LoTR: Logic-of-Thought Routing for Plug-and-Play Reasoning of LLMs

Anonymous authors
Paper under double-blind review

Abstract

Reasoning has become a primary means of improving the problem-solving capabilities of Large Language Models (LLMs), with paradigms prescribing *external reasoning strategies* such as step-by-step, iterative revision, and search-and-sampling. However, while these paradigms guide how the model reasons, they do not consider whether the model's *internal computation pathway* is well-matched to the dynamic reasoning logic state under the prescribed strategy, limiting the potential under this strategy. We propose *Logic-of-Thought Routing* (LoTR), a plug-and-play reasoning module that strengthens existing reasoning paradigms through logic-conditioned internal pathway routing. Concretely, LoTR identifies dynamic logic states from the model's internal information transfer, couples them with attention-head pathway patterns, and uses a lightweight probe to infer the current logic-state mixture and softly route the corresponding pathway. This enables the model to better realize the prescribed reasoning strategy through an internal pathway matched to the current reasoning logic. Across 3 backbones, 10 benchmarks, and 8 reasoning paradigms, LoTR improves matched accuracy by 3.60% on average, reaching 5.37% on Llama with only +0.15% completion tokens and a favorable latency profile across backbones. It also outperforms the mean of 4 inference-time intervention baselines on all 3 backbones.

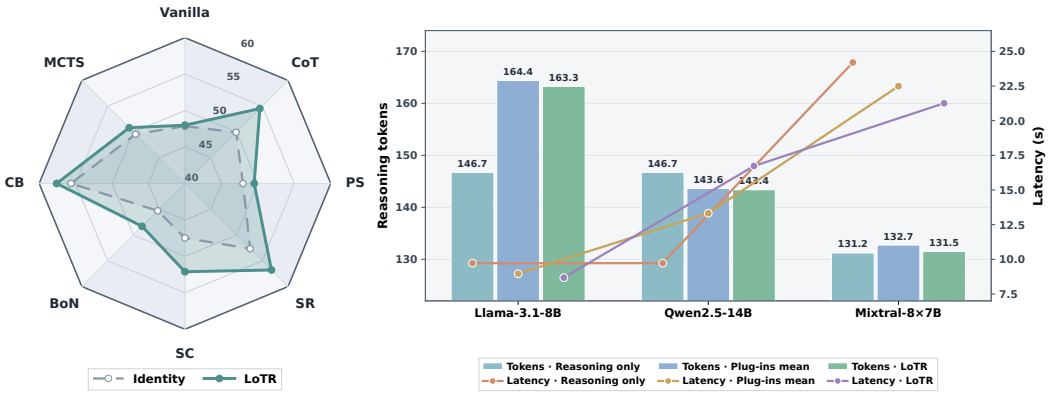


Figure 1: **LoTR improves reasoning without expanding inference cost.** Llama accuracy across paradigms (left); reasoning tokens (bars) and latency (lines) across backbones (right).

1 Introduction

Reasoning has become a primary way to improve the problem-solving ability of Large Language Models (LLMs) (Chen et al., 2023; Gao et al., 2023; Yao et al., 2023b; Zhou et al., 2023). Recent progress largely comes from extensive *reasoning paradigms* that prescribe *external reasoning strategies*, such as structured step decomposition (Wang et al., 2023a; Wei et al., 2022), iterative revision (Madaan et al., 2023), planning (Wang et al., 2023a; Yao et al., 2023a), and search-and-sampling (Brown et al., 2024; Snell et al., 2025; Wang et al., 2023b). Although differing in form, these paradigms follow the same basic mechanism: push the model toward better reasoning performance by specifying stronger external procedures (Chen et al., 2023; Gao et al., 2023) or by spending more computation on sampling, branching, or verification (Lightman et al., 2024; Xie et al., 2024). This

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

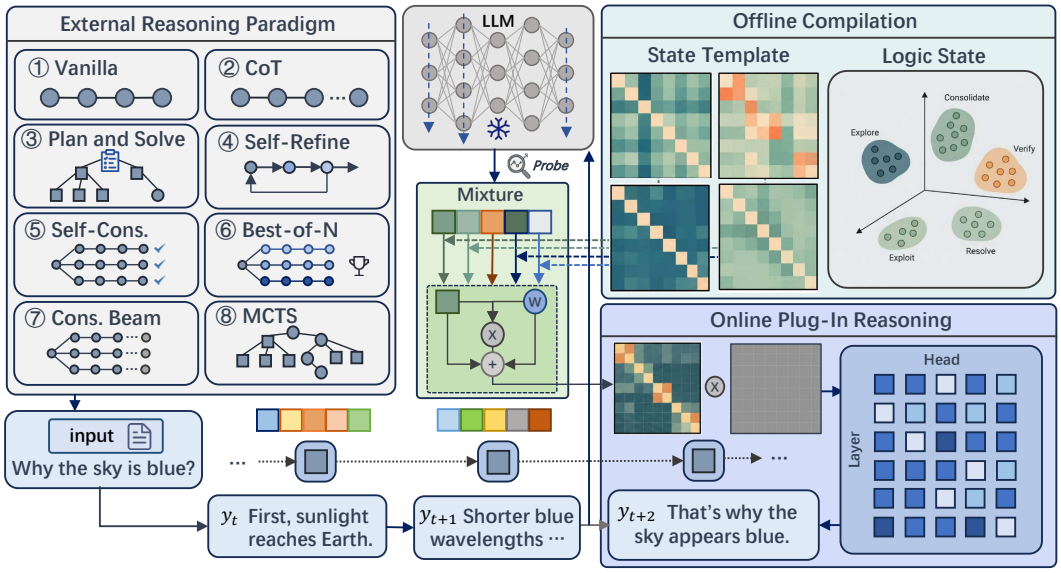


Figure 2: LoTR as plug-and-play reasoning through logic-conditioned internal pathway routing.

mechanism brings high effectiveness but still leaves an important complementary dimension less examined. Specifically, the model executes reasoning through fixed *internal computation pathways* with reasoning strategies prescribed, rarely asking whether those internal pathways are well matched to the *dynamic reasoning logic* induced by the strategy. This mismatch may leave part of the benefit of external reasoning strategies unrealized, especially when different reasoning steps require different internal modes of execution.

This limitation motivates us to examine the model's *internal logic states* as latent execution modes that evolve throughout reasoning, and to use them to guide reasoning through more suitable internal computation pathways. Prior studies on model internals (Zou et al., 2023) and reasoning interpretability (Zhang et al., 2025) suggest that reasoning is internally non-uniform, with different strategies (Hao et al., 2024; He et al., 2026) and even different steps within the same strategy (Sun et al., 2026, Gong et al., 2026) inducing distinct logic states. Meanwhile, analyses of transformer computation suggest that attention heads are differentiated contributors to reasoning-time information transfer, collectively shaping the pathways determining how the reasoning is carried out (Michel et al., 2019; Olsson et al., 2022; Voita et al., 2019). Together, these observations suggest that internal logic states and head-level differences exist, yet existing reasoning paradigms rarely couple the two: they prescribe *which strategy* to follow externally, but do not decide *which internal pathway* to use internally under dynamic reasoning logic. Therefore, these motivations raise a question: *Can dynamic logic states be coupled with internal pathway routing to enable reasoning paradigms to execute more effectively?*

We propose **Logic-of-Thought Routing (LoTR)**, a lightweight plug-and-play reasoning module that strengthens existing reasoning paradigms through *logic-conditioned internal pathway routing*. LoTR builds a compact view of dynamic logic states from the model's internal information transfer and links these states to stable patterns of attention-head pathways. During reasoning, a lightweight linear probe infers the current logic-state mixture from the ongoing step, and the model is then softly routed through the pathway most suitable for the current reasoning logic. In this way, LoTR complements existing reasoning paradigms with prescribed external strategies by making their internal execution more adaptive to the logic that each reasoning step actually requires.

Across 3 backbones, 10 benchmarks, and 8 reasoning paradigms, LoTR improves matched accuracy by 5.37%, 2.82%, and 2.61% on Llama-3.1-8B, Qwen2.5-14B, and Mixtral-8×7B, respectively. On the complete 8-benchmark × 8-paradigm Llama matrix, it improves or ties 51/64 cells while changing total completion tokens by only +0.15% and retaining a favorable latency profile. LoTR also outperforms the mean of 4 inference-time intervention baselines on all 3 backbones, by up to 2.86 points. These results show that LoTR improves reasoning paradigms not by expanding search or sampling, but by routing each step through an internal pathway matched to its current logic.

2 LOGIC-OF-THOUGHT ROUTING

2.1 PROBLEM FORMULATION

Let f_θ be a frozen Transformer with L layers, where layer ℓ contains H_ℓ attention heads, and let $\mathcal{H} = \{(\ell, h) \mid \ell \in \{0, \dots, L-1\},\ h \in \{1, \dots, H_\ell\}\}$ denote the heads. For an input problem x, let p be the prompt constructed under a fixed external *reasoning strategy* π, and let $y_{1:T} = (y_1, \dots, y_T)$ denote the resulting reasoning trajectory, where each y_t is a sentence-level reasoning step. We write $\mathcal{H}_{\text{route}} = \{(\ell, h) \in \mathcal{H} : \ell \in \mathcal{L}_{\text{route}}\}$ for the heads controlled by LoTR. Under such a strategy, each reasoning step is produced through head-level information transfer across layers, so the internal execution of reasoning can be viewed through the pattern of head participation.

To make this view operational, we introduce a step-level *internal computation pathway* $\mathcal{P}_t = \{\gamma_t^{(\ell,h)}\}_{(\ell,h)\in\mathcal{H}}$, where $\gamma_t^{(\ell,h)} \in [0, 1]$ denotes the participation strength assigned to head (ℓ, h) when producing step t. Rather than assuming a discrete or uniquely identifiable circuit, $\mathcal{P}_t$ serves as a soft pathway descriptor over attention heads. The induced step-level generation can then be written as $p(y_t \mid p, y_{<t}; \pi, \mathcal{P}_t)$. The central problem of LoTR is to characterize the *reasoning logic* induced by the fixed external strategy at the current step (Sec. 2.2), and to instantiate a soft internal pathway $\mathcal{P}_t$ that is better matched to that logic (Sec. 2.3), as shown in Figure 2 and Algorithm 1. LoTR compiles a shared state basis and one head-gating template per state with the language model frozen. At each sentence boundary, layer-wise probes mix these cached templates into the pathway used for the next step, adapting internal execution without changing the external reasoning strategy.

2.2 OFFLINE LOGIC MODELING

The offline stage builds a compact basis of recurring logic states and couples them to stable head-level pathway patterns with details in Appendix B. This shifts the structural computation away from inference time, so the online controller only needs lightweight readout and template composition.

Logic-state basis. We first collect reasoning trajectories from diverse reasoning paradigms and pool their step-level representations into a shared bank. For each reasoning step t, we extract a compact reference readout $r_t = \phi(p, y_{\le t}; f_\theta) \in \mathbb{R}^d$, instantiated as the mean-pooled hidden representation over the current sentence span from the final block, where d is the hidden width. Pooling across paradigms is essential: LoTR treats logic as a shared internal basis rather than as a byproduct of any single external strategy.

We apply a fixed low-dimensional transform and run K-means, using the centroids $\{c_k\}_{k=1}^K$ as anchors of recurring logic states. A distance-based target $\alpha_t^\star \in [0, 1]^K$ retains graded state proximity; Appendix I gives its exact construction. These targets supervise state readout, while the online activations remain non-exclusive to permit mixed logic states.

Layer-wise logic probes. To make the state basis usable online, LoTR trains a lightweight linear probe at each routed layer (Alain & Bengio, 2016). For layer $\ell \in \mathcal{L}_{\text{route}}$, let $r_t^{(\ell)} \in \mathbb{R}^d$ be the sentence-level readout extracted from layer ℓ, using the same sentence span and pooling rule. Each probe, parameterized by $A^{(\ell)} \in \mathbb{R}^{K\times d}$ and $b^{(\ell)} \in \mathbb{R}^K$, maps its layer-specific representation into the shared final-layer state basis, predicting a non-exclusive state mixture

$$q_t^{(\ell)} = \sigma\Big(\beta\left(A^{(\ell)} r_t^{(\ell)} + b^{(\ell)}\right)\Big) \in (0, 1)^K, \tag{1}$$

where $\sigma(\cdot)$ is the element-wise sigmoid and β is a fixed backbone-level calibration scalar (Appendix I). We deliberately avoid simplex-constrained normalization, since multiple logic states may co-activate within the same reasoning step. The probes are trained by binary cross-entropy against the soft state targets:

$$\mathcal{L}_{\text{probe}} = -\sum_{\ell\in\mathcal{L}_{\text{route}}} \mathbb{E}_t \sum_{k=1}^{K} \Big[\alpha_t^\star(k) \log q_t^{(\ell)}(k) + \big(1 - \alpha_t^\star(k)\big) \log\big(1 - q_t^{(\ell)}(k)\big)\Big]. \tag{2}$$

Logic–pathway coupling. With the state basis fixed, we next couple each state to a stable head-level pathway template. In the final estimator, each state instead has trainable logits $\Theta_k = \{\theta_k^{(\ell,h)}\}_{(\ell,h)\in\mathcal{H}_{\text{route}}}$ and gates $G_k = \sigma(\Theta_k)$. Let $\mathcal{S}_k^+$ and $\mathcal{S}_k^-$ be equal-size compilation sentence

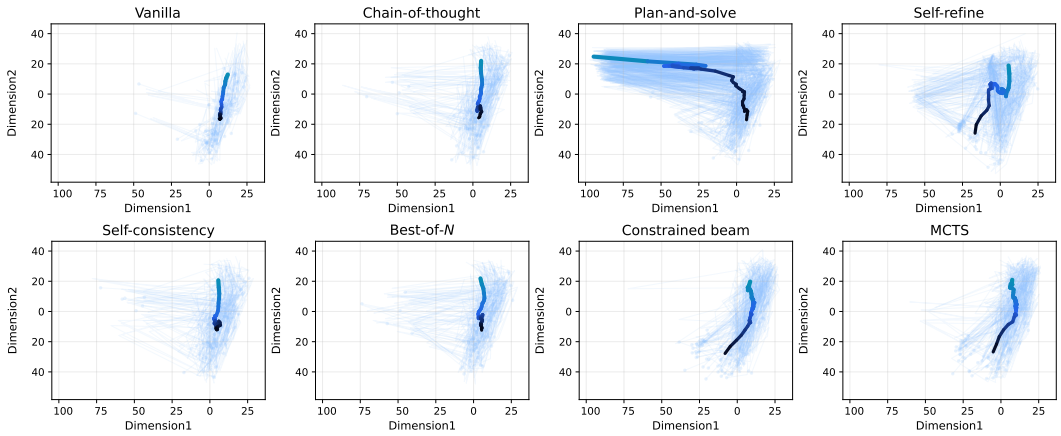


Figure 3: **Logic states across reasoning paradigms.** Different external strategies occupy different regions and transition patterns, suggesting distinct distributions over the same latent logic-state space.

pools that the frozen final-layer probe assigns to state k, drawn respectively from correct and incorrect trajectories. For a sentence set $\mathcal{S}$, let $\bar{\ell}(\mathcal{S}; G)$ be the token-weighted mean teacher-forced log-likelihood over all target tokens in $\mathcal{S}$ under gates G. We optimize only Θ_k:

$$\mathcal{L}_{\text{tpl}}(k) = -\bar{\ell}(\mathcal{S}_k^+; G_k) + \lambda \bar{\ell}(\mathcal{S}_k^-; G_k) + \mu \frac{1}{|\mathcal{H}_{\text{route}}|} \sum_{(\ell,h)\in\mathcal{H}_{\text{route}}} (1 - G_k^{(\ell,h)})^2. \tag{3}$$

Here $\lambda, \mu \geq 0$ weight the contrast and identity regularizer. The three terms preserve successful state-k reasoning, contrast it against unsuccessful trajectories, and regularize unnecessary deviation from the intact pathway. Only Θ_k is optimized; the resulting gates are cached as the template $M_k = \{m_k^{(\ell,h)}\}_{(\ell,h)\in\mathcal{H}_{\text{route}}}$, with $m_k^{(\ell,h)} = G_k^{(\ell,h)}$ (Appendix I). Attention heads thus provide a compact control surface for adapting cross-token information transfer without altering the external reasoning algorithm or expanding the model.

Therefore, this offline stage yields a shared logic space in which different reasoning paradigms can be compared directly. Although their external strategies differ, they need not occupy that space in the same way: some repeatedly concentrate on a few states, whereas others traverse a broader or more mixed subset as shown in Figure 3. This leads to the first structural observation behind LoTR: external reasoning paradigms differ not only in their trajectory-level procedures, but also in the distributions they induce over latent internal states. We call these states *logic states* because they appear as recurring, probe-identifiable execution regimes along reasoning trajectories, and later validate their functional relevance through state-conditioned routing and ablation analyses.

2.3 Online Probe-Based Plug-in Reasoning

Pathway instantiation. As reasoning unfolds, LoTR dynamically reads the layer-wise sentence representations $\{r_t^{(\ell)}\}_{\ell\in\mathcal{L}_{\text{route}}}$ and evaluates the trained probes in Eq. (1). The resulting activations $\{q_t^{(\ell)}\}$ provide a compact state-level description of the logic currently expressed by the ongoing step.

LoTR then composes the pathway for the subsequent step by mixing the cached state templates according to these activations. For each routed head $(\ell, h) \in \mathcal{H}_{\text{route}}$, we compute

$$\gamma_{t+1}^{(\ell,h)} = \text{clip}\left(\sum_{k=1}^{K} q_t^{(\ell)}(k)\, m_k^{(\ell,h)},\ 0,\ 1\right), \qquad (\ell,h) \in \mathcal{H}_{\text{route}}, \tag{4}$$

which instantiates the pathway $\mathcal{P}_{t+1} = \{\gamma_{t+1}^{(\ell,h)}\}_{(\ell,h)\in\mathcal{H}}$ for generating step y_{t+1}, with the coefficients directly applied to the corresponding head outputs. Heads outside $\mathcal{H}_{\text{route}}$ retain unit coefficients.

Plug-in reasoning. The resulting controller is lightweight: online inference requires only one probe evaluation per layer at each sentence boundary, followed by a weighted combination over cached

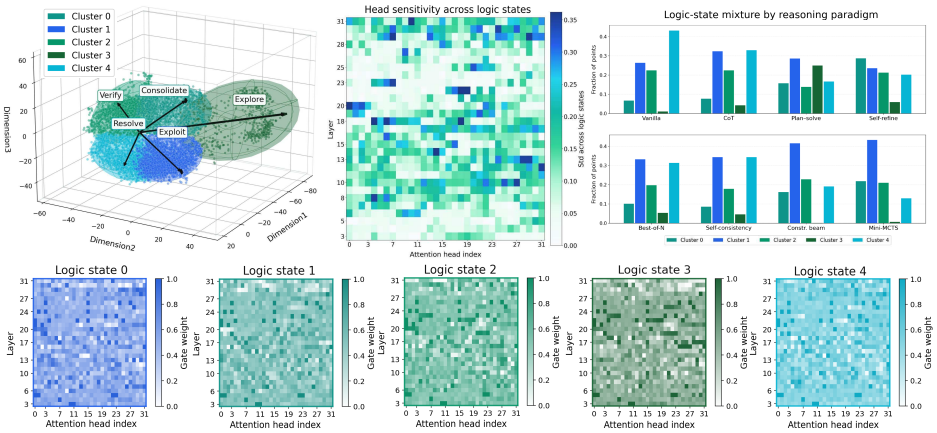


Figure 4: **Stable pathway patterns under different logic states.** Distinct recurring logic states are coupled to distinct head-level routing templates, enabling reusable pathway control.

templates, with details in Appendix C. This simple state-to-pathway map is already sufficient to make internal execution adaptive to the dynamic reasoning logic induced by a fixed external strategy.

Figure 4 illustrates the second structural observation behind LoTR: the discovered logic states behave as recurring internal regimes, rather than merely isolated hidden-state clusters, and are empirically coupled to distinct head-level routing patterns that can be reused during reasoning. In this sense, LoTR does not alter the external reasoning paradigm itself; instead, it improves its execution by aligning each reasoning step with an internal pathway suited to the logic state it currently induces.

Design interpretation. LoTR separates two complementary decisions that standard reasoning systems conflate: the external paradigm determines *what* procedure to follow, whereas the controller determines *how* the current step is internally executed. The non-exclusive probe mixture is central to this separation. A step can simultaneously express, for example, exploratory and verification-like structure, so composing state templates preserves transitions that a hard state switch would discard. Because the templates are compiled once and mixed through bounded head coefficients, this adaptivity remains local to internal execution and leaves the outer prompt, search rule, and stopping condition unchanged.

Reusable rather than paradigm-specific control. Pooling trajectories across paradigms makes the state basis a common interface between heterogeneous external procedures and the backbone's internal pathways. The same probe–template system can therefore respond differently as a trajectory evolves without learning a separate controller for CoT, revision, sampling, or search. This factorization is what makes LoTR plug-and-play: it reuses stable pathway primitives while allowing their mixture to track the step-level logic induced by any supported reasoning scaffold.

3 Experiments

3.1 Experimental Setup

Backbone and benchmarks. We evaluate LoTR on 3 LLM backbones: **Llama-3.1-8B** (Grattafiori et al., 2024), **Qwen2.5-14B** (Yang et al., 2024), and **Mixtral-8x7B** (Jiang et al., 2024). We use 10 benchmarks spanning Boolean, formal, numerical, multi-hop, scientific, commonsense, coding, and long-form reasoning. The complete Llama study uses 8 benchmarks; the complementary Qwen and Mixtral panels use 6 and 4, respectively. Appendix G gives the exact panel composition and native metrics.

External reasoning paradigms. We instantiate LoTR on Vanilla and 7 representative reasoning paradigms: **Chain-of-Thought** (CoT) (Wei et al., 2022), **Plan-and-Solve** (PS) (Wang et al., 2023a), **Self-Refine** (SR) (Madaan et al., 2023), **Self-Consistency** (SC) (Wang et al., 2023b), **Best-of-N** (BoN) (Chow et al., 2024; Lightman et al., 2024), **Constrained Beam** (CB) (Hokamp & Liu, 2017; Post & Vilar, 2018), and **Monte Carlo Tree Search** (MCTS) (Xie et al., 2024). These paradigms span direct generation, decomposition, revision, sampling, reranking, constrained decoding, and

Table 1: **Main results on Llama-3.1-8B (accuracy, %).** Each reasoning paradigm is paired with LoTR on the complete 8×8 matrix. Parentheses are changes from matched Identity; Average weights datasets equally.

Method	BoolQ	DROP	HumanEval	QASC	ScienceQA	MMLU	NarrativeQA	FOLIO	Average
Vanilla	80.0	**7.9**	68.0	38.4	68.0	**46.0**	**25.2**	49.5	47.9
+ **LoTR**	**80.0** (0.0)	7.6 (-0.3)	**70.0** (+2.0)	**38.4** (0.0)	**68.0** (0.0)	45.6 (-0.4)	24.9 (-0.3)	**49.5** (0.0)	**48.0** (+0.13)
CoT	75.5	11.3	54.0	38.4	73.6	**59.2**	42.8	44.6	49.9
+ **LoTR**	**79.5** (+4.0)	**13.9** (+2.6)	**58.0** (+4.0)	**48.4** (+10.0)	**76.4** (+2.8)	58.4 (-0.8)	**49.1** (+6.3)	**52.5** (+7.9)	**54.5** (+4.61)
Plan-and-Solve	72.5	10.7	**34.0**	**45.2**	78.4	48.0	46.4	**48.5**	48.0
+ **LoTR**	**74.0** (+1.5)	**14.0** (+3.3)	32.0 (-2.0)	44.4 (-0.8)	**81.6** (+3.2)	**53.2** (+5.2)	**49.3** (+2.9)	47.5 (-1.0)	**49.5** (+1.54)
Self-Refine	72.5	27.1	48.0	55.2	77.6	59.6	39.7	41.6	52.7
+ **LoTR**	**73.5** (+1.0)	**31.7** (+4.6)	**58.0** (+10.0)	**62.0** (+6.8)	**83.2** (+5.6)	**60.4** (+0.8)	**43.9** (+4.2)	**41.6** (0.0)	**56.8** (+4.13)
SC	72.0	14.3	**42.0**	46.4	74.8	46.0	41.7	42.6	47.5
+ **LoTR**	**80.0** (+8.0)	**17.9** (+3.6)	40.0 (-2.0)	**47.6** (+1.2)	**82.4** (+7.6)	**54.4** (+8.4)	**43.0** (+1.4)	**51.5** (+8.9)	**52.1** (+4.64)
BoN	72.5	13.4	44.0	44.8	77.2	52.8	25.0	**32.7**	45.3
+ **LoTR**	**78.0** (+5.5)	**16.2** (+2.8)	**50.0** (+6.0)	**48.4** (+3.6)	**78.0** (+0.8)	**57.6** (+4.8)	**29.6** (+4.6)	28.7 (-4.0)	**48.3** (+3.02)
Constrained Beam	78.5	**33.3**	62.0	48.0	72.4	50.4	**50.9**	49.5	55.6
+ **LoTR**	**82.0** (+3.5)	31.5 (-1.7)	**62.0** (0.0)	**49.6** (+1.6)	**77.6** (+5.2)	**56.8** (+6.4)	49.7 (-1.2)	**51.5** (+2.0)	**57.6** (+1.97)
Mini-MCTS	79.5	20.5	52.0	40.0	66.8	48.0	**47.2**	**42.6**	49.6
+ **LoTR**	**79.5** (0.0)	**23.4** (+2.9)	**58.0** (+6.0)	**40.8** (+0.8)	**67.2** (+0.4)	**51.2** (+3.2)	44.8 (-2.3)	41.6 (-1.0)	**50.8** (+1.25)

tree search. We evaluate the full 8-paradigm matrix on Llama, with complementary Self-Refine and CoT transfer panels on Qwen and Mixtral. All methods within a panel share the questions, prompt, decoding settings, checkpoint, and scorer.

Plug-in baselines. We compare LoTR with 4 frozen-backbone inference-time interventions: **ITI** (Li et al., 2023), **ActAdd** (Turner et al., 2023), **STIR** (Shi et al., 2026), and **RISER** (Ye et al., 2026). We report every method and their fixed arithmetic mean on common-support panels; Appendix H provides configurations and exact support.

Metrics. We use benchmark-native accuracy, normalized F1, or execution-based Pass@1 and macro-average datasets equally within each panel. Every paired Identity–LoTR comparison shares questions, prompts, decoding, parsing, and scoring; Appendix G gives the complete evaluation protocol. We recompute final-call and all-call completion tokens from the paired records, using all-call tokens as the decoding-work proxy for multi-call paradigms.

3.2 Main Results

Overall effectiveness. Table 1 shows that LoTR raises the equal-weight Llama macro by 5.37% across the complete eight-benchmark, eight-paradigm matrix. It improves 44 of 64 cells and ties 7 more, while every benchmark-level macro improves. The breadth across direct generation, decomposition, revision, sampling, reranking, constrained decoding, and tree search supports logic-conditioned routing rather than wrapper-specific tuning.

Comparison with plug-in interventions. On the exact common CoT support in Table 2, LoTR reaches 55.2%, compared with 50.6% for Identity and 52.4% for the four-plug-in mean. It is best or tied-best on four of five datasets and leads the intervention aggregate by 2.86 points. Thus, the gain is not reproduced by a single fixed steering direction: conditioning the pathway on the evolving logic state provides a more consistent control signal across tasks.

Table 2: **Standalone intervention baselines on common CoT support.**

Method	BoolQ	DROP	HumanEval	QASC	ScienceQA	Average
Identity	75.5	11.3	54.0	38.4	73.6	50.6
+ ITI	73.0 (-2.5)	13.4 (+2.0)	58.0 (+4.0)	42.8 (+4.4)	78.0 (+4.4)	53.0 (+2.5)
+ ActAdd	77.5 (+2.0)	11.0 (-0.3)	56.0 (+2.0)	43.6 (+5.2)	76.0 (+2.4)	52.8 (+2.3)
+ STIR	75.5 (0.0)	11.6 (+0.3)	52.0 (-2.0)	41.2 (+2.8)	78.8 (+5.2)	51.8 (+1.3)
+ RISER	75.0 (-0.5)	11.6 (+0.2)	54.0 (0.0)	41.2 (+2.8)	77.6 (+4.0)	51.9 (+1.3)
+ **LoTR**	**79.5** (+4.0)	**13.9** (+2.6)	**58.0** (+4.0)	**48.4** (+10.0)	**76.4** (+2.8)	**55.2** (+4.7)

Backbone transfer. Table 3 shows that the gain transfers to Qwen and Mixtral, improving their matched macros by 2.82% and 2.61%. Against plug-in interventions, LoTR exceeds the mean of ITI, ActAdd, STIR, and RISER by 2.86 points on Llama and 0.51 points on Qwen. On Mixtral, it remains above the same four-method mean by 0.16 points.

Table 3: **Cross-backbone transfer (%).** Matched panels; the final column reports LoTR minus the mean of 4 plug-ins on common support.

Backbone	Identity	LoTR	Gain	vs. plug-ins
Llama-3.1-8B	49.5	**52.2**	**+5.37%**	+2.86 pp
Qwen2.5-14B	75.0	**77.1**	**+2.82%**	+0.51 pp
Mixtral-8×7B	35.6	**36.5**	**+2.61%**	+0.16 pp

3.3 Efficiency Analysis

Efficiency. LoTR changes internal routing without adding sampling or search calls. Across the complete 64-cell Llama panel, it improves accuracy by 5.37% while changing total completion tokens by only +0.15%; final-call length changes by -0.43%. The gain therefore does not come from a larger decoding budget. Figure 1 further shows the corresponding token–latency profile across backbones.

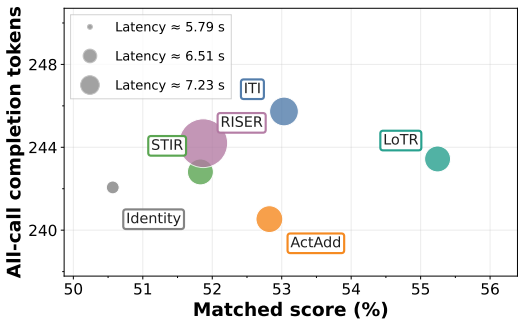


Figure 5: **Llama CoT accuracy–efficiency trade-off.** Points use the final five-dataset common support; bubble area encodes controlled latency.

Table 4: **Token efficiency on Llama.** Identity / LoTR, macro-averaged over 8 datasets and 8 paradigms.

Measure	Identity / LoTR	Change
Final call	141.3 / 140.7	-0.43%
All calls	382.4 / 383.0	+0.15%

Trade-off frontier. LoTR reaches the highest matched score (55.2%) at 243.4 all-call tokens, compared with 50.6% at 242.1 tokens for Identity. The four fixed plug-ins span 51.8–53.0% and 240.5–245.7 tokens. LoTR therefore shifts the score frontier without buying the gain through longer decoding; its latency bubble also remains comparable to ITI, ActAdd, and STIR and below RISER. The horizontal separation with little vertical displacement is the desired signature: LoTR improves answer quality while leaving the decoding workload essentially unchanged, rather than exchanging longer traces for accuracy.

3.4 Ablation Studies

State differentiation. Table 5 compares the deployed six-state controller with both Identity routing and an independently compiled one-state controller on a matched three-dataset, four-paradigm panel. Full LoTR improves the panel macro by 3.12 points over Identity and by 2.87 points over true $K{=}1$, showing that the gain is not recovered by a single globally routed template. Although $K{=}1$ improves DROP, only the differentiated basis leads on both BoolQ and ScienceQA-text and attains the strongest macro. The useful signal is therefore state-selective pathway composition rather than a uniform global intervention.

Table 5: **Aligned mechanism ablation on Llama-3.1-8B.** Percentages average Vanilla, CoT, SC, and genuine BoN on matched items.

Variant	BoolQ	DROP	ScienceQA-text	Avg.
Identity routing	75.0	11.7	73.4	53.4
True $K{=}1$	72.9	**16.3**	71.7	53.6
Full LoTR ($K{=}6$)	**79.4**	13.9	**76.2**	**56.5**

3.5 Sensitivity Analysis

Figure 6 varies the state basis and probe-logit scale while keeping the remaining configuration fixed. The deployed multi-state basis improves the matched macro over both Identity and independently compiled $K{=}1$, while LoTR–Identity gains remain positive throughout $\beta \in [0.35, 0.65]$ and peak at

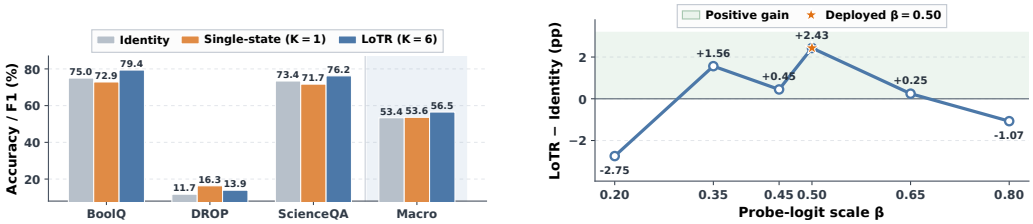


Figure 6: **Measured LoTR sensitivity on Llama-3.1-8B.** Left: matched Identity, an independently compiled single-state basis (K=1), and the deployed multi-state basis (K=6). Right: development-set macro changes across measured probe-logit scales; the star marks the deployed β=0.50.

the deployed $\beta = 0.50$ (Appendix Table 8). Together, these results show that LoTR benefits from differentiated states without requiring a knife-edge routing strength.

3.6 Case Study

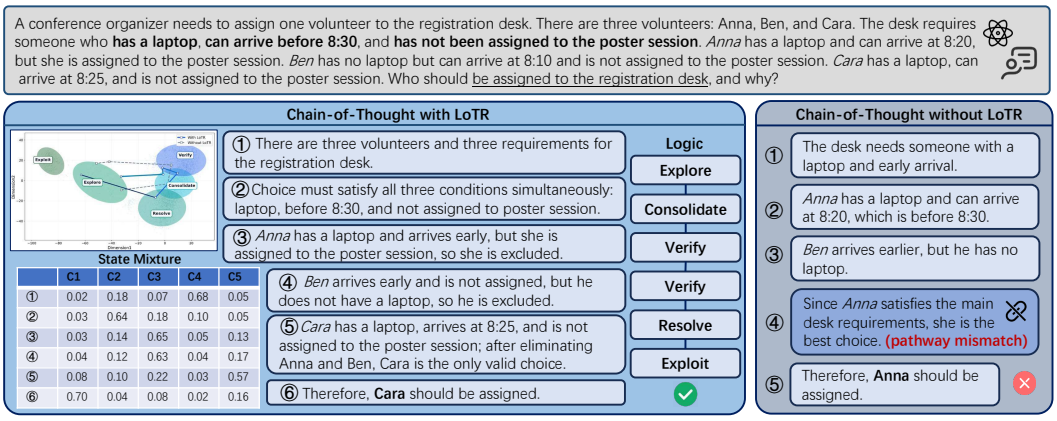


Figure 7: **Qualitative case study.** Representative trajectory *with* vs. *without* LoTR under CoT.

Figure 7 compares two trajectories under the same Chain-of-Thought scaffold. The unmodified CoT trajectory identifies the relevant requirements, but prematurely selects Anna after matching laptop availability and early arrival, losing the negative poster-session constraint. Thus, the failure is not the absence of an external scaffold: the steps are locally plausible, but internal execution drifts from constraint verification to early selection. With LoTR, the same scaffold yields a different step-level pattern: the model parses all requirements, eliminates Anna and Ben by their violated constraints, verifies Cara against the full condition set, and stops at the only valid candidate. This example illustrates LoTR's role: it does not rewrite the outer reasoning program, but routes each reasoning sentence through an internal pathway matched to its current logic state.

Evidence synthesis. The results jointly isolate pathway routing from simpler explanations. Every Llama paradigm macro rises, 51 of 64 cells improve or tie, and the design transfers to Qwen and Mixtral, indicating a mechanism broader than one wrapper or task. The gain is not purchased through longer decoding or a generally stronger intervention: all-call length changes by only $+0.15\%$, while LoTR exceeds the four-method plug-in mean by 2.86 points. Finally, the K=1 ablation and case study connect this breadth to differentiated state-conditioned execution. Together, these results support the core claim that external reasoning becomes more reliable when its internal pathway tracks the logic currently being executed.

4 Related Work

External reasoning paradigms for LLMs. A large body of work improves LLM reasoning by changing the *external* procedure followed at inference time, including chain-of-thought prompting (Wei et al., 2022), least-to-most decomposition (Zhou et al., 2023), plan-based prompting (Wang et al., 2023a), program-aided reasoning (Chen et al., 2023; Gao et al., 2023), iterative self-refinement

(Madaan et al., 2023), self-consistency and sample aggregation (Lightman et al., 2024; Wang et al., 2023b), best-of-N sampling and test-time compute scaling (Brown et al., 2024; Chow et al., 2024; Snell et al., 2025), and tree- or search-based reasoning (Xie et al., 2024; Yao et al., 2023a;b). These methods primarily prescribe a stronger trajectory-level scaffold or allocate more computation to sampling, branching, verification, or tool-like decomposition. LoTR is orthogonal to this direction: it keeps the external scaffold fixed and studies whether each reasoning step can be internally executed through a pathway matched to the logic state induced by that scaffold.

Internal and latent structure of reasoning. Another line of work analyzes the internal organization of transformer computation, including attention-head specialization (Michel et al., 2019; Voita et al., 2019), induction heads and circuit-level mechanisms (Olsson et al., 2022), and structured representation geometry in language models (Zou et al., 2023). Recent studies further suggest that reasoning is not internally uniform: hidden states may encode correctness or verification signals (Zhang et al., 2025), and multi-step trajectories can exhibit step-specific representation geometry (Sun et al., 2026). Closely related latent-reasoning methods move reasoning into continuous hidden space rather than decoded token space (Hao et al., 2024). LoTR builds on the view that reasoning has structured latent states, but differs in purpose: it neither replaces language-level reasoning with a latent trajectory nor stops at interpretation. Instead, it turns recurring logic states into reusable control signals for routing internal computation under existing reasoning paradigms.

Plug-in control and inference-time steering. LoTR is also related to frozen-model intervention methods, including inference-time intervention over selected heads (Li et al., 2023), activation steering through additive latent directions (Rimsky et al., 2024; Turner et al., 2023), and representation engineering approaches that manipulate behavior through latent directions (Zou et al., 2023). Recent reasoning-oriented steering methods further retrieve or compose latent steering signals during inference, such as STIR and RISER (Shi et al., 2026; Ye et al., 2026). These methods show that model behavior can be changed without weight updates, but their control objects are typically global directions, retrieved impulses, or routed steering vectors. LoTR instead uses a sentence-level mixture over recurring logic states to compose *head-level pathway templates*, tying the intervention to the evolving logic of multi-step reasoning rather than to a fixed attribute direction or task-level steering.

Probes and representation-based control. Probing methods use lightweight classifiers to read structured information from model representations (Alain & Bengio, 2016; Belinkov, 2022). In reasoning settings, probes have been used to expose hidden correctness or verification signals and to support decisions such as early stopping (Zhang et al., 2025). LoTR uses probes in a different role: the probe is not a final verifier or post-hoc interpreter, but an online estimator of logic-state activations. These activations are mapped to a reusable routing policy over head-level pathway templates. Thus, LoTR turns representation readout into reasoning-time control: latent states are not only measured, but used to determine how the next reasoning step is internally realized.

5 Conclusion

We presented LoTR, a plug-and-play reasoning controller that strengthens existing reasoning paradigms by routing each step through an internal pathway matched to its inferred logic state. The key insight is that reasoning quality depends not only on the external scaffold a model follows, but also on the internal computation through which each step is realized. LoTR implements this view by extracting a shared logic-state basis, reading state activations with lightweight probes, and composing reusable head-level routing templates at inference time. Across models, benchmarks, and paradigms, LoTR improves the accuracy–efficiency frontier under controlled cost, showing that internal pathway control can serve as a practical complement to external reasoning design. More broadly, LoTR separates *what* procedure a reasoning scaffold prescribes from *how* the frozen model internally carries out each step. This separation offers a reusable route to improving diverse reasoning systems without retraining the backbone or redesigning their external procedures.

Reproducibility statement. The complete LoTR pipeline and its offline and online algorithms are specified in Appendix A–C; Appendix G–I records models, datasets, baselines, metrics, and configurations. The supplementary material provides the anonymous core implementation and submission-aligned configurations.

Ethics statement. This work studies inference-time reasoning with publicly released pretrained models and established public benchmarks. It uses no human subjects, private data, or personally identifying information.

AI use statement. Generative AI tools were used to assist with language editing, presentation and organization, literature search, software engineering, and the analysis and visualization of experimental results. The authors determined the research questions, methodology, experimental design, and scientific conclusions, and independently reviewed the code, source evidence, and reported results. All AI-assisted material was reviewed and revised by the authors, who take full responsibility for the paper.

REFERENCES

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

Bradley Brown, Jordan Juravsky, Ryan Saul Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. CODAH: An adversarially authored question-answer dataset for common sense. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pp. 63–69, 2019.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *Transactions on Machine Learning Research*, 2023.

Yinlam Chow, Guy Tennenholtz, Izzeddin Gur, Vincent Zhuang, Bo Dai, Sridhar Thiagarajan, Craig Boutilier, Rishabh Agarwal, Aviral Kumar, and Aleksandra Faust. Inference-aware fine-tuning for best-of-n sampling in large language models. *arXiv preprint arXiv:2412.15287*, 2024.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2924–2936, 2019.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of NAACL-HLT*, pp. 2368–2378, 2019.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International Conference on Machine Learning*, 2023.

Zhiren Gong, Yikun Hou, Zihao Zeng, Ming Xiao, Chau Yuen, and Wei Yang Bryan Lim. State of thought enables endogenous reasoning. *arXiv preprint arXiv:2609.16055*, 2026. doi: 10.48550/arXiv.2609.16055.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alex Wardle-Solano, Hannah Szabo, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, Ansong Ni, Linyong Nan, Jungo Kasai, Tao Yu, Rui Zhang, Alexander R. Fabbri, Wojciech Kryscinski, Semih Yavuz, Ye Liu, Xi Victoria Lin, Shafiq Joty, Yingbo Zhou, Caiming Xiong, Rex Ying, Arman Cohan, and Dragomir Radev. Folio: Natural language reasoning with first-order logic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22017–22031, 2024.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024. doi: 10.48550/arXiv.2412.06769.

Zhenghao He, Guangzhi Xiong, Bohan Liu, Sanchit Sinha, and Aidong Zhang. Reasoning beyond chain-of-thought: A latent computational mode in large language models. *arXiv preprint arXiv:2601.08058*, 2026. doi: 10.48550/arXiv.2601.08058.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Chris Hokamp and Qun Liu. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1535–1546, 2017.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. QASC: A dataset for question answering via sentence composition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8082–8090, 2020.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2018. doi: 10.1162/tacl_a_00023.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems*, volume 36, pp. 41451–41530, 2023. URL https://openreview.net/forum?id=aLLuYpn83y.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *Advances in Neural Information Processing Systems*, 2022.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, 2023.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, 2019.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

Matt Post and David Vilar. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, pp. 1314–1324, 2018.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15504–15522, 2024.

Zhenning Shi, Yijia Zhu, Junhan Shi, Xun Zhang, Lei Wang, and Congcong Miao. Internalizing llm reasoning via discovery and replay of latent actions. *arXiv preprint arXiv:2602.04925*, 2026. doi: 10.48550/arXiv.2602.04925.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, 2025.

Lihao Sun, Hang Dong, Bo Qiao, Qingwei Lin, Dongmei Zhang, and Saravan Rajmohan. Llm reasoning as trajectories: Step-specific representation geometry and correctness signals. *arXiv preprint arXiv:2604.05655*, 2026. doi: 10.48550/arXiv.2604.05655.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023. doi: 10.48550/arXiv.2308.10248.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5797–5808, 2019. doi: 10.18653/v1/P19-1580.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pp. 2609–2634, 2023a.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023b.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P. Lillicrap, Kenji Kawaguchi, and Michael Shieh. Monte carlo tree search boosts reasoning via iterative preference learning. In *NeurIPS 2024 Workshop on System-2 Reasoning at Scale*, 2024. doi: 10.48550/arXiv.2405.00451. URL https://openreview.net/forum?id=s004OmYP2P.

An Yang, Baosong Yang, Beichen Zhang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. doi: 10.48550/arXiv.2412.15115.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, 2023a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023b.

Wencheng Ye, Xiaoyang Yuan, Yi Bin, Pengpeng Zeng, Hengyu Jin, Liang Peng, and Heng Tao Shen. Riser: Orchestrating latent reasoning skills for adaptive activation steering. *arXiv preprint arXiv:2601.09269*, 2026. doi: 10.48550/arXiv.2601.09269.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. *arXiv preprint arXiv:2504.05419*, 2025. doi: 10.48550/arXiv.2504.05419.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. Least-to-most prompting enables complex reasoning in large language models. In *International Conference on Learning Representations*, 2023.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023.

Appendix Overview

This appendix provides a complete and implementation-aligned extension of LoTR.

Part A: Algorithmic Details. This part provides the complete LoTR pipeline and separates offline and online procedures.

- Appendix A: end-to-end LoTR pipeline.
- Appendix B: offline logic-space construction, probe fitting, and template caching.
- Appendix C: online sentence-level probe-and-route reasoning.

Part B: Theoretical Analysis. This part formalizes the central structural claims behind LoTR.

- Appendix D: why external paradigms alone can underexplore the reasoning frontier when they ignore pathway–logic coupling.
- Appendix E: why logic-conditioned routing can improve a fixed paradigm.
- Appendix F: why the learned logic basis can generalize across problems and paradigms.

Part C: Experimental Setup. This part extends the main paper with full protocol details.

- Appendix G: datasets, models, reasoning paradigms, metrics, and protocol.
- Appendix H: baseline families and comparison rationale.
- Appendix I: hyperparameter configuration.

Part D: Additional Results and Analysis. This part provides pipeline-level analyses and additional backbone results.

- Appendix J: final-artifact state readout, pathway differentiation, and cross-paradigm recurrence under deployed $K{=}6$.
- Appendix K: detailed cross-backbone and standalone-comparator results on their matched evaluation panels.

Part E: Limitations and Discussion. This part summarizes current boundaries and next steps.

- Appendix L: limitations.
- Appendix M: future work.

Part A: Algorithmic Details

A Overall LoTR Pipeline

This section specifies the end-to-end realization of LoTR using the notation of the main text. LoTR is a two-stage plug-in system: offline logic modeling followed by online sentence-level pathway routing under a fixed external reasoning strategy π, with an explicit runtime pipeline and artifact boundary.

Algorithm 1 LoTR: End-to-End Pipeline

Require: Frozen backbone f_θ, pooled trajectory set $\mathcal{D} = \{(p, y_{1:T}, \pi)\}$, routed layers $\mathcal{L}_{\text{route}}$, test inputs $\mathcal{X}$

Ensure: Offline artifact $\mathcal{A} = \{\{c_k\}_{k=1}^K, \{A^{(\ell)}, b^{(\ell)}\}_{\ell \in \mathcal{L}_{\text{route}}}, \{M_k\}_{k=1}^K, \eta\}$, routed reasoning outputs

Offline stage (once per backbone)
  Build pooled sentence-level readout bank $\mathcal{R} = \{r_t\}$ from trajectories in $\mathcal{D}$
  Cluster $\mathcal{R}$ into K recurring logic states; obtain centroids $\{c_k\}_{k=1}^K$
  Construct soft targets $\alpha_t^\star$ and fit layer-wise probes $q_t^{(\ell)}$
  Optimize state-conditioned head gates with the backbone frozen and cache templates $M_k = \{m_k^{(\ell,h)}\}$
  Save artifact $\mathcal{A}$
Online stage (for each $x \in \mathcal{X}$ under fixed strategy π)
  Initialize the paradigm-specific prompt and begin sentence-level generation
for each completed reasoning sentence step t **do**
  Read layer-wise sentence readouts $r_t^{(\ell)}$ at routed layers
  Infer logic activations $q_t^{(\ell)}$ using the cached probes
  Compose routed coefficients $\gamma_{t+1}^{(\ell,h)} = \text{clip}\Big(\sum_k q_t^{(\ell)}(k) m_k^{(\ell,h)}, 0, 1\Big)$
  Instantiate the pathway $\mathcal{P}_{t+1}$ and generate the next reasoning sentence
end for
Return the routed trajectory, final answer, and routing diagnostics

Algorithm 1 is a direct algorithmic restatement of Sec. 2: LoTR does not replace the external reasoning strategy π, but improves step-level internal execution by instantiating pathway coefficients γ according to the current logic activations.

A.1 Trajectory Pooling and Sentence-Level Readout Interface

To keep the controller reusable across reasoning paradigms, LoTR first converts all collected trajectories into a unified sentence-level readout bank. The readout interface follows Sec. 2.2:

$$r_t = \phi(p, y_{\le t}; f_\theta) \in \mathbb{R}^d. \tag{5}$$

Each bank entry pairs a readout r_t with its outer paradigm π, and clustering plus probe fitting operate on this paradigm-pooled bank.

Sentence boundary convention. Online routing is triggered only when a sentence boundary is reached, e.g., a newline or other configured delimiter. This matches the sentence-level control granularity in Sec. 2.3, avoids token-level oscillation, and keeps probe invocation lightweight.

Artifact boundary. The offline stage outputs one reusable artifact:

$$\mathcal{A} = \{\{c_k\}_{k=1}^K,\ \{A^{(\ell)}, b^{(\ell)}\}_{\ell \in \mathcal{L}_{\text{route}}},\ \{M_k\}_{k=1}^K,\ \eta\}, \tag{6}$$

which is reused by all online runs on the same backbone. Here η denotes the serialized compilation metadata, including dimensions, layer conventions, probe/template construction settings, and the preprocessing specification; fitted PCA statistics are not required by online routing. Offline choices

such as the selected K, soft-target temperature τ, and routed-layer set $\mathcal{L}_{\text{route}}$ are fixed at compile time; backbone weights remain frozen throughout. Serialized in the release format, the complete LoTR artifacts occupy 3.05 MiB for Llama-3.1-8B, 5.70 MiB for Qwen2.5-14B, and 3.05 MiB for Mixtral-8×7B. They contain only the learned state centroids, layer-wise probe parameters, pathway gates, preprocessing specification, and configuration metadata; prompts and generated trajectories are not part of the deployed artifact. Online deployment requires read/write hooks for intermediate attention-head outputs and is therefore designed for locally served checkpoints. No model weight is updated, and removing the hooks restores the underlying generation path.

B Offline Algorithm

This section details how LoTR compiles a shared logic basis and corresponding pathway templates from pooled trajectories. The offline stage uses one unified sentence-level readout bank, one shared clustering space, and one layer-wise probe family, all tied to the same frozen backbone.

Algorithm 2 LoTR Offline Compilation

Require: Pooled compilation trajectory set $\mathcal{D}$, frozen backbone f_θ, routed layers $\mathcal{L}_{\text{route}}$, K-search range $[K_{\min}, K_{\max}]$, temperature τ
Ensure: Centroids $\{c_k\}_{k=1}^{K}$, trained probe parameters $\{A^{(\ell)}, b^{(\ell)}\}_{\ell \in \mathcal{L}_{\text{route}}}$, pathway templates $\{M_k\}_{k=1}^{K}$, reusable artifact $\mathcal{A}$
Step 1: Build sentence-level readout bank
for each trajectory step t in $\mathcal{D}$ **do**
  Extract sentence readout $r_t = \phi(p, y_{\le t}; f_\theta)$
  Store readout r_t with paradigm label π
end for
Step 2: Logic-state discovery
Fit standardization and PCA on $\mathcal{R}$, row-normalize $\widetilde{r}_t = \psi(r_t)$, and cluster $\{\widetilde{r}_t\}$
Compare candidate cardinalities using compilation-stage cluster/probe diagnostics and separate development runs, freeze K before release evaluation, and obtain centroids $\{c_k\}_{k=1}^{K}$ and assignments $\widehat{k}_t$
Step 3: Soft state targets
for each stored step t **do**

  Compute distance softmax z_t in PCA space and $\alpha_t^\star = 0.85 z_t + 0.15 e_{\widehat{k}_t}$
end for
Step 4: Probe fitting
for each routed layer $\ell \in \mathcal{L}_{\text{route}}$ **do**
  Extract layer-wise sentence readouts $r_t^{(\ell)}$
  Fit sigmoid probe $q_t^{(\ell)} = \sigma(\beta(A^{(\ell)} r_t^{(\ell)} + b^{(\ell)}))$
end for
Step 5: Template construction
for each state k **do**

  Assign balanced correct/incorrect compilation sentences to state k with the frozen final-layer probe
  Optimize only $G_k = \sigma(\Theta_k)$ using Eq. (3), fixing foundation-layer gates to one
  Cache $M_k \leftarrow G_k$
end for
Save artifact $\mathcal{A} \leftarrow \{\{c_k\}, \{A^{(\ell)}, b^{(\ell)}\}, \{M_k\}, \eta\}$

Implementation alignment. Algorithm 2 is a direct expansion of Sec. 2.2: sentence-level readout extraction, pooled clustering with quality-based K selection, soft state targets, layer-wise probe fitting, and template caching for online reuse.

B.1 Cluster Selection and Soft Target Construction

Let $\mathcal{R} = \{r_t\}_{t=1}^{N}$. We fit standardization and a three-component PCA transformation on the compilation readouts, then ℓ_2-normalize each row before clustering. Equation (26) blends the distance softmax with the assigned-cluster one-hot vector. Although these training targets sum to one, the independently sigmoid-activated online probe outputs do not, allowing co-activation

during routing. All preprocessing, K, and probe-selection decisions are frozen before the evaluation questions are decoded.

B.2 Layer-Wise Probe Fitting and Template Caching

For each routed layer $\ell \in \mathcal{L}_{\text{route}}$, we fit probe outputs by Eq. (1) using the objective in Eq. (2). Then, for each state k, we optimize the bounded gates over $\mathcal{H}_{\text{route}}$ by Eq. (3); the backbone and probes remain frozen. The resulting cached template family $\{M_k\}_{k=1}^{K}$ is consumed directly by the online routing rule in Eq. (4).

Complexity summary. Offline complexity is dominated by (i) clustering over N sentence-level readouts and (ii) one frozen-backbone gate-fitting pass for each state. This cost is paid once per backbone and is not part of online inference. Online complexity at each sentence boundary is dominated by one probe forward per routed layer plus one template mixture, namely

$$O\left(|\mathcal{L}_{\text{route}}|K + \sum_{\ell \in \mathcal{L}_{\text{route}}} H_\ell K\right). \tag{7}$$

This is small relative to the cost of the wrapped backbone forward pass, which is why LoTR adds only lightweight overhead at runtime.

C Online Algorithm

This section details runtime LoTR inference under any fixed external reasoning paradigm π. Online LoTR does not modify the external reasoning policy itself; it only updates the internal pathway at sentence boundaries using the cached offline artifact.

Algorithm 3 LoTR Online Sentence-Level Routing

Require: Frozen backbone f_θ, offline artifact $\mathcal{A}$, fixed reasoning strategy π, prompt p, maximum reasoning steps $T_{\max}$
Ensure: Routed trajectory $y_{1:T}$, final answer a, routing diagnostics

Initialize $y_{<1} \leftarrow \varnothing$
Initialize pathway coefficients $\gamma_1^{(\ell,h)} \leftarrow 1$ for all routed heads
for $t = 1, 2, \ldots, T_{\max}$ **do**
  Generate reasoning sentence y_t under the current pathway $\mathcal{P}_t$ and fixed strategy π
  Check whether the generated step reaches a sentence boundary
  if a sentence boundary is reached **then**
    Extract layer-wise sentence readouts $\{r_t^{(\ell)}\}_{\ell \in \mathcal{L}_{\text{route}}}$
    for each routed layer $\ell \in \mathcal{L}_{\text{route}}$ **do**
      Compute state activations $q_t^{(\ell)} \leftarrow \sigma(\beta(A^{(\ell)} r_t^{(\ell)} + b^{(\ell)}))$
      Compose routed coefficients

$$\gamma_{t+1}^{(\ell,h)} \leftarrow \text{clip}\left(\sum_{k=1}^{K} q_t^{(\ell)}(k)\, m_k^{(\ell,h)}, 0, 1\right), \qquad (\ell,h) \in \mathcal{H}_{\text{route}}$$

    end for
    Instantiate the next pathway $\mathcal{P}_{t+1} = \{\gamma_{t+1}^{(\ell,h)}\}_{(\ell,h)\in\mathcal{H}}$, keeping non-routed coefficients at one
  else
    Keep the current routed pathway unchanged for subsequent token generation
  end if
  if the external reasoning paradigm reaches its stop condition **then**
    break
  end if
end for
Decode final answer a from the routed trajectory

The key runtime property is low overhead: probe evaluation is triggered only at sentence boundaries, while pathway updates are obtained by lightweight template mixing over the cached artifact.

C.1 Boundary-Triggered Routing Dynamics

LoTR updates routing only when a sentence step is completed. Between two sentence boundaries, the current pathway $\mathcal{P}_t$ is kept fixed and is used for all subsequent token generation inside that step. This design matches the sentence-level control granularity of Sec. 2.3: it avoids token-level oscillation, keeps the controller stable, and makes the online cost independent of token-by-token routing updates.

Initialization. At the first reasoning step, all routed coefficients are initialized to 1, so the first sentence is generated under the unmodified backbone pathway. The first probe-and-route update is therefore triggered only after the first sentence boundary is reached, at which point LoTR instantiates $\mathcal{P}_2$ for the next step.

Boundary and update controls. In the frozen runtime, newline and period delimiter tokens trigger a boundary check. We require at least six newly generated tokens since the previous boundary, impose a two-token cooldown between accepted updates, and cap each generation at eight routing updates. A delimiter that fails these checks leaves the current pathway unchanged. These controls are fixed for both Identity and LoTR arms and for every retained dataset and paradigm.

Causal head gating. Each coefficient scales its head's concatenated output immediately before the attention output projection; applying it after that projection would no longer isolate a head. Because the evaluator recomputes the prefix without a key–value cache, it stores the gate assigned to every causal position: a new sentence receives the route inferred at the preceding boundary, while gates on the prompt and completed sentences remain unchanged. Thus, a later routing decision never retroactively alters an earlier token.

Paradigm compatibility. The online controller is agnostic to the outer reasoning scaffold. Different paradigms may impose different stop rules, revision loops, sampling branches, or search states, but LoTR only requires access to the current sentence boundary, the corresponding layer-wise readouts, and the next-step generation interface. This is why the same offline artifact can be reused across heterogeneous reasoning paradigms on the same backbone.

Part B: Theoretical Analysis

D Why Existing Paradigms Can Underexplore the Frontier

This section formalizes the central claim of the paper: external reasoning paradigms can differ in trajectory mechanism while still sharing one limitation, namely the lack of explicit control over the relationship between dynamic logic state and internal computation pathway.

Setup. Fix an external reasoning strategy π. Let the one-step utility under pathway $\mathcal{P}_t$ be

$$J_t(\pi, \mathcal{P}_t) = \mathbb{E}[u_t \mid p, y_{<t}; \pi, \mathcal{P}_t]. \tag{8}$$

where u_t is a bounded step-quality utility. Let $k \in \{1, \dots, K\}$ index the recurring logic states. We write $\tilde{\alpha}_t^\star(k)$ for the normalized state weights induced by the current step, so that $\sum_{k=1}^{K} \tilde{\alpha}_t^\star(k) = 1$. We use $J_t(\pi, \mathcal{P} \mid k)$ to denote the same one-step utility conditioned on the current state k.

For each state k, define a state-optimal pathway by

$$\mathcal{P}_{t,k}^\star \in \arg\max_{\mathcal{P}} J_t(\pi, \mathcal{P} \mid k). \tag{9}$$

Assumption D.1 (state-sensitive optimality). There exist two states $k \neq k'$ with $\tilde{\alpha}_t^\star(k) > 0$ and $\tilde{\alpha}_t^\star(k') > 0$ such that no single pathway $\mathcal{P}$ is simultaneously optimal for both states. Equivalently,

$$\forall \mathcal{P}, \quad \exists j \in \{k, k'\} \text{ such that } J_t(\pi, \mathcal{P} \mid j) < J_t(\pi, \mathcal{P}_{t,j}^\star \mid j).$$

Assumption D.2 (External-only control). An external-only controller may vary the outer strategy π, but for fixed π it uses one shared pathway rule $\bar{\mathcal{P}}_t^{(\pi)}$ across all states and does not condition $\mathcal{P}_t$ on the current state k.

Proposition D.1 (Frontier underexploration under state mismatch). Under D.1–D.2, for fixed π,

$$\mathbb{E}_{k \sim \tilde{\alpha}_t^\star} J_t(\pi, \bar{\mathcal{P}}_t^{(\pi)} \mid k) \;<\; \mathbb{E}_{k \sim \tilde{\alpha}_t^\star} J_t(\pi, \mathcal{P}_{t,k}^\star \mid k). \tag{10}$$

Proof. For every state k, by definition of $\mathcal{P}_{t,k}^\star$,

$$J_t(\pi, \bar{\mathcal{P}}_t^{(\pi)} \mid k) \leq J_t(\pi, \mathcal{P}_{t,k}^\star \mid k). \tag{11}$$

Multiplying by $\tilde{\alpha}_t^\star(k)$ and summing over k yields

$$\mathbb{E}_{k \sim \tilde{\alpha}_t^\star} J_t(\pi, \bar{\mathcal{P}}_t^{(\pi)} \mid k) \leq \mathbb{E}_{k \sim \tilde{\alpha}_t^\star} J_t(\pi, \mathcal{P}_{t,k}^\star \mid k).$$

It remains to show strictness. By Assumption D.1, among the positive-mass states there exist $k \neq k'$ such that no single pathway is optimal for both. Since $\bar{\mathcal{P}}_t^{(\pi)}$ is shared across states by Assumption D.2, it must be strictly suboptimal for at least one positive-mass state $j \in \{k, k'\}$. Hence

$$J_t(\pi, \bar{\mathcal{P}}_t^{(\pi)} \mid j) < J_t(\pi, \mathcal{P}_{t,j}^\star \mid j),$$

for some j with $\tilde{\alpha}_t^\star(j) > 0$. Therefore the weighted sum is also strict, giving Eq. (10). □

Interpretation. This proposition complements stronger external procedures: even under a fixed external strategy, explicitly coupling states to pathways can improve execution whenever positive-mass logic states require different optimal pathways.

E Why LoTR Improves Fixed Paradigms

This section formalizes why LoTR can improve a fixed external reasoning strategy π without altering its outer program: the gain comes from better matching the internal pathway to the step-level logic state mixture currently induced by the trajectory.

Setup. Fix an external reasoning strategy π. Let $\tilde{\alpha}_t^\star \in \Delta^{K-1}$ denote the normalized ideal state mixture induced by the current reasoning step, where Δ^{K-1} is the probability simplex, and let $\mathcal{P}_t^\star$ denote the corresponding state-optimal pathway under π. Define the ideal template mixture

$$\widetilde{\mathcal{P}}_t := \sum_{k=1}^{K} \tilde{\alpha}_t^\star(k) M_k, \tag{12}$$

and let

$$\bar{q}_t := \frac{1}{|\mathcal{L}_{\text{route}}|} \sum_{\ell \in \mathcal{L}_{\text{route}}} q_t^{(\ell)}$$

be the layer-aggregated probe mixture. The LoTR pathway instantiated online is

$$\widehat{\mathcal{P}}_t := \sum_{k=1}^{K} \bar{q}_t(k) M_k, \tag{13}$$

which abstracts the layer-wise routing rule in Eq. (4) at the pathway level. For notational simplicity, this section indexes the pathway by the decision step it is intended to execute; thus $\widehat{\mathcal{P}}_t$ should be read as the pathway instantiated from the current state mixture for the next generated reasoning step.

Assumption E.1 (Probe consistency). The aggregated probe prediction error is bounded:

$$\mathbb{E}[\|\bar{q}_t - \tilde{\alpha}_t^\star\|_1] \le \epsilon_q. \tag{14}$$

Assumption E.2 (Template approximation). The ideal state mixture over cached templates approximates the state-optimal pathway:

$$\mathbb{E}\Big[\|\widetilde{\mathcal{P}}_t - \mathcal{P}_t^\star\|_1\Big] \le \epsilon_m. \tag{15}$$

Assumption E.3 (Pathway-Lipschitz utility). For fixed π, one-step utility is L_u-Lipschitz in pathway:

$$|J_t(\pi, \mathcal{P}) - J_t(\pi, \mathcal{P}')| \le L_u \|\mathcal{P} - \mathcal{P}'\|_1. \tag{16}$$

Assumption E.4 (Bounded template mixing). The template mixing operator is L_M-Lipschitz with respect to the state mixture, i.e.,

$$\left\| \sum_{k=1}^{K} \beta(k) M_k - \sum_{k=1}^{K} \beta'(k) M_k \right\|_1 \le L_M \|\beta - \beta'\|_1 \qquad \forall \beta, \beta' \in [0,1]^K. \tag{17}$$

Proposition E.1 (Additive approximation bound under fixed π). Under E.1–E.4,

$$\mathbb{E}[J_t(\pi, \widehat{\mathcal{P}}_t)] \ge \mathbb{E}[J_t(\pi, \mathcal{P}_t^\star)] - L_u\big(L_M \epsilon_q + \epsilon_m\big). \tag{18}$$

Hence, when $L_M \epsilon_q + \epsilon_m$ is small, LoTR approaches state-optimal execution under the same fixed strategy π.

Proof. By Assumption E.3,

$$J_t(\pi, \widehat{\mathcal{P}}_t) \ge J_t(\pi, \mathcal{P}_t^\star) - L_u \|\widehat{\mathcal{P}}_t - \mathcal{P}_t^\star\|_1. \tag{19}$$

Insert $\widetilde{\mathcal{P}}_t$ and apply the triangle inequality:

$$\|\widehat{\mathcal{P}}_t - \mathcal{P}_t^\star\|_1 \le \|\widehat{\mathcal{P}}_t - \widetilde{\mathcal{P}}_t\|_1 + \|\widetilde{\mathcal{P}}_t - \mathcal{P}_t^\star\|_1. \tag{20}$$

For the first term, using the definitions of $\widehat{\mathcal{P}}_t$ and $\widetilde{\mathcal{P}}_t$ together with Assumption E.4 gives

$$\|\widehat{\mathcal{P}}_t - \widetilde{\mathcal{P}}_t\|_1 = \left\| \sum_{k=1}^{K} \bar{q}_t(k) M_k - \sum_{k=1}^{K} \tilde{\alpha}_t^\star(k) M_k \right\|_1 \le L_M \|\bar{q}_t - \tilde{\alpha}_t^\star\|_1. \tag{21}$$

Taking expectations and applying Assumptions E.1 and E.2 yields

$$\mathbb{E}\Big[\|\widehat{\mathcal{P}}_t - \mathcal{P}_t^\star\|_1\Big] \le L_M \epsilon_q + \epsilon_m. \tag{22}$$

Substituting Eq. (22) into Eq. (19) proves Eq. (18). □

Interpretation. Proposition E.1 formalizes the mechanism claim in the main paper: LoTR does not need to alter the external reasoning trajectory to improve a fixed paradigm. Its gain comes from reducing the mismatch between the logic state mixture expressed by the current step and the internal pathway used to execute the next one.

F Why Logic States Generalize Across Problems and Paradigms

This section formalizes the design choice of pooling trajectories from multiple reasoning paradigms and fitting one shared probe family. The key claim is that if the geometry of state-conditioned readouts is sufficiently stable across paradigms, then a pooled probe can remain effective beyond the paradigms used for offline compilation.

Setup. Fix a routed layer $\ell \in \mathcal{L}_{\text{route}}$. Let $r_t^{(\ell)} \in \mathbb{R}^d$ denote the sentence-level readout at step t, and let $k \in \{1, \dots, K\}$ denote the recurring logic state induced by the shared offline basis. Let $\mathcal{D}_\pi$ denote the joint distribution of $(r_t^{(\ell)}, k)$ under external reasoning strategy π, and let

$$\mathcal{D}_{\text{pool}} := \sum_\pi w_\pi \mathcal{D}_\pi$$

be the pooled training distribution over paradigms, with $w_\pi \geq 0$ and $\sum_\pi w_\pi = 1$. For a probe $g : \mathbb{R}^d \to [0,1]^K$ and bounded loss $\ell(g(r), k) \in [0,1]$, define

$$\mathcal{R}_\pi(g) := \mathbb{E}_{(r,k)\sim\mathcal{D}_\pi} \ell(g(r), k), \qquad \mathcal{R}_{\text{pool}}(g) := \mathbb{E}_{(r,k)\sim\mathcal{D}_{\text{pool}}} \ell(g(r), k). \tag{23}$$

Let $\hat{g}_{\text{pool}}$ denote a pooled risk minimizer (or its empirical approximation).

Assumption F.1 (Shared state geometry). For every state k, the conditional readout distributions are stable across paradigms up to bounded total-variation shift:

$$\sup_{\pi,\pi'} \text{TV}\big(p(r_t^{(\ell)} \mid k, \pi),\, p(r_t^{(\ell)} \mid k, \pi')\big) \leq \delta. \tag{24}$$

Assumption F.2 (Shared state support). The target paradigm does not introduce logic states outside the pooled support, so both pooled and target risks decompose over the same state index set $\{1, \dots, K\}$.

Proposition F.1 (Cross-paradigm transferability of a pooled probe family). Under F.1–F.2, for any target paradigm π',

$$\mathcal{R}_{\pi'}(\hat{g}_{\text{pool}}) \leq \mathcal{R}_{\text{pool}}(\hat{g}_{\text{pool}}) + \epsilon_{\text{shift}} + \epsilon_{\text{marg}}, \tag{25}$$

where $\epsilon_{\text{shift}} = O(\delta)$ controls state-conditioned geometry shift and ϵ_{marg} captures the mismatch between pooled and target state marginals.

Proof. Conditioning on the state index,

$$\mathcal{R}_{\pi'}(g) = \sum_{k=1}^{K} \Pr_{\pi'}(k)\, \mathbb{E}_{r\sim p(r|k,\pi')} \ell(g(r), k),$$

and likewise

$$\mathcal{R}_{\text{pool}}(g) = \sum_{k=1}^{K} \Pr_{\text{pool}}(k)\, \mathbb{E}_{r\sim p(r|k,\text{pool})} \ell(g(r), k).$$

The target–pooled risk gap has two sources: conditional readout shift within each state and mismatch in state marginals. Because $\ell \in [0,1]$, the first is bounded by the corresponding total-variation discrepancy and hence by an $O(\delta)$ term, ϵ_{shift}, under F.1; the second contributes ϵ_{marg}. Applying the decomposition to $g = \hat{g}_{\text{pool}}$ proves Eq. (25). □

Interpretation. Proposition F.1 justifies the pooled offline strategy of LoTR. If recurring logic states induce sufficiently stable readout geometry across paradigms and state support is shared, then one pooled probe family can remain effective beyond the paradigms used for compilation. This is why LoTR can pool heterogeneous reasoning trajectories offline, build one shared logic basis, and still reuse one probe/template system online.

Part C: Experimental Setup

G Datasets, Models, Paradigms, Metrics, and Protocol

Datasets and task families. We report ten benchmarks across the model-specific panels summarized in Table 6. The primary Llama study contains eight benchmarks and the full eight-paradigm matrix; the complementary Qwen and Mixtral studies use fixed Self-Refine and CoT panels, respectively. Dataset membership is constant across all methods within each reported comparison. **BoolQ** (Clark et al., 2019) asks yes/no questions grounded in Wikipedia passages. **HumanEval** (Chen et al., 2021) measures functional correctness on Python docstring-to-code synthesis. **DROP** (Dua et al., 2019) tests discrete reasoning over passages; **QASC** (Khot et al., 2020) tests multi-hop composition of scientific facts; **ScienceQA-text** (Lu et al., 2022) uses the text-only portion of multimodal science questions; **CODAH** (Chen et al., 2019) evaluates adversarial commonsense completion; and **HellaSwag** (Zellers et al., 2019) evaluates contextual commonsense inference. We additionally retain **MMLU** (Hendrycks et al., 2021), **FOLIO** (Han et al., 2024), and **NarrativeQA** (Kočiský et al., 2018) to cover broad-domain knowledge, natural-language deduction, and long-form reading comprehension. Unless noted otherwise, prompts and graders follow the dataset releases packaged with our evaluation code.

Backbone models. We evaluate **Llama-3.1-8B** (Grattafiori et al., 2024), **Qwen2.5-14B** (Yang et al., 2024), and **Mixtral-8×7B** (Jiang et al., 2024). Llama and Qwen are evaluated in bfloat16. Mixtral uses 4-bit weight quantization with bfloat16 computation so that the sparse-MoE checkpoint and its controller fit on one 80-GiB accelerator. Every Identity, LoTR, and standalone-baseline comparison for a given backbone uses the same precision and checkpoint; results are compared within backbone rather than across numerical formats.

Reasoning paradigms. LoTR is evaluated under eight *external* scaffolds (including Vanilla). **Vanilla** is greedy completion without an explicit reasoning recipe. **Chain-of-Thought (CoT)** (Wei et al., 2022) elicits step-by-step rationales before an answer. **Plan-and-Solve (PS)** (Wang et al., 2023a) separates planning from execution for structured decomposition. **Self-Refine (SR)** (Madaan et al., 2023) iterates draft–critique–revise loops under a scripted program. **Self-Consistency (SC)** (Wang et al., 2023b) samples multiple chains and aggregates by majority vote. **Best-of-N (BoN)** (Chow et al., 2024; Lightman et al., 2024) draws N candidates and reranks or selects the strongest finish. **Constrained Beam (CB)** (Hokamp & Liu, 2017; Post & Vilar, 2018) expands a bounded beam of hypotheses with lexical or structural constraints at each step. **Monte Carlo Tree Search (MCTS)** (Xie et al., 2024) budgets iterative expansion/scoring of partial reasoning states before committing to an answer. LoTR never replaces π: it only swaps internal attention pathways between sentence boundaries while π schedules decoding.

Metrics. We report benchmark-native accuracy (accuracy, F1, or execution-based Pass@1 as in Table 6) Each macro weights the datasets in its stated panel equally, and every reported gain is an absolute percentage-point difference unless explicitly labeled otherwise. Identity, LoTR, and standalone baselines use the same items and dataset-specific scorer within a comparison. We compute paired item-level 95% bootstrap intervals and exact binary paired tests for each cell, and cell-bootstrap intervals for macro claims; the per-cell statistics accompany the result manifest. Generated-token counts are recomputed from the final raw records: we distinguish the final-call completion length from the sum over all calls and use the latter as the decoding-work proxy for multi-call paradigms. Latency comparisons use paired runs under the same backbone-specific software and hardware configuration.

Trajectory collection and offline pooling protocol. For offline compilation we pool sentence-level readouts from trajectories logged under the paradigms above, merge them into $\mathcal{R}$, and fit one clustering/probe/template artifact per backbone. No backbone weights are updated. For each backbone, the compilation bank contains approximately 6,400 problem–paradigm trajectories from GSM8K, MMLU, FOLIO, and HotpotQA (about 200 problems per dataset under each of eight paradigms). Compilation and evaluation questions are disjoint. For the deployed Llama artifact, clustering uses 12,460 sentence readouts from correct compilation trajectories; template fitting then draws 64 correct and 64 incorrect state-assigned sentences per state after a deterministic record

shuffle and samples 12+12 sentences per update. The same procedure is run once per backbone, after which one compiled artifact is reused without task- or scaffold-specific fitting across every reported evaluation panel.

Offline cost. Compilation is a one-time per-backbone cost amortized across all downstream datasets and paradigms. The Llama bank contains 879,382 generated completion tokens and 1,936,233 prompt tokens (2.816M processed tokens in total). After readout extraction, the final six-state template optimization—80 updates per state with 12 correct and 12 incorrect sentences per update—takes 19.2 minutes on one RTX 6000 Ada GPU. Trajectory collection and readout extraction remain the dominant hardware-dependent costs; neither is repeated during online evaluation.

H BASELINES

Plug-in baselines. Besides raw paradigm runs, we compare against inference-time steering methods that keep weights frozen. **Inference-Time Intervention (ITI)** (Li et al., 2023) applies additive activations along selected attention-head directions discovered from contrastive pairs. **Activation Addition (ActAdd)** (Turner et al., 2023) steers generations by injecting fixed activation offsets during forward passes. **STIR** (Shi et al., 2026) retrieves latent steering impulses conditioned on the ongoing trajectory rather than using a single static vector. **RISER** (Ye et al., 2026) routes among multiple reusable steering experts so impulses adapt to inferred reasoning regimes.

Baseline coverage. These families respectively cover head masking/addition, global offsets, retrieval-conditioned impulses, and routed mixtures—complementary axes of inference-time control—while remaining comparable to LoTR in compute regime (no weight updates).

Baseline configuration and selection boundary. Each method uses one fixed intervention configuration per backbone across all retained datasets. ITI uses magnitude 0.5 at layers 15–17, ActAdd uses magnitude 1.0 at layer 20, and RISER uses magnitude 0.5 at layer 16. STIR uses temperature 1.0 at layer 18, with one global magnitude per backbone (0.006/0.008/0.006 for Llama/Qwen/Mixtral). The Llama and Qwen STIR magnitudes are selected by a frozen four-value macro grid on held-out development examples; the Llama magnitude is transferred to Mixtral. All configurations are fixed before evaluation.

I HYPERPARAMETER CONFIGURATION

Configuration principles. We compile one offline artifact per backbone using pooled trajectories, then freeze $\{K, \tau, \mathcal{L}_{\text{route}}\}$, probes, and templates for every benchmark and paradigm. The same artifact is reused across every retained dataset and reasoning paradigm for that backbone. The evaluation questions are excluded from clustering, probe fitting, template fitting, and hyperparameter selection. Table 7, 9 lists the deployed settings.

Table 6: **Evaluation scope and benchmark-native metrics.** Checkmarks indicate inclusion in the Llama (L), Qwen (Q), and Mixtral (M) panels.

Dataset	L	Q	M	Task type	Primary metric
BoolQ	✓	✓		Yes/no reading QA	Accuracy
DROP	✓		✓	Discrete multi-hop QA	F1
HumanEval	✓		✓	Python code synthesis	Pass@1
QASC	✓	✓	✓	Science fact composition	Accuracy
ScienceQA-text	✓	✓		Science multiple choice	Accuracy
MMLU	✓	✓		Multi-domain MC QA	Accuracy
NarrativeQA	✓			Long-document QA	F1
FOLIO	✓			Logic-in-NL inference	Accuracy
CODAH		✓	✓	Commonsense completion	Accuracy
HellaSwag		✓		Contextual commonsense	Accuracy

Soft-target construction. Let $\widetilde{r}_t = \psi(r_t) \in \mathbb{R}^3$ denote the standardized, PCA-projected, and row-normalized sentence readout, and let $\widehat{k}_t$ be its assigned cluster. The probe target used by the released artifact is

$$z_t(k) = \frac{\exp\big(-\|\widetilde{r}_t - c_k\|_2^2/(2\tau^2)\big)}{\sum_{j=1}^{K} \exp(-\|\widetilde{r}_t - c_j\|_2^2/(2\tau^2))}, \qquad \alpha_t^\star = 0.85 z_t + 0.15 e_{\widehat{k}_t}, \tag{26}$$

Here $e_{\widehat{k}_t}$ is the one-hot vector of the assigned cluster and $\tau = 1$ is the distance-softmax temperature. This construction and all constants are fixed during artifact compilation and shared by every evaluation dataset for that backbone.

Table 7: **LoTR hyperparameters** (offline compilation + deployment).

Component	**Setting**	**Value**
Clustering	Cardinality K	6
Clustering	Readout preprocessing	standardize, PCA-3, ℓ_2-normalize
Soft targets	Temperature τ	1.0
Probe training	AdamW learning rate	10^{-2}
Probe training	Weight decay	10^{-4}
Probe training	Epochs (per routed layer)	30
Probe training	Batch size	256
Probe training	Validation fraction	0.1
Template fitting	Adam steps / learning rate	80 / 0.05
Template fitting	Contrast / identity weights (λ, μ)	(0.3, 0.3)
Template fitting	Sentences/state; sampled/update	64+64; 12+12
Routing	Foundation / routed layers (0-based)	0–2 / 3–$L-1$
Routing	Probe-logit scale β (Llama/Qwen/Mixtral)	0.5 / 0.25 / 0.5
Routing	Maximum sentence updates	8
Online routing	γ clip (elementwise)	$[0, 1]$

Table 8: **Probe-scale sensitivity on Llama-3.1-8B development data.** Entries are mean LoTR–Identity accuracy differences (percentage points) over BoolQ, FOLIO, GSM8K, and MMLU.

Probe scale β	0.20	0.35	0.45	0.50	0.65	0.80
Mean Δ	-2.75	+1.56	+0.45	**+2.43**	+0.25	-1.07

Table 9: **Evaluation decoding hyperparameters.**

Baseline	Parameter	Value
Vanilla	Max new tokens	512
	Temperature	0
CoT	Max new tokens	768
	Temperature	0
PS	Planning tokens	96
	Solving tokens	320
	Temperature	0
SR	Draft tokens	256
	Critique tokens	96
	Revision tokens	256
	Temperature	0
SC / BoN	Sample count N	3
	Temperature	0.7
	Max tokens/sample	384
CB	Beam width	3
	Maximum depth	3
	Branch factor	2
	Reasoning (cap)	56
	Judge (cap)	12
	Answer (cap)	256
	Tree budget	3200
	Temperature	0
MCTS	Iterations	6
	Branching factor	2
	c_{puct}	1.2
	Reasoning (cap)	56
	Judge (cap)	12
	Answer (cap)	256
	Tree budget	3200
	Temperature	0

Part D: Additional Results and Analysis

J Pipeline-Level Analysis

LoTR depends on a two-stage mechanism: the offline stage must recover a compact set of stable logic states from sentence-level readouts, and the online stage must use these states to induce differentiated head-routing patterns during reasoning. We analyze the frozen deployed $K=6$ Llama artifact using its 12,460-row PCA-3 compilation matrix, held-out probe split, and final gradient-learned templates.

J.1 Offline stage effects: state readout and pathway differentiation

Final-artifact state readout. The deployed clustering has silhouette 0.566 on the compiler's fixed diagnostic subset, with state supports ranging from 561 to 3,505 sentence readouts. More importantly for online use, the independently selected layer probes achieve 93.3% mean argmax accuracy and 93.6% mean macro recall on the frozen probe-validation split across routed layers 3–31. Thus, intermediate blocks can read out the shared final-layer state basis without introducing a second clustering space.

Final-artifact pathway differentiation. Figure 8 visualizes intervention strengths $1 - M_k$ from the final gradient-learned templates. Across the 928 routed layer–head positions, 557 have a cross-state gate range above 0.05; 14.9% of state–position gates are below 0.5 and 38.9% are below 0.9. The controller therefore preserves much of the backbone while retaining substantial state-dependent pathway variation.

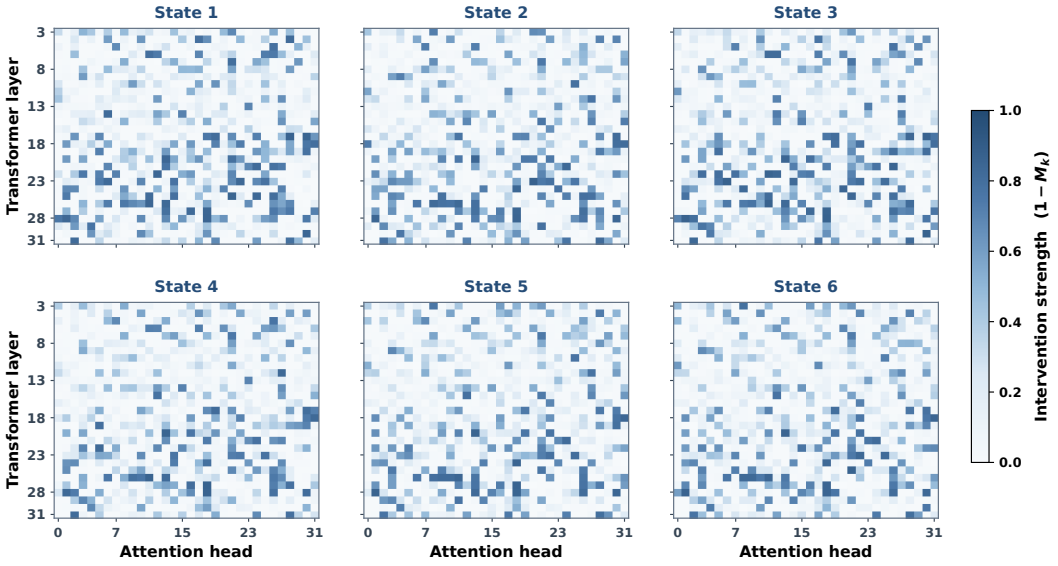


Figure 8: **Pathway templates from the final $K=6$ artifact.** Each panel shows intervention strength $1 - M_k$ over Transformer layers 3–31 and attention heads. State indices are unsupervised identifiers, not semantic annotations.

J.2 Cross-paradigm recurrence under the final $K=6$ basis

Figure 9 counts assignments of the exact compilation readouts to their deployed centroids. Every state occurs under at least four of the eight external paradigms, and five of six states occur under at least seven; the mean normalized paradigm-spread entropy is 0.766. State occupancy also adapts to task demand (adjusted mutual information with dataset identity is 0.422). Together, these measurements support a reusable procedural basis whose mixture responds to the problem being solved, rather than a fixed partition by outer reasoning paradigm.

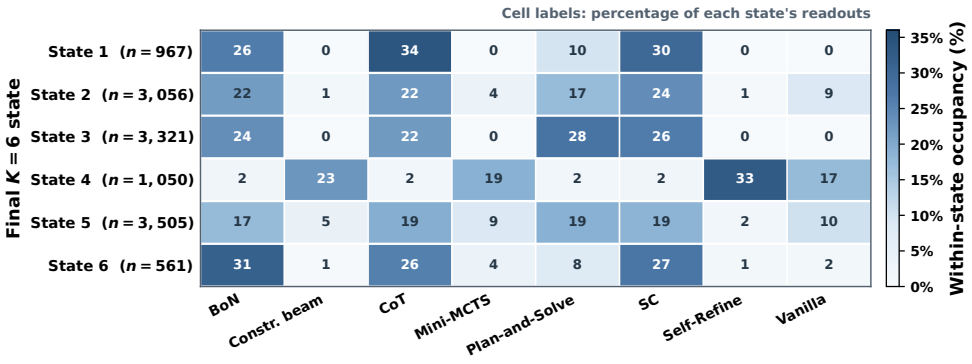


Figure 9: **Cross-paradigm recurrence of the final $K=6$ states.** Rows show within-state occupancy over the eight paradigms, computed from correct offline compilation trajectories.

K Detailed Cross-Backbone and Comparator Results

Table 10 expands the aggregate transfer result into the complete matched Qwen and Mixtral panels and the common-support Llama comparison. Every row places LoTR beside Identity and all four standalone interventions on identical examples and scoring rules. Relative to Identity, Qwen improves on four of six datasets and ties one, while Mixtral improves on three of four and ties one; their macro gains are $+2.12$ and $+0.93$ points, respectively.

Table 10: **Shared-panel plug-in comparisons (%).** Every row uses identical examples, paradigm, checkpoint, and scorer; LoTR parentheses are changes from Identity.

Backbone	Paradigm	Dataset	Identity	ITI	ActAdd	STIR	RISER	LoTR
Llama-3.1-8B	CoT	BoolQ	75.5	73.0	77.5	75.5	75.0	**79.5** (+4.0)
		DROP	11.3	13.4	11.0	11.6	11.6	**13.9** (+2.6)
		HumanEval	54.0	58.0	56.0	52.0	54.0	**58.0** (+4.0)
		QASC	38.4	42.8	43.6	41.2	41.2	**48.4** (+10.0)
		ScienceQA	73.6	78.0	76.0	78.8	77.6	**76.4** (+2.8)
Qwen2.5-14B	Self-Refine	BoolQ	79.0	81.0	83.0	83.0	83.5	**81.5** (+2.5)
		MMLU	76.0	78.0	78.8	78.0	75.2	**77.2** (+1.2)
		QASC	47.0	49.0	49.0	45.0	45.0	**56.0** (+9.0)
		ScienceQA	87.0	91.0	88.0	90.0	92.0	**89.0** (+2.0)
		CODAH	87.0	90.0	88.0	92.0	88.0	**87.0** (0.0)
		HellaSwag	74.0	73.0	73.0	71.0	74.0	72.0 (-2.0)
Mixtral-8×7B	CoT	CODAH	56.4	54.0	55.2	50.8	55.2	**56.4** (0.0)
		DROP	7.6	8.1	7.0	7.8	7.1	**8.9** (+1.3)
		HumanEval	40.0	46.0	42.0	44.0	48.0	**42.0** (+2.0)
		QASC	38.4	40.0	40.0	40.4	36.4	**38.8** (+0.4)

Reading the comparator pattern. The advantage is distributed across complementary regimes. On Llama, LoTR is best or tied on four of five common-support datasets and gains 10.0 points on QASC, separating it from both Identity and the four-method mean. On Qwen, LoTR again supplies the largest QASC gain and raises the matched macro; on Mixtral, the same controller improves the matched macro and remains above the intervention mean. Together, the panels establish breadth across backbone families, scales, and outer scaffolds under fixed within-panel protocols.

Connection to the mechanism. LoTR composes bounded state-indexed templates at sentence boundaries. These cells align with Section J: recurring states induce differentiated pathways whose conditional composition raises matched accuracy across model families, closing the chain from state structure to pathway selection and performance.

Part E: Limitations and Discussion

L LIMITATIONS

Practical value and impact scope. LoTR targets a practical deployment setting where backbone weights and external reasoning scaffolds are fixed, but inference quality/efficiency still need improvement. In this scope, the method can reduce trial-and-error cost when integrating reasoning wrappers and can improve reliability across heterogeneous reasoning tasks and scaffolds.

Method scope.

- **Compilation coverage.** Logic states are compiled from a pooled trajectory bank spanning all eight paradigms. Reasoning modes absent from that bank may benefit from adaptive artifact updates.
- **Per-backbone compilation.** LoTR compiles one lightweight artifact per backbone and then reuses it across tasks and reasoning paradigms. Direct artifact transfer across model families remains an opportunity for further amortization.

Evaluation scope.

- **Backbone and task scope.** Our study covers three model families, ten benchmarks, and eight heterogeneous reasoning paradigms. Extending the same evaluation to longer contexts and additional generation domains is a natural next step.

Deployment considerations.

- **Step-boundary interface.** The current router updates at sentence boundaries, which aligns naturally with explicit reasoning steps. Finer-grained boundaries may be useful for less structured generation formats.
- **Compilation and routing cost.** LoTR adds one-time artifact compilation and lightweight online probes; the measured token–latency profile remains favorable across backbones.

M FUTURE WORK

- **Uncertainty-aware routing.** Add calibrated uncertainty signals so low-confidence state estimates trigger conservative routing updates.
- **Broader robustness evaluation.** Extend evaluation to more out-of-domain tasks and longer-context settings.
- **Artifact-level repeatability.** Study how independently compiled trajectory banks and clustering seeds affect downstream routing while retaining the same global protocol.
- **Cross-backbone alignment.** Study partial transfer of logic states/templates across related backbones to reduce per-backbone compilation cost.