MITIGATING EFFICIENCY-INDUCED PREMATURE STOPPING IN KNOWLEDGE-INTENSIVE AGENTIC REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Agentic reasoning is essential for knowledge-intensive tasks, where models must iteratively decide whether to continue reasoning, call a tool to retrieve external evidence, or produce an answer. However, these models often engage in lengthy deliberation before taking an action, substantially increasing token cost and latency without making meaningful progress. Existing test-time efficient methods mainly rely on efficiency-oriented early stopping, which may terminate reasoning before a necessary tool call and lead to severe performance degradation. To address this efficiency-induced premature termination, we propose OPERA, a lightweight test-time framework that uses only the reasoning model to reduce redundant deliberation while preserving productive reasoning. OPERA uses hierarchical action probing to assess whether the model is ready to take the next action, and identifies potential verbosity on-the-fly to estimate when to probe. It terminates reasoning only when probes consistently support the same action. Extensive experiments across knowledge-intensive benchmarks show that OPERA consistently reduces token usage while preserving or improving answer accuracy and more effectively mitigates premature termination than baselines.

1 INTRODUCTION

Language models have demonstrated strong reasoning abilities in mathematics and coding, yet they continue to struggle with knowledge-intensive tasks, where the information required to answer a question is not provided in the input and must be retrieved from external sources (Wu et al., 2025; Asai et al., 2024). Agentic reasoning is essential in this setting because it makes information retrieval an active part of the reasoning process rather than a fixed preprocessing step. An agent can determine when external evidence is needed, formulate an appropriate search query, and incorporate the retrieved information (Chen et al., 2025; Zhang et al., 2026; Jiang et al., 2023). This adaptive interaction becomes essential for knowledge-intensive tasks, where evidence must be gathered and integrated across multiple retrieval rounds. Therefore, agentic reasoning enables models to progressively acquire the information needed to produce well-supported answers (Jin et al., 2025; Yao et al., 2023; Lewis et al., 2020).

An important challenge in applying agentic reasoning to knowledge-intensive tasks is the verbosity of reasoning that occurs before a tool call (Song et al., 2026; Qian et al., 2026; Su et al., 2026). As shown by the agent reasoning example in Figure 1, models spend substantial computation repeatedly checking the context, and reconsidering what information to search for. Much of this deliberation produces little informative progress while delaying the next search action or final answer (Chen et al., 2026; Sui et al., 2026). The problem is particularly significant in smaller models, which are often used as low-cost search workers in deep-research style multi-agent systems to gather information in parallel (Yang et al., 2026b; Yi et al., 2026; An et al., 2026). Because the effectiveness of such systems depends on the agentic capabilities of workers to call tools timely (Li et al., 2025a; Xu et al., 2025), improving the efficiency of agentic reasoning on knowledge-intensive tasks is critical.

As depicted in the efficient reasoning case in Figure 1, existing test-time efficiency optimization frameworks typically employ early-stopping mechanisms that terminate an ongoing reasoning trajectory (Snell et al., 2025; Muennighoff et al., 2025; Sui et al., 2025). These mechanisms are often

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.

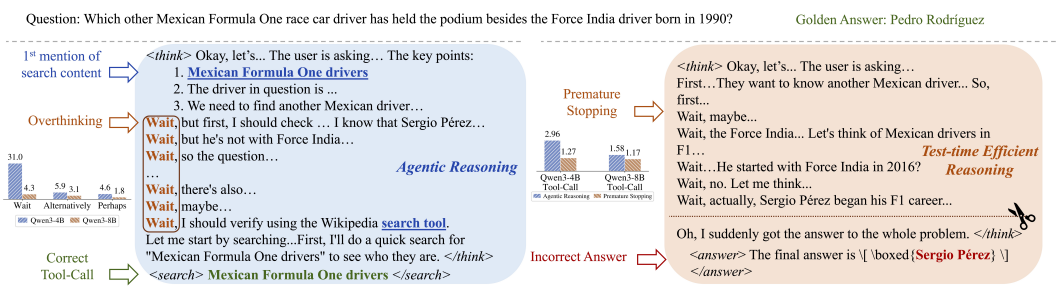


Figure 1: Agentic reasoning on knowledge-intensive tasks often spends excessive tokens overthinking instead of calling tools, while test-time efficiency frameworks may prematurely terminate productive reasoning before the model retrieves the required information, leading to incorrect answers.

supported either by pretrained auxiliary models to evaluate intermediate reasoning states (Sun et al., 2026; Huang et al., 2026; Li et al., 2026a), or by periodically probing the reasoning model's current answers to measure answer consistency as a stopping signal (Fu et al., 2026; Yang et al., 2026a; Liu & Wang, 2025). Both approaches introduce extraneous costs, through increased deployment complexity or additional computation that may not be reflected in reported efficiency gains. More importantly, because these methods prioritize early termination, they may stop a trajectory before it reaches the decision to call the correct tool. Such premature stopping can cause severe performance degradation on knowledge-intensive tasks, where an answer produced without the necessary external evidence is unreliable (Zhang et al., 2024). Therefore, the central challenge is not merely to stop reasoning earlier, but to distinguish genuinely redundant deliberation from productive reasoning that is still progressing toward a necessary tool call or a well-supported answer.

To address this challenge, we propose On-the-Fly Probing of Agentic Actions for Efficient Reasoning (OPERA), a lightweight test-time framework that relies solely on the reasoning model, without auxiliary models or classifiers to assess its reasoning progress. OPERA uses consensus-based hierarchical action probing to determine whether the current trajectory can be terminated without disrupting meaningful progress, and learns decoding patterns of potential verbosity on-the-fly to estimate when probing is most beneficial and efficient. By evaluating whether the model is ready to call a tool or produce a final answer, this probing mechanism directly supports the agentic reasoning. OPERA stops the reasoning process only when action probes from multiple probe points within a rolling monitor window consistently indicate the same next action. In this way, OPERA improves the test-time efficiency of knowledge-intensive agents while preserving their ability to retrieve the information needed for accurate answers. We summarize our contributions as follows:

- We design Consensus-based Hierarchical Action Probing, which detects and verifies the model's intended action and corresponding content across multiple probe points to improve the efficiency of agentic reasoning while mitigating premature stopping.
- We further introduce On-the-Fly Probe-Timing Estimation, which learns verbosity patterns at test time and determines when action probing is most beneficial, reducing unnecessary probe tokens while further lowering the risk of premature stopping.
- We conduct extensive experiments across a diverse set of knowledge-intensive benchmarks. OPERA notably reduces reasoning-token usage while preserving or improving answer accuracy, while substantially reducing the premature termination observed in efficiency-oriented baselines.

2 Related Work

2.1 Agentic Reasoning for Knowledge-Intensive Task

Agentic reasoning enables language models to actively interact with external tools while solving complex problems, extending reasoning beyond a fixed sequence of internal computations (Schick et al., 2023; Qin et al., 2023). This capability is particularly important for knowledge-intensive tasks,

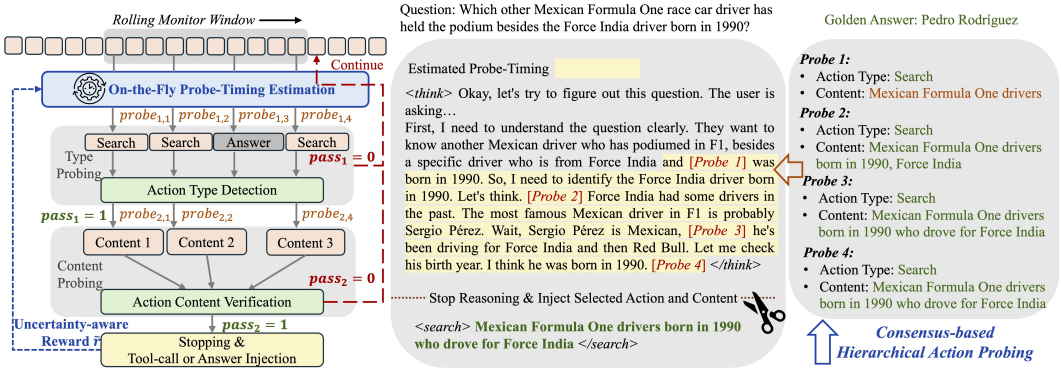


Figure 2: OPERA uses consensus-based hierarchical action probing with an on-the-fly probe-timing estimation to dynamically determine when to probe the ongoing generation. At each estimated probe time, OPERA sequentially performs action type detection and action content verification, stopping the generation and injecting the selected action only when both stages pass.

where the information required to answer a question may be unavailable in the initial context and must be retrieved and integrated across multiple steps. ReAct established the reasoning-and-action paradigm by interleaving reasonings with tool interactions (Yao et al., 2023). More recent works, including Search-R1 and Search-o1, further develop multi-round search capabilities for knowledge-intensive reasoning (Jin et al., 2025; Li et al., 2025b; Dong et al., 2026; Li et al., 2026b). Their evaluations across knowledge-intensive benchmarks highlight both the importance of processing complex external information, and the central role of agentic reasoning in acquiring it. However, existing methods primarily focus on improving search policies and answer accuracy, leaving the inference efficiency of agentic reasoning less explored.

2.2 Test-time Efficient Reasoning

Existing methods improve test-time reasoning efficiency by terminating a reasoning trajectory once its intermediate state appears sufficient. One group relies on pretrained auxiliary models, such as process reward models, classifiers, or embedding models, to evaluate reasoning progress and determine when to stop (Sun et al., 2026; Zhang et al., 2025; Afzal et al., 2025; Kuhn et al., 2023; Farquhar et al., 2024). Another group periodically probes the reasoning model at fixed intervals and uses answer confidence or self-consistency as the stopping signal (Fu et al., 2026; Laaouach, 2025; Li et al., 2024; Schuster et al., 2022). However, these methods are primarily efficiency-oriented and may terminate a productive trajectory before the model reaches a necessary tool call or gathers sufficient external evidence. Such efficiency-induced premature stopping can substantially reduce accuracy on knowledge-intensive tasks. OPERA addresses this limitation by making stopping decisions aware of the model's next action, thereby reducing genuinely redundant reasoning while preserving the information-gathering process required for accurate answers.

3 On-the-Fly Probing of Agentic Actions for Efficient Reasoning

Knowledge-intensive tasks rely on information not provided in the input and therefore require agentic reasoning, in which the model takes actions across multiple rounds to retrieve external evidence and produce the final answer. To formalize this agentic reasoning, let x denote the input knowledge-intensive task and π_θ the reasoning model. At each agentic round, the model generates a thinking span y based on the history h of previous reasoning and retrieved evidence:

$$y = (y_1, \ldots, y_T), \quad y_t \sim \pi_\theta\left(\cdot \mid x, h, y_{<t}\right), \tag{1}$$

where T is the total number of tokens in the thinking span. After generating y, the model emits one of two agentic actions: (i) a *search* action submits a query to the retriever, then the returned evidence is added to the history; or (ii) an *answer* action produces the final response and terminates the reasoning. However, during agentic reasoning, the model may have already determined its next

action yet continue redundant deliberation. OPERA detects and verifies this inefficiency through on-the-fly probing of agentic actions, then advances the generation directly to the intended action. As shown in Figure 2, OPERA consists of the following modules:

- **Consensus-based Hierarchical Action Probing**: At each probe time, OPERA sequentially verifies the type and content of the model's intended next action. It stops the ongoing reasoning only when both probing stages pass their respective consistency assessments.
- **On-the-Fly Probe-Timing Estimation**: OPERA uses entropy-aware reward signals to learn patterns of redundant deliberation on-the-fly and estimate when action probing is most likely to enable beneficial stopping.

3.1 Consensus-based Hierarchical Action Probing

To determine whether stopping can save computation without causing premature termination, OPERA probes the model's intended agentic action and evaluates its consistency across multiple probe points. Because agentic reasoning on knowledge-intensive tasks may have occasional inconsistencies, as illustrated by the case study in Figure 2, OPERA decomposes action probing into two hierarchical stages and applies distinct consistency assessments: (i) action type detection, which determines whether the model intends to search or answer; and (ii) action content verification, which checks the corresponding contents of search query or final answer.

Stage 1: Action type detection. In order to reduce sensitivity to token-level fluctuations, OPERA maintains a rolling window that captures a contiguous and sufficiently broad view of the recent reasoning. Within the rolling window, OPERA places C probes at fixed intervals with probe points $\{c_1, \ldots, c_C\}$. At each probe point, OPERA appends a short action type probe prefix ρ_{type} to the current reasoning:

$$p_{1,j} \sim \pi_\theta\big(\cdot \mid x, h, y_{\le c_j} \oplus \rho_{\text{type}}\big), \qquad p_{1,j} \in \mathcal{A} \cup \{\perp\}, \quad j = 1, \ldots, C. \tag{2}$$

where $\mathcal{A} \subseteq \{\textit{search}, \textit{answer}\}$ is the set of agentic actions, and $\perp$ denotes an unrecognized action type. OPERA uses majority voting to identify the model's dominant action intention $a^\star$ and tolerate occasional inconsistent predictions:

$$a^\star = \arg\max_{a \in \mathcal{A}} \sum_{j=1}^{C} \mathbf{1}[p_{1,j} = a], \qquad \text{pass}_1 = \mathbf{1}\left[\frac{1}{C}\sum_{j=1}^{C} \mathbf{1}[p_{1,j} = a^\star] \ge \eta\right], \quad \eta > \tfrac{1}{2}. \tag{3}$$

where $p_{1,j}$ denotes the action type probe at probe point j, and η is the majority voting threshold. If $\text{Pass}_1 = 0$, meaning that no action reaches the voting threshold, the stopping decision is rejected without performing content probing. This lightweight action type probing stage filters ambiguous action intentions, which indicate that reasoning is still ongoing, thereby mitigating premature stopping while avoiding the unnecessary cost of further content probing. Otherwise, if $\text{Pass}_1 = 1$, the intended action types are sufficiently consistent across probe points. Then we proceed with the probes supporting the selected action $a^\star$: $J^\star = \{j : p_{1,j} = a^\star\}$ to action content verification.

Stage 2: Action content verification. Based on the action type detection results, we construct an action-conditioned content probe prefix $\rho_{\text{content}} = \rho_{\text{content}}(a^\star)$ that specifies the selected action $a^\star$. At each type-supporting probe point, OPERA appends this prefix and continues decoding to obtain a candidate action content:

$$p_{2,j} \sim \pi_\theta\big(\cdot \mid x, h, y_{\le c_j} \oplus \rho_{\text{content}}\big), \quad j \in J^\star, \tag{4}$$

where $p_{2,j}$ denotes the action content probe at type-supporting probe point j. Specifically, $p_{2,j}$ is a candidate search query when $a^\star = \textit{search}$, and a candidate answer when $a^\star = \textit{answer}$.

OPERA then groups candidate contents whose pairwise action content similarity exceeds the threshold τ and selects the content $u^\star$ from the largest cluster:

$$s_j = \sum_{i \in J^\star} \mathbf{1}[\text{sim}(p_{2,j}, p_{2,i}) \ge \tau], \quad j \in J^\star, \qquad j^\star \in \arg\max_{j \in J^\star} s_j, \quad u^\star = p_{2,j^\star}. \tag{5}$$

where s_j is the number of action content probes similar to $p_{2,j}$. Content verification passes only when the largest similarity-consistent cluster contains at least an η fraction of the content probes:

$$\text{pass}_2 = \mathbf{1}\left[\frac{s_{j^\star}}{|J^\star|} \geq \eta\right]. \tag{6}$$

When $\text{pass}_2 = 1$, indicating that both the intended action type and content are consistent across probe points, OPERA stops the reasoning and injects the selected action type and content into the main reasoning trajectory:

$$y \leftarrow y_{\leq t} \oplus (a^\star, u^\star). \tag{7}$$

Consequently, a search action ends the current round and returns the retrieved evidence to the model, whereas an answer action terminates the multi-round agentic reasoning. When $\text{pass}_2 = 0$, the model is considered not ready to stop and its reasoning may still be progressing. Therefore, OPERA discards the action probes and allows the reasoning to continue unchanged. In this way, OPERA mitigates premature stopping while improving the efficiency of agentic reasoning on knowledge-intensive tasks by preserving genuinely productive reasoning and terminating redundant repetition.

3.2 On-the-Fly Probe-Timing Estimation

Applying probing at fixed intervals overlooks variations in intermediate reasoning progress, potentially introducing unnecessary computation and increasing the risk of premature stopping. Instead, OPERA learns verbosity patterns on-the-fly and estimates effective probe-timing, activating action probing only when potential redundant deliberation is detected.

OPERA learns decoding patterns of redundant deliberation on-the-fly. As ground-truth answers for the input tasks are unavailable during inference, OPERA cannot use final-answer correctness as a direct learning signal. Therefore, it uses an uncertainty-aware reward with two parts: (i) an effectiveness reward evaluates whether an estimated probe time enables successful stopping while accounting for the probing cost; (ii) an uncertainty shifting weight measures how different the reasoning of the current monitor window is from previous monitor windows, allowing more informative probings to have a stronger influence on on-the-fly learning.

After each probe-timing estimation, OPERA measures its effectiveness based on whether the subsequent action probing successfully stops redundant deliberation. Let b denote the number of probe tokens generated, which is normalized by the maximum probing budget $B_{\max}$:

$$r = w_1 * \text{pass}_2 - w_2 \, \min\left(1, \frac{b}{B_{\max}}\right). \tag{8}$$

If action probing successfully stops redundant deliberation, the estimated probe time receives a positive reward; otherwise, it is penalized for the consumed probe tokens. This encourages reliable probe-timing estimation with minimal probing cost.

To measure the uncertainty shifting of the resulting action probing, OPERA uses top-K token-level Shannon entropy to assign greater importance to probe-timing estimations with meaningful changes in the ongoing reasoning:

$$H_t = -\sum_{v \in V_t} p_t(v) \log p_t(v). \tag{9}$$

where V_t is the set of top-K candidate tokens when generating the t-th token, and $p_t(v)$ is the probability of candidate token v. OPERA then normalizes it with the full vocabulary size to improve comparability across models. Let $\widehat{H}$ and σ be the mean entropy and standard deviation of the normalized entropy within the monitoring window. The bounded deviation score measures how much the token-level uncertainty in the current monitoring window shifts from that in previous monitoring windows, and the resulting weighted reward is defined as follows:

$$\Delta = \min\left(1, \frac{\left|\widehat{H}^{\text{current}} - \widehat{H}^{\text{history}}\right|}{\sigma^{\text{history}}}\right), \qquad \widetilde{r} = r(1 + \Delta). \tag{10}$$

Consequently, effective probe times with more uncertainty shifts in the ongoing reasoning have a stronger influence on the verbosity-pattern learning and probe-timing estimation.

At each reasoning token, OPERA estimates the next probe-timing from a discrete set D: an estimation of 0 triggers the action probing immediately, whereas a positive estimation d allows the model to generate d additional tokens before probing. For each candidate estimation $d \in D$, let $n(d)$ denote the number of times d is selected, and $\widehat{\mu}(d)$ the mean of N most recent rewards. OPERA then selects the probe-timing with the highest estimated value of Garivier & Moulines (2011):

$$d \in \arg\max_{d \in D} \left[\widehat{\mu}(d) + \sqrt{\frac{\log(N+1)}{n(d)}} \right]. \tag{11}$$

In this way, OPERA adaptively estimates when to probe, reducing unnecessary probe tokens and further mitigating the risk of premature stopping.

4 Experiments

4.1 Experimental Settings

Datasets. To evaluate how effectively OPERA mitigates premature stopping in efficient agentic reasoning, we conduct experiments on a diverse set of knowledge-intensive tasks: (i) **Single-hop QA**: *Natural Questions* (NQ) (Kwiatkowski et al., 2019), an open-domain QA benchmark that requires retrieving external evidence to answer each question; (ii) **Multi-hop QA**: *HotpotQA* (Yang et al., 2018), *MuSiQue* (Trivedi et al., 2022), and *Bamboogle* (Press et al., 2023), which require multiple rounds of retrieval and evidence aggregation across search steps; and (iii) **Complex Reasoning**: *GPQA-Diamond* (GPQA-D) (Rein et al., 2024), the most challenging subset of a PhD-level multiple-choice benchmark, which requires complex scientific reasoning.

Baselines. We use vanilla agentic reasoning without any early-stopping as a reference and compare OPERA with recently proposed test-time methods for efficient reasoning: (i) **Agentic Reasoning** (Li et al., 2025b; Jin et al., 2025): the vanilla multi-round agentic reasoning that interleaves reasoning with search-tool calls and retrieved information until the final answer, without applying any early-stopping mechanism. We use it as a reference for measuring changes in answer accuracy and token usage; (ii) **Dynasor** (Fu et al., 2026): an early-stopping method that periodically probes the model for a provisional answer and terminates reasoning once the predicted answers become sufficiently consistent; (iii) **HALT-CoT** (Laaouach, 2025): an entropy-based early-stopping method that monitors the model's answer distribution during reasoning and terminates generation once its entropy falls below a predefined threshold; and (iv) **REFRAIN** (Sun et al., 2026): a reasoning-suppression method that uses MiniLM sentence embeddings to identify redundant reasoning and a controller to adapt the suppression threshold to each question.

To ensure fair comparisons, OPERA and all baselines use the same agent prompt and decoding configuration. Following common practice (Jin et al., 2025; Trivedi et al., 2023), all methods also use the same BM25-based search tool built on the 2018 Wikipedia dump to retrieve the top-3 passages.

Hyperparameters. We evaluate Qwen3-4B-Thinking-2507, the latest Qwen3-4B variant, and Qwen3-8B in thinking mode (Yang et al., 2025). Both models are served with the inference engine vLLM (Kwon et al., 2023), with the temperature set to 0.7 and top-p to 1.0. OPERA uses a 256-token rolling window with $C{=}4$ uniformly spaced action probes and a probing token budget of $B_{\max}{=}22$ (action type: 2; action content: 20). The probe-timing estimation set is $D{=}\{0, 32, 64, 128, 256\}$. All settings are fixed across benchmarks without dataset-specific tuning to evaluate the generalizability of each method. Detailed implementation settings are provided in Appendix A.

4.2 Main Results

Table 1 presents the main comparisons between OPERA and the efficiency-oriented baselines. We measure answer quality using exact match (EM) and efficiency using the average total number of generated tokens per question (Avg. L). All of the reasoning tokens and decoded probe tokens are included to reflect the complete test-time decoding cost. We summarize key observations as follows:

Improve the effectiveness and efficiency of agentic reasoning. OPERA maintains the average EM of vanilla agentic reasoning, with 34.90% vs. 34.40% on Qwen3-4B and 33.00% vs. 31.93%

Table 1: Evaluation results of OPERA with vanilla agentic reasoning and test-time efficient methods across knowledge-intensive benchmarks. We report exact match (EM) and mean total decoded tokens (L) in thousands (K). The highest EM is **bolded**.

Method	Single-hop QA		Multi-hop QA						Complex Reas.	
	NQ		HotpotQA		MuSiQue		Bamboogle		GPQA-D	
	EM ↑	Avg. L ↓	EM ↑	Avg. L ↓	EM ↑	Avg. L ↓	EM ↑	Avg. L ↓	EM ↑	Avg. L ↓
Qwen3-4B-Thinking-2507										
Agentic Reas.	29.64	2.15	39.84	2.42	11.29	3.32	39.20	2.38	52.02	5.58
+ Dynasor	27.56	1.09	32.92	1.10	7.74	1.80	33.60	1.43	38.38	0.76
+ HALT-CoT	25.82	0.60	27.94	0.65	6.37	1.19	23.20	0.62	38.89	1.80
+ REFRAIN	28.84	1.09	33.60	1.12	8.81	1.61	26.40	1.25	51.52	4.75
+ **OPERA**	**30.36**	1.91	**41.97**	2.04	**12.04**	2.93	**37.60**	2.15	**52.53**	4.98
Qwen3-8B										
Agentic Reas.	25.43	0.62	35.40	0.87	11.50	1.16	36.80	0.69	50.51	4.73
+ Dynasor	25.43	0.60	31.94	0.62	9.18	0.76	33.60	0.58	34.34	0.49
+ HALT-CoT	22.99	0.26	26.13	0.30	5.92	0.43	24.80	0.23	38.89	0.98
+ REFRAIN	25.07	0.56	31.75	0.60	9.72	0.87	37.60	0.55	**55.05**	4.15
+ **OPERA**	**25.93**	0.56	**36.06**	0.65	**10.76**	0.97	**39.20**	0.59	53.03	3.11

on Qwen3-8B, while achieving relative decoded length reductions of 9.66-15.70% and 9.68-34.25% across benchmarks. Among the test-time efficient reasoning methods, OPERA achieves the highest aggregate EM, outperforming the strongest baseline by 5.07% (34.90% vs. 29.83%) with Qwen3-4B and 1.16% (33.00% vs. 31.84%) with Qwen3-8B. These results demonstrate that OPERA preserves agentic reasoning quality while notably reducing reasoning cost.

Mitigate premature stopping at test time without auxiliary evaluation models. We further examine the changes in EM and decoded length relative to vanilla agentic reasoning. On HotpotQA with Qwen3-4B, OPERA reduces decoded length by 15.70% relative to vanilla agentic reasoning while improving EM by 2.13%; in comparison, Dynasor has a 54.55% relative decoded-length reduction but decreases EM by 6.92%. On GPQA-Diamond with Qwen3-8B, OPERA achieves a 34.25% relative decoded-length reduction while improving EM by 2.52%, whereas Dynasor reduces decoded length by 89.64% at the cost of a 16.17% EM drop. OPERA also outperforms the auxiliary model-aided REFRAIN in most model-benchmark settings. For example, on HotpotQA with Qwen3-8B, OPERA reduces decoded length by 25.29% while improving EM by 0.66%, whereas REFRAIN has a 31.03% relative decoded-length reduction but decreases EM by 3.65%. The more moderate reductions in decoded length achieved by OPERA compared with efficiency-oriented baselines further support its effectiveness in mitigating premature stopping. The substantially larger reductions achieved by the baselines are accompanied by notable EM degradation, suggesting that their aggressive stopping mechanisms may truncate productive reasoning before a necessary tool call or well-supported final answer is reached. In contrast, OPERA reduces redundant reasoning while preserving or improving answer quality and lowering the overall test-time token cost.

4.3 Reasoning Quality Analysis

Reduce repetitive uncertainty during reasoning. We then measure the average frequency of common uncertainty keywords to examine how OPERA reduces redundant deliberation. As shown in Figure 3, OPERA consistently reduces all uncertainty keywords across both model sizes. Significantly, for Qwen3-4B, the average occurrence of *Wait* decreases from 31.0 to 23.1 per question. A similar pattern

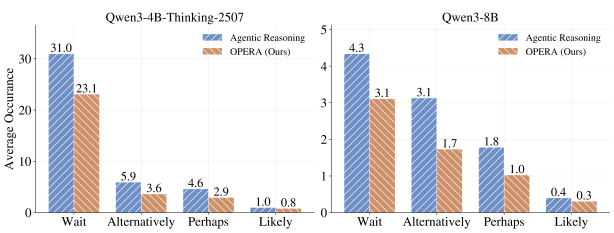


Figure 3: Average occurrences of uncertainty words.

appears for Qwen3-8B, where it decreases from 4.3 to 3.1. Together with the accuracy improvements in Table 1, these reductions suggest that OPERA removes repetitive reconsideration while preserving productive reasoning rather than inducing premature stopping.

Improve retrieval quality. To further analyze how OPERA contributes to its performance gains, we evaluate retrieval quality using cumulative golden retrieval at each tool-call round. We identify supporting retrievals using benchmark-provided supporting-fact annotations when available and retrieved passages that include the correct answer. As shown in Figure 4, OPERA achieves the highest cumulative golden retrieval at every round across benchmarks. By the final round, OPERA reaches 58.46% versus 55.33% for vanilla agentic reasoning on HotpotQA, and 57.32% versus 56.82% on NQ. The improvement is particularly significant on GPQA-D, where OPERA reaches 4.04%, four times the 1.01% obtained by vanilla agentic reasoning. These results suggest that OPERA not only removes redundant deliberation but also improves the model's ability to issue effective search queries and acquire answer-relevant evidence.

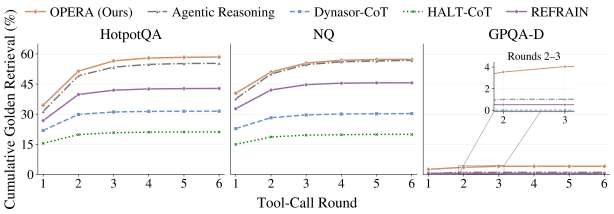


Figure 4: Average cumulative golden retrieval.

4.4 Probing Behavior Analysis

Task-adaptive action probing token usage. To examine the computational cost of action probing, we compare OPERA with Dynasor of both EM and the fraction of total decoded tokens spent on probes. As shown in Figure 5, OPERA uses substantially fewer probe tokens on retrieval-oriented tasks while achieving higher accuracy. On NQ, probe tokens account for only 8.2% of total computation with Qwen3-4B and 7.0% with Qwen3-8B, compared with 32.8% and 15.2% for Dynasor. In contrast, OPERA allocates more probe tokens on GPQA-D with Qwen3-8B, at 19.0% versus 10.6% for Dynasor, while improving EM by 18.69%. These results show that OPERA does not minimize probing indiscriminately. Instead, it limits probing when reasoning progress can be assessed efficiently and allocates more action probing to complex tasks where intended actions are less consistent, thereby reducing the risk of premature stopping.

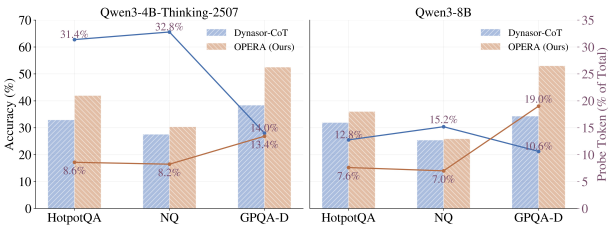


Figure 5: Comparison of probing cost. **Line**: probe tokens (% of total); **Bar**: exact-match accuracy (%).

Task-adaptive probe-timing estimations. We further analyze the patterns produced by OPERA's on-the-fly probe-timing estimation. As shown in Figure 6, the selected probe timings vary across benchmarks and model sizes rather than converging to a single fixed choice. HotpotQA and NQ favor short and intermediate timings, with 64 tokens as the most frequently selected option in aggregate across both benchmarks and model sizes. In contrast, GPQA-D favors longer timings, with 256 tokens selected most frequently by Qwen3-4B and 128 tokens by Qwen3-8B. Although the overall distributions remain broad, these task-dependent shifts indicate that OPERA's on-the-fly probe-timing mechanism adapts to different reasoning and verbosity

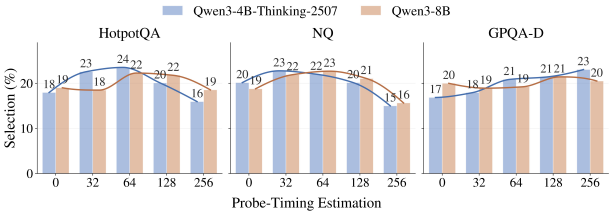


Figure 6: Distribution of probe-timing estimations.

Table 2: Ablation results for OPERA and comparison results with agentic-refined Dynasor.

Method	HotpotQA		NQ		GPQA-D	
	EM	L	EM	L	EM	L
Qwen3-4B-Thinking-2507						
Dynasor-Agentic	39.68	3.42	28.25	3.15	50.00	6.69
OPERA (Ours)	**41.97**	2.04	**30.36**	1.91	**52.53**	4.98
w/o Hierarchical Probing	40.24	2.06	27.95	1.86	52.02	5.18
w/o Probe-Timing Estimation	40.95	2.00	29.78	1.86	46.46	5.05
Qwen3-8B						
Dynasor-Agentic	35.37	0.89	26.15	0.83	49.49	4.08
OPERA (Ours)	**36.06**	0.65	25.93	0.56	**53.03**	3.11
w/o Hierarchical Probing	35.53	0.65	25.57	0.55	50.51	3.03
w/o Probe-Timing Estimation	35.83	0.64	**26.43**	0.55	49.49	2.88

patterns. It allows longer uninterrupted reasoning on complex problems while checking progress more frequently on retrieval-oriented tasks.

4.5 Ablation Analysis

To assess the contributions of Consensus-based Hierarchical Action Probing and On-the-Fly Probe-Timing Estimation, we conduct ablation analyses by replacing them with one-stage action probing and fixed probing schedules. The ablation results in Table 2 show that both components contribute to mitigating premature stopping: (i) compared with one-stage probing, hierarchical action probing improves the average EM by 1.55% with Qwen3-4B and 1.14% with Qwen3-8B; (ii) compared with fixed probe schedules, on-the-fly probe-timing estimation improves the average EM by 2.56% and 1.09%. The benefit of adaptive probe timing is particularly significant on GPQA-D, where OPERA improves EM over the fixed-schedule variant by 6.07% with Qwen3-4B and 3.54% with Qwen3-8B. These accuracy improvements indicate that hierarchical consensus prevents incomplete or unreliable action predictions from prematurely terminating reasoning, while probe-timing estimation identifies genuinely redundant deliberation and determines when action probing is most beneficial.

4.6 Comparison Analysis with Dynasor-Agentic

To further evaluate OPERA's efficiency and ability to mitigate premature stopping, we adapt Dynasor by extending its answer-only probes to action probes that predict a search query or a final answer. Each probed action is assessed using Dynasor's original consistency and certainty criteria. As shown in Table 2, compared with Dynasor-Agentic, OPERA improves the average EM by 2.31% while achieving a 32.65% relative reduction in average decoded length with Qwen3-4B. With Qwen3-8B, OPERA improves the average EM by 1.34% while achieving a 25.52% relative reduction in average decoded length. These results show that simply integrating agentic actions into early-stopping assessment is insufficient. OPERA additionally identifies genuinely redundant deliberation through on-the-fly probe-timing estimation and verifies candidate actions through hierarchical consensus, enabling more accurate and efficient agentic reasoning.

5 Conclusion

We presented OPERA, a lightweight test-time framework that utilizes Consensus-based Hierarchical Action Probing and On-the-Fly Probe-Timing Estimation to improve the efficiency of agentic reasoning without training or auxiliary models. Across challenging knowledge-intensive benchmarks, OPERA consistently reduces inference costs while preserving or improving answer accuracy. Compared with existing efficiency-oriented methods, it effectively mitigates premature stopping and reduces token usage without disrupting productive reasoning. Further analysis indicates that OPERA improves reasoning quality by removing genuinely redundant deliberation and enabling more timely retrieval of supporting evidence. These findings highlight test-time efficiency optimization as a promising direction for developing more efficient and reliable agentic reasoning systems.

AI USE STATEMENT

In this work, we used generative AI tools to assist with implementing methods, polishing manuscript and proof writing, and finding related work. We did not use generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean or reformat datasets, support qualitative or thematic data analysis, or interpret experimental results. We have reviewed all AI-assisted implementations and writing. LLM-generated code was verified and tested for correctness. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the information necessary to reproduce our experiments throughout the paper and supplementary materials. The datasets, reasoning models, evaluation metrics, and compared baselines are described in Section 4.1. Detailed inference and probing hyperparameters, retrieval configuration, the tool-integrated system prompt, and action probe prefixes are provided in Appendix A. The complete codebase and experiment configurations will be made publicly available upon publication under the MIT License.

REFERENCES

Anum Afzal, Florian Matthes, Gal Chechik, and Yftah Ziser. Knowing before saying: Llm representations encode information about chain-of-thought success before completion. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 12791–12806, 2025.

Sohyun An, Ruochen Wang, Tianyi Zhou, and Cho-Jui Hsieh. Don't think longer, think wisely: Optimizing thinking dynamics for large reasoning models. *Advances in Neural Information Processing Systems*, 38:111021–111046, 2026.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International conference on learning representations*, volume 2024, pp. 9112–9141, 2024.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, sunhaoze, ZhouYijie, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. Research: Learning to reason with search for LLMs via reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=OuGAwwAT8G.

Yifei Chen, Guanting Dong, and Zhicheng Dou. ET-agent: Incentivizing effective tool-integrated reasoning agent via behavior calibration. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, July 2026.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Agentic reinforced policy optimization. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=TX4k7BF6aO.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 630(8017):625–630, 2024.

Yichao Fu, Junda Chen, Siqi Zhu, Fu Fu, Zhongdongming Dai, Yonghao Zhuang, Yian Ma, Aurick Qiao, Tajana S Rosing, Ion Stoica, et al. Efficiently scaling llm reasoning programs with certaindex. *Advances in Neural Information Processing Systems*, 38:141987–142025, 2026.

Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for switching bandit problems. In *International Conference on Algorithmic Learning Theory (ALT)*, 2011.

Weiyang Huang, Xuefeng Bai, Kehai Chen, Xinyang Chen, Yibin Chen, Weili Guan, and Min Zhang. Sat: Balancing reasoning accuracy and efficiency with stepwise adaptive thinking. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 43384–43402, 2026.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 7969–7992, 2023.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=VD-AYtP0dve.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

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.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Wenjun Li, Dexun Li, Kuicai Dong, Cong Zhang, Hao Zhang, Weiwen Liu, Yasheng Wang, Ruiming Tang, and Yong Liu. Adaptive tool use in large language models with meta-cognition trigger. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13346–13370, 2025a.

Xianzhi Li, Ethan Callanan, Abdellah Ghassel, and Xiaodan Zhu. Entropy-gated branching for efficient test-time reasoning. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, March 2026a.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 5420–5438, 2025b.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. Webthinker: Empowering large reasoning models with deep research capability. *Advances in Neural Information Processing Systems*, 38:120091–120131, 2026b.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and Kan Li. Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=ndR8Ytrzhh.

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.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tur, Gokhan Tur, and Heng Ji. Toolrl: Reward is all tool learning needs. *Advances in Neural Information Processing Systems*, 38:105523–105553, 2026.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *The twelfth international conference on learning representations*, 2023.

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.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472, 2022.

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 *International Conference on Learning Representations*, volume 2025, pp. 10131–10165, 2025.

Sangjun Song, Minjae Oh, Seungkyu Lee, Sungmin Jo, and Yohan Jo. Thinkbrake: Efficient reasoning via log-probability margin guided decoding. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 21765–21790, 2026.

Qisheng Su, Shiting Huang, Zhen Fang, Ziyan Chen, Zehui Chen, and Feng Zhao. Beyond accuracy: Unveiling inefficiency patterns in tool-integrated reasoning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7456–7477, 2026.

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

Yuan Sui, Yulin Chen, Yibo Li, Xue Jiang, Yufei He, Yihong Dong, Xiaoxin He, Tianyu Gao, and Bryan Hooi. Tact: Mitigating overthinking and overacting in coding agents via activation steering. *arXiv preprint arXiv:2605.05980*, 2026.

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.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 10014–10037, 2023.

Junde Wu, Jiayuan Zhu, Yuyuan Liu, Min Xu, and Yueming Jin. Agentic reasoning: A streamlined framework for enhancing llm reasoning with agentic tools. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 28489–28503, 2025.

Hongshen Xu, Zihan Wang, Zichen Zhu, Lei Pan, Xingyu Chen, Shuai Fan, Lu Chen, and Kai Yu. Alignment for efficient tool calling of large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 17787–17803, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 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. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=NpU7ZXafRi.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. Towards thinking-optimal scaling of test-time compute for llm reasoning. *Advances in Neural Information Processing Systems*, 38:43605–43631, 2026b.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

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

Jingyang Yi, Jiazheng Wang, and Sida Li. Shorterbetter: Guiding reasoning models to find optimal inference length for efficient reasoning. *Advances in Neural Information Processing Systems*, 38: 39011–39043, 2026.

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*, 2025. URL https://openreview.net/forum?id=O6I0Av7683.

Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, et al. Process vs. outcome reward: Which is better for agentic rag reinforcement learning. *Advances in Neural Information Processing Systems*, 38: 58701–58729, 2026.

Zihan Zhang, Meng Fang, and Ling Chen. Retrievalqa: Assessing adaptive retrieval-augmented generation for short-form open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 6963–6975, 2024.

A IMPLEMENTATION

Hyperparameters. Table 3 summarizes the inference and action probing hyperparameters used in the experiments. To ensure fair comparisons, all general inference configurations are shared between OPERA and all baselines. We evaluate OPERA using Qwen3-4B-Thinking-2507 and Qwen3-8B with thinking mode enabled. Considering their different reasoning and decoding behaviors, we use model-specific majority-voting thresholds η for action probing. Qwen3-4B tends to generate more redundant deliberation and occasional inconsistencies among action probes; therefore, we set $\eta = 0.75$ to tolerate inconsistent predictions while still identifying the dominant action intention. In contrast, Qwen3-8B produces more consistent action probing predictions, and thereby we adopt a stricter threshold of $\eta = 1.0$ to require unanimous agreement before early termination and further mitigate potential premature stopping. All experiments are conducted on 6 x NVIDIA L40S GPUs (48 GB). The reported EM results are exact-match accuracy at one generation (EM@1). The complete codebase and all configurations will be made publicly available upon publication of the paper under the MIT License.

Table 3: Inference and probing hyperparameter settings of OPERA. All general inference configurations are shared with all baselines to ensure fair comparisons.

Hyperparameter	**Value**
General Inference Configuration	
Base Model	Qwen3-4B-Thinking-2507 / Qwen3-8B (Thinking Mode)
Temperature	0.7
Top-p	1.0
Shannon Entropy Top-K	Top-20
Random Seed	0
Max Tool-Call Round	6
Max Generation Length per Round	1024
Consensus-based Hierarchical Action Probing	
Rolling Monitor Window Length	256
Majority Voting Threshold η	0.75 / 1.0
Content Similarity Threshold τ	0.9
Action Probing Budget B_{max} per Probe	22 (Action Type: 2; Action Content: 20)
Action Probe Count C	4
On-the-Fly Probe-Timing Estimation	
Probe-Timing Estimation Set D	0, 32, 64, 128, 256
Recorded Recent Rewards N	32
Effectiveness Reward Weight	w_1 = 2/ w_2 = 0.5

Tool-integrated system prompt and action probe prefix. Table 4 presents the system prompt and hierarchical action probe prefixes used by OPERA. The same system prompt is applied to OPERA and all baselines to ensure consistent reasoning and output formats. All methods are also provided with the same BM25 retriever using 2018 Wikipedia database, which returns the top-3 passages for each search query. For OPERA, the action type probe prefix is used to predict whether its next action is *search* or *answer*, while the action content prefix conditions subsequent generation on the selected action type a^* to preview the corresponding search query or final answer. This shared system prompt and search-tool setup ensures that the observed performance gains are attributable to OPERA's on-the-fly probing rather than differences in the underlying instructions or retrieval tools.

Table 4: System prompt and hierarchical action probing prefixes of OPERA.

System Prompt

You are a helpful assistant that can solve the given question step by step with the help of the wikipedia search tool.
Given a question, you need to first think about the reasoning process in the mind and then provide the answer.
During thinking, you can invoke the wikipedia search tool to search for fact information about specific topics if needed.
The reasoning process and answer are enclosed within <*think*> </*think*> and <*answer*> </*answer*> tags respectively, and the search query is enclosed within <*search*> </*search*>.
After a query, it will return the top searched results between <*result*> and </*result*>.
For example,
<*think*> This is the reasoning process. </*think*> <*search*> query here </*search*> <*result*> search results here </*result*>
<*think*> This is the reasoning process. </*think*> <*answer*> The final answer is \[\boxed{answer here} </*answer*>.
In the last part of the answer, the final exact answer is enclosed within \boxed{} with latex format.

Action Type Probe Prefix

... Wait, I suddenly know it. Let me start with one tag: <*search*> or <*answer*>. </*think*><

Action Content Probe Prefix

If selected action type $a^* =$ search:
... Wait, I suddenly know it. Let me start searching. </*think*><*search*>
If selected action type $a^* =$ answer:
... Wait, I suddenly know it. Let me start answering. </*think*><*answer*> The final answer is \[\boxed{

B ALGORITHM WORKFLOW OF OPERA

As summarized in Algorithm 1, OPERA operates alongside the ongoing reasoning and adaptively determines when to evaluate whether the model is ready to take its next action. At each step, OPERA estimates the next probe time from the recent probe-timing reward history and allows the model to continue reasoning for the selected number of tokens while updating a rolling monitoring window with normalized entropy. If the model naturally emits a search or answer action before the scheduled probe, OPERA returns that action and updates the probe-timing reward history accordingly. Otherwise, OPERA places multiple probes at fixed intervals within the monitoring window and first evaluates whether they reach sufficient consensus on the next action type. Only when the action-type vote passes does OPERA generate action-content previews and assess their consistency using an action-aware similarity threshold. If both stages pass, OPERA stops the remaining reasoning and outputs the agreed action and content. If either stage fails, the probe outputs are discarded and the original reasoning trajectory resumes. After each probing, OPERA updates the recent timing rewards using the probing outcome, probe-token cost, and entropy-based weighting, which enables subsequent probe-timing estimations to adapt to the evolving reasoning process.

Algorithm 1 On-the-Fly Probing of Agentic Actions for Efficient Reasoning (OPERA)

Input: Reasoning trajectory generated by π_θ; hyperparameters $C, \eta, \tau, D, N, B_{\max}, w_1, w_2$

Initialize recent N timing-reward history
Initialize a rolling monitor window of recent reasoning tokens
while no *search* or *answer* action has been emitted **do**
On-the-fly probe-timing estimation
    Estimate the next probe time $d \in D$ (Equation 11)
    Continue the reasoning trajectory for up to d tokens
    Update the rolling monitor window with normalized entropy
    if π_θ emits a natural action (a, u) **then**
        Compute weighted reward $\widetilde{r}$ (Equation 10)
        Append $(d, \widetilde{r})$ to reward records and retain its N most recent entries
        Output:(a, u)
    Select C fixed-interval probe points in the rolling window
Stage 1: Action-type probing
    for $j = 1, \ldots, C$ **do**
        Generate $a_j \in \{\textit{search}, \textit{answer}\}$ using probe prefix ρ_{type}
    Update $a^\star$ and $J^\star$ (Equation 3)
    Evaluate pass_1 with voting threshold η (Equation 3)
    if $\text{pass}_1 = 1$ **then**
Stage 2: Action-content probing
        for $j \in J^\star$ **do**
            Generate u_j using probe prefix $\rho_{\text{content}}(a^\star)$
        Update $u^\star$ using content similarity threshold τ (Equation 5)
        Evaluate pass_2 with voting threshold η (Equation 6)
    Update reward with pass_2 and probe cost b
    Append (d, r) to recent reward records and retain its N most recent entries
    if $\text{pass}_2 = 1$ **then**
        Stop reasoning
        Output:$(a^\star, u^\star)$
    else
        Discard probe outputs and resume reasoning

C Case Study

To further examine the reasoning and tool-call quality of OPERA, we analyze examples generated by Qwen3-4B-2507 on NQ, HotpotQA, and GPQA-Diamond. These examples cover stopping with both *search* and *answer* actions and show how OPERA identifies redundant deliberation on the fly, initiates action probing when efficient, and uses consensus-based hierarchical probing to mitigate premature stopping. Overall, OPERA accelerates an action only when the probes agree on both its type and content, thereby improving reasoning efficiency while preserving high-quality tool calls and well-supported answers.

NQ. As shown in Table 5, the model recognizes that external evidence is needed to identify the location of the training camp in Remember the Titans, but continues considering several unsupported possibilities before calling a search. OPERA detects this potential redundant deliberation and triggers action probing during the reasoning process. All four probes predict a search action and generate the same search query, indicating strong agreement on both the next action type and its content. OPERA therefore injects the search call early, and the retrieved evidence directly supports the correct answer. This example shows that OPERA can shorten hesitation before a necessary tool call without changing the intended retrieval target.

HotpotQA. Table 6 demonstrates OPERA's behavior when the next appropriate action is to answer. OPERA does not intervene in the first two rounds, allowing the model to perform the two searches required to identify the book's author and verify Hal Moore's military rank. However, in the third round, the model begins reconsidering an answer that is already supported by the re-

Table 5: OPERA probing example for NQ.

Question
Where did Remember the Titans camp take place?

Correct Answer
Gettysburg College

Round 1
<think>
Okay, the user is asking where the Remember the Titans camp took place. First, I need to recall what "Remember the Titans" refers to. I think it's a movie based on a true story about a football team. The main characters are the T.C. Williams High School football team in Virginia. Wait, but the question is about a camp, not the actual season.

Maybe the camp is part of the story. Let me check. The movie is about the integration of the team in 1971, with the coach and players from different racial backgrounds. The camp might be where they**[Probe 1]** trained before the season. But I'm not sure about the specific location.

I should use the Wikipedia search tool to find more details. Let me search for "Remember the Titans camp location".

Looking up the search results, the first result mentions that the camp was held at the T.C. Williams High School in**[Probe 2]** Virginia. Wait, but maybe it's a different location. Another result says the training camp was at the University of Virginia's campus. Hmm, conflicting information.

Wait, the actual movie was filmed in Virginia, but the story is set in T.C. Williams High School. The camp might be part of the school**[Probe 3]**'s facilities. Alternatively, some sources might mention a specific training ground. Let me check another source.

Another search result states that the team's training camp was held at the T.C. Williams High School's football field. So the answer is T.C. Williams High School in Virginia. But I need to confirm.**[Probe 4]***</think><search>*where did remember the titans camp take place*</search>*

Action Probing Results
Probe 1:
Action type: Search (tool-call)
Action content: where did remember the titans camp take place
Probe 2:
Action type: Search (tool-call)
Action content: where did remember the titans camp take place
Probe 3:
Action type: Search (tool-call)
Action content: where did remember the titans camp take place
Probe 4:
Action type: Search (tool-call)
Action content: where did remember the titans camp take place

Injected Action
*</think><search>*where did remember the titans camp take place*</search>*

Round 2
<think>
The training camp for the "Remember the Titans" team took place at Gettysburg College. This location was used for the team's preparation before the 1971 season, as mentioned in the first search result.
</think>
<answer> The final answer is "["boxed –Gettysburg College˝ "] *</answer>*

Action Probing Results
No probing was triggered in Round 2.

trieved evidence. OPERA then initiates probing: three probes predict the answer action and produce the same answer, while one probe predicts another search. The hierarchical consensus mechanism tolerates this occasional inconsistency and accepts the dominant and content-consistent answer. Thus, OPERA removes unnecessary final-stage deliberation while preserving the multi-hop retrieval needed to reach the correct conclusion.

Table 6: OPERA probing example for HotpotQA.

Question
Myron F. Diduryk's exploits were described in a book by an author holding what military rank?

Correct Answer
Lieutenant General

Round 1
<think>
Okay, let's try to figure out this question. The user is asking about Myron F. Diduryk's exploits being described in a book by an author holding a specific military rank. Hmm, first I need to find out who Myron F. Diduryk is and what book he's associated with.
Wait, Myron F. Diduryk... I'm not immediately familiar with that name. Maybe it's a less common name or a historical figure. Let me check Wikipedia for "Myron F. Diduryk" to see if there's information about him.
I'll do a search for "Myron F. Diduryk" using the Wikipedia search tool. Let me see what comes up. *</think>*
<search> Myron F. Diduryk *</search>*

Action Probing Results
No probing was triggered in Round 1.

Round 2
<think>
Okay, so from the search results, it looks like Myron F. Diduryk was a U.S. Army Major who served in the Vietnam War, specifically in the Battle of Ia Drang. His exploits were described by Hal Moore in the book "We Were Soldiers Once and Young."
Wait, the question is asking for the military rank of the author who described his exploits. So according to the search results, Hal Moore is the author. Now, what was Hal Moore's military rank?
Let me check the Wikipedia page for Hal Moore to find his rank. Let's do a search for "Hal Moore" to get his rank. *</think>*
<search> Hal Moore *</search>*

Action Probing Results
No probing was triggered in Round 2.

Round 3
<think>
Hmm, the search results for Hal Moore indicate he was a lieutenant general. Wait, but in the first document it mentions "in which Moore describes Diduryk as '... the finest battlefield company commander I had ever seen, bar none.'" So the author of the book "We Were Soldiers Once and Young" is Hal Moore.
Wait, in the first result for Myron**[Probe 1]** F. Diduryk: "His exploits in that battle were described by Hal Moore in, "We Were Soldiers Once and Young"."
Looking at Hal**[Probe 2]** Moore's Wikipedia page (from the search results), he was a United States Army lieutenant general. Wait, but in the context of the book, maybe he held**[Probe 3]** a different rank at the time of writing the book? Let me think.
In the third document about Hal Moore: "Hal Moore Harold Gregory "Hal" Moore**[Probe 4]**,*</think>**<answer>* The final answer is "["boxed –Lieutenant General˝ "] *</answer>*

Action Probing Results
Probe 1:
  Action type: Answer
  Action content: The final answer is "["boxed –Lieutenant General˝ "]
Probe 2:
  Action type: Answer
  Action content: The final answer is "["boxed –Lieutenant General˝ "]
Probe 3:
  Action type: Answer
  Action content: The final answer is "["boxed –Lieutenant General˝ "]
Probe 4:
  Action type: Search (tool-call)
  Action content: No search-content probe was run.

Injected Action
*</think>**<answer>* The final answer is "["boxed –Lieutenant General˝ "] *</answer>*

GPQA-Diamond. The GPQA-Diamond example in Tables 7-9 shows how OPERA handles a more complex reasoning process. In the first round, the model has identified epigenetic silencing as

a plausible explanation but remains uncertain about the precise mechanism. The probes consistently predict a search action and generate closely aligned queries about why Klinefelter's syndrome has milder effects than Down's syndrome. After the retrieved evidence points to X-chromosome inactivation, OPERA triggers another search in the second round, where the probes again agree on a more targeted query about the role of X-inactivation. Importantly, OPERA does not trigger probing in the final round, allowing the model to integrate the evidence, distinguish X-inactivation from the other mechanisms listed in the question, and select option C independently. This example shows that OPERA does not stop reasoning indiscriminately. It accelerates clearly established search actions while preserving the productive reasoning needed for a well-supported final answer.

Table 7: OPERA probing example for GPQA-Diamond (Round 1 of 3).

Question

In gametogenesis, the germline undergoes two cycles of meiosis to give rise to haploid gametes. The goal of meiosis, unlike mitosis, is to separate not only the replicated sister chromatids but also the homologous chromosome pairs to form haploid cells. Fertilized gametes enter embryogenesis and ultimately develop into an organism. Meiosis is not always perfectly executed, and most such cases result in embryonic lethality. Viable events can already be detected in the karyotype of embryonic cells from the amniotic fluid. One such event results in Klinefelter's syndrome in boys. Which molecular mechanism is responsible for less prominent phenotypic consequences of Klinefelter's syndrome compared to better known Down's syndrome?

A. progression of the polymerase alpha in the morula/blastocyst
B. attachment of spindle to kinetochores in metaphase I
C. chromatin methylation by histone methyltransferases in the post-zygote
D. chiasmata resolution by separase in diakinesis

Correct Answer

C. chromatin methylation by histone methyltransferases in the post-zygote

Round 1

<think>

Okay, let's tackle this question step by step. The question is about Klinefelter's syndrome and why it has less prominent phenotypic consequences compared to Down's syndrome. The choices are A to D.

First, I need to recall what Klinefelter's syndrome is. From what I remember, Klinefelter's syndrome occurs when a male has an extra X chromosome, so his karyotype is 47,XXY. Down's syndrome is caused by trisomy 21, meaning an extra chromosome 21.

The question is asking which molecular mechanism leads to less severe symptoms in Klinefelter's compared to Down's. So, Down's syndrome is more severe because trisomy 21 affects many genes, leading to significant developmental issues. Klinefelter's (XXY) is usually milder, with symptoms like infertility and some physical characteristics, but not as severe as Down's.

Now, looking at the choices:

A. progression of polymerase alpha in the morula/blastocyst

Polymerase alpha is involved in DNA replication, right? It's part of the process of initiating DNA synthesis. The morula and blastocyst are early embryonic stages. I'm not sure how this would relate to Klinefelter's vs Down's. Maybe this is about DNA replication errors, but I don't see a direct link here.

B. attachment of spindle to kinetochores in metaphase I

This is related to meiosis. Kinetochore attachment is crucial for proper chromosome segregation. Errors here can lead to aneuploidy. But Klinefelter's is a result of nondis**[Probe 1]**junction in meiosis I or II, leading to XXY. However, Down's syndrome is also due to nondisjunction. But how does this mechanism relate to the less severe phenotype in Klinefelter's?

C. chromatin methylation by histone methyltransferases in the post-zygote

[Probe 2]Chromatin methylation affects gene expression. If there's methylation of certain genes in the post-zygote stage, maybe it buffers the effects of an extra chromosome. For example, in Klinefelter's, the extra X chromosome might have some silencing through methylation, so the genes aren't**[Probe 3]** overexpressed as much as in Down's syndrome where chromosome 21 has more genes that are critical. Down's syndrome involves trisomy 21, which has about 200 genes, and the extra copy causes more severe issues because those genes are dosage-sensitive. In contrast, XXY males have**[Probe 4]** one*</think><search>* Klinefelter's syndrome mechanism less severe than Down's syndrome *</search>*

Action Probing Results

Probe 1:
  Action type: Search (tool-call)
  Action content: Klinefelter's syndrome molecular mechanism less severe than Down's syndrome
Probe 2:
  Action type: Search (tool-call)
  Action content: Klinefelter's syndrome and Down's syndrome molecular mechanisms comparison
Probe 3:
  Action type: Search (tool-call)
  Action content: Klinefelter's syndrome mechanism less severe than Down's syndrome
Probe 4:
  Action type: Search (tool-call)
  Action content: Klinefelter's syndrome mechanism less severe than Down's syndrome

Injected Action

</think><search> Klinefelter's syndrome mechanism less severe than Down's syndrome *</search>*

Table 8: OPERA probing example for GPQA-Diamond (Round 2 of 3).

Round 2

<think>

Hmm, the search results mention that Klinefelter's is XXY and that it's often milder than Down's. But the question is about the molecular mechanism responsible for less severe symptoms.

Let me think again. Down's syndrome is trisomy 21, which has about 200 genes. Klinefelter's is XXY, so the extra X chromosome. In females, one X is inactivated (X-inactivation), so in Klinefelter's (XXY), one X is inactivated, leading to a milder phenotype. That's a key point. So the mechanism is X-chromosome inactivation (lyonization) which silences one X chromosome in each cell. Therefore, even though there are two X chromosomes, one is inactivated, so the effective gene dosage is similar to a normal male (XY). In Down's syndrome, there's no such inactivation for chromosome 21, so all three copies are active, leading to more severe effects.

So the molecular mechanism here would be X-chromosome inactivation, which is mediated by the Xist RNA and subsequent methylation. But looking at the choices given:

C. chromatin methylation by histone methyltransferases in the post-zygote

Chromatin methylation is part of epigenetic silencing. X-inactivation involves DNA methylation and histone modifications. So this might be the answer. Let's check the choices.

Choice C says "chromatin methylation by histone methyltransferases in the post-zygote". Histone methyltransferases add methyl groups to histones, which can lead to gene silencing. X-inactivation occurs**[Probe 1]** early in embryonic development (post-zygote stage), so this might be it.

Wait, the answer options are given as choices. Let me re-express each choice:

A. polymerase alpha progression in morula/blastocyst: morula is 4-day embryo, blastocyst is **[Probe 2]**5-6 days. Polymerase alpha is for DNA replication initiation. Not sure how this relates to X-inactivation.

B. spindle attachment in metaphase I: meiosis I. Nondisjunction here causes aneuploidy, but the question is about why Klinefelter's is less severe,**[Probe 3]** not the cause of the aneuploidy.

C. chromatin methylation by histone methyltransferases post-zygote: this is part of the epigenetic mechanism. X-inactivation happens early in development, involving histone modifications and DNA methylation. So in Klinefelter's,**[Probe 4]** the*</think><search>* X-chromosome inactivation mechanism and Klinefelter's syndrome *</search>*

Action Probing Results

Probe 1:
Action type: Search (tool-call)
Action content: X-inactivation mechanism and its role in Klinefelter's syndrome

Probe 2:
Action type: Search (tool-call)
Action content: Klinefelter syndrome less severe due to X inactivation mechanism

Probe 3:
Action type: Search (tool-call)
Action content: X-inactivation mechanism in Klinefelter syndrome

Probe 4:
Action type: Search (tool-call)
Action content: X-chromosome inactivation mechanism and Klinefelter's syndrome

Injected Action

</think><search> X-chromosome inactivation mechanism and Klinefelter's syndrome *</search>*

Table 9: OPERA probing example for GPQA-Diamond (Round 3 of 3).

Round 3

<think>

Okay, so the search results indicate that in Klinefelter's syndrome (XXY), X-inactivation occurs, which silences one X chromosome, reducing the impact of the extra X. This is why the symptoms are less severe compared to Down's syndrome (trisomy 21), where there's no such inactivation for chromosome 21.

The molecular mechanism here is X-inactivation, which involves epigenetic modifications, including histone methylation and DNA methylation. Among the choices given, option C states: "chromatin methylation by histone methyltransferases in the post-zygote". Histone methyltransferases add methyl groups to histones, leading to gene silencing. X-inactivation occurs post-zygotically (after fertilization), so this fits the "post-zygote" part.

Let me check the other choices again.

Option B is about spindle attachment in metaphase I. That's related to meiosis errors causing aneuploidy (like nondisjunction), but the question is about why the phenotype is less severe, not the cause of the aneuploidy.

Option D: chiasmata resolution by separase in diakinesis. Chiasmata are involved in crossing over during prophase I of meiosis. Separase resolves chiasmata, but this is part of meiosis I, not directly related to the phenotypic consequences in the embryo.

So the correct answer is C. The chromatin methylation (via histone methyltransferases) in the post-zygote stage leads to X-inactivation, which silences the extra X chromosome in Klinefelter's, resulting in milder symptoms compared to Down's syndrome where chromosome 21 doesn't have such a silencing mechanism.

</think>

The question asks about the molecular mechanism responsible for the less prominent phenotypic consequences of Klinefelter's syndrome (XXY) compared to Down's syndrome (trisomy 21).

Klinefelter's syndrome involves an extra X chromosome (47,XXY), while Down's syndrome results from trisomy 21. The key difference in phenotypic severity lies in the mechanism of **X-chromosome inactivation**. In Klinefelter's syndrome, one of the X chromosomes is inactivated (via epigenetic modifications like histone methylation), which reduces the impact of the extra X chromosome. This inactivation ensures that only one X chromosome remains active per cell, similar to normal males (XY). In contrast, Down's syndrome lacks such a mechanism for chromosome 21, leading to overexpression of genes on chromosome 21 and more severe phenotypic consequences.

Reviewing the options:
- **Option A** (polymerase alpha progression) relates to DNA replication and is unrelated to X-inactivation.
- **Option B** (spindle attachment in metaphase I) refers to meiotic errors that cause aneuploidy but does not explain reduced severity in Klinefelter's.
- **Option C** (chromatin methylation by histone methyltransferases in the post-zygote) aligns with X-inactivation, which involves histone modifications and occurs post-zygotically to silence the extra X chromosome in Klinefelter's syndrome.
- **Option D** (chiasmata resolution) is a meiotic process unrelated to post-zygotic epigenetic silencing.

The correct answer is **C**, as chromatin methylation by histone methyltransferases enables X-inactivation, mitigating the effects of the extra X chromosome in Klinefelter's syndrome.

<answer> The final answer is "["boxed –C˝ "] *</answer>*

Action Probing Results

No probing was triggered in Round 3.