SyncThink: A Training-Free Strategy to Align Inference Termination with Reasoning Saturation

Gengyang Li[1,2*] **Wang Cai**[1,2*] **Yifeng Gao**[1,2] **Yunfang Wu**[1,3†]
[1]National Key Laboratory for Multimedia Information Processing, Peking University
[2]School of Software and Microelectronics, Peking University
[3]School of Computer Science, Peking University
{ligengyang, caiwang, yifgao26}@stu.pku.edu.cn wuyf@pku.edu.cn

Abstract

Chain-of-Thought (CoT) prompting improves reasoning but often produces long and redundant traces that substantially increase inference cost. We present **SyncThink**, a training-free and plug-and-play decoding method that reduces CoT overhead without modifying model weights. Rather than treating reasoning completeness as a theoretically guaranteed criterion, SyncThink uses an empirically calibrated internal stopping signal derived from the model's reasoning-to-answer transition. Our analysis provides mechanistic evidence that answer generation relies disproportionately on the transition token </think>, consistent with an information-bottleneck effect, and we leverage its logit dynamics to detect reasoning saturation and terminate reasoning early. Experiments on GSM8K, MMLU, GPQA, and BBH across three DeepSeek-R1 distilled models show that SyncThink achieves 62.00% average Top@1 accuracy using 656 generated tokens and 28.68s latency, compared to 61.22%, 2141 tokens, and 92.01s for full CoT decoding. On long-horizon tasks such as GPQA, SyncThink further yields up to +8.1 absolute accuracy by mitigating over-thinking.

1 Introduction

Large language models (LLMs) (Vaswani et al., 2017; Zhang et al., 2025b) achieve strong reasoning with Chain-of-Thought (CoT) prompting (Wei et al., 2022), but long CoT traces substantially increase inference cost. As Figure 1 shows, accuracy often saturates beyond a certain token budget, yielding diminishing returns. Together with prior findings that models may over-think (Chen et al., 2025), often reach correct answers early (Liu and Wang, 2025), and can perform well with concise drafts (Xu et al., 2025), this suggests a mismatch between generated length and actual reasoning demand.

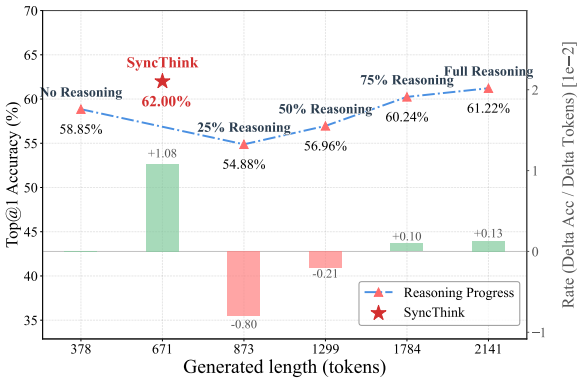


Figure 1: **Efficiency-Accuracy Trade-off.** The blue curve depicts the base model's capability boundary. **SyncThink** (red star) significantly surpasses this limit, occupying the **top-left Pareto frontier** with higher accuracy and reduced token usage. The bottom bars represent the **marginal reasoning efficiency** (ΔAcc/ΔTokens). Our method achieves the highest score (1.08), demonstrating that it effectively retains critical reasoning while pruning redundancy. More results across four datasets can be viewed in Appendix B.

Existing solutions remain limited. Training-based methods (Yao et al., 2023; Bi et al., 2024; Ye et al., 2024; Zhang et al., 2024) require additional fine-tuning or specialized data, while training-free methods such as Answer Convergence (Liu and Wang, 2025) and sampling-based stopping rules (Sui et al., 2025) rely on black-box consistency checks or repeated probing, often adding extra inference cost.

We characterize this inefficiency as **Cognitive Lag**: the model has already reached a sufficiently informative reasoning state, yet continues generating redundant tokens. Our analyses suggest that the reasoning-to-answer transition is often associated with a boundary token (e.g., </think>) that serves as an information bridge for answer generation. We therefore track the step-wise logit rank of this token as an internal stopping signal. Macro-level results on BBH further show that accuracy saturates in an

[*] Equal contribution.
[†] Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 4657–4672
July 2-7, 2026

"optimal truncation zone" (starting at roughly 60% of the reasoning progress), well before the model naturally emits its termination token.

However, existing solutions remain limited. Training-based methods (Yao et al., 2023; Bi et al., 2024; Ye et al., 2024; Zhang et al., 2024) require additional fine-tuning or specialized data, reducing deployability. Existing training-free approaches, such as Answer Convergence (Liu and Wang, 2025) and sampling-based stopping rules (Sui et al., 2025), are largely based on black-box consistency checks or repeated probing at the answer level, which can introduce additional inference cost without explicitly leveraging the model's internal transition dynamics.

We characterize this inefficiency as **Cognitive Lag**: the model has already reached a sufficiently informative reasoning state, yet continues generating redundant tokens. To study this phenomenon, we analyze the transition from reasoning to answering in reasoning-capable models. Attention and saliency analyses (figures 2 and 3) suggest that a transition token (e.g., `</think>`) can act as an **information bridge**, concentrating information needed for answer generation. We therefore use the step-wise logit rank of this token as a lightweight internal probe of the reasoning-to-answer transition. Macro-level results on BBH further show that answer accuracy saturates in an "optimal truncation zone" (starting at roughly 60% of the reasoning progress), well before the model naturally emits its termination token.

To bridge this lag, we propose **SyncThink**, a training-free decoding framework motivated by this transition-state analysis. Rather than treating reasoning completeness as a theoretically guaranteed criterion, SyncThink uses an empirically calibrated stopping policy based on the model's internal reasoning-to-answer signal. In reasoning-specialized models with an explicit boundary token (e.g., `</think>` in DeepSeek-R1), SyncThink monitors the token's logit *rank* together with distributional *entropy*, and triggers *dynamic early stopping* when the model reaches a saturated reasoning state.

Experiments show that SyncThink achieves a better efficiency–accuracy trade-off. In Figure 1, SyncThink reaches 62.00% Top@1 accuracy with ∼656 tokens, outperforming full CoT (61.22%, ∼2141 tokens) while reducing the token budget by 69.4% (3.21× speedup).

Our contributions are as follows:

- We define **Cognitive Lag** in CoT reasoning and show, through micro- and macro-level analyses, that answer accuracy can saturate well before the model's natural termination point.
- We provide mechanistic evidence that the reasoning-to-answer transition is associated with an information-bridging token such as `</think>`, and show that its logit dynamics offer a useful internal signal for monitoring reasoning saturation.
- We propose **SyncThink**, a training-free decoding method that uses an empirically calibrated internal stopping signal to dynamically truncate redundant reasoning, improving efficiency while preserving or improving accuracy across tasks.

2 Related Work

2.1 CoT Compression

Efficient CoT generation is commonly studied from two directions: training-based compression and inference-time optimization (Qu et al., 2025a; Sui et al., 2025). Training-based methods reduce reasoning overhead by pruning explicit reasoning tokens (Han et al., 2024; Zhang et al., 2025a) or compressing reasoning into latent representations (Chen et al., 2024; Qu et al., 2025b). Inference-time methods instead improve efficiency without updating model weights, e.g., by generating concise reasoning drafts (Aytes et al., 2025; Xu et al., 2025) or by stopping early based on answer-level or transition-level signals (Liu and Wang, 2025; Sui et al., 2025).

SyncThink is most closely related to inference-time stopping methods. DEER(Yang et al., 2025) or Answer Convergence(Liu and Wang, 2025) uses lexical transition cues to trigger trial-answer generation and rollback, with efficiency often relying on branch-parallel / overlapped execution. SSW(Wei et al., 2025) detects a Reasoning Completion Point from short-window `</think>` rank dynamics using lightweight rules. By contrast, SyncThink tracks an intrinsic readiness signal directly from normal decoding logits, namely the boundary-token rank together with an entropy-aware threshold, and performs early stopping in a strictly single-pass manner. Relative to DEER, it avoids trial answers and rollback; relative to SSW, it introduces uncertainty-aware control and is additionally motivated by a mechanistic bottleneck interpretation from attention and saliency analysis.

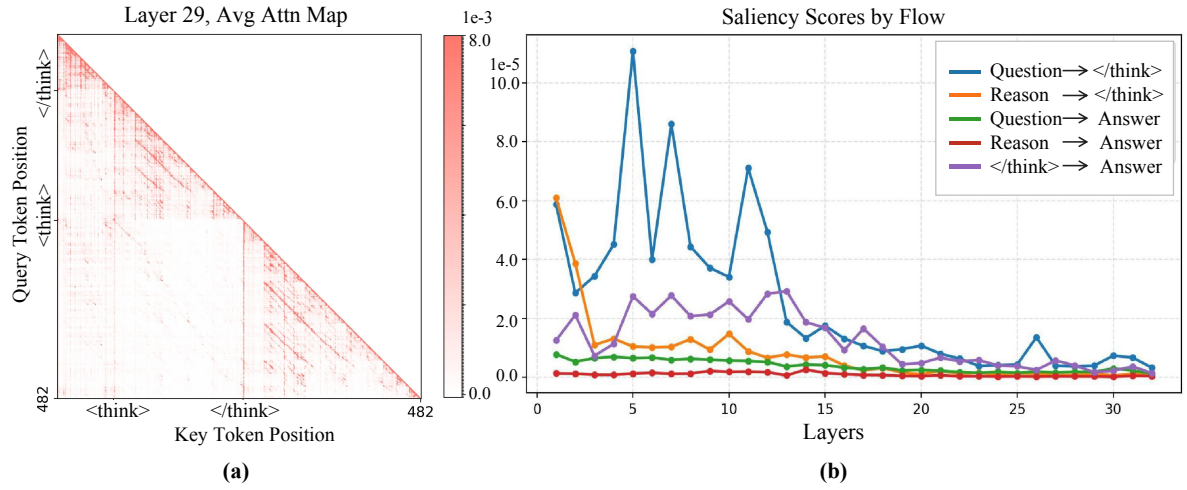


Figure 2: Empirical evidence of the **Universal Reasoning Bottleneck** on a GSM8K example. (a) At Layer 29, attention concentrates on the `</think>` boundary rather than reasoning tokens (more layers in Fig. 8). (b) Saliency analysis shows that information transfer from Reasoning to Answer is negligible (red line), while the dominant pathway is mediated by `</think>` (purple line), suggesting reasoning is compressed into this transition token.

3 Methodology

We propose **SyncThink**, a *training-free* framework for improving CoT reasoning efficiency by reducing redundant computation. To motivate this approach from the model's internal dynamics, our methodology follows a three-stage analytical pipeline:

(1) **Mechanistic Analysis (§3.2):** Through attention and saliency analyses, we provide mechanistic evidence that the `</think>` token is closely associated with the reasoning-to-answer transition and can serve as a useful internal signal for monitoring reasoning progression.

(2) **Phenomenon Quantification (§3.3):** By tracking the temporal dynamics of this signal, we characterize **Cognitive Lag**. Our micro-level phase analysis and macro-level accuracy statistics jointly show that answer accuracy can saturate before the model reaches its natural termination point.

(3) **Algorithm Design (§3.4):** Building on these observations, we formulate SyncThink as an empirically calibrated stopping strategy that uses dynamic rank and entropy thresholds to truncate redundant reasoning once the model reaches a sufficiently saturated reasoning state.

3.1 Problem Formulation: The Efficiency-Accuracy Trade-off

Given a question q, an LLM generates a reasoning chain $r = (r_1, \ldots, r_L)$ followed by an answer a. While r generally enhances the conditional probability $P(a|q, r)$, the computational cost scales linearly with L. Recent observations in the literature (Chen et al., 2025; Liu and Wang, 2025) indicate that this performance gain exhibits *diminishing returns*: accuracy tends to saturate rapidly after an initial rise, implying that extending L beyond a critical point yields negligible gains.

Therefore, our objective is to find an optimal truncation point $\hat{t}$ that maximizes accuracy while minimizing cost:

$$\hat{t} = \arg\max_t \left(A(t) - \alpha \cdot \text{Cost}(t)\right) \quad (1)$$

Since the true accuracy $A(t)$ is unobservable during inference, we seek an intrinsic proxy from the model's internal state to estimate reasoning sufficiency.

3.2 Attention Analysis: The Information Bottleneck

To identify an intrinsic proxy, we first examine internal attention maps, as self-attention governs information flow in Transformers. As shown in Figure 2, deep-layer attention allocates negligible mass to individual reasoning tokens and instead concentrates on the `</think>` boundary. This observation suggests that (i) the reasoning span contains substantial redundancy, and (ii) `</think>` serves as an **information bottleneck** where critical context is compressed and accumulated for answer generation.

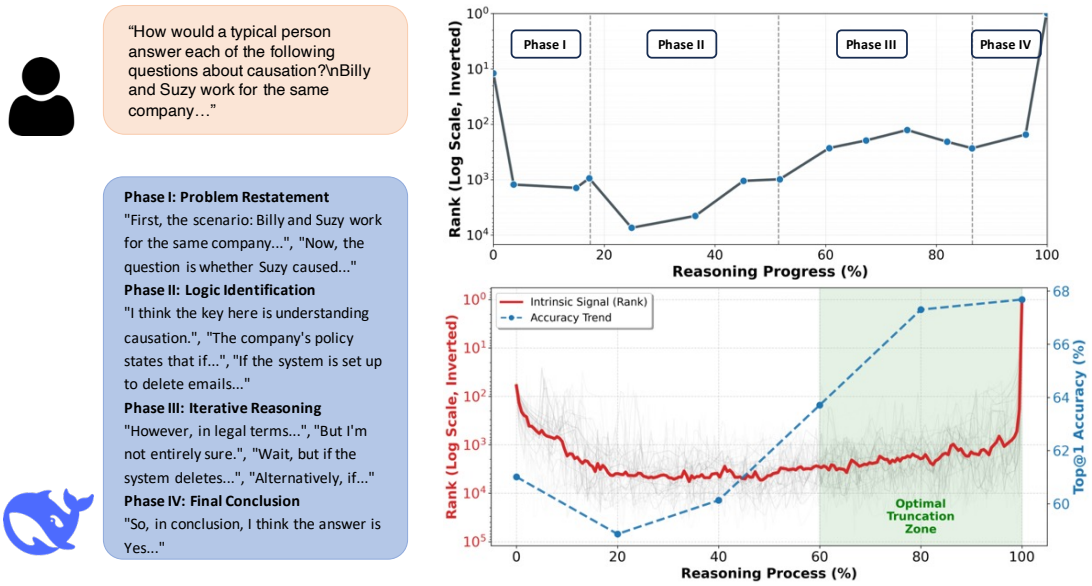


Figure 3: **The rationale behind SyncThink.** (Top) **Micro-Dynamics:** The </think> rank provides a real-time signal that tracks the four reasoning phases in §3.3. (Bottom) **Macro-Statistics:** On BBH, we observe a *cognitive lag*: truncation accuracy (blue) saturates within the shaded *optimal truncation zone* well before the model's intrinsic termination signal (red).

Further, to validate these hypotheses, we employ Saliency Score analysis following Wang et al. (2023). This framework quantifies the contribution of specific attention regions by measuring the sensitivity of the loss function to masked perturbations via a first-order Taylor approximation:

$$\Delta L \approx \alpha \left\langle A_{h,l} \odot \frac{\partial L}{\partial A_{h,l}}, M \right\rangle_F. \quad (2)$$

This approximation reveals that saliency is essentially the element-wise product of the attention activations and their gradients within the masked region.

As illustrated in Figure 2, this analysis yields three key insights regarding the reasoning mechanism:

- **The </think> Token as an Information Bridge:** The information flow follows a "Reasoning → </think> → Answer" path. Direct saliency from reasoning tokens to the answer is negligible, confirming that </think> serves as a critical mediator that aggregates context before generation.
- **Early Aggregation and Logic Construction:** Saliency peaks in shallow layers (1–2) for context compression, while core logical decision-making is centralized in the middle layers (5–15). This indicates that the primary information processing occurs in the early-to-mid stages of the network.
- **Completion of Information Integration:** In deeper layers (> 20), saliency scores across all key paths diminish significantly (quiescence). This confirms that critical information migration to </think> is fully concluded, rendering further explicit retrieval of reasoning history redundant.

Our saliency analysis identifies </think> as a critical information bottleneck where reasoning content is distilled, while intermediate tokens exhibit rapid saliency decay. This phenomenon is not limited to specific instances and is analyzed in detail in Appendix C. Consequently, the model's primary reliance on this distilled signal supports our strategy of using it for dynamic early stopping.

3.3 The Temporal Dynamics of Reasoning Completeness

Building on the identification of *where* information concentrates, we now examine the temporal dynamics to determine *when* reasoning becomes sufficient. Specifically, we track the evolution of the </think> token's rank, defined as the rank of its logit among the full vocabulary at each decoding step. We use this step-wise logit rank as a lightweight proxy for the model's internal reasoning state. As shown in Figure 3 (Top), the rank trajectory maps to four distinct cognitive phases:

Phase I: Problem Restatement (Sharp Descent). The rank initially precipitates from the start to $> 10^4$. This marks the transition from instruction encoding to reasoning initiation, indicating the model's commitment to generating a long-form

chain rather than an immediate answer.

Phase II: Logic Identification (Initial Recovery). A steady recovery to $\sim 10^3$ follows, corresponding to the *Orientation* stage. As the model structures the problem space and identifies a viable logical path, the uncertainty regarding termination decreases.

Phase III: Iterative Reasoning (Volatile Plateau). The rank oscillates between 10^2 and 10^3. These fluctuations reflect *Self-Reflection*: local peaks suggest the completion of sub-steps, while subsequent drops indicate the activation of verification or backtracking mechanisms during conflict resolution.

Phase IV: Final Conclusion (Linear Ascent). Upon resolving the internal debate, the trajectory shifts to a decisive, linear ascent to Top-1. Unlike the fluctuations in Phase III, this irreversible rise serves as a deterministic signal that the reasoning process is concluded.

This analysis highlights the limitation of fixed token budgets, which fail to distinguish between necessary formulation (Phase II) and potentially redundant exploration (Phase III). Our framework leverages these rank trends to detect latent termination signals, enabling dynamic truncation during periods of "overthinking" to prevent redundant computation.

Macro-Statistics: The Cognitive Lag. We further aggregate reasoning trajectories across the BBH dataset to analyze macro-level trends (Figure 3, Bottom). Here, the red curve depicts the median rank of the </think> token overlaid on individual sample traces (grey), while the blue dashed line indicates the Top-1 accuracy achievable by truncating generation at the corresponding progress percentage.

This macro-view reveals a critical decoupling between model capability and generation policy, termed **"Cognitive Lag"**. Specifically, as reasoning progresses beyond 60%, the truncation accuracy (blue) enters a saturation plateau, recovering to near-optimal levels. Paradoxically, the model's intrinsic termination signal (red) remains deeply suppressed (rank $\sim 10^3$) throughout this phase, indicating a lack of confidence to terminate despite having effectively solved the problem. This misalignment drives the model to engage in prolonged, low-utility reasoning to pursue negligible performance gains.

Consequently, a dynamic intervention mechanism is required to decouple termination from the model's inertia, intervening precisely when information saturation is detected.

3.4 SyncThink: A Dynamic Reasoning Termination Method Based on the Logit

Our investigation reveals a mechanism of **Progressive Information Condensation**, where the model aggregates reasoning logic into evolving hidden states and employs the </think> token as a "summary embedding" of completeness. This dynamic reflects a **Mechanism Competition** between *Exploration* and *Conclusion* modes, where a rising rank of </think>—indicative of state saturation—signals the model's readiness to transition. Addressing the noise sensitivity caused by the magnitude dominance of raw logits, we demonstrate that the token **Rank** serves as an *intrinsic proxy for information sufficiency*.

Motivated by this, we introduce a heuristic dynamic truncation mechanism. When the rank of the terminator token ascends to a sufficiently high position, we manually inject </think> to terminate the reasoning process. We formulate this strategy as follows:

$$\mathbb{I}_{\text{stop}}(t) = \begin{cases} 1 & \text{if } \mathcal{R}_t(\texttt{</think>}) \leq \tau(t, \mathbf{p}_t) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Here, $\mathcal{R}_t(\texttt{</think>})$ denotes the rank of the terminator token within the current probability distribution $\mathbf{p}_t$ (0-indexed). The dynamic threshold $\tau(t, \mathbf{p}_t)$ is modulated by both temporal pacing and model uncertainty:

$$\tau(t, \mathbf{p}_t) = \lfloor \beta(t) \cdot \exp(-\lambda \cdot \mathcal{H}(\mathbf{p}_t)) \rfloor \quad (4)$$

The components of the threshold are defined as follows:

- **Time-Dependent Pacing $\beta(t)$:** We directly employ t as the time-scale control factor. As t increases, the threshold rises accordingly, aligning with the intuition that longer reasoning chains are increasingly prone to redundancy. Additionally, we impose an upper bound T_{max} to prevent the threshold from becoming excessively loose in extended sequences, thereby avoiding indiscriminate truncation.

- **Entropy-Aware Scaling $\mathcal{H}(\mathbf{p}_t)$:** We compute the Shannon entropy of the next-token distribution to adjust for model confidence:

$$\mathcal{H}(\mathbf{p}_t) = -\sum_{v \in V} p_t(v) \log p_t(v) \tag{5}$$

where V is the vocabulary. High entropy (uncertainty) reduces the threshold via the term $e^{-\lambda \mathcal{H}}$, tightening the constraint to prevent premature truncation during exploration phases (Wang et al., 2025). Conversely, low entropy encourages commitment to a conclusion.

Once the condition $\mathbb{I}_{\text{stop}}(t) = 1$ is met, we manually inject the `</think>` token and proceed to the answer generation phase.

4 Experimental Setup

Datasets To comprehensively evaluate our proposed method across diverse reasoning and knowledge-intensive scenarios, we conduct experiments on the following four benchmark datasets:

- **GSM8K** (Cobbe et al., 2021): Evaluates multi-step arithmetic reasoning via grade-school math problems.
- **MMLU** (Hendrycks et al., 2020): Assesses general knowledge across 57 diverse domains, ranging from elementary to professional levels.
- **GPQA** (Rein et al., 2024): Tests high-level conceptual understanding with graduate-level scientific questions.
- **BBH** (Suzgun et al., 2022): Stress-tests models on hard-to-solve tasks involving symbolic and logical planning.

Backbone. We select models in the 7B–14B range, reflecting practical constraints while maintaining strong reasoning capabilities.

Qwen2.5-7B/14B (Yang et al., 2024): Leading open-source models with robust reasoning performance. *LLaMA3.1-8B* (Grattafiori et al., 2024): A balanced model optimized for efficient instruction following. Identical decoding settings are applied across all baselines for consistent comparison.

Baselines. We compare SyncThink against five baselines: (1) **Full CoT**: Generates complete reasoning traces, serving as the inference-time quality upper bound; (2) **No CoT**: Directly generates answers without reasoning; (3) **Fixed-Ratio Truncation**: A heuristic that stops reasoning at a predetermined length ratio; (4) **Answer Convergence** (Liu and Wang, 2025): A probing method that triggers early stopping when intermediate answer predictions stabilize across k consecutive segments. (5) **DEER**: A stronger test-time reasoning baseline that improves answer quality through trial-answer generation and rollback, with reported latency benefiting from branch-parallel / overlapped execution; Additionally, on GSM8K, we include an **SFT Oracle/Upper Bound** (fine-tuned on gold reasoning traces) to benchmark the maximum achievable performance via supervised learning.

All baselines are evaluated with identical prompts and answer extraction, and we match the generation budget across methods whenever applicable, enabling an apples-to-apples comparison of accuracy and efficiency.

Metrics. We evaluate each method from both effectiveness and efficiency perspectives using three primary metrics: Top-1 accuracy (Top@1↑), wall-clock inference time (Time↓), and the number of generated tokens (Tokens↓). Top@1 accuracy is computed as the exact-match rate under a unified answer parser for each benchmark.

Decoding parameters. Unless otherwise specified, all results are obtained under **greedy decoding** (temperature $= 0$ and no stochastic sampling) with `max_new_tokens`$= 8192$ for all methods. Under this setting, generation is (near-)deterministic given the same model, tokenizer, prompts, and inputs; consequently, we report results from a **single deterministic run** per configuration. More specific details can be seen in Appendix A. **For our truncation strategy** in Eq. 4, we tune λ on the held-out validation set and fix it to $\lambda = 0.8$, which is then applied unchanged to all test benchmarks; we also set $\beta(t) = t$ (the decoding step) to gradually relax the early-exit constraint as generation proceeds.

5 Results and Analysis

5.1 Main Results

Table 1 presents the performance comparison across four benchmarks. Overall, **SyncThink** achieves a strong trade-off between reasoning quality and inference efficiency, outperforming standard baselines such as **Full Reasoning**, **No Reasoning**, fixed-ratio truncation, and **Answer Convergence**. In particular, **SyncThink** attains the highest average Top@1 accuracy among these methods (**62.00%**), surpassing **Full Reasoning** (61.22%) while using only ∼30% of the token budget (655.80 vs. 2141.25 tokens).

Method	GSM8K			MMLU			GPQA			BBH			Avg Res		
	Top@1	Time	Tokens	Top@1	Time	Tokens	Top@1	Time	Tokens	Top@1	Time	Tokens	Top@1	Time	Tokens
DeepSeek-R1-Qwen2.5-7b															
Full Reasoning	87.57	18.79	440	61.15	73.00	1676	30.30	231.58	5444	65.86	44.65	1005	61.22	92.01	2141
No Reasoning	88.09	11.39	266	53.26	12.01	276	32.82	25.90	609	61.21	16.08	362	58.85	16.35	378
Reasoning Ratio($r = 0.25$)	85.52	13.45	317	49.85	28.41	674	24.75	78.89	1879	59.39	19.51	439	54.88	35.07	827
Reasoning Ratio($r = 0.5$)	85.97	15.22	359	52.48	42.61	1005	25.76	131.20	3170	63.64	29.35	660	56.96	54.60	1299
Reasoning Ratio($r = 0.75$)	86.43	17.05	417	57.55	58.75	1421	31.31	184.43	4458	65.67	37.38	841	60.24	74.40	1784
SFT Model	77.41	4.53	106	51.41	15.80	363	32.32	39.44	927	48.28	30.84	694	52.36	22.65	523
Answer Convergence	87.82	15.28	278	57.12	34.64	754	31.38	180.38	3919	62.36	27.21	503	59.67	64.38	1364
Stop When Enough	86.50	21.90	585	58.20	43.20	983	37.50	73.50	1695	62.00	37.50	825	61.05	44.03	1022
Stop Spinning Wheels	88.50	16.10	290	56.40	33.90	740	32.20	175.00	3800	63.10	28.00	520	60.05	63.25	1338
DEER	89.42	16.28	366	60.20	35.38	748	36.93	51.47	1384	64.58	28.39	533	62.78	32.88	758
SyncThink	87.49	14.39	383	59.10	28.13	643	**38.38**	**47.73**	**1116**	63.03	24.46	541	62.00	28.68	671
DeepSeek-R1-Qwen2.5-14b															
Full Reasoning	93.33	35.24	493	81.60	105.67	1465	43.43	378.28	5300	85.25	92.34	1287	75.90	152.88	2136
No Reasoning	92.04	18.07	251	74.88	21.65	299	40.91	49.92	698	77.37	17.66	247	71.30	26.83	374
Reasoning Ratio($r = 0.25$)	91.15	22.59	319	66.52	44.68	630	35.48	140.23	2021	76.88	37.40	525	67.50	61.23	874
Reasoning Ratio($r = 0.5$)	91.62	27.09	379	70.03	63.90	888	36.92	217.19	3173	82.38	57.00	807	70.24	91.30	1312
Reasoning Ratio($r = 0.75$)	92.12	31.17	459	76.80	87.14	1271	44.88	300.80	4455	85.00	73.78	1070	74.70	123.22	1814
SFT Model	84.43	156.58	2191	64.23	329.49	4569	44.04	367.26	5145	70.59	341.63	4763	65.82	298.74	4167
Answer Convergence	92.23	33.49	381	75.33	70.29	744	41.91	188.38	3741	79.38	68.38	735	72.21	90.14	1400
Stop When Enough	90.80	42.12	592	75.00	82.55	1144	43.50	170.04	2353	80.70	78.26	1086	72.50	93.24	1294
Stop spinning wheels	93.10	34.20	390	76.20	68.50	730	43.00	129.93	2039	80.10	66.20	720	73.10	85.98	1360
DEER	93.43	35.38	401	79.38	73.38	775	47.48	141.58	1844	84.38	59.39	757	76.17	77.43	944
SyncThink	93.03	31.59	442	77.99	61.59	852	**46.97**	**126.29**	**1764**	83.84	58.65	817	75.46	69.53	969
DeepSeek-R1-LLaMA3.1-8b															
Full Reasoning	78.47	23.66	498	62.80	99.38	2076	24.24	283.49	5978	73.33	69.03	1445	59.71	118.89	2499
No Reasoning	80.51	12.72	267	58.33	15.15	315	32.83	28.16	595	63.03	12.68	263	58.68	17.18	360
Reasoning Ratio($r = 0.25$)	76.63	15.52	340	51.20	36.58	768	19.80	97.74	2146	66.13	27.98	588	53.44	44.46	963
Reasoning Ratio($r = 0.5$)	77.04	18.49	403	53.90	58.99	1236	20.61	159.96	3455	70.86	40.87	850	55.60	69.58	1525
Reasoning Ratio($r = 0.75$)	77.45	21.16	486	59.10	79.42	1731	25.05	220.97	4885	73.12	54.95	1195	58.68	94.13	2049
SFT Model	63.99	3.99	84	46.45	10.53	220	24.75	23.51	496	49.29	17.12	358	46.12	13.79	289
Answer Convergence	78.18	15.33	385	59.13	54.93	538	28.98	174.58	3959	65.38	43.78	840	57.92	72.16	1431
Stop When Enough	76.90	27.58	574	58.30	59.22	1239	32.30	107.94	2268	66.10	54.32	1127	58.40	62.27	1302
Stop Spinning Wheels	77.20	16.10	435	60.60	55.20	895	28.40	96.80	1790	68.10	35.10	915	58.58	50.80	1009
DEER	80.26	18.28	399	63.83	59.38	848	32.38	102.48	1707	70.38	36.48	804	61.71	54.16	940
SyncThink	78.32	19.85	419	59.98	41.91	872	**33.84**	**76.29**	**1602**	67.68	38.24	796	59.96	44.07	922

Table 1: Comparison of different methods across GSM8K, MMLU, GPQA, and BBH datasets on DeepSeek-R1-Qwen2.5 and LLaMA3.1 models. We report Top@1 accuracy, Time, and Tokens usage.

Compared with **DEER**, SyncThink does not always achieve higher final accuracy. However, DEER relies on a more complex inference pipeline, including trial-answer generation, rollback, and branch-parallel / overlapped execution, which partially hides its extra probing cost. By contrast, SyncThink is strictly single-pass: it monitors intrinsic transition signals from normal decoding and performs early stopping without trial answers or rollback. As shown in Appendix D, under a purely serial implementation without overlap, DEER becomes substantially slower than SyncThink across all evaluated models. This suggests that SyncThink offers a simpler and more deployment-friendly efficiency gain, whereas DEER attains stronger accuracy with additional engineering complexity and extra test-time compute.

5.2 Mitigating Over-Thinking via Information Synchronization

Models often succumb to "over-thinking," where valid reasoning degrades into hallucination due to delayed termination. **SyncThink** addresses this by synchronizing generation with internal information saturation.

Case Study: Pruning Harmful Redundancy. Figure 6 presents a representative GPQA example where *more* reasoning becomes *worse*. The Full CoT baseline reaches a correct intermediate state and is already poised to answer. However, instead of committing, it continues generating additional derivations and self-corrections (e.g., *"Wait, but I need to relate this to..."*), gradually drifting into an unnecessary re-formulation of the problem and ultimately outputting an incorrect option (C). In contrast, **SyncThink** monitors the logit dynamics of the reasoning-state transition token `</think>` and identifies the onset of this detrimental redundancy. As highlighted in the figure, SyncThink triggers an early exit at the *earlier* `</think>` boundary (upper marker), where the model's information has already saturated, and directly produces the correct answer (A), avoiding the later over-thinking branch that leads to error.

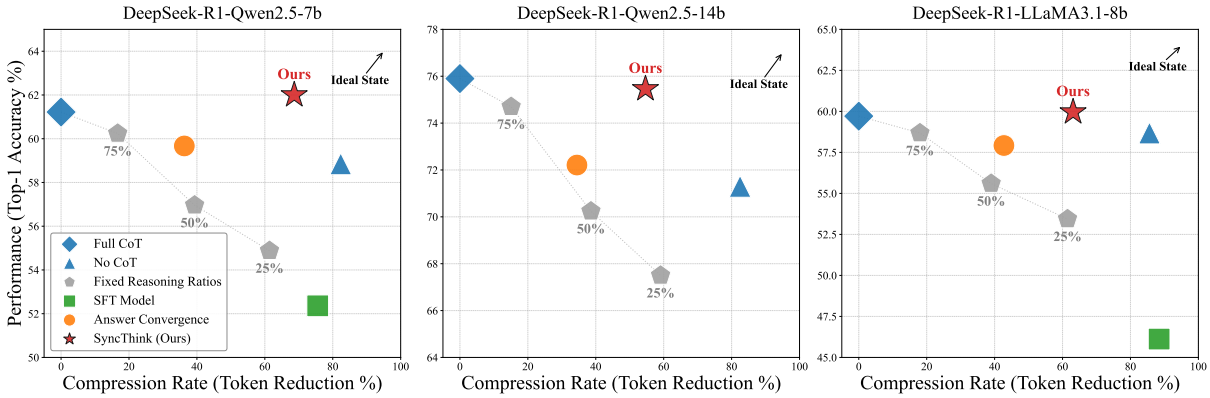


Figure 4: **Efficiency-Accuracy Trade-off.** SyncThink demonstrates superior Pareto efficiency compared to Full CoT, SFT, and static truncation baselines. The red star indicates our method achieves the best balance between token reduction and model performance across all evaluated models.

Adaptive Truncation. Our method also adapts to task complexity. On simple benchmarks (GSM8K), **SyncThink** applies conservative truncation, preserving linear reasoning paths. Conversely, on complex tasks (GPQA), it exhibits a higher truncation rate. This confirms that **SyncThink** selectively targets the "over-thinking" loops prevalent in difficult reasoning, rather than indiscriminately shortening all responses.

5.3 Universality of the Reasoning Bottleneck

A natural question is whether our reliance on the </think> signal is specific to DeepSeek-R1-style training. We clarify that our universality claim is at the *mechanism level*, rather than the *token-format level*. Specifically, we do not assume that a fixed surface token such as </think> must appear in all models. Instead, our claim is that, near reasoning saturation, reasoning-capable models often exhibit a stable reasoning-to-answer routing pattern in which deep-layer attention or attribution concentrates on a small set of boundary-like tokens immediately before answer generation.

To examine whether this phenomenon extends beyond the DeepSeek-R1 distilled family, we conduct additional analyses on Qwen2.5-7B-Instruct, LLaMA3.1-8B-Instruct, Qwen3-8B, and GPT-OSS-20B. For models without a native explicit boundary, we elicit a more consistent reasoning format through few-shot prompting; for models with native thinking-style generation, we use their default format. We find that RLVR-trained reasoning models such as Qwen3 and GPT-OSS exhibit a sharper and more stable bottleneck pattern, whereas non-RLVR models tend to show weaker and more distributed peaks, often aligned with structural segmentation tokens such as \n. This suggests that the routing mechanism is broadly present, while reasoning-oriented training sharpens it into a more singular transition.

Method	Top@1	Tokens
Full Reasoning	85.49	1.00×
SyncThink	86.08	< 0.50×

Table 2: Preliminary validation on Qwen3-8B over MATH500.

We also perform a preliminary end-to-end validation on Qwen3-8B using MATH500. As shown in Table 2, SyncThink preserves performance while substantially reducing token usage, suggesting that the method is not limited to the DeepSeek-R1 family.

Additional visualizations and model-specific examples are provided in Appendix C.

SyncThink improves the accuracy–efficiency Pareto trade-off. On average, it matches **Full Reasoning** accuracy (62.00% vs. 61.22%) while reducing token usage by ∼70% (655.80 vs. 2141.25). Unlike static truncation heuristics, which mainly trade accuracy for speed, SyncThink better preserves useful reasoning while removing redundant computation. Its gains are especially clear on harder tasks such as GPQA, where **SyncThink** (38.38%) outperforms both Full Reasoning (30.30%) and No Reasoning (32.82%), suggesting that early stopping can mitigate late-stage error accumulation. Overall, SyncThink provides a favorable operating point with practical reductions in latency and memory cost.

Architectural Robustness. Table 1 shows that **SyncThink** remains effective across different model families and scales (Qwen and LLaMA, 7B–14B). Notably, it achieves about a 3× speedup on Qwen-7B while preserving accuracy, indicating that the transition-token signal is a useful and robust stopping cue across architectures.

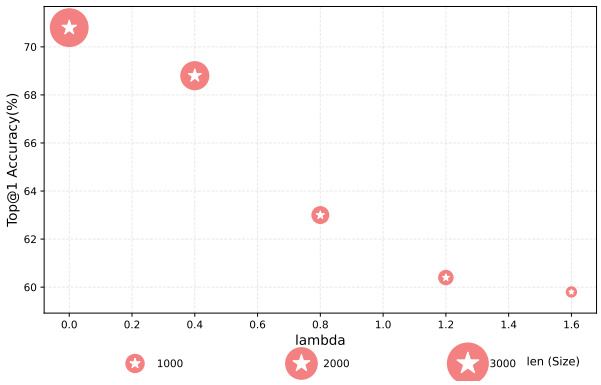


Figure 5: Ablation study on λ.

5.4 Parameter and Stopping-Rule Analysis

To avoid overfitting to evaluation benchmarks, we tune the entropy weight λ on a held-out MATH500 (Lightman et al., 2023) validation set, sweeping λ and measuring Top@1 accuracy and average generation length (Figure 5). Increasing λ monotonically shortens generations, yielding a smooth accuracy–length Pareto trade-off rather than a narrow "sweet spot." This suggests that λ is better viewed as an interpretable operating knob for controlling the efficiency–accuracy balance, rather than a brittle parameter. We use $\lambda = 0.8$ as a balanced default in all experiments.

We further study the pacing term $\beta(t)$ on a subset of MATH500 using DeepSeek-R1-Qwen-7B. As shown in Table 3, the simple linear schedule $\beta(t) = t$ achieves the best or near-best trade-off among the tested variants. Intuitively, $\beta(t)$ acts as a relaxation schedule: as decoding proceeds, redundant continuation becomes more likely, so the stopping threshold should gradually relax.

Finally, the stopping rule combines two complementary signals. The transition-token rank provides a direct probe of readiness to switch from reasoning to answering, while entropy reflects uncertainty in the next-token distribution. Using both yields a simple uncertainty-aware stopping policy that is more robust than relying on either signal alone.

Schedule	Acc	AvgLen	Speed
$\beta(t) = \text{const}$	80.80	1051	43.15
$\beta(t) = t$	83.20	1004	41.89
$\beta(t) = \sqrt{t}$	82.40	1012	41.74
$\beta(t) = t^2$	82.20	937	39.32

Table 3: Ablation on the pacing schedule $\beta(t)$ on a subset of MATH500 using DeepSeek-R1-Qwen-7B. The linear schedule $\beta(t) = t$ achieves the best or near-best trade-off among the tested variants.

5.5 Performance Across Diverse Tasks

SyncThink generalizes effectively across diverse domains, surpassing the **Full Reasoning** baseline with over 60% computational savings. Notably, this is achieved using a unified hyperparameter configuration without task-specific fine-tuning. On complex benchmarks like GPQA, our method successfully mitigates "over-thinking" by halting generation before hallucinations occur: for Qwen2.5-7B, SyncThink improves accuracy from 30.30% (Full Reasoning) to 38.38%. This stability confirms that SyncThink captures intrinsic reasoning patterns. **The hyperparameters thus serve not as sensitive tuning variables, but as a consistent lever for users to prioritize either extended reasoning for marginal gains or aggressive truncation for efficiency**.

6 Conclusion

We propose **SyncThink**, a training-free and model-agnostic decoding framework that synchronizes CoT generation with the model's intrinsic reasoning sufficiency signals. Our key insight is that these signals already indicate *where* and *when* reasoning becomes sufficient. Attention analyses reveal an **information bottleneck** at the reasoning boundary token (e.g., `</think>`), while temporal dynamics show the model can *sense* its reasoning state. Due to training-induced generation bias, the emitted termination is often delayed, causing **Cognitive Lag**—the model is already decidable yet continues producing redundant tokens. SyncThink addresses this misalignment with a lightweight decision rule that aligns decoding with information saturation, reducing unnecessary computation while preserving answer quality. Empirically, SyncThink matches **Full Reasoning** accuracy on average (62.00% vs. 61.22%) with ∼70% fewer tokens (655.80 vs. 2141.25), and improves GPQA to 38.38% (vs. 30.30% Full; 32.82% No).

7 Limitations

While SyncThink is effective in practice, the current work has two main limitations.

- **Heuristic stopping criterion.** Our stopping rule is motivated by mechanistic analysis and supported by empirical calibration, but it remains heuristic rather than fully causal. The transition-token signal is a practical proxy for reasoning saturation, yet it does not fully characterize reasoning completeness. A more principled and causal formulation remains an important direction for future work.
- **Limited exploration in RLVR.** We only study the transition signal as an inference-time stopping cue. Its potential use as a process-level supervision signal in RLVR, such as for intermediate reward shaping or reasoning-state guidance, is left unexplored due to time constraints and deserves further investigation.

8 Ethical Considerations

We use publicly available datasets and model checkpoints under licenses permitting research use; license terms and restrictions are summarized in Section 4. All artifacts are used in accordance with their intended purposes specified by the original providers. We also use an LLM only for writing assistance and language polishing.

References

Simon A Aytes, Jinheon Baek, and Sung Ju Hwang. 2025. Sketch-of-thought: Efficient llm reasoning with adaptive cognitive-inspired sketching. *arXiv preprint arXiv:2503.05179*.

Zhenni Bi, Kai Han, Chuanjian Liu, Yehui Tang, and Yunhe Wang. 2024. Forest-of-thought: Scaling test-time compute for enhancing llm reasoning. *arXiv preprint arXiv:2412.09078*.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025. Do not think that much for 2+3=? on the overthinking of o1-like llms. *Preprint*, arXiv:2412.21187.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, and 1 others. 2024. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2024. Token-budget-aware llm reasoning. *arXiv preprint arXiv:2412.18547*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Xin Liu and Lu Wang. 2025. Answer convergence as a signal for early stopping in reasoning. *arXiv preprint arXiv:2506.02536*.

Xiaoye Qu, Yafu Li, Zhaochen Su, Weigao Sun, Jianhao Yan, Dongrui Liu, Ganqu Cui, Daizong Liu, Shuxian Liang, Junxian He, and 1 others. 2025a. A survey of efficient reasoning for large reasoning models: Language, multimodality, and beyond. *arXiv preprint arXiv:2503.21614*.

Yuxiao Qu, Matthew YR Yang, Amrith Setlur, Lewis Tunstall, Edward Emanuel Beeching, Ruslan Salakhutdinov, and Aviral Kumar. 2025b. Optimizing test-time compute via meta reinforcement fine-tuning. *arXiv preprint arXiv:2503.07572*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Hanjie Chen, Xia Hu, and 1 others. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, and 1 others. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

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

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. Label words are anchors: An information flow perspective for understanding in-context learning. *arXiv preprint arXiv:2305.14160*.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, and 1 others. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Zihao Wei, Liang Pang, Jiahao Liu, Jingcheng Deng, Shicheng Xu, Zenghao Duan, Jingang Wang, Fei Sun, Xunliang Cai, Huawei Shen, and 1 others. 2025. Stop spinning wheels: Mitigating llm overthinking via mining patterns for early reasoning exit. *arXiv preprint arXiv:2508.17627*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.

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

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. 2025. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*.

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

Hai Ye, Mingbao Lin, Hwee Tou Ng, and Shuicheng Yan. 2024. Multi-agent sampling: Scaling inference compute for data synthesis with tree search-based agentic collaboration. *arXiv preprint arXiv:2412.17061*.

Jintian Zhang, Yuqi Zhu, Mengshu Sun, Yujie Luo, Shuofei Qiao, Lun Du, Da Zheng, Huajun Chen, and Ningyu Zhang. 2025a. Lightthinker: Thinking step-by-step compression. *arXiv preprint arXiv:2502.15589*.

Wenqi Zhang, Yongliang Shen, Linjuan Wu, Qiuying Peng, Jun Wang, Yueting Zhuang, and Weiming Lu. 2024. Self-contrast: Better reflection through inconsistent solving perspectives. *arXiv preprint arXiv:2401.02009*.

Xiaoying Zhang, Da Peng, Yipeng Zhang, Zonghao Guo, Chengyue Wu, Chi Chen, Wei Ke, Helen Meng, and Maosong Sun. 2025b. Will pre-training ever end? a first step toward next-generation foundation mllms via self-improving systematic cognition. *arXiv preprint arXiv:2503.12303*.

A Implementation Details

A.1 Inference Configuration

We conduct all inference experiments using the `transformers` library on a high-performance cluster node equipped with **8 × NVIDIA H800 GPUs**. To ensure reproducibility, we set the random seed to 5 across all runs. The specific hyperparameters for generation are listed below:

- **Decoding Strategy:** We utilize greedy decoding (`do_sample=False`) to eliminate randomness in the generation trajectory and ensure deterministic evaluation results.
- **Precision:** All models are loaded in `bfloat16` precision to balance computational efficiency and numerical stability.
- **Batch Size:** We set the inference batch size to 1 to simulate a real-world streaming latency scenario.
- **Max Generation Length:** The maximum number of new tokens is set to 8192 to accommodate the full reasoning chain of the baseline models.

A.2 Latency Measurement Protocol

To rigorously evaluate the efficiency of our proposed **SyncThink** method, we define the inference latency metric to include the full end-to-end processing time. Specifically, the reported time T_{total} is calculated as:

$$T_{total} = T_{gen} + T_{metric} + T_{eval} \tag{6}$$

where:

- T_{gen} denotes the raw model generation time.
- T_{metric} represents the computational overhead introduced by our method, including the real-time extraction and calculation of token logits and attention entropy.
- T_{eval} includes the time required for the final answer extraction and correctness verification.

This comprehensive measurement ensures that the reported speedup ($3.21\times$) reflects the net performance gain after accounting for all computational costs associated with our dynamic monitoring mechanism.

B Pareto Frontier Analysis of SyncThink

To examine the accuracy–efficiency trade-off, we sweep the truncation ratio of a fixed-ratio baseline and plot accuracy against the truncation ratio on four benchmarks. This sweep induces a set of candidate operating points, whose upper envelope forms an *empirical Pareto frontier* for uniform truncation policies. As shown in Figure 7, **SyncThink** consistently lies in the upper-left region relative to this frontier across all datasets, indicating a strictly better operating point: it achieves higher accuracy under the same truncation ratio, or equivalently reaches comparable accuracy with fewer reasoning tokens. Importantly, this gain cannot be replicated by simply selecting a different global truncation ratio. Instead, it stems from **SyncThink**'s instance-adaptive, logit-driven termination rule, which detects when additional reasoning becomes redundant and reallocates the reasoning budget more effectively than any single fixed-ratio strategy.

C The Universality of the Reasoning Bottleneck

In this section, we discuss the theoretical grounding of our method, arguing that the effectiveness of the `</think>` token is not an artifact of specific model training (e.g., DeepSeek-R1) but a manifestation of a universal mechanism in Large Language Models (LLMs).

The Attention Sink Hypothesis. Recent studies on efficient streaming LLMs have proposed the *Attention Sink* hypothesis (Xiao et al., 2023), which suggests that attention heads tend to allocate massive attention scores to specific "sink" tokens (often initial tokens or special separators) to maintain internal state stability and aggregate historical context. In the specific context of Chain-of-Thought (CoT) reasoning, the transition from a *reasoning phase* to an *answering phase* necessitates a semantic boundary—a token or sequence that compresses the preceding divergent rationale into a converged state for answer generation.

Function over Form. While DeepSeek-R1 explicates this boundary as `</think>`, we argue that other models employ implicit equivalents to fulfill the same functional role. For instance, standard base models often rely on linguistic markers such as "*Therefore*", "*So*", or "*Thus*" to signal this state shift. similarly, instruction-tuned models may utilize special tokens like `<|start_header_id|>`

or specific role separators. Consequently, the </think> token in our **SyncThink** framework essentially functions as a specialized *Reasoning Attention Sink*. It signals the phase shift from a high-entropy *exploration state* to a low-entropy *determination state*.

This theoretical perspective implies that **SyncThink** is generalizable to a broader range of CoT-capable models. The core methodology—detecting the saturation of reasoning information via a bottleneck token—remains valid, provided one identifies the model-specific transition token (the "sink"), which distinguishes itself only in surface form rather than functional utility.

D Comparison with DEER under Serial Execution

We additionally compare SyncThink with DEER under a purely serial execution setting. While DEER achieves stronger accuracy in our main-table comparison, its reported latency depends heavily on a more complex inference pipeline, including trial-answer generation, rollback, and branch-parallel / overlapped execution. In contrast, SyncThink is strictly single-pass and plug-and-play: it only monitors standard decoding logits (transition-token rank and entropy) and triggers an early exit without trial-answer generation or rollback.

To make the latency comparison more apples-to-apples, we implement a serial variant of DEER based on the paper description, where trial-answer generation and rollback are executed sequentially without branch overlap. Table 4 reports the measured end-to-end latency under the same evaluation protocol as our main experiments. Numbers in parentheses denote the runtime ratio relative to SyncThink on the same model and benchmark.

The results show that, under serial execution, DEER becomes substantially slower than SyncThink across all three models, with a consistent slowdown of around $1.6\times$–$1.8\times$. This highlights an important practical distinction between the two approaches: DEER improves accuracy by spending additional test-time compute through trial answers and rollback, whereas SyncThink is an online stopping policy that reads an intrinsic readiness signal from normal decoding and exits early in a single pass. Notably, on harder benchmarks such as GPQA, SyncThink remains particularly competitive and can outperform DEER in our main-table comparison, suggesting that the two methods follow different technical routes to improve reasoning efficiency.

Model	GSM8K	MMLU	GPQA	BBH	Avg
DeepSeek-R1-Qwen2.5-7B	25.25 (×1.75)	48.51 (×1.72)	79.51 (×1.67)	40.00 (×1.64)	48.32
DeepSeek-R1-Qwen2.5-14B	54.71 (×1.73)	107.60 (×1.75)	223.72 (×1.77)	103.86 (×1.77)	122.47
DeepSeek-R1-LLaMA3.1-8B	33.92 (×1.71)	73.65 (×1.76)	126.66 (×1.66)	63.90 (×1.67)	74.53

Table 4: End-to-end latency of DEER under a purely serial execution setting (no branch-parallel / overlapped execution). Numbers in parentheses denote the runtime ratio relative to SyncThink on the same benchmark.

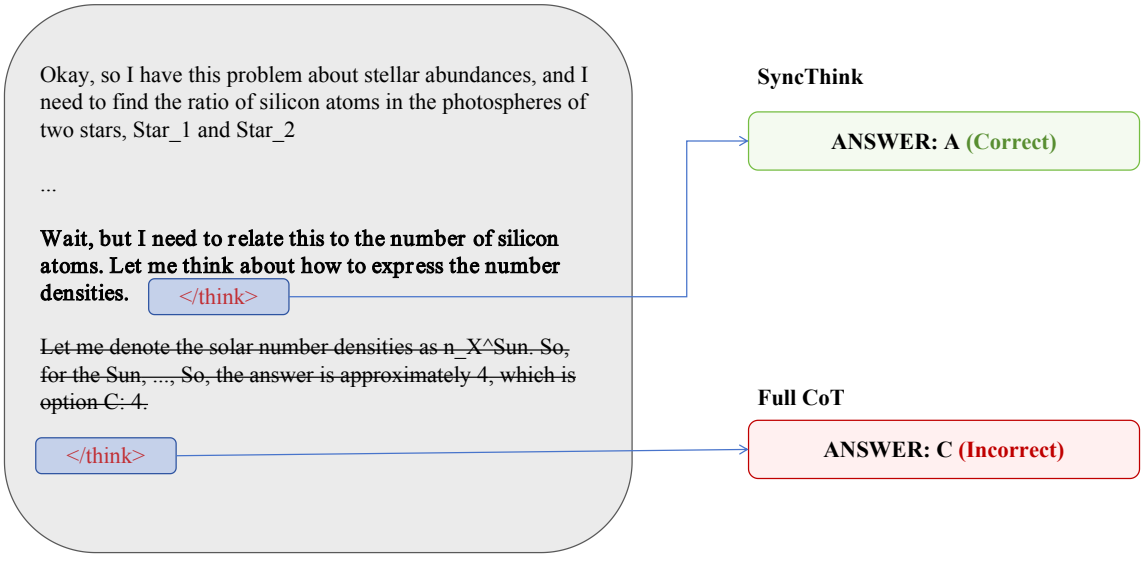
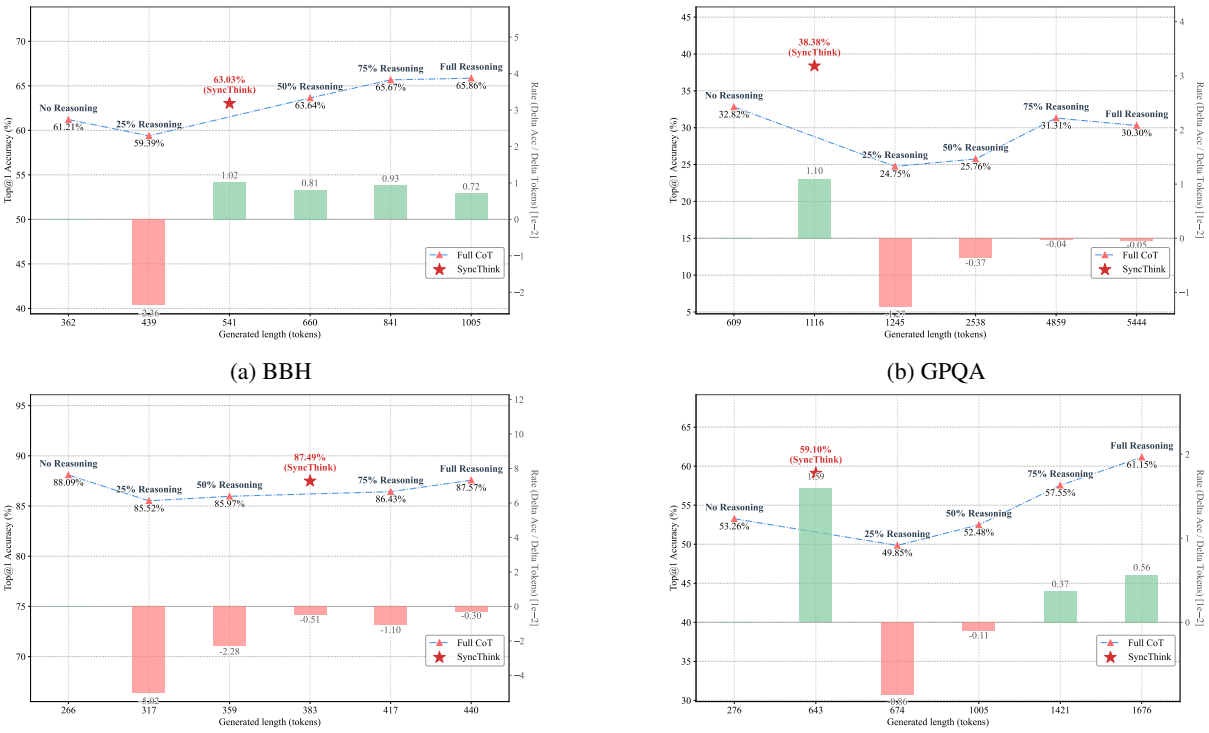


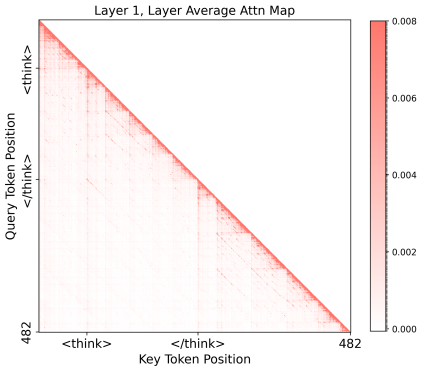
Figure 6: Case study on preventing model over-thinking. While the Full CoT path drifts into erroneous reasoning after a correct intermediate state, our SyncThink method detects the onset of this detrimental redundancy. By truncating the generation early, our method successfully avoids the subsequent error and retains the correct answer.



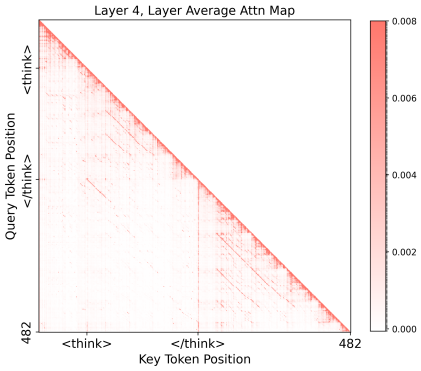
(a) BBH (b) GPQA

(c) GSM8K (d) MMLU

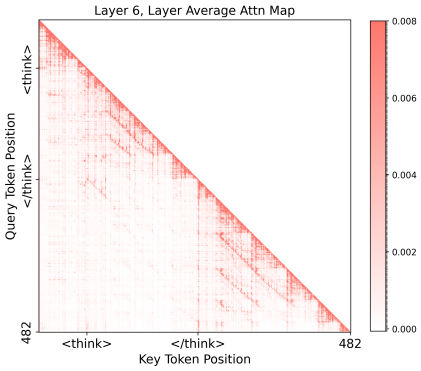
Figure 7: **Accuracy–length trade-off of SyncThink on DeepSeek-R1-Distill-Qwen-7B.** Blue curves denote fixed-ratio truncation baselines on full CoT traces, while the red star marks **SyncThink**. Across BBH, GPQA, GSM8K, and MMLU, **SyncThink** consistently lies in the upper-left region, achieving higher accuracy with fewer generated tokens, with the largest gain on GPQA.



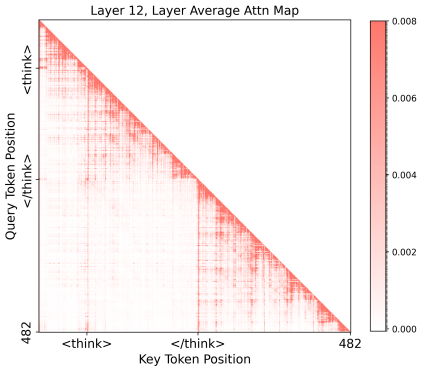
(a) Layer 1



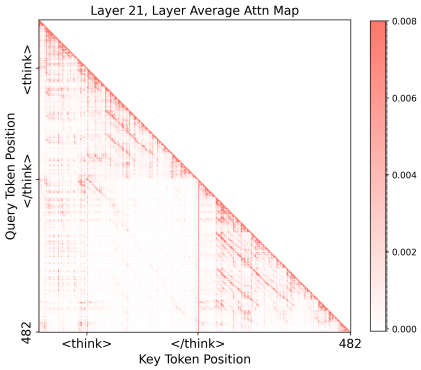
(b) Layer 4



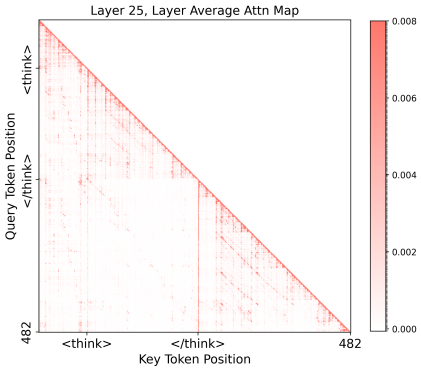
(c) Layer 6



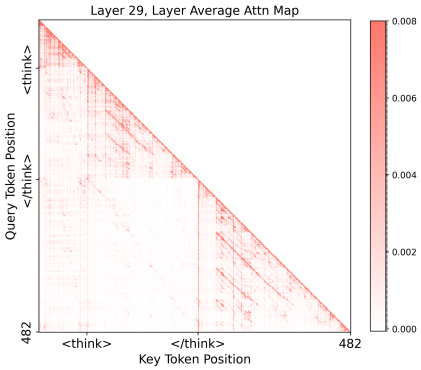
(d) Layer 12



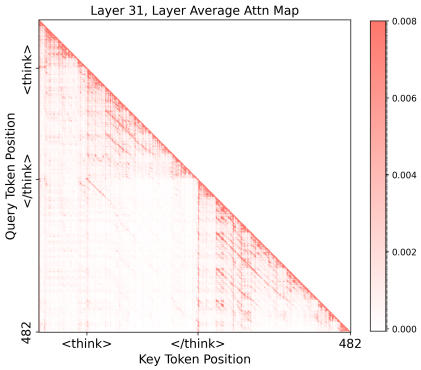
(e) Layer 21



(f) Layer 25



(g) Layer 29



(h) Layer 31

Figure 8: Attention heatmaps across different layers of DeepSeek-R1-Distill-LLaMA-8B on a GSM8K sample (Cobbe et al., 2021). Tokens within the <think>...</think> span receive uniform attention in early layers, but deeper layers gradually shift focus to the boundary tokens, indicating information migration and compression of reasoning content. Similar observations can be found in other models and datasets