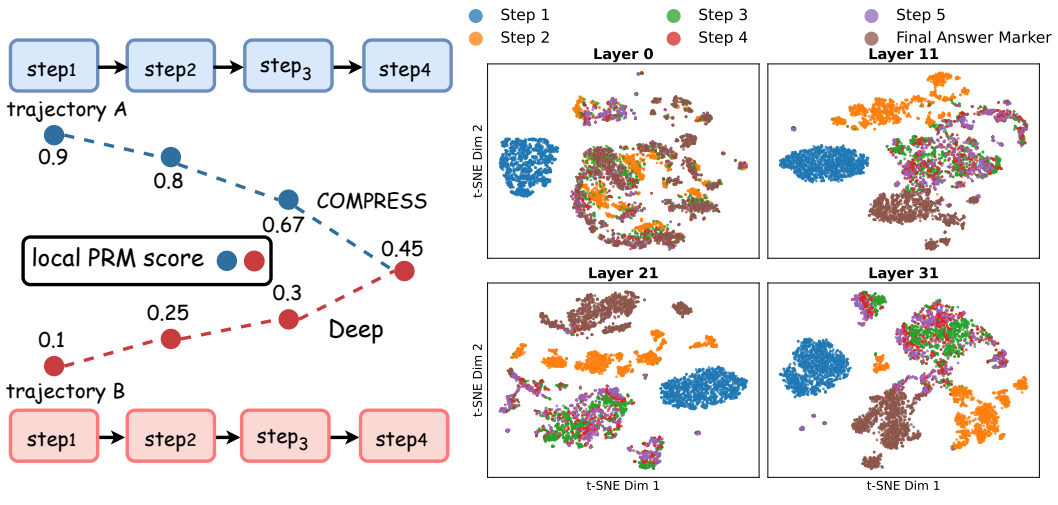
ARC: TRAJECTORY-AWARE ADAPTIVE REASONING CONTROL VIA LATENT-STATE DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Reasoning Models (LRMs) have demonstrated strong capabilities in solving complex tasks through extended chain-of-thought reasoning. However, existing approaches to efficient reasoning primarily rely on truncation, compression, or local step-level signals, limiting adaptive computation allocation across the reasoning trajectory. Our empirical analysis shows that local signals can be ambiguous across different reasoning stages, whereas hidden states capture richer information about the evolution of reasoning. Building on this observation, we propose the **Adaptive Reasoning Controller (ARC)**, a lightweight trajectory-aware controller that leverages the hidden-state history of a frozen reasoning model to select among four discrete actions—*Deep*, *Normal*, *Compress*, and *Conclude*—for regulating subsequent computation. ARC introduces fewer than 2.8M trainable parameters, a negligible overhead over the frozen reasoning model. ARC is initialized through supervised learning on latent reasoning trajectories using pseudo-action labels derived from PRM signals, and is further optimized with Group Relative Policy Optimization (GRPO) using final-answer correctness and reasoning cost as learning signals. Under this hierarchical formulation, the frozen reasoning model serves as the low-level reasoning policy, whereas ARC learns a high-level strategy for adaptive computation allocation. Experiments across four LRMs and four benchmarks show that ARC reduces reasoning tokens by 32.8% on average (up to 55.3%) while increasing macro-averaged accuracy from 58.43% to 58.79%.



(a) Instantaneous Score-Based Control

(b) Layer-wise t-SNE Visualization of Hidden States

Figure 1: (a) Local PRM scores provide ambiguous computation signals for different reasoning trajectories. (b) Layer-wise t-SNE visualizations of hidden states from Llama-3.1-8B-Instruct on GSM8K reveal structured patterns across reasoning steps.

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.

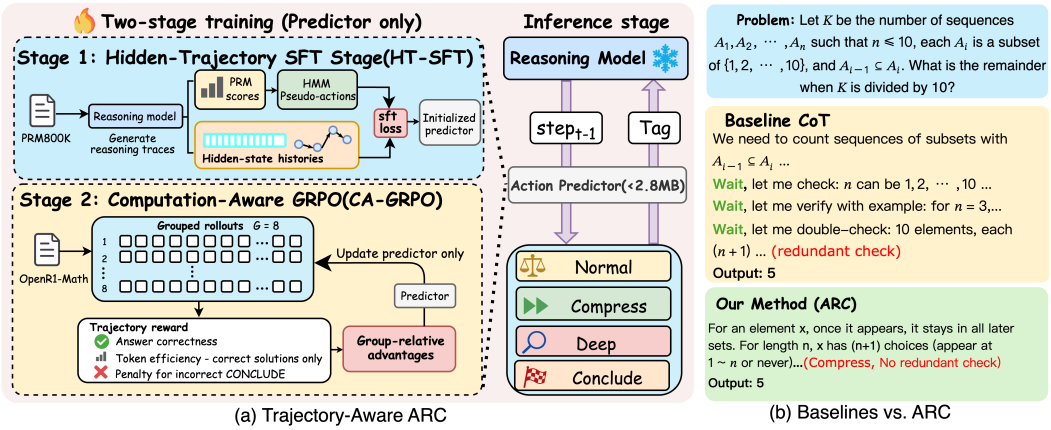
1 INTRODUCTION

Recent advances in large reasoning models (LRMs) have demonstrated remarkable reasoning capabilities on challenging mathematical and logical tasks by generating explicit intermediate reasoning trajectories. Chain-of-thought (CoT) prompting enables models to decompose complex problems into a sequence of intermediate steps and significantly improves reasoning performance (Wei et al., 2022; Kojima et al., 2022; Xia et al., 2025). More recently, reasoning-oriented models have further shown that scaling inference-time computation can substantially boost performance, driven by longer reasoning trajectories and reinforcement learning-based training (Snell et al., 2025; Guo et al., 2025). However, longer reasoning does not always yield better outcomes: excessive reasoning can introduce redundant computation, and the optimal reasoning budget varies considerably across problems and models (Feng et al., 2025; Sui et al., 2025). Therefore, efficient reasoning is not simply a matter of reducing CoT length, but of adaptively allocating computation throughout the reasoning process, deciding when additional reasoning is necessary, when it becomes redundant, and when the model should stop. Existing adaptive reasoning methods typically rely on trajectory truncation, compression, or step-level signals from process reward models (PRMs) (Lightman et al., 2024; Sun et al., 2026; Snell et al., 2025; Xu et al., 2025), where computation actions are determined primarily by instantaneous step-level information, such as PRM scores or hidden representations, without explicitly modeling the global reasoning trajectory. Although various heuristics, including length penalties, confidence thresholds, and transition rules, have been incorporated to improve robustness, temporal consistency is typically treated as an external constraint rather than an intrinsic property of the reasoning trajectory itself. These observations lead us to ask whether reasoning trajectories themselves exhibit latent temporal structures that can support more principled adaptive computation allocation.

To investigate this question, we analyze the internal representations underlying reasoning trajectories and find that hidden states carry informative signals about the evolution of the reasoning process (Fig. 1). Recent studies show that reasoning trajectories exhibit structured geometric patterns in representation space, where step-specific hidden states become increasingly separable across layers and encode signals related to reasoning progression and correctness (Zhang et al., 2025a; Wang et al., 2025; Zhang et al., 2025b; Azaria & Mitchell, 2023). As shown in Fig. 1(a), however, local PRM-based signals only characterize the current reasoning state and may fail to distinguish trajectories with different future computation requirements: a high score does not necessarily imply that further computation is unnecessary, while a low score may reflect either redundant exploration or a genuinely challenging stage. In contrast, the layer-wise hidden-state visualization in Fig. 1(b) reveals structured variations across reasoning steps, suggesting that internal representations progressively accumulate temporal information throughout generation. These observations indicate that computation demand is not merely a property of individual reasoning steps but is reflected in the temporal evolution of hidden states along the trajectory. Since existing PRMs are designed to evaluate the quality of intermediate steps rather than to optimize future computation allocation, mapping their scores to actions through fixed thresholds or heuristic transition rules only crudely approximates the optimal policy. Adaptive reasoning control should instead be formulated as a sequential decision-making problem (Yao et al., 2023), where actions are selected based on their long-term impact on the trajectory and final outcome.

Motivated by these observations, we propose ARC, a lightweight hidden-state-guided controller that formulates adaptive reasoning as a sequential decision process: it monitors the reasoning trajectory of a frozen reasoner and intervenes step by step. At each step, the controller takes the reasoner's hidden-state history as input and predicts a discrete computation action, such as continuing, compressing, or terminating the current reasoning path. This action is then transformed into a structured control signal that steers subsequent generation, allowing ARC to allocate computation adaptively based on the evolving internal state, without modifying or fine-tuning the underlying reasoner.

Our contributions are summarized as follows: (i) We reveal that hidden-state trajectories encode temporal information useful for adaptive reasoning control. (ii) We propose ARC, a lightweight controller that regulates frozen reasoners through discrete computation actions. (iii) We introduce a two-stage training strategy combining PRM-based initialization and GRPO refinement.



(a) Trajectory-Aware ARC (b) Baselines vs. ARC

Figure 2: Overview of ARC. (a) ARC models the latent-state trajectory of a frozen reasoning LLM to dynamically select DEEP, NORMAL, COMPRESS, or CONCLUDE, and steers subsequent reasoning with lightweight in-context tags. (b) Compared with vanilla CoT, ARC suppresses redundant reasoning and allocates computation adaptively.

2 RELATED WORK

2.1 EFFICIENT AND ADAPTIVE REASONING

Recent studies show that increasing inference-time computation can substantially improve LLM reasoning. Chain-of-Thought prompting enables explicit intermediate reasoning steps (Wei et al., 2022), while self-consistency improves robustness through multiple sampled trajectories (Wang et al., 2023). Recent reasoning models further demonstrate the effectiveness of inference-time scaling on challenging tasks (Guo et al., 2025). However, longer reasoning trajectories often introduce redundant computation. Existing approaches explore reasoning compression and adaptive computation allocation, such as Chain-of-Draft for concise reasoning traces (Xu et al., 2025) and budget-aware reasoning strategies (Snell et al., 2025). Most existing methods rely on predefined heuristics, confidence thresholds, or local signals, without explicitly modeling the temporal evolution of reasoning trajectories. ARC instead learns computation policies from hidden-state trajectories, enabling trajectory-aware adaptive reasoning.

2.2 PROCESS REWARD MODELS

Process Reward Models (PRMs) provide step-level supervision by evaluating intermediate reasoning processes, enabling more precise credit assignment than outcome-based rewards (Lightman et al., 2024). Building on this idea, subsequent work has explored scalable and automatic process supervision that reduces reliance on costly human annotation (Wang et al., 2024). Despite their effectiveness as evaluators, however, PRMs are not designed as computation controllers: their scores measure the quality of the current reasoning step but do not directly determine the optimal computation action. ARC therefore uses PRM signals only for weak supervision and instead optimizes its computation policy through reinforcement learning based on downstream reasoning utility.

3 METHODOLOGY

3.1 PROBLEM FORMULATION

We formulate adaptive reasoning as a sequential computation control problem over the reasoning trajectory of a frozen large reasoning model (LRM). Given an input problem x, the frozen reasoner generates a chain-of-thought trajectory consisting of T reasoning steps:

$$\tau = (y_1, y_2, \ldots, y_T). \tag{1}$$

where each reasoning step y_t is a text segment delimited by a newline character "\n" in the generated CoT. Rather than directly controlling token generation, ARC determines the computation allocated after each completed reasoning step. At step t, the controller observes the historical reasoning trajectory and selects a computation action a_t based on the internal reasoning state s_t extracted from the frozen LRM; the selected action a_t then induces the next reasoning step y_{t+1}.

3.2 State and Action

The state of ARC is derived from the hidden representations of the frozen reasoner. Given a completed reasoning step y_t, we extract the final-token hidden representation:

$$h_t = H_\theta^{(L)}(x, y_{\le t})_{e_t}. \tag{2}$$

where L denotes the final transformer layer and e_t represents the final token position of step t. We use the final-layer representation as the default state representation, while further investigating the effect of hidden states from different transformer layers in Appendix C.2. Although h_t summarizes the previous context through causal attention, a single-step representation does not explicitly model the temporal evolution of reasoning. ARC therefore maintains the hidden-state history $H_t = (h_1, h_2, \ldots, h_t)$ as the controller state, and based on H_t selects one of four discrete computation actions: DEEP, NORMAL, COMPRESS, or CONCLUDE.

These actions respectively encourage additional reasoning for challenging steps, preserve standard reasoning behavior, reduce redundant computation for simple steps, or smoothly guide the model toward final completion.

3.3 Transition

After observing the current state $H_{i,t}^{(m)}$, ARC selects an action

$$a_{i,t} \sim \pi_\phi(\cdot | H_{i,t}^{(m)}). \tag{3}$$

which determines how the frozen reasoner proceeds to the next reasoning step. For DEEP, NORMAL, and COMPRESS, the selected action is converted into a lightweight control instruction $g(a_{i,t})$ and inserted into the context:

$$y_{i,t+1} \sim \pi_\theta^{(m)}(\cdot | x_i, y_{i,\le t}, g(a_{i,t})). \tag{4}$$

For CONCLUDE, ARC softly guides the reasoner to complete the trajectory rather than applying hard truncation. The generated step $y_{i,t+1}$ then determines the next hidden-state trajectory $H_{i,t+1}^{(m)}$, forming the closed loop $H_{i,t}^{(m)} \to a_{i,t} \to y_{i,t+1} \to H_{i,t+1}^{(m)}$. The detailed state-transition and tag-insertion procedure is provided in Algorithm 1.

3.4 Controller Training

ARC is trained in two stages: supervised initialization on offline reasoning trajectories, followed by on-policy reinforcement learning that optimizes the controller for downstream reasoning utility.

Hidden-Trajectory SFT Stage (HT-SFT). We first construct an offline reasoning corpus from PRM800K problems (Lightman et al., 2024). For each problem x_i, Qwen3-30B-A3B (Yang et al., 2025) generates a chain-of-thought trajectory, segmented into reasoning steps as defined in Section 3.1. We then use Skywork-o1-Open-PRM-7B (PRM-7B) (He et al., 2024) to evaluate each reasoning prefix:

$$r_{i,t} = V_{\mathrm{PRM}}(x_i, y_{i,\le t}), \qquad r_{i,t} \in [0, 1]. \tag{5}$$

Since PRM scores are bounded and individual evaluations can be noisy, we apply a logit transformation $s_{i,t} = \log \frac{\tilde{r}_{i,t}}{1-\tilde{r}_{i,t}}$, where $\tilde{r}_{i,t}$ denotes the clipped PRM score, and fit a four-state Gaussian hidden Markov model (HMM) on the transformed score trajectories, with latent state $z_{i,t} \in \{0, 1, 2, 3\}$ transitioning as $z_{i,t} \sim p(z_{i,t} \mid z_{i,t-1})$ and emitting $s_{i,t} \sim \mathcal{N}(\mu_{z_{i,t}}, \sigma^2_{z_{i,t}})$. We order the states by their emission means $\mu_0 < \mu_1 < \mu_2 < \mu_3$ and define boundaries using adjacent state midpoints in the transformed space:

$$c_k = \frac{\mu_{k-1} + \mu_k}{2}, \qquad b_k = \sigma(c_k), \tag{6}$$

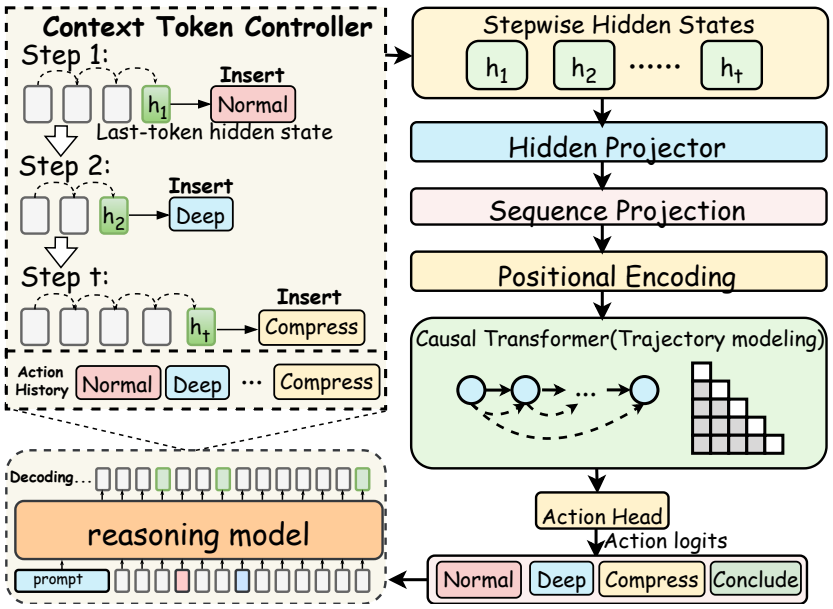


Figure 3: Architecture and Trajectory-Aware Pipeline of the Action Predictor.

where $\sigma(\cdot)$ maps the boundaries back to the original PRM-score space. The HMM thus provides a data-driven partition of the PRM score trajectory rather than semantic reasoning states.

To suppress short-term score fluctuations during pseudo-label construction, we apply a causal moving average

$$\bar{r}_{i,t}^{(w)} = \frac{1}{\min(w,t)} \sum_{j=\max(1,t-w+1)}^{t} r_{i,j} \tag{7}$$

and map the smoothed score to a pseudo-action:

$$\tilde{a}_{i,t} = q_{b,w}(r_{i,\le t}) = \begin{cases} \textsc{Deep}, & \bar{r}_{i,t}^{(w)} < b_1, \\ \textsc{Normal}, & b_1 \le \bar{r}_{i,t}^{(w)} < b_2, \\ \textsc{Compress}, & b_2 \le \bar{r}_{i,t}^{(w)} < b_3, \\ \textsc{Conclude}, & \bar{r}_{i,t}^{(w)} \ge b_3. \end{cases} \tag{8}$$

This mapping serves only as an initialization prior: PRM scores evaluate the current reasoning prefix, whereas optimal computation actions should depend on their downstream effect on future reasoning. The pseudo-actions are therefore not treated as ground-truth control decisions and are subsequently refined through on-policy optimization.

Since the pseudo-actions are constructed from shared Qwen3-30B-A3B trajectories while ARC operates on hidden-state histories of each target reasoner, we teacher-force the same trajectories through the target model $\pi_\theta^{(m)}$ and extract the corresponding hidden-state histories $H_{i,t}^{(m)}$ using Equation 2. For target model m, ARC is initialized by minimizing

$$\mathcal{L}_{\text{SFT}} = -\frac{1}{N} \sum_{i=1}^{N} \frac{1}{T_i} \sum_{t=1}^{T_i} \log \pi_\phi \left(\tilde{a}_{i,t} \mid H_{i,t}^{(m)} \right) \tag{9}$$

where normalization by T_i prevents long trajectories from dominating the objective. This stage provides a structured behavioral prior for the subsequent on-policy reinforcement learning stage, which adapts ARC to trajectories induced by the target reasoner and the controller's own actions.

Computation-Aware GRPO (CA-GRPO). We further refine ARC using on-policy grouped rollouts while keeping the target reasoner $\pi_\theta^{(m)}$ frozen. For an input problem x, the controller and

reasoner jointly induce a reasoning trajectory

$$\tau_i = \left\{ \left(H_{i,t}^{(m)}, a_{i,t}, y_{i,t+1} \right) \right\}_{t=1}^{T_i} \tag{10}$$

followed by a final answer. Here, $H_{i,t}^{(m)} = [h_{i,1}^{(m)}, \ldots, h_{i,t}^{(m)}]$ contains only the causally available hidden states up to step t.

Trajectory reward. We define the terminal reward as

$$R(\tau_i) = R_{\text{acc}}(\tau_i) + R_{\text{tok}}(\tau_i) + R_{\text{con}}(\tau_i) \tag{11}$$

where the correctness reward is

$$R_{\text{acc}}(\tau_i) = \begin{cases} +1, & \text{if the final answer is correct,} \\ -1, & \text{otherwise.} \end{cases} \tag{12}$$

To encourage efficient reasoning without rewarding short but incorrect solutions, the token-efficiency term is applied only to correct trajectories. For problem x, let b_x denote its adaptive token baseline and let L_i be the number of generated reasoning-step tokens in trajectory τ_i, excluding controller-tag tokens and final-answer tokens. When b_x is available,

$$R_{\text{tok}}(\tau_i) = \begin{cases} \text{clip}\left(-\lambda_{\text{tok}} \frac{L_i - b_x}{b_x}, -c_{\text{tok}}, c_{\text{tok}} \right), & \text{if } \tau_i \text{ is correct,} \\ 0, & \text{otherwise,} \end{cases} \tag{13}$$

where $\lambda_{\text{tok}} = 0.11$ and $c_{\text{tok}} = 0.3$. Thus, among correct solutions, trajectories shorter than the baseline receive a positive efficiency reward, whereas longer trajectories are penalized. Incorrect trajectories receive no token-efficiency reward.

We additionally penalize premature termination:

$$R_{\text{con}}(\tau_i) = \begin{cases} -\lambda_{\text{con}}, & \text{if the trajectory terminates with CONCLUDE and the final answer is incorrect,} \\ 0, & \text{otherwise,} \end{cases} \tag{14}$$

with $\lambda_{\text{con}} = 0.3$. Because correctness contributes ± 1 while the token term is bounded to $[-0.3, 0.3]$, reasoning efficiency cannot dominate final-answer correctness.

Adaptive per-problem token baseline. Rather than using a single manually chosen token budget for all problems, we maintain a separate baseline b_x for each problem. For a previously unseen problem, the baseline is initialized only if the current rollout group contains at least two correct trajectories. Let

$$\mathcal{L}_x^+ = \{L_i : \tau_i \text{ is correct}\}$$

, We initialize

$$b_x = \max\left(Q_{0.25}(\mathcal{L}_x^+), b_{\min}^{\text{auto}} \right). \tag{15}$$

where $Q_{0.25}$ denotes the 25th percentile. If fewer than two correct trajectories are available, no baseline is initialized and $R_{\text{tok}} = 0$ for that group.

The lower bound $b_{\min}^{\text{auto}}$ adapts to the empirical distribution of previously initialized problem baselines. Before sufficient statistics are accumulated, it is fixed to 512 tokens. After at least eight valid problem baselines are available, we use

$$b_{\min}^{\text{auto}} = \max\left(512, Q_{0.10}(\mathcal{B}) \right) \tag{16}$$

where $\mathcal{B}$ is the set of historical problem-specific baselines.

For an already initialized problem, the current baseline is first lower-bounded by $b_{\min}^{\text{auto}}$. After scoring the current group, the baseline is updated only when a shorter correct trajectory is observed. Let

$$L_{\min}^+ = \min \mathcal{L}_x^+$$

If $L_{\min}^+ < b_x$, we update

$$b_x \leftarrow \max\left((1-\alpha) b_x + \alpha L_{\min}^+, b_{\min}^{\text{auto}} \right) \tag{17}$$

with EMA coefficient $\alpha = 0.2$. This conservative update reduces sensitivity to isolated unusually short trajectories while allowing the efficiency target to improve gradually during training.

Group-relative advantage. For each problem, we sample a group of G controller-induced trajectories $\{\tau_i\}_{i=1}^{G}$ and normalize rewards only within the same problem:

$$\hat{A}_i = \frac{R(\tau_i) - \mu_R}{\sigma_R + \epsilon}, \qquad \mu_R = \frac{1}{G}\sum_{j=1}^{G} R(\tau_j) \tag{18}$$

where σ_R is the standard deviation of rewards within the group and $\epsilon = 10^{-6}$ is used for numerical stability. This group-relative normalization reduces the effect of differences in difficulty across problems.

Because the reasoner remains frozen, the trainable policy acts over a small discrete computation-action space rather than vocabulary tokens. For trajectory τ_i, the policy-gradient objective aggregates the log-probabilities of all controller decisions:

$$\mathcal{L}_{\mathrm{PG}} = -\frac{1}{G}\sum_{i=1}^{G} \hat{A}_i \sum_{t=1}^{T_i} \log \pi_\phi \left(a_{i,t} \mid H_{i,t}^{(m)}\right). \tag{19}$$

We regularize the updated controller toward the SFT-initialized reference policy π_{ref} and encourage exploration using an entropy bonus:

$$\mathcal{L} = \mathcal{L}_{\mathrm{PG}} + \beta_{\mathrm{KL}} D_{\mathrm{KL}}\left(\pi_\phi \| \pi_{\mathrm{ref}}\right) - \beta_{\mathrm{ent}} \mathcal{H}(\pi_\phi) \tag{20}$$

where $\beta_{\mathrm{KL}} = 0.03$ and $\beta_{\mathrm{ent}} = 0.01$. Only the controller parameters ϕ are optimized; gradients are not propagated into the frozen reasoner or through sampled text generation. This on-policy refinement exposes ARC to trajectories induced by its own control decisions, reducing the distribution shift between offline supervision and controlled inference while directly optimizing the accuracy–efficiency trade-off.

4 Experiments

4.1 Experimental Setup

Datasets and Metrics. We evaluate ARC on four reasoning benchmarks: AIME24 and AIME25 (Mathematical Association of America, a), AMC23 (Mathematical Association of America, b), and GPQA-Diamond (GPQA-D) (Rein et al., 2024). AIME24, AIME25, and AMC23 evaluate competition-level mathematical reasoning. GPQA-D consists of expert-level multiple-choice questions in biology, chemistry, and physics, providing an evaluation of reasoning ability beyond mathematics. We report **accuracy** and **average generated tokens per query**.

Backbone LRMs and Baselines. We evaluate ARC across diverse backbones spanning two model families and multiple parameter scales: the Qwen3 series (Yang et al., 2025) (4B and 8B) and the DeepSeek-R1-Distill-Qwen series (Guo et al., 2025) (1.5B and 7B). We compare ARC with three representative baselines: **CoT**, Chain-of-thought reasoning; **DEER** (Yang et al., 2026), which performs confidence-based dynamic early termination at potential reasoning transition points; and **DynaSor** (Fu et al., 2025), which periodically probes intermediate answers and terminates reasoning once consecutive probe answers become sufficiently stable.

Implementation Details. Unless otherwise specified, each controlled reasoning method is evaluated under the same decoding configuration, and token reduction is computed relative to the corresponding averaged vanilla CoT result. All experiments are conducted on NVIDIA H100 GPUs. For all methods, we use temperature 0.6, top-p $= 0.95$, and a maximum generation length of $32{,}768$ tokens.

4.2 Overall Performance

Table 1 shows that ARC(GRPO) achieves the strongest overall accuracy–efficiency trade-off across four backbones and four reasoning benchmarks. It reduces generated tokens in all 16 settings by 17.33%–55.30% (average 32.78%), while slightly improving mean accuracy over CoT by $+0.36$

Model	Method	AIME24		AMC		AIME25		GPQA	
		Acc (↑, %)	Tokens (↓)	Acc (↑, %)	Tokens (↓)	Acc (↑, %)	Tokens (↓)	Acc (↑, %)	Tokens (↓)
DeepSeek-R1-Distill-Qwen-1.5B	CoT	29.33	14313.43	72.50	8830.10	23.33	14363.91	30.84	14731.59
	DEER	20.00 (-9.33)	14542.50 (+1.60%)	67.50 (-5.00)	5941.68 (-32.71%)	**26.67** (+3.34)	12726.40 (-11.40%)	32.83 (+1.99)	10292.06 (-30.14%)
	DynaSor	**26.67** (-2.66)	**9229.70** (-35.52%)	**70.00** (-2.50)	**4538.12** (-48.61%)	23.33 (+0.00)	**9017.23** (-37.22%)	29.29 (-1.55)	**3015.57** (-79.53%)
	ARC	**26.67** (-2.66)	11163.83 (-22.00%)	67.50 (-5.00)	4804.30 (-45.59%)	23.33 (+0.00)	9031.53 (-37.12%)	**34.85** (+4.01)	6584.92 (-55.30%)
Qwen3-4B	CoT	66.67	13464.93	93.70	8089.00	60.00	17097.57	55.10	8611.00
	DEER	66.67 (+0.00)	13163.73 (-2.24%)	90.00 (-3.70)	7121.18 (-11.96%)	56.67 (-3.33)	14336.57 (-16.15%)	**56.57** (+1.47)	8004.87 (-7.04%)
	DynaSor	**73.33** (+6.66)	12658.97 (-5.99%)	90.00 (-3.70)	**5330.60** (-34.10%)	56.67 (-3.33)	14253.30 (-16.64%)	53.03 (-2.07)	**4894.07** (-43.16%)
	ARC	70.00 (+3.33)	**9684.80** (-28.07%)	**97.50** (+3.80)	5335.70 (-34.04%)	**63.33** (+3.33)	**12026.70** (-29.66%)	53.54 (-1.56)	4972.90 (-42.25%)
DeepSeek-R1-Distill-Qwen-7B	CoT	53.33	13712.60	82.50	5641.42	38.00	14439.21	44.95	7609.73
	DEER	46.67 (-6.66)	9839.00 (-28.25%)	**85.00** (+2.50)	4451.00 (-21.10%)	30.00 (-8.00)	13775.20 (-4.60%)	**48.48** (+3.53)	7442.42 (-2.20%)
	DynaSor	**50.00** (-3.33)	**7091.47** (-48.29%)	82.50 (+0.00)	**3189.82** (-43.46%)	30.00 (-8.00)	**9131.27** (-36.76%)	34.34 (-10.61)	**1437.55** (-81.11%)
	ARC	**50.00** (-3.33)	8836.87 (-35.56%)	82.50 (+0.00)	4594.91 (-18.55%)	**36.67** (-1.33)	9459.17 (-34.49%)	44.95 (+0.00)	4570.54 (-39.94%)
Qwen3-8B	CoT	70.00	15402.50	97.50	8089.15	60.00	16691.80	57.07	8813.39
	DEER	66.67 (-3.33)	11442.03 (-25.71%)	90.00 (-7.50)	6679.57 (-17.43%)	56.67 (-3.33)	14497.37 (-13.15%)	56.06 (-1.01)	9117.62 (+3.45%)
	DynaSor	70.00 (+0.00)	**10746.87** (-30.23%)	**95.00** (-2.50)	6293.48 (-22.20%)	53.33 (-6.67)	15101.93 (-9.52%)	**58.08** (+1.01)	**4084.84** (-53.65%)
	ARC	**76.67** (+6.67)	12335.70 (-19.91%)	**95.00** (-2.50)	**5808.12** (-28.20%)	**60.00** (+0.00)	**13799.87** (-17.33%)	**58.08** (+1.01)	5598.43 (-36.48%)

Table 1: Performance on reasoning benchmarks. Acc (↑) and Tokens (↓) are reported. CoT serves as the reference and is shaded in gray. Changes relative to CoT are shown in orange and blue, respectively. The best result among the non-CoT methods for each metric is **bolded**.

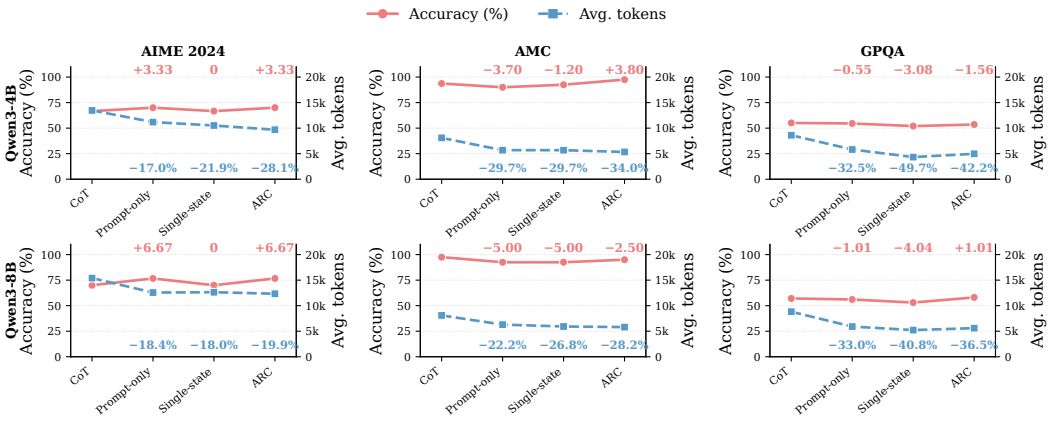


Figure 4: Ablation study of adaptive reasoning strategies on AIME24, AMC, and GPQA with Qwen3-4B and Qwen3-8B.

percentage points (pp). Compared with DEER and DynaSor, ARC(GRPO) better preserves accuracy while still providing substantial token savings. It also consistently improves over ARC(SFT), which reduces tokens by 25.94% on average but incurs a -3.14 pp accuracy drop. Overall, trajectory-aware on-policy optimization enables substantial reasoning compression with minimal loss in answer quality.

4.3 Ablation Study

We compare CoT, Prompt-only, a Single-state predictor, and ARC across two model scales (Qwen3-4B/8B) and three benchmarks (AIME24, AMC, GPQA). The Single-state predictor conditions only on the current hidden state and step index, $\pi(a_t \mid h_t, t)$, processing each decision independently without access to earlier hidden states; ARC instead conditions on the full hidden-state history available at the current step. As shown in Figure 4, Prompt-only reduces token usage substantially but has mixed, roughly neutral effects on accuracy (average -0.04 pp, -25.5% tokens across the six settings, relative to CoT). The Single-state predictor cuts tokens further (-31.2%) but consistently hurts accuracy (-2.22 pp on average). ARC achieves the best trade-off, reducing tokens by 31.5% on average while *improving* accuracy by 1.79 pp, matching or exceeding CoT accuracy in four of six settings. Relative to the Single-state predictor, ARC yields higher accuracy in all six settings (average gain 4.01 pp) and uses fewer tokens in four of six, with the remaining two trading a small

extra token cost for a sizable accuracy gain. These results indicate that explicitly modeling the hidden-state trajectory, rather than acting on the current state alone, is key to ARC's accuracy–efficiency advantage; further matched controls are needed to isolate the contribution of the temporal structure itself.

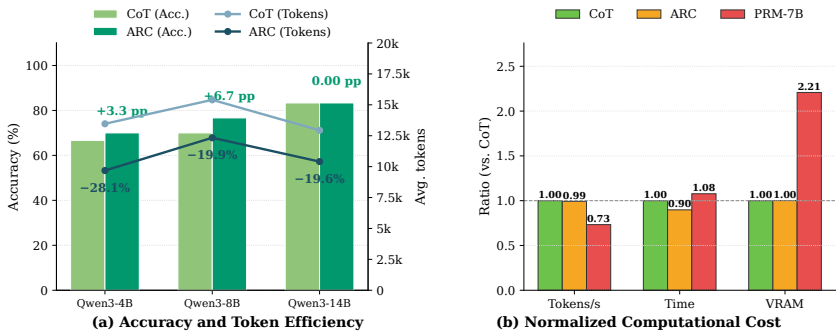


Figure 5: Efficiency analysis of ARC. (a) Accuracy and token usage comparison between CoT and ARC across different model scales on AIME24. (b) Normalized computational cost, including inference throughput, wall-clock time, and GPU memory usage, on Qwen3-14B with AIME24, using CoT as the baseline.

4.4 Generalization across Model Scales

We investigate the scalability of ARC across Qwen3 models of different sizes. As shown in Figure 5(a), ARC consistently reduces reasoning tokens while maintaining or improving accuracy across Qwen3-4B, Qwen3-8B, and Qwen3-14B on AIME24. Token usage drops by 28.1%, 19.9%, and 19.6% at the three scales, respectively, with accuracy changes of $+3.3$, $+6.7$, and $+0.0$ pp relative to CoT. These results indicate that the ARC framework remains effective across model scales rather than relying on a particular backbone size.

4.5 Additional Computation Cost

ARC introduces an additional controller during inference, so its efficiency gains must account for the cost of making step-level decisions. As shown in Figure 5(b), on Qwen3-14B with AIME24, ARC achieves $0.99\times$ CoT's token throughput, indicating a small overhead from the controller itself. Its inference time nevertheless drops to $0.90\times$ that of CoT, while GPU memory usage remains at $1.00\times$, so the reduction in generated reasoning outweighs the controller's runtime overhead in this setting. For comparison, PRM-7B attains only $0.73\times$ CoT's token throughput, requires $1.08\times$ the inference time, and consumes $2.21\times$ the GPU memory. These measurements show that ARC's controller has a substantially smaller resource footprint than the evaluated PRM-7B alternative; the comparison is specific to the reported Qwen3-14B and AIME24 configuration.

5 Conclusion

We study adaptive reasoning as trajectory-level computation control and show that hidden-state dynamics provide useful signals beyond instantaneous scores or handcrafted heuristics. Based on this insight, we propose ARC, a lightweight controller that operates on a frozen reasoner and selects computation actions from hidden-state histories. ARC is initialized with PRM-based pseudo-actions and further optimized with GRPO using answer correctness and reasoning efficiency, enabling adaptive computation allocation without modifying the underlying model. Across four reasoning models and four benchmarks, ARC reduces reasoning tokens by 32.8% on average (up to 55.3%) while preserving competitive accuracy, demonstrating the potential of trajectory-aware latent-state modeling for efficient reasoning.

AI USE STATEMENT

Large Language Models (LLMs) were used solely for language polishing during the preparation of this manuscript, including grammar checking, sentence restructuring, and the refinement of academic expressions in selected passages. Their use was limited to linguistic improvement and did not involve the generation or modification of substantive research content, including research ideas, methodological design, experimental procedures, data analysis, or conclusions. All core scientific contributions of this work were independently developed by the authors. All LLM-assisted revisions were carefully reviewed and verified, and the authors assume full responsibility for the final content of the manuscript.

ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics. We conducted the research with integrity, transparency, and respect for established ethical principles throughout all stages of the study.

- **Research integrity.** All results reported in this paper are presented accurately and faithfully. We do not fabricate, falsify, or intentionally misrepresent experimental results. The methods, evaluation protocols, and implementation details are described transparently to facilitate reproducibility.
- **Data usage.** All datasets used in this work were obtained and used in accordance with their respective licenses and terms of use. Where applicable, data handling follows relevant privacy and ethical requirements.
- **Attribution and authorship.** We appropriately acknowledge and cite prior work that contributed to the development of this study. All authors have made substantial contributions to the research and have approved the submission.
- **Broader impact.** Our work aims to improve the efficiency of reasoning in large language models by reducing unnecessary computation while preserving reasoning quality. As with other methods for controlling model behavior, the proposed approach may have broader applications beyond those studied here. We therefore encourage responsible evaluation and deployment, particularly in settings where model outputs may affect high-stakes decisions.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide detailed descriptions of the experimental setup, hyperparameters, and implementation details. Code and supplementary materials are made available where possible to support verification and future research.

REFERENCES

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 967–976, 2023.

Sicheng Feng, Gongfan Fang, Xinyin Ma, and Xinchao Wang. Efficient Reasoning Models: A Survey. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=sySqlxj8EB.

Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Yonghao Zhuang, Yian Ma, Aurick Qiao, Tajana Rosing, Ion Stoica, and Hao Zhang. Efficiently Serving LLM Reasoning Programs with Certaindex. In *Advances in Neural Information Processing Systems*, volume 38, pp. 157566–157604, 2025. doi: 10.52202/085713-4751. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/d037fd021c9aace128b8ce25001cdb6c-Paper-Conference.pdf.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng,

Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://www.nature.com/articles/s41586-025-09422-z.

Jujie He, Tianwen Wei, Rui Yan, Jiacai Liu, Chaojie Wang, Yimeng Gan, Shiwen Tu, Chris Yuhao Liu, Liang Zeng, Xiaokun Wang, Boyang Wang, Yongcong Li, Fuxiang Zhang, Jiacheng Xu, Bo An, Yang Liu, and Yahui Zhou. Skywork-o1 Open Series. Zenodo, 2024. URL https://doi.org/10.5281/zenodo.16998085. Version 1.0.0.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large Language Models are Zero-Shot Reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22199–22213, 2022. doi: 10.52202/068431-1613. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

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 *The Twelfth International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/aca97732e30bcf1303bc22ac3924fd16-Abstract-Conference.html.

Mathematical Association of America. MAA Invitational Competitions. https://maa.org/maa-invitational-competitions/, a. Accessed: 2026-09-22.

Mathematical Association of America. AMC Registration + Information. https://maa.org/amcreg/, b. Accessed: 2026-09-22.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A Graduate-Level Google-Proof Q&A Benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM Test-Time Compute Optimally Can be More Effective than Scaling Parameters for Reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1b623663fd9b874366f3ce019fdfdd44-Abstract-Conference.html.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, Hanjie Chen, and Xia Hu. Stop Overthinking: A Survey on Efficient Reasoning for Large Language Models. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=HvoG8SxggZ.

Renliang Sun, Wei Cheng, Dawei Li, Haifeng Chen, and Wei Wang. Stop When Enough: Adaptive Early-Stopping for Chain-of-Thought Reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27250–27268. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1256. URL https://aclanthology.org/2026.acl-long.1256/.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, 2024.

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 *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Yiming Wang, Pei Zhang, Baosong Yang, Derek F. Wong, and Rui Wang. Latent space chain-of-embedding enables output-free llm self-evaluation. In *International Conference on Learning Representations (ICLR)*, 2025.

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 *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022. doi: 10.52202/068431-1800. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Yu Xia, Rui Wang, Xu Liu, Mingyan Li, Tong Yu, Xiang Chen, Julian McAuley, and Shuai Li. Beyond Chain-of-Thought: A Survey of Chain-of-X Paradigms for LLMs. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 10795–10809. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.coling-main.719/.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of Draft: Thinking Faster by Writing Less. *arXiv preprint arXiv:2502.18600*, 2025. URL https://arxiv.org/abs/2502.18600.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 Technical Report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. Dynamic Early Exit in Reasoning Models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=NpU7ZXafRi.

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 (NeurIPS)*, volume 36, 2023.

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. In *Second Conference on Language Modeling*, 2025a. URL https://openreview.net/forum?id=O6I0Av7683.

Zhaoshi Zhang, Xiaomeng Hu, Haiquan Zhang, Jiaming Zhang, and Xiaojun Wan. Icr probe: Tracking hidden state dynamics for reliable hallucination detection in LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17986–18002, 2025b.

A Control Tags and System Prompts

A.1 Control Tags

To enable dynamic control over reasoning depth, we introduce a set of explicit control tags that are inserted into the model context based on the action selected by the ARC controller. These tags are predefined in the system prompt and function as high-level instructions that regulate the model's reasoning behavior through instruction following.

The action space of the ARC controller is defined as:

$$\mathcal{A} = \{\textsc{Deep}, \textsc{Normal}, \textsc{Compress}, \textsc{Conclude}\}. \tag{21}$$

Each action is associated with a corresponding control tag that guides the model to adopt a specific reasoning strategy.

- Deep: Encourages the model to allocate additional reasoning effort to the current step by performing more detailed exploration and verification.
- Normal: Maintains the model's default reasoning behavior and preserves the standard step-by-step reasoning process.
- Compress: Reduces unnecessary reasoning expansion by encouraging the model to summarize intermediate results and avoid redundant computation when the current step is relatively straightforward.
- Conclude: Indicates that sufficient reasoning progress has been achieved and guides the model to consolidate existing information and move toward final answer generation.

By dynamically inserting these control tags according to the ARC controller's predicted states, the model can adaptively adjust its reasoning depth throughout the generation process, enabling more efficient allocation of computation while maintaining solution quality.

A.2 System Prompt

We provide the complete system prompt used for model initialization and behavior specification below. During inference, this prompt is prepended to the user query to guide the model's reasoning behavior.

System Prompt

Please reason step by step, and put your final answer within \boxed{}.

During your thinking, you may see the following tags after reasoning steps:

[Deep_Step]: The last reasoning step is important and should be preserved. Continue with careful, detailed step-by-step reasoning.

[Normal_Step]: The last reasoning step is useful but not critical. Continue with concise normal reasoning.

[Compress_Step]: The last reasoning step is likely redundant or repetitive. Avoid repeating it; make the next step brief and only add new useful information.

[Conclude_Step]: The solution is likely determined. Stop further exploration, close the thinking phase, and give the final answer in \boxed{}.

B Implementation Details

All experiments are conducted on 8 NVIDIA H100 GPUs.

B.1 SFT Initialization and HMM Diagnostics

B.1.1 HMM-calibrated Pseudo-action Construction

We use the HMM only to obtain a data-driven discretization of the PRM trajectory for SFT initialization; its latent states are not interpreted as semantic reasoning stages or optimal computation actions.

Because PRM scores are bounded in $[0, 1]$, we first clip the scores to $[10^{-4}, 1 - 10^{-4}]$ and apply the logit transformation

$$s_{i,t} = \log \frac{\tilde{r}_{i,t}}{1 - \tilde{r}_{i,t}}. \tag{22}$$

We then fit a one-dimensional Gaussian HMM to the transformed score trajectories. The latent states are ordered according to their emission means,

$$\mu_0 < \mu_1 < \mu_2 < \mu_3.$$

The three decision boundaries are defined by the midpoints between adjacent emission means in logit space and mapped back to the original PRM-score space:

$$c_k = \frac{\mu_{k-1} + \mu_k}{2}, \qquad b_k = \sigma(c_k), \qquad k = 1, 2, 3. \tag{23}$$

This construction yields unique, ordered, and fully data-driven thresholds without requiring manually selected score cutoffs.

B.1.2 Temporal-structure Validation

We further examine whether the HMM captures genuine temporal structure rather than merely partitioning the marginal distribution of PRM scores. Problems are split into disjoint training, validation, and test sets before model fitting. We evaluate HMMs with $K \in \{2, 3, 4\}$ and compare them against independent Gaussian mixture models with the same number of components.

We additionally perform two temporal-order controls. First, we shuffle PRM scores within each trajectory, preserving its score multiset and length while destroying temporal order. Second, we perform a position-controlled shuffle that preserves score distributions within relative trajectory-position bins while disrupting local adjacency. As shown in Figure 6, the original trajectories obtain higher held-out likelihood than both the independent GMM and the two shuffled controls, indicating that PRM trajectories contain temporal dependencies beyond their marginal score distribution and global trajectory position.

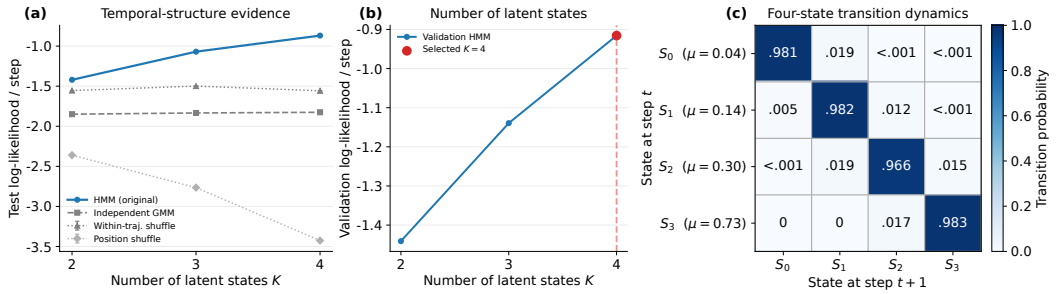


Figure 6: Diagnostics of the HMM-based latent computation-state discovery used for constructing SFT pseudo-actions. **(a)** Held-out evaluation of temporal structure by comparing the four-state HMM likelihood on original PRM trajectories with an independent GMM baseline and two shuffle controls. **(b)** Validation log-likelihood across candidate latent-state numbers ($K = 2, 3, 4$), motivating the use of a four-state representation. **(c)** Transition matrix of the four-state HMM, where states are ordered by increasing PRM emission mean. These diagnostics demonstrate that PRM-score trajectories contain persistent temporal structure beyond marginal score distributions and trajectory-position effects.

The four-state model further exhibits strong diagonal transition probabilities and multi-step expected dwell times, showing that the inferred regimes are temporally persistent rather than independent step-wise score bins. These results support the use of the HMM as a temporal calibration mechanism for constructing the initial SFT pseudo-actions.

B.2 GRPO Training

B.2.1 Dataset Preparation

We further optimize the ARC controller using Group Relative Policy Optimization (GRPO), while keeping the underlying reasoner frozen. The GRPO training set is constructed from the OpenR1-Math corpus, from which we sample 1,000 mathematical problems with expert reasoning traces ranging from 512 to 8,192 tokens. These samples cover diverse reasoning difficulties and provide sufficiently long trajectories for learning fine-grained reasoning control.

B.2.2 Asynchronous Rollout Execution

Step-wise reasoning control requires repeated interaction between the controller and the frozen reasoner, which can introduce substantial latency under sequential execution. We therefore implement rollout collection asynchronously. The frozen reasoner is served with SGLang, while controller inference and optimization are handled by a separate Ray actor. Multiple problem groups and trajectories are executed concurrently through asynchronous requests, allowing controller decisions for some trajectories to proceed while others are waiting for reasoner generation. This design amortizes the overhead of fine-grained controller–reasoner interaction without modifying the frozen reasoner.

B.2.3 Training Configuration

Table 2 summarizes the controller architecture, backbone-dependent input dimensions, and GRPO training configuration. All predictors share the same architecture; only the input hidden-state dimension varies with the corresponding frozen reasoner.

B.3 State-Transition Inference

ARC uses a lightweight state-transition mechanism during inference. Unlike methods that rely on a predefined multi-step smoothing window, we set all persistence windows to 1, so each newly predicted state takes effect immediately without additional hysteresis. To avoid redundant context

Configuration	Value	
Shared predictor architecture		
Hidden projection size	256	
Controller hidden size	256	
Transformer layers	2	
Attention heads	8	
MLP hidden size	256	
Backbone-dependent configuration		
Qwen3-4B	Hidden dim 2560;	2.37M
Qwen3-8B	Hidden dim 4096;	2.76M
DeepSeek-R1-Distill-Qwen-1.5B	Hidden dim 1536;	2.11M
DeepSeek-R1-Distill-Qwen-7B	Hidden dim 3584;	2.63M
GRPO optimization		
GRPO group size G	8 trajectories	
Problems per update	2	
Trajectories per update	16	
Learning rate	5×10^{-5}	
Optimizer	AdamW	
Gradient clipping	1.0	
Controller sampling temperature	1.0	
Reasoner rollout temperature	0.0	

Table 2: Controller architecture and GRPO training configuration. All predictors use a 256-dimensional hidden projection and controller representation, two Transformer layers, eight attention heads, and a 256-dimensional MLP. Only the input hidden-state dimension varies across frozen reasoners, resulting in predictors with 2.11M–2.76M trainable parameters. The target reasoner remains frozen throughout optimization.

Algorithm 1 ARC State-Transition Inference

Input: frozen reasoner f_θ, controller π_ϕ
Initialize: $t \leftarrow 1$, $s_0 \leftarrow$ NORMAL, $\mathcal{H}_0 \leftarrow \emptyset$
while the final answer has not been generated **do**
  // 1. Reasoning Step
  $y_t \leftarrow f_\theta(\text{context})$
  Extract final-token hidden state h_t
  $\mathcal{H}_t \leftarrow (\mathcal{H}_{t-1}, h_t)$
  // 2. Action Prediction
  $\hat{s}_t \leftarrow \arg\max_s \pi_\phi(s \mid \mathcal{H}_t)$
  // 3. State Transition
  $s_t \leftarrow \hat{s}_t$ {persistence window $= 1$}
  // 4. Event-Triggered Steering
  if $s_t \neq s_{t-1}$ **then**
    Insert tag $g(s_t)$ into the context
  end if
  $s_{t-1} \leftarrow s_t$
  $t \leftarrow t + 1$
end while

injection, a control tag is inserted only when the effective state changes; otherwise, generation continues without adding a new tag. Algorithm 1 summarizes the complete procedure.

B.4 BASELINE REPRODUCTION

We compare ARC against three reasoning baselines: **CoT**, standard chain-of-thought reasoning; **DEER** (Yang et al., 2026), which performs confidence-based dynamic early termination at potential reasoning transition points; and **DynaSor** (Fu et al., 2025), which periodically probes intermediate answers and terminates reasoning once consecutive probe answers become sufficiently stable.

		AIME24		AMC		AIME25		GPQA	
Backbone	**Method**	**Acc.**	**Tokens**	**Acc.**	**Tokens**	**Acc.**	**Tokens**	**Acc.**	**Tokens**
DeepSeek-R1-Distill-Qwen-1.5B	ARC-SFT	26.67	11,756.20	60.00	6,790.77	23.33	10,770.13	31.31	6,977.36
	ARC-GRPO	**26.67**	**11,163.83**	**67.50**	**4,804.30**	**23.33**	**9,031.53**	**34.85**	**6,584.92**
Qwen3-4B	ARC-SFT	66.67	11,683.50	90.00	5,480.52	**66.67**	12,396.40	50.00	**4,919.09**
	ARC-GRPO	**70.00**	**9,684.80**	**97.50**	**5,335.70**	63.33	**12,026.70**	**53.54**	4,972.90
DeepSeek-R1-Distill-Qwen-7B	ARC-SFT	50.00	9,059.70	80.00	5,097.02	33.33	9,568.27	42.93	5,234.74
	ARC-GRPO	**50.00**	**8,836.87**	**82.50**	**4,594.91**	**36.67**	**9,459.17**	**44.95**	**4,570.54**
Qwen3-8B	ARC-SFT	70.00	13,429.77	90.00	6,335.15	46.67	15,314.47	57.07	6,228.62
	ARC-GRPO	**76.67**	**12,335.70**	**95.00**	**5,808.12**	**60.00**	**13,799.87**	**58.08**	**5,598.43**

Table 3: Comparison between the SFT-initialized controller and the GRPO-refined ARC.

For DEER and DynaSor, we directly reproduce the methods using the official code released by the corresponding authors, without modifying their core control or termination mechanisms. To ensure a fair comparison, all methods are evaluated using the same underlying reasoner, base problem prompt, decoding configuration, and maximum generation budget. Specifically, unless otherwise stated, we use temperature 0.6, top-$p = 0.95$, and a maximum generation length of $32{,}768$ tokens. Thus, differences in accuracy and token consumption primarily reflect the computation-control strategy rather than differences in decoding settings.

For the vanilla CoT baseline, we use the following prompt template:

System Prompt

```
Please reason step by step, and put your final answer within
\boxed{}.

Problem:
{question}
```

C Additional Experimental Analysis

C.1 SFT vs. GRPO Improvement

To isolate the effect of on-policy refinement, we compare the SFT-initialized controller with the corresponding GRPO-refined ARC under identical inference configurations. Table 3 shows that GRPO consistently improves the accuracy–efficiency trade-off across model families and scales. Among the 16 model–benchmark settings, GRPO improves accuracy in 12, matches SFT in 3, and decreases accuracy in only 1 setting. Meanwhile, it reduces token usage in 15 of the 16 settings. Averaged across all settings, GRPO improves accuracy by 3.50 percentage points while further reducing token usage by 8.77% relative to the SFT controller. These results indicate that on-policy optimization does not merely preserve the PRM-derived initialization policy, but substantially adapts the controller toward downstream reasoning utility.

C.2 Cross-layer Hidden States for ARC

ARC derives its controller state from the hidden representation of a designated transformer layer at each reasoning step. Although we use the final-layer representation by default, hidden states at different depths may encode different aspects of the reasoning process. Earlier and intermediate layers may preserve more local or lexical information, whereas deeper layers are expected to provide increasingly task-conditioned representations after repeated contextual transformation. To examine the sensitivity of ARC to this design choice, we compare controllers using hidden states extracted from layer 12, layer 24, and the final layer (layer 36) on Qwen3-4B and Qwen3-8B. All variants

Table 4: Comparison of controller hidden-state layer choice (layer 12, layer 24, final layer) on Qwen3-4B and Qwen3-8B across four benchmarks. Bold indicates the best within each model group.

Model	Layer	AIME24		AMC		AIME25		GPQA	
		Acc.↑	Tokens↓	Acc.↑	Tokens↓	Acc.↑	Tokens↓	Acc.↑	Tokens↓
Qwen3-4B	Layer 12	**70.00**	12217.20	92.50	**4984.48**	56.67	12977.30	**53.54**	**4621.66**
	Layer 24	66.67	11129.00	95.00	5626.75	60.00	12874.20	52.53	4683.27
	Layer 36	**70.00**	**9684.80**	**97.50**	5335.70	**63.33**	**12026.70**	**53.54**	4972.90
Qwen3-8B	Layer 12	70.00	**11373.47**	**95.00**	6375.93	**60.00**	14580.50	52.53	**5536.83**
	Layer 24	66.67	11859.73	**95.00**	6650.57	**60.00**	14960.80	57.57	5564.99
	Layer 36	**76.67**	12335.70	**95.00**	**5808.12**	**60.00**	**13799.87**	**58.08**	5598.43

use the same controller architecture and evaluation protocol, and are evaluated on AIME24, AMC, AIME25, and GPQA.

As shown in Table 4, hidden states from all three depths provide usable signals for computation control, indicating that information related to reasoning progress is not confined to a single layer. However, the final-layer representation yields the most stable accuracy–efficiency trade-off across model scales and benchmarks. For Qwen3-4B, layer 36 achieves the highest accuracy on AMC (97.50%) and AIME25 (63.33%), while matching the best-performing layer on AIME24 (70.00%) and GPQA (53.54%). It also remains competitive in token usage across the four benchmarks. For Qwen3-8B, the same trend remains broadly consistent: layer 36 obtains the highest accuracy on AIME24 (76.67%) and GPQA (58.08%), while remaining competitive on AMC and AIME25. Although layer 12 is more token-efficient for several Qwen3-8B settings, this efficiency is not accompanied by uniformly better accuracy, suggesting a stronger accuracy–computation trade-off at shallower depths.

The intermediate representation from layer 24 is comparatively less stable. In particular, it gives the lowest AIME24 accuracy for both model scales (66.67%), despite using a similar or larger amount of computation in several settings. This suggests that intermediate-layer representations may capture only partially consolidated reasoning signals, making them less reliable for determining high-level computation actions.

Overall, the results indicate that ARC is not critically dependent on a single specific layer, but deeper hidden representations provide a more robust control signal across tasks and model sizes. We therefore use the final-layer hidden state as the default controller input in the main experiments. This choice also keeps the controller interface simple: ARC operates directly on the representation produced at the end of the frozen reasoner's forward computation, without introducing additional layer-selection mechanisms.

D Case Studies

We further present two representative GPQA cases from Qwen3-8B to illustrate how ARC changes the reasoning trajectory in practice. For each case, we compare an independent vanilla CoT rollout with an ARC-controlled rollout under the same decoding configuration. We show selected reasoning excerpts for readability, while the reported final answers correspond to the complete runs. Controller-inserted action tags are placed at the positions where they affect subsequent generation; when shown in gray, a CONCLUDE tag is generated by the reasoner itself rather than inserted by the controller. The two examples highlight complementary behaviors of ARC. In the first case, both CoT and ARC reach the correct answer, but vanilla CoT repeatedly re-verifies an already established intermediate result. ARC instead preserves and reuses the computed quantity, leading to a more direct reasoning path with less redundant computation. In the second case, vanilla CoT fails to account for relativistic time dilation and ultimately selects an incorrect option. The ARC-controlled trajectory identifies the missing correction, after which a DEEP action reinforces this direction and preserves the corrected reasoning until the final answer. These examples are intended only as qualitative illustrations of ARC's behavior, while the aggregate effects are evaluated quantitatively in the main experiments and ablation study.

D.1 QWEN3-8B / GPQA

Prompt.

A spin-half particle is in a linear superposition $0.8\,|\uparrow\rangle + 0.6\,|\downarrow\rangle$ of its spin-up and spin-down states. If $|\uparrow\rangle$ and $|\downarrow\rangle$ are the eigenstates of σ_z, then what is the expectation value, up to one decimal place, of the operator $10\sigma_z + 5\sigma_x$? Here, symbols have their usual meanings.

A. 7.6 B. 7.2 C. 8.6 D. 9.1

Please reason step-by-step and put your choice letter without any other text with `\boxed{}` at the end.

Reference answer: A

CoT reasoning excerpts

[...] $\psi = [0.8,\ 0.6]^T$. [...]
Now, σ_z is the Pauli matrix for the z-component, which is $\mathrm{diag}(1, -1)$. So $\sigma_z = \left(\begin{smallmatrix} 1 & 0 \\ 0 & -1 \end{smallmatrix}\right)$. Similarly, $\sigma_x = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$.
So, first compute $\langle\sigma_z\rangle$. Let's compute $\langle\psi|\sigma_z|\psi\rangle$.
[...] $[1{\cdot}0.8 + 0{\cdot}0.6,\ 0{\cdot}0.8 + (-1){\cdot}0.6]^T = [0.8, -0.6]^T$.
Then $\langle\psi|\sigma_z|\psi\rangle$ is the dot product of $[0.8, 0.6]$ and $[0.8, -0.6]$: $0.8{\cdot}0.8 + 0.6{\cdot}(-0.6) = 0.64 - 0.36 = 0.28$.
So $\langle\sigma_z\rangle$ is 0.28. Multiplying by 10 gives $10{\cdot}0.28 = 2.8$.
Now, moving on to σ_x. Let's compute $\langle\sigma_x\rangle$. Similarly, $\sigma_x|\psi\rangle = [0{\cdot}0.8{+}1{\cdot}0.6,\ 1{\cdot}0.8{+}0{\cdot}0.6]^T = [0.6, 0.8]^T$.
Then $\langle\psi|\sigma_x|\psi\rangle$ is the dot product of $[0.8, 0.6]$ and $[0.6, 0.8]$: $0.8{\cdot}0.6{+}0.6{\cdot}0.8 = 0.48{+}0.48 = 0.96$.
Therefore $\langle\sigma_x\rangle$ is 0.96. Multiplying by 5 gives $5{\cdot}0.96 = 4.8$.
Adding the two parts: $2.8 + 4.8 = 7.6$. So the expectation value is 7.6, option A.
Wait, let me check my calculations again. [...] So $10{\cdot}0.28$ is 2.8, $5{\cdot}0.96$ is 4.8. Adding gives 7.6. So the answer is A, 7.6.
Let me check another way: compute $10\sigma_z + 5\sigma_x$ as a matrix and multiply by the state vector, then take the inner product. [...]
$\left(\begin{smallmatrix} 10{+}0 & 0{+}5 \\ 0{+}5 & -10{+}0 \end{smallmatrix}\right) = \left(\begin{smallmatrix} 10 & 5 \\ 5 & -10 \end{smallmatrix}\right)$. [...]
First component: $10{\cdot}0.8 + 5{\cdot}0.6 = 8 + 3 = 11$.
Second component: $5{\cdot}0.8{+}(-10){\cdot}0.6 = 4{-}6 = -2$. So the resulting vector is $[11, -2]^T$.
Then $\langle\psi|$ is $[0.8, 0.6]$, so the inner product is $0.8{\cdot}11 + 0.6{\cdot}(-2) = 8.8 - 1.2 = 7.6$. Same result, confirmed.
Therefore, the answer is 7.6, option A. [...]

FINAL ANSWER A / Correct

ARC reasoning + action tags

[...] $O = 10\sigma_z + 5\sigma_x = \left(\begin{smallmatrix} 10 & 5 \\ 5 & -10 \end{smallmatrix}\right)$.
The expectation value $\langle O\rangle$ is $\langle\psi|O|\psi\rangle$. The state vector $|\psi\rangle$ is $[0.8, 0.6]^T$; since the coefficients are real, the bra $\langle\psi|$ is also $[0.8, 0.6]$.
So, first compute $O|\psi\rangle$: with $O = \left(\begin{smallmatrix} 10 & 5 \\ 5 & -10 \end{smallmatrix}\right)$, first component $10{\cdot}0.8 + 5{\cdot}0.6 = 8 + 3 = 11$; second component $5{\cdot}0.8 + (-10){\cdot}0.6 = 4 - 6 = -2$.

`[Deep_Step]` So, $O|\psi\rangle = [11, -2]^T$. Then $\langle\psi|O|\psi\rangle$ is the inner product of $\langle\psi|$ and $O|\psi\rangle$: $0.8{\cdot}11 + 0.6{\cdot}(-2) = 8.8 - 1.2 = 7.6$. That's option A. But wait, let me check if I did all steps correctly.

`[Normal_Step]` Wait, let me confirm the matrix multiplication again. First component $10{\cdot}0.8 + 5{\cdot}0.6 = 8 + 3 = 11$; second component $5{\cdot}0.8 + (-10){\cdot}0.6 = 4{-}6 = -2$. Correct. Inner product $0.8{\cdot}11 + 0.6{\cdot}(-2) = 8.8 - 1.2 = 7.6$. So yes, 7.6 – answer A. But let me check another way, using $\langle\sigma_z\rangle$ and $\langle\sigma_x\rangle$ separately. [...]

`[Conclude_Step]` The expectation value is 7.6, so the answer is $\boxed{A}$. [...]

FINAL ANSWER A / Correct

After `[Deep_Step]`, ARC reuses the already-computed vector in the inner product rather than recomputing it; the check after `[Normal_Step]` confirms the same result a second way.

D.2 QWEN3-8B / GPQA

Prompt.

X is a meson resonance. What is the mean decay distance? Knowing that the production energy is $E_X = 8\,\text{GeV}$, the mass $m_X = 1.2\,\text{GeV}$, and the width $\Gamma_X = 320\,\text{MeV}$.

A. 4.0655×10^{-15} m B. 4.0655×10^{-16} m C. 5.0223×10^{-15} m D. 5.0223×10^{-16} m

Please reason step-by-step and put your choice letter without any other text with `\boxed{}` at the end.

Reference answer: A

CoT reasoning excerpts

[...] Since Γ is in MeV and $\hbar c$ is in MeV·fm, $\tau = \hbar/\Gamma$. Recall $\hbar \approx 6.582 \times 10^{-25}\,\text{GeV} \cdot \text{s}$. With $\Gamma = 0.32$ GeV, $\tau = (6.582 \times 10^{-25})/0.32 \approx 2.057 \times 10^{-25}$ s. [...]
$E = 8$ GeV, $m = 1.2\,\text{GeV}/c^2$, so $(pc)^2 = 8^2 - 1.2^2 = 64 - 1.44 = 62.56\,(\text{GeV})^2$, so $pc \approx 7.91$ GeV, $p = 7.91\,\text{GeV}/c$.
Momentum relates to velocity by $p = \gamma m v$, with $\gamma = E/(mc^2) = 8/1.2 \approx 6.6667$. So $v = p/(\gamma m)$. [...] Denominator $6.6667 \times 1.2 = 8$, so $v = 7.91/8\,c \approx 0.98875\,c$ – very close to the speed of light.
Therefore the mean decay distance is $d = v\tau$. [...] $0.98875 \times 3 \approx 2.96625$, then $2.96625 \times 2.057 \approx 6.101$, so $d \approx 6.1 \times 10^{-17}$ m. But the options are 10^{-15} or 10^{-16} – this seems far too small. Maybe there's a mistake. [...]
Alternatively, using $\hbar c = 197.3$ MeV·fm: $d = (\hbar c/\Gamma)\,(v/c) = (197.3/320) \times 0.9887$ fm $= 0.6166 \times 0.9887 \approx 0.609$ fm $= 6.09 \times 10^{-16}$ m – close to 6.1×10^{-16}, which is not among the options, but the closest listed options are B and D. [...]
At this point I think there is a mistake I can't find. Given that, and the closest option to my calculation, I will choose D. [...]

FINAL ANSWER D / Incorrect

ARC reasoning + action tags

[...] Convert Γ to GeV: 320 MeV $= 0.32$ GeV, so $\tau = 6.582 \times 10^{-25}/0.32$ GeV·s. Dividing 6.582 by 0.32 gives roughly 20.56875, so $\tau \approx 20.56875 \times 10^{-25}$ s. [...]
Earlier, $v \approx 0.9887\,c$. So $d = 0.9887\,c\,\tau$: $3 \times 10^8 \times 2.056875 \times 10^{-24} \approx 6.17 \times 10^{-16}$ m; times $0.9887 \approx 6.10 \times 10^{-16}$ m. Still not matching the options. Maybe the approach is wrong – let me think again. [...]
Wait: the mean decay distance is the product of velocity and lifetime, but since the particle moves at relativistic speed, is the lab-frame lifetime $\gamma\tau$ rather than τ? The lifetime in the lab frame is $\gamma\tau$ because of time dilation. So the distance travelled before decay may be $d = v\,\gamma\,\tau$. Let me clarify.
`[Deep_Step]` The mean decay distance is $d = \beta\gamma c\tau$, where $\beta = v/c$ and γ is the Lorentz factor – the proper lifetime is τ, and in the lab frame the lifetime is $\gamma\tau$, so the distance travelled is $d = v\,\gamma\,\tau = \beta c\,\gamma\,\tau$. With $\gamma = E/(mc^2) \approx 6.6667$ and $\beta = \sqrt{1 - 1/\gamma^2} \approx 0.9887$, $\beta\gamma \approx 0.9887 \times 6.6667 \approx 6.59$. And $c\tau \approx 3 \times 10^8 \times 2.056875 \times 10^{-24} \approx 6.17 \times 10^{-16}$ m. Multiplying, $6.59 \times 6.17 \times 10^{-16} \approx 4.06 \times 10^{-15}$ m – that matches one of the options. Let me check that again. [...] Therefore, the correct answer should be A. [...]

FINAL ANSWER A / Correct

The time-dilation correction is proposed just before `[Deep_Step]`*; the reasoning that follows retains it and arrives at the option matching the reference answer.*