REASONING JUST ENOUGH: LEARNING LATENT SUFFICIENCY BOUNDARIES FOR ADAPTIVE EXIT

Anonymous authors
Paper under double-blind review

ABSTRACT

Large reasoning models often continue generating after sufficient information for a correct answer has already emerged, increasing inference cost and sometimes degrading accuracy. We introduce **SPARE** (**S**ufficiency **P**robing for **A**daptive **R**easoning **E**xit), a dynamic early-exit framework that detects a sample-dependent latent sufficiency boundary. SPARE derives anchor-relative supervision from verified answer-related expressions and trains a lightweight linear probe on multi-layer hidden states to identify answer-sufficient reasoning prefixes. GraphRACE then performs compute-efficient offline calibration through shared stopping-event evaluation, behavioral consolidation, and risk-aware Pareto selection, yielding a frozen persistence-based stopping policy. At inference time, SPARE requires only the trained probe and calibrated policy, without correctness feedback, online search, or backbone modification. Across multiple reasoning models and mathematical and scientific benchmarks, SPARE largely preserves accuracy and often improves it—by more than 15 percentage points in the strongest case—while reducing net token usage by approximately 30%–50%. These results show that latent sufficiency signals can support reasoning that is not maximal, but just enough.

1 INTRODUCTION

Large reasoning models achieve strong performance on mathematics, science, and other reasoning-intensive tasks by generating extended chains of thought (Wei et al., 2022; DeepSeek-AI, 2025; Snell et al., 2024). This additional inference-time computation allows models to explore alternative solution paths, revise intermediate conclusions, and recover from errors. However, longer reasoning is not always more useful. Models often continue generating after sufficient information for a correct answer has already emerged, producing redundant verification, repeated derivations, or unnecessary self-correction. Such *overthinking* increases inference cost and latency and may even overturn an initially correct solution (Chen et al., 2024; Cuadron et al., 2025; Sui et al., 2025).

Existing approaches address overthinking through post-training, budget allocation, or inference-time stopping. Training- and budget-based methods encourage shorter reasoning traces or allocate computation in advance (Ma et al., 2025; Fang et al., 2026; Han et al., 2025b), while inference-time methods monitor answer convergence, linguistic cues, entropy, semantic redundancy, or trial-answer confidence (Yang et al., 2025; Sun et al., 2026; Laaouach, 2025; Fu et al., 2024). Recent studies further suggest that intermediate hidden states encode information about reasoning correctness and completion (Akgül et al., 2025; Liu et al., 2026; Zhang et al., 2025). Nevertheless, three questions remain unresolved: whether answer sufficiency exhibits a consistent transition along individual reasoning trajectories, whether this transition can be learned without exhaustively evaluating intermediate exits, and how an effective stopping policy can be calibrated efficiently while limiting premature-exit risk.

We introduce **SPARE** (**S**ufficiency **P**robing for **A**daptive **R**easoning **E**xit), a hidden-state-based framework that detects when accumulated reasoning becomes sufficient for a correct answer. SPARE models this transition as a sample-dependent latent sufficiency boundary and learns it from completed trajectories through verified answer-related anchors. These anchors provide multiple anchor-relative labels for training a lightweight linear probe that predicts whether the current reasoning prefix is answer-sufficient. **GraphRACE** then exploits shared stopping-event evaluations, behavioral clustering, and risk-aware Pareto selection to calibrate a frozen persistence-based stopping

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.

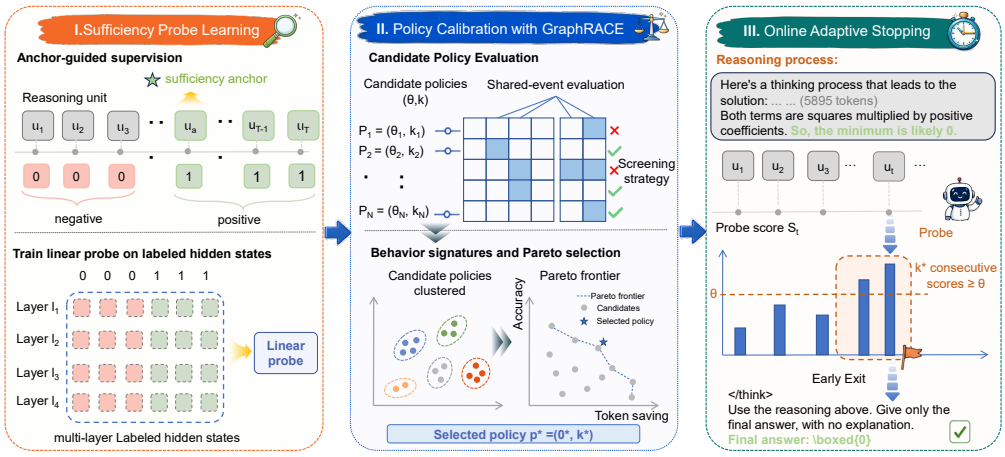


Figure 1: Overview of SPARE. Verified anchors supervise a hidden-state probe; GraphRACE calibrates a persistence-based stopping policy. At inference time, the probe monitors reasoning-unit endpoints and triggers a guided answer readout when the stopping condition is met.

policy offline. At inference time, SPARE leaves the reasoning backbone unchanged and requires no auxiliary verifier, correctness feedback, policy updates, or online search.

We evaluate SPARE on four reasoning models ranging from 1.5B to 9B parameters across mathematical and scientific reasoning benchmarks. SPARE largely preserves accuracy and often improves it—by more than 15 percentage points in the strongest case—while reducing net token usage by approximately 30%–50%, including stopping-time readout overhead. These results indicate that reasoning sufficiency can be detected from intermediate representations and translated into substantial inference savings without requiring models to complete their full reasoning trajectories.

Our contributions are threefold:

- **Latent reasoning sufficiency boundaries.** We formulate overthinking as computation beyond a sample-dependent sufficiency boundary and provide empirical evidence that answer correctness follows an approximately monotonic transition along reasoning trajectories.
- **Anchor-guided hidden-state supervision.** We derive multiple anchor-relative labels from verified answer-related anchors, enabling a lightweight probe to detect answer-sufficient states without modifying the reasoning backbone.
- **Compute-efficient risk-aware calibration.** We develop GraphRACE, an offline procedure that reuses shared stopping events and selects a frozen persistence-based policy from the empirical accuracy–efficiency frontier.

2 Related Work

2.1 Efficient Reasoning

Prior work improves reasoning efficiency through model training, computation control, and trace compression. Training-based methods encourage shorter reasoning or earlier termination through supervised or reinforcement learning (Ma et al., 2025; Fang et al., 2026; Han et al., 2025a), but require additional optimization, data construction, or reward design. Budget-aware approaches allocate computation by problem difficulty (Han et al., 2025b; Li et al., 2025), while others compress reasoning steps (Xu et al., 2025; Aytes et al., 2025), suppress redundant reflection, or steer internal representations toward concise trajectories (Huang et al., 2026b; Wang et al., 2025; Liu et al., 2025; Huang et al., 2026a). APR uses answer-related anchors to penalize answer-stable tails during training (Chang et al., 2026). SPARE instead freezes the backbone and uses verified anchors only to supervise a lightweight post-hoc detector.

2.2 Adaptive Early Exit for Reasoning Models

Adaptive early-exit methods primarily rely on answer-level signals or reasoning dynamics. Answer-level approaches generate or estimate intermediate answers and stop according to confidence, convergence, entropy, or cross-sample agreement (Yang et al., 2025; Liu & Wang, 2025; Mao et al., 2025; Laaouach, 2025; Fu et al., 2024). These methods incur auxiliary generation or sampling costs, while answer confidence remains an indirect proxy for correctness sufficiency. Other approaches monitor reflection redundancy, path deviation, semantic convergence, or temporal confidence changes (Sun et al., 2026; Guan et al., 2026; Wei et al., 2025; Hosseini et al., 2026; Min et al., 2026; Tikhonov et al., 2026), often using heuristic linguistic or statistical patterns. DTSR prompts the original reasoning model at reflection cues to assess whether the current thought is sufficient, requiring an additional full-model call for each check (Xiang et al., 2026). SPARE instead applies a lightweight linear classifier to hidden representations at reasoning-unit boundaries, avoiding repeated answer generation and online textual self-assessment.

2.3 Hidden-State Sufficiency Probing

Intermediate activations encode information about correctness, reasoning outcomes, and completion progress (Zhang et al., 2025; David, 2025; Liu et al., 2026; Eisenstadt et al., 2025). FlashThink trains an exit verifier from truncated-prefix outcomes, requiring candidate endpoints to be completed and evaluated during supervision construction (Jiang et al., 2025). LYNX probes hidden states at naturally occurring reflection cues, constructs labels through forced-exit readouts, and applies split conformal prediction to control premature-exit risk (Akgül et al., 2025); its supervision therefore requires multiple readouts and remains tied to reflection events. SPARE instead derives multiple anchor-relative labels from each verified anchor and detects sample-dependent sufficiency transitions using a lightweight linear probe. GraphRACE further calibrates persistence-based stopping policies against empirical accuracy risk and end-to-end token cost.

3 Empirical Evidence for Reasoning Sufficiency Boundaries

We conduct a controlled truncation study to examine whether answer sufficiency exhibits a consistent transition as reasoning progressively unfolds.

3.1 Controlled Truncation Study

We randomly select 200 problems from the MATH calibration set and generate complete trajectories with Qwen3.5-4B. Each trajectory is truncated at nine fractions, $\mathcal{Q} = \{0.1, 0.2, \ldots, 0.9\}$, measured over the reasoning trace excluding the original final-answer segment. Each target is aligned backward to the nearest complete reasoning-unit boundary to avoid incomplete sentences, equations, or reasoning steps.

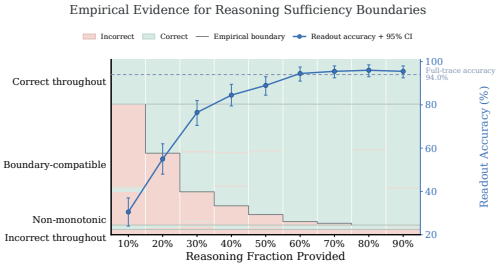


Figure 2: Controlled truncation study on 200 MATH calibration problems using Qwen3.5-4B. Background cells show sample-level readout correctness across nine reasoning fractions, the gray staircase denotes empirical transition boundaries, and the blue curve reports aggregate readout accuracy with bootstrap 95% confidence intervals. The horizontal reference marks full-trace accuracy.

At each endpoint, the model produces a concise answer from the problem and retained prefix. Evaluating it against the reference answer yields

$$\mathbf{y}_i = (y_{i,0.1}, y_{i,0.2}, \ldots, y_{i,0.9}). \qquad (1)$$

where $y_{i,q} \in \{0, 1\}$, with $y_{i,q} = 1$ indicating that the answer read out from the prefix at fraction q is scored as correct. Readout correctness is measured under a fixed answer-readout template, decoding configuration, and scoring protocol, and serves as an empirical indicator of reasoning sufficiency.

3.2 Boundary Patterns and Empirical Findings

A trajectory exhibits an *exact reasoning sufficiency boundary* when its correctness sequence follows a single monotonic transition,

$$0, \ldots, 0, 1, \ldots, 1. \tag{2}$$

An *approximate boundary* differs from its closest monotonic step sequence at no more than one checkpoint. We term both cases *boundary-compatible*; uniformly correct or incorrect trajectories are categorized as *correct throughout* or *incorrect throughout*, and larger deviations as *non-monotonic*.

Among the 200 trajectories, 126 exhibit an exact boundary and 12 an approximate boundary; 51 are correct throughout, six are incorrect throughout, and five are strongly non-monotonic. Among the 143 trajectories with observable correctness changes, 138 (96.5%) are boundary-compatible. Incorrect-to-correct transitions also substantially outnumber correct-to-incorrect transitions, 148 to 18. We quantify this asymmetry as

$$\mathrm{Dir} = \frac{N_{0\to1} - N_{1\to0}}{N_{0\to1} + N_{1\to0}}, \tag{3}$$

obtaining $\mathrm{Dir} = 0.783$, with a bootstrap 95% confidence interval of $[0.699, 0.869]$.

Figure 2 shows that readout accuracy rises from 30.5% at 10% of the reasoning trace to 76.5% at 30%, approaching the 94.0% full-trace accuracy after roughly 60%. Sample-level outcomes similarly concentrate around monotonic incorrect-to-correct transitions, and a permutation test indicates that this structure is unlikely under randomly ordered checkpoints ($p < 10^{-4}$).

Algorithm 1 GraphRACE: Offline Risk-Aware Policy Calibration

Require: Candidate policies $\mathcal{P}$, calibration set $\mathcal{I}_{\mathrm{cal}}$, and risk tolerances
Ensure: Frozen policy $p^\star$

1: Compute stopping mappings $\boldsymbol{\tau}(p)$ and normalized profiles $\mathbf{f}(p)$ for all $p \in \mathcal{P}$; deduplicate identical mappings.
2: Query shared stopping events once and cache their readout outcomes and costs.
3: Remove provisionally unsafe or dominated policies.
4: Complete-linkage cluster the survivors into $\{\mathcal{C}_j\}_{j=1}^{J}$, with $J \le 10$.
5: Select minimax representatives

$$\mathcal{M} \leftarrow \left\{ \arg\min_{p\in\mathcal{C}_j} \max_{q\in\mathcal{C}_j} d(\mathbf{f}(p), \mathbf{f}(q)) \right\}_{j=1}^{J}.$$

6: Query uncached events and fully evaluate all $p \in \mathcal{M}$.
7: Construct the frontier by maximizing accuracy gain and net saving and minimizing harm:

$$\mathcal{F} \leftarrow \mathrm{Pareto}_3\big(\mathcal{M} \cap \mathcal{P}_{\mathrm{feas}}\big).$$

8: $p^\dagger \leftarrow \textsc{SelectPolicy}(\mathcal{F})$.
9: $\mathcal{N} \leftarrow \textsc{Refine}(p^\dagger)$; fully evaluate $\mathcal{N}$ and set $\mathcal{M} \leftarrow \mathcal{M} \cup \mathcal{N}$.
10: $\mathcal{F} \leftarrow \mathrm{Pareto}_3(\mathcal{M} \cap \mathcal{P}_{\mathrm{feas}})$.
11: $p^\star \leftarrow \textsc{SelectPolicy}(\mathcal{F})$.
12: **return** $p^\star$

Together, the dominant transition direction, high prevalence of boundary-compatible trajectories, and early accuracy plateau support a sample-dependent sufficiency boundary that SPARE seeks to detect from intermediate hidden representations.

4 METHOD

Overview. As illustrated in Figure 1, SPARE follows a learn–calibrate–exit pipeline. It first derives anchor-relative supervision from verified answer-related anchors and trains a lightweight linear probe on multi-layer hidden representations to detect answer-sufficient reasoning states. GraphRACE then performs offline risk-aware calibration through shared-event screening, behavioral clustering, and Pareto selection, producing a frozen persistence-based stopping policy. At inference time, the probe scores each valid reasoning-unit boundary, and generation stops once the calibrated persistence condition is satisfied, followed by a single concise final-answer readout.

4.1 PROBLEM FORMULATION

Given an input problem x_i, a reasoning model generates a sequence of reasoning units

$$\mathcal{R}_i = (u_{i,1}, u_{i,2}, \ldots, u_{i,T_i}), \tag{4}$$

followed by a final answer. Each unit $u_{i,t}$ is a semantically complete segment, such as a sentence, equation, or self-contained reasoning step, and $\mathcal{R}_{i,\le t}$ denotes the prefix ending at unit t. We segment reasoning at sentence endings and structural boundaries (paragraphs, headings, and list items),

treating the resulting unit endpoints as valid boundaries. Detailed segmentation rules are provided in Appendix A.3.

To determine whether a prefix already contains sufficient information, we terminate reasoning at unit t and ask the same model to produce a concise final-answer readout. We define its empirical sufficiency label as

$$y_{i,t} = \mathbb{I}\left[\mathrm{Eval}\left(\mathrm{Readout}\left(x_i, \mathcal{R}_{i,\le t}\right)\right) = 1\right], \tag{5}$$

where $\mathrm{Eval}(\cdot)$ is the task-specific answer evaluator. Thus, $y_{i,t} = 1$ indicates that the reasoning accumulated through unit t is sufficient to recover the correct answer.

When the correctness sequence contains a stable correct suffix, we define the empirical reasoning sufficiency boundary as

$$\tau_i^* = \min\left\{t : y_{i,s} = 1,\ \forall s \ge t\right\}. \tag{6}$$

If no stable correct suffix exists, we set $\tau_i^* = \infty$. The boundary τ_i^* marks the onset of a consistently correct readout suffix under the specified readout protocol; We distinguish the empirical boundary τ_i^* from the supervision anchor a_i, which serves as a proxy for this boundary, and the actual exit position $\widehat{\tau}_i$, determined by the stopping policy. SPARE uses anchor supervision to learn sufficiency-related stopping signals from intermediate hidden representations, without evaluating $y_{i,t}$ at every reasoning endpoint.

4.2 Anchor-Guided Sufficiency Probe Learning

Observing $y_{i,t}$ at every reasoning-unit boundary would require a separate answer readout at each endpoint. SPARE instead treats verified answer-related anchors as weak estimates of the latent sufficiency boundary and derives multiple supervision events from each verification.

Anchor localization. Given a completed trajectory $\mathcal{R}_i$, we first search its reasoning units in order for an answer-related expression matching the reference answer. Only if no such match is found do we repeat the search using the completed trajectory's answer as a self-reference. Candidates include explicit conclusions and answer-bearing mathematical expressions with local context. We denote the first matching unit under this reference-first search order by a_i.

Anchor reliability filtering. Because an answer mention need not imply sufficient preceding reasoning, we truncate the trajectory at a_i and generate an anchor-level answer $\widehat{z}_i^{\,a}$. A parsable candidate is retained only if

$$\frac{b_{i,a_i}}{B_i^{\mathrm{reason}}} \le \gamma \quad \wedge \quad \mathrm{Match}\big(\widehat{z}_i^{\,a}; z_i^*, z_i^{\mathrm{orig}}\big). \tag{7}$$

Here, b_{i,a_i} and B_i^{reason} denote the anchor endpoint and total reasoning length. The match condition holds when $\widehat{z}_i^{\,a}$ is correct with respect to z_i^* or equivalent to the completed answer z_i^{orig}. The position constraint excludes late anchors. Reference correctness remains the primary signal. Agreement between the anchor readout and the completed answer indicates that the latter can already be elicited from the retained prefix. We use this agreement as auxiliary weak supervision for answer consistency, not as evidence of gold correctness.

Anchor-relative label sampling. For each accepted anchor, we select up to 30 reasoning units before and up to 30 after it, using seeded hash ranking of endpoint identifiers on each side. The anchor itself is included separately. Let $\mathcal{P}_i \subseteq \{1, \ldots, a_i - 1\}$ and $\mathcal{Q}_i \subseteq \{a_i + 1, \ldots, T_i\}$ denote the selected pre-anchor and post-anchor indices. We assign

$$\widetilde{y}_{i,t} = \begin{cases} 0, & t \in \mathcal{P}_i, \\ 1, & t \in \{a_i\} \cup \mathcal{Q}_i. \end{cases} \tag{8}$$

The pre- and post-anchor labels are weak supervision assumptions. Each accepted anchor yields at most 61 labeled endpoints.

Table 1: Main results under a unified post-truncation readout and scoring protocol. Acc., Tok., and CR denote final-answer accuracy, average completion-token cost, and compression rate relative to VANILLA, respectively. Token costs include readout and auxiliary-generation overhead. Overall results aggregate all 758 test examples. Best and second-best distinct results within each model–benchmark block are bolded and underlined, respectively; ties share the same formatting.

Model	Method	MATH-500		GPQA-D		AIME25		AIME26		Overall		
		Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	Tok. ↓	CR ↓
DeepSeek-1.5B	Vanilla	83.00	100.00	**31.31**	100.00	**23.33**	100.00	16.67	100.00	**64.51**	7,391	100.00
	DEER	80.80	**50.74**	30.81	98.51	**23.33**	78.69	16.67	79.56	62.93	5,357	72.49
	LYNX$_{c097}$	81.80	55.18	30.81	98.21	**23.33**	87.43	16.67	92.42	63.59	5,639	76.30
	RPDI	82.20	81.06	**31.31**	99.48	**23.33**	82.96	13.33	68.18	63.85	6,403	86.64
	REFRAIN	83.00	81.63	29.80	79.71	**23.33**	80.35	**20.00**	83.59	64.25	5,988	81.02
	SPARE	**83.20**	54.95	29.29	**68.33**	**23.33**	**62.72**	16.67	**62.26**	64.12	**4,506**	**60.97**
Qwen3.5-2B	Vanilla	65.80	100.00	55.56	100.00	20.00	100.00	10.00	100.00	59.10	18,987	100.00
	DEER	**82.80**	75.77	**56.06**	96.79	**26.67**	79.77	**26.67**	81.49	**71.37**	15,313	80.65
	LYNX$_{c097}$	73.40	**31.37**	55.05	67.56	10.00	59.17	3.33	53.96	63.32	**7,967**	**41.96**
	RPDI	68.40	46.64	51.52	76.31	13.33	**49.88**	3.33	**47.07**	59.23	10,037	52.86
	REFRAIN	74.00	69.45	55.05	94.11	20.00	93.17	13.33	90.52	64.51	14,687	77.36
	SPARE	82.20	50.06	55.56	**56.38**	**26.67**	66.30	13.33	60.35	70.32	10,074	53.06
Qwen3.5-4B	Vanilla	90.60	100.00	78.28	100.00	50.00	100.00	60.00	100.00	84.56	11,183	100.00
	DEER	**94.40**	68.57	78.28	98.67	**63.33**	78.75	**73.33**	81.96	**88.13**	8,677	77.59
	LYNX$_{c097}$	94.00	**58.92**	**78.79**	90.85	56.67	80.15	70.00	75.16	87.60	7,788	69.64
	RPDI	90.00	78.33	78.28	93.99	50.00	**72.85**	56.67	**73.66**	84.04	9,051	80.94
	REFRAIN	92.40	75.56	77.27	85.82	53.33	88.70	66.67	84.71	85.88	8,941	79.96
	SPARE	**94.40**	60.52	77.27	**71.73**	50.00	76.15	70.00	77.56	87.20	**7,392**	**66.10**
Qwen3.5-9B	Vanilla	94.00	100.00	**82.32**	100.00	50.00	100.00	66.67	100.00	88.13	10,689	100.00
	DEER	94.80	65.71	**82.32**	99.71	60.00	72.16	73.33	78.93	89.31	8,157	76.31
	LYNX$_{c097}$	93.00	**51.83**	80.81	82.91	**66.67**	74.00	66.67	69.93	87.73	6,800	63.62
	RPDI	92.40	78.75	81.82	95.08	53.33	**62.75**	66.67	74.85	87.07	8,668	81.10
	REFRAIN	94.20	77.77	81.31	87.51	53.33	90.79	70.00	88.90	88.26	8,823	82.54
	SPARE	**95.40**	53.64	80.81	**62.79**	**66.67**	64.40	**76.67**	**69.13**	**89.71**	**6,248**	**58.45**

Hidden-state sufficiency probe. For each labeled endpoint, we concatenate hidden states from four approximately evenly spaced decoder layers, standardize them using probe-training statistics, and apply a linear probe:

$$\mathbf{v}_{i,t} = [\mathbf{h}_{i,t}^{(\ell_1)}; \ldots; \mathbf{h}_{i,t}^{(\ell_4)}], \quad \widetilde{\mathbf{v}}_{i,t} = (\mathbf{v}_{i,t} - \boldsymbol{\mu}) \oslash \boldsymbol{\sigma}, \quad s_{i,t} = \mathbf{w}^\top \widetilde{\mathbf{v}}_{i,t} + b. \tag{9}$$

Here, $\mathbf{h}_{i,t}^{(\ell)}$ is the hidden state at the last token of reasoning unit $u_{i,t}$ in decoder layer ℓ, and $\oslash$ denotes element-wise division. Larger $s_{i,t}$ indicates stronger evidence of reasoning sufficiency. The feature-wise mean $\boldsymbol{\mu}$ and standard deviation $\boldsymbol{\sigma}$ are estimated from the probe-training set. Hidden-state extraction details are provided in Appendix A.4.

Let $\mathcal{E}_{\text{train}}$ denote the retained labeled reasoning-unit endpoints for a given backbone. We train the probe using class-balanced binary cross-entropy:

$$\mathcal{L}_{\text{probe}} = -\frac{1}{|\mathcal{E}_{\text{train}}|} \sum_{(i,t)\in\mathcal{E}_{\text{train}}} \Big[w_+ \, \widetilde{y}_{i,t} \log \text{sigmoid}(s_{i,t}) \quad + (1 - \widetilde{y}_{i,t}) \log(1 - \text{sigmoid}(s_{i,t})) \Big]. \tag{10}$$

Here, $w_+ = N_-/N_+$, where N_- and N_+ count endpoints with labels 0 and 1, respectively, in $\mathcal{E}_{\text{train}}$. The weight is computed once from this training set and remains fixed during training.

4.3 RISK-AWARE STOPPING POLICY CALIBRATION

Persistence-based stopping. Given probe scores $\mathbf{s}_i = (s_{i,1}, \ldots, s_{i,T_i})$, SPARE defines a policy $p = (\theta, k, m)$, where θ is the logit threshold, k is the persistence length, and m is the earliest permitted exit unit. The stopping position is

$$\widehat{\tau}_i(p) = \min \left\{ t \geq \max(m, k) : \min_{0 \leq j < k} s_{i,t-j} \geq \theta \right\}. \tag{11}$$

Table 2: Inference-time ablations on Qwen3.5-4B and Qwen3.5-9B. Fixed Token replaces adaptive stopping with a shared token budget, while Fixed Threshold $q_{0.90}$ uses the 90th percentile of calibration-set reasoning-unit logits with persistence $k = 2$. CR is normalized to the corresponding VANILLA backbone. The highest accuracy and lowest token cost within each model block are bolded.

		MATH-500		GPQA-D		AIME25		AIME26		Overall		
Model	Method	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	Tok. ↓	CR ↓
Qwen3.5-4B	Fixed Token 8K	90.80	**56.47**	75.76	74.13	36.67	**31.36**	40.00	**30.91**	82.72	**6,236**	**55.76**
	Fixed Token 16K	94.00	82.03	**78.28**	97.85	50.00	58.98	60.00	60.86	86.81	9,114	81.50
	Fixed Threshold $q_{0.90}$	93.80	74.47	**78.28**	84.42	**60.00**	88.18	**70.00**	86.69	**87.47**	8,850	79.14
	SPARE	**94.40**	60.52	77.27	**71.73**	50.00	76.15	**70.00**	77.56	87.20	7,392	66.10
Qwen3.5-9B	Fixed Token 8K	91.40	61.53	78.79	69.10	40.00	**31.83**	43.33	**32.55**	84.17	**6,199**	**58.00**
	Fixed Token 16K	94.40	85.86	80.81	92.90	60.00	60.52	60.00	62.66	88.13	8,887	83.14
	Fixed Threshold $q_{0.90}$	95.00	72.95	**81.31**	81.75	**70.00**	83.59	**76.67**	86.78	**89.71**	8,285	77.51
	SPARE	**95.40**	**53.64**	80.81	**62.79**	66.67	64.40	**76.67**	69.13	**89.71**	6,248	58.45

If such a position exists, reasoning stops at $\widehat{\tau}_i(p)$, followed by one concise final-answer readout; otherwise, generation continues to completion. Persistence suppresses isolated score fluctuations, while m prevents implausibly early exits. We fix m before calibration, so GraphRACE searches only over (θ, k).

Calibration objectives. Let $b_{i,t}$ denote the tokens generated through unit t, $r_{i,t}$ the corresponding readout length, and B_i the full-generation cost. The realized cost is $C_i(p) = b_{i,\widehat{\tau}_i(p)} + r_{i,\widehat{\tau}_i(p)}$ when an exit is triggered and $C_i(p) = B_i$ otherwise. We measure efficiency by $\mathrm{Save}_{\mathrm{net}}(p) = 1 - \sum_i C_i(p) / \sum_i B_i$, including readout overhead. Let $c_i(p)$ denote the correctness of the triggered readout, or the baseline answer when no exit occurs, and let c_i^{base} denote baseline correctness. The accuracy change is $\Delta\,\mathrm{Acc}(p) = N_{\mathrm{cal}}^{-1} \sum_i \big(c_i(p) - c_i^{\mathrm{base}}\big)$. We additionally record the conditional harm rate $\mathrm{Harm}(p)$, defined as the fraction of baseline-correct examples made incorrect by early stopping, and use it as an auxiliary safety objective.Using fixed calibration folds, we define the empirically risk-feasible set as

$$\mathcal{P}_{\mathrm{feas}} = \big\{p :\ \Delta\,\mathrm{Acc}(p) \geq -\epsilon, \quad \min_f \Delta\,\mathrm{Acc}_f(p) \geq -\epsilon_{\mathrm{fold}}, \quad \mathrm{Trig}(p) > 0\big\}. \tag{12}$$

Here, $\Delta\,\mathrm{Acc}_f(p)$ is the accuracy change on fold f, $\mathrm{Trig}(p)$ is the trigger rate, and ϵ and ϵ_{fold} are prescribed risk tolerances.

Offline calibration with GraphRACE. GraphRACE amortizes readout cost by caching each stopping event (i, t) once and reusing its outcome across all policies that trigger there. After provisional safety and dominance screening, each surviving policy is represented by a normalized stopping profile $\mathbf{f}(p)$, where $f_i(p) = b_{i,\widehat{\tau}_i(p)}/B_i$ for triggered exits and $f_i(p) = 1$ otherwise. The behavioral distance $d(\mathbf{f}(p), \mathbf{f}(q))$ is the mean absolute difference between profile components. Starting from singleton clusters, complete-linkage repeatedly merges the pair with the smallest maximum cross-cluster distance until at most ten clusters remain. Each cluster's minimax representative, which minimizes the maximum behavioral distance to its members, is then fully evaluated. It then constructs a risk-feasible Pareto frontier over $\Delta\,\mathrm{Acc}(p)$, $\mathrm{Save}_{\mathrm{net}}(p)$, and conditional harm, selects a provisional policy, evaluates neighboring thresholds, and rebuilds the frontier after merging the refined candidates. The final policy is chosen by distance to the ideal point in the normalized three-objective space when the frontier contains at least three policies, with deterministic tie-breaking otherwise. The resulting $p^\star = (\theta^\star, k^\star)$ is frozen before formal evaluation. Algorithm 1 summarizes the complete calibration procedure.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Benchmarks and data splits. We evaluate SPARE on MATH-500 (Hendrycks et al., 2021), GPQA-Diamond (GPQA-D) (Rein et al., 2023), AIME25, and AIME26 (Zhang & Math-AI, 2025; 2026), with 500, 198, 30, and 30 problems, respectively (758 total). Following Akgül et al. (2025), we split 2,000 stratified MATH training examples 2:1 into 1,333 probe-training and 667 policy-calibration examples, disjoint from each other and all test benchmarks. Both probe and policy are frozen before evaluation. We set the anchor position limit to $\gamma = 0.6$ and earliest exit to $m = 32$ reasoning units, and search $k \in \{1, 2, 3\}$ with 40 threshold representatives per persistence spanning the calibration-score range. See Appendices A.3–A.4 for supervision and probe training, and Appendices B.1–B.3 for calibration settings and selected policies.

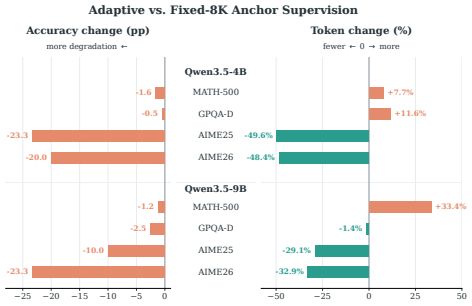


Figure 3: Ablation of sample-adaptive anchor supervision. Bars report Fixed-8K minus SPARE changes in accuracy (percentage points) and average effective completion tokens (relative percentage); negative token changes indicate lower cost. Both variants retain adaptive detector-based stopping and GraphRACE calibration.

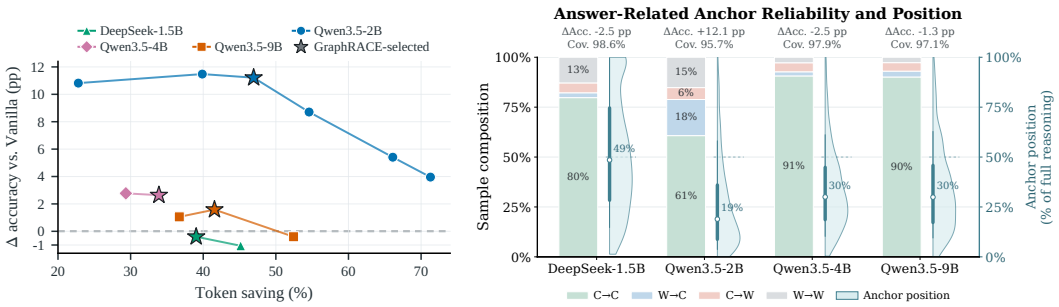
Backbones and baselines. We evaluate four reasoning backbones: DeepSeek-R1-Distill-Qwen-1.5B (hereafter DeepSeek-1.5B)(DeepSeek-AI, 2025) and Qwen3.5 models with 2B, 4B, and 9B parameters (Qwen Team, 2026). VANILLA denotes unmodified full-chain generation and serves as the accuracy and token-cost reference. We compare SPARE with LYNX, which probes hidden states at reflection cues with conformal calibration (Akgül et al., 2025); RPDI, which monitors reasoning-path deviation around high-entropy transitions (Guan et al., 2026); DEER, which exits according to trial-answer confidence (Yang et al., 2025); and REFRAIN, which detects redundant reflection using an adaptive threshold (Sun et al., 2026). All experiments use a maximum generation length of 32K tokens. Backbone decoding settings and the shared readout template are provided in Appendix A.2. LYNX uses conformal confidence $c = 0.97$ in Table 1; results for $c \in \{0.90, 0.95, 0.97\}$ are reported in Appendix C.2.

Metrics and evaluation protocol. We report final-answer accuracy (**Acc.**), average effective completion tokens (**Tok.**), and compression rate (**CR**), defined as the aggregate token cost relative to the corresponding VANILLA backbone; net token saving is therefore $1 - \mathrm{CR}$. Effective token counts include retained reasoning prefixes, stopping-time readouts, and method-specific auxiliary generations. We apply the same post-truncation answer template and evaluator to all methods. This controlled protocol isolates their ability to identify effective exit positions from differences in answer elicitation or scoring.

5.2 MAIN RESULTS

Overall performance. Table 1 shows that SPARE substantially reduces token usage while largely preserving or improving accuracy. Across the four backbones, it achieves 33.90%–46.94% net token savings, including stopping-time readout overhead. Relative to VANILLA, overall accuracy improves by 11.22, 2.64, and 1.58 percentage points on Qwen3.5-2B, Qwen3.5-4B, and Qwen3.5-9B, respectively, while DeepSeek-1.5B incurs only a 0.39-point decrease with 39.03% token saving. As reported in Appendix B.5, GraphRACE reduces unique calibration readout queries by 65.0–73.9% relative to exhaustive evaluation of all 120 candidate policies with shared-event reuse.

Comparison with baselines and cross-task trends. SPARE provides a favorable accuracy–efficiency balance across the Qwen3.5 series. On Qwen3.5-2B, DEER is 1.05 points more accurate but requires substantially more tokens, whereas LYNX is more efficient but markedly less accurate. On Qwen3.5-4B, SPARE achieves the lowest token cost while remaining within one accuracy point of the best baseline; on Qwen3.5-9B, it attains both the highest overall accuracy and lowest token cost. RPDI and REFRAIN are competitive in selected settings but yield less consistent overall trade-offs. SPARE performs particularly strongly on MATH-500 and across several AIME settings, while providing substantial GPQA-D compression with modest accuracy changes.



(a) (b)

Figure 4: Analysis of SPARE. **(a)** Formal-test accuracy change versus net token saving relative to VANILLA. Stars mark GraphRACE-selected policies; lines connect policies frozen during calibration. **(b)** Stacked bars show correctness transitions from full-generation answers to anchor readouts; labels report accuracy changes and coverage. Half-violins show normalized anchor positions, with thin lines, thick lines, and white circles indicating the 10th–90th percentiles, interquartile ranges, and medians.

5.3 ABLATION STUDIES

Inference-time design choices. Table 2 evaluates adaptive stopping and policy calibration. Fixed Token 8K sacrifices accuracy for compression, while 16K recovers accuracy with limited token savings. Compared with Fixed Threshold $q_{0.90}$, SPARE achieves lower overall CR on both models, at a 0.27-percentage-point accuracy cost on Qwen3.5-4B and with unchanged overall accuracy on Qwen3.5-9B. These results support the accuracy–efficiency benefits of adaptive stopping and GraphRACE calibration.

Effect of adaptive anchor supervision. Figure 3 compares adaptive anchors with fixed-8K supervision under unchanged adaptive inference and GraphRACE calibration. Fixed anchors reduce accuracy in all eight model–benchmark settings. On AIME25/26, they use 29.1%–49.6% fewer tokens than SPARE but lose 10.0–23.3 accuracy points, while providing no consistent savings on MATH-500 or GPQA-D.

5.4 FURTHER DISCUSSION

Accuracy–efficiency trade-offs. Figure 4a shows post-refinement policies frozen before formal evaluation. Most Qwen3.5 operating points improve on full-generation accuracy across a broad range of token savings, supporting multiple effective accuracy–efficiency trade-offs.

Reliability of answer-related anchors. Figure 4b shows anchor coverage of 95.7%–98.6% and median normalized positions of 18.9%–48.5%. Anchor readout accuracy stays within 2.5 points of full-generation accuracy on DeepSeek-1.5B, Qwen3.5-4B, and Qwen3.5-9B, and improves by 12.1 points on Qwen3.5-2B.

6 CONCLUSION

We introduced SPARE, an adaptive early-exit framework that identifies sample-dependent sufficiency boundaries beyond which further reasoning is often unnecessary. SPARE derives anchor-relative supervision from verified answer-related anchors, learns a lightweight hidden-state sufficiency probe, and uses GraphRACE to calibrate a frozen persistence-based stopping policy through shared-event evaluation and risk-aware Pareto selection. Across four reasoning models and four mathematical and scientific benchmarks, SPARE reduces net token usage by 30%–50%. These results demonstrate that intermediate representations contain actionable signals of answer sufficiency, shifting efficient reasoning from preallocating computation toward detecting when sufficient reasoning has already been completed.

AI USE STATEMENT

We used large language models, specifically ChatGPT (GPT-6, OpenAI) and Gemini (Google), in limited and auxiliary ways during this work. First, we used them to improve grammar, phrasing, and the clarity of exposition, without relying on them to generate substantive technical content. Second, we occasionally consulted them to help identify potentially relevant related work; all resulting references and citations were independently checked and verified by the authors. Third, at early stages of the project, we used them as brainstorming tools to explore alternative perspectives and possible problem formulations. All research directions, methodological designs, and experiments were conceived, developed, and validated by the authors. Any outputs produced by large language models were critically reviewed before being incorporated into the manuscript, and all intellectual contributions of this work are entirely those of the authors.

REPRODUCIBILITY STATEMENT

Experimental settings, evaluation protocols, and hyperparameters are documented in Section 5.1 and Appendices A and B. We will release the code and experimental configurations to support reproducibility.

REFERENCES

Ömer Faruk Akgül, Yusuf Hakan Kalaycı, Rajgopal Kannan, Willie Neiswanger, and Viktor Prasanna. Lynx: Learning dynamic exits for confidence-controlled reasoning. *arXiv preprint arXiv:2512.05325*, 2025.

Simon A Aytes, Jinheon Baek, and Sung Ju Hwang. Sketch-of-thought: Efficient llm reasoning with adaptive cognitive-inspired sketching. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 24307–24331, 2025.

Kaiyan Chang, Chenwei Zhu, Yingfeng Luo, Yifu Huo, Chenglong Wang, Xiaoqian Liu, Qiaozhi He, Tong Xiao, Zhengtao Yu, and Jingbo Zhu. Apr: Penalizing structural redundancy in large reasoning models via anchor-based process rewards. *arXiv preprint arXiv:2602.00760*, 2026.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*, 2024.

Alejandro Cuadron, Dacheng Li, Wenjie Ma, Xingyao Wang, Yichuan Wang, Siyuan Zhuang, Shu Liu, Luis Gaspar Schroeder, Tian Xia, Huanzhi Mao, et al. The danger of overthinking: Examining the reasoning-action dilemma in agentic tasks. *arXiv preprint arXiv:2502.08235*, 2025.

Joey David. Temporal predictors of outcome in reasoning language models. *arXiv preprint arXiv:2511.14773*, 2025.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Roy Eisenstadt, Itamar Zimerman, and Lior Wolf. Overclocking llm reasoning: Monitoring and controlling thinking path lengths in llms. *arXiv preprint arXiv:2506.07240*, 2025.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Thinkless: Llm learns when to think. *Advances in neural information processing systems*, 38:151268–151295, 2026.

Yichao Fu, Junda Chen, Siqi Zhu, Zheyu Fu, Zhongdongming Dai, Yonghao Zhuang, Yian Ma, Aurick Qiao, Tajana Rosing, Ion Stoica, et al. Efficiently scaling llm reasoning with certaindex. *arXiv preprint arXiv:2412.20993*, 2024.

Weixin Guan, Liang Li, Jiapeng Liu, Bing Li, Peng Fu, Chengyang Fang, Xiaoshuai Hao, Can Ma, and Weiping Wang. Mitigating overthinking in large reasoning language models via reasoning path deviation monitoring. *arXiv preprint arXiv:2603.14251*, 2026.

Jinyi Han, Ying Huang, Ying Liao, Zishang Jiang, Xikun Lu, Haiquan Zhao, Xinyi Wang, Guanghao Zhou, Sihang Jiang, Jiaqing Liang, et al. Your models have thought enough: Training large reasoning models to stop overthinking. *arXiv preprint arXiv:2509.23392*, 2025a.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware llm reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24842–24855, 2025b.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

Parsa Hosseini, Sumit Nawathe, Mahdi Salmani, Meisam Razaviyayn, and Soheil Feizi. Early stopping for large reasoning models via confidence dynamics. *arXiv preprint arXiv:2604.04930*, 2026.

Jiameng Huang, Baijiong Lin, Guhao Feng, Jierun Chen, Di He, and Lu Hou. Efficient reasoning for large reasoning language models via certainty-guided reflection suppression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 31176–31184, 2026a.

Yao Huang, Huanran Chen, Shouwei Ruan, Yichi Zhang, Xingxing Wei, and Yinpeng Dong. Mitigating overthinking in large reasoning models via manifold steering. *Advances in Neural Information Processing Systems*, 38:102543–102568, 2026b.

Guochao Jiang, Guofeng Quan, Zepeng Ding, Ziqin Luo, Dixuan Wang, and Zheng Hu. Flashthink: An early exit method for efficient reasoning. *arXiv preprint arXiv:2505.13949*, 2025.

Yassir Laaouach. Halt-cot: Model-agnostic early stopping for chain-of-thought reasoning via answer entropy. In *4th Muslims in ML Workshop co-located with ICML 2025*, 2025.

Junyan Li, Wenshuo Zhao, Yang Zhang, and Chuang Gan. Steering llm thinking with budget guidance. *arXiv preprint arXiv:2506.13752*, 2025.

Kaiyuan Liu, Chen Shen, Zhanwei Zhang, Junjie Liu, Xiaosong Yuan, et al. Efficient reasoning through suppression of self-affirmation reflections in large reasoning models. *arXiv preprint arXiv:2506.12353*, 2025.

Kang Liu, Yongkang Liu, Xiaocui Yang, Peidong Wang, Wen Zhang, Shi Feng, Yifei Zhang, and Daling Wang. Neat: Neuron-based early exit for large reasoning models. *arXiv preprint arXiv:2602.02010*, 2026.

Xin Liu and Lu Wang. Answer convergence as a signal for early stopping in reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 17907–17918, 2025.

Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. Cot-valve: Length-compressible chain-of-thought tuning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6025–6035, 2025.

Minjia Mao, Bowen Yin, Yu Zhu, and Xiao Fang. Early stopping chain-of-thoughts in large language models. *arXiv preprint arXiv:2509.14004*, 2025.

Dehai Min, Giovanni Vaccarino, Huiyi Chen, Yongliang Wu, Gal Yona, and Lu Cheng. Stop when reasoning converges: Semantic-preserving early exit for reasoning models. *arXiv preprint arXiv:2605.17672*, 2026.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

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. *arXiv preprint arXiv:2311.12022*, 2023.

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

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Shaochen Zhong, Na Zou, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

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, 2026.

Pavel Tikhonov, Ivan Oseledets, and Elena Tutubalina. Confidence leaps in llm reasoning: Early stopping and cross-model transfer. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 602–616, 2026.

Chenlong Wang, Yuanning Feng, Dongping Chen, Zhaoyang Chu, Ranjay Krishna, and Tianyi Zhou. Wait, we don't need to" wait"! removing thinking tokens improves reasoning efficiency. *arXiv preprint arXiv:2506.08343*, 3, 2025.

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

Zihao Wei, Liang Pang, Jiahao Liu, Wenjie Shi, Jingcheng Deng, Shicheng Xu, Zenghao Duan, Fei Sun, Huawei Shen, and Xueqi Cheng. The evolution of thought: Tracking llm overthinking via reasoning dynamics analysis. *arXiv preprint arXiv:2508.17627*, 2025.

Yang Xiang, Yixin Ji, Ruotao Xu, Dan Qiao, Zheming Yang, Juntao Li, and Min Zhang. When is thinking enough? early exit via sufficiency assessment for efficient reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23541–23556, 2026.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*, 2025.

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. *arXiv preprint arXiv:2504.15895*, 2025.

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.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2025, 2025.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2026, 2026.

A Experimental Protocol and Probe Construction

A.1 Development Data

We sample 2,000 problems from the training split of the Hendrycks MATH dataset (Hendrycks et al., 2021), stratified by mathematical subject and difficulty level. We use 1,333 examples for sufficiency-probe training and the remaining 667 disjoint examples for stopping-policy calibration. All development examples are disjoint from the formal evaluation benchmarks.

A.2 Generation, Evaluation, and Readout

Reasoning generation. We use the officially recommended thinking-mode decoding settings for each model family rather than imposing a shared configuration. Table 3 reports the settings for each evaluated model. Thinking mode is enabled for all models with random seed 42, repetition penalty 1.0, and a maximum generation length of 32,768 tokens. Experiments are conducted on eight NVIDIA RTX 3090 GPUs.

Table 3: Model inference settings. All models use a maximum generation length of 32,768 tokens. Pres. denotes presence penalty.

Model	Temp.	Top-p	Top-k	Pres.	Max tokens
DeepSeek-1.5B	0.6	0.95	-1	0.0	32,768
Qwen3.5-2B	1.0	0.95	20	1.5	32,768
Qwen3.5-4B	1.0	0.95	20	1.5	32,768
Qwen3.5-9B	1.0	0.95	20	1.5	32,768

Post-truncation readout. After an early-exit decision, we retain the reasoning prefix, append the closing thinking marker, and request a concise answer. The readout has the following structure:

```
</think>
Use the reasoning above.  Give only the final answer,
with no explanation.
```

For mathematical tasks, the answer prefill is `Final answer: boxed`; for GPQA-Diamond, it is `ANSWER:`. Final answers are evaluated with the EvalScope-based task-specific scoring protocol used by our implementation. Effective completion cost includes the retained reasoning prefix and the generated post-truncation answer.

A.3 Reasoning Units and Anchor Localization

Algorithm 2 scans each completed trajectory from left to right and returns the earliest answer-equivalent expression supported by a nearby conclusion or verification signal. The rules below make every step deterministic.

Algorithm 2: Verified Anchor Localization

Require: Reasoning trajectory R, gold answer y, parsed trajectory answer $\hat{y}$, position limit γ
Ensure: Candidate anchor a or $\varnothing$

Segment R into ordered units $\mathcal{U} = (u_1, \ldots, u_T)$
$a \leftarrow \varnothing$
if y is available **then**
  $a \leftarrow \textsc{FirstMatch}(\mathcal{U}, y)$
end if
if $a = \varnothing$ and $\hat{y}$ is available **then**
  $a \leftarrow \textsc{FirstMatch}(\mathcal{U}, \hat{y})$
end if
if $a = \varnothing$ or $\mathrm{tokEnd}(a)/|R|_{\mathrm{tok}} > \gamma$ **then**
  return $\varnothing$
end if
return a
function $\textsc{FirstMatch}(\mathcal{U}, r)$
for $t = 1$ to T **do**
  $\mathcal{C}_t \leftarrow \textsc{Deduplicate}(\textsc{ExtractCandidates}(u_t))$
  for candidate $c \in \mathcal{C}_t$ in extraction order **do**
    if $\textsc{MathEqual}(c, r)$ and $\textsc{ContextValid}(\mathcal{U}, t)$ **then**
      return u_t
    end if
  end for
end for
return $\varnothing$

Reasoning units. We first divide each completed reasoning trajectory into semantically coherent units. Boundaries are introduced at blank-line-separated paragraphs, Markdown headings, numbered or bulleted list items, and sentence endings marked by a period, question mark, or exclamation mark when the following text begins a new sentence or reasoning step. Each character-level boundary is then mapped to its exact prefix length with the corresponding model tokenizer. Candidate anchors and inference-time exit positions are restricted to these unit endpoints, avoiding cuts inside an incomplete sentence or reasoning step.

Candidate extraction. Table 4 lists the five candidate sources in their extraction order. If a unit contains several boxed expressions, only the last is used; line-based rules are then applied from top to bottom. We normalize candidates and preserve only the first occurrence of each normalized expression.

Table 4: Candidate sources used by the production anchor localizer.

Source	Extraction rule
Boxed expression	Content of the last `\boxed{...}` in the unit.
Explicit answer	Text following *final answer*, *answer:*, *answer is*, *result is*, or *solution is*.
Inline mathematics	Expressions enclosed by `$...$` or `\(...\)`.
Equality	Right-hand side of the final equality on a line.
Guarded line ending	Final integer, decimal, or percentage on a line after the numerical guards described below.

Structured expressions extracted from these sources include integers, decimals, fractions, percentages, square roots, and multiples of π. The guarded line-ending source is deliberately narrower because an unqualified final number is especially likely to be incidental. It rejects numbers that are fragments of a power, subscript, LaTeX fraction, or slash fraction; parameters of `log`, `ln`, or `sqrt`; single-symbol parameters; and enumeration labels introduced by *step*, *case*, *option*, *choice*, *item*, *part*, *problem*, or *question*. These guards apply to the permissive line-ending source rather than deleting valid structured expressions extracted by stronger rules.

Mathematical equivalence. We first normalize whitespace, mathematical wrappers, `\left`/`\right`, `\dfrac`/`\tfrac`, and simple `\text{...}` wrappers. Exact normalized

matches are accepted immediately. Otherwise, a restricted arithmetic evaluator compares integers, decimals, percentages, fractions, roots, powers, and multiples of π under a restricted deterministic arithmetic grammar. Sequences are compared elementwise, while intervals additionally require matching open or closed endpoints. Expressions outside this restricted grammar fail explicitly rather than being approximately interpreted.

Context gate. Answer equivalence alone is insufficient: the same value may occur in an intermediate computation. Table 5 gives the complete production cue sets and their allowed positions relative to a candidate unit u_t. Matching is case-insensitive. The gate checks the preceding conclusion, current conclusion, following conclusion, and following verification contexts in that order, accepting the first hit.

Table 5: Context signals required for accepting an answer-equivalent candidate in unit u_t.

Signal	Allowed unit	Complete cue set
Conclusion	Previous, current, or next	*final answer*; *answer is*; *answer:*; *result is*; *solution is*; *therefore*; *hence*; *should be*.
Verification	Next only	*check*; *verify*; *make sure*; *double-check*; *confirm*; *wait*; *straightforward*; *miss anything*; *is that right*; *is that correct*; *is that all*.

The preceding/current/following conclusion windows allow the answer expression and its verbal conclusion to fall on adjacent unit boundaries. Verification is restricted to the following unit because it must react to the candidate rather than precede it.

Reference priority and position filtering. Units are searched chronologically, and the first candidate satisfying both mathematical equivalence and the context gate is returned. The complete gold-reference search is performed first. Only when it finds no valid anchor do we restart from the first unit using the parsed answer of the completed model trajectory as a weaker self-reference. Thus, a self-matching expression can never override an available gold-matching anchor.

After localization, we compute the anchor fraction with exact tokenizer positions, $\rho = \mathrm{tokEnd}(a)/|R|_{\mathrm{tok}}$. Probe training retains the sample only when $\rho \leq \gamma = 0.6$; otherwise it is discarded. This position rule is a training-data quality constraint, not part of candidate matching, and it does not search for a replacement anchor. Samples with no valid gold or self anchor likewise contribute no anchor-relative labels.

A.4 Anchor-Relative Supervision and Probe Training

Anchor-relative labels. For each accepted anchor, we construct dense weak labels from the surrounding reasoning units. Up to 30 units before the anchor receive label 0, while the anchor unit and up to 30 units after it receive label 1:

$$\underbrace{0,\ldots,0}_{\text{pre-anchor}} \mid \underbrace{1,\ldots,1}_{\text{anchor and post-anchor}} .$$

This produces multiple supervision events from one verified anchor without requiring a separate answer readout at every reasoning-unit endpoint.

Probe training. At each labeled reasoning-unit endpoint, we extract hidden states from four approximately evenly spaced decoder layers spanning the network depth. The four representations are concatenated, standardized using statistics estimated from the probe-training split, and supplied to a lightweight linear classifier. We train the classifier independently for each backbone using class-balanced binary cross-entropy for 20 epochs, a learning rate of 10^{-3}, batch size 4096, and weight decay 10^{-5}.

B GraphRACE Calibration and Cost

B.1 Policy Space and Balanced Folds

Policy space. Each stopping policy consists of a probe threshold θ, a persistence length k, and a fixed minimum number of reasoning units m. The stopping rule exits at the first eligible unit whose current score and preceding $k-1$ scores all exceed θ. We fix $m = 32$ before calibration, consider $k \in \{1, 2, 3\}$, and construct 40 behavior-mass threshold representatives per persistence over the full calibration-score range. All remaining policy selection uses only Math667.

Balanced calibration folds. We deterministically divide Math667 into five folds while balancing baseline correctness and completion length. Specifically, examples are divided into five equal-frequency length bins and stratified by the Cartesian product of baseline correctness and length bin. Within each stratum, examples are assigned to the currently smallest folds, with aggregate baseline tokens and a hash of the sample identifier providing deterministic tie-breaking. This produces folds of 133 or 134 examples with closely matched baseline accuracy and length distributions.

B.2 Shared-Event Search and Risk-Aware Selection

Shared-event evaluation. GraphRACE represents a candidate policy by the stopping event it requires for each calibration sample. Multiple policies can request the same sample–unit event, so each distinct event is read out once and its correctness and token cost are shared by all policies that use it.

Behavioral consolidation and selection. After deduplicating policies with identical stopping mappings, Net20 removes policies whose gross saving does not exceed 20%. The pilot queries shared events in batches of 128, updating feasibility and dominance bounds from partial observations. Policies may be eliminated based on these bounds, and provisional cluster representatives are updated after screening. Stability checks begin at 512 queried events, with a maximum budget of 1,024 events. After the pilot terminates, the surviving policies are grouped by complete-linkage behavioral clustering, and ten minimax representatives are selected for full evaluation.

For a fully evaluated policy p, let $\Delta C(p)$ be its change in the number of correct answers, $S(p)$ its net saved tokens after readout cost, and

$$H(p) = \frac{N_{1\to 0}(p)}{N_{\text{base}=1}} \tag{13}$$

its conditional harm rate, where $N_{1\to 0}$ counts examples that change from a correct baseline answer to an incorrect early-exit answer. A policy is considered safe only if $\Delta C(p) \geq 0$ both overall and in every one of the five folds. Among safe policies, we construct the Pareto frontier that maximizes ΔC and S while minimizing H.

For a frontier containing at least three policies, each objective is min–max normalized within that frontier:

$$a_p = \frac{\Delta C(p) - \min_q \Delta C(q)}{\max_q \Delta C(q) - \min_q \Delta C(q)}, \tag{14}$$

$$s_p = \frac{S(p) - \min_q S(q)}{\max_q S(q) - \min_q S(q)}, \tag{15}$$

$$h_p = \frac{\max_q H(q) - H(p)}{\max_q H(q) - \min_q H(q)}. \tag{16}$$

A constant objective is assigned normalized value one. We select the policy minimizing

$$d(p) = \|(1, 1, 1) - (a_p, s_p, h_p)\|_2 . \tag{17}$$

With two frontier policies, the lower-harm policy is selected; a unique frontier policy is selected directly. Remaining ties are resolved deterministically by harm, overall and worst-fold correctness, net saving, persistence, threshold, and policy identifier, in that order. If no safe policy exists, the

Table 6: Readout cost for constructing training and calibration assets. R denotes distinct readout events and Tok. denotes generated readout completion tokens.

Model	Method	Train R / Tok.	Calibration R / Tok.	Total R / Tok.
DeepSeek-1.5B	SPARE	1,315 / 14,597	4,000 / 60,114	5,315 / 74,711
	LYNX	16,714 / 411,685	7,639 / 183,369	24,353 / 595,054
Qwen3.5-2B	SPARE	1,276 / 13,851	4,984 / 63,009	6,260 / 76,860
	LYNX	27,940 / 517,105	14,137 / 263,134	42,077 / 780,239
Qwen3.5-4B	SPARE	1,305 / 8,497	4,037 / 29,917	5,342 / 38,414
	LYNX	9,351 / 72,028	4,532 / 36,480	13,883 / 108,508
Qwen3.5-9B	SPARE	1,295 / 9,496	4,571 / 36,582	5,866 / 46,078
	LYNX	10,937 / 143,355	5,231 / 68,170	16,168 / 211,525
Total	**SPARE**	**5,191 / 46,441**	**17,592 / 189,622**	**22,783 / 236,063**
	LYNX	**64,942 / 1,144,173**	**31,539 / 551,153**	**96,481 / 1,695,326**

predefined fallback prioritizes lower harm and smaller accuracy regression rather than consulting formal-test results.

Finally, one local refinement round evaluates at most four thresholds around the provisional winner: the midpoints toward the left and right behavioral-region boundaries and the immediately adjacent score breakpoints. Candidates with duplicate stopping mappings are discarded, the new observations are merged into the shared cache, and the same three-objective selector is rerun on the combined certified set.

B.3 Frozen Policies

All policy selection and refinement are performed exclusively on Math667. The selected policies are frozen before formal evaluation; formal-test correctness is never used for policy selection, refinement, or reselection. The final persistence and threshold values are:

Model	Threshold	Persistence	Minimum units
DeepSeek-1.5B	5.330383	2	32
Qwen3.5-2B	4.201902	2	32
Qwen3.5-4B	6.206575	2	32
Qwen3.5-9B	5.706828	2	32

B.4 Data Construction and Calibration Cost

Accounting protocol. Among the evaluated baselines, LYNX is the closest to SPARE because both train hidden-state probes using intermediate answer readouts and calibrate an early-exit rule offline. Table 6 compares their training- and calibration-data construction costs. A readout event denotes one unique sample–position query, and output tokens are the completion tokens generated by these readouts.

Both methods use the same Math1333 training pool and disjoint Math667 calibration pool, the same deterministic concise-readout template, and the same answer scorer. SPARE uses one readout per selected anchor; each accepted anchor yields up to 61 weak labels (30 pre-anchor, one anchor, and 30 post-anchor) without further generation. LYNX instead reads out every retained cue event after skipping the first five cues, up to 30 events per trajectory. On its calibration split, these cue readouts construct the labeled set used by split conformal calibration.

Across the four models, LYNX requires 4.23 times as many auxiliary readout calls and generates 7.18 times as many readout output tokens as SPARE. Equivalently, SPARE reduces these quantities by 76.4% and 86.1%, respectively. The largest difference occurs during probe-data construction: SPARE expands one verified anchor into multiple hidden-state labels, whereas LYNX obtains a separate answer label at each retained cue. This anchor-centered construction substantially lowers

Table 7: GraphRACE candidate funnel and calibration cost. Exhaustive is the number of unique readout events required to certify all 120 policies. Certification counts the additional events required to fully evaluate the ten minimax representatives after the pilot. Red. is the reduction relative to Exhaustive.

Model	Initial	After Arm	Medoids	Exhaustive	Pilot	Certification	GraphRACE / Red.
DeepSeek-1.5B	120	93	10	12,158	640	3,360	4,000 / 67.1%
Qwen3.5-2B	120	60	10	19,066	640	4,344	4,984 / 73.9%
Qwen3.5-4B	120	60	10	11,903	640	3,397	4,037 / 66.1%
Qwen3.5-9B	120	63	10	13,071	640	3,931	4,571 / 65.0%

the supervision cost before policy calibration, allowing SPARE to prepare the evidence needed for early-exit selection with fewer auxiliary generations.

B.5 GraphRACE Search Efficiency

After Net20 structural filtering of the 120 candidates, each model queries 640 distinct pilot events, with no further policy elimination. Complete-linkage clustering selects ten minimax representatives for full certification. Certification counts the additional events required to fully evaluate the ten minimax representatives after the pilot. No model required an additional distinct event during local refinement in the final Balanced-F5 run.

As shown in Table 7, GraphRACE reduces readout queries by 65.0–73.9% per model and 68.7% overall relative to exhaustive evaluation. Shared-event evaluation, risk-aware elimination, and behavioral consolidation thus identify the best risk-feasible early-exit policy under the specified calibration objectives using only a fraction of exhaustive calibration cost.

C Control Experiments and Baseline Sensitivity

C.1 Mechanism Controls

Full-Reasoning Readout runs under the shared 32K maximum generation length and applies SPARE's answer template after closing `</think>`. Table 8 reports results aggregated over the same 758 formal-test examples.

In Close-Only, an early-exit trigger is followed only by `</think>`, after which the model output is parsed using the same scorer. The bridge instruction and dataset-aware answer prefill are omitted.

Random Anchor replaces sample-adaptive anchor localization during probe training with one deterministic random reasoning-unit boundary per training example. It retains the same ordered pre-/post-boundary labels, linear probe, GraphRACE calibration, guided readout, and formal-test protocol as SPARE.

Table 8: Mechanism controls on Qwen3.5-4B and Qwen3.5-9B. Full-Reasoning Readout applies SPARE's guided answer template after complete reasoning under the 32K maximum generation length. In Close-Only, an early-exit trigger is followed only by </think>; the bridge instruction and dataset-aware answer prefill are omitted. Random Anchor replaces sample-adaptive anchor localization with a deterministic random reasoning-unit boundary during probe training. Acc. and CR are percentages; Tok. is the average effective completion-token count per example. Per-benchmark columns report Acc./CR, and Overall reports Acc./Tok./CR.

		MATH-500		**GPQA-D**		**AIME25**		**AIME26**		**Overall**		
Model	**Variant**	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	Tok. ↓	CR ↓
Qwen3.5-4B	Vanilla	90.60	100.00	78.28	100.00	50.00	100.00	60.00	100.00	84.56	11,183	100.00
	Full-Reasoning Readout	91.00	92.90	78.79	90.70	50.00	97.24	60.00	97.01	84.96	10,421	93.18
	Close-Only	17.40	61.00	8.59	72.28	0.00	76.30	0.00	77.74	13.72	7,441	66.54
	Random Anchor	93.80	62.29	74.75	59.98	43.33	62.74	66.67	57.93	85.75	6,866	61.40
	SPARE	94.40	60.52	77.27	71.73	50.00	76.15	70.00	77.56	87.20	7,392	66.10
Qwen3.5-9B	Vanilla	94.00	100.00	82.32	100.00	50.00	100.00	66.67	100.00	88.13	10,689	100.00
	Full-Reasoning Readout	93.80	91.84	82.83	91.13	56.67	97.41	66.67	96.65	88.39	9,901	92.63
	Close-Only	11.40	54.21	11.11	63.36	0.00	64.59	0.00	69.30	10.42	6,301	58.95
	Random Anchor	95.00	70.45	80.30	72.63	63.33	66.08	73.33	71.59	89.05	7,558	70.71
	SPARE	95.40	53.64	80.81	62.79	66.67	64.40	76.67	69.13	89.71	6,248	58.45

Full-Reasoning Readout changes accuracy by only $+0.40$ and $+0.26$ percentage points while retaining 93.18% and 92.63% of Vanilla generation cost on Qwen3.5-4B and Qwen3.5-9B, respectively. The guided answer interface alone therefore does not account for SPARE's accuracy–efficiency gains.

Close-Only achieves only 13.72% and 10.42% accuracy, with CR values of 66.54% and 58.95%, compared with 66.10% and 58.45% for SPARE on Qwen3.5-4B and Qwen3.5-9B, respectively. Among triggered examples, its readout accuracy is 3.80% and 3.88%, respectively. Thus, closing the thinking block alone does not reliably elicit a concise, scorable answer; the explicit bridge and dataset-aware prefill are necessary for dependable post-truncation readout. Together, the two controls separate the role of adaptive stopping from that of the answer interface: stopping provides the large token reduction, while guided readout converts the retained reasoning state into a reliable final answer.

Random Anchor still learns a useful notion of reasoning progress, but it does not reproduce SPARE's accuracy–efficiency trade-off. On Qwen3.5-4B, it saves 38.60% of tokens, compared with 33.90% for SPARE, but its overall accuracy is 1.45 percentage points lower. On Qwen3.5-9B, SPARE achieves both higher accuracy (89.71% versus 89.05%) and greater token saving (41.55% versus 29.29%). The consistent GPQA-D degradation under Random Anchor further indicates that generic progress supervision is less reliable than sample-adaptive sufficiency anchors.

C.2 LYNX Confidence Sweep

Among the evaluated baselines, LYNX is the only method with a directly adjustable confidence parameter. It trains a cue-level classifier to predict whether an answer read out at the current reasoning position would be correct. On an independent calibration set, split conformal prediction converts the classifier scores into prediction sets at target coverage c. LYNX exits only when the prediction set is the singleton positive class; otherwise reasoning continues, and samples without a reliable exit retain the full generation. Here, c is a target coverage level rather than the probability that an individual exit is correct. Larger c generally produces more conservative exits. The main text uses $c = 0.97$, while Table 9 reports all evaluated settings in $\{0.90, 0.95, 0.97\}$. These settings share the same probe, cue-level training data, and calibration labels; only the prespecified conformal confidence level changes.

Table 9: Complete LYNX confidence sweep under the matched post-truncation readout, scoring, and token-accounting protocol. Acc. denotes final-answer accuracy, Tok. denotes average effective completion tokens per example, and CR denotes aggregate completion-token cost relative to the corresponding Vanilla backbone. All values except Tok. are percentages.

		MATH-500		GPQA-D		AIME25		AIME26		Overall		
Model	**Method**	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	CR ↓	Acc. ↑	Tok. ↓	CR ↓
DeepSeek-1.5B	Vanilla	83.00	100.00	31.31	100.00	23.33	100.00	16.67	100.00	64.51	7,391	100.00
	LYNX $c = .90$	76.40	33.66	29.29	80.99	23.33	64.86	16.67	70.01	59.63	4,151	56.17
	LYNX $c = .95$	79.60	44.91	30.81	97.02	23.33	81.43	13.33	85.47	62.01	5,166	69.89
	LYNX $c = .97$	81.80	55.18	30.81	98.21	23.33	87.43	16.67	92.42	63.59	5,639	76.30
	SPARE	83.20	54.95	29.29	68.33	23.33	62.72	16.67	62.26	64.12	4,506	60.97
Qwen3.5-2B	Vanilla	65.80	100.00	55.56	100.00	20.00	100.00	10.00	100.00	59.10	18,987	100.00
	LYNX $c = .90$	58.60	19.67	47.98	48.04	13.33	38.96	3.33	40.05	51.85	5,315	27.99
	LYNX $c = .95$	68.40	26.82	53.54	61.25	13.33	55.12	3.33	46.77	59.76	7,009	36.91
	LYNX $c = .97$	73.40	31.37	55.05	67.56	10.00	59.17	3.33	53.96	63.32	7,967	41.96
	SPARE	82.20	50.06	55.56	56.38	26.67	66.30	13.33	60.35	70.32	10,074	53.06
Qwen3.5-4B	Vanilla	90.60	100.00	78.28	100.00	50.00	100.00	60.00	100.00	84.56	11,183	100.00
	LYNX $c = .90$	90.80	38.92	76.77	75.16	43.33	58.71	60.00	53.99	84.04	5,633	50.38
	LYNX $c = .95$	92.80	50.65	78.79	86.24	53.33	73.70	70.00	66.73	86.68	6,973	62.36
	LYNX $c = .97$	94.00	58.92	78.79	90.85	56.67	80.15	70.00	75.16	87.60	7,788	69.64
	SPARE	94.40	60.52	77.27	71.73	50.00	76.15	70.00	77.56	87.20	7,392	66.10
Qwen3.5-9B	Vanilla	94.00	100.00	82.32	100.00	50.00	100.00	66.67	100.00	88.13	10,689	100.00
	LYNX $c = .90$	85.40	30.26	71.72	56.61	40.00	45.67	43.33	42.74	78.36	4,240	39.67
	LYNX $c = .95$	91.60	43.40	78.79	75.30	60.00	64.82	70.00	57.81	86.15	5,877	54.98
	LYNX $c = .97$	93.00	51.83	80.81	82.91	66.67	74.00	66.67	69.93	87.73	6,800	63.62
	SPARE	95.40	53.64	80.81	62.79	66.67	64.40	76.67	69.13	89.71	6,248	58.45

At $c = 0.90$, LYNX exits aggressively but trails SPARE in overall accuracy by 4.49, 18.47, 3.16, and 11.35 percentage points across the four backbones. The degradation is especially visible on the Qwen AIME sets: for example, Qwen3.5-9B drops from SPARE's 66.67%/76.67% on AIME25/AIME26 to 40.00%/43.33%. Increasing c recovers accuracy but also raises token cost. At $c = 0.97$, LYNX has higher overall CR than SPARE on DeepSeek-1.5B, Qwen3.5-4B, and Qwen3.5-9B; Qwen3.5-2B remains less costly, but its overall accuracy is 7.00 percentage points lower.

D Detector Diagnostics

Readout-correctness discrimination. We evaluate the Qwen3.5-4B detector on 1,784 unique endpoints from the 200 MATH calibration problems in Section 3.1, using observed readout correctness as labels. With equal total weight per problem, the detector achieves an AUROC of 0.851 and an AP of 0.941 (weighted positive rate: 0.797). These results provide direct evidence that the classifier distinguishes reasoning prefixes yielding correct versus incorrect readouts, supporting its use as a sufficiency-related signal for adaptive early exit.

Progress-based aggregation. We examine how the detector score evolves over the formal-test reasoning traces. For sample i, we define the relative progress of reasoning-unit endpoint t as

$$r_{i,t} = \frac{e_{i,t}}{\max_u e_{i,u}}, \tag{18}$$

where $e_{i,t}$ is the endpoint token position. At each target progress in $\{10\%, 20\%, \ldots, 90\%\}$, we select the closest available endpoint for every sample and aggregate detector logits across all 758 examples. This retrospective analysis is not used for training, calibration, or policy selection.

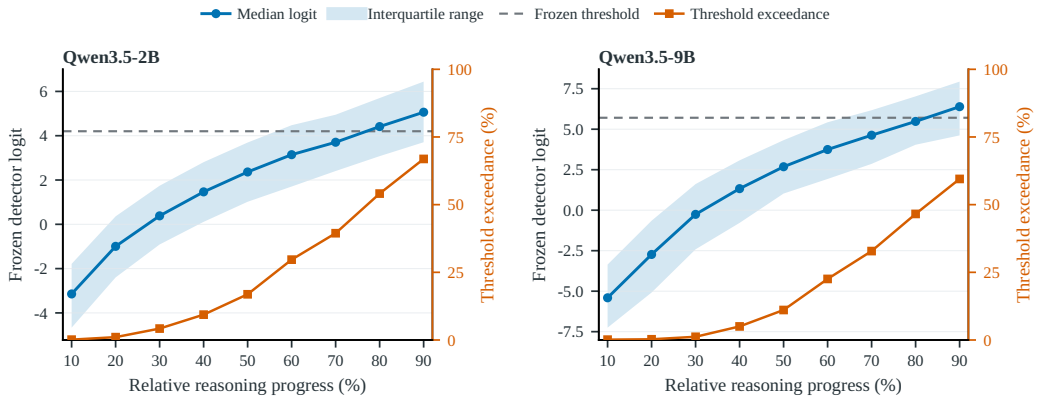


Figure 5: Detector behavior over relative reasoning progress. The blue curve and shaded region show the median and interquartile range of detector logits across formal-test examples, and the dashed line is the model-specific threshold selected on Math667. The orange curve reports the fraction of samples whose logit exceeds that threshold.

Observed score evolution. Figure 5 shows a clear aggregate progression from negative early-stage logits toward stronger stopping evidence later in the trace. For Qwen3.5-2B, the median logit rises from -3.14 at 10% progress to 5.06 at 90%, while threshold exceedance rises from 0.1% to 66.9%. Qwen3.5-9B exhibits the same trend, with median logits increasing from -5.41 to 6.39 and exceedance rising from 0.1% to 59.5%. Together, they indicate that the detector assigns progressively stronger stopping evidence as reasoning advances.

E Qualitative Examples

Figures 6 and 7 present two representative examples of our method from the completed formal-test outputs. The left column shows what occurs under uninterrupted generation, while the right column inserts SPARE's actual unit-level decisions into the reasoning prefix and follows the triggered answer readout. Text is verbatim except for spans explicitly marked as omitted.

Case 1: Qwen3.5-2B avoids a 32K-token interpretation loop. **Question:** Remmy wants to divide 10 by $\frac{2}{3}$. By what number should he multiply 10?

Online decision path: reasoning-unit endpoint $\longrightarrow$ probe logit z_t $\longrightarrow$ compare with frozen $\theta = 4.202$ $\longrightarrow$ update consecutive-hit counter $\longrightarrow$ **CONTINUE** or **EXIT** $\longrightarrow$ guided answer readout.

Vanilla (full generation)

[Earlier reasoning omitted.]
The thing is the reciprocal of the divisor. The divisor is $2/3$. The reciprocal is $3/2$. So he multiplies 10 by $3/2$.
Is it possible that the answer is 3? Some might say "To divide by $2/3$, multiply by $3/2$." Wait, if you have 10, and you need to divide it by $2/3$. You think: "Oh, I don't do division."

Generation continues past the sufficient state.

"I'll multiply 10 by 3." Then "divide by 2." $10 \times 3 = 30$. $30/2 = 15$. This sequence involves multiplication by 3 and a division by 2. The question asks for "what number should he multiply."
Is it possible the intended answer is just the reciprocal $3/2$? Or the integer 3? Could the answer be 1.5? Or $15/2$? Or $3/2$?

[28,000+ tokens of repeated reinterpretation omitted.]

Wait. I'll write 1.5. Okay. I'll write 1.5. Wait. I'll write $3/2$. Okay. I'll write $3/2$. Wait. I'll write 1.5. Okay. I'll write 1.5. Wait. I'll write $3/2$. . . .
Outcome: context limit reached; no scorable final answer.
Tokens: 32,768.

SPARE (online unit monitoring)

[Same earlier reasoning omitted.]
The thing is the reciprocal of the divisor. The divisor is $2/3$. The reciprocal is $3/2$.
So he multiplies 10 by $3/2$.

Unit 216: $z = 3.227 < \theta = 4.202 \Rightarrow$ streak $0/2$, **CONTINUE**.

Is it possible that the answer is 3? Some might say "To divide by $2/3$, multiply by $3/2$."

Units 217–218: $z = 0.406, 1.004 < \theta \Rightarrow$ streak remains $0/2$, **CONTINUE**.

Wait, if you have 10, and you need to divide it by $2/3$.

Unit 219: $z = 4.732 > \theta \Rightarrow$ streak $1/2$, **CONTINUE FOR CONFIRMATION**.

You think: "Oh, I don't do division."

Unit 220: $z = 6.377 > \theta \Rightarrow$ streak $2/2$, **EXIT**.

Guided answer readout:

Final answer: $\boxed{1.5}$ or $\boxed{\frac{3}{2}}$.

EvalScope parses $\frac{3}{2}$, matching the reference answer.
Outcome: correct.
Tokens: $3,877$ retained $+64$ readout $= 3,941$.
Net token reduction: 88.0%.

Figure 6: A Qwen3.5-2B formal-test trajectory. Both columns reproduce the same reasoning up to unit 220. Vanilla reopens the settled reciprocal and exhausts the 32K-token budget, whereas SPARE observes two consecutive above-threshold unit logits and invokes a short answer readout.

Case 2: Qwen3.5-9B preserves a correct extremal construction. **Question:** If $|ax^2 + bx + c| \leq 1$ for every $x \in [0, 1]$, maximize $|a| + |b| + |c|$.

Online decision path: reasoning-unit endpoint $\longrightarrow$ probe logit z_t $\longrightarrow$ compare with frozen $\theta = 5.707$ $\longrightarrow$ update consecutive-hit counter $\longrightarrow$ **CONTINUE** or **EXIT** $\longrightarrow$ guided answer readout.

Vanilla (full generation)

[Earlier derivation omitted.]
For $u = w = 1$, optimal is $B = -1$, implying $a = 8$, $b = -8$, $c = 1$. Thus $S = |a| + |b| + |c| = 17$.
Now consider general u, w. Let's recalculate b. $a+b+c = w$, $a/4+b/2+c = v$, and $c = u$. So $a = 2w - 4v$. Then $b = w - u - a = 4v - w - u$.
Wait, my previous derivation was $b = 4v - w - 3u$. Let me check that.

Generation continues after constructing the optimum.

Previous: $a = 2(u - 2v + w)$. This matches $2w - 4v + 2u$. Ah, the order of subtraction in my previous derivation was different. Let's restart carefully.
$c = u$. $P(1) = w = a+b+u$, so $a+b = w-u$. $P(1/2) = v = a/4 + b/2 + u$. Multiplying by four gives $a + 2b = 4v - 4u$. Therefore $b = 4v - w - 3u$.
My previous derivation was different. Where is the discrepancy?

[16,000+ tokens of repeated re-parameterization omitted.]

The trajectory repeatedly restarts the coefficient analysis and moves away from the already valid construction $a = 8, b = -8, c = 1$.
Outcome: incorrect answer 4.
Tokens: 32,768.

SPARE (online unit monitoring)

[Same earlier derivation omitted.]
For $u = w = 1$, optimal is $B = -1$, implying $a = 8$, $b = -8$, $c = 1$. Thus $S = |a| + |b| + |c| = 17$.
Now consider general u, w. Let's recalculate b. $a+b+c = w$, $a/4+b/2+c = v$, and $c = u$.

Units 591–592: $z = 1.696, 3.073 < \theta = 5.707 \Rightarrow$ streak $0/2$, **CONTINUE**.

So $a = 2w - 4v$. Then $b = w - u - a = 4v - w - u$.

Unit 593: $z = 0.559 < \theta \Rightarrow$ streak remains $0/2$, **CONTINUE**.

Wait, my previous derivation was $b = 4v - w - 3u$.

Unit 594: $z = 5.742 > \theta \Rightarrow$ streak $1/2$, **CONTINUE FOR CONFIRMATION**.

Let me check that.

Unit 595: $z = 7.120 > \theta \Rightarrow$ streak $2/2$, **EXIT**.

Guided answer readout:

Final answer: $\boxed{17}$.

The readout preserves the extremal construction already present in the retained prefix.
Outcome: correct.
Tokens: $16,265$ retained $+5$ readout $= 16,270$.
Net token reduction: 50.3%.

Shared lesson. In both examples, the decisive information is already present before the full trajectory ends. The calibrated persistence rule detects two consecutive sufficient states, allowing SPARE to preserve the available answer before continued generation enters an interpretation loop or destabilizes a valid construction.

Figure 7: A Qwen3.5-9B formal-test trajectory. The model has already constructed the extremizer yielding 17 before both columns begin. The uninterrupted trajectory reopens the derivation and drifts to an incorrect parsed answer, while SPARE requires two consecutive above-threshold units before preserving the available solution through its guided readout.