ATR^{++}: Ambiguous Multi-hop Binding as Pointer-Chasing, with Mechanistic Certificates

Liz Lemma Future Detective

January 18, 2026

Abstract

State space models and other subquadratic mixers can match Transformers on perplexity yet often fail at robust retrieval. Recent work showed that on Associative Recall (AR) and a hierarchical variant Associative Treecall (ATR), different architectures reach similar accuracies while implementing fundamentally different mechanisms (induction/bind-then-read vs direct retrieval at the last state), revealed by causal interchange interventions. We push this program into the 2026 setting by introducing ATR^{++}: a controlled synthetic benchmark for ambiguous and multi-hop binding. ATR^{++} extends ATR with (i) explicit mention disambiguation (coreference-like ambiguity), (ii) recursion (unbounded structure under truncation), and (iii) k-hop ancestor queries (compositional retrieval). We give a formal reduction from ATR^{++} to pointer-chasing, proving an expressivity upper bound for k-layer Transformers and a lower bound for a large class of linear-time streaming models without sufficient state capacity. Finally, we define a mechanistic leaderboard that requires not only accuracy but also causal signatures measured by interchange interventions, enabling architecture comparisons that are explanatory and predictive of generalization.

Table of Contents

4. 4. Mechanistic Evaluation Protocol: interchange interventions; definition of causal-signature vector $\mathsf{Sig}(M)$; what constitutes bind-then-read vs direct-retrieval signatures; leaderboard rules.

5. 5. Upper Bound Constructions: explicit Transformer circuit for k-hop ancestor retrieval under ambiguity; depth dependence $\Theta(k)$; discussion of dimension and positional encoding requirements.

6. 6. Lower Bounds for Direct-Retrieval Streaming Models: reduction to pointer-chasing; memory/state dimension lower bounds; implications for SSM-like models without adequate local mixing / scratch space.

7. 7. Experimental Design (Optional but Strengthening): architecture suite (attention, Based-like, Mamba-like, hybrids, memory tokens), ablations (kernel size, depth, PE schemes), and mechanistic-vs-behavioral analyses.

8. 8. Discussion: what the bounds imply about architectural primitives; when local mixing helps; how ATR^{++} can be extended (multi-query, tool-call analogs); limitations.

9. 9. Conclusion: ATR^{++} as a mechanistic benchmark with theory-backed separations; invitation to mechanistic leaderboard participation.

1 Introduction

We study a distinction that has become operationally important for evaluating sequence models: *retrieval* versus *binding.* By retrieval we mean selecting and reproducing a previously observed token type (or a function thereof) from an input sequence. By binding we mean constructing and later using a *relation* between two occurrences—e.g. that a marked occurrence v stands in a parent relation to some other occurrence in an implicit tree—so that a subsequent query can request information about a bound partner. In practice, high accuracy on retrieval-heavy benchmarks need not imply that a model forms stable bindings that compose under ambiguity and recursion; conversely, a model that forms bindings may fail if evaluation confounds binding with incidental positional or lexical cues.

A substantial body of recent evaluation has used next-token prediction tasks designed to elicit in-context retrieval behavior (e.g. copying, associative recall, and simple key–value patterns). These tasks are valuable because they yield clean, analyzable circuits in attention-based models and because they can be scaled synthetically. However, by 2026 the empirical landscape has changed: models trained at scale often solve classical retrieval tasks via multiple qualitatively different strategies, including heuristics that are not robust to distributional shift. Two limitations recur. First, many retrieval datasets admit *shortcuts* in which relative position features (distance to the query, recency, or fixed formatting) carry nontrivial mutual information with the answer; a model can then succeed without representing the intended relation. Second, standard associative recall and its descendants tend to be *unambiguous*: the key identifying the answer is unique in the context, so the task does not force an explicit choice among repeated terminal types or multiple candidate referents. In such regimes, it is difficult to separate mechanisms that genuinely bind entities from mechanisms that merely retrieve a nearby span.

The ATR family of tasks was an important step toward binding evaluation by introducing tree-structured dependencies and ancestor queries. Nevertheless, in the form most commonly studied, ATR retains a rightmost-mention bias: when terminal types do not repeat and the query refers to a unique surface form, a heuristic that selects the most recent matching token can approximate the intended retrieval. Moreover, if the data distribution ties the location of the referent v or its ancestor $\mathrm{anc}_k(v)$ to predictable offsets, then one can solve many instances with an effectively local read. Our goal is to define a task family that (i) compels nontrivial binding under ambiguity, (ii) supports clean algorithmic upper bounds for models known to implement explicit pointer-jumping, (iii) admits matching lower bounds for broad classes of linear-time direct-retrieval architectures, and (iv) enables mechanistic evaluation that distinguishes these solution classes causally rather than descriptively.

We therefore define ATR$^{++}(\Pi)$, a synthetic binding-and-retrieval task parameterized by a recursive PCFG G, an ambiguity model $\mathcal{A}$, and a hop count k. A sampled instance consists of a document d (a yield of G augmented by $\mathcal{A}$), a divider EOS, and a query that names a particular terminal occurrence v via explicit referent markers and requests the terminal type of $\mathrm{anc}_k(v)$ in the latent tree T induced by the derivation. The ambiguity model $\mathcal{A}$ forces repeated terminal types and ensures that referent identification cannot be reduced to matching a unique token type; the grammar recursion produces variable-depth, variable-length instances. Crucially, we impose an anti-shortcut design constraint: under the data distribution, superficial positional statistics of v and $\mathrm{anc}_k(v)$ are constructed (and then empirically checked) to have negligible mutual information with the answer y. As a result, high accuracy requires learning a procedure that follows bindings rather than exploiting stable offsets.

The task family is designed to support two complementary forms of analysis. The first is algorithmic: we exhibit an explicit Transformer construction that solves ATR^{++} exactly by implementing pointer-jumping. Informally, the query representation selects the marked occurrence v; then each subsequent layer performs one hop by attending from the current node representation to a parent "anchor" position determined by the head-terminal property of the grammar. After k layers the residual stream encodes $\mathrm{anc}_k(v)$, and the output head reads out its terminal type. Formally, for $M = |\Sigma|$ we construct a depth $L = \Theta(k)$ one-head Transformer with embedding dimension $d = \tilde{O}(M)$ that achieves zero error on all well-formed instances. This construction is not merely existential; it provides a reference mechanism against which trained models can be compared.

The second form of analysis is lower-bounding and separation: we identify a broad class of "direct-retrieval" streaming models that process the document left-to-right with a bounded-dimensional state $s_t \in \mathbb{R}^d$ and must answer after reading the query. Such models include many RNN/SSM-like architectures and also hybrids that add bounded-radius local mixing but do not store token-indexed external memory. We show that, on a distribution over ATR^{++} instances that encodes pointer-chasing, any one-pass streaming algorithm with memory $m = o(M)$ incurs constant error for fixed hop count k. Under standard simulation assumptions (bounded precision and Lipschitz updates), this yields a dimension lower bound $d = \Omega(M)$ for streaming direct-retrieval models to achieve small error. This separation formalizes an empirical observation: linear-time architectures often excel at recency-based retrieval yet struggle on compositional binding problems where the relevant dependency may be far from the query and may require multiple nonlocal hops.

Beyond behavioral accuracy, we require a mechanistic criterion. A model might match the Transformer upper bound behaviorally while realizing a different internal computation; conversely, it might partially solve the task via a brittle heuristic. We therefore introduce a mechanistic evaluation protocol based on interchange interventions. Given a clean example o and a corrupted counterpart c (created by altering a binding-relevant marker or head token), we swap internal activations at designated sites f from the clean run into the corrupted run, yielding $M^{f \leftarrow f^*}(c, o)$. We score each site by likelihood restoration $\mathrm{Attrib}(f)$, and we aggregate these scores into a mechanistic signature vector $\mathsf{Sig}(M)$. The key property is that the explicit bind-then-read Transformer circuit concentrates causal dependence at hop-specific anchors, whereas direct-retrieval streaming solutions (when they exist on easier distributions) concentrate dependence near the query and late tokens. Thus $\mathsf{Sig}(M)$ is intended to be predictive of length extrapolation and compositional generalization, not merely a post hoc interpretability artifact.

In summary, our contributions are threefold: (i) a task family $\mathrm{ATR}^{++}(\Pi)$ that enforces ambiguity, recursion, and anti-shortcut constraints while remaining synthetically controllable; (ii) matching-style upper and lower bounds separating global-attention pointer-jumping from broad classes of linear-time direct-retrieval models as a function of depth and memory; and (iii) a causal, intervention-based signature that operationalizes the distinction between binding mechanisms and retrieval heuristics. The remainder of the paper situates these results relative to prior work, details the constructions and reductions, and validates that the mechanistic signatures align with generalization outcomes across in-distribution and out-of-distribution splits.

2 Related Work

Associative recall, in-context retrieval, and induction heads. A large fraction of mechanistic work on sequence models has focused on synthetic tasks in which the correct output is a function of a small number of *retrieval* operations: copying a token, selecting a value given a key, or composing a short chain of such selections. In these settings one can often isolate relatively simple attention-based circuits (notably so-called *induction heads*) that implement a form of pattern completion by attending from the query to a previous occurrence of a matching prefix and then shifting attention forward to retrieve the subsequent token **??**. These analyses clarify how global attention can realize algorithmic behavior at test time, and they provide a vocabulary for causal testing via activation patching. At the same time, the canonical associative-recall distributions are typically *unambiguous* (keys appear once) and frequently admit recency or positional heuristics; consequently, high behavioral accuracy does not by itself certify that the model has learned a stable *binding* relation that composes under ambiguity and recursion. Our task family is designed to preserve the analyzability of synthetic retrieval benchmarks while explicitly removing several common shortcut channels.

Tree-structured retrieval and the ATR lineage. The ATR family introduced a qualitatively richer setting by placing the relevant dependency in a latent tree and querying ancestors, thereby forcing a multi-step computation that is naturally expressed as pointer-following **?**. This shift is important: the intended solution is no longer "retrieve the value paired with the key," but "identify a node and follow a structural relation k times." Such tasks connect directly to algorithmic primitives (pointer-jumping) for which Transformers have clean constructions. However, in the most widely used formulations, the surface form of a referent can be unique or nearly unique, and terminal types need not repeat, which reintroduces a rightmost-mention or nearest-match heuristic. Moreover, unless care is taken in data generation, the yield positions of v and $\mathrm{anc}_k(v)$ can correlate with the answer in ways that a local model can exploit. We view ATR^{++} as an attempt to retain the essential tree-structured query semantics while strengthening the distributional conditions so that the binding relation, rather than a positional proxy, carries the information needed to answer.

Binding, entity tracking, and memory in language models. The binding problem we isolate is closely related to entity tracking and coreference in natural language: a model must maintain an association between a *marker* (a mention, a name, an index) and an underlying referent, and later answer a query that depends on that association. Empirically, attention-based models can represent some entity-state information in the residual stream and can route it through attention in ways that resemble explicit memory access **??**. There is also a long line of work on augmenting sequence models with external memories or neural caches to improve retrieval and factual consistency **??**. Our setting is deliberately more austere: we do not provide an explicit key–value store or supervision of intermediate states, and we require correctness under repeated terminal types and explicit referent markers. This places the focus on whether the model constructs a *composable internal representation of bindings* rather than on whether it can opportunistically retrieve a nearby span.

State-space and recurrent models: empirical strengths and retrieval limits. Linear-time architectures—RNN variants, convolutional models, and more recent state-space models (SSMs)—are often competitive on long sequences precisely because they avoid the quadratic cost of global attention. Many such models exhibit strong performance on local pattern matching, recency-weighted retrieval, and tasks where the answer depends on a small window of recent tokens. Yet a recurrent update $s_t = F(s_{t-1}, e(x_t))$ with $s_t \in \mathbb{R}^d$ is, from an algorithmic standpoint, a streaming algorithm with bounded memory. This viewpoint motivates formal limitations: when the answer depends on selecting one item among M possibilities with adversarially arranged dependencies, a state of dimension $d \ll M$ cannot, in general, retain sufficient information. Prior empirical studies have reported degradation of SSM/RNN-like models on long-range copying and multi-step retrieval under distribution shift **??**. Our contribution is to tie these observations to a concrete pointer-chasing reduction embedded in ATR^{++}, yielding a dimension lower bound under standard bounded-precision assumptions.

Causal interventions, activation patching, and causal abstraction. Mechanistic interpretability has increasingly emphasized *causal* rather than merely correlational analyses. Techniques such as causal tracing, activation patching, and path patching intervene on internal activations to test whether particular components mediate a model's behavior on a specific input **??**. The interchange-intervention formalism (swapping activations between a clean and a corrupted run) is especially well suited to synthetic tasks because it yields controlled counterfactuals with known ground-truth structure **??**. In parallel, the causal abstraction literature asks when a high-level algorithm (e.g. pointer-jumping on a latent tree) can be mapped onto a low-level neural computation via an abstraction map that preserves counterfactual behavior **?**. We adopt this perspective in defining $\mathsf{Sig}(M)$: rather than claiming that a trained model "uses binding" because a head attends to a certain position, we require that appropriate site-level swaps *restore* the correct answer under a targeted corruption. This allows us to separate mechanisms that merely correlate with the answer from mechanisms that are causally responsible for it.

Pointer-chasing and streaming lower bounds from communication complexity. The hardness backbone of our lower bound is classical. Pointer-chasing (iterating a function f for k steps) and indexing are canonical problems exhibiting separations between models with random access and one-pass streaming algorithms with limited memory. Communication complexity provides sharp tools: a single-pass streaming algorithm with memory m induces a one-way (or two-party) protocol with communication m, and known lower bounds then imply that m must scale with the domain size M to achieve small error **??**. Similar arguments appear in analyses of online computation and in lower bounds for bounded-width branching programs. Our use of these results is conceptually straightforward but technically useful: by embedding a random instance of pointer-chasing into the latent bindings of ATR^{++}, we obtain a distribution on which any direct-retrieval streaming model with $d = o(M)$ fails with constant probability. This establishes a principled barrier for linear-time architectures without token-indexed memory and motivates the mechanistic comparison to explicit pointer-jumping circuits in Transformers.

3 ATR^{++} Task Family

We define a family of synthetic binding-and-retrieval tasks, denoted ATR$^{++}(\Pi)$, where an instance consists of a *document* generated by a recursive probabilistic context-free grammar (PCFG) together with an explicit *query* asking for a k-hop ancestor in a latent tree. The parameter tuple is

$$\Pi \;=\; (G, \mathcal{A}, k, n_{\max}, \text{split spec}),$$

where G is a PCFG over nonterminals N and terminals Σ with locally normalized rule probabilities $p(\cdot \mid A)$ for each $A \in N$; $\mathcal{A}$ is an ambiguity model controlling repetition of terminal types and insertion of referent markers; $k \in \mathbb{N}$ is the hop count; $n_{\max}$ is a length cap; and split spec describes out-of-distribution (OOD) evaluation conditions.

PCFG generation and the head-terminal induced tree. We sample a derivation from G starting at a start symbol S, recursively expanding nonterminals until all leaves are terminals or a truncation criterion triggers, and we read off the yield as a terminal sequence of length $n \le n_{\max}$. For mechanistic analyses we require that G satisfy a *head-terminal property*: each production $A \to \alpha$ designates a unique head terminal occurrence in the yield of α (e.g. via an annotated head child and a head terminal within that child). This induces a rooted ordered tree T over *terminal occurrences* $\{v_1, \ldots, v_n\}$, where each v_i is a node and the parent pointer is defined by head propagation through the parse. We write $\mathrm{anc}_k(v)$ for the k-hop ancestor of a node v, obtained by applying the parent pointer k times; if the ancestor does not exist, we map to a designated null terminal $\texttt{NULL} \in \Sigma$ (this convention can be toggled, but we fix it for concreteness).

Ambiguity model and referent markers. A central requirement of ATR^{++} is that surface tokens do not uniquely identify nodes. We therefore introduce an ambiguity model $\mathcal{A}$ that operates on the generated yield and produces an *augmented document* over $\Sigma \cup$ markers. Concretely, $\mathcal{A}$ enforces two properties.

- **Repeated terminal types.** Terminal types repeat frequently: we either sample terminals from a small active subset of Σ (so collisions are common) or apply controlled relabeling to ensure that multiple distinct occurrences share the same terminal type. This removes any heuristic that relies on terminal identity being a proxy for node identity.
- **Unique referent identification.** We insert explicit markers that uniquely name a *particular occurrence* v among all occurrences of the same type. For example, we may attach an occurrence identifier $\mathrm{id}(v) \in \{1, \ldots, n\}$ as a pair of bracket tokens around the terminal at v, or emit

a separate marker token in a nearby position whose semantics is "this marker refers to occurrence v." The specific marker vocabulary is fixed across the dataset and is treated as part of the input alphabet but not part of Σ (the answer vocabulary).

The crucial point is that the query can refer to v only through these markers, while the answer depends on the latent binding structure in T. By construction, simple strategies such as selecting the rightmost mention of a terminal type, or selecting a nearest matching token in the yield, are information-theoretically insufficient when $\mathcal{A}$ produces multiple confusable mentions with distinct markers.

Instance format and k-hop ancestor queries. An instance is formed by concatenating the augmented document, a divider EOS, and a query sequence. The query encodes (i) the referent marker naming a node v, and (ii) the hop count k. We denote the full input by

$$x \;=\; d \;\|\; \mathsf{EOS} \;\|\; q(v,k),$$

and the target label is

$$y \;=\; \text{type}(\text{anc}_k(v)) \in \Sigma,$$

where type(u) returns the terminal type at occurrence u and returns `NULL` if u is undefined. The model is evaluated only on the final next-token prediction at the end of the query. We emphasize that the query is *not* a natural-language question; it is a compact, machine-readable encoding that isolates the binding computation.

Anti-shortcut constraints. The distribution is designed so that positional and recency cues do not carry reliable information about y. In particular, we enforce an anti-shortcut invariant: features such as the absolute position of v, the absolute position of anc$_k(v)$, or their relative displacement in the yield have high variance and (approximately) zero mutual information with the answer under the data distribution. Practically, we achieve this by (a) allowing recursion and variable arity so that the same structural relation can occur at many yield distances, (b) randomizing local expansion choices in G so that yield positions are not rigidly tied to tree depth, and (c) sampling query nodes across the document rather than concentrating them near the end. We treat this constraint as a *data-generation specification* and validate it empirically by measuring correlations and mutual-information estimates on large samples; failures trigger adjustments to G and $\mathcal{A}$ (e.g. increasing repetition, widening depth variability, or changing marker placement).

Evaluation splits and OOD generalization. We evaluate models both in-distribution and under principled distribution shifts that target the intended algorithmic content.

- **Held-out bindings.** We partition a subset of parent–child terminal-type pairs (or more generally, head-terminal bindings) and ensure that some bindings are absent from training but present at test time. This tests whether a model has learned a compositional procedure for following pointers, rather than memorizing frequent type transitions.
- **Held-out structures.** We hold out particular production-rule patterns or subtrees (e.g. specific expansions of a nonterminal, or particular depth-2 templates) while keeping the terminal vocabulary fixed. This targets generalization across latent tree shapes and discourages reliance on superficial regularities of the training grammar.
- **Length extrapolation.** We train on documents with $n \leq n_{\text{train}}$ and test on $n \in (n_{\text{train}}, n_{\text{max}}]$, preserving the same generation process. Since $\text{dist}(v, \text{anc}_k(v))$ in the yield can grow with n, this split probes whether the mechanism scales with sequence length.

For each split, we keep the query semantics fixed (always k-hop ancestor retrieval) so that success requires invariance of the learned computation rather than adaptation to a new task.

Remarks on scope. We stress that ATR^{++} is intended to be *mechanistically discriminative*: the same behavioral objective admits qualitatively different internal solutions (explicit pointer-jumping versus compressed direct retrieval), and our generation constraints aim to eliminate degenerate heuristics that would otherwise blur this distinction. In the next section we specify an intervention-based protocol that operationalizes this goal by measuring where, in a model's computation, the binding information is written and later read.

4 Mechanistic Evaluation Protocol

Our behavioral objective fixes what must be computed, but not *how* it is computed internally. We therefore accompany accuracy with a mechanistic evaluation that is tailored to ATR$^{++}(\Pi)$: we (i) construct counterfactual corruptions that selectively break the binding computation while leaving most of the document unchanged, (ii) perform interchange interventions that swap internal activations between clean and corrupted runs, and (iii) summarize the resulting causal attributions into a compact signature vector $\mathsf{Sig}(M_\theta)$. The protocol is designed so that the explicit pointer-jumping Transformer construction in Section 5 yields a distinctive "bind-then-read" signature, whereas one-pass "direct-retrieval" mechanisms concentrate causal influence near the end of the sequence.

Clean instances and structured corruptions. Let (x, y, T, v) be an instance sampled as in Section 3, with $x = d \| \mathsf{EOS} \| q(v, k)$. We define two corruption operators that are binding-relevant by construction. First, a *referent-marker corruption* changes the marker that identifies v in either the document or the query (depending on the marker scheme), producing a corrupted input $c_{\mathrm{ref}}(x)$ that now refers to a different occurrence $v' \neq v$ while keeping all terminal types in Σ unchanged. Second, a *head-anchor corruption* changes a single binding-critical token that participates in the induced parent-pointer encoding (e.g. a head-terminal anchor whose type is later read out), producing $c_{\mathrm{head}}(x)$ that preserves the referent marker but perturbs the latent pointer-chase outcome. In both cases we choose the corruption position uniformly from a pre-specified set of eligible markers/anchors so that surface cues (position, recency) are not predictive of which corruption was applied. We denote a generic corruption by $c(x)$ and the clean input by $o = x$.

Interchange interventions and likelihood restoration. Fix a model M_θ and a set of intervention sites SITES, where each site f identifies a particular internal activation tensor at a particular layer and token position (e.g. residual-stream vector at layer ℓ and position t; or a designated sub-activation such as a mixer output, when exposed by the implementation). For each example we run the model on the clean input o and the corrupted input c, recording the activation at site f on both runs, denoted $f(o)$ and $f(c)$. We then define the *interchange-intervened* run $M_\theta^{f \leftarrow f(o)}(c)$ to be the model executed on c with the activation at f replaced by the clean activation $f(o)$, leaving all other activations as in the corrupted computation. This produces a next-token distribution at the end of the query, hence a probability assigned to the correct label y.

We quantify the causal contribution of site f to solving the corrupted instance via a normalized likelihood-restoration score,

$$\mathrm{Attrib}(f) \;=\; \mathbb{E}\left[\frac{\log p_\theta(y \mid M_\theta^{f \leftarrow f(o)}(c)) - \log p_\theta(y \mid M_\theta(c))}{\log p_\theta(y \mid M_\theta(o)) - \log p_\theta(y \mid M_\theta(c))}\right],$$

where the expectation is over instances and over the corruption choice (and we discard degenerate cases where the denominator is numerically close to 0). Intuitively, $\mathrm{Attrib}(f) \approx 1$ means that swapping only site f from clean to corrupted nearly restores the clean likelihood of the correct answer, while $\mathrm{Attrib}(f) \approx 0$ indicates that the site is not on the causal path by which the corruption harms performance. We emphasize that this is a *causal* diagnostic under the intervention semantics: it distinguishes correlation from necessity in the model's computation on these counterfactuals.

Choice of sites. The set SITES must be rich enough to cover both hypothesized mechanism classes while remaining model-agnostic. Concretely, we include (i) all token positions in the query (including the hop-count encoding), (ii) all positions containing referent markers in the document, and (iii) a grammar-dependent but data-computable set of candidate "anchor" positions in the document (e.g. terminals or marker-adjacent terminals), which are the only plausible locations where a bind-then-read mechanism could store intermediate bindings. We additionally stratify by layer, so each (ℓ, t) pair is a distinct site. The protocol never uses the latent tree T to select sites at evaluation time; site selection depends only on the observable tokenization and marker scheme. This restriction prevents trivially "probing" the answer by direct access to T, and ensures that high attribution at a site must arise from the model's own internal organization of computation.

Causal-signature vector. A single scalar $\mathrm{Attrib}(f)$ is informative only relative to other sites and other examples. We therefore aggregate into a fixed-dimensional summary statistic

$$\mathsf{Sig}(M_\theta) \in \mathbb{R}^D,$$

computed by pooling $\mathrm{Attrib}(f)$ across (layer, position, site-type) groups. A minimal instantiation that suffices for our separations includes: (1) *anchor concentration*: the maximum and mass of Attrib over document-anchor sites versus query sites; (2) *layer-of-write*: the distribution of the argmax layer among high-attribution anchor sites; (3) *hop sensitivity*: how the attribution pattern shifts as k varies; and (4) *stability under OOD*: the same pooled statistics computed separately on held-out bindings, held-out structures, and length-extrapolation splits, reported as divergences from in-distribution values. In practice we treat $\mathsf{Sig}(M_\theta)$ as a vector of these pooled quantities (means, maxima, and quantiles), enabling simple downstream classification of mechanism class.

Bind-then-read versus direct-retrieval signatures. We operationalize two mechanism classes by where causal influence localizes. A *bind-then-read* mechanism is one in which the model writes a representation of the relevant binding (or intermediate pointer) into the residual stream at specific document positions, and later reads it at the query. Under this mechanism, corrupting the referent marker or a head anchor disrupts a specific write location; swapping the activation at that document anchor from the clean run should restore the answer, yielding a signature with (i) large Attrib at a small set of document-anchor sites, often at layers aligned with hop count, and (ii) comparatively small Attrib at query-only sites. Conversely, a *direct-retrieval* mechanism compresses document information into a running state such that the decisive information is only accessible near the end (or in query-adjacent positions) rather than stored at token-indexed anchors. This predicts (i) negligible attribution at document-anchor sites and (ii) attribution concentrated at late layers and query positions (or, in streaming models, at the final state-update sites). The point of ATR^{++} is that both strategies are behaviorally plausible in principle, yet they entail sharply different interchange-intervention profiles.

Leaderboard and reporting rules. To make results comparable, we fix (a) the dataset generator Π including marker scheme, corruption distributions, and OOD splits; (b) a reference SITES specification expressed in terms of layers and observable token types; and (c) evaluation metrics. Submissions report (i) accuracy and calibrated log-loss on each split, (ii) the full $\mathsf{Sig}(M_\theta)$ with confidence intervals over random seeds, and (iii) the per-site $\mathrm{Attrib}(f)$ heatmaps for reproducibility. Models may not use privileged access to T or derivation traces at training or test time, and may not introduce auxiliary supervision that explicitly encodes parent pointers; all supervision is via next-token prediction of y. For mechanistic claims, the intervention code must execute the stated activation swaps (rather than approximations via gradient-based proxies), and must evaluate on the fixed corruption operators above.

With this protocol in place, we can state and verify a sharp mechanistic prediction: the explicit Transformer construction that exactly solves ATR^{++} implements pointer-jumping by writing hop-wise bindings to designated anchors, and therefore yields a bind-then-read Sig. We now give the construction and analyze its depth and dimension requirements.

5 Upper-Bound Constructions: Pointer-Jumping with a 1-Head Transformer

We fix a hop count k and exhibit an explicit decoder-only Transformer of depth $L = \Theta(k)$ and embedding dimension $d = \tilde{O}(M)$ that solves $\mathrm{ATR}^{++}(\Pi)$ exactly (in the sense of correct argmax prediction) on all well-formed instances generated under Π. The construction implements the intended latent computation—k successive parent-pointer applications in the induced terminal-occurrence tree T—by a layerwise pointer-jumping circuit.

Observable encoding of bindings. We assume (as permitted by Π) that the ambiguity model $\mathcal{A}$ inserts constant-vocabulary markers that make two pieces of information observable in the token stream: (i) a unique *occurrence key* for each terminal occurrence v_i, and (ii) a *parent key* specifying the key of its parent $\mathrm{par}(v_i)$ in T.[1] Separately, $\mathcal{A}$ designates the referent occurrence v by a distinguished marker (or an occurrence key repeated in the query) so that the query uniquely identifies v despite repeated terminal types.

Let the key space have size B. We choose Π so that $B \geq n_{\max}$ and hence the keys used within a document are collision-free. When $B = \tilde{O}(M)$ (e.g. by setting $n_{\max} \leq \tilde{O}(M)$), our final dimension bound is $d = \tilde{O}(M)$; otherwise the same construction yields $d = \tilde{O}(M + B)$.

Embedding layout. We decompose the residual space as a direct sum

$$\mathbb{R}^d \;=\; \mathbb{R}^{d_{\text{key}}} \oplus \mathbb{R}^{d_{\text{par}}} \oplus \mathbb{R}^{d_{\text{type}}} \oplus \mathbb{R}^{d_{\text{ctrl}}}, \qquad d_{\text{key}} = d_{\text{par}} = B, \quad d_{\text{type}} = M, \quad d_{\text{ctrl}} = O(1).$$

For each occurrence key $b \in [B]$ we fix a basis vector $\kappa(b) \in \mathbb{R}^{d_{\text{key}}}$, and similarly for parent keys in $\mathbb{R}^{d_{\text{par}}}$. For each terminal type $a \in \Sigma$ we fix a basis vector $\tau(a) \in \mathbb{R}^{d_{\text{type}}}$. At the token position corresponding to occurrence v_i with terminal type a_i, occurrence key b_i, and parent key $p_i = b_{\mathrm{par}(i)}$, we set the (pre-activation) residual input to contain

$$h_i^{(0)} \;=\; \big(\kappa(b_i),\, \kappa(p_i),\, \tau(a_i),\, 0\big).$$

At the query position t_{qry}, we embed the referent key b_v in the $\mathbb{R}^{d_{\text{key}}}$ subspace and set a control bit in $\mathbb{R}^{d_{\text{ctrl}}}$ indicating that this position is the unique *workspace* token whose state will be updated across layers:

$$h_{t_{\text{qry}}}^{(0)} \;=\; \big(\kappa(b_v),\, 0,\, 0,\, \mathbf{1}\big).$$

We include standard positional encodings if desired, but choose all projection matrices below to ignore the positional subspace, so positional information is not used for correctness.

One hop per layer via attention. We work with a simplified pre-norm decoder block containing a single attention head and an MLP sublayer which we set to 0 (the identity map), since the computation is purely associative. In layer $\ell \in \{1, \dots, k\}$ we implement the update

$$\text{current key } b^{(\ell-1)} \;\mapsto\; \text{parent key } b^{(\ell)} = b_{\mathrm{par}(\cdot)}.$$

Let Q_ℓ, K_ℓ, V_ℓ be the query/key/value projections and O_ℓ the output projection. We choose Q_ℓ to extract the key-subspace component at the workspace token, and we choose K_ℓ to extract the occurrence-key component at all document tokens. We choose V_ℓ to extract the parent-key component at all

[1]Concretely, one may realize this with paired markers adjacent to each terminal occurrence, e.g. `[ID=`b_i`]` `[PAR=`$b_{\mathrm{par}(i)}$`]` `a` for terminal type a. The lower bounds in Section 6 use the same observable encoding to embed pointer-chasing instances; our upper bound simply shows that global attention can exploit it with depth $\Theta(k)$.

document tokens, and O_ℓ to write the attended parent key back into the key subspace of the workspace token. On non-workspace tokens we enforce $Q_\ell h_i^{(\ell-1)} = 0$ (e.g. by using the control bit), so only the workspace position produces a nontrivial query.

Because keys within a document are unique, the dot product $\langle Q_\ell h_{t_{\text{qry}}}^{(\ell-1)}, K_\ell h_i^{(\ell-1)} \rangle$ is maximized at the unique position i whose occurrence key equals the current key $b^{(\ell-1)}$. By scaling the projections by a factor $\alpha = \Theta(\log n_{\max})$, we ensure that the softmax weight on this matching position is $1 - \eta$ with $\eta \leq n_{\max}^{-c}$ for any fixed constant $c > 0$. Thus the residual update at the workspace token satisfies

$$h_{t_{\text{qry}}}^{(\ell)} = h_{t_{\text{qry}}}^{(\ell-1)} + \big(\kappa(b^{(\ell)}), 0, 0, 0\big)$$

up to leakage η into other basis directions; we will maintain a large margin so that leakage cannot flip the final argmax.

Reading out the ancestor type. After k hop layers, the workspace token contains (approximately) the key of $\text{anc}_k(v)$. We add a final readout layer $\ell = k+1$ whose attention again matches the workspace key against document occurrence keys, but whose values extract the terminal-type basis $\tau(a_i)$ rather than the parent key. That is, V_{k+1} projects onto $\mathbb{R}^{d_{\text{type}}}$, while Q_{k+1}, K_{k+1} are as before. The resulting workspace activation contains $\tau(a_{\text{anc}_k(v)})$ with arbitrarily small contamination. The unembedding matrix W_{out} is chosen to map $\tau(a)$ to a logit vector with a fixed margin $\gamma > 0$ between the correct label a and all others; choosing α large enough relative to γ makes the correct label the unique argmax for all instances.

Correctness by induction and depth dependence. Let $b^{(0)} = b_v$ and $b^{(\ell)}$ denote the workspace key after ℓ hop layers. By construction of V_ℓ, the unique matching document position for key $b^{(\ell-1)}$ contributes parent key $b^{(\ell)} = b_{\text{par}(\text{anc}_{\ell-1}(v))}$, hence $b^{(\ell)}$ equals the key of $\text{anc}_\ell(v)$. An induction on ℓ yields that after k hop layers the workspace key identifies $\text{anc}_k(v)$, and the final readout layer returns its terminal type, which is exactly the label y. The circuit uses one layer per pointer application and hence requires $L = k + 1 = \Theta(k)$ depth; this linear dependence is the natural cost of composing k successor (parent) operations in the absence of an explicit iterative loop.

Dimension and positional requirements. The dimension requirement is transparent in this explicit encoding: we need M directions to represent terminal types in Σ and B directions to represent collision-free binding keys and parent keys. Under the parameter regime $B = \tilde{O}(M)$ (equivalently $n_{\max} \leq \tilde{O}(M)$ under our chosen keyspace), we obtain $d = \tilde{O}(M)$ as claimed. No special positional encoding is required for correctness beyond the causal constraint that the query token appears after the document; indeed, our projections may be chosen to be invariant to position, so that the model relies only on marker-defined identity and the encoded parent pointers rather than any recency heuristic.

6 Lower Bounds for Direct-Retrieval Streaming Models

We now formalize a class of "direct-retrieval" mechanisms and show that, on an appropriate distribution over $\mathrm{ATR}^{++}(\Pi)$ instances, any such mechanism implemented in a single left-to-right pass requires state dimension $\Omega(M)$ to achieve vanishing error. The proof proceeds by an explicit reduction from pointer chasing (equivalently, iterated indexing) and is therefore insensitive to training data size: the obstacle is representational rather than statistical.

Streaming direct-retrieval model class. Fix an input $x = (x_1, \dots, x_T)$ consisting of the document tokens, the divider EOS, and the query tokens. A streaming model maintains a state $s_t \in \mathbb{R}^d$ and updates it causally as

$$s_t \;=\; F(s_{t-1}, e(x_t)), \qquad t = 1, \dots, T,$$

for some embedding map $e : \Sigma \cup \text{markers} \to \mathbb{R}^d$ and update F (possibly depending on a fixed set of parameters and the layer index in a depth-L stack). The prediction is obtained from s_T and the query encoding, e.g. via a readout $\hat{y} = \arg\max_{a \in \Sigma} \langle w_a, s_T \rangle$. This abstraction covers RNNs and many SSM-like architectures at inference time, including cases in which the per-token computation is linear time and does not preserve a token-indexed key–value cache. The salient limitation is that the model does not retain a separate addressable memory per token; all information about the prefix must be compressed into s_t.

To state a lower bound that is robust to real-valued computation, we impose a standard bounded-precision (or, equivalently, finite-information) assumption: the effective memory capacity of the state is $m = O(d)$ words over an alphabet of size $\mathrm{poly}(M)$, or $O(d \log M)$ bits. This is satisfied by typical implementations with fixed-precision parameters and activations, and is the regime in which communication-complexity reductions apply.

Hard distribution via pointer chasing. We instantiate Π so that the observable markers (inserted by $\mathcal{A}$) expose, for each terminal occurrence v_i, a collision-free occurrence key $b_i \in [B]$ and its parent key $p_i = b_{\mathrm{par}(i)}$. The query specifies a referent key b_v and hop count k, and the label is the terminal type of $\mathrm{anc}_k(v)$. We consider a distribution over documents in which the induced parent-pointer structure encodes a random function $f : [B] \to [B]$ by setting $p_i = f(b_i)$ for a selected subset of occurrences, and by arranging the document so that all pairs $(b, f(b))$ are present exactly once as marker pairs. The query then asks for $f^{(k)}(b_v)$ (followed by a final lookup to map a key to a terminal type). The anti-shortcut constraints in Π are enforced by randomizing yield positions and by repeating terminal types; thus, any method that relies on recency, absolute position, or unambiguous surface forms fails on average, and the only reliable signal is the marker-defined pointer structure.

In this construction, solving ATR^{++} on the hard distribution requires computing an iterated function value $f^{(k)}(x)$ for a uniformly random $x \in [B]$. This is precisely the pointer-chasing problem, known to have linear memory (or communication) requirements when $k \geq 2$ and the function is random.

Reduction to a two-party protocol. We reduce a hypothetical accurate streaming solver to a low-communication protocol, contradicting standard lower bounds. Partition the stream into two contiguous segments: a prefix that contains the encoding of the function f (i.e. all the $[\text{ID} = b][\text{PAR} = f(b)]$ pairs), and a suffix that contains EOS and the query (the referent key x and hop count k). In the corresponding two-party setting, Alice receives the prefix and Bob receives the suffix. Alice simulates the streaming model up to the boundary and sends Bob the memory contents needed to continue the simulation; under our bounded-precision assumption, this message has size $O(m)$. Bob resumes the simulation on his suffix and outputs $\hat{y}$.

If the streaming model achieves error ϵ on the hard distribution, then this protocol computes pointer chasing with the same error using communication $O(m)$. By known randomized communication lower bounds for pointer chasing / iterated indexing, any such protocol with error bounded away from $1/2$ requires $\Omega(B)$ communication for constant k. Therefore $m = \Omega(B)$, and since $m = O(d)$ up to logarithmic factors, we obtain $d = \Omega(B)$. Under the parameter regime in which $B = \Theta(M)$ (or more generally $B = \tilde{\Theta}(M)$), this yields $d = \Omega(M)$, matching the qualitative separation suggested by the upper bound in Section 5.

Implications for SSM-like architectures. The conclusion is not that streaming models are universally incapable of binding retrieval, but that any *one-pass* solution without an addressable memory must allocate $\Omega(M)$ state dimension to store enough information about the random pointer structure. In particular, increasing depth L at fixed $d = o(M)$ does not help on this distribution: although additional layers can implement more complex per-token updates, they do not change the fact that the entire document must be compressed into s_T before the query is read, whereas the query may require retrieving one of B mutually-incoherent pieces of information (the next pointer out of the current key) repeatedly for k steps.

This lower bound also clarifies the role of "local mixing" hybrids. Suppose the model maintains a state s_t but also allows a bounded-radius operator of window size r (e.g. a convolution) that mixes only nearby tokens before updating s_t. Such mixing can improve constant factors and ease optimization, but it does not create a token-indexed scratch space: the memory passed forward remains $O(d)$. Consequently the $\Omega(M)$ memory requirement persists for distributions that encode random pointers over $[B]$. Separately, if one considers purely local architectures that do *not* even maintain a global state but instead propagate information through depth (e.g. bounded-window attention or convolutions without external memory), then worst-case instances in which $\mathrm{dist}(v, \mathrm{anc}_k(v)) = \Omega(n)$ force depth $L = \Omega(n/r)$ by a cone-of-influence argument, formalizing a different obstruction.

Summary. On the hard ATR^{++} distribution, the task reduces to computing $f^{(k)}(x)$ for a random f. Any architecture whose inference can be simulated by a single-pass streaming algorithm with $m = O(d)$ effective memory must satisfy $d = \Omega(M)$ to achieve small error. This furnishes a principled barrier for direct-retrieval mechanisms and motivates the mechanistic distinction tested by our intervention-based signature: a model that succeeds at small d must, in effect, materialize bindings at identifiable sites (as in the bind-then-read circuit), rather than compressing the entire document into a fixed-size state and attempting to retrieve on demand at the query.

7 Experimental Design (Optional but Strengthening)

We now specify an experimental suite whose purpose is twofold: (i) to test behavioral generalization on $\mathrm{ATR}^{++}(\Pi)$ across the in-distribution and the prescribed OOD splits, and (ii) to adjudicate *mechanism class* (bind-then-read versus direct retrieval) via interchange interventions and the resulting causal signature $\mathsf{Sig}(M_\theta)$. The design emphasizes architectural comparability and explicit ablations that isolate the primitives implicated by the upper and lower bounds.

Architecture suite. We consider four families of models, all trained autoregressively on the same next-token objective with the answer token as the supervised target.

(A) Global-attention Transformers. We train standard decoder-only Transformers with $L \in \{2, 4, 8, 16\}$, hidden size d, and either one attention head (to match the clean theoretical construction) or a small number of heads to test robustness. We include variants with full attention and with restricted attention windows, the latter serving as a controlled interpolation toward local-only computation. For each model we record attention maps

on the query tokens and on designated marker tokens to facilitate qualitative checks, but our primary mechanistic evidence will be intervention-based rather than attention-based.

(B) Streaming "SSM-like" models. We instantiate one-pass state models in the style of modern SSMs, including Mamba-like selective scan blocks and Based-like linear-time mixers. Concretely, each block updates a state $s_t \in \mathbb{R}^d$ with a causal recurrence; we ensure inference is linear time and that no token-indexed cache is preserved. To prevent conflating recurrence with explicit attention, we disallow cross-token dot-product attention except in explicitly defined hybrid variants below.

(C) Hybrid local-mixing + state models. To test whether bounded-radius mixing can close the gap without global read mechanisms, we include hybrids that apply a radius-r operator (e.g. depthwise convolution or limited-window attention) prior to the recurrent update. We sweep $r \in \{0, 4, 16, 64\}$ (with $r = 0$ reducing to the pure streaming class). Where appropriate, we also vary whether mixing is applied to embeddings only, within each block, or both.

(D) Memory-augmented controls. Since our bounds separate "no addressable memory" from mechanisms that effectively materialize bindings, we include controlled augmentations: (i) a small number m_{mem} of learned "memory tokens" prepended to the document in a Transformer, (ii) a recurrent model equipped with an explicit key–value table of size $O(m_{\text{mem}})$ with learned addressing, and (iii) a hybrid that permits a tiny global-attention read only on the query suffix. These controls clarify whether performance changes are attributable to representational capacity per se or specifically to an addressable storage primitive.

Matching compute and capacity. To make architectural comparisons meaningful, we report results under two matching schemes: (i) matched parameter count (within $\pm 5\%$), and (ii) matched training compute (total floating-point operations to reach a fixed token budget). For streaming models, we account for per-token costs of mixers and gates; for Transformers, we distinguish full attention from windowed attention so that n^2 versus n scaling is explicit. We also include a small "overprovisioned" regime (larger d) to verify that any observed failures are not merely underparameterization artifacts.

Data regimes and OOD splits. We generate datasets by sampling $\text{ATR}^{++}(\Pi)$ with fixed $|\Sigma| = M$ and varying $(k, n_{\max})$. We train on a base distribution with $n \leq n_{\text{train}}$ and evaluate on: (i) *held-out bindings* (disjoint subsets of marker keys and/or head-terminal pairings), (ii) *held-out structures* (production rules or subtrees excluded from training), and (iii) *length extrapolation* (documents with $n \in (n_{\text{train}}, n_{\text{test}}]$). We additionally vary ambiguity strength in $\mathcal{A}$ (degree of terminal-type repetition and density of referent markers) to ensure that success cannot be attributed to accidental unambiguity. In all cases we verify the anti-shortcut constraints empirically by measuring that simple baselines using relative/absolute positions, recency, or unigram heuristics achieve chance-level performance.

Ablations. We perform targeted ablations that correspond to the hypothesized computational bottlenecks.

Depth versus hop count. For each architecture we sweep L at fixed k and sweep k at fixed L. For Transformers we expect a sharp transition around $L \approx k$ in the clean setting (modulo constant overhead for parsing the query); for streaming models we expect that increasing L at fixed d yields limited benefit once the state bottleneck dominates.

Local mixing radius. For hybrid models we sweep r and record both in-distribution accuracy and extrapolation accuracy. This ablation tests whether improvements, when they occur, track the receptive-field growth predicted by cone-of-influence considerations, and whether any gains persist when $\mathrm{dist}(v, \mathrm{anc}_k(v))$ is deliberately made large by construction.

Positional encoding schemes. Because the task is defined to suppress positional shortcuts, we treat positional encoding (PE) as a potential confounder for both optimization and mechanism. We compare learned absolute PE, sinusoidal PE, RoPE, ALiBi, and a no-PE control (where permitted by the model). We report not only accuracy but also mechanistic signatures: a model that solves the task by a binding circuit should exhibit stable $\mathsf{Sig}(\cdot)$ across PE choices, whereas a spurious position-dependent strategy (when it exists) should collapse under PE perturbations or OOD splits.

Marker dependence. We vary the marker vocabulary size B and the explicitness of the referent encoding in the query (e.g. single key token versus a structured tuple). We also introduce controlled corruptions that change only a marker token while preserving all terminal types, and conversely corruptions that change a head-terminal type while keeping markers fixed. These ablations isolate whether models truly track the marker-defined binding graph.

Mechanistic versus behavioral evaluation. Behavioral accuracy alone is insufficient to distinguish bind-then-read from direct retrieval when both succeed in-distribution. We therefore compute $\mathsf{Sig}(M_\theta)$ using interchange interventions as follows. For each example we construct a clean run o and a corruption c that perturbs exactly one binding-relevant token (marker or head anchor) so that the correct answer changes. We cache internal activations at a predefined set of sites SITES (layers $\times$ positions, and when available, sub-ports such as mixer outputs). For each site $f \in$ SITES we form $M^{f \leftarrow f^*}(c, o)$ and compute likelihood restoration and the corresponding Attrib(f). Aggregating across examples yields (i) a layer-of-write profile (where restoration concentrates), (ii) an anchor-versus-query attribution ratio, and (iii) a hop-alignment score measuring whether the most restorative sites correspond to the predicted hop anchors.

We then compare $\mathsf{Sig}(M_\theta)$ to two reference signatures: the explicit upper-bound circuit (bind-then-read) and a streaming baseline trained to best effort (direct retrieval). Operationally, we cluster models by $\mathsf{Sig}(\cdot)$ (e.g. via cosine similarity) and test whether clusters predict OOD generalization, especially length extrapolation. This addresses a concrete empirical question suggested by the theory: does mechanistic class, as measured by causal attribution patterns, forecast robustness better than in-distribution accuracy?

Reporting and robustness. We report mean and standard error over multiple random seeds (data generation and initialization). For mechanistic quantities we also report concentration (e.g. top-p mass of Attrib over sites) and stability across splits. Finally, we include negative controls: interventions at irrelevant positions (random tokens, non-binding markers) should yield Attrib ≈ 0, and intervention effects should localize to the specific corrupted binding component. These controls ensure that $\mathsf{Sig}(\cdot)$ is not an artifact of global sensitivity but reflects structured causal dependence aligned with the binding computation.

8 Discussion

Our upper and lower bounds isolate a small set of architectural primitives that suffice (and, in certain regimes, are necessary) for reliable binding-based retrieval. Theorem 1 shows that global attention can implement a literal pointer-jumping algorithm: at the query, we identify the referred occurrence v via markers, and then each subsequent layer performs one hop along the parent pointer in the latent tree T until we reach $\mathrm{anc}_k(v)$. Mechanistically, the key property is not "attention" per se, but the availability of an *addressable, token-indexed store* whose contents can be accessed by content-based addressing with $O(1)$ depth overhead per hop. In contrast, Theorem 2 formalizes a barrier for one-pass state updates $s_t = F(s_{t-1}, e(x_t))$ that do not maintain such an addressable store: when the instance distribution encodes essentially random pointer structure over $M = |\Sigma|$ types, any streaming direct-retrieval approach with $m = O(d) = o(M)$ memory must incur constant error on k-hop queries. In this sense, ATR^{++} separates "remember the whole document in a compressed state" from "materialize bindings so that they can be chased."

We view this separation as conceptually analogous to the difference between (i) computing a function of the entire prefix and (ii) supporting *random access* into a set of intermediate facts indexed by markers. The upper-bound circuit does not require that the model store the document verbatim; rather, it must store enough local information at many token positions so that later queries can locate the correct predecessor. This is precisely what the anchor-centric causal signature in Proposition 4 detects: if bindings are materialized at anchors, interchange interventions at the anchors restore the answer; if bindings are "compressed away" into a final state, restoration concentrates near the query. We emphasize that this mechanistic distinction can matter even when both model classes achieve high in-distribution accuracy. In particular, a model that succeeds via a bind-then-read computation should, by construction, degrade primarily when depth is insufficient for the hop count k (or when marker parsing fails), whereas a direct-retrieval solver is expected to degrade when the effective memory load increases (e.g. larger M, stronger ambiguity $\mathcal{A}$, or more adversarial pointer structure), even at fixed k.

Theorem 3 clarifies when *local mixing* can and cannot compensate for the lack of global read. If the only cross-token interaction per layer has radius r, then information can propagate at speed $O(r)$ tokens per layer; hence exact retrieval in worst-case instances requires $L = \Omega(n/r)$ whenever $\mathrm{dist}(v, \mathrm{anc}_k(v)) = \Omega(n)$. This does not imply that local mixing is useless; rather, it predicts a sharp dependence on the *instance geometry.* When the grammar G and ambiguity model $\mathcal{A}$ concentrate probability mass on trees whose relevant ancestors are typically nearby in yield order, or when we truncate recursion so that typical documents have small effective diameter, modest r may suffice empirically. Conversely, when we deliberately enforce large yield distance between bound occurrences and their ancestors (while preserving the anti-shortcut constraint), local mixing should fail unless depth grows with n. Thus, local mixing helps precisely when it expands the receptive field enough to cover the typical distances induced by G, not because it changes the underlying need for addressable retrieval.

The formulation of ATR^{++} is intentionally modular, and several extensions appear immediate while preserving the core theoretical phenomenon (pointer chasing under ambiguity). First, we can move from a single query to a *multi-query* regime: append a sequence of queries after EOS, each referencing a possibly distinct v and hop count k, and supervise the model on the corresponding sequence of answers. This stresses whether a model can reuse stored bindings repeatedly, as opposed to amortizing a single retrieval into a special-purpose state. Second, we can introduce *compositional queries* that request multiple ancestors (e.g. $\mathrm{anc}_{k_1}(v)$ and $\mathrm{anc}_{k_2}(v)$) or that request a relation between them (e.g. equality of terminal types). These variants remain reducible to repeated pointer jumps plus a small comparison circuit, and hence preserve the qualitative distinction between addressable binding and direct retrieval. Third, we can enrich the latent structure from a tree to a bounded-outdegree directed acyclic graph by allowing certain nonterminals to re-use previously introduced referents (a limited "reentrancy" operation). This would align the task more closely with program-like reference patterns while still permitting controlled anti-shortcut constructions.

A further extension, motivated by contemporary tool-use pipelines, is to interpret markers as *call identifiers* and terminal anchors as *return values.* The query then resembles a tool-call resolution step: "given call-id v, retrieve the value produced by its k-th enclosing context." Under this interpretation, bind-then-read corresponds to building a table of call frames (addressable by identifiers) and performing scoped lookup, whereas direct retrieval corresponds to attempting to summarize all pending frames into a bounded state. While the analogy is stylized, it suggests that ATR^{++} can serve as a controlled proxy for testing whether an architecture supports stable, indexable intermediate representations across long contexts and nested scopes.

We also note limitations. Our lower bound (Theorem 2) relies on an adversarially hard distribution (random pointer structure) and on standard bounded-precision/streaming-simulation assumptions. It therefore does not preclude that particular structured instances admit compact streaming solutions, nor does it rule out architectures that implement an external memory with learned addressing (which we explicitly treat as a separate mechanism class). Similarly, our upper bound (Theorem 1) assumes a head-terminal property and explicit markers that uniquely identify an occurrence; these are design choices that make the binding graph legible and the mechanistic analysis crisp. In naturalistic text, referents are rarely tagged unambiguously, and successful systems must jointly infer referents and retrieve bindings. We regard ATR^{++} not as a full model of natural language reference, but as a mechanistic benchmark in which we can (i) enforce anti-shortcut constraints, (ii) generate ground-truth latent bindings, and (iii) make precise causal claims about internal computation via interventions. Finally, interchange interventions scale with the number of sites and thus are expensive at large L and n; developing cheaper yet faithful approximations to $\mathsf{Sig}(M)$ remains an important practical problem if one wishes to turn mechanistic evaluation into a routine diagnostic.

9 Conclusion

We have introduced ATR$^{++}(\Pi)$ as a controlled family of binding-and-retrieval problems in which the latent structure is explicit (a tree T induced by a recursive PCFG G), the referent in the query is unambiguous (via markers inserted by $\mathcal{A}$), and yet superficial shortcuts are suppressed by construction (anti-shortcut constraints on positional features and on repeated terminal types). The point of this design is not to mimic natural text, but to isolate a mechanistic requirement—indexable access to intermediate bindings—that is easy to state, easy to generate at scale, and difficult to satisfy for a broad class of linear-time models without an explicit addressable store.

The theoretical picture is correspondingly sharp. On the positive side, Theorem 1 gives an explicit upper bound: a depth-$\Theta(k)$, 1-head Transformer with $d = \tilde{O}(|\Sigma|)$ can solve the task exactly by a literal pointer-jumping computation, where each layer performs a single hop from the current node representation to its parent anchor and the query merely initiates the chain by selecting the referred occurrence v. On the negative side, Theorem 2 isolates a principled obstruction for one-pass direct-retrieval schemes $s_t = F(s_{t-1}, e(x_t))$ without an addressable token-indexed memory: under a natural hard distribution (pointer chasing with random parent structure over $M = |\Sigma|$), any such streaming algorithm with $m = o(M)$ memory suffers constant error. Theorem 3 complements this by clarifying that "local mixing" does not by itself remove the need for global access; it only shifts the regime of success as a function of the induced yield distances and the receptive field growth $O(rL)$.

The mechanistic component of the benchmark is not an afterthought. We have defined an intervention-based diagnostic, $\mathsf{Sig}(M)$, built from likelihood-restoration attributions $\mathrm{Attrib}(f)$ computed via interchange interventions between clean and corrupted runs. Proposition 4 shows that, for the explicit upper-bound circuit, causal responsibility concentrates at hop-specific anchor sites, yielding a characteristic bind-then-read signature that is stable across the usual distribution shifts (held-out structures, held-out bindings, and length extrapolation). Conversely, for models that succeed by compressing the document into a final state, restoration concentrates near the end of the sequence and near query-adjacent sites. Thus ATR^{++} offers a setting in which behavioral success and mechanistic success can be separated: two models can achieve similar in-distribution error while implementing computations with different causal footprints and different extrapolation profiles.

We therefore propose ATR^{++} as a mechanistic benchmark in the strongest sense: (i) instances are generated from a transparent latent program (a derivation in G plus a marker process $\mathcal{A}$); (ii) the task admits a clean algorithmic solution whose internal steps can be localized to identifiable token positions; (iii) the benchmark comes with theory-backed separations that predict when certain architectural classes should fail; and (iv) we can test, via interventions, whether a trained model implements the intended algorithmic mechanism or a qualitatively different one. In particular, reporting only error rates obscures the central question: whether a model has learned to materialize and later reuse bindings, or whether it has learned to "guess" answers from a compressed summary that happens to work on the training regime.

To make this useful as a community diagnostic, we invite participation in a mechanistic leaderboard organized around two axes. The first axis is standard generalization: accuracy on in-distribution data and on explicit OOD splits (held-out bindings, held-out grammar structures, and length extrapolation beyond the training $n_{\max}$). The second axis is mechanism: submitters report $\mathsf{Sig}(M)$ computed on a fixed suite of corruption/intervention templates, together with a minimal description of the intervention sites used (layers, token positions, and any designated "anchor" ports). We emphasize that $\mathsf{Sig}(M)$ is not intended to be a single scalar; the most informative submissions will include layer-wise concentration measures (where restoration is written), position-wise localization measures (whether restoration occurs at anchors or at query-adjacent sites), and stability statistics across splits. In this format, the leaderboard becomes not merely a ranking of architectures by accuracy, but a catalog of which architectural features induce which mechanism class on a task with a known algorithmic solution.

We expect several concrete outcomes from such a benchmark. First, it should expose when linear-time sequence models achieve high in-distribution accuracy by exploiting regularities of a particular generator rather than by implementing indexable binding retrieval; the anti-shortcut constraints reduce trivial heuristics, but they do not remove the need for mechanistic auditing. Second, it should permit controlled ablations: by varying M, k, ambiguity strength in $\mathcal{A}$, recursion depth in G, and the split specification, one can probe the predicted tradeoffs between memory, depth, and addressability. Third, it provides a setting in which new architectural proposals—external memory, learned indexing, hybrid global-local mixers, or alternative attention sparsity patterns—can be evaluated not only by whether they solve the task, but by whether they solve it in a way that is causally legible and robust.

Several technical directions remain. On the theory side, it is natural to formalize intermediate mechanism classes (e.g. bounded associative memories or learned key-value stores) and to characterize their sample and compute requirements as functions of M and k. On the methods side, interchange interventions are expensive; approximations to $\mathsf{Sig}(M)$ that preserve fidelity while reducing the number of forward passes would make mechanistic evaluation routine. On the benchmark side, extending ATR^{++} to multi-query documents, limited reentrancy, or compositional queries should preserve the core pointer-chasing structure while stressing reuse and interaction between retrieved facts.

Our intended use of ATR^{++} is thus straightforward: we treat it as a minimal testbed in which architectural claims about retrieval, memory, and binding can be made precise, falsified behaviorally, and corroborated mechanistically. We invite the community to use the generator Π, the OOD splits, and the $\mathsf{Sig}(M)$ protocol to build a shared empirical record of which mechanisms emerge in which models, and under which training regimes those mechanisms remain stable under distribution shift.