t \leq K = O(s)$, this is $O(s^2)$. Writing a coordinate as p/q_t therefore gives

$$\log q_t = O(s^3), \qquad \log(1+|p|) \leq \log q_t + \log(1+A_t) = O(s^3).$$

These safe bounds also include biases and the final affine readout.

Score maps are used for selection, not forwarded recursively as coordinates. Each query or key coordinate is an affine expression with denominator dividing Dq_t and polynomial-bit numerator. A dot product of at most W such coordinate pairs has denominator dividing $(Dq_t)^2$ and polynomial-bit numerator; exact comparison by cross multiplication remains polynomial-bit. Although arbitrarily many input positions may be compared, every comparison has this fixed bit bound. Hence a whole reachable vector has polynomial encoding length and there are at most $2^{\mathrm{poly}(s)}$ possible vectors for a fixed source. Fixed-width addition, multiplication, comparison, and multiplexing give polynomial-size Boolean circuits for each arithmetic step; no unreduced-fraction representation is needed.

This is not an arbitrary rounding convention for UHATs: rounding may change equalities, ties, or thresholds. On the recurrent side, the specified finite format is implemented exactly by its transition circuit. The lower bounds require only the charged persistent state bits.

Hypotheses that affect the conclusions. The explicit alphabet is essential for a polynomial description bound. With a succinctly described alphabet of exponential size, storing its two orders need not be polynomial in the embedding circuit. Reflexive unmasking is also essential: if a position may attend everywhere except itself, it can detect whether its symbol occurs elsewhere, which the two-order profile does not record. The last-position readout recovers the final symbol from the last-occurrence order. An ordinary extra beginning-of-sequence symbol does not by itself invalidate the profile proof; a genuinely position-dependent embedding or readout requires a new argument. Finally, hard *unique* selection is different from averaging over tied maxima. A constant score with leftmost tie-breaking can read the first symbol, so permutation invariance is not the appropriate invariant here.

Language-class terminology. A language is regular if some finite automaton recognizes it. A star-free language can be defined using finite-language constants, Boolean operations, and concatenation, without the Kleene-star operation; equivalently it is definable in first-order logic with the linear order on positions, written $\mathrm{FO}[<]$. The equivalence between the relevant masked UHAT/B-RASP setting and star-free languages is background (Yang et al., 2024). We use only the containment in regular languages and the explicit translations cited in Proposition 4.2, not an unproved converse at the same description size.

B Sequential circuits, automata, and boundary states

B.1 Circuit realization of the general RNN model

The recurrent semantics in Definition 2.1 are those of Bergsträßer et al. (2026, Section 2.4): one fixed transition map advances a finite hidden vector for each input symbol, and one fixed map decides acceptance. Their model does not prescribe a particular affine update or neural-cell topology. Our additional representation convention specifies these two finite maps by Boolean circuits and charges their encodings in $|R|$. This is the condition that makes the constructive upper bounds quantitative.

Proposition B.1 (Bit-state correspondence)**.** *For the circuit representation in Definition 2.2, flattening a finite-precision RNN gives a Boolean-circuit sequential acceptor with the same $b = pD$ persistent bits and polynomially related description size. Conversely, every Boolean-circuit sequential acceptor with b stored bits is such an RNN with $D = b$, $p = 1$, and $\mathcal{F}_1 = \{0, 1\}$, preserving its hidden-state width and circuit description up to fixed encoding overhead.*

Proof. Let $\mathrm{code} : \mathcal{F}_p^D \to \{0,1\}^{pD}$ be the specified state encoding. On valid codes, the bit transition and readout are

$$\delta(x, a) = \mathrm{code}\big(g(\mathrm{code}^{-1}(x), a)\big), \qquad \alpha(x) = f(\mathrm{code}^{-1}(x)).$$

They are precisely the maps represented by C_g and C_f. Invalid codes can be assigned a fixed rejecting behaviour and are never reached from the valid initial state. The initial bit string is $x_0 = \mathrm{code}(h_0)$, and induction on input length proves identical decisions. If one starts instead with a b-bit circuit recurrence (x_0, δ, α), regard its bits as rational coordinates taking values zero and one, and set $h_0 = x_0$, $g = \delta$, and $f = \alpha$. This is an instance of the general recurrent semantics, without adding input steps or persistent memory. If functions on all of $\mathbb{Q}^b$ are desired, extend them arbitrarily off the reachable Boolean domain; those values are irrelevant to the finite-precision run. □

Consequently the profile, activation-summary, temporal-monitor, and NFA-subset circuit constructions below are already RNN constructions under the stated encoding. No general Boolean circuit is silently converted to a single affine recurrent cell. All state-count lower bounds are unchanged; their corresponding bit lower bounds become explicit-description lower bounds via $|R| \geq b(R)$. Proposition B.1 records the resource-preserving identification with a Boolean-circuit sequential acceptor under the specified representation.

For a particular finite-precision neural or SSM implementation, concatenate every persistent coordinate and buffer. Compile each arithmetic primitive, including its rounding or saturation convention, into Boolean gates. If those primitive circuits have polynomial size, the resulting representation has polynomial overhead and preserves the stored-bit count. Our lower bounds therefore apply to it. An upper-bound circuit may require more resources when implemented in a restricted neural-cell syntax, and no equal-parameter-count converse is asserted. The general RNN language class contains all regular languages, including one-bit parity languages outside the positional-encoding-free UHAT class.

Proposition B.2 (Automata associated with an RNN)**.** *An RNN with b persistent bits induces a DFA with at most $2^b + 1$ states. An explicit Q-state DFA over Σ has an RNN encoding using $\lceil \log Q \rceil$ persistent bits and description polynomial in its transition table. An explicit Q-state NFA has an RNN simulation using Q bits and a transition circuit of size $O(|\Sigma|Q^2)$ up to symbol-decoding overhead.*

Proof. For the first direction, use the 2^b bit strings as states with the circuit-defined transitions. Add a fresh rejecting initial state whose outgoing transitions on a go to $\delta(x_0, a)$. This enforces the convention that no empty word is accepted, even if $\alpha(x_0) = 1$.

For a DFA, encode its state number and implement the finite transition table with comparisons and multiplexers. For an NFA without epsilon transitions, store its current reachable-state subset. The next subset coordinate for state q is

$$x'_q = \bigvee_{p \in [Q]} \big(x_p \wedge \mathbf{1}[(p, a, q) \text{ is a transition}]\big).$$

The acceptance circuit tests intersection with the final states. Epsilon closures, if present, can first be incorporated in the fixed transition relation. The transition circuit is polynomial in the explicit NFA size; no table over all 2^Q subsets is constructed. □

B.2 Boundary states and communication

For a language $\mathcal{L}$ and prefix p, write $F_p(q) = \mathbf{1}[pq \in \mathcal{L}]$ for its future-answer map. Two prefixes are distinguishable if a common suffix requires different decisions.

Lemma B.3 (Classical boundary-state principle)**.** *If a language has Q distinguishable nonempty prefixes, every equivalent recurrent recognizer needs $b \geq \lceil \log Q \rceil$ persistent bits. Under a distribution on prefixes and suffix queries, a b-bit boundary state is a one-way communication message of at most b bits.*

The future-answer maps are the rows of the Boolean Hankel matrix $H_{\mathcal{L}}[p, q] = \mathbf{1}[pq \in \mathcal{L}]$. Suppose two nonempty prefixes p, p' leave an RNN in the same state. Determinism implies that every common suffix q leaves the same final state in both runs. Consequently pq and $p'q$ have equal acceptance values. Distinct Hankel rows therefore require distinct states. The restriction to nonempty prefixes avoids the exceptional treatment of the empty word; whenever needed below, the nonempty row subfamily already gives the claimed integer bit bound.

For a communication interpretation, Alice runs the prefix and sends its state to Bob, who continues on the query suffix. All information persisting across this boundary is charged to the message, and input-independent public randomness may be exposed for free. A randomized machine also yields such a protocol, with its coins made public if desired. This can strengthen the protocol and hence is valid for a lower bound. The underlying prefix equivalence is the Myhill–Nerode relation (Sipser, 2013).

B.3 Classical verification baseline

Proposition B.4 (Succinct-state verification)**.** *RNN non-emptiness, universality, and equivalence are* PSPACE*-complete.*

Here PSPACE is the class decidable with polynomial working space. This is the standard succinct-state reachability phenomenon (Galperin & Wigderson, 1983; Papadimitriou & Yannakakis, 1986), specialized to the circuit representation of Definition 2.2.

Proof. For non-emptiness, guess input symbols while storing the current b-bit state and a counter up to 2^b. If any nonempty accepting run exists, a shortest one has length at most 2^b: repeated states can be deleted unless the only repetition closes the run from the initial state, in which case its length is still at most the number of states. Evaluating the transition circuit and maintaining the counter use polynomial space. Nondeterministic polynomial space equals deterministic polynomial space (Savitch, 1970).

For hardness, fix a polynomial-space decider and an instance. A polynomial-size circuit maps a coded configuration to its next configuration, with accepting and rejecting halting configurations absorbing. Use a unary input alphabet to advance this circuit. The initial state encodes the starting configuration and the acceptance predicate tests for the accepting halt. Some positive-length word is accepted exactly when the decider accepts. The construction has polynomial description.

For equivalence, search the product state space for a word on which acceptance differs. Polynomial space is closed under complement. Hardness follows by comparison with an always-rejecting RNN. Universality is the complement version of non-emptiness because flipping the acceptance bit complements the language within Σ^+. These arguments concern circuit-defined acceptors, not every restricted named SSM syntax. □

The prior transformer verification result is EXPSPACE-completeness, where EXPSPACE allows exponential working space (Bergsträßer et al., 2026). We use this only as context. The unconditional succinctness lower bounds in this paper come from explicit distinguishable-prefix families, not from an assumed separation between complexity classes.

C What follows from the prior logical construction

This appendix establishes Proposition 4.2 from the prior logical and witness constructions. Item 1 invokes Bergsträßer et al. (2026, Propositions 12–13), with arithmetic details given in Appendix A. The following subsections give the resource transfers for items 2–4.

Linear temporal logic (LTL) describes word-position properties using Boolean and temporal operators. A deterministic finite automaton (DFA) stores one state; a nondeterministic finite automaton (NFA) permits several possible successors and can be simulated by retaining the reachable-state set.

A Q-state NFA uses Q subset bits; the recurrent circuit need not explicitly list its 2^Q possible subsets. This distinction avoids an extra exponential in the mixed upper bound. For the causal lower bound, the inherited family has source size $\mathrm{poly}(n)$ and shortest accepted word length at least 2^{2^n}. A nonempty language of a b-bit recognizer has an accepted word of length at most 2^b, giving $b \geq 2^n$. The tiling witness is inherited; the mixed dictionary below uses a different many-prefix construction.

C.1 General mixed-mask upper bound

Let $s = |T|$ and let φ_T be the formula supplied by the prior UHAT-to-LTL translation, of size $h \leq 2^{p(s)}$ for a fixed polynomial p. The standard temporal-tableau method gives an exponential-size NFA (Sistla & Clarke, 1985). The usual future-only finite-word construction is background (De Giacomo & Vardi, 2013); we spell out the past-and-future case to match our strict operators and last-position readout.

For completeness, the exponential NFA bound holds with both past and future operators under our finite-word, last-position convention. Let $\mathrm{Sub}(\varphi)$ be the subformulas of φ, and let $r = |\mathrm{Sub}(\varphi)|$. At each input position, an NFA guesses the r truth values of these subformulas. It checks atomic propositions against the current symbol and checks all Boolean connectives locally. For adjacent truth assignments v and v' it checks, for each strict since or until subformula,

$$v'(\alpha S \beta) = v(\beta) \vee \big(v(\alpha) \wedge v(\alpha S \beta)\big), \qquad v(\alpha U \beta) = v'(\beta) \vee \big(v'(\alpha) \wedge v'(\alpha U \beta)\big).$$

At the first position all strict-since values are required to be false; at the last position all strict-until values are required to be false, and φ must be true. Existential past and future operators are the special cases $\top S \beta$ and $\top U \beta$. For each temporal subformula, its recurrence and endpoint condition uniquely determine its values once its proper subformulas are fixed. Induction on subformula structure therefore proves that the accepted guesses are exactly the semantic truth assignments. The NFA stores one assignment and a fresh nonaccepting start state, so it has at most $1 + 2^r = 2^{O(|\varphi|)}$ states. Its explicit transition relation has size $2^{O(|\varphi|)}|\Sigma|$, which remains exponential. This also rejects the empty word as required.

Apply this construction to φ_T. With $Q \leq 2^{O(h)}$ NFA states, Proposition B.2 stores the reachable subset in Q bits and realizes its update by a circuit polynomial in Q and the explicit alphabet. Consequently

$$|R| \leq \mathrm{poly}(Q, |\Sigma|) = 2^{2^{\mathrm{poly}(s)}}.$$

This proves Proposition 4.2(2). Explicit determinization could instead use 2^Q DFA states. Retaining a subset in a circuit recurrence avoids that extra exponential in recurrent description.

C.2 Causal specialization and the monitor recurrence

The strict-future-masking clauses in Bergsträßer et al. (2026, Appendix A.5, equations (26)–(28)) use only Boolean operations and past temporal operators. Embeddings and ReLUs introduce no future operator. By induction, a fully causal network therefore has a past-only formula of size $2^{\mathrm{poly}(s)}$, valid for either permitted tie rule. This is a property of the displayed construction, stronger than a bare translation to unrestricted LTL.

For clarity, write $P\beta$ for "β held at some strictly earlier position", and use strict since semantics

$$(\alpha \, S \, \beta)_t \quad \Longleftrightarrow \quad \exists j < t : \ \beta_j \ \wedge \bigwedge_{j<r<t} \alpha_r.$$

A monitor stores subformula truth values from the preceding step. If z_t is the value of $\alpha S \beta$, then

$$z_1 = 0, \qquad z_t = \beta_{t-1} \vee (\alpha_{t-1} \wedge z_{t-1}) \quad (t > 1).$$

Similarly, $(P\beta)_1 = 0$ and $(P\beta)_t = \beta_{t-1} \vee (P\beta)_{t-1}$. Boolean combinations are computed in subformula order at the current position. A first-step bit enforces zero values for all strict-past

temporal subformulas at the first position. Alternate inclusive-since conventions have the equally local recurrence $z_t = \beta_t \vee (\alpha_t \wedge z_{t-1})$; no change in asymptotic resources is involved.

There are $O(h)$ subformulas, so $O(h)$ state bits and a polynomial-size transition circuit suffice. With $h = 2^{\mathrm{poly}(s)}$, this proves item 3 of Proposition 4.2. The direct compiler in Appendix E is useful because it retains value-cardinality information lost when only the formula-size estimate is used.

C.3 The inherited exponential lower bound

The prior hard family has size polynomial in n, uses causal masking and rightmost selection, and has no accepted word shorter than 2^{2^n}; non-emptiness is part of that witness. An RNN with b persistent bits recognizing a nonempty language accepts some word of length at most 2^b, by the shortest-run argument above. Hence this family forces $b \geq 2^n$ and $|R| \geq 2^n$. This proves item 4 of Proposition 4.2 directly for RNNs.

The state-count proof applies to the general recurrent semantics regardless of the implementation of its transition map. It does not rely on a lower bound valid only for a restricted neural-cell syntax. The explicit initial-state and circuit encoding then converts the hidden-state bound into the description bound in Definition 2.2. The old tiling reduction is needed to establish that family; once its theorem is imported, repeating the reduction is unnecessary. The new dictionary construction below uses a different, many-row argument, since its own shortest member is only about $2\ell_n$ symbols long.

D Unmasked profiles: characterization and compilation

The two-order quotient is classical in regular-band theory (Kimura, 1958, Section 6); see also Polák (2006, Section 2, Result 1(ix)). The proof below concerns the exact UHAT characterization and its circuit resource bounds.

Lemma D.1 (Equal-symbol invariant)**.** *For an unmasked UHAT and each layer j, there is a map Φ_j from occurring input symbols to current vectors, depending only on* $\mathrm{prof}(w)$*, such that $v_i^j = \Phi_j(w_i)$ at every position i.*

Proof. At the embedding layer the map is the symbol embedding. A position-wise operation composes it with the same map everywhere. For an attention layer, fix a query symbol a and write $u_c = \Phi_{j-1}(c)$. The maximizing keys are exactly the positions whose symbols lie in

$$Y(a) = \operatorname*{arg\,max}_{c \text{ occurring in } w} \langle A(u_a), B(u_c) \rangle.$$

Because attention includes every position, this is a union of entire symbol classes. The selected symbol is the σ_w-first element of $Y(a)$ under leftmost selection, or the π_w-last element under rightmost selection. Define $\Phi_j(a) = C(u_a, u_{c^*})$. This depends only on the two symbol orders and the previous map, proving the induction. Different symbols may have identical embeddings or merge later; indexing by input symbols keeps the argument valid in either case. At every layer there are at most $|\Sigma|$ realized values. □

D.1 Proof of the polynomial RNN upper bound

Let $m = |\Sigma|$. Use two arrays of m slots, each holding a code from $\{0, 1, \ldots, m\}$, with zero denoting unused slots. They store σ and π in increasing first- and last-occurrence order. On input a, append it to σ only if absent. Remove it from π if present and append it at the end. Equality tests, prefix Boolean operations, and conditional shifts give a transition circuit polynomial in m. The total persistent-bit count is exactly $2m\lceil \log(m+1) \rceil$.

For acceptance, initialize one vector for each live symbol and simulate T using Lemma D.1. At a layer compute the pairwise scores on the live slots, identify each maximum, and choose the winner using the stored occurrence orders. Empty slots are excluded by an explicit validity flag, not assigned an arbitrary numeric score that might win. Arithmetic circuits use the polynomial rational bit bound of Appendix A. There are polynomially many operations in $|T|$ and m, and the final symbol is the last element of π. The acceptance circuit is therefore polynomial in $|T|$. It is combinational computation, not additional persistent state. This proves the recurrent upper bound and the forward characterization in Theorem 4.1.

D.2 Recovering a profile and evaluating its predicate

Use one-hot symbol coordinates and a shared Boolean vector E of extracted symbols. Initially $E = 0$. At each position carrying symbol a, materialize a Boolean flag $f = E_a$ with position-wise gates. Give every key score $-f$ and use leftmost selection. If an unextracted symbol remains, the selected key is its earliest occurrence. Otherwise the selected flag is one, allowing an exhaustion gate to write a null slot. Broadcast the selected one-hot symbol to all positions, append it to a list when nonexhausted, and update E by coordinatewise Boolean OR with the selected one-hot symbol.

After m selections this recovers σ_w at every position, with unused slots padded at the tail. Reset the extracted-symbol vector E to zero before beginning the rightmost pass. This pass produces the live entries of π_w in reverse order, followed by null slots. Compute their number, reverse only those live entries, and retain tail padding in the remaining slots. Boolean counting, multiplexing, and compaction realize this rearrangement with polynomial-size position-wise circuitry, so the resulting encoding is exactly the one expected by the profile predicate circuit. AND, OR, and NOT on Boolean coordinates have constant-size affine/ReLU implementations, for example $x \wedge y = \mathrm{ReLU}(x + y - 1)$ and $\neg x = 1 - x$. An arbitrary profile predicate circuit can thus be evaluated position-wise after extraction. The construction has size polynomial in m and the circuit size. Every profile-determined language over a fixed finite alphabet has some Boolean predicate circuit because there are finitely many profiles. This proves the converse of Theorem 4.1. The polynomial relation is to *profile predicate circuits*; it is not a claim of matching minimal RNN polynomial degree.

The canonical representative. Every pair of orderings (σ, π) of the same nonempty symbol set is realized by the word $\sigma\pi$ of Section 4.1. Its length is at most $2m$, its first-occurrence order is σ, and its last-occurrence order is π. This gives a second direct way to evaluate a profile predicate defined by a UHAT: simulating T on the representative with the arithmetic bounds of Appendix A uses at most $2m \leq 2|T|$ positions, with unused slots excluded by validity flags, and yields an acceptance circuit polynomial in $|T|$. The number of possible profiles is

$$\sum_{j=1}^{m} \binom{m}{j} (j!)^2.$$

Including the empty initial profile, a direct profile automaton has $1 + \sum_{j=1}^{m} \binom{m}{j}(j!)^2$ states, but its explicit table may be much larger than the circuit-based RNN of Theorem 4.1.

Why one dictionary target cannot be assumed for all three masks. Let $x = 0^{2^n}$ and obtain x' by changing only the last payload bit to one. The words $B_x\$B_x$ and $B_{x'}\$B_x$ have the same first-occurrence order: before the changed payload, the addressed dictionary block has already introduced 0, 1, and $\#$. They have the same last-occurrence order because the query block is identical and contains all three symbols after the separator. The first word belongs to D_n and the second does not. Hence D_n is not profile-determined for any $n \geq 1$ and no unmasked UHAT recognizes it exactly.

Proposition D.2 (Verification consequence)**.** *With an explicit variable alphabet, unmasked UHAT non-emptiness is* NP*-complete, and equivalence and universality are* coNP*-complete. For a fixed alphabet these problems are decidable in polynomial time.*

D.3 Proof of Proposition D.2

For non-emptiness, guess a profile and simulate the UHAT on its canonical representative. The word has at most $2m$ symbols and every value has polynomial bit length, giving an NP certificate. For hardness, start with a Boolean formula $F(x_1, \ldots, x_r)$ and use an alphabet with a pair p_i, q_i for each variable. At one leftmost unmasked selection per variable, assign score one to the pair and zero to other symbols; output one if the selected symbol is p_i and zero otherwise. If neither pair member occurs the result is zero. Thus every word induces an assignment, and every assignment is realized by some word. Position-wise Boolean gates evaluate F on the extracted assignment. The network has polynomial size and accepts some word exactly when F is satisfiable.

Non-equivalence or non-universality has a short profile certificate, giving the coNP upper bounds. The satisfiability reduction gives equivalence hardness by comparison with an always-rejecting network. Flipping the final Boolean output reduces unsatisfiability to universality. For fixed m, the

displayed number of profiles is constant and exhaustive simulation is polynomial in $|T|$. These claims concern the two-rule unmasked class; the non-emptiness reduction itself uses only leftmost ties.

Proposition D.3 (A polynomial-scale memory lower bound)**.** *Over an explicitly listed alphabet of $m \geq 2$ symbols, the language in which every symbol occurs has a polynomial-size unmasked UHAT, and every equivalent RNN needs at least m state bits.*

Proof. For each symbol, an unmasked selection maximizing its one-hot indicator tests whether it occurs. A conjunction of these m tests gives a polynomial-size UHAT. For the lower bound, consider prefixes listing subsets of symbols. Distinct subsets S, S' differ in some symbol x; after exchanging their names if necessary, take $x \in S' \setminus S$. A suffix listing $\Sigma \setminus S'$ makes the S' prefix contain every symbol, while the S prefix still lacks x. Hence there are 2^m distinct rows, of which at least $2^m - 1$ have nonempty representatives. Since $\lceil \log(2^m - 1) \rceil = m$ for $m \geq 2$, the RNN requires at least m persistent bits. □

If $C_0(s)$ is the largest minimal RNN size among unmasked descriptions of size at most s over all explicit alphabets, this example and Theorem 4.1 yield the broad bounds $s^{\Omega(1)} \leq C_0(s) \leq s^{O(1)}$ for sufficiently large budgets, using the polynomial-size family and monotonicity. They do not say that attention saves a polynomial factor over an optimal recurrent description. At a fixed alphabet, every such language already has a constant-size explicit profile automaton; the growth statement above allows the explicit alphabet to grow.

The all-later fragment. If every attention operation has mask $j > i$, the last position always retrieves zero. Induction through the layers shows that its final value depends only on the last symbol. Evaluating the network on each one-letter input determines the accepted-symbol set. An RNN stores a single acceptance bit overwritten at each step, with a polynomial-size symbol-decoding circuit. Reversing an attention direction without reversing the readout is therefore not a symmetry of this setup.

E Direct causal compilation

E.1 Proof of Theorem 4.3

For a causal network, every value at position i depends only on the prefix through i. Let V_j be a finite superset of the possible vectors after layer j. The finite-value bound gives $|V_j| \leq 2^{p(|T|)}$ for a fixed polynomial p. We may store the vectors themselves in slots, rather than indices into a hard-wired transition table.

For each attention operation, maintain the distinct lower-layer values previously seen, in first-occurrence order and in last-occurrence order. Each list has at most $|V_j|$ entries, and each entry uses polynomially many bits. At a new input symbol, initialize its embedding and compute the whole current column through the network. For an attention operation, compare its current query against the summary's distinct keys. Among maximal values choose the earliest first occurrence or latest last occurrence as specified by the tie rule. If the summary is empty, choose zero. This reproduces exactly the selected vector of a strict causal attention operation.

Only after that query has been evaluated is the current lower-layer value inserted into the corresponding summary. Append a new value to the first-occurrence list; remove and reappend it in the last-occurrence list. Summaries for different layers are separate, so computing the current column in layer order never introduces position i as an eligible key for its own strict-past query. Position-wise operations are evaluated locally. The final current-column readout is stored as a single Boolean coordinate or recomputed from the final-column registers.

The number of summary entries is at most $2 \sum_j |V_j|$, where j ranges over the lower layers of the attention operations; this is exponential in $|T|$. Equality tests, score comparisons, stable deletion, insertion, and masking of unused entries all admit circuits polynomial in this slot count and the vector precision. Thus both persistent-bit count and description size are polynomial in $|T|$ and $\sum_j |V_j|$, hence $2^{\mathrm{poly}(|T|)}$. This proves Theorem 4.3. Unlike the general NFA route, the proof never enumerates a powerset of an exponential automaton.

What the direct argument adds. The past-LTL monitor in Appendix C already establishes the same tower-height upper bound. The activation construction exposes a separate parameter: how many lower-layer values can occur. Replacing its uniform exponential estimate by the depth recursion gives Theorem 4.4. Neither argument requires a new tiling reduction. Combining the compiler with the PSPACE RNN reachability algorithm also recovers an exponential-space verification upper bound, consistent with the prior causal hardness result.

E.2 Proof of Theorem 4.4

Ignore position-wise maps when counting retrieval depth, since they do not increase the number of possible values. Let V_0 be the image of the symbol embedding. Through the k causal selections, choose finite supersets recursively by applying each update to a current value and either a previous value or the null value. Writing $v_j = |V_j|$, we have

$$v_{j+1} \le v_j(v_j + 1), \qquad v_0 \le |\Sigma|.$$

Induction gives $v_j \le (|\Sigma| + 1)^{2^j} - 1$: if $v_j \le M - 1$, then $v_j(v_j + 1) \le (M - 1)M \le M^2 - 1$. This is the activation-count recursion in Yang et al. (2024, Appendix C.2, Lemma 22); we use it to size the causal summaries before the realized-profile computation. The generic rational value bound separately gives $v_j \le 2^{\mathrm{poly}(|T|)}$. Unreachable values in these supersets are harmless; summaries need only have enough slots for all possible realized values.

Use the direct causal compiler above to compute the current output of this prefix online. In addition, maintain the two-order profile of the emitted V_k-valued sequence. Each summary entry is an actual polynomial-bit rational vector; membership tests and comparisons avoid any assumed succinct encoding of a giant transition table. Total state and transition size are polynomial in $|T|$ and the sum of the value-set bounds, hence at most the expression in Theorem 4.4.

For the remaining unmasked suffix, fix the current realized profile. Lemma D.1 shows that its next operation acts by a common value map on this word. To update the first-occurrence list, apply the map and keep the earliest occurrence of each resulting value. To update the last-occurrence list, apply the map and keep the latest occurrence of each resulting value. Position-wise maps are handled the same way. This reconstructs successive profiles without enumerating the union of values from all possible inputs. The last entry of the final last-occurrence list yields the last-position value and readout.

The entire unmasked suffix is evaluated in the acceptance circuit from the realized profile. It does not affect the online causal summaries. The bound is polynomial when $2^k \log(|\Sigma| + 1) = O(\log |T|)$. Arbitrarily large k can still be expensive even if every layer is causal or if an identity unmasked layer is appended; the shape alone is not a polynomiality guarantee.

F Dictionary realization and the mixed-mask gap

Figure 1 in Section 3 traces the construction below on the accepted input $B_{1000}B_{1010}\$B_{1010}$ with $n = 2$; Table 4 lists its retrievals.

F.1 Dictionary answer maps

Lemma F.1 (Dictionary answer rows)**.** *The language D_n has at least 2^{N_n} distinct dictionary answer maps, and every equivalent RNN needs $b(R) \ge N_n$ persistent bits.*

Let $\mathcal{X} = \{0,1\}^{2^n}$ and $N = |\mathcal{X}| = 2^{2^n}$. Fix an ordering of $\mathcal{X}$, and let w_S concatenate B_x for $x \in S$ in that order. If $S \ne S'$, choose $q \in S \triangle S'$. Exactly one of $w_S\$B_q$ and $w_{S'}\$B_q$ belongs to D_n. In particular, the distinguishing suffix is the same literal word in both comparisons. The 2^N subsets give 2^N different future-answer maps. All but the empty-set prefix are nonempty, and

$$\lceil \log(2^N - 1) \rceil = N \qquad (N \ge 2).$$

Lemma B.3 therefore gives $b(R) \ge N$, including under the positive-word convention.

For each fixed n, the language is regular: a finite automaton can store the set of completed dictionary blocks, its progress through the current cell and block, and its progress through the query. Thus equivalent finite-state recurrent recognizers exist. The question is their required representation size.

F.2 Compact recognition and restricted selections

Proposition F.2 (Polynomial-size dictionary UHAT). *There is $T_n \in \mathsf{UHAT}_M$ of size $\mathrm{poly}(n)$ and retrieval depth $O(n)$ recognizing exactly D_n. It uses causal operations and one unmasked query fetch; block aggregation uses rightmost selection.*

We prove the proposition by specifying the selections, format guard, and block aggregation below, followed by the resource accounting in Appendix F.5.

We use the restricted Boolean RASP selection form of Bergsträßer et al. (2026, Lemma 9 and Appendix A.3). With already materialized Boolean coordinates P_t, a selection searches for a key satisfying

$$S(j) \wedge \bigwedge_{t \in K} \big(P_t(i) \leftrightarrow P_t(j)\big), \tag{3}$$

where S depends only on the source position. It returns the leftmost or rightmost matching key under the specified mask, with an explicit default if no matching key exists. Value expressions are Boolean combinations of query and source coordinates; defaults are Boolean combinations of query coordinates.

For completeness, the following implementation makes both null handling and retrieval count explicit. Materialize S and a constant coordinate $c = 1$ at every position. With $q = |K|$, use score

$$\sum_{t \in K} \big(P_t(i)P_t(j) + (1 - P_t(i))(1 - P_t(j))\big) - (1 - S(j)).$$

Its value is q exactly at matching keys and is smaller at every other key. The prescribed tie rule consequently selects the desired match whenever one exists. The affine value map copies the required source coordinates, including c, S, and the equality coordinates, into slots with zero affine bias. Thus a null selected vector contributes zeros to every copied slot; a separately retained local constant may remain one. Write these copies as $c^{\mathrm{sel}}, S^{\mathrm{sel}}, P_t^{\mathrm{sel}}$ and compute

$$r = c^{\mathrm{sel}} \wedge S^{\mathrm{sel}} \wedge \bigwedge_{t \in K} [P_t(i) = P_t^{\mathrm{sel}}].$$

For an eligible but nonmatching selected key, $r = 0$. If the eligible set is empty, the selected vector is zero, so $c^{\mathrm{sel}} = 0$ and again $r = 0$. Each returned Boolean coordinate is then $rV \vee (\neg r)D$, where V is its value expression and D its default. In particular, the source predicate is retrieved as a materialized coordinate; it is not recomputed on a padded null vector.

One retrieval can return a tuple: its affine value map copies all needed source coordinates once, and position-wise gates compute all output components. This uses the same selected key throughout the tuple. There is no additional retrieval for each component. The score is affine/bilinear, and AND, OR, NOT, and conditional Boolean outputs have affine/ReLU implementations. These facts establish the mask-preserving polynomial compiler needed here directly, including the case $K = \varnothing$.

F.3 An explicit format guard

All address, payload, and matching fields below are anchored at the *completed-cell position*, namely its terminating #. This convention avoids assigning a payload interpretation to an unfinished cell. We first construct a guard for the syntax

$$B_{x_1} \cdots B_{x_r} \,\$\, B_q, \qquad r \geq 0, \tag{4}$$

without yet requiring that q occur in the dictionary.

Local histories. For $t \in \{0, \ldots, n+2\}$, let $H_t(i)$ contain the one-hot symbol at position $i - t$ and a validity bit $u_t(i)$. If $i - t < 1$, all these coordinates are zero. The embedding supplies H_0 with $u_0 = 1$. For $t = 1, \ldots, n+2$, one constant-score, rightmost strict-past retrieval copies $H_{t-1}(i-1)$ into $H_t(i)$ while preserving the existing histories. At $i = 1$, null selection gives the required zero vector. Induction establishes all histories using exactly $n + 2$ causal retrievals.

Write Z_t, O_t, A_t, E_t for the four one-hot coordinates of H_t, corresponding to $0, 1, \#, \$$, respectively; their position argument is omitted when unambiguous. Put $B_t = Z_t \vee O_t$. Define the completed-cell flag

$$C_i = A_0 \wedge \bigwedge_{t=1}^{n+1} B_t \wedge \big(A_{n+2} \vee E_{n+2} \vee \neg u_{n+2}\big).$$

Thus a completed cell ends exactly $n+1$ bits after the preceding cell boundary, separator, or start. At such a position define the address, in most-significant-bit-first order, and the payload by

$$a_i = (O_{n+2-t})_{t=1}^{n}, \qquad v_i = O_1.$$

The same coordinates are defined at other positions, but only completed-cell flags give them semantic force.

Region and preceding cell. One causal source-only existence selection computes

$$Q_i = [\exists j < i : \ w_j = \$].$$

Its return is one and its default is zero. A second causal source-only selection takes the rightmost $j < i$ with $C_j = 1$. It returns a tuple

$$(e_i, Q_i^-, a_i^-) = (1, Q_j, a_j),$$

with the all-zero tuple as default. Hence e_i records whether a previous completed cell exists. Define

$$J_i = e_i \wedge [Q_i = Q_i^-], \qquad Z(a) = [a = 0^n], \quad A(a) = [a = 1^n].$$

The flag J_i says that the current region and the previous completed cell's region agree. Let $\mathrm{Succ}(a^-, a)$ mean $a = a^- + 1 \pmod{2^n}$. It is a Boolean predicate of polynomial size: bit t flips precisely when all less significant bits of a^- are one, and each resulting bit is compared with a_t.

Violation flags. At every position compute bad_i as the disjunction of the following seven Boolean predicates:

$$\begin{aligned}
b_i^{\mathrm{cell}} &= A_0 \wedge \neg C_i, \\
b_i^{\mathrm{run}} &= \bigwedge_{t=0}^{n+1} B_t, \\
b_i^{\mathrm{place}} &= E_0 \wedge \neg(\neg u_1 \vee A_1), \\
b_i^{\mathrm{repeat}} &= E_0 \wedge Q_i, \\
b_i^{\mathrm{address}} &= C_i \wedge \big((\neg J_i \wedge \neg Z(a_i)) \vee (J_i \wedge \neg \mathrm{Succ}(a_i^-, a_i))\big), \\
b_i^{\mathrm{wrap}} &= C_i \wedge Q_i \wedge J_i \wedge Z(a_i), \\
b_i^{\mathrm{end}} &= E_0 \wedge e_i \wedge \neg A(a_i^-).
\end{aligned}$$

In order, these flag an incorrectly delimited cell, a run of $n+2$ bits, a separator outside a cell boundary, a repeated separator, an incorrect first or successor address, an internal query wraparound, and an unfinished dictionary block before the separator. The terminal guard is

$$G_i = C_i \wedge Q_i \wedge A(a_i).$$

The format condition at the last position L is $G_L = 1$ and $\mathrm{bad}_i = 0$ for every $i \le L$. In particular, the internal-query-wraparound flag rejects the *first* successor transition from 1^n to 0^n within the query region. It does not wait for a second such transition.

Correctness of the guard. Every word of the form (4) satisfies these predicates. Conversely, suppose the format condition holds. The terminal flag requires a completed cell after a separator. The repeated-separator flag gives exactly one separator, and its placement flag puts it at the start or immediately after a cell. Every $\#$ closes exactly $n+1$ bits since the previous cell, separator, or start; overlong bit runs are forbidden; the last symbol closes a cell. Consequently the two regions are concatenations of complete cells, with the dictionary possibly empty and the query nonempty.

The first completed cell of either region has $J_i = 0$ and address 0^n. All later cells in the same region have $J_i = 1$ and successive addresses modulo 2^n. If the dictionary is nonempty, its final address is 1^n by the separator guard, so its cells split into whole cycles of 2^n addresses. In the query, a second occurrence of address 0^n would trigger the internal-wraparound flag. Its first address is 0^n and its terminal address is 1^n, so it contains exactly one cycle. This proves that passing the guard is equivalent to (4). The argument includes a missing or repeated separator, a short or overlong cell, an incomplete final cell, an incorrect address order, two query blocks, and an empty dictionary. A word with an empty dictionary can pass the format guard; the membership test below correctly rejects it.

For example, $B_x\$B_xB_x$ is rejected at the first completed cell of the second query block. There $C_i = Q_i = J_i = Z(a_i) = 1$, because the previous completed cell is a query cell of address 1^n. A valid single-block query has $J_i = 0$ at its first cell and nonzero addresses at all later cells, so its internal-wraparound flag is always zero.

F.4 Query retrieval and block aggregation

Use one unmasked restricted selection whose source predicate is $C_j \wedge Q_j$ and whose equality coordinates are the n address bits. Return the source payload, with default zero, and call the result $\widehat{q}_i$. Under the format condition there is exactly one query cell at every address. Therefore every dictionary cell retrieves $\widehat{q}_i = q[a_i]$, independently of the tie rule. Define the dictionary comparison flag

$$f_i = C_i \wedge \neg Q_i \wedge [v_i = \widehat{q}_i].$$

Under the format guard the query selection always succeeds at dictionary cells, so f_i needs no additional success factor. Fields on malformed words cannot cause acceptance because the final decision includes the format guard.

Next select the rightmost strict predecessor satisfying

$$C_j \wedge \neg Q_j \wedge \big(\neg f_j \vee Z(a_j)\big).$$

Return $(e_i^*, a_i^*, f_i^*) = (1, a_j, f_j)$, with an all-zero default. Define

$$m_i = C_i \wedge \neg Q_i \wedge A(a_i) \wedge f_i \wedge e_i^* \wedge Z(a_i^*) \wedge f_i^*. \tag{5}$$

On a well-formed word, a dictionary block endpoint has a strictly earlier start cell because $n \geq 1$. That start is always an eligible matching source, and lies after every cell of earlier blocks. If any interior cell before the endpoint mismatches, the rightmost such mismatch is selected instead of the start, and its nonzero address makes (5) false. Otherwise the start is selected. The selected f_i^* detects a mismatch at the start, and the current f_i detects a mismatch at the endpoint. Thus $m_i = 1$ precisely at the endpoint of a dictionary block equal to the query. Earlier blocks cannot interfere, and query cells cannot mark themselves as dictionary matches.

Two final causal existence selections compute

$$M_i = [\exists j < i :\ m_j = 1], \qquad V_i = [\exists j < i :\ \mathrm{bad}_j = 1].$$

Accept at the last position exactly when

$$G_L \wedge \neg\, \mathrm{bad}_L \wedge \neg V_L \wedge M_L = 1. \tag{6}$$

Every matching dictionary endpoint precedes the final query cell, so the strict mask in M_L loses no relevant match. The explicit bad_L includes the current violation. If the dictionary is empty, every m_i is zero and acceptance fails. The format lemma and block-matching argument prove both directions of $L(T_n) = D_n$.

F.5 Retrieval and binary-description accounting

There are exactly $n + 8$ retrieval operations: $n + 4$ causal operations, the query fetch, and three causal operations. In particular, there is exactly one unmasked retrieval. The violation predicates are already determined before the query fetch; retaining their coordinates through subsequent operations preserves their meaning. Source predicates, equality tests, defaults, and returned tuples are implemented by the preceding restricted-selection construction, so none hides an additional retrieval.

Table 4: The retrievals in the dictionary construction. Boolean computations between rows require no additional retrieval. A tuple return shares one selected source.

Operation	Number	Source test / equalities	Mask
Predecessor histories	$n+2$	constant score, rightmost	causal
Region flag Q_i	1	source is \$	causal
Previous completed cell	1	C_j, rightmost	causal
Query fetch	1	$C_j \wedge Q_j$; equal addresses	unmasked
Block aggregation	1	$C_j \wedge \neg Q_j \wedge (\neg f_j \vee Z(a_j))$, rightmost	causal
Match existence M_i	1	m_j	causal
Violation existence V_i	1	bad_j	causal

Here is a conservative explicit description bound. Histories and returned tuples use $O(n)$ coordinates. Expanding all Boolean tests, including successor tests, without relying on sharing requires at most $O(n^2)$ Boolean gates. Retaining every gate output therefore permits width $O(n^2)$ and $O(n^2)$ affine/ReLU operations, in addition to the $n+8$ retrievals. A dense affine or attention map at this width has $O(n^4)$ scalar entries. All gates use integer coefficients, and the score constants and sums have magnitude polynomial in n, hence $O(\log(n+2))$-bit encodings. Thus

$$|T_n| = O\big(n^6 \log(n+2)\big)$$

is a sufficient conservative bound, including dense zero entries, the four-symbol embedding, readout, masks, tie rules, and architecture encoding. Tighter polynomial degrees are unnecessary here. For fixed n, the construction and its source description are independent of the number of dictionary blocks and the total input length. This proves Proposition F.2.

Corollary F.3 (Automata consequences)**.** *The dictionary language requires at least 2^{N_n} DFA states and at least N_n NFA states. Together with the compact mixed source and inherited upper constructions, the general UHAT gap is triply exponential against DFAs and doubly exponential against NFAs.*

A DFA has one active state; an NFA can be simulated by a set of active states. The DFA count is one exponential above the recurrent bit bound: an NFA with Q states has at most 2^Q such subsets. The NFA double-exponential tower was already established by Bergsträßer et al. (2026, Theorem 17); the dictionary yields the larger deterministic-state requirement. The language-specific causal/mixed separation is Corollary 4.5.

F.6 Proof of Theorem 3.2 and Corollaries F.3 and 4.5

For every equivalent RNN, the row bound gives

$$|R| \geq b(R) \geq 2^{2^n},$$

while the construction above has polynomial source description. Its explicitly growing history construction gives a family of unbounded source sizes, as required by Definition 2.3. Thus the family witnesses a doubly exponential succinctness lower bound in that definition. The general translation in Appendix C supplies the matching bounded-expansion tower. No step converts arbitrary circuit recurrences into a particular named neural cell; the state lower bound already applies to every more restricted finite-state recurrent architecture.

The 2^N distinct rows require at least $2^N = 2^{2^n}$ DFA states. An NFA with Q states has at most 2^Q determinized subset states, so it requires $Q \geq N = 2^{2^n}$. Conversely, the inherited general translation gives a doubly exponential NFA upper bound and a triply exponential DFA upper bound. These comparisons concern explicit automata representations. The NFA lower tower was already supplied by the prior shortest-word witness; the dictionary construction supplies the stronger deterministic-state lower tower.

Finally, if a causal UHAT U recognizes D_n, Theorem 4.3 gives an RNN of size at most $2^{p(|U|)}$ for a fixed polynomial p. The dictionary lower bound forces $p(|U|) \geq 2^n$, hence $|U| \geq 2^{\Omega(n)}$. Together with Proposition F.2, this proves Corollary 4.5.

G Approximation, length, and finite compression

G.1 Proof of Theorem 3.3

Write $N = N_n$ for the binary dictionary task. Let M be the boundary state and let ρ expose all input-independent random coins to both encoder and decoder. Giving the decoder these coins can only strengthen it. Since M has at most 2^b possible values, its conditional entropy is at most b. The vector X remains independent of ρ. For each coordinate i, let ε_i be the error of the decoder when the query is i. Binary Fano's inequality gives

$$H(X_i \mid M, \rho) \le h_2(\varepsilon_i).$$

Here H denotes Shannon entropy in bits. Conditional subadditivity and concavity of h_2 imply

$$\begin{aligned} N = H(X \mid \rho) &\le H(M \mid \rho) + H(X \mid M, \rho) \\ &\le b + \sum_{i=1}^{N} H(X_i \mid M, \rho) \\ &\le b + \sum_{i=1}^{N} h_2(\varepsilon_i) \le b + N h_2(\varepsilon). \end{aligned}$$

Rearrangement proves the binary statement. Subadditivity needs no conditional independence of the membership bits given (M, ρ), and concavity applies to the average error, so the individual ε_i need not be at most ε or $1/2$. More formally, if only an upper bound on average error is given, the monotonicity of h_2 on $[0, 1/2]$ gives the displayed bound with that upper bound. Private random coins are included in ρ, so the argument covers private randomness as well. No information depending on the prefix may be exposed as a free public variable.

G.2 Finite answer alphabets

Proposition G.1 (Finite-answer random access)**.** *Let $X \in [V]^N$ have independent uniform coordinates, with $V \ge 2$, and let the query index be independent and uniform in $[N]$. A b-bit boundary state answering X_I with average error at most $\varepsilon \le 1 - 1/V$ satisfies*

$$b \ge N\big(\log V - h_2(\varepsilon) - \varepsilon \log(V-1)\big).$$

The bound allows private or input-independent public randomness. For $V = 1$ the answer is constant and requires no stored information.

For $X \in [V]^N$ uniform, $H(X \mid \rho) = N \log V$ and Fano gives

$$H(X_i \mid M, \rho) \le h_2(\varepsilon_i) + \varepsilon_i \log(V-1).$$

The same summation and concavity yield the V-ary expression. The uncertainty term $h_2(\varepsilon) + \varepsilon \log(V-1)$ is increasing on $[0, 1-1/V]$, which explains the stated range. This is the classical random-access-code converse; compare Nayak (1999, Theorem 2.3) for pointwise binary decoding success. The proof here explicitly handles the stated classical average-error and randomness conventions, and the V-ary extension. It is also consistent with the rate–distortion viewpoint (Shannon, 1959).

G.3 Sparse dictionaries and prefix length

Proposition G.2 (Sparse dictionaries and prefix length)**.** *An exact boundary state for all dictionaries of r distinct blocks needs at least $\log \binom{N_n}{r}$ bits. For $1 \le r \le \sqrt{N_n}$ this is $\Omega(r2^n) = \Omega(\ell/n)$, where $\ell = r\ell_n$ is the prefix length. For every fixed $0 < \varepsilon < 1/2$, public-coin one-way membership with error at most ε for every size-r dictionary and every query, with probability over the public coins, has optimal communication $\Theta(r)$ for $1 \le r \le N_n/2$. This is a boundary protocol bound, excluding a complete parser's implementation cost.*

Sparse dictionaries have exactly r present blocks, so their possible membership tables form a smaller set than in the dense distribution. Counting these tables gives the exact bound; indexing and hashing give the randomized communication bound, as proved below. This randomized bound uses worst-case error over dictionary/query pairs, unlike the dense average-error criterion: under a uniform query, always answering "absent" already has error r/N_n.

Proof. Write $N = N_n$. Restrict the dictionary rows to subsets of exactly r of the N possible blocks. They remain pairwise distinguishable, so $b \geq \log \binom{N}{r}$. The standard binomial estimate gives

$$\log \binom{N}{r} \geq r \log(N/r) = r(2^n - \log r).$$

For $r \leq \sqrt{N} = 2^{2^{n-1}}$, this is at least $r2^{n-1}$. Since the dictionary length is $\ell = r2^n(n+2)$, it is at least $\ell/[2(n+2)] = \Omega(\ell/n)$.

In the randomized statement, the error bound holds for each fixed size-r set and each fixed query, over input-independent public randomness. For the lower bound, partition $2r$ universe elements into r disjoint pairs and restrict each dictionary to contain exactly one element from every pair. The r choices encode an arbitrary r-bit vector, and a membership query to a designated element of a pair retrieves its bit. The classical bounded-error one-way indexing lower bound is $\Omega(r)$ (Kremer et al., 1999). It also follows by applying Theorem 3.3 to these r independent bits at constant error.

For the matching public-coin upper bound, choose a public random hash $h : [N] \to [M]$ with $M = \lceil r/\varepsilon \rceil$ and store a bit vector marking the hash buckets of the set elements. A present query is always accepted; a fixed absent query collides with a set element with probability at most $r/M \leq \varepsilon$. The vector takes $O(r/\varepsilon)$ bits, hence $O(r)$ at constant error. Standard multi-hash variants give $O(r \log(1/\varepsilon))$ bits. The randomness describing a universal hash can be public in this communication model. This upper bound does not price a complete streaming parser, address computation, or a succinct implementation of an arbitrary public hash on the long block encodings. The exact whole-language RNN bound and the boundary protocol bound must not be conflated. □

G.4 The exact finite communication relaxation

If an encoder may send any one of 2^b messages after seeing all of $X \in \{0,1\}^N$, a deterministic decoder assigns one reconstructed bit vector to each message. A nearest-neighbour encoder is optimal for a fixed reconstruction codebook. Therefore the best mean query accuracy is

$$\mathrm{Acc}^*(N,b) = 1 - \frac{1}{N2^N} \min_{\substack{C \subseteq \{0,1\}^N \\ |C| \leq 2^b}} \sum_{x \in \{0,1\}^N} \min_{c \in C} d_H(x,c), \tag{7}$$

where d_H counts differing bit positions. Randomization cannot improve this average optimum: fixing the coins of any randomized scheme achieves at least its expected accuracy for some choice of coins. This is finite Hamming quantization, a classical source-coding problem (Shannon, 1959).

At $b = N$, an online machine can attain zero error by storing the entire incidence vector and setting the appropriate bit when a dictionary block is read; additional parsing state is separate. At intermediate budgets an optimal codebook in (7) need not admit an online implementation with the same persistent-bit budget. Thus the curve is an information-theoretic upper envelope on accuracy at a boundary budget, not an asserted achievable RNN tradeoff for the full language.

H Causal depth, fixed alphabets, and binary encoding

H.1 Depth witness statements and scope

Theorem H.1 (Fixed-alphabet depth witness)**.** *For $\Sigma_\star = \{\mathsf{D}, \mathsf{Q}\} \times \{0,1\}^4$ and each $k \geq 1$, there is a UHAT W_k with k causal retrievals followed by one unmasked retrieval, width $O(2^k)$, and description size $2^{O(k)}$, whose language requires 2^{2^k} recurrent bits. Its canonical inputs carry parity and ruler-clock fields, and its construction uses rightmost selection.*

Each causal operation doubles an assembled window. At a final query endpoint, one retrieval tests membership among 2^{2^k} candidate windows. Equal-length dictionary prefixes permit a common query suffix, establishing the state bound. The clock fields are supplied in input symbols rather than computed from absolute position. This double exponential is in k: because the witness description grows as $2^{O(k)}$, it is distinct from the polynomial-in-n dictionary construction in the main text.

Corollary H.2 (Binary encoding and the depth threshold)**.** *A binary-alphabet family retains the same bit lower bound with $k + 13$ causal retrievals followed by one unmasked retrieval, and size $2^{O(k)}$. The encoding retains the clock fields. For fixed alphabet, the upper bound is polynomial at $k \le \log\log|T| + O(1)$; padded families above that scale can require superpolynomial size whenever the underlying construction fits the size budget.*

The following subsections give the doubling invariant, the length-14 binary code, and the exact padding condition. The binary lower bound needs correct simulation on encoded witness strings, without a claim to recognize exactly the encoded language on malformed strings. Replacing the final unmasked retrieval by a causal one preserves the canonical witness behavior and gives a causal-only variant with one additional causal operation. Under the retrieval-depth convention of Appendix A, the k doubling operations and the 13 decoding operations are retrievals; affine-only layers belong to the interleaved position-wise computation.

H.2 An address-carrying warm-up

Before proving the fixed-alphabet result, it is useful to see the doubling construction with explicit addresses. Fix $q \ge 2$, let $h = 2^k$, and take symbols $(\rho, i, a) \in \{\mathsf{D}, \mathsf{Q}\} \times [h] \times [q]$. A block for $x \in [q]^h$ lists $(\rho, 1, x_1), \ldots, (\rho, h, x_h)$. At doubling step j, an endpoint with index divisible by 2^j selects the rightmost strict predecessor whose index is its own minus 2^{j-1}. One-hot index fields implement this score. The selected and current lower-level windows are the two adjacent halves of the new window. Rightmost selection chooses the occurrence in the current block, even if earlier blocks have the same index.

After k such operations, every block endpoint carries its whole word. A final membership selection at the query endpoint tests dictionary endpoints. All subsets of $[q]^h$ can be distinguished, giving q^{2^k} state bits. The parameter description is polynomial in $k, 2^k, q$, but the alphabet grows with k. The fixed-alphabet construction replaces these explicit index fields by two single-bit schedule fields.

H.3 The ruler identity

For an integer $i \ge 1$, write $i = 2^{\nu(i)}(2u_i + 1)$, where $\nu(i)$ is the exponent of its largest power-of-two divisor, and define

$$t_i = \mathbf{1}[i \text{ is even}], \qquad s_i = u_i \bmod 2.$$

The latter is the binary digit immediately above the lowest set bit of i. Its first sixteen values are $0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0$.

Lemma H.3 (Ruler identity)**.** *For $i = 2^j r$ with $j, r \ge 1$, $s_{i-2^{j-1}} = 1$ exactly when r is even, equivalently when 2^{j+1} divides i.*

Proof. $i - 2^{j-1} = 2^{j-1}(2r - 1)$, whose odd part is $2(r-1) + 1$. Hence the relevant ruler bit is $(r - 1) \bmod 2$, which is one exactly for even r. □

These bits are ordinary input fields on the canonical witness strings. The network does not compute arbitrary absolute parity from a position-free unary string. The statement is therefore compatible with the logical limitations on positional-encoding-free hard attention.

H.4 Proof of Theorem H.1

Put $h = 2^k$, $\mathcal{X} = \{0,1\}^h$, and $M = 2^h$. A symbol of $\Sigma_\star$ has fields (ρ, p, t, s, d): dictionary/query region, presence, parity, ruler, and data. For each subset $S \subseteq \mathcal{X}$, its prefix P_S contains one length-h dictionary slot for every $x \in \mathcal{X}$, in a fixed lexicographic order. A slot's data are the bits of x and its presence field is $\mathbf{1}[x \in S]$ at every cell. At absolute position i, its clock fields are (t_i, s_i). All prefixes therefore have the same length Mh.

For $y \in \mathcal{X}$, the suffix Q_y is one length-h query slot with data y, presence zero, and clock fields at positions $Mh + 1, \ldots, Mh + h$. Because the prefix lengths agree, Q_y is literally the same word for every S. This is needed for the prefix-distinguishability argument; a suffix allowed to depend on the prefix length would not suffice.

Doubling invariant. The embedding retains the five fields and a constant coordinate, an initial alignment flag one, and padded windows of ruler and data bits. Before operation j, let $a = 2^{j-1}$. At each position i divisible by a, the alignment flag is one and the value stores the reversed length-a window

$$(s_i, d_i), (s_{i-1}, d_{i-1}), \ldots, (s_{i-a+1}, d_{i-a+1}).$$

All other positions have alignment zero. Query with a constant and give key score $2\alpha - 1$, where α is the key's old alignment flag. For an aligned position $i \geq 2a$, rightmost strict-past selection returns exactly the aligned predecessor $i - a$. Concatenating its window with the current one forms the length-$2a$ window.

For the first operation set the new alignment to $\mathrm{ReLU}(\alpha + t_i - 1)$. For later operations set it to

$$\mathrm{ReLU}(\alpha + s_{i-a/2} - 1),$$

using a coordinate already in the current half-window. Lemma H.3 says this is one exactly when $2a$ divides i. At the first possible aligned endpoint $i = a$, there is no earlier aligned endpoint, but the new gate is zero: $t_1 = 0$ in the first operation and $s_{a/2} = 0$ thereafter. Any arbitrary selected lower-alignment value is discarded there. At position one the null-selection convention supplies zero. An old-unaligned position stays unaligned. Windows may be gated to zero at unaligned positions using Boolean-controlled ReLU gates, so every relevant coordinate remains bounded. This proves the invariant, including the boundary cases.

After k operations, every multiple of h contains its full block's data. A valid dictionary key is an aligned dictionary endpoint with presence one. At the last query endpoint, score a candidate by

$$H \ \mathrm{valid}(x) + \sum_{r=1}^{h} (2y_r - 1)(2x_r - 1), \qquad H > 2h. \tag{8}$$

All data coordinates are Boolean, so the correlation is in $[-h, h]$. Thus every valid key beats every invalid key and valid keys are ordered by Hamming distance to y. A Boolean comparison of the selected data with y, conjoined with the selected validity bit, accepts exactly when some present slot equals y. If no valid key exists the validity test forces rejection. The score is bilinear after adding a constant coordinate, and all gates have affine/ReLU implementations.

We have constructed $W_k(P_S Q_y) = \mathbf{1}[y \in S]$. There are 2^M equal-length nonempty prefixes, so Lemma B.3 gives $b \geq M = 2^{2^k}$. Width is $O(h)$, the k dense doubling maps use $O(kh^2)$ scalar parameters, and coefficients and score bonuses need at most $O(k)$ bits. For instance $O(k^2 4^k)$ is a sufficient bit-description bound, hence $2^{O(k)}$. Theorem 4.4 gives $2^{O(2^k)} \, \mathrm{poly}(|W_k|)$ on the fixed alphabet, matching the dependence on k in the exponent. This proves Theorem H.1.

A causal-only variant. Replace the last unmasked membership selection by the causal mask. On every canonical witness the valid dictionary endpoints precede the final query endpoint, so the membership decision is unchanged. The resulting network has $k + 1$ causal retrieval operations and the same persistent-bit lower bound. Its language away from the canonical witnesses need not be specified for this argument: the hard prefix–suffix submatrix already belongs to its full language. This supplies a fixed-alphabet witness for the causal tower height as well.

H.5 Binary simulation in Corollary H.2

Identify the 32 source symbols with five-bit strings $a = a_1 \cdots a_5$. Use the length-14 code

$$c(a) = 111 \, \phi(a_1) \cdots \phi(a_5) \, 0, \qquad \phi(0) = 00, \quad \phi(1) = 01. \tag{9}$$

The encoded payload begins and ends with zero and has no adjacent ones. Consequently the marker 111 occurs exactly at codeword starts in a concatenation of codewords, never inside a payload or across a boundary. This explicit marker is the synchronization mechanism; the construction does not need a general coding theorem.

Thirteen causal predecessor operations recover the length-14 bit window ending at each position. Use a constant score and rightmost selection, and concatenate the current bit with the retrieved history. A validity history distinguishes padding before the first symbol if needed. Arrange the

recovered window in chronological order and write $x = (w_{i-13}, \ldots, w_i)$ for binary input w, padding unavailable positions with zero. For each of the 32 codewords z, the gate

$$\mathrm{ReLU}\left(1 - \sum_{\substack{r\in[14]\\ z_r=0}} x_r - \sum_{\substack{r\in[14]\\ z_r=1}} (1 - x_r)\right)$$

is one exactly when the window equals z. The marker property means that, on encoded strings, such a match occurs exactly at a true codeword endpoint and reveals its source symbol. Leading padding cannot imitate a codeword because a codeword begins with three ones. Let e_i be the resulting endpoint flag.

Simulate W_k at these endpoints. For each source layer j, bound all source-score magnitudes on the bounded reachable coordinates by B_j, and, at key position r, add a bonus $C_j e_r$, where $C_j > 2B_j$. The bound also includes the zero simulation vector used at interior bit positions. The bonus is an ordinary affine/bilinear term using a constant query coordinate. Every eligible endpoint then beats every interior position; all endpoints get the same bonus, preserving score order and ties. After the simulated update, gate interior simulation values to zero. For a nonnegative coordinate $z \le U$, the gate $\mathrm{ReLU}(z - U(1 - e_i))$ does this; signed coordinates can be stored in positive and negative parts.

If a causal query has no earlier endpoint, every selectable interior position carries simulation value zero, matching the source null value. The current endpoint is excluded by the strict mask. In the final unmasked operation, endpoints include precisely all source positions. Induction therefore proves exact simulation on every correctly encoded source word; the final bit position is the final source endpoint.

The decoder uses 13 additional causal retrieval operations, and the simulation uses k more followed by one unmasked operation. Its width remains $O(2^k)$ and its description $2^{O(k)}$. The encoded prefixes $c(P_S)$ retain equal lengths and $c(Q_y)$ remains a common suffix, so the 2^{2^k}-bit lower bound persists. The hard submatrix alone proves the lower bound. The five source fields, including both clocks, remain in the codeword payload.

H.6 The size–depth threshold and its scope

For fixed Σ, the upper bound is polynomial whenever $2^k = O(\log s)$, where $s = |T|$. Equivalently, every fixed additive constant in $k \le \log\log s + O(1)$ gives a polynomial guarantee, possibly with a different polynomial degree.

Conversely, let $f(s) \to \infty$ and choose

$$k(s) = \lfloor \log\log s + f(s) \rfloor,$$

provided the unpadded construction's $2^{O(k(s))}$ description fits within the budget s. Pad by unused coordinates or other syntactic identity data, without adding retrieval operations, to a description of order s. Then

$$2^{2^{k(s)}} \ge s^{\,2^{f(s)-1}},$$

which is superpolynomial. For the binary family, the fixed additional 13 operations only shift the depth threshold by a constant. This is an existential worst-case statement about the explicit syntactic budget; it does not show that every deep network is hard or that the padded size is a minimal representation of its language.

Even without padding, $|W_k| = 2^{O(k)}$ and $b \ge 2^{2^k}$ show that no polynomial simulation bound holds uniformly over all unbounded k. Depth alone is not sufficient to force a lower bound: redundant or constant operations can be arbitrarily deep. The witness uses doubling, rightmost retrieval, and a suitable family of input answer maps.