LEARNING TO GENERALIZE RECURSIVELY: STRUCTURAL INSIGHTS FROM THE TOWER OF HANOI

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformer-based reasoning models achieve strong performance on multi-step reasoning benchmarks, yet their ability to generalize out-of-distribution (OOD) on structured reasoning and planning tasks, especially those involving recursive strategies, remains brittle. We study this limitation through the 3-peg Tower of Hanoi, a canonical recursive planning problem that requires preserving structural invariants over exponentially long execution horizons. We formulate the task as deterministic policy learning with a minimal state-based decoding rule, which separates one-step policy prediction from long-horizon deployment. We then analyze the structure of the optimal Hanoi trajectory and show that it has a parity-controlled cyclic pattern, which reveals two inductive biases important for recursive generalization: relative, shift-invariant interactions across disks and hierarchical computation aligned with recursive structure. Guided by these insights, we propose a hierarchy-aware Transformer-LSTM hybrid architecture that combines recurrence over disk order with rotary positional encoding (RoPE). Through systematic length-extrapolation experiments, we show that hierarchical recurrence substantially improves stability, and that combining it with RoPE enables near-perfect generalization far beyond the training regime. These results identify architectural inductive biases required to preserve structural consistency under deep recursive composition.

1 INTRODUCTION

Recent advances in large language models (LLMs) and large reasoning models (LRMs) have substantially improved performance on mathematical, symbolic, and structured reasoning tasks (Achiam et al., 2023; Trinh et al., 2024; Jaech et al., 2024; Guo et al., 2025; Comanici et al., 2025), aided by chain-of-thought reasoning, post-training, and increased test-time computation (Wei et al., 2022; Guo et al., 2025; Snell et al., 2024). At the same time, growing evidence suggests that such reasoning remains brittle under distribution shift, especially in compositional and algorithmic settings (Valmeekam et al., 2022; Dziri et al., 2023; Huang et al., 2023; McCoy et al., 2023; Turpin et al., 2023; Yuan et al., 2024; Mirzadeh et al., 2024; Shojaee et al., 2025). In particular, *The Illusion of Thinking* (Shojaee et al., 2025) shows that frontier reasoning models collapse in controlled puzzle environments as compositional depth increases, even when they generate extended reasoning traces.

A central manifestation of this limitation is out-of-distribution (OOD) generalization under increasing compositional depth (Hupkes et al., 2020; Delétang et al., 2022; Anil et al., 2022; Zhou et al., 2023). In recursive problems, scaling the instance size does not merely lengthen the input; it increases the depth at which the same local rule must be composed. Success therefore requires preserving structural invariants across repeated composition, rather than only recognizing local patterns. Prior work shows that Transformer-based models often struggle in precisely this regime, exhibiting sharp degradation in length extrapolation and compositional generalization (Anil et al., 2022; Zhang et al., 2023; Dziri et al., 2023; Mirzadeh et al., 2024; Zhang et al., 2024).

In this work, we study recursive OOD generalization through the classical 3-peg Tower of Hanoi, a canonical recursive planning problem. Tower of Hanoi has recently served as a controlled testbed for reasoning systems, including in *The Illusion of Thinking* (Shojaee et al., 2025), where frontier large reasoning models exhibit sharp performance degradation as problem size increases. At the same time, the puzzle has a transparent recursive structure, making it a clean and analytically tractable setting for studying how neural architectures extrapolate under increasing recursive depth.

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.

We follow a three-step approach. We first analyze the structure of the optimal Hanoi trajectory, then use the resulting insights to design a hierarchy-aware Transformer–LSTM architecture, and finally evaluate the learned policy in both one-step prediction and long-horizon deployment beyond the training regime. Our goal is not to characterize recursive reasoning in general, but to use this controlled setting to isolate architectural and training factors that support recursive length extrapolation.

Contributions. Our major contributions include:

1. *Policy learning formulation with structural decoding.* We formulate the 3-peg Tower of Hanoi as a deterministic policy learning problem and introduce a smallest-disk-based decoding rule that recovers the optimal action without explicit legality constraints.
2. *Structural characterization of the optimal solution.* We characterize the optimal Hanoi trajectory through a parity-controlled cyclic structure, showing that each disk follows a periodic orbit whose orientation is determined by a global parity variable.
3. *Hierarchy-aware architecture with structural inductive bias.* Guided by this structure, we propose a Transformer–LSTM hybrid architecture that combines relative interactions via RoPE with hierarchical recurrence over disk order.
4. *Controlled empirical study of recursive generalization.* Within the 3-peg Hanoi setting, we systematically evaluate one-step prediction and closed-loop deployment beyond the training regime. Our results show that combining hierarchical recurrence with relative positional encoding substantially improves recursive length extrapolation and long-horizon stability compared with the evaluated Transformer-only and recurrent baselines.

1.1 Related Works

Reasoning Models. Chain-of-thought prompting, supervised fine-tuning on reasoning traces, self-consistency, and reinforcement learning have substantially improved reasoning performance in LLMs (Wei et al., 2022; Kojima et al., 2022; Lewkowycz et al., 2022; Chung et al., 2024; Christiano et al., 2017; Wang et al., 2022; Ouyang et al., 2022; Weng et al., 2023; Shao et al., 2024; Yao et al., 2023; Besta et al., 2024; Snell et al., 2024; Guo et al., 2025). However, stronger reasoning traces do not necessarily imply systematic generalization: prior work reports brittleness in structured planning, compositional reasoning, and symbolic generalization, especially as problem complexity increases (Valmeekam et al., 2022; Dziri et al., 2023; Mirzadeh et al., 2024; Shojaee et al., 2025).

Neural Algorithmic Reasoning and Execution. Neural algorithmic reasoning studies whether neural networks can learn structured procedures that generalize beyond training complexity. Recurrent, convolutional, and graph-based models have been used to learn algorithmic tasks and generalize across problem sizes (Zaremba & Sutskever, 2014; Kaiser & Sutskever, 2015; Veličković et al., 2019; Veličković & Blundell, 2021). More recent approaches introduce iterative, recurrent, or hierarchical computation into attention-based models to improve systematic extrapolation (Dehghani et al., 2018; Yang et al., 2023; Fan et al., 2024; Wang et al., 2025; Altabaa et al., 2025).

Out-of-Distribution and Length Generalization. Length generalization evaluates sequence models on instances larger than those observed during training (Anil et al., 2022; Zhou et al., 2023; 2024). Prior work highlights the importance of positional encoding and architectural inductive bias for extrapolation (Kazemnejad et al., 2023; Su et al., 2024; Ruoss et al., 2023; Press et al., 2021). More broadly, successful algorithmic generalization often requires intermediate computation and architectures aligned with the underlying problem structure (Nye et al., 2021; Schwarzschild et al., 2021; Veličković et al., 2022; Altabaa et al., 2025; He, 2026).

2 Tower of Hanoi as a Testbed for Recursive Reasoning

Understanding whether neural networks truly internalize algorithmic procedures remains an open challenge. Recent work suggests that the apparent reasoning ability of modern models may often reflect shallow heuristics rather than genuine procedural understanding, creating an "illusion of thinking" (Shojaee et al., 2025). Controlled algorithmic puzzles are useful for studying this question because their correct solutions are known exactly, making it possible to evaluate generalization directly. Among such tasks, the *three-peg Tower of Hanoi* provides a particularly clean benchmark for recursive reasoning and long-horizon planning.

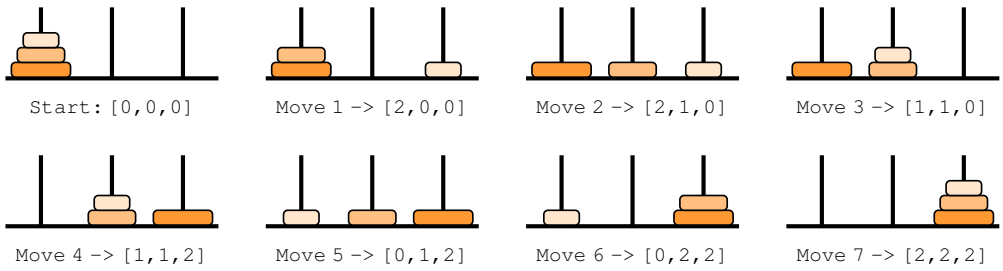

Figure 1: Example of the optimal solution for a 3-disk Tower of Hanoi puzzle. The sequence of configurations illustrates the unique shortest path from the initial state `[0,0,0]` to the goal state `[2,2,2]`, consisting of $2^3 - 1 = 7$ moves.

Problem Setting. The Tower of Hanoi consists of three pegs $\{0, 1, 2\}$ and N disks of distinct sizes. Initially, all disks are stacked on a source peg in decreasing order of size, with the largest at the bottom. The goal is to move the entire stack to a target peg subject to three rules: (i) only one disk may be moved at a time; (ii) only the top disk of a peg can be moved; and (iii) a larger disk may never be placed on top of a smaller one. Figure 1 illustrates the puzzle for $N = 3$.

Despite its simple rules, the puzzle admits a recursive optimal solution. To move a stack of N disks from a source peg to a target peg, one must first move the top $N - 1$ disks to the auxiliary peg, then move the largest disk to the target peg, and finally move the $N - 1$ disks from the auxiliary peg to the target peg. This reduces the original problem to two subproblems of the same form, differing only in the number of disks and the roles of the pegs. As a result, the recursion depth grows linearly with N, while the optimal solution length grows exponentially:

$$T_1 = 1, \qquad T_N = 2T_{N-1} + 1 = 2^N - 1 \ \text{ for } N \geq 2.$$

This combination of simple local rules and exponentially long optimal trajectories makes Tower of Hanoi a stringent test of recursive OOD generalization. Training on small instances and evaluating on larger ones requires extrapolation to deeper recursive structure and much longer execution horizons than those seen during training. Moreover, although many moves may be locally valid at a given state, only a highly structured sequence yields the optimal solution. Tower of Hanoi therefore provides a controlled setting for studying whether neural architectures can acquire transferable recursive structure, generalize beyond the training regime, and remain stable under repeated composition.

3 Modeling the Tower of Hanoi

We now introduce the modeling framework used throughout the paper. We first summarize the Transformer encoder and positional encoding schemes used in our models (§3.1), and then formulate Tower of Hanoi as a next-step prediction problem over disk states (§3.2).

3.1 Transformer Architecture and Positional Encoding

We use an encoder-only Transformer (Vaswani et al., 2017) as the backbone of our models. Given a sequence of input tokens, the encoder applies stacked self-attention and feed-forward layers to produce contextualized token representations. Full architectural and mathematical details are deferred to Appendix §B.2. Because self-attention is permutation-invariant, positional information must be introduced explicitly. We consider two positional encoding schemes with different inductive biases. *Absolute positional encoding (APE)* adds an index-dependent positional vector to each input embedding, directly tying the representation of a token to its absolute position. In contrast, *Rotary Positional Embeddings (RoPE)* (Su et al., 2024) multiplicatively transform the query and key vectors through index-dependent rotations inside the attention mechanism. Although these rotations are defined using absolute indices, their pairwise interaction depends only on the relative offset between token positions. Thus, APE introduces position *additively and absolutely* at the input level, whereas RoPE introduces it *multiplicatively* within attention and induces relative position-dependent interactions. This distinction will be important when we relate positional inductive bias to the recursive structure of the Hanoi dynamics.

3.2 Transformer Formulation of the Tower of Hanoi as Policy Learning

To study whether neural models can learn the recursive procedure underlying the Tower of Hanoi, we formulate the problem as deterministic policy learning over the puzzle state space.

MDP formulation. A state is represented as $\mathbf{x} = (x_0, x_1, \cdots, x_{N-1}) \in \mathcal{X} = \{0,1,2\}^N$, where x_i denotes the peg occupied by `disk i`, with disks indexed in increasing order of size (`disk 0` is the smallest). Because disks on each peg are always ordered by size, this representation uniquely specifies a valid configuration. An action is a triple $a = (i, s, t) \in \{0, 1, \cdots, N-1\} \times \{0,1,2\}^2$, meaning that `disk i` is moved from peg s to peg t. The dynamics are deterministic: for any valid state $\mathbf{x}$ and legal action a, the successor state is uniquely determined by a function $\mathbf{x}' = f(\mathbf{x}, a)$. Figure 12 illustrates one such transition. Thus, instead of viewing the policy as a distribution over actions, we may equivalently write it as a distribution over successor states, denoted by $\pi(\mathbf{x}' \mid \mathbf{x})$.

We fix the goal state to be $\mathbf{x}_{\text{goal}} = (2, 2, \cdots, 2)$. For the three-peg Tower of Hanoi, every non-terminal state lies on a unique shortest path to the goal, so the optimal policy $\pi^*(\cdot \mid \mathbf{x})$ is deterministic and selects a unique successor state at each step.

Policy learning via next-state prediction. Rather than modeling $\pi(a \mid \mathbf{x})$ directly, we learn the induced optimal transition $\mathbf{x} \mapsto \mathbf{x}'$. We model this mapping with a Transformer encoder that processes $\mathbf{x}$ as a sequence of disk tokens and predicts, in parallel, a conditional distribution over next peg assignments:

$$P_\theta(x_i' \mid \mathbf{x}), \qquad i = 0, 1, \cdots, N-1.$$

This yields an implicit policy $\pi_\theta(a \,|\, \mathbf{x}) \equiv \pi_\theta(\mathbf{x}' \,|\, \mathbf{x})$, where the action a is recovered from the predicted successor state $\mathbf{x}'$.

Structural reduction via `disk 0` decoding. A central challenge is to extract a valid successor state from the predicted marginals, since Tower of Hanoi imposes strong structural constraints across disks. Let

$$\mathcal{S}(\mathbf{x}) = \{\mathbf{x}' : \mathbf{x}' = f(\mathbf{x}, a),\ a \text{ is a legal move}\}$$

denote the set of legal successor states. Although $|\mathcal{X}| = 3^N$, one always has $|\mathcal{S}(\mathbf{x})| \leq 3$.

A naïve decoder would score all legal successors in $\mathcal{S}(\mathbf{x})$. Instead, we exploit a key structural property of the puzzle: the next move is uniquely determined by the behavior of the smallest disk (`disk 0`):

- If the predicted peg of `disk 0` changes, then `disk 0` itself is the moved disk.
- If the predicted peg of `disk 0` remains unchanged, then the move is determined by the unique legal move among the remaining disks, except in degenerate fully stacked states where no such move exists.

Hence the successor state can be decoded solely from whether `disk 0` moves and, if so, to which peg; for more details, see Appendix §C.1 and Figure 13. As a result, although the model predicts all disk marginals in parallel, only the logits for `disk 0` are needed to recover the optimal action.

Training objective. Based on this decoding rule, we train the model with the cross-entropy loss

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}}\big[-\log P_\theta(x_0' \mid \mathbf{x})\big], \tag{1}$$

where x_0' is the ground-truth next peg of `disk 0` under the optimal policy. Although supervision is applied only to `disk 0`, the structural properties of the puzzle ensure that correctly predicting its behavior determines the correct global transition.

4 Algorithmic Structure and Architecture Design

In this section, we analyze the structure of the optimal Hanoi trajectory and derive two architectural design principles from it. We first establish parity-controlled cyclic properties of the optimal solution (§4.1), and then use these insights to motivate our hybrid Transformer–LSTM architecture (§4.2).

4.1 Parity-Controlled Cyclic Structure of Optimal Trajectories

The optimal Tower of Hanoi trajectory is highly structured. Rather than exploring arbitrary configurations, it follows a deterministic hierarchical pattern in which each disk moves periodically, and the orientation of this motion is governed by parity.

We begin with the smallest disk, whose motion acts as the organizing signal for the entire trajectory.

Theorem 1 (Parity-controlled cyclic trajectory)**.** *Consider the optimal move sequence $\mu^\star$ of the N-disk, 3-peg Tower of Hanoi puzzle with source `peg 0` and target `peg 2`. Let p_t denote the peg location of the smallest disk (`disk 0`) after t moves along $\mu^\star$. Then:*

(a) `disk 0` moves on every odd-numbered step and remains fixed on every even-numbered step.

(b) Its motion follows a 3-cycle whose orientation depends only on the parity $(N \bmod 2)$:

$$p_{2k+1} = \sigma_N(p_{2k}),$$

where σ_N is (012) if N is even and (021) if N is odd.

(c) The entire optimal move sequence is uniquely determined by this disk-0 trajectory.

Theorem 1 shows that the smallest disk follows a deterministic parity-dependent 3-cycle across the pegs. This cycle acts as a global clock for the optimal solution: once the trajectory of `disk 0` is fixed, all remaining moves are uniquely determined by the puzzle constraints. An illustration is provided in Figure 14 and Appendix §C.3.

This structure extends beyond the smallest disk. Each disk moves at its own time scale, and its cyclic orientation is determined by its relative position in the recursive hierarchy.

Theorem 2 (Per-disk periodicity and recursive cycles)**.** *Consider the optimal move sequence $\mu^\star$ of the N-disk, 3-peg Tower of Hanoi puzzle with source peg 0 and target peg 2. Fix a disk index $0 \le i \le N-1$. Let*

$$m_i(1) < m_i(2) < \cdots < m_i(2^{N-i-1})$$

be the move indices at which disk i is moved along $\mu^\star$. Then:

*(a) **Periodicity.** The move times of disk i are*

$$m_i(k) = 2^i(2k-1), \qquad k = 1, 2, \cdots, 2^{N-i-1}.$$

In particular, disk i moves once every 2^{i+1} steps.

*(b) **Parity-controlled cycle.** At its move times, disk i follows a 3-cycle whose orientation is determined by the parity of its recursive depth $(N-i)$.*

Theorem 2 reveals two key features of the optimal Hanoi dynamics. First, `disk i` moves once every 2^{i+1} steps, so the solution unfolds over exponentially separated time scales. Second, the orientation of each disk's cyclic motion depends on its relative recursive depth $(N-i)$ rather than its absolute index. As a result, local interaction patterns are shared across disk levels up to a phase shift. A worked example is provided in Appendix §C.3 and Figure 15.

These structural properties suggest two architectural design principles. First, because the transition rules depend primarily on *relative disk indices*, positional encodings that preserve relative relationships are more natural than absolute ones. Second, because determining the next move amounts to identifying the active level of the recursive decomposition, the model should support hierarchical computation depth that scales with the disk ordering. Together, these observations motivate combining relative positional encoding with recurrence over disk hierarchy, which leads to the hybrid Transformer–LSTM architecture introduced next.

4.2 Hybrid Transformer-LSTM Architecture

The structural analysis in §4.1 suggests that an effective neural solver should capture two properties of the Hanoi dynamics: relative, shift-invariant interactions across disks, and hierarchical computation aligned with recursive structure. To incorporate these properties, we propose a hybrid architecture that combines a Transformer encoder with a recurrent module operating over disk order; see Figure 2.

Here we summarize two design principles inspired by the Hanoi structure:

(P1) **Relative, shift-invariant interactions.** The local transition rules depend primarily on relative disk order rather than absolute indices, suggesting the use of positional encodings that preserve relative relationships.

(P2) **Hierarchical, sequential inference with scalable computation.** Determining the next move requires scanning the disk hierarchy to identify the active recursive subproblem. This procedure is inherently ordered (from larger disks to smaller ones) and involves propagating information across levels. Consequently, it induces a computation whose effective depth grows with the number of disks N.

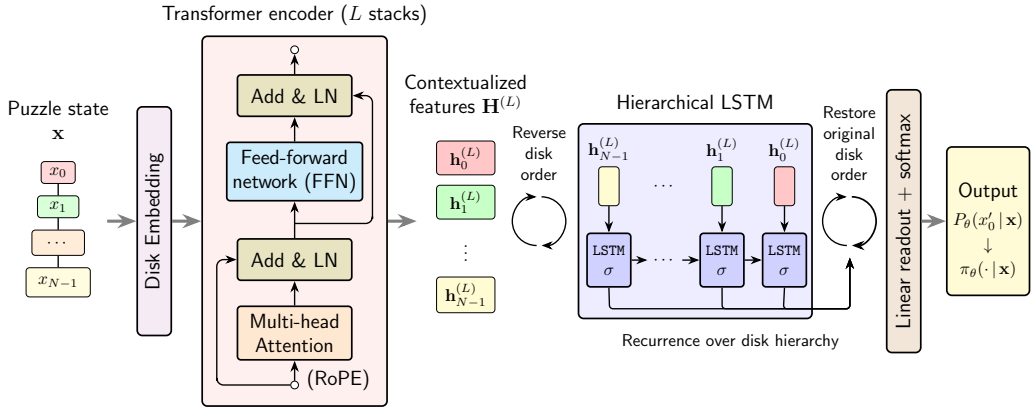

Figure 2: Transformer–LSTM hybrid architecture. The Transformer encoder produces contextualized disk features, which are reversed and processed by a hierarchical LSTM to enforce recurrence over disk hierarchy. The sequence is restored and passed to a shared linear readout for next-state prediction.

Now we explain how we incorporate these properties into our architecture.

Transformer encoder with relative positional encoding (P1). Given the current puzzle state $\mathbf{x} = (x_0, x_1, \cdots, x_{N-1})$, a Transformer encoder with L stacks processes the corresponding disk sequence and produces contextualized representations

$$\mathbf{H}^{(L)} = \big(\mathbf{h}_0^{(L)}, \mathbf{h}_1^{(L)}, \cdots, \mathbf{h}_{N-1}^{(L)}\big) \in \mathbb{R}^{N \times d}.$$

Through self-attention, each disk representation aggregates information from the full configuration. To encode position, we employ Rotary Positional Embeddings (RoPE), which make attention depend on relative offsets between disk indices. This matches the shift-invariant structure highlighted in §4.1, where local interaction patterns recur across disk levels up to phase shift.

Hierarchical recurrence over disk order (P2). To implement the sequential inference suggested by (P2), we reflect the disk hierarchy by reversing the disk order and processing the Transformer features from larger disks to smaller disks:

$$\widetilde{\mathbf{H}}^{(L)} = \big(\mathbf{h}_{N-1}^{(L)}, \mathbf{h}_{N-2}^{(L)}, \cdots, \mathbf{h}_0^{(L)}\big).$$

We then apply an LSTM (Hochreiter & Schmidhuber, 1997) over this reversed sequence,

$$(\mathbf{y}_i, \mathbf{s}_i, \mathbf{c}_i) = \mathrm{LSTM}\big(\mathbf{h}_{N-1-i}^{(L)}, \mathbf{s}_{i-1}, \mathbf{c}_{i-1}\big), \qquad i = 0, \cdots, N-1,$$

so that information is propagated along the disk hierarchy from larger to smaller disks. This mirrors the recursive organization of the puzzle, where the sub-goal of the smaller disks depends on the state of the larger recursive context.

Readout and interpretation. The LSTM outputs are reversed back to the original disk order and mapped to logits over peg indices through a shared linear head:

$$\mathbf{z}_i = \mathbf{W}_o \mathbf{y}_{N-1-i} + \mathbf{b}_o, \qquad P_\theta(x_i' = r \mid \mathbf{x}) = \mathrm{Softmax}(\mathbf{z}_i)_r, \quad r \in \{0, 1, 2\}.$$

Although the model predicts successor distributions for all disks, the decoding rule in §3.2 uses only the logits for `disk 0` to recover the next action. Accordingly, the model is trained using the cross-entropy loss on the next-state prediction of `disk 0`; see equation 1 defined in §3.2.

Overall, the Transformer serves as a global relational encoder, while the LSTM introduces hierarchy-aware computation over disk order. This recurrence is applied not across time steps, but across the structural ordering of disks within a single configuration. In this way, the architecture encodes the two inductive biases implied by the Hanoi dynamics: relative position-aware interactions and computation depth aligned with recursive structure.

The implementation details of the architecture, including data generation, training protocol, and hyperparameter settings, are deferred to Appendix B.

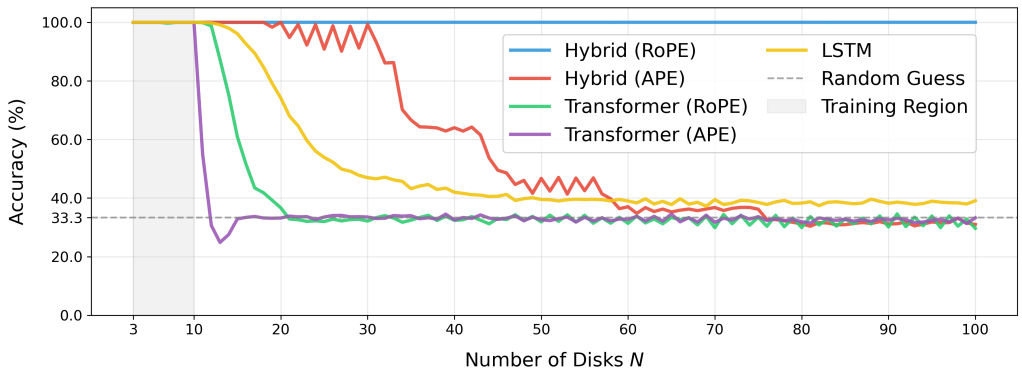

Figure 3: Local policy accuracy as a function of the number of disks N. The hybrid Transformer-LSTM with RoPE achieves near-perfect extrapolation, while Transformer-only and LSTM baselines degrade beyond the training range.

5 Length Generalization Experiments

In this section, we evaluate the learned policy π_θ from three complementary perspectives. We first measure one-step policy accuracy in §5.1, asking whether the model predicts the optimal action from a given state. We then study end-to-end deployment on complete Hanoi instances beyond the training regime in §5.2. Finally, in §5.3, we test whether stable extrapolation depends on identifying the global parity variable, which is studied by the theory in §4.1.

5.1 Local Policy Accuracy

We first evaluate how well the learned policy $\pi_\theta(\cdot \mid \mathbf{x})$ approximates the optimal policy $\pi^*(\cdot \mid \mathbf{x})$ at the one-step level. Since the optimal policy induces a unique successor state at each non-terminal configuration, this reduces to predicting the correct next state from the current state, without the confounding effects of long-horizon error accumulation.

Models for evaluation. To understand which architectural components are necessary for accurate policy approximation and length generalization, we compare five neural architectures that differ in their inductive biases:

- **LSTM** is a purely recurrent baseline that processes disks sequentially in reverse order.
- **Transformer (APE)** is an encoder-only Transformer with absolute positional embeddings, while **Transformer (RoPE)** replaces absolute encoding with rotary positional embeddings.
- **Hybrid (APE)** and **Hybrid (RoPE)** combine a Transformer encoder with a hierarchical LSTM operating over reversed disk order, enforcing recurrence along the disk hierarchy. The two variants differ only in their positional encoding scheme.

Full architectural and training details are given in Appendix §B.3.

Evaluation protocol. For each disk count $N \in \{3, \cdots, 100\}$, we evaluate on up to 10,000 puzzle states $\mathbf{x}$. For small N, these are drawn from the full valid state space; for larger N, they are sampled from the canonical optimal trajectory (Appendix §B). Given $\mathbf{x}$, an oracle solver provides the optimal successor state $\mathbf{x}^*$ under π^*, and the model produces a distribution $\pi_\theta(\cdot \mid \mathbf{x})$ over successor states (implicitly via disk-level predictions). We obtain a predicted successor state $\mathbf{x}'$ by selecting the highest-probability outcome:

$$\mathbf{x}' = \arg\max_{\widetilde{\mathbf{x}}} \ \pi_\theta(\widetilde{\mathbf{x}} \mid \mathbf{x}),$$

which in practice is determined via the `disk 0` decoding rule described in §3.2. The prediction is correct if $\mathbf{x}' = \mathbf{x}^*$, and local accuracy is the proportion of evaluated states for which this holds.

Results. Figure 3 shows a clear separation between architectures. Pure Transformer models degrade rapidly beyond the training range, even with RoPE, indicating that relative positional encoding alone is insufficient for stable recursive extrapolation. The LSTM baseline generalizes further, but still exhibits gradual degradation, suggesting that sequential inference without global relational reasoning

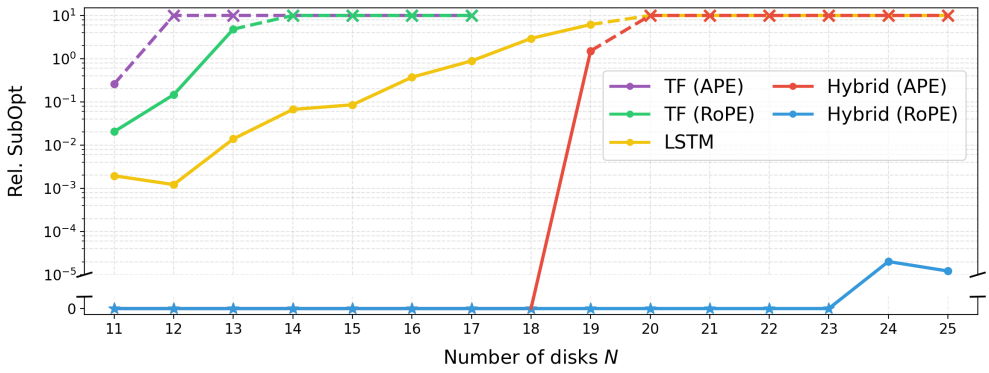

Figure 4: Deterministic deployment results under the step budget $B(N) = 10(2^N - 1)$. The y-axis shows the relative suboptimality $T/(2^N - 1) - 1$, where T is the rollout length. Perfect solutions are marked by $\star$, successful but suboptimal runs by $\bullet$, and failed runs by $\times$ placed at 10 for visualization. Dashed segments indicate the failure plateau. Hybrid (RoPE) remains near-optimal up to $N = 25$, whereas other architectures either fail earlier or incur rapidly increasing suboptimality.

is not enough. In contrast, Hybrid (RoPE) maintains near-perfect accuracy up to $N = 100$, while Hybrid (APE) degrades more slowly but still fails at very large sizes. Overall, these results imply that stable length generalization requires both hierarchical recurrence and relative positional encoding.

5.2 End-to-End Deployment

While §5.1 evaluates one-step policy accuracy, solving a complete Hanoi puzzle requires stable execution over an exponentially long horizon. For an N-disk instance, the optimal trajectory has length $T_N = 2^N - 1$, so even small local errors can accumulate and disrupt the recursive structure of the solution. As a result, a policy that closely approximates π^* at the one-step level may still fail under long-horizon deployment. We therefore evaluate the learned policy π_θ in an online setting on the canonical task of moving N disks from the initial state $\mathbf{x}_0 = (0, 0, \cdots, 0)$ to the goal state $\mathbf{x}_{\text{goal}} = (2, 2, \cdots, 2)$. We consider both stochastic and deterministic deployment strategies, each equipped with a cycle-avoidance mechanism; full details are deferred to Appendix §B.4.

In all cases, we impose a step budget $B(N) = 10(2^N - 1)$, and declare a rollout successful if the goal state is reached within this budget. For a successful rollout with total step count T, we report the relative suboptimality

$$\mathsf{SubOpt} := \frac{T}{2^N - 1} - 1,$$

which measures the fraction of redundant steps beyond the optimal solution length. Because rollout length grows exponentially with N, we evaluate this setting up to $N = 25$.

Results. Figure 4 summarizes deterministic deployment performance. Pure Transformer models remain unstable: TF (APE) fails beyond $N = 11$, while TF (RoPE) succeeds only up to $N = 13$ before incurring rapidly increasing suboptimality. The LSTM baseline solves instances up to $N = 19$, but its relative suboptimality grows quickly with N, indicating progressive drift from the optimal recursive trajectory. In contrast, both hybrid models execute the optimal solution for $N \leq 18$, after which their behavior diverges: Hybrid (APE) fails for $N \geq 20$ and already shows substantial overhead at $N = 19$, whereas Hybrid (RoPE) maintains near-perfect execution up to $N = 25$, with only negligible relative suboptimality at the largest tested sizes. Full numerical results for both deterministic and stochastic deployment are provided in Appendix §B.4.

These results reinforce the conclusions of §4.1. Hierarchical recurrence substantially improves long-horizon stability, but recurrence alone is not sufficient: robust deployment beyond the training regime requires combining hierarchical computation with relative positional encoding.

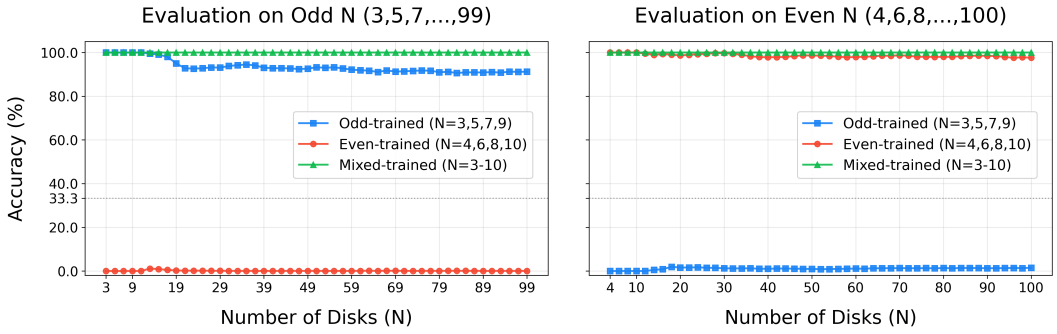

Figure 5: One-step policy accuracy of Hybrid (RoPE) under different parity training regimes. Models trained on odd-only or even-only data exhibit persistent even–odd oscillations in test accuracy, whereas mixed-parity training yields stable generalization across both parities.

5.3 Parity Ablation

The analysis in §4.1 shows that the optimal Hanoi trajectory is governed by a parity-controlled cyclic structure: the parity $(N \bmod 2)$ determines the global orientation of the recursive solution. This suggests that stable extrapolation requires identifying parity as a global structural variable. To test this hypothesis, we train Hybrid (RoPE) under three parity regimes: (i) *odd-only*: $N \in \{3, 5, 7, 9\}$; (ii) *even-only*: $N \in \{4, 6, 8, 10\}$; and (iii) *mixed-parity*: $N \in \{3, 4, 5, 6, 7, 8, 9, 10\}$. As in §5.1, we evaluate one-step policy accuracy for $N = 3, \cdots, 100$.

Results. Figure 5 shows that single-parity training induces a systematic failure mode: the model achieves high accuracy on test instances whose parity matches the training data, but fails on the opposite parity, and this oscillatory behavior persists even at large problem sizes. In contrast, mixed-parity training removes the oscillation and yields stable generalization across all tested lengths. This supports the structural picture developed in §4.1: parity acts as a global latent variable governing recursive orientation, and generalization fails when this variable is not identifiable from the training distribution. This conclusion is even strengthened in Appendix §B.5, which shows that adding even a single training length from the opposite parity is enough to correct the failure.

We also provide additional ablations on training-length coverage and imbalanced parity supervision. Details and results are reported in Appendix §B.5.

6 Conclusion

We studied recursive out-of-distribution generalization using the 3-peg Tower of Hanoi as a controlled and analytically tractable testbed. By separating one-step policy prediction from end-to-end deployment, we showed that, in this setting, strong local accuracy does not necessarily translate into stable execution over exponentially long horizons. Our structural analysis identified two properties of the optimal Hanoi solution—relative structure across disk indices and hierarchical recursive organization—which motivated a hybrid Transformer–LSTM architecture. Empirically, combining hierarchical recurrence with RoPE yields substantially more stable length extrapolation than the evaluated Transformer-only and recurrent baselines, including under long-horizon closed-loop deployment. We also found that parity acts as a global structural variable in the Hanoi solution: when the training distribution does not expose both parity regimes, the learned policy exhibits systematic parity-dependent generalization failures. Together, these experiments illustrate how architectural inductive biases and training coverage can interact with known algorithmic structure in a controlled recursive planning problem.

Our conclusions are intentionally limited to this setting. Tower of Hanoi has unusually explicit recursive structure, a deterministic optimal policy, and a compact state representation, and the architectural choices studied here are partly informed by these properties. We therefore do not claim that the same mechanisms are necessary or sufficient for recursive reasoning more generally. Rather, our results provide a concrete example in which aligning model structure and supervision with the underlying recursion can substantially improve OOD extrapolation. Extending this analysis to other recursive planning and algorithmic domains is an important direction for future work.

AI USE STATEMENT

In this work, we used generative AI tools to assist with literature discovery, coding, collation and formatting of experimental results, and language polishing. For literature discovery, AI tools were used to identify potentially relevant prior work; all cited papers were subsequently checked by the authors for relevance and accuracy. For coding assistance, AI-generated or AI-modified code was reviewed, tested, and validated by the authors before being used in the experiments. AI tools were also used to help aggregate experimental outputs and prepare tables or other summaries; the reported values were checked against the underlying experimental results. Finally, AI tools were used for grammatical correction, wording, and improving the clarity of the manuscript.

Generative AI was not used to develop the research ideas, derive the proposed architectural design, design the experiments, or determine the overall organization and scientific framework of the paper. The authors made al scientific decisions and independently reviewed all AI-assisted content. We take responsibility for the final content of this work, including text, claims, code, and other artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies neural models in the synthetic and fully specified Tower-of-Hanoi environment and does not involve human subjects, personally identifiable information, or sensitive datasets. The experiments are intended to study out-of-distribution generalization and architectural inductive biases in a controlled reasoning setting, rather than to make claims about human cognition or to support high-stakes decision making. We are not aware of specific societal harms arising from the methods or datasets used in this work. All reported experimental results were generated and evaluated using programmatically defined environments and evaluation procedures.

REPRODUCIBILITY STATEMENT

We have taken several steps to support reproducibility. The appendix provides the implementation details required to reproduce our experiments, including data generation and preprocessing, model architectures and hyperparameters, training protocols, deployment procedures, and evaluation settings. It also includes the full proofs of the theoretical results stated in the main text. We do not provide a public code repository during the anonymous review process; however, we plan to release the implementation and experiment code after the review process is completed.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Awni Altabaa, Siyu Chen, John Lafferty, and Zhuoran Yang. Unlocking out-of-distribution generalization in transformers via recursive latent space reasoning. *arXiv preprint arXiv:2510.14095*, 2025.

Cem Anil, Yuhuai Wu, Anders Andreassen, Aitor Lewkowycz, Vedant Misra, Vinay Ramasesh, Ambrose Slone, Guy Gur-Ari, Ethan Dyer, and Behnam Neyshabur. Exploring length generalization in large language models. *Advances in Neural Information Processing Systems*, 35:38546–38556, 2022.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 17682–17690, 2024.

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. On the practical ability of recurrent neural networks to recognize hierarchical languages. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 1481–1494, 2020.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. *arXiv preprint arXiv:1807.03819*, 2018.

Grégoire Delétang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, et al. Neural networks and the chomsky hierarchy. *arXiv preprint arXiv:2207.02098*, 2022.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. Faith and fate: Limits of transformers on compositionality. *Advances in neural information processing systems*, 36:70293–70332, 2023.

Ying Fan, Yilun Du, Kannan Ramchandran, and Kangwook Lee. Looped transformers for length generalization. *arXiv preprint arXiv:2409.15647*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Zhiyuan He. Exploring depth generalization in large language models for solving recursive logic tasks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(37):30996–31004, 2026.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*, 2023.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795, 2020.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Łukasz Kaiser and Ilya Sutskever. Neural GPUs learn algorithms. *arXiv preprint arXiv:1511.08228*, 2015.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36:24892–24928, 2023.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35: 22199–22213, 2022.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Eran Malach, Omid Saremi, Sinead Williamson, Arwen Bradley, Aryo Lotfi, Emmanuel Abbe, Josh Susskind, and Etai Littwin. To infinity and beyond: Tool-use unlocks length generalization in state space models. *arXiv preprint arXiv:2510.14826*, 2025.

R Thomas McCoy, Shunyu Yao, Dan Friedman, Matthew Hardy, and Thomas L Griffiths. Embers of autoregression: Understanding large language models through the problem they are trained to solve. *arXiv preprint arXiv:2309.13638*, 2023.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. GSM-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*, 2024.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021.

Anian Ruoss, Grégoire Delétang, Tim Genewein, Jordi Grau-Moya, Róbert Csordás, Mehdi Bennani, Shane Legg, and Joel Veness. Randomized positional encodings boost length generalization of transformers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1889–1903, 2023.

Avi Schwarzschild, Eitan Borgnia, Arjun Gupta, Furong Huang, Uzi Vishkin, Micah Goldblum, and Tom Goldstein. Can you learn an algorithm? generalizing from easy to hard problems with recurrent networks. *Advances in Neural Information Processing Systems*, 34:6695–6706, 2021.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Parshin Shojaee, Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. *arXiv preprint arXiv:2506.06941*, 2025.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Trieu H Trinh, Yuhuai Wu, Quoc V Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems*, 36:74952–74965, 2023.

Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Large language models still can't plan (a benchmark for LLMs on planning and reasoning about change). In *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Petar Veličković and Charles Blundell. Neural algorithmic reasoning. *Patterns*, 2(7), 2021.

Petar Veličković, Rex Ying, Matilde Padovano, Raia Hadsell, and Charles Blundell. Neural execution of graph algorithms. *arXiv preprint arXiv:1910.10593*, 2019.

Petar Veličković, Adrià Puigdomènech Badia, David Budden, Razvan Pascanu, Andrea Banino, Misha Dashevskiy, Raia Hadsell, and Charles Blundell. The CLRS algorithmic reasoning benchmark. In *International Conference on Machine Learning*, pp. 22084–22102. PMLR, 2022.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. Hierarchical reasoning model. *arXiv preprint arXiv:2506.21734*, 2025.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 2550–2575, 2023.

Liu Yang, Kangwook Lee, Robert Nowak, and Dimitris Papailiopoulos. Looped transformers are better at learning learning algorithms. *arXiv preprint arXiv:2311.12424*, 2023.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.

Qifan Yu, Zhenyu He, Sijie Li, Zhou Xun, Jun Zhang, Jingjing Xu, and Di He. Enhancing autoregressive chain-of-thought through loop-aligned reasoning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2206–2222, 2026.

Yu Yuan, Lili Zhao, Kai Zhang, Guangting Zheng, and Qi Liu. Do LLMs overcome shortcut learning? an evaluation of shortcut challenges in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 12188–12200, 2024.

Wojciech Zaremba and Ilya Sutskever. Learning to execute. *arXiv preprint arXiv:1410.4615*, 2014.

Dylan Zhang, Curt Tigges, Zory Zhang, Stella Biderman, Maxim Raginsky, and Talia Ringer. Transformer-based models are not yet perfect at learning to emulate structural recursion. *arXiv preprint arXiv:2401.12947*, 2024.

Shizhuo Dylan Zhang, Curt Tigges, Stella Biderman, Maxim Raginsky, and Talia Ringer. Can transformers learn to solve problems recursively? *arXiv preprint arXiv:2305.14699*, 2023.

Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Josh Susskind, Samy Bengio, and Preetum Nakkiran. What algorithms can transformers learn? a study in length generalization. *arXiv preprint arXiv:2310.16028*, 2023.

Yongchao Zhou, Uri Alon, Xinyun Chen, Xuezhi Wang, Rishabh Agarwal, and Denny Zhou. Transformers can achieve length generalization but not robustly. *arXiv preprint arXiv:2402.09371*, 2024.

A Related Work: Approaches to Length and Recursive Generalization

Puzzle-based evaluation of reasoning models. Recent work has used controlled puzzle environments to evaluate whether large reasoning models execute systematic reasoning procedures or instead rely on reasoning-like traces that fail under increasing complexity (Valmeekam et al., 2022; Mirzadeh et al., 2024; Dziri et al., 2023; Huang et al., 2023; Shojaee et al., 2025). In particular, Shojaee et al. (2025) study several algorithmic puzzles, including the Tower of Hanoi, and show that frontier reasoning models exhibit sharp accuracy collapse as problem complexity increases, even when they generate extended thinking traces. Their analysis also reveals that providing an explicit algorithm does not necessarily prevent failure, suggesting limitations not only in discovering a solution strategy but also in reliably executing recursive procedures. Tower of Hanoi is a particularly useful diagnostic in this context: its rules are simple, its optimal solution is uniquely characterized, and increasing the number of disks increases both recursive depth and execution horizon. Our work builds on this diagnostic setting but studies a different question. Rather than evaluating black-box language models through prompted full-solution generation, we train explicit neural policies on compact puzzle states and analyze whether their learned transition rules extrapolate beyond the training regime.

Depth generalization and recursive decomposition. A complementary line of work distinguishes *depth generalization* from *length generalization*. While length generalization asks whether a model can extrapolate to longer input sequences, depth generalization asks whether it can handle deeper nested computations, such as balanced brackets, recursive symbolic expressions, or hierarchical logical clauses (Bhattamishra et al., 2020; Delétang et al., 2022; Zhang et al., 2023). Recent work by He (2026) studies this distinction explicitly in recursive Boolean algebra, propositional logic, and compositional arithmetic tasks, showing that standard Transformers degrade much more strongly with recursion depth than with sequence length. This supports the view that recursive reasoning requires a form of computation depth or stack-like state that is not naturally supplied by fixed-depth Transformer layers. To mitigate this failure, He (2026) propose a looped locate-and-replace pipeline: a locator identifies directly evaluable subexpressions, a replacer evaluates them, and the reduced expression is fed back into the loop until the whole expression is resolved. This improves depth extrapolation by externalizing recursive evaluation as an iterative decomposition procedure. Our setting is different: rather than repeatedly rewriting a symbolic expression until it reduces to a base case, we ask whether a model can infer the next optimal Hanoi transition from a compact state representation.

Recursive and iterative latent computation. Another approach to recursive generalization is to give models additional computation depth through recurrence or iterative latent updates. Recurrent and looped Transformer architectures apply the same computational block repeatedly, allowing the amount of computation to scale with input complexity rather than being fixed by the number of feedforward layers (Dehghani et al., 2018; Yang et al., 2023; Fan et al., 2024). Recent work has further shown that such iteration can be made more algorithmically meaningful by aligning intermediate representations with a reference procedure. For example, Altabaa et al. (2025) propose recursive latent-space reasoning with input-adaptive recurrence, latent algorithmic supervision, discrete bottleneck states, and self-correction; their model learns to solve modular arithmetic on computation graphs by computing one graph layer per recurrent step. Similarly, Yu et al. (2026) align looped Transformer iterations with chain-of-thought rounds, using iteration-wise supervision to generate reliable reasoning traces that can subsequently improve an autoregressive CoT model. Our recurrence plays a more targeted role: rather than iterating the Transformer backbone with input-adaptive depth, we append a recurrent module over the disk hierarchy. Its computation depth therefore scales with the number of disks, and its loop structure directly corresponds to scanning for the active recursive subproblem.

Tool-augmented SSMs and external memory. A different route is to augment fixed-memory sequence models with external tools. Malach et al. (2025) argue that standalone SSMs cannot robustly solve long-form generation tasks whose output space grows with problem complexity, but that interactive tool use can overcome this limitation by providing external memory. In their framework, the model learns local read/write or pointer-based update rules, while the tool maintains the growing algorithmic state. This yields strong extrapolation on arithmetic, logical reasoning, and coding task. However, their Tower-of-Hanoi results are more limited: their Mamba model trained up to 8 disks reaches only partial extrapolation to 12 disks, and they note that Hanoi is especially sensitive because full-solution generation has exponential length. Our work considers the opposite regime:

the model has no external memory or tool-use traces. Rather than generating the full Hanoi solution, we learn a compact state-to-policy map and evaluate its extrapolation through one-step accuracy and closed-loop deployment. This formulation leads to substantially stronger Hanoi extrapolation, suggesting that for recursive planning tasks, learning the local policy structure can be more effective than treating the problem as long-form trajectory generation.

Position of our work. Taken together, prior work suggests that failures on recursive reasoning tasks are not merely failures of long-context modeling, but reflect a deeper difficulty in preserving algorithmic structure as recursive depth increases. Our work studies this issue in the Tower of Hanoi, where the optimal policy has an explicit recursive and parity-controlled structure. We show that stable extrapolation can be obtained by aligning the architecture with this structure: RoPE captures relative disk-index relationships, while hierarchical recurrence provides computation depth over the disk hierarchy. Together with a compact disk-0 decoding rule, these ingredients yield a neural policy that internalizes the recursive transition structure, rather than relying on external tools, algorithmic traces, or full trajectory generation.

B Experiment Details

B.1 Data Format and Generation

A data instance is a dictionary

$$\{\texttt{state}, \texttt{teacher_move}, \texttt{noisy_target}, \texttt{legal_moves}, \texttt{N}\},$$

where the state is represented by a length-N vector $\texttt{state} = x = (x_0, x_1, \cdots, x_{N-1}) \in \{0,1,2\}^N$, with x_i indicating the peg index of disk i (disks are indexed by $0, 1, \cdots, N-1$ from smallest to largest). The supervision signal $\texttt{teacher_move} = (r_{\text{from}}, r_{\text{to}}) \in \{0,1,2\}^2$ specifies the optimal one-step move from x toward the goal configuration (all disks on peg 2). We also store `legal_moves`, the set of all legal one-step moves from x.

Finally, `noisy_target` stores a soft next-state target in an $N \times 3$ format, where row i is a probability distribution over the next peg of disk i. The oracle successor receives mass $1-\epsilon$, and the remaining mass ϵ is distributed uniformly over the successor states induced by the other legal moves, with $\epsilon > 0$ the noise rate. In the actual training objective, however, only the row corresponding to `disk 0` is used, consistent with the loss in Eq. equation 1. Thus, although `noisy_target` is stored as an $N \times 3$ matrix for convenience, the effective supervision used for training is the 3-dimensional soft target for `disk 0`.

We generate the mapping from states to optimal moves using three regimes:

- For small problem sizes ($N = 3$–12), we construct the full state space by performing breadth-first search (BFS) starting from the target configuration and enumerating all reachable states, which exhausts the entire configuration space of size 3^N. This provides complete supervision over all valid states and their optimal successors.
- For intermediate sizes ($N = 13$–20), enumerating all 3^N states becomes computationally expensive. Instead, we generate data by recursively simulating the canonical optimal solution of the Tower of Hanoi, which contains 2^N states along the minimal-length trajectory. This yields exact state–successor pairs restricted to the optimal path.
- For larger instances ($N > 20$), explicitly constructing the entire trajectory is memory-intensive. We therefore adopt a direct large-sampling strategy: we uniformly sample a time index t from the optimal trajectory and compute the corresponding state $x(t)$ and successor $x(t+1)$ analytically via the recursive structure of the problem, without materializing the full trajectory. This procedure requires only $\mathcal{O}(N)$ time per sample and enables scalable data generation beyond exhaustive enumeration.

B.2 The Complete Transformer Architecture

We briefly review the Transformer architecture and positional encoding mechanisms, which will be used in our design.

Encoder-only Transformer. We employ an encoder-only Transformer (Vaswani et al., 2017) which processes a sequence of tokens in parallel and produces contextualized representations via self-

attention. Let $\mathbf{H}^{(0)} \in \mathbb{R}^{N\times d}$ denote the input embedding matrix, where N is the sequence length and d is the hidden dimension. The Transformer encoder consists of L stacked layers. For layer ℓ, the hidden states evolve according to the standard pre-LayerNorm formulation:

$$\mathbf{Z}^{(\ell)} = \mathrm{MHA}\big(\mathrm{LN}(\mathbf{H}^{(\ell-1)})\big) + \mathbf{H}^{(\ell-1)},$$
$$\mathbf{H}^{(\ell)} = \mathrm{FFN}\big(\mathrm{LN}(\mathbf{Z}^{(\ell)})\big) + \mathbf{Z}^{(\ell)},$$

where MHA denotes multi-head self-attention, FFN denotes a position-wise feed-forward network, and LN denotes Layer Normalization. In a Transformer encoder, the self-attention mechanism allows each token to aggregate information from all other tokens, enabling the model to capture global interactions within the sequence, and Feed-Forward Network (FFN) is applied independently to each token and refines the aggregated contextual information.

Absolute Positional Encoding. Unlike recurrent or convolutional sequence models, the Transformer architecture itself is permutation-invariant, so positional information must be injected into the input representations. A common approach is to add an index-dependent positional embedding to each token. Let $\mathbf{e}_i \in \mathbb{R}^d$ denote the embedding of token i. The input representation is formed as

$$\mathbf{h}_i^{(0)} = \mathbf{e}_i + \mathbf{p}_i,$$

where $\mathbf{p}_i$ is an absolute positional encoding vector. A widely used choice is the sinusoidal positional embedding introduced in Vaswani et al. (2017). For an even hidden dimension d, the encoding is defined component-wise as

$$\mathbf{p}_i = [\sin(i\omega_0), \cos(i\omega_0), \cdots, \sin(i\omega_{d/2-1}), \cos(i\omega_{d/2-1})], \qquad \omega_k = \beta^{-2k/d}.$$

These embeddings provide a deterministic representation of token indices and allow the model to learn position-dependent interactions.

Rotary Positional Embeddings (RoPE). Absolute positional embeddings encode position through fixed index-dependent vectors, which can bias the model toward position-specific patterns. An alternative positional encoding method is Rotary Positional Embedding (RoPE) (Su et al., 2024), which incorporates positional information directly into the attention mechanism.

Instead of adding positional vectors to token embeddings, RoPE applies position-dependent rotations to the query and key representations. Consider a single attention head with dimension d_h (assumed even). Let $\mathbf{q}_i, \mathbf{k}_i \in \mathbb{R}^{d_h}$ denote the query and key vectors associated with token i. RoPE partitions these vectors into two-dimensional subvectors and applies a rotation whose angle is proportional to the token index. Let $\mathbf{\Theta} = (\theta_0, \cdots, \theta_{d_h/2-1})$ denote a set of frequencies. The rotation matrix $\mathbf{R}_{\Theta,i}$ is block diagonal with 2×2 blocks

$$\mathbf{R}_{\theta_k,i} = \begin{pmatrix} \cos(i\theta_k) & -\sin(i\theta_k) \\ \sin(i\theta_k) & \cos(i\theta_k) \end{pmatrix}, \quad i = 0, 1, \cdots, d/2-1.$$

The rotated query and key vectors are $\mathbf{q}'_i = \mathbf{R}_{\Theta,i}\mathbf{q}_i$ and $\mathbf{k}'_j = \mathbf{R}_{\Theta,j}\mathbf{k}_j$, respectively. The attention score then satisfies

$$\langle \mathbf{q}'_i, \mathbf{k}'_j \rangle = \langle \mathbf{R}_{\Theta,i}\mathbf{q}_i, \mathbf{R}_{\Theta,j}\mathbf{k}_j \rangle = \langle \mathbf{q}_i, \mathbf{R}_{\Theta,j-i}\mathbf{k}_j \rangle.$$

which depends only on the relative offset $(j - i)$ between token indices. Thus RoPE introduces positional information through relative rotations while preserving the structure of the attention mechanism.

B.3 Model Setting and Training

Models for Evaluation. We evaluate five neural solvers. Their architectural sizes are not matched exactly; instead, each model is configured to achieve reasonably good training and validation performance within the training range, so that the comparison is not dominated by obviously under-tuned baselines.

- **LSTM.** A purely recurrent baseline that processes the disk sequence in reverse order (from larger disks to smaller disks) using a 4-layer LSTM with hidden dimension $d_{\text{model}} = 64$. A shared step-wise readout is applied to the LSTM outputs to predict next-state logits for each disk. This model contains 133.6K learnable parameters and does not employ self-attention or explicit positional encoding.

- **Transformer (APE).** An encoder-only Transformer with absolute positional embeddings, using 4 attention heads, model dimension $d_{\text{model}} = 64$, feed-forward dimension $d_{\text{ff}} = 128$, and 10 encoder layers. Absolute positions are embedded using a base $\beta = 10000$. This model contains 337.5K learnable parameters.
- **Transformer (RoPE).** An encoder-only Transformer with the same architecture as Transformer (APE), but replacing absolute positional embeddings with rotary positional embeddings (RoPE) using base $\theta = 10000$. This model has 337.5K learnable parameters.
- **Hybrid (APE).** A Transformer–LSTM hybrid model that augments a 2-layer Transformer encoder (4 heads, $d_{\text{model}} = 64$, $d_{\text{ff}} = 128$) with a 2-layer LSTM ($d_{\text{model}} = 64$) operating over reversed disk order. Absolute positional embeddings with base $\beta = 10000$ are used. The LSTM hidden dimension is set to $d_{\text{model}} = 64$. This model contains 133.6K learnable parameters.
- **Hybrid (RoPE).** The same hybrid architecture as above, but using rotary positional embeddings with base $\theta = 10000$ in place of absolute positional embeddings. This model contains approximately 133.6K learnable parameters.

Label Smoothing. We use label smoothing with $\epsilon = 0.2$ during the early stage of training for the Transformer-based models to improve optimization stability. After this initial phase, training proceeds with standard one-hot targets. We study the effect of label smoothing on length generalization separately in Appendix B.6.

Training Protocol. All models are trained within the length range $N \in \{3, \cdots, 10\}$, using curriculum learning where specified. Hyperparameters are tuned independently for each architecture to ensure fair comparison.

- **LSTM.** The LSTM baseline is optimized using Adam with learning rate $\eta = 10^{-4}$ and batch size 256. The model is trained directly on instances with $N \in \{3, \cdots, 10\}$ for 200 epochs without staged curriculum.
- **Transformer (APE/RoPE).** Transformer-only models are optimized using AdamW with learning rate $\eta = 2 \times 10^{-4}$, $\beta = (0.9, 0.999)$, weight decay 10^{-6}, batch size 64, gradient clipping at 1.0, and dropout 0.20. Training follows a curriculum over problem size:
  - 600 epochs on $N \in \{3, \cdots, 8\}$,
  - 200 epochs on $N \in \{3, \cdots, 9\}$,
  - 200 epochs on $N \in \{3, \cdots, 10\}$.

  Label smoothing with $\epsilon = 0.2$ is applied during the first 100 epochs of the first stage, after which training uses one-hot targets.
- **Hybrid (APE/RoPE).** Hybrid models are optimized using AdamW with learning rate $\eta = 2 \times 10^{-4}$, $\beta = (0.9, 0.999)$, weight decay 10^{-6}, batch size 64, gradient clipping at 1.0, and dropout 0.10. Training proceeds in three stages:
  - 200 epochs on $N \in \{3, \cdots, 8\}$,
  - 50 epochs on $N \in \{3, \cdots, 9\}$,
  - 50 epochs on $N \in \{3, \cdots, 10\}$.

  Label smoothing with $\epsilon = 0.2$ is applied during the first 100 epochs of the first stage, after which training uses one-hot targets.

We note that our training settings are tuned separately for each architecture rather than enforcing an identical optimization protocol across all models, since their convergence behavior differs substantially. In particular, using the shorter hybrid training schedule for Transformer-only baselines leads to poor training-set convergence, while shallower Transformer variants perform poorly even with RoPE. Likewise, increasing the LSTM learning rate from 10^{-4} to 2×10^{-4} substantially degrades in-distribution validation performance under both one-hot and label-smoothed supervision, despite nearly perfect training accuracy; we therefore use 10^{-4} for the LSTM baseline. These architecture-specific configurations are chosen to obtain strong in-distribution fits before evaluating length extrapolation.

Compute resources. All experiments were run on a workstation with 4 NVIDIA GeForce RTX 3090 GPUs, each with 24GB memory, using CUDA 12.8. Training a single Hybrid model required less than 1 hour for 300 epochs, while training Transformer-only models required approximately

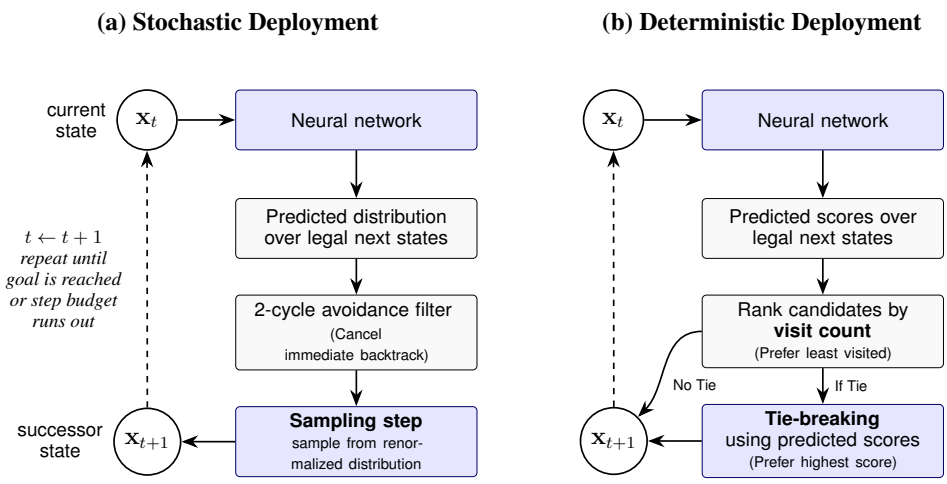

Figure 6: Two deployment strategies for rollout evaluation.

2-3 hours for 1,000 epochs due to greater model depth. The most time-consuming component is end-to-end deployment, whose cost grows exponentially with puzzle size. In particular, a single deterministic rollout for $N = 25$ required about 5 days, corresponding to an execution speed of roughly 80 steps per second.

B.4 End-to-End Policy Deployment

While §5.1 evaluates one-step policy accuracy, this does not guarantee successful execution under long-horizon deployment. We therefore evaluate the learned policy π_θ in an online setting, where it is iteratively applied to solve complete Tower of Hanoi instances. This appendix presents the full end-to-end deployment results corresponding to the experiments summarized in §5.2, including both stochastic and deterministic deployment strategies and their associated cycle-avoidance mechanisms.

The end-to-end deployment setting is fundamentally more challenging. The optimal trajectory has length $T_N = 2^N - 1$, which grows exponentially with N. Even under an optimistic independence assumption with per-step correctness p, the probability of completing the entire trajectory without error is p^{T_N}, which decays exponentially in T_N. Moreover, errors are not local: a single incorrect move can disrupt the recursive structure of the current subproblem, requiring recovery over a block of size $\Theta(2^k)$ for some subproblem size k. Such disruptions can trigger cascades of further errors. As a result, policies that closely approximate π^* at the one-step level may still fail catastrophically when deployed over the exponentially long horizon of the complete puzzle.

To evaluate whether the learned models can be effectively deployed as decision-making agents, we consider two online deployment strategies that use π_θ as a heuristic to guide action selection, which are shown in Figure 6.

Evaluation protocol. We evaluate policy deployment on the canonical task of moving N disks from the initial state $\mathbf{x}_0 = (0, 0, \cdots, 0)$ to the goal state $\mathbf{x}_{\text{goal}} = (2, 2, \cdots, 2)$. The optimal solution length is $2^N - 1$, and we impose a step budget of $B(N) = 10(2^N - 1)$. A rollout is declared successful if the goal state is reached within this budget.

B.4.1 Stochastic deployment with 2-cycle avoidance

We first treat the model as a stochastic policy. At each state $\mathbf{x}$, we enumerate all legal successor states $\mathcal{S}(\mathbf{x})$ and score them using π_θ. We then sample the next state from the resulting softmax distribution. To prevent trivial oscillations, immediate backtracking to the previous state is prohibited.

For each (model, N) pair, we perform 20 independent runs from the standard initial configuration $\mathbf{x}_0$ under step budget $B(N)$, and report:

- **Success ratio:** the fraction of runs that successfully reach the goal within the step budget.
- **Mean steps:** the average number of steps taken among successful runs (reported in thousands), together with the standard deviation.
- **Relative suboptimality:** for each successful rollout, we compute

$$\mathsf{SubOpt} = \frac{T}{2^N - 1} - 1,$$

where T is the total number of executed steps and $2^N - 1$ is the optimal solution length for the N-disk puzzle. This quantity measures the fraction of redundant steps beyond the optimal trajectory. We then report the mean SubOpt over successful runs only. A value of 0 indicates exact optimal execution, while larger values indicate increasing deviation from the optimal recursive solution.

Statistics for step count and optimality gap are computed conditional on success. "N/A" indicates that no run succeeded within the step budget.

Results. Detailed numerical results are reported in Table 1. Each configuration is evaluated over 20 independent runs under the step budget $B(N) = 10(2^N - 1)$, and we report success ratios, mean steps executed, and optimality gaps. In these results, the gap between local prediction accuracy and long-horizon execution becomes immediately apparent:

(i) Pure Transformer models deteriorate quickly: TF (APE) fails beyond $N = 11$, and TF (RoPE) collapses after $N = 13$. Although RoPE improves short-range stability, it does not prevent long-horizon drift without hierarchical recurrence.

(ii) The LSTM baseline remains stable up to $N = 14$, but performance degrades rapidly as N increases, reflecting accumulating execution errors over long trajectories.

(iii) In contrast, the Hybrid (APE) model follows the exact optimal trajectory up to $N = 18$, but fails abruptly at $N \geq 19$, indicating sensitivity to absolute positional encoding when extrapolating to deeper recursion levels. The Hybrid (RoPE) model, however, achieves perfect success ratio and exact optimal execution for all tested sizes $N = 11$–20, with the number of steps exactly matching $2^N - 1$ and zero variance across runs.

These results highlight the importance of architectural inductive bias for stable long-horizon execution. While one-step prediction accuracy alone is insufficient, combining hierarchical recurrence with relative positional encoding enables the model to preserve the recursive structure of the Hanoi solution under stochastic deployment.

B.4.2 Deterministic deployment with cycle detection

We next evaluate a deterministic deployment strategy with explicit cycle detection. We maintain visit counts for previously encountered states. At each step, the agent selects the successor with the smallest visit count to discourage revisiting previously explored regions of the state space. When multiple successors share the same minimal count, π_θ is used for *tie-breaking* by selecting the move with the highest predicted score. Both Figure 4 and Table 2 report deterministic rollout results:

(i) Pure Transformer models remain unstable: TF-APE fails beyond $N = 11$, while TF-RoPE succeeds only up to $N = 13$ before incurring rapidly increasing suboptimality.

(ii) The LSTM baseline solves instances up to $N = 19$, but the number of excess steps grows quickly with N, indicating increasing deviation from the optimal recursive trajectory.

(iii) Both hybrid models execute the optimal solution for $N \leq 18$. Beyond this range their behavior diverges: Hybrid (APE) fails for $N \geq 20$ and already exhibits significant overhead at $N = 19$, whereas Hybrid (RoPE) maintains near-perfect execution up to $N = 25$, with only negligible excess steps at the largest sizes tested.

These deterministic results reinforce the structural conclusions of Section 4.1. Hierarchical recurrence stabilizes long-horizon reasoning by enabling sequential inference over disk hierarchy, while relative positional encoding preserves the shift-invariant interactions required for consistent recursion across larger puzzles.

Key takeaways from end-to-end deployment

1. High one-step accuracy does *not* guarantee successful long-horizon execution due to exponential error amplification.
2. Transformer architectures without recurrence fail to maintain global consistency under recursive composition.
3. Hierarchical recurrence stabilizes execution but is insufficient alone for extrapolation.
4. Combining recurrence with relative positional encoding (Hybrid + RoPE) enables near-perfect execution far beyond the training regime.

Table 1: Stochastic deployment results with two-cycle avoidance for $N = 11$–20. For each model and disk count N, results are averaged over 20 independent runs under the step budget $B(N) = 10(2^N - 1)$. We report the success ratio (fraction of runs reaching the goal within the budget), the mean number of steps taken (in thousands) with standard deviation conditioned on success, and the optimality gap defined as the ratio between the achieved number of steps and the optimal solution length $2^N - 1$. A value of 1.00 in the optimality gap indicates execution of the exact optimal trajectory without deviation. "N/A" indicates that the model failed to solve any instance within the allotted step budget.

Model	**Metric**	11	12	13	14	15	16	17	18	19	20
Optimal	Steps	2047	4095	8191	16383	32767	65535	131071	262143	524287	1048575
TF (APE)	Success ratio	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Steps ($\times 10^3$)	$16.30_{\pm 2.97}$	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	SubOpt	$6.96_{\pm 1.45}$	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
TF (RoPE)	Success ratio	1.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Steps ($\times 10^3$)	$2.947_{\pm 0.063}$	$6.411_{\pm 0.153}$	$28.99_{\pm 6.97}$	N/A	N/A	N/A	N/A	N/A	N/A	N/A
	SubOpt	$0.44_{\pm 0.03}$	$0.57_{\pm 0.04}$	$2.54_{\pm 0.85}$	N/A	N/A	N/A	N/A	N/A	N/A	N/A
LSTM	Success ratio	1.00	1.00	1.00	1.00	0.95	0.10	0.00	0.00	0.00	0.00
	Steps ($\times 10^3$)	$2.055_{\pm 0.005}$	$4.120_{\pm .013}$	$8.258_{\pm .122}$	$17.83_{\pm 0.41}$	$121.7_{\pm 75.0}$	$339.8_{\pm 164.9}$	N/A	N/A	N/A	N/A
	SubOpt	$0.00_{\pm 0.00}$	$0.01_{\pm 0.00}$	$0.04_{\pm 0.01}$	$0.09_{\pm 0.03}$	$2.71_{\pm 2.29}$	$4.19_{\pm 2.52}$	N/A	N/A	N/A	N/A
Hybrid (APE)	Success ratio	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.00	0.00
	Steps ($\times 10^3$)	$2.047_{\pm 0}$	$4.095_{\pm 0}$	$8.191_{\pm 0}$	$16.383_{\pm 0}$	$32.767_{\pm 0}$	$65.535_{\pm 0}$	$131.071_{\pm 0}$	$262.143_{\pm 0}$	N/A	N/A
	SubOpt	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	N/A	N/A
Hybrid (RoPE)	Success ratio	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	Steps ($\times 10^3$)	$2.047_{\pm 0}$	$4.095_{\pm 0}$	$8.191_{\pm 0}$	$16.383_{\pm 0}$	$32.767_{\pm 0}$	$65.535_{\pm 0}$	$131.071_{\pm 0}$	$262.143_{\pm 0}$	$524.287_{\pm 0}$	$1048.575_{\pm 0}$
	SubOpt	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$	$0.00_{\pm 0.00}$

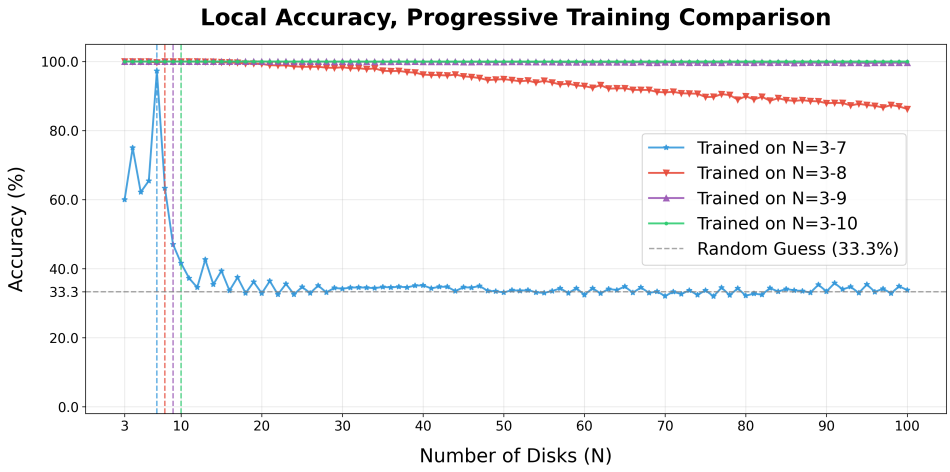

Figure 7: Forward accuracy under progressive training length coverage. Increasing the maximum training length substantially improves extrapolation to larger N. Training on $N \leq 7$ yields limited short-range extrapolation, while training on $N \leq 10$ achieves stable generalization up to $N = 100$.

B.5 Ablation Studies on Training Length Coverage

To better understand the source of length generalization in Hybrid-RoPE, we conduct a series of controlled ablation studies along two structural axes of the Tower of Hanoi algorithm. Building on the structural analysis in §4.1, we examine two key axes of the Hanoi solution: (i) recursive depth and (ii) parity-controlled orientation. These experiments aim to identify which aspects of the training distribution are necessary for learning a policy that extrapolates beyond the training regime.

B.5.1 Effect of Training Length Coverage

We first investigate how the maximum training length affects extrapolation to larger problem sizes. Since the Tower of Hanoi admits a recursive solution, length generalization requires the model to internalize and compose this recursive structure beyond the depths observed during training.

We progressively train Hybrid-RoPE on increasing ranges of puzzle size N:

(i) $N \in \{3, 4, 5, 6, 7\}$,

(ii) $N \in \{3, 4, 5, 6, 7, 8\}$,

(iii) $N \in \{3, 4, 5, 6, 7, 8, 9\}$, and

(iv) $N \in \{3, 4, 5, 6, 7, 8, 9, 10\}$.

All models are evaluated on forward next-step prediction for N up to 100.

As shown in Figure 7, training on $N \leq 7$ does not immediately collapse beyond the training range: the model maintains accuracy significantly above random guessing for $N = 8$–10. However, performance gradually deteriorates as N increases further and eventually approaches chance level. Extending the training range to $N \leq 8$ improves stability but still exhibits progressive degradation. Training on $N \leq 9$ nearly eliminates this trend, and training on $N \leq 10$ achieves stable generalization up to $N = 100$.

This pattern reflects the recursive depth of the Hanoi solution. The optimal policy for N disks decomposes into subproblems of size $N - 1$, so training up to size K exposes the model to recursive compositions of depth K. Extrapolating to $N > K$ therefore requires composing the learned recursion one level deeper than observed during training. When the training range is insufficient, the learned structure generalizes locally but fails to maintain global consistency at greater depths. These results show that sufficient coverage of recursion depth is essential for stable length generalization.

Table 2: Deterministic cycle-aware deployment under step budget $B(N) = 10(2^N - 1)$. We report solved status, excess steps $T - (2^N - 1)$, and relative gap $T/(2^N - 1) - 1$.

(a) $N = 11$–17 (All Models)

Model(s)	**Metric**	11	12	13	14	15	16	17
Optimal	Steps	2,047	4,095	8,191	16,383	32,767	65,535	131,071
TF (APE)	Solved	✓	✗	✗	✗	✗	✗	✗
	Excess	532	N/A	N/A	N/A	N/A	N/A	N/A
	SubOpt	0.260	N/A	N/A	N/A	N/A	N/A	N/A
TF (RoPE)	Solved	✓	✓	✓	✗	✗	✗	✗
	Excess	42	603	39,507	N/A	N/A	N/A	N/A
	SubOpt	2.05×10^{-2}	0.147	4.82	N/A	N/A	N/A	N/A
LSTM	Solved	✓	✓	✓	✓	✓	✓	✓
	Excess	4	5	114	1,096	2,800	24,387	115,429
	SubOpt	1.95×10^{-3}	1.22×10^{-3}	1.39×10^{-2}	6.69×10^{-2}	8.55×10^{-2}	0.372	0.881
Hybrid (APE/RoPE)	Solved	✓	✓	✓	✓	✓	✓	✓
	Excess	0	0	0	0	0	0	0
	SubOpt	0	0	0	0	0	0	0

(b) $N = 18$–25 (Hybrid Comparison)

Model	**Metric**	18	19	20	21	22	23	24	25
Optimal	Steps	262,143	524,287	1,048,575	2,097,151	4,194,303	8,388,607	16,777, 215	33,554,431
LSTM	Solved	✓	✓	✗	✗	✗	✗	✗	✗
	Excess	770,681	3,227,135	N/A	N/A	N/A	N/A	N/A	N/A
	SubOpt	2.94	6.16	N/A	N/A	N/A	N/A	N/A	N/A
Hybrid (APE)	Solved	✓	✓	✗	✗	✗	✗	✗	✗
	Excess	0	792,900	N/A	N/A	N/A	N/A	N/A	N/A
	SubOpt	0	1.51	N/A	N/A	N/A	N/A	N/A	N/A
Hybrid (RoPE)	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	0	0	0	0	0	0	339	409
	SubOpt	0	0	0	0	0	0	2.02×10^{-5}	1.22×10^{-5}

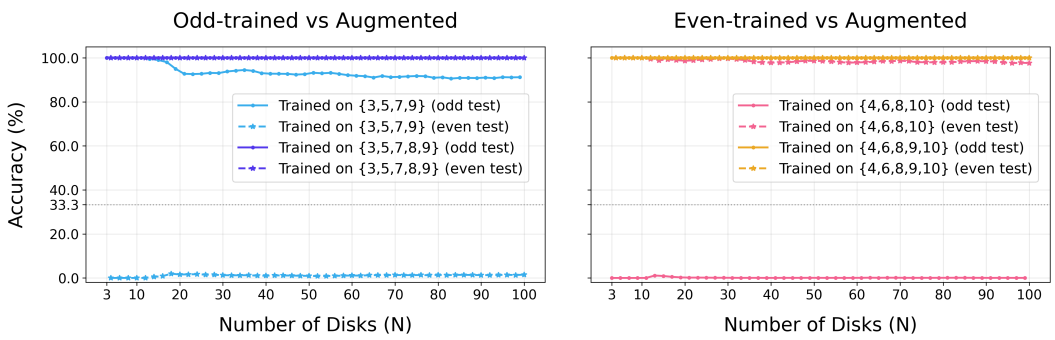

Figure 8: Effect of minimal parity supervision on length generalization. Left: odd-trained vs. odd-dominant (augmented with one even example). Right: even-trained vs. even-dominant (augmented with one odd example). Single-parity training leads to systematic failure on the opposite parity, resulting in near-zero accuracy and persistent oscillations. In contrast, adding a single opposite-parity example eliminates this failure mode and yields flat, near-perfect accuracy across all N.

While training length determines exposure to recursive depth, the Hanoi solution also depends on a global structural variable—parity—which governs the orientation of recursion. We next examine how parity coverage affects generalization.

B.5.2 Parity Dependence

The analysis in § 4.1 shows that the optimal Hanoi trajectory is governed by a parity-controlled cyclic structure: the parity $(N \bmod 2)$ determines the global orientation of the recursive solution. This suggests that successful generalization requires identifying this global phase variable.

Single-parity training. To test this hypothesis, we restrict the training distribution to a single parity. We train Hybrid-RoPE under three regimes:

(i) *odd-only*: $N \in \{3, 5, 7, 9\}$,
(ii) *even-only*: $N \in \{4, 6, 8, 10\}$,
(iii) *mixed-parity*: $N \in \{3, 4, 5, 6, 7, 8, 9, 10\}$.

As in § 5.1, we evaluate the local policy accuracy of the trained models π_θ for $N = 3$ to 100.

Figure 5 shows that single-parity training leads to persistent oscillatory behavior: the model achieves high accuracy on test instances whose parity matches the training data, but fails on the opposite parity. This oscillation persists even at very large N. In contrast, mixed-parity training eliminates the oscillation and yields stable generalization across all problem sizes.

These results indicate that parity acts as a global latent variable governing the recursive structure of the solution. When training data contains only a single parity, this variable is unidentifiable. As a result, the learned policy extrapolates a fixed orientation to all problem sizes, and produces systematic errors on unseen parity.

Imbalanced parity training. To further validate this interpretation, we introduce a minimal amount of opposite-parity supervision. Specifically, we augment each single-parity training regime with a single training length from the opposite parity:

(i) *odd-dominant*: $N \in \{3, 5, 7, 9\} \cup \{\mathbf{8}\}$,
(ii) *even-dominant*: $N \in \{4, 6, 8, 10\} \cup \{\mathbf{9}\}$.

Notably, these training sets remain highly imbalanced with respect to parity, containing only a single training length from the opposite regime.

Figure 8 compares these imbalanced regimes with their corresponding single-parity baselines. Under single-parity training, the model exhibits a clear failure mode: it achieves high accuracy on test instances whose parity matches the training data, but collapses on the opposite parity, resulting in persistent oscillations. In contrast, introducing a single training length from the opposite parity

is sufficient to eliminate this behavior. Both odd-dominant and even-dominant training yield flat, near-perfect accuracy across all tested problem sizes, with no observable oscillation.

This result confirms that the failure of single-parity training is due to a lack of identifiability of the global parity variable, rather than insufficient model capacity or optimization. Once this variable is weakly supervised, the model recovers the correct recursive structure and generalizes reliably to unseen problem sizes.

Key factors for recursive generalization

- **Recursive depth coverage.** Training must expose the model to sufficiently deep recursive compositions; otherwise, generalization fails beyond the training range.
- **Global phase (parity) identifiability.** Parity determines the orientation of recursion. Training on a single parity leads to systematic extrapolation errors.
- **Minimal supervision suffices.** A small number of examples from the opposite parity is sufficient to recover stable generalization.
- **Generalization requires both scale and structure.** Successful extrapolation depends not only on increasing problem size, but also on identifying global structural variables governing the solution.

B.6 Label Smoothing

We further study the effect of label smoothing on length generalization. For each training instance, we replace the one-hot next-action label with an ε-smoothed target distribution, assigning probability $1 - \varepsilon$ to the optimal action and distributing the remaining mass uniformly over the other two actions. We use $\varepsilon = 0.2$ throughout this experiment. This modification improves extrapolation for all evaluated architectures, suggesting that smoothing the supervision can reduce sensitivity to the finite training-length distribution.

One-step accuracy. Figure 9 reports forward next-step accuracy under label-smoothed supervision. Hybrid-RoPE remains the strongest model, achieving *exact zero-error prediction* across all tested lengths $N = 3$–100 on our sampled evaluation set. Hybrid-APE also generalizes substantially beyond the training regime, but its performance is consistently below Hybrid-RoPE and exhibits visible length-dependent fluctuations, with accuracy dropping to approximately 97–99% at several extrapolation lengths. The LSTM baseline likewise benefits from label smoothing, but still degrades progressively with increasing N, reaching below 50% accuracy at $N = 100$.

The improvement observed across architectures is consistent with a regularization effect of label smoothing. By discouraging overly confident predictions on the finite training lengths, label smoothing prevents the output distribution from becoming excessively saturated and may soften the learned decision boundaries. This can reduce sensitivity to length-specific patterns in the training data and make the learned policy more tolerant to the representation shifts encountered when extrapolating to larger N. We emphasize that this provides a plausible explanation for the observed improvement rather than a directly established causal mechanism.

Deployment. The distinction between the two hybrid models is especially important under closed-loop deployment. Solving an N-disk puzzle requires $2^N - 1$ sequential decisions, so even a small per-step error rate can accumulate into substantial deviations over exponentially long horizons. This is reflected in Tables 3 and 4. For the canonical source configuration $0^N \to 2^N$, Hybrid-RoPE recovers the exact optimal trajectory through all tested sizes up to $N = 26$, whereas Hybrid-APE begins to incur excess steps at larger N. The same qualitative pattern holds for the alternative source configuration $1^N \to 2^N$: Hybrid-RoPE remains essentially exact, while Hybrid-APE and LSTM accumulate substantially larger deviations. Thus, the advantage of relative positional encoding is visible not only in one-step prediction but also in long-horizon execution.

Label smoothing also yields observable gains for Hybrid-RoPE at the longest deployment horizons. Under one-hot supervision, the model exhibits slight suboptimality at $N = 24$ and $N = 25$, on the order of 10^{-5} relative to the optimal trajectory length (Table 2). With label-smoothed supervision, these residual deviations disappear for the canonical $0^N \to 2^N$ configuration, and the model follows

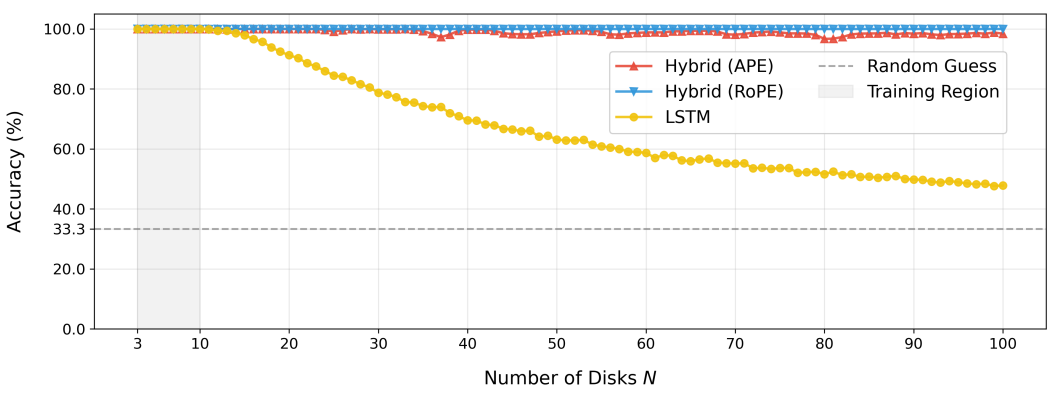

Figure 9: Forward next-step accuracy under label-smoothed supervision ($\varepsilon = 0.2$). Hybrid-RoPE maintains 100% accuracy across all tested lengths $N = 3$–100, while Hybrid-APE exhibits small but visible length-dependent degradation and LSTM progressively deteriorates outside the training range. The shaded region denotes the training lengths $N = 3$–10.

the exact optimal trajectory throughout the tested range. This indicates that label smoothing can improve deployment robustness even when one-step accuracy already appears saturated.

Overall, these results separate two complementary effects. First, label smoothing improves extrapolation stability across the evaluated architectures, but it does not eliminate their architectural differences: the LSTM still degrades substantially with length, while both hybrid models remain much more stable. Second, within the hybrid architecture, RoPE provides an additional and clearly observable advantage over absolute positional encoding. Hybrid-APE maintains high accuracy but exhibits persistent length-dependent errors, whereas Hybrid-RoPE remains exact across the full evaluation range. Together, these results suggest that stable recursive extrapolation benefits from both hierarchical recurrence and a positional representation aligned with the relative structure of the Hanoi recursion, with label smoothing serving as an additional regularizer rather than replacing these structural inductive biases.

Table 3: Deterministic cycle-aware full-puzzle rollout results for models trained with ϵ-noisy supervision ($\epsilon = 0.2$). This table reports excess steps and relative suboptimality for the canonical source configuration $0^N \to 2^N$.

(a) $N = 11$–17 (All Models)

Model(s)	**Metric**	11	12	13	14	15	16	17	18
Optimal	Steps	2,047	4,095	8,191	16,383	32,767	65,535	131,071	262,143
LSTM	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	7	14	283	687	4,327	7,550	57,986	70,147
	SubOpt	3.42×10^{-3}	3.42×10^{-3}	3.46×10^{-2}	4.19×10^{-2}	0.132	0.115	0.442	0.268
Hybrid (APE)	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	0	0	0	0	0	0	0	0
	SubOpt	0	0	0	0	0	0	0	0
Hybrid (RoPE)	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	0	0	0	0	0	0	0	0
	SubOpt	0	0	0	0	0	0	0	0

(b) $N = 18$–26 (Hybrid Comparison)

Model	**Metric**	19	20	21	22	23	24	25	26
Optimal	Steps	524,287	1,048,575	2,097,151	4,194,303	8,388,607	16,777, 215	33,554,431	67,108,863
LSTM	Solved	✓	✓	✓	✓	✗	✗	✗	✗
	Excess	629,724	898,715	11,589,895	25,304,065	N/A	N/A	N/A	N/A
	SubOpt	1.20	0.857	5.53	6.03	N/A	N/A	N/A	N/A
Hybrid (APE)	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	0	0	0	0	143	4,182	23,102	5,701
	SubOpt	0	0	0	0	1.70×10^{-5}	2.49×10^{-4}	6.88×10^{-4}	8.50×10^{-5}
Hybrid (RoPE)	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	0	0	0	0	0	0	0	0
	SubOpt	0	0	0	0	0	0	0	0

Table 4: Deterministic cycle-aware full-puzzle rollout results for models trained with ϵ-noisy supervision ($\epsilon = 0.2$). This table reports excess steps and relative suboptimality for the alternative source configuration $1^N \to 2^N$.

(a) $N = 11$–17 (All Models)

Model(s)	**Metric**	11	12	13	14	15	16	17	18
Optimal	Steps	2,047	4,095	8,191	16,383	32,767	65,535	131,071	262,143
LSTM	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	0	27	337	553	3,364	11,127	39,574	125,162
	SubOpt	0	6.59×10^{-3}	4.11×10^{-2}	3.38×10^{-2}	0.103	0.170	0.302	0.477
Hybrid (APE)	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	0	0	0	0	0	0	0	22
	SubOpt	0	0	0	0	0	0	0	8.39×10^{-5}
Hybrid (RoPE)	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	0	0	0	0	0	0	0	0
	SubOpt	0	0	0	0	0	0	0	0

(b) $N = 18$–25 (Hybrid Comparison)

Model	**Metric**	19	20	21	22	23	24	25	26
Optimal	Steps	524,287	1,048,575	2,097,151	4,194,303	8,388,607	16,777, 215	33,554,431	67,108,863
LSTM	Solved	✓	✓	✓	✓	✗	✗	✗	✗
	Excess	398,569	1,303,577	12,897,510	21,037,722	N/A	N/A	N/A	N/A
	SubOpt	0.760	1.24	6.15	5.02	N/A	N/A	N/A	N/A
Hybrid (APE)	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	212	4,546	5,644	27,885	152,125	1,044,756	567,924	2,679,372
	SubOpt	4.04×10^{-4}	4.34×10^{-3}	2.69×10^{-3}	6.65×10^{-3}	1.81×10^{-2}	6.23×10^{-2}	1.69×10^{-2}	3.99×10^{-2}
Hybrid (RoPE)	Solved	✓	✓	✓	✓	✓	✓	✓	✓
	Excess	0	0	0	0	2	2	3	2
	SubOpt	0	0	0	0	2.38×10^{-7}	1.19×10^{-7}	8.94×10^{-8}	2.98×10^{-8}

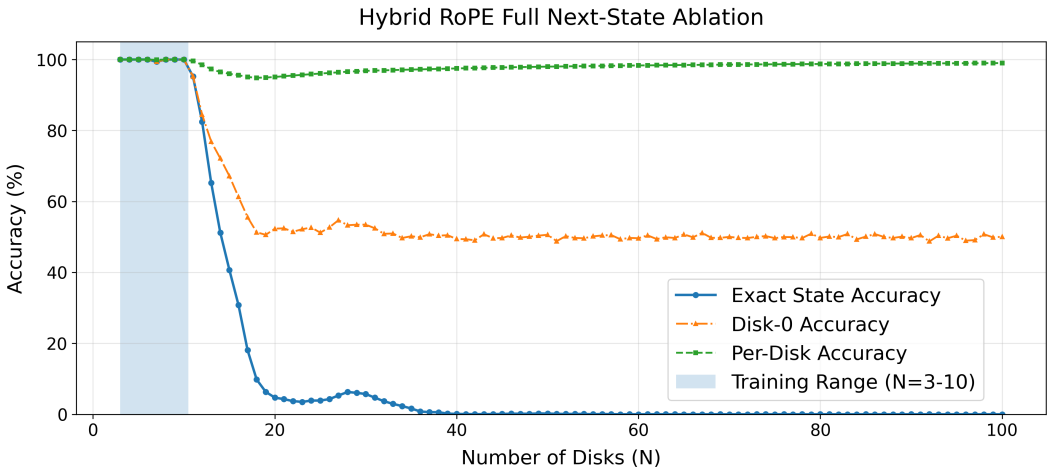

Figure 10: Full next-state prediction ablation for Hybrid-RoPE. The model predicts the next peg of every disk independently, without the structured disk-0 decoding rule. Exact-state accuracy rapidly collapses beyond the training range, while disk-0 accuracy approaches 50% and per-disk accuracy approaches the identity-prediction baseline $1 - 1/N$.

B.7 Additional Ablation Studies

B.7.1 Full Next-State Prediction Ablation

Our main formulation exploits a structural property of the three-peg Tower of Hanoi: the optimal successor can be uniquely determined from the predicted next position of the smallest disk together with the puzzle constraints. Accordingly, the main model is supervised only on disk 0, reducing next-step prediction to a three-way classification problem. To examine the importance of this structured prediction interface, we perform an ablation in which Hybrid-RoPE instead predicts the complete successor state directly.

We retain the same Hybrid-RoPE architecture and training setup, but apply the next-state classification objective independently to all N disk tokens. At inference time, the peg of each disk is obtained by taking the argmax of its three output logits, yielding an unconstrained prediction

$$\hat{\mathbf{x}}' = (\hat{x}'_0, \ldots, \hat{x}'_{N-1}).$$

No disk-0 decoding rule, legality constraint, or other post-processing is applied. We report three metrics: *exact-state accuracy*, which requires all N predicted disk positions to match the oracle successor; *disk-0 accuracy*; and *per-disk accuracy*, averaged over all individual disk predictions.

Figure 10 shows that the full-state model fits the training regime almost perfectly but generalizes poorly beyond it. Exact-state accuracy decreases from 95.25% at $N = 11$ to 65.23% at $N = 13$, 30.82% at $N = 16$, and below 10% by $N = 18$. It becomes effectively zero for larger problem sizes. Disk-0 accuracy exhibits a similar initial degradation and then stabilizes near 50% for large N.

In contrast, per-disk accuracy remains above 94% throughout the extrapolation regime and eventually increases toward 99%. This apparently strong token-level performance is misleading because a Hanoi transition changes the position of exactly one disk. Therefore, the trivial identity predictor

$$\hat{\mathbf{x}}' = \mathbf{x}$$

already achieves per-disk accuracy

$$\frac{N-1}{N} = 1 - \frac{1}{N},$$

despite never predicting a move correctly. The large-N behavior of the trained model closely approaches this baseline. For example, at $N = 100$, its per-disk accuracy is exactly 99%, while both exact-state accuracy and moved-disk accuracy are 0%. Likewise, its disk-0 accuracy is approximately 50%, consistent with copying the current disk-0 position because disk 0 moves on every other step of

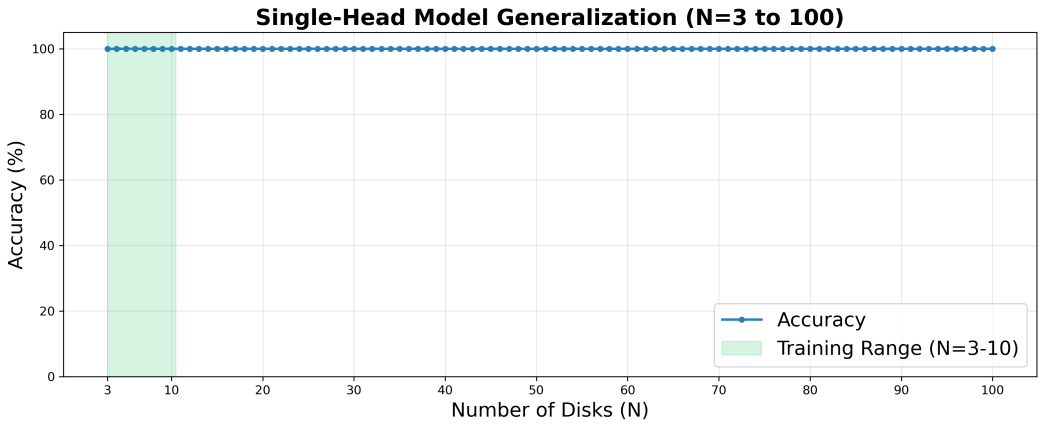

Figure 11: One-step policy accuracy of the Hybrid (RoPE) model with a single attention head. The model achieves near-perfect accuracy across all tested problem sizes ($N = 3$ to 100), matching the performance of the multi-head variant. The shaded region indicates the training range ($N \in [3, 10]$).

the optimal trajectory. These results indicate that under length extrapolation, the full-state predictor increasingly defaults to preserving the input configuration rather than identifying the disk that should move.

This failure mode arises from the strong imbalance inherent in full-state supervision. Each transition contains $N - 1$ unchanged disk labels but only one changed label, so as N increases the training and prediction problem becomes increasingly dominated by state preservation. The disk-0 formulation removes this redundancy and focuses supervision on the minimal decision sufficient to identify the successor move. The ablation therefore shows that the strong extrapolation of our main model depends not only on the hierarchical and relative-position inductive biases of Hybrid-RoPE, but also on using an output representation aligned with the sparse transition structure of the Hanoi dynamics.

B.7.2 Effect of Attention Head Number

In the main experiments, all Transformer-based models use multi-head self-attention with four heads. To assess whether multi-head attention is necessary for length generalization, we conduct an additional experiment using a single attention head.

Specifically, we train the Hybrid (RoPE) model with a single-head Transformer encoder, keeping all other architectural and training settings unchanged. The model is evaluated using the same one-step policy accuracy protocol as in §5.1.

As shown in Figure 11, the single-head model achieves near-perfect accuracy across all tested problem sizes (up to $N = 100$), matching the performance of the multi-head variant.

This result indicates that increasing the number of attention heads is not a determining factor for generalization in this task. Instead, generalization is primarily driven by the structural inductive biases introduced by relative positional encoding and hierarchical recurrence.

C The Optimal Solution of the Tower of Hanoi Puzzle

C.1 Illustrative Examples for State Representation and Action Decoding

This subsection complements the formulation in §3.2 by providing two concrete illustrations: one for the state–action representation of Tower of Hanoi, and one for the `disk 0`-based decoding rule used throughout the paper.

State representation and deterministic transition. As described in §3.2, we represent a puzzle configuration by a state vector

$$\mathbf{x} = (x_0, x_1, \cdots, x_{N-1}) \in \{0, 1, 2\}^N,$$

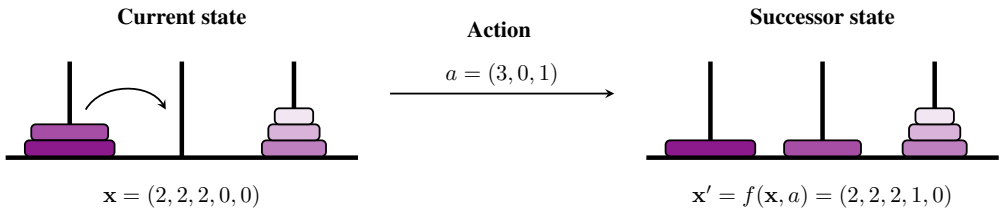

Figure 12: Illustration of the Tower of Hanoi as a deterministic Markov Decision Process (MDP). Each configuration is represented by a state vector $\mathbf{x} \in \{0, 1, 2\}^N$, where x_i denotes the peg index of `disk i`. An action $a = (i, s, t)$ corresponds to moving `disk i` from `peg s` to `peg t`, subject to the legality constraints of the puzzle. The transition dynamics are deterministic, so each state-action pair uniquely determines the successor state $\mathbf{x}'$. In this 5-disk example, action $a = (3, 0, 1)$ moves `disk 3` from `peg 0` to `peg 1`, resulting in the transition from $\mathbf{x} = (2, 2, 2, 0, 0)$ to $\mathbf{x}' = (2, 2, 2, 1, 0)$.

where x_i denotes the peg occupied by `disk i`, with disks indexed in increasing order of size. Because disks on each peg are always ordered by size, this representation uniquely determines a valid configuration. A legal move is specified by an action

$$a = (i, s, t),$$

meaning that `disk i` is moved from peg s to peg t. The transition is deterministic, so each valid state–action pair uniquely determines a successor state

$$\mathbf{x}' = f(\mathbf{x}, a).$$

Figure 12 gives an example of such a transition.

Why `disk 0` is sufficient for decoding. In the main text, we formulate policy learning as next-state prediction and decode the optimal action using only the predicted next peg of `disk 0`. Figure 13 illustrates why this is possible.

For a given state $\mathbf{x}$, let

$$\mathcal{S}(\mathbf{x}) = \{\mathbf{x}' : \mathbf{x}' = f(\mathbf{x}, a),\ a \text{ is legal}\}$$

denote the set of legal successor states. Although the full state space has size 3^N, the set $\mathcal{S}(\mathbf{x})$ is always small, with at most three legal successors. The key observation is that the next move is uniquely determined by the behavior of the smallest disk:

- If the predicted peg of `disk 0` changes, then `disk 0` itself must be the moved disk.
- If the predicted peg of `disk 0` remains unchanged, then the move is the unique legal move among the remaining disks.

Hence the successor state can be decoded solely from whether `disk 0` moves and, if so, to which peg. This is exactly the structural reduction used in §3.2; Figure 13 visualizes the two cases.

Implementation detail for fully stacked states. The `disk 0` decoding rule described in §3.2 assumes that when the predicted next peg of `disk 0` is unchanged, there is a unique legal move among the remaining disks. This holds for all non-stacked states on the optimal trajectory, but fails at fully stacked states such as the canonical initial configuration, where every legal move necessarily involves `disk 0`. In deployment, we handle this edge case by masking the invalid "`disk 0` stays" branch, equivalently assigning it score $-\infty$ before action selection. We do not apply any special masking during training. In practice, this case is rarely triggered because the learned models achieve high local accuracy and correctly predict that `disk 0` moves at such states.

C.2 Proof of Theorems 1 and 2

In the main text, we present Theorem 1 and Theorem 2, which state the structural results governing parity dependence in the 3-peg Tower of Hanoi and emphasizes the recursive orientation and its learning implications. In this appendix, we provide a fully formal proof of the results.

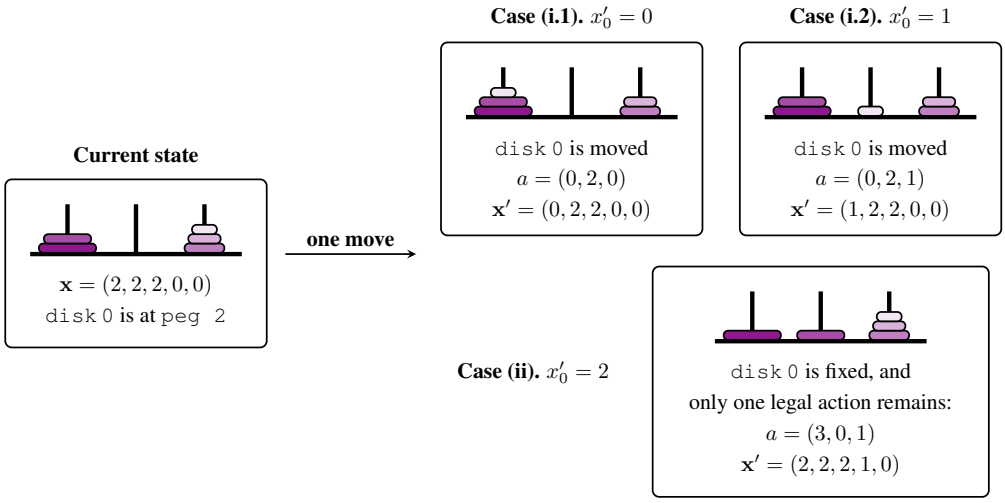

Figure 13: An example for why tracking `disk 0` suffices to decode the next action. From the current state $\mathbf{x} = (2, 2, 2, 0, 0)$, there are exactly three legal successor states. If the predicted peg of `disk 0` changes, then `disk 0` itself must be the moved disk, which uniquely determines the action. If the predicted peg of `disk 0` remains unchanged, then `disk 0` is fixed, and only one legal move remains. Thus the next action can be decoded solely from the predicted successor location of `disk 0`.

Notation. Prior to stating the structural result, we briefly clarify the group-theoretic notation. We denote by S_3 the *symmetric group* acting on the set $\{0, 1, 2\}$, i.e., the group of all permutations of the three peg labels. We use cycle notation: for example, (012) denotes the 3-cycle mapping $0 \mapsto 1$, $1 \mapsto 2$, and $2 \mapsto 0$, while (021) denotes the reversed cycle $0 \mapsto 2$, $2 \mapsto 1$, and $1 \mapsto 0$. A *transposition*, such as (01), swaps two peg labels and fixes the third. For $\tau, \sigma \in \mathrm{S}_3$, the conjugation $\tau^{-1} \circ \sigma \circ \tau$ represents relabeling the pegs by τ, applying σ, and then undoing the relabeling.

The formulation below is mathematically equivalent to Theorem 1 in the main text, but makes the recursive decomposition and the conjugation argument in S_3 explicit. No additional assumption is introduced; we simply expand the structural argument into a complete inductive proof.

Theorem 3 (Parity determines recursive orientation)**.** *Consider the Tower of Hanoi puzzle with 3 pegs labeled $\{0, 1, 2\}$. Let $N \in \mathbb{N}$ denote the number of disks, and index the disks by $\{0, 1, \cdots, N-1\}$ in increasing size (i.e., `disk 0` is the smallest and disk $N-1$ is the largest). All disks are initially placed on peg 0, and the goal state places all disks on peg 2.*

Let $\mu^\star$ denote the optimal move sequence of length $2^N - 1$, and let S_3 denote the symmetric group acting on the peg set $\{0, 1, 2\}$. There exists a permutation $\sigma_N \in \mathrm{S}_3$ such that:

- *(a)* ***Interlacing structure.*** *Along the optimal move sequence $\mu^\star$, `disk 0` moves on every odd-numbered step and remains fixed on every even-numbered step.*
- *(b)* ***Parity-controlled cycle.*** *The trajectory of `disk 0` follows the 3-cycle σ_N. More specifically, let $p_t \in \{0, 1, 2\}$ denote the peg location of `disk 0` after t moves along $\mu^\star$, with $p_0 = 0$. Then*
$$p_{2k+1} = \sigma_N(p_{2k}), \qquad p_{2k+2} = p_{2k+1},$$
where
$$\sigma_N = \begin{cases} (012), & \text{if } N \text{ is even}, \\ (021), & \text{if } N \text{ is odd}. \end{cases}$$
In particular, the cyclic orientation of `disk 0` depends only on $N \bmod 2$.
- *(c)* ***Recursive conjugation.*** *The permutation σ_N satisfies*
$$\sigma_N = \tau^{-1} \circ \sigma_{N-1} \circ \tau,$$
where τ is a transposition in S_3. Thus the orientation flips at each recursive depth.
- *(d)* ***Uniqueness.*** *The entire optimal move sequence $\mu^\star$ is uniquely determined by the disk-0 trajectory $(p_0, p_1, \cdots, p_{2^N-1})$.*

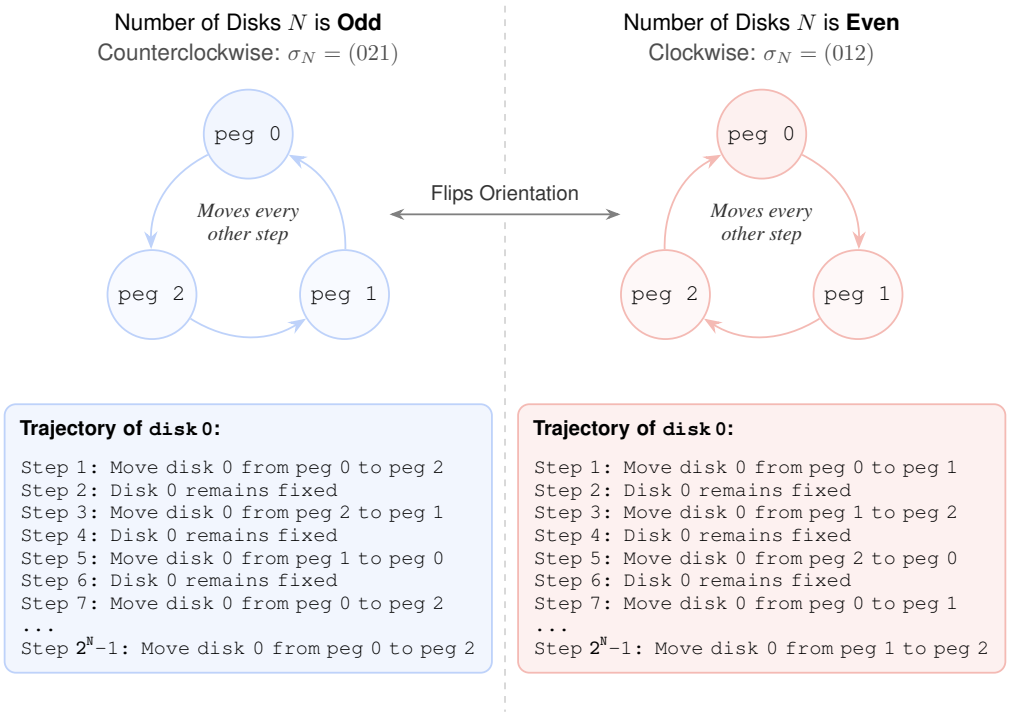

Figure 14: Parity-controlled cycle of the smallest disk in the optimal Tower of Hanoi solution. `Disk 0` moves every other step and follows a fixed cyclic trajectory over the three pegs. For odd N, the cycle is (021); for even N, it reverses to (012). The lower panels show representative moves from optimal solutions illustrating that `disk 0` consistently follows the parity-dependent cycle.

Proof of Theorem 3. We proceed by induction on N.

Step I: Interlacing. We first prove that the optimal move sequence has an interlacing structure: *`disk 0` moves on every odd-numbered move, and stays fixed on every even-numbered move.*

- For $N = 1$, the solution consists of a single move moving `disk 0` from peg 0 to peg 2, so the claim holds trivially.
- Consider the optimal procedure that moves a stack of N disks from peg s to peg t using the remaining peg $u \in \{0, 1, 2\} \setminus \{s, t\}$ as the spare. The classical recursion is

$$\text{Hanoi}(N; s \to t) = \big[\underbrace{\text{Hanoi}(N-1; s \to u)}_{\text{Subproblem 1}} \,||\, (s \to t) \,||\, \underbrace{\text{Hanoi}(N-1; u \to t)}_{\text{Subproblem 2}}\big],$$

  where $(s \to t)$ denotes the single move of the largest disk from peg s to peg t.

  Assume the statement holds for $N - 1$. The optimal move sequences for both sub-problems have length $2^{N-1} - 1$, which is odd. By the induction hypothesis, within each subproblem `disk 0` moves exactly on odd-numbered steps (relative to that subproblem). Since the middle step (moving disk $N - 1$ from peg s to peg t) occurs at the even global index 2^{N-1}, `disk 0` moves precisely on odd-numbered global steps and never on even-numbered steps.

This proves the interlacing structure, and we obtain that $p_{2k+2} = p_{2k+1}$.

Step II: Orientation of the cycle. In the recursive decomposition, the first subproblem swaps the roles of destination and spare pegs relative to the $(N-1)$-disk problem. We let $\tau_{12} = (12) \in \mathrm{S}_3$, i.e. τ_{12} fixes 0 and swaps $1, 2$. Then

$$\tau_{12}(p_{2k+1}) = (\sigma_{N-1} \circ \tau_{12})(p_{2k}), \quad k = 0, 1, \cdots, 2^{N-2} - 1.$$

Similarly, the second subproblem swaps the roles of source and spare pegs relative to the $(N-1)$-disk problem. Then

$$\tau_{01}(p_{2k+1}) = (\sigma_{N-1} \circ \tau_{01})(p_{2k}), \quad k = 2^{N-2}, 2^{N-2} + 1, \cdots, 2^{N-1} - 1,$$

where $\tau_{01} = (01) \in \mathrm{S}_3$ swaps $0, 1$ and fixes 2. Hence

$$p_{2k+1} = \begin{cases} (\tau_{12}^{-1} \circ \sigma_{N-1} \circ \tau_{12})(p_{2k}), & \text{if } k = 0, 1, \cdots, 2^{N-2} - 1, \\ (\tau_{01}^{-1} \circ \sigma_{N-1} \circ \tau_{01})(p_{2k}), & \text{if } k = 2^{N-2}, 2^{N-2} + 1, \cdots, 2^{N-1} - 1. \end{cases}$$

In the symmetric group S_3, one verifies

$$(01)(012)(01) = (12)(012)(12) = (021), \quad \text{and} \quad (01)(021)(01) = (12)(021)(12) = (012).$$

Since the base case $N = 1$ yields the cycle $\sigma_1 = (021)$, it follows inductively that

$$\sigma_N = \tau_{12}^{-1} \circ \sigma_{N-1} \circ \tau_{12} = \tau_{01}^{-1} \circ \sigma_{N-1} \circ \tau_{01} = \begin{cases} (021) & \text{if } N \text{ is odd}, \\ (012) & \text{if } N \text{ is even}. \end{cases}$$

with $p_{2k+1} = \sigma_N(p_{2k})$ for $k = 0, 1, \cdots, 2^{N-1} - 1$.

Step III: Reconstruction. At each odd step, `disk 0` moves according to $p_{2m} \mapsto p_{2m+1}$. At each even step, `disk 0` remains fixed. At such states, exactly one legal move exists among the remaining disks: among the two pegs not containing `disk 0`, the smaller exposed disk must move onto the larger (or empty) peg. Therefore the non-disk-0 move is uniquely determined. Thus the entire move sequence $\mu^\star$ is uniquely reconstructed from $\{p_t\}$. □

Theorem 3 characterizes the behavior of the smallest disk and shows that its motion follows a parity-controlled 3-cycle. This phenomenon is not unique to `disk 0`. In fact, every disk exhibits a similar cyclic structure, with the period doubling as the disk size increases, and the orientation determined by the parity of its effective recursion depth.

More precisely, disk i moves periodically with period 2^{i+1}, and whenever it moves, its peg location advances along a 3-cycle whose direction depends only on $N - i$. Thus the entire optimal Hanoi solution can be viewed as a hierarchy of coupled cyclic processes, where each disk operates at its own frequency and orientation. The following theorem is the same Theorem 2 in the main text, which formalizes this generalization.

Theorem 4 (Per-disk periodicity and parity-controlled cycles)**.** *Consider the Tower of Hanoi puzzle with 3 pegs labeled $\{0, 1, 2\}$. Let $N \in \mathbb{N}$ denote the number of disks, and index the disks by $\{0, 1, \cdots, N-1\}$ in increasing size (i.e., `disk 0` is the smallest and disk $N-1$ is the largest). All disks are initially placed on peg 0, and the goal state places all disks on peg 2.*

Denote by $\mu^\star$ the optimal move sequence of length $2^N - 1$. Fix a disk index $i \in \{0, 1, \cdots, N-1\}$. Let $m_i(1) < m_i(2) < \cdots < m_i(2^{N-i-1})$ be the move indices at which disk i is moved along $\mu^\star$ (so disk i moves exactly 2^{N-i-1} times). Then:

(a) ***Periodicity.*** *The move times of disk i are*

$$m_i(k) = 2^i(2k - 1), \qquad k = 1, 2, \cdots, 2^{N-i-1}.$$

In particular, disk i moves once every 2^{i+1} steps.

(b) ***Parity-controlled cycle.*** *Let $p_t^{(i)} \in \{0, 1, 2\}$ be the peg location of disk i after t moves along $\mu^\star$. Define the 3-cycle*

$$\sigma_{N-i} = \begin{cases} (012), & \textit{if } N - i \textit{ is even}, \\ (021), & \textit{if } N - i \textit{ is odd}. \end{cases}$$

Then disk i follows this cycle at its move times:

$$p_{m_i(k)}^{(i)} = \sigma_{N-i}\left(p_{m_i(k)-1}^{(i)}\right), \qquad k = 1, 2, \cdots, 2^{N-i-1}.$$

Equivalently, if $q_i(k) := p_{m_i(k)}^{(i)}$ denotes the peg of disk i immediately after its k-th move, then

$$q_i(k) = \sigma_{N-i}^k(0), \qquad k = 1, 2, \cdots, 2^{N-i-1}.$$

Proof. We prove the statement by induction on N for a fixed disk index i.

Base case $N = i + 1$**.** Disk i is the largest disk. The optimal sequence consists of moving the stack of i smaller disks from peg 0 to peg 1, then moving disk i once, and finally moving the smaller stack from peg 1 to peg 2. Thus disk i moves exactly once at step 2^i, i.e. $m_i(1) = 2^i = 2^i(2 \cdot 1 - 1)$, establishing (a). Moreover, $N - i = 1$ is odd so $\sigma_{N-i} = (021)$, and disk i moves from peg 0 to peg 2, which is exactly the action of (021) applied to 0. Hence (b) holds.

Inductive step. Assume $N \geq i + 2$, and the theorem holds for the puzzle with $(N - 1)$ disks and the same index i. Consider the optimal solution for N disks from peg 0 to peg 2 with spare peg 1. The standard recursion is

$$\mathrm{Hanoi}(N; 0 \to 2) = \big[\mathrm{Hanoi}(N-1; 0 \to 1) \;||\; (0 \to 2) \;||\; \mathrm{Hanoi}(N-1; 1 \to 2)\big],$$

where $(0 \to 2)$ is the single move of disk $N - 1$ at global step 2^{N-1}.

Step 1: periodicity [part (a)]. As we have seen, disk i (with $i \leq N - 2$) moves only inside the two $(N - 1)$-disk subproblems. Let $T_{N,i}^{(1)}$ be the set of global move indices at which disk i moves in the first subproblem, and $T_{N,i}^{(2)}$ the analogous set for the second subproblem. By the induction hypothesis applied to the $(N - 1)$-disk problem, within each subproblem disk i moves at local indices

$$2^i(2k - 1), \qquad k = 1, 2, \cdots, 2^{N-i-2}.$$

Therefore,

$$T_{N,i}^{(1)} = \{2^i(2k - 1) \;:\; 1 \leq k \leq 2^{N-i-2}\}.$$

The second subproblem starts after 2^{N-1} moves, hence its global indices are shifted by 2^{N-1}:

$$T_{N,i}^{(2)} = \{2^{N-1} + 2^i(2k - 1) \;:\; 1 \leq k \leq 2^{N-i-2}\}.$$

Since $i \leq N - 2$, we have $2^{N-1} = 2^{i+1} \cdot 2^{N-i-2}$, so

$$2^{N-1} + 2^i(2k - 1) = 2^i\big(2^{N-i-1} + 2k - 1\big) = 2^i\big(2(2^{N-i-2} + k) - 1\big).$$

Thus $T_{N,i}^{(2)}$ is exactly the continuation of the same arithmetic progression, and

$$T_{N,i}^{(1)} \cup T_{N,i}^{(2)} = \{2^i(2k - 1) \;:\; 1 \leq k \leq 2^{N-i-1}\}.$$

This proves (a).

Step 2: parity-controlled cycle [part (b)]. We track the *peg labels* relevant to disk i across the two subproblems. The key fact is that each $(N - 1)$-disk subproblem is the same Hanoi problem up to a relabeling (permutation) of the pegs, and such relabeling acts by conjugation in S_3, as we discussed before. Let $\sigma_{N-1,i}$ denote the 3-cycle governing disk i in an $(N - 1)$-disk problem in the canonical setting (source 0, target 2). By the induction hypothesis, this cycle is

$$\sigma_{N-1,i} = \sigma_{N-1-i}.$$

Now we track the cyclic pattern in two subproblems.

- *First subproblem.* The procedure $\mathrm{Hanoi}(N - 1; 0 \to 1)$ is obtained from the canonical problem $\mathrm{Hanoi}(N - 1; 0 \to 2)$ by swapping peg labels 1 and 2. Let $\tau_{12} = (12) \in \mathrm{S}_3$. Under this relabeling, the cycle governing disk i becomes

$$\sigma_{N,i}^{(1)} = \tau_{12}^{-1} \circ \sigma_{N-1-i} \circ \tau_{12}.$$

- *Second subproblem.* $\mathrm{Hanoi}(N - 1; 1 \to 2)$ is obtained from the canonical problem by swapping peg labels 0 and 1. Let $\tau_{01} = (01) \in \mathrm{S}_3$. The cycle becomes

$$\sigma_{N,i}^{(2)} = \tau_{01}^{-1} \circ \sigma_{N-1-i} \circ \tau_{01}.$$

Recall that in S_3, conjugating a 3-cycle by any transposition reverses orientation:

$$\tau^{-1} \circ (012) \circ \tau = (021), \qquad \tau^{-1} \circ (021) \circ \tau = (012), \qquad \text{where } \tau \text{ is a transposition.}$$

Hence both $\sigma_{N,i}^{(1)}$ and $\sigma_{N,i}^{(2)}$ equal the *orientation-reversed* cycle of $\sigma_{(N-1)-i}$, i.e.

$$\sigma_{N,i}^{(1)} = \sigma_{N,i}^{(2)} = \sigma_{N-i}.$$

Now we concatenate the two subproblems together. Let $K = 2^{N-i-2}$, so that disk i moves K times in each of the two $(N-1)$-disk subproblems. By the induction hypothesis, during the first subproblem its successive positions satisfy

$$q_i(k) = \sigma_{N-i}^k(0), \qquad k = 1, \cdots, K.$$

Disk i does not move during the middle move of disk $N-1$, so its peg at the start of the second subproblem is still

$$q_i(K) = \sigma_{N-i}^K(0).$$

Since the second subproblem is governed by the same cycle σ_{N-i}, after $\ell = 1, \cdots, K$ additional moves of disk i we have

$$q_i(K+\ell) = \sigma_{N-i}^\ell\big(q_i(K)\big) = \sigma_{N-i}^{K+\ell}(0).$$

Hence the two subproblems concatenate without phase shift, and therefore

$$q_i(k) = \sigma_{N-i}^k(0), \qquad k = 1, \cdots, 2^{N-i-1}.$$

This proves (b), completing the induction. □

The preceding results characterize the cyclic structure of the optimal trajectory, including the parity-controlled orientation and the periodic motion of individual disks. However, these results are naturally indexed by the move counter t (e.g., "`disk 0` moves on every other step", "disk i moves every 2^{i+1} steps"). In practice, one may only observe a configuration x that is known to lie on the canonical optimal path, while the associated time stamp t is not available. In this setting, directly leveraging the cycle laws is nontrivial: without knowing where x sits within the interlaced schedule, it is ambiguous whether the next move is the disk-0 move or the unique non-disk-0 move.

Crucially, resolving this ambiguity does not require reconstructing the full time index. It suffices to determine the parity of t, since the interlacing property implies that `disk 0` moves exactly on one parity class of steps. This motivates the following lemma, which constructs a simple *parity discriminant* $D_N(\mathbf{x})$ whose parity agrees with the time parity along the optimal trajectory and can be computed directly from the state x.

Lemma 5 (Discriminant for time parity)**.** *Consider the Tower of Hanoi puzzle with 3 pegs labeled $\{0,1,2\}$. Let $N \in \mathbb{N}$ denote the number of disks, and index the disks by $\{0,1,\cdots,N-1\}$ in increasing size (i.e., `disk 0` is the smallest and disk $N-1$ is the largest). All disks are initially placed on peg 0, and the goal state places all disks on peg 2. Let $\mathbf{x}(t) = (x_0(t), \cdots, x_{N-1}(t)) \in \{0,1,2\}^N$ be the canonical optimal trajectory of this puzzle.*

Define the parity discriminant $D_N : \{0,1,2\}^N \to \mathbb{Z}$ *by*

$$D_N(\mathbf{x}) \;:=\; \mathbf{1}\{x_{N-1} = 2\} \;+\; \sum_{i=0}^{N-2} \mathbf{1}\{x_i = x_{i+1}\}. \tag{2}$$

Then along the canonical optimal path, the parity of D_N flips at every move:

$$D_N(\mathbf{x}(t+1)) \equiv D_N(\mathbf{x}(t)) + 1 \pmod 2, \qquad t = 0, 1, \cdots, 2^N - 2. \tag{3}$$

Consequently,

$$D_N(\mathbf{x}(t)) \equiv t + N - 1 \pmod 2. \tag{4}$$

In particular,

$$D_N(\mathbf{x}(t)) \equiv \begin{cases} t \pmod 2 & \text{if } N \text{ is odd,} \\ t+1 \pmod 2 & \text{if } N \text{ is even.} \end{cases}$$

Proof of Lemma 5. We prove equation 3 by induction on N.

Base case $N = 1$**.** There is a single disk, so $D_N(\mathbf{x}) = \mathbf{1}\{x_0 = 2\}$. Along the optimal path, x_0 moves $0 \to 2$ in one step, so the parity of D_N flips once. Thus equation 3 holds.

Inductive step. Assume equation 3 holds for problem size $N-1$ (canonical source 0, target 2). Consider the size-N optimal trajectory from peg 0 to peg 2 with spare peg 1. Write the standard concatenation:

$$\mathrm{Hanoi}(N;0\to 2)=\big[\underbrace{\mathrm{Hanoi}(N-1;0\to 1)}_{\text{Phase I}}\ ||\ \underbrace{(0\to 2)}_{\text{middle move of disk } N\text{–}1}\ ||\ \underbrace{\mathrm{Hanoi}(N-1;1\to 2)}_{\text{Phase II}}\big].$$

We show that D_N flips on every move within Phase I, on the middle move, and within Phase II.

1) The middle move flips the parity of D_N. Immediately before the middle move, all smaller disks are stacked on peg 1, so

$$x_0=x_1=\cdots=x_{N-2}=1,\qquad x_{N-1}=0.$$

Immediately after the middle move, disk $N-1$ moves to peg 2 while x_{N-2} remains 1, hence

$$x_0=x_1=\cdots=x_{N-2}=1,\qquad x_{N-1}=2.$$

Therefore, among the summands defining D_N in equation 2, only the term $\mathbf{1}\{x_{N-1}=2\}$ changes from 0 to 1, while $\mathbf{1}\{x_{N-2}=x_{N-1}\}=0$ both before and after. Hence D_N increases by 1 and thus its parity flips at this move.

2) Every move in Phase I flips the parity of D_N. During Phase I, disk $N-1$ stays fixed at peg 0, hence the term $\mathbf{1}\{x_{N-1}=2\}$ is constantly 0. Thus

$$D_N(\mathbf{x})=\sum_{i=0}^{N-3}\mathbf{1}\{x_i=x_{i+1}\}\ +\ \mathbf{1}\{x_{N-2}=x_{N-1}\},\qquad \text{with } x_{N-1}\equiv 0.$$

Consider the auxiliary statistic on the first $N-1$ disks (indices $0,\cdots,N-2$):

$$D_{N-1}(y):=\mathbf{1}\{y_{N-2}=2\}+\sum_{i=0}^{N-3}\mathbf{1}\{y_i=y_{i+1}\},$$

which is exactly the statistic D_{N-1} for an $(N-1)$-disk problem.

Let $\mathbf{y}(t):=(x_0(t),\cdots,x_{N-2}(t))$ be the restriction of the state $\mathbf{x}(t)=(x_0(t),x_1(t),\cdots,x_{N-1}(t))$ to the disks $0,1,\cdots,N-2$ during Phase I. Note that Phase I is precisely the optimal trajectory of $\mathrm{Hanoi}(N-1;0\to 1)$, which is obtained from the canonical $\mathrm{Hanoi}(N-1;0\to 2)$ trajectory by swapping peg labels 1 and 2.

Let $\tau=(12)\in \mathrm{S}_3$ act on peg labels, and $\tilde{\mathbf{y}}(t):=\tau(\mathbf{y}(t))$, where τ is applied coordinatewise. Then $t\mapsto\tilde{\mathbf{y}}(t)$ evolves exactly as the canonical $\mathrm{Hanoi}(N-1;0\to 2)$ trajectory. By the induction hypothesis, we have

$$D_{N-1}(\tilde{\mathbf{y}}(t+1))\equiv D_{N-1}(\tilde{\mathbf{y}}(t))+1\pmod 2\qquad\text{for every move in Phase I.}$$

It remains to relate $D_N(\mathbf{x})$ to $D_{N-1}(\tilde{\mathbf{y}})$. Because $x_{N-1}\equiv 0$ in Phase I, we have

$$\mathbf{1}\{x_{N-2}=x_{N-1}\}=\mathbf{1}\{x_{N-2}=0\}\quad\text{and}\quad\mathbf{1}\{\tilde{y}_{N-2}=2\}=\mathbf{1}\{\tau(x_{N-2})=2\}=\mathbf{1}\{x_{N-2}=1\}.$$

Along Phase I, disk $N-2$ moves only from pegs 0 to 1 and never visits peg 2. Hence

$$\mathbf{1}\{x_{N-2}=x_{N-1}\}=\mathbf{1}\{x_{N-2}=0\}=1-\mathbf{1}\{x_{N-2}=1\}=1-\mathbf{1}\{\tilde{y}_{N-2}=2\}.$$

Therefore, during Phase I,

$$\begin{aligned}D_N(\mathbf{x})&=\sum_{i=0}^{N-3}\mathbf{1}\{x_i=x_{i+1}\}+\mathbf{1}\{x_{N-2}=x_{N-1}\}\\&=\sum_{i=0}^{N-3}\mathbf{1}\{\widetilde{y}_i=\widetilde{y}_{i+1}\}+1-\mathbf{1}\{\widetilde{y}_{N-2}=2\}\\&\equiv D_{N-1}(\tilde{\mathbf{y}})+1\pmod 2,\end{aligned}$$

and hence

$$D_N(\mathbf{x}) \equiv D_{N-1}(\tilde{\mathbf{y}}) + 1 \pmod 2.$$

Since the parity of $D_{N-1}(\tilde{\mathbf{y}})$ flips at each move in Phase I, the same holds for $D_N(\mathbf{x})$.

3) Every move in Phase II flips the parity of D_N. Phase II is $\mathrm{Hanoi}(N-1; 1 \to 2)$ while disk $N-1$ stays fixed at peg 2, hence $\mathbf{1}\{x_{N-1} = 2\}$ is constantly 1 and

$$D_N(\mathbf{x}) = 1 + \sum_{i=0}^{N-3} \mathbf{1}\{x_i = x_{i+1}\} + \mathbf{1}\{x_{N-2} = x_{N-1}\}, \qquad \text{with } x_{N-1} \equiv 2.$$

Now define $\tau' = (01) \in \mathrm{S}_3$, and let $\tilde{\mathbf{y}}(t) := \tau' \circ (x_0(t), \cdots, x_{N-2}(t))$. Then $\tilde{\mathbf{y}}(t)$ evolves as the canonical $\mathrm{Hanoi}(N-1; 0 \to 2)$ trajectory. By the induction hypothesis, $D_{N-1}(\tilde{\mathbf{y}})$ flips every move. As in Phase I, we relate the boundary term: since $x_{N-1} \equiv 2$ in Phase II,

$$\mathbf{1}\{x_{N-2} = x_{N-1}\} = \mathbf{1}\{x_{N-2} = 2\}.$$

Also,

$$\mathbf{1}\{\tilde{y}_{N-2} = 2\} = \mathbf{1}\{\tau'(x_{N-2}) = 2\} = \mathbf{1}\{x_{N-2} = 2\}.$$

Thus the boundary term matches exactly, and we get

$$D_N(\mathbf{x}) \equiv D_{N-1}(\tilde{\mathbf{y}}) + 1 \pmod 2.$$

Hence the parity $D_N(\mathbf{x})$ flips at every move in Phase II.

Combining Steps 1) through 3), the parity of D_N flips on every move of the full size-N optimal trajectory, proving equation 3. To conclude the proof, we point out that equation 4 follows from the fact $D_N(\mathbf{x}(0)) = D_N(0, \cdots, 0) = N - 1$, which is clear by definition. □

State-only decoding via the parity discriminant. The parity discriminant D_N provides an elegant mechanism for decoding the optimal move from a state on the canonical trajectory, without recursion or explicit step counting. Recall that along the optimal path,

$$D_N(\mathbf{x}(t)) \equiv t + (N-1) \pmod 2.$$

Therefore, the parity of $D_N(\mathbf{x})$ directly reveals whether the current state occurs at an even or odd time index (up to a fixed offset determined by N). Given a state $\mathbf{x}$ on the optimal trajectory (while the time stamp t is unknown), this observation yields a simple state-only decoder:

- If $D_N(\mathbf{x})$ indicates that $\mathbf{x} = \mathbf{x}(t)$ for an even t, then the next move must be the disk-0 move. The destination is determined by the parity-controlled 3-cycle σ_N described in Theorem 3.
- If $D_N(\mathbf{x})$ indicates that $\mathbf{x} = \mathbf{x}(t)$ for an odd t, then `disk 0` does not move. In this case, there exists exactly one legal move not involving `disk 0`, and this move must be the optimal one.

Thus, the optimal successor can be computed deterministically from the state itself using only the parity discriminant D_N and the legality constraint.

This avoids recursive decomposition or explicit reconstruction of the time index. In particular, while the classical solution is usually described recursively, the parity discriminant shows that the optimal successor function admits a direct state-based characterization.

C.3 Worked Example: Parity-Controlled Cyclic Structure

To illustrate the structural properties described in §4.1, we visualize the full optimal trajectory for a 5-disk Tower of Hanoi instance in Figure 15. Each state is represented as $\mathbf{x} = (x_0, x_1, \cdots, x_4)$, where $x_i \in \{0, 1, 2\}$ denotes the peg occupied by `disk i`.

Rather than viewing the trajectory as a sequence of global configurations, it is instructive to track the behavior of each disk individually:

Disk-wise trajectories. Each disk follows a periodic trajectory over the three pegs, with both its frequency and orientation determined by its index and the global problem size. In particular:

- `disk 0` (smallest) moves every other step (i.e., at odd-numbered moves) and cycles through the pegs with orientation $0 \to 2 \to 1 \to 0$.
- `disk 1` moves every 2^2 steps (moves $2, 6, 10, 14, 18, 22, 26, 30$) and follows the opposite cycle orientation $0 \to 1 \to 2 \to 0$.
- `disk 2` moves every 2^3 steps (moves $4, 12, 20, 28$) and follows the same orientation as `disk 0`.
- `disk 3` moves every 2^4 steps (moves $8, 24$) and follows the same orientation as `disk 1`.
- `disk 4` moves once at move $2^4 = 16$, following the same orientation as `disk 0`.

Summary of structure. From these observations, we can identify two key structural properties of the optimal trajectory, which are described in Theorem 1 and 2:

- **Exponentially increasing period.** Each `disk i` moves every 2^{i+1} steps, leading to a hierarchy of time scales across disks.
- **Parity-controlled cycle orientation.** The direction of each disk's cyclic motion is governed by $(N - i) \bmod 2$, yielding two alternating orientation classes corresponding to the recursive decomposition.

Together, these properties show that the optimal solution is a superposition of synchronized cyclic processes, coordinated by a single global parity variable. This perspective motivates the hierarchy-aware and parity-sensitive inductive biases introduced in §4.2.

LIMITATIONS AND BROADER IMPACTS

Limitations. Our study is conducted on the classical 3-peg Tower of Hanoi, a deterministic and highly structured recursive planning problem. While this setting provides a clean testbed for analyzing recursive generalization, the specific architecture and conclusions studied here do not apply directly to more general reasoning or planning tasks with stochastic dynamics, richer action spaces, or partial observability. More broadly, however, we hope that the structural principles identified in this work can help inform the design of models for such settings.

Broader impacts. This work is primarily foundational research on recursive reasoning in a controlled combinatorial environment. A potential positive impact is that the structural insights developed here may help inform the design of more reliable reasoning and planning systems. A potential negative impact is that improved reasoning architectures could eventually contribute to more capable automated decision-making systems, which may be over-relied upon or misused if deployed without appropriate safeguards.

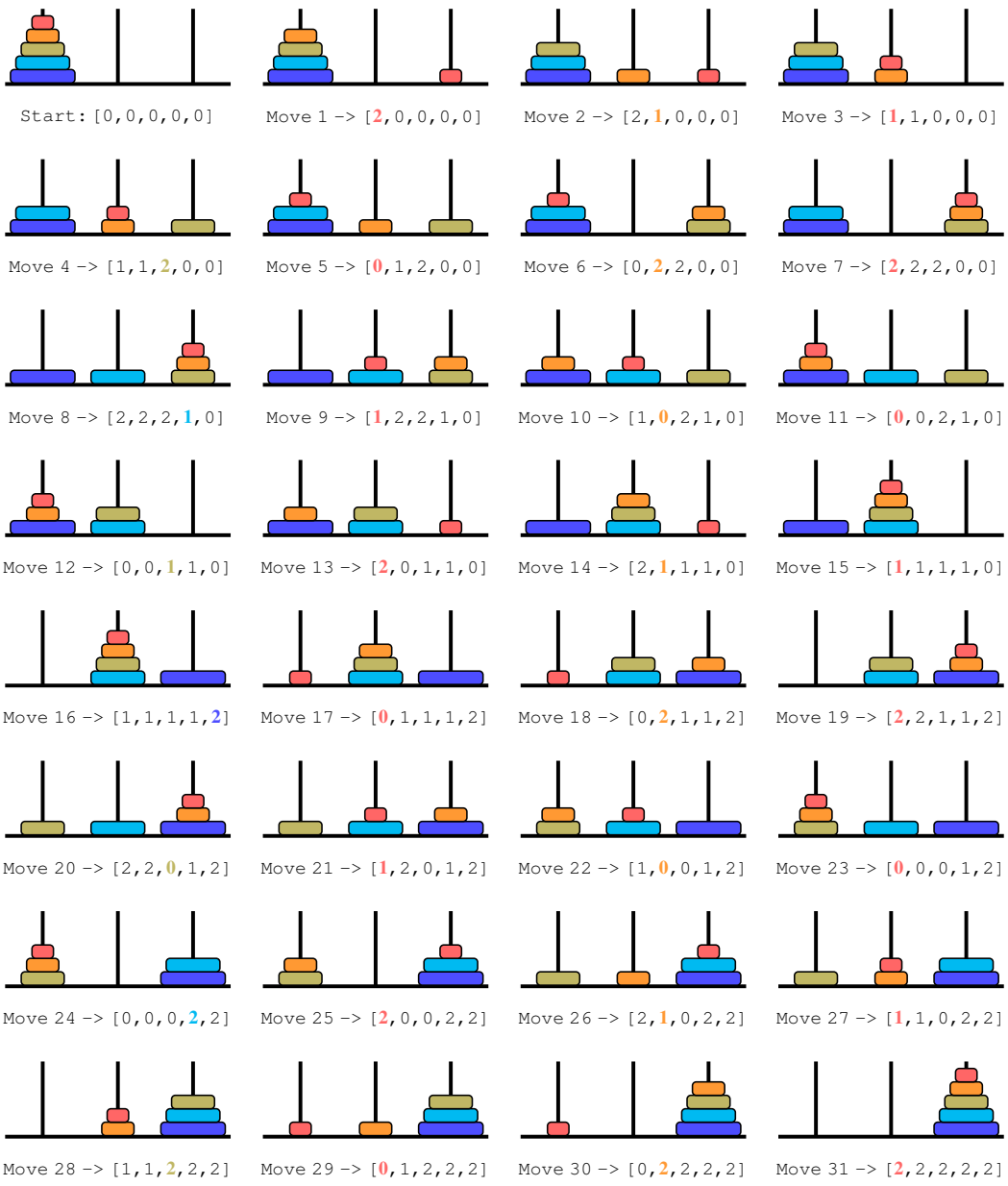

Figure 15: Optimal trajectory of the 5-disk Tower of Hanoi. Each panel shows the configuration after one move, with state vector $\mathbf{x} = (x_0, \cdots, x_4)$ indicating the peg index of each disk. Colored entries highlight the disk that moves at each step. Tracking each disk across the sequence reveals a periodic structure.