TRANSITION THINKING: TRAINING-FREE PREDICTION ARBITRATION FOR REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Training-free methods such as Soft Thinking seek latent reasoning in a frozen model by feeding probability-weighted mixtures of token embeddings back into the Transformer. However, their advantage over chain-of-thought is inconsistent across models and benchmarks. We find that the soft mixture usually stays near its dominant token embedding and rarely changes the next generated tokens, so Soft Thinking behaves much like argmax decoding. This helps explain why its gains vary. Argmax trajectories can be as accurate as sampled ones when they finish but are vulnerable to repetition loops. Soft Thinking terminates looping generations with Cold Stop, yet these generations are often incorrect. Layerwise next-token predictions reveal a transition at a specific intermediate layer, where entropy falls sharply while predictions still differ from the final layer. At this layer, hesitation and repetition loops produce opposite confidence patterns in the two layers. When the final layer hesitates, the transition layer can hold a confident alternative, whereas inside a loop the final layer is nearly deterministic and the transition layer is diffuse. We propose *Transition Thinking*, which keeps argmax decoding for thinking, arbitrates hesitations with the transition-layer prediction, and controls repetition with a thinking-time penalty and an in-place escape from verified loops. The transition layer is selected once per model from an unlabeled profile, and answer generation is unchanged. Across three Qwen3 models, Transition Thinking improves average accuracy over chain-of-thought on mathematics and science by 0.5 to 2.3 points, stays close to it on coding tasks, and shortens correct generations. It requires no training, additional forward passes, or task-specific layer search.

1 INTRODUCTION

Large language models (LLMs) based on the Transformer (Vaswani et al., 2017) now solve increasingly difficult reasoning tasks (Guo et al., 2025; Yang et al., 2025; Muennighoff et al., 2025), and chain-of-thought (CoT) reasoning further improves this ability by expressing intermediate computation as natural-language steps (Wei et al., 2022; Kojima et al., 2022; Wang et al., 2022). Because each generated token becomes the next input, the model commits to one token at a time and discards the other candidates of its next-token distribution.

Latent reasoning methods relax this commitment by carrying intermediate computation in continuous representations (Deng et al., 2023; Cheng & Van Durme, 2024; Xu et al., 2025a; Tang et al., 2026). For example, COCONUT (Hao et al., 2024) feeds hidden states back into the model instead of emitting each reasoning step as discrete tokens. However, the model must be trained to produce and read such representations, which limits their use with already pretrained reasoning models. Training-free methods avoid this requirement by feeding soft inputs to a pretrained model at inference time (Fu & Luo, 2026; Shi et al., 2026). Soft Thinking (Zhang et al., 2026c) replaces the next discrete thinking token with a probability-weighted mixture of the top-k token embeddings, and Stochastic Soft Thinking (Wu et al., 2026) adds randomness to this mixture through schemes such as Gumbel-Softmax. However, their gains are reported mainly on relatively large models, and the reference implementation cautions that smaller models may benefit less from soft inputs.[1]

[1] `github.com/UCSB-AI/Soft-Thinking`, Reproduction Instructions.

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.

To understand this variation, we compare the Soft Thinking input with the discrete embedding used by standard decoding. Prior analyses show that a soft input yields nearly the same next-token prediction (Wu et al., 2026) and hidden states (Rizvi-Martel et al., 2026) as its top-1 token. We find that this similarity also holds at the generation level, and that its extent is set by the uncertainty of the top candidate, which is zero or small on most thinking steps (Section 3.1).

Because the mixture is built from the final distribution, it also does not directly use signals in intermediate layers that are absent from the final prediction. The resulting argmax-like trajectory is vulnerable to persistent repetition (Holtzman et al., 2019; Pipis et al., 2025). These two limitations raise two questions: whether an intermediate-layer prediction can improve token selection during reasoning, and how a trajectory can recover once repetition has begun.

We therefore examine how next-token predictions develop across layers by applying the output head to each intermediate hidden state (nostalgebraist, 2020). Across Qwen3 model sizes, these predictions become sharply more confident at a specific intermediate layer, which we call the *transition layer*. At some steps, the transition layer already favors a clear token while the final layer remains uncertain. Inside a repetition loop, the pattern reverses: the final layer is nearly deterministic, while the transition-layer prediction is diffuse. This contrast suggests that token selection and loop recovery need separate mechanisms, as we show in Section 3.

Based on these observations, we propose *Transition Thinking*, which builds on argmax decoding and adds two mechanisms. Hesitation arbitration mixes the transition-layer and final predictions when the transition-layer prediction is confident and the final layer hesitates. Loop recovery applies a repetition penalty during thinking and escapes verified loops in place without ending the thinking phase. The transition layer is selected once per model from an unlabeled entropy profile, and the method requires no training, additional forward passes, or change to answer generation (Section 4). Across Qwen3-4B, 8B, and 14B, Transition Thinking exceeds the overall average accuracy of CoT and shortens correct generations by 5.2 to 6.5%. On mathematics and science, its average accuracy rises by 0.5 to 2.3 points.

Our contributions are threefold:

- Through a geometric analysis of the soft input, we show that its departure from the argmax embedding is set by the uncertainty of the top candidate and that the resulting argmax-like decoding loses accuracy mainly through repetition loops (Sections 3.1 and 3.2).
- We identify a transition layer where hesitation and repetition loops behave oppositely (Section 3).
- We propose Transition Thinking, which combines hesitation arbitration at the transition layer with loop recovery, and measure each component by ablation (Sections 4 and 5.3).

2 Related work

Latent reasoning. Trained methods feed hidden states or learned continuous tokens back into the model (Hao et al., 2024; Tan et al., 2026a; Cheng & Van Durme, 2024; Xu et al., 2025a). Among training-free methods, Soft Thinking feeds back a probability-weighted mixture of token embeddings (Zhang et al., 2026c). Wu et al. (2026) find that models largely follow the top-1 token of this mixture and propose Stochastic Soft Thinking, which randomizes it through Dirichlet sampling or Gumbel-Softmax. SeLaR uses soft embeddings only at high-entropy steps (Fu & Luo, 2026), SwiReasoning switches between latent and explicit reasoning (Shi et al., 2026), and Multiplex Thinking merges sampled tokens and is trained with reinforcement learning (Tang et al., 2026). Rizvi-Martel et al. (2026) find signs of superposition only in models trained from scratch on latent thoughts. Our method instead feeds back a discrete token and changes only which token is selected.

Intermediate-layer decoding. The logit lens and tuned lens decode predictions from intermediate hidden states (nostalgebraist, 2020; Belrose et al., 2023). Contrastive Decoding, DoLa, and SLED refine the final prediction against a smaller model or earlier layers (Li et al., 2023b; Chuang et al., 2024; Zhang et al., 2024). Entropy-Lens tracks the entropy of intermediate predictions across layers (Ali et al., 2025), DEGS scores sampled trajectories by the depth of entropy collapse (Meng et al., 2026), and LED selects high-entropy depths to restore exploration (Tan et al., 2026b). Confident Layer Decoding searches backward at each token for a confident near-final layer (Zhang et al., 2026b), and early-exit methods use intermediate predictions to skip computation (Schuster et al.,

2022; Elhoushi et al., 2024). Unlike these methods, we fix one transition layer per model and use its logits when the final layer hesitates and the transition-layer prediction is confident.

Layer selection and intervention. Prior interventions choose layers or heads by sweeps or probe accuracy (Zou et al., 2023; Li et al., 2023a), learn input-dependent selection (Gadgil et al., 2026), or use hidden-state norms (Zhang et al., 2026a). Hidden-state probes can diagnose reasoning errors without providing a causal control signal (Yuan et al., 2026a), while interpretability-guided interventions on latent vectors improve latent reasoning (Chang et al., 2026). Our rule fixes one mid-depth layer per model without labels or an accuracy sweep instead of choosing a layer at each token.

Repetition and thinking length. Greedy decoding falls into self-reinforcing repetition loops (Holtzman et al., 2019; Xu et al., 2022; Fu et al., 2021). Reasoning models also loop at low temperature, and raising the temperature reduces looping without removing its cause (Pipis et al., 2025). Generations that exhaust the token budget are far less accurate than those that finish (Oladri et al., 2026). Other methods control thinking length through token budgets, progress representations, or halt steering directions (Muennighoff et al., 2025; Li et al., 2026; Eisenstadt et al., 2025; Jayabahu & Adeleke, 2026). Word Salad Chopper detects repeated chunks with a linear probe on hidden states, truncates them, and prompts regeneration (Xie et al., 2025), and DTS favors shorter completed branches of a sketched reasoning tree (Xu et al., 2025b). Our recovery uses no learned probe and removes no tokens. It adds randomness for 64 steps only after a loop is verified and forces the end of thinking only when a loop is verified again after three escapes.

3 The Prediction Transition

3.1 What Final-Layer Feedback Carries

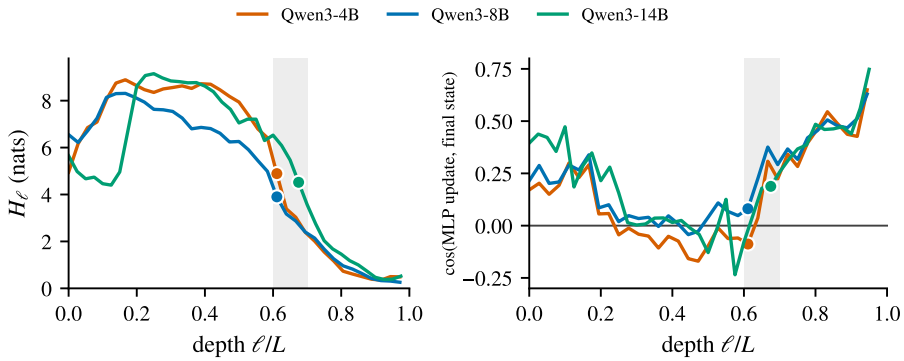
Soft Thinking (Zhang et al., 2026c) and Stochastic Soft Thinking (Wu et al., 2026) feed back a probability-weighted mixture of the top-k candidate embeddings instead of a single token. Prior work shows that a forward pass on this mixture behaves like one on its top-1 token (Wu et al., 2026; Rizvi-Martel et al., 2026). Through a geometric analysis, we show that the mixture itself usually lies close to its top-1 embedding. We call the displacement of the mixture from the argmax embedding the *payload*, and its size in units of the median embedding norm $\bar{n}$ factorizes into the uncertainty $1 - p_1$ and a geometry term D_t, the normalized distance from the argmax embedding to the mean of the other candidates. Here p_1 is the renormalized probability of the top candidate, so the uncertainty $1 - p_1$ is the probability left for all other candidates.

$$\|r_t\|/\bar{n} = (1 - p_1)\, D_t \tag{1}$$

The geometry term stays of order one, with medians of 0.94 and 1.32 in Qwen3-1.7B and Qwen3-8B, so a small payload reflects low uncertainty rather than close candidate embeddings. To measure the effect of the payload, we extend the single-step intervention of Rizvi-Martel et al. (2026) and take a checkpoint every 50 thinking steps on Qwen3-1.7B and Qwen3-8B. The uncertainty is exactly zero on 42–48% of these checkpoints, and feeding the mixture instead of the argmax embedding leaves the next 20 generated tokens unchanged on 81–84% of those with a nonzero payload. This indicates that the mixture rarely passes information beyond its top-1 token. Moreover, like argmax, Soft Thinking repeats long thinking prefixes across runs and ends in repetition loops. A soft input built from the final-layer distribution is therefore nearly the argmax embedding, and Soft Thinking behaves much like argmax decoding. Appendix A derives Eq. 1 and reports the full measurements.

3.2 Repetition Loops in Argmax Decoding

Because Soft Thinking behaves much like argmax decoding, it inherits a known weakness of deterministic decoding, namely self-reinforcing repetition loops (Holtzman et al., 2019; Pipis et al., 2025). On Qwen3-8B AIME24–26, argmax exhausts the token budget about twice as often as sampled CoT, and most of the extra truncation ends in literal repetition. However, argmax generations that finish are slightly more accurate on average, at 76.3% against 75.2%. This suggests that argmax tends to lose to sampling through loops rather than through less accurate reasoning, so recovering from loops can raise its accuracy. Soft Thinking instead ends such low-entropy states with Cold Stop (Zhang et al., 2026c). Although ST finishes 53 of its 56 generations with repeated tails, only 21%



(a) Intermediate-prediction entropy (b) MLP update vs. final prediction

Figure 1: Prediction transition across three Qwen3 checkpoints. (a) Dots mark ℓ^*, the largest decrease in mean intermediate-prediction entropy, selected once per model from unlabeled thinking tokens and located at 0.61–0.68 of model depth (Eq. 3). (b) Cosine between each MLP update and the normalized final state, near zero in the middle layers and positive and growing after ℓ^*.

Table 1: Intermediate predictions around ℓ^* on six AIME24 traces per model, 2,560 positions each. Confident is the rate of entropy below one nat, and Differs is how often it misses the final top-1.

	Qwen3-4B		Qwen3-8B		Qwen3-14B	
Layer	Confident	Differs	Confident	Differs	Confident	Differs
$\ell^* - 6$	0.3	93	1.2	96	3.5	99
$\ell^* - 1$	1.8	76	4.6	88	6.1	85
ℓ^*	**5.9**	**51**	**10.1**	**77**	**9.7**	**80**
at final hesitation	2.9	65	8.1	93	7.1	89
$\ell^* + 1$	16.8	46	18.8	70	16.6	55
$\ell^* + 6$	60.2	34	55.4	48	58.7	37
$L - 2$	88.1	8	88.7	9	89.5	6

of the 56 are correct, against 70% without repeated tails. Therefore, a loop should be exited so that reasoning can continue rather than cut off, which motivates the loop recovery of Section 4.2. However, loop recovery alone cannot give argmax information beyond the final distribution, and the next subsection addresses this limitation. Appendix E details truncation and accuracy per benchmark.

3.3 The Transition Layer

To find information beyond the final distribution, we examine how next-token predictions form across layers. We read each intermediate hidden state with the final normalization N and the LM head W_U, as in the logit lens (nostalgebraist, 2020). The hidden state after layer ℓ at thinking position t is $h_{t,\ell}$, and the subscript F denotes the final layer $L-1$. On unlabeled thinking tokens from six AIME24 traces, we select the layer with the largest drop in the mean prediction entropy $\mathcal{H}_\ell$, taking the shallower layer in a tie.

$$z_{t,\ell} = W_U N(h_{t,\ell}) \qquad p_{t,\ell} = \mathrm{softmax}(z_{t,\ell}) \tag{2}$$

$$\ell^* = \arg\max_{1 \le \ell < L-1} \left(\mathcal{H}_{\ell-1} - \mathcal{H}_\ell\right) \tag{3}$$

Figure 1a shows that entropy stays high through the middle layers and then drops sharply in all three models. Accordingly, the rule selects layers 22, 22, and 27 of Qwen3-4B, 8B, and 14B, at 61–68% of depth, without correctness labels or an accuracy sweep. Moreover, each layer adds an attention update and an MLP update to its hidden state, and the MLP update changes direction at the transition. Figure 1b shows that this update is nearly orthogonal to the normalized final state through the middle layers, with mean cosine between -0.07 and 0.03, and that after ℓ^* it stays positive at every later layer and grows with depth. Although this signal plays no role in selection, it is consistent with the entropy transition. Appendix B details the calibration profiles and the MLP analysis.

However, locating the transition is not enough for arbitration, which also needs the transition-layer prediction to be confident and to differ from the final prediction. Confident denotes the share of positions whose prediction has entropy below one nat, and Differs denotes the share of these confident predictions whose top-1 differs from the final top-1. As shown in Table 1, confident predictions are

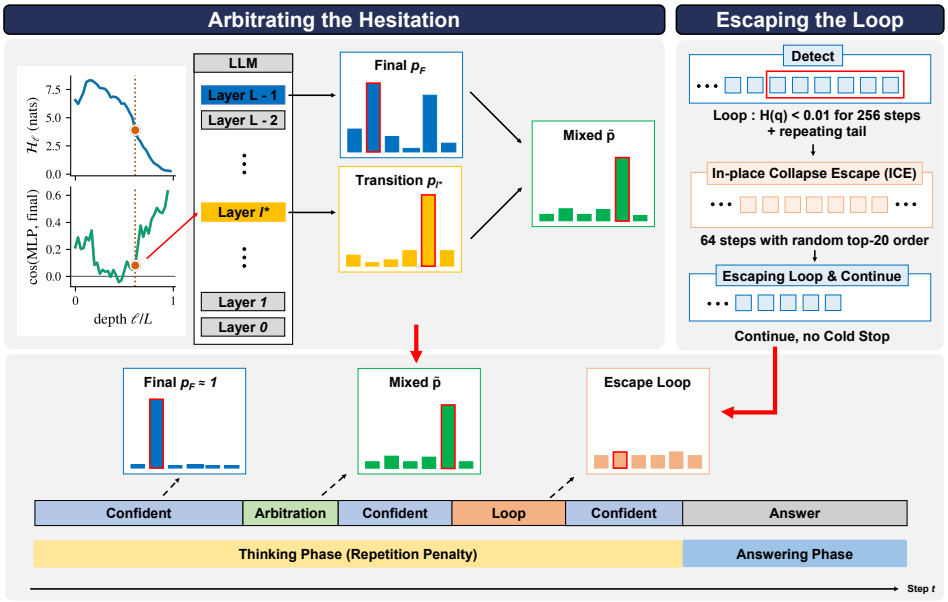


Figure 2: **Transition Thinking.** *Left:* when the final prediction hesitates and the ℓ^* prediction is confident, the gate g mixes the two logits. *Right:* a loop is suspected when the entropy of the penalized next-token distribution at temperature 0.6 stays low. After a verified loop ICE selects 64 tokens from the final top-20 and continues without rewinding. *Bottom:* one thinking trajectory under the penalty.

rare before ℓ^*, while later layers increasingly repeat the final top-1. At ℓ^*, 5.9–10.1% of positions are confident and 51–80% of those differ from the final top-1, against 6–9% near the output. This indicates that ℓ^* is where confident alternatives appear before intermediate predictions become redundant with the final one. We use these alternatives to resolve hesitant decisions of the final layer, and Section 5 shows that doing so improves accuracy.

The transition layer and the final layer also behave oppositely in the two states of argmax decoding. When the final layer hesitates, the transition layer can hold a confident alternative. Counting only positions where the final top-1 probability is below 0.5, the transition layer is confident on 2.9 to 8.1% of them, and 65 to 93% of these confident predictions differ from the final top-1. Inside a loop, however, the final layer puts almost all its mass on the repeated token, while the transition-layer prediction is diffuse with entropy near 6.6 nat on Qwen3-8B. Because the arbitration weight scales with final-layer uncertainty, it is then close to zero and cannot break the loop. We therefore use the transition layer only for arbitration and handle loops in Section 4.2.

4 Methodology

Building on these observations, *Transition Thinking* follows a single argmax thinking trajectory and adds two mechanisms, which are described in the following subsections. Hesitation arbitration lets the transition-layer prediction influence small-margin decisions (Section 4.1), and loop recovery combines a repetition penalty with In-place Collapse Escape (ICE), which exits the loops that still form without ending the thinking phase (Section 4.2). Figure 2 shows both mechanisms on one thinking trajectory, and Algorithm 1 in Appendix C gives the complete decoding rule. The procedure uses no additional Transformer forward pass.

4.1 Hesitation Arbitration

The transition-layer prediction participates only when its entropy is below a fixed threshold of one nat, and its weight scales with final-layer uncertainty. We define the gate and the interpolated logits.

$$g_t = \frac{1}{2}\big(1 - \max_v p_{t,F}(v)\big)\,\mathbf{1}\big[H_{t,\ell^*} < 1\big] \tag{4}$$

$$\widetilde{z}_t = (1 - g_t)\, z_{t,F} + g_t\, z_{t,\ell^*} \qquad x_{t+1} = \arg\max_v R_\rho\big(\widetilde{z}_t\big)(v) \tag{5}$$

The indicator asks whether the transition-layer prediction has formed a local decision. The continuous factor approaches zero when the final layer is confident, so a settled final decision is not replaced merely because an intermediate layer is sharp. Here R_ρ is the repetition penalty of Section 4.2, and the next step receives the ordinary embedding of x_{t+1}. With $g_t = 0$ and $\rho = 1$, the rule reduces exactly to final-layer argmax decoding. With $\rho = 1$, arbitration changes the argmax token only if the weighted transition-layer preference exceeds the final-layer margin. We use the same one-nat threshold and factor $1/2$ for every model and benchmark.

4.2 Loop Recovery

As discussed in Section 3.3, arbitration cannot break an established loop, because its weight is close to zero once the final layer becomes nearly one-hot. To address this limitation, we apply the repetition penalty of Keskar et al. (2019) in the sign-aware form used by common inference libraries, with $\rho = 1.05$ at every thinking step, and switch it off for the answer. The penalty R_ρ in Eq. 5 divides the interpolated logit of every token that has already appeared in the output by ρ when positive and multiplies it by ρ when negative. Inside a loop every token of the repeated span is penalized, so a competing candidate can overtake the repeated token and the trajectory continues from the same prefix. However, a strongly preferred repeated token can still exceed its penalized competitors, so the penalty alone cannot exit every loop.

We therefore add ICE, which acts only after a loop has been verified. The right panel of Figure 2 illustrates this procedure. ICE tracks the entropy of the penalized next-token distribution and suspects a loop when it stays below 0.01 nat for 256 consecutive thinking steps. This entropy is computed at temperature 0.6, while tokens are still chosen by argmax. It then checks for a repeating pattern with a period p of at most 1,024 tokens by comparing the latest 256 or 512 tokens with the tokens p positions earlier. A loop is verified if at least 90% of these pairs match for some p, as in `ABCABC...` with period $p = 3$. If this check fails the counter restarts, and a fourth failed check switches the detector off. After a verified loop, ICE picks each of the next 64 tokens at random from the final top-20 candidates with the repetition penalty still applied and then returns to arbitration without rewinding. At most three escapes are allowed, and a fourth verified loop ends thinking with `</think>`, the only case in which an answer is forced. Unlike Cold Stop, low entropy alone never ends thinking.

5 Experiments and Results

5.1 Experiment Setup

Models. We evaluate Qwen3-4B, Qwen3-8B, and Qwen3-14B (Yang et al., 2025) in thinking mode, with 36, 36, and 40 layers. Three sizes of one open reasoning family test whether the transition layer and its effect hold across scale under a fixed architecture.

Benchmarks. We cover mathematical reasoning, scientific question answering, and code generation. Mathematics uses AIME 2024, 2025, and 2026 (Mathematical Association of America, 2026), and science uses GPQA-Diamond (Rein et al., 2023). Code uses HumanEval+ (Liu et al., 2023), the sanitized MBPP test set (Austin et al., 2021), and LiveCodeBench release v5 (Jain et al., 2025).

Baselines. We compare against sampled chain-of-thought (CoT; Wei et al., 2022), Soft Thinking (ST; Zhang et al., 2026c), and Stochastic Soft Thinking (SST; Wu et al., 2026). For a fair comparison, all methods are training-free and run in the same engine. CoT samples its thinking tokens with temperature 0.6, top-p 0.95, and top-k 30, and ST and SST use their reference settings, including Cold Stop.

Transition Thinking. TT combines hesitation arbitration at ℓ^* with the repetition penalty and ICE of Section 4.2 on top of argmax thinking, and it does not use Cold Stop. The transition layer is selected once per model by Eq. 3, giving layer 22 for Qwen3-4B and Qwen3-8B and layer 27 for Qwen3-14B, and is kept fixed across all tasks.

Table 2: Accuracy and length of correct generations on mathematics and science to the left of the gray line and on coding to the right. Avg. is the overall mean, the values beside it show the change from CoT, and bold and underline mark the best and second-best averages.

		AIME 2024	AIME 2025	AIME 2026	GPQA Diamond	Human Eval+	MBPP	LiveCode Bench	Avg.	
Qwen3-4B										
CoT	Accuracy	72.5	63.8	60.0	52.0	88.8	92.2	57.0	69.5	
	Gen. Length	13.1	15.5	14.1	6.2	3.7	2.9	8.5	9.1	
ST	Accuracy	69.2	61.3	58.3	50.0	87.4	88.5	48.7	66.2	(−3.3)
	Gen. Length	13.2	14.3	13.4	5.9	3.1	2.4	7.9	**8.6**	(−5.9%)
SST	Accuracy	73.3	64.6	62.5	55.6	89.3	92.5	53.8	<u>70.2</u>	(+0.8)
	Gen. Length	13.4	14.2	13.5	6.5	3.4	2.8	8.7	8.9	(−2.3%)
TT (ours)	Accuracy	77.9	59.6	65.0	54.9	90.5	91.1	54.5	**70.5**	(+1.0)
	Gen. Length	11.8	14.9	13.8	6.4	3.4	2.3	8.1	<u>8.7</u>	(−5.2%)
Qwen3-8B										
CoT	Accuracy	73.8	66.7	69.6	58.2	76.8	94.9	59.5	<u>71.4</u>	
	Gen. Length	13.3	14.3	15.4	8.2	3.9	3.0	9.3	9.6	
ST	Accuracy	73.3	66.2	60.0	56.2	75.8	92.8	55.2	68.5	(−2.9)
	Gen. Length	12.8	15.3	13.8	7.6	3.3	2.7	8.2	<u>9.1</u>	(−5.5%)
SST	Accuracy	77.9	67.5	63.7	58.2	76.8	94.9	58.8	71.1	(−0.2)
	Gen. Length	13.3	14.8	13.9	8.2	3.4	2.9	9.0	9.4	(−2.8%)
TT (ours)	Accuracy	75.4	72.9	67.1	61.9	77.5	93.4	59.9	**72.6**	(+1.2)
	Gen. Length	12.8	14.3	13.7	7.3	3.4	2.7	8.8	**9.0**	(−6.5%)
Qwen3-14B										
CoT	Accuracy	82.9	71.7	72.1	59.6	94.5	95.8	65.2	<u>77.4</u>	
	Gen. Length	13.0	13.9	14.3	5.3	3.1	2.2	9.1	8.7	
ST	Accuracy	78.8	68.8	70.8	57.1	89.6	95.4	57.0	73.9	(−3.5)
	Gen. Length	12.6	13.1	13.3	5.1	2.8	2.3	7.7	**8.1**	(−6.6%)
SST	Accuracy	80.8	69.2	70.8	59.6	92.5	96.0	59.5	75.5	(−1.9)
	Gen. Length	13.1	13.5	13.3	5.2	2.8	2.2	7.9	8.3	(−4.8%)
TT (ours)	Accuracy	82.5	71.7	73.8	60.5	96.1	96.2	65.9	**78.1**	(+0.7)
	Gen. Length	12.0	13.5	13.1	6.1	2.6	2.0	8.3	<u>8.2</u>	(−5.4%)

Decoding and metrics. Every method has a budget of 32,768 generated tokens and uses the same answer sampler after `</think>`, with temperature 0.6, top-p 0.95, and top-k 20. We draw eight generations per problem and average accuracy over generations and then over problems (avg@8). We note that decoders that do not sample thinking tokens still diverge across repeated runs, possibly due to numerical variation in batched bf16 inference (Yuan et al., 2026b), which we have not isolated. Following Zhang et al. (2026c), generation length is the mean number of generated tokens over correct generations, reported in thousands of tokens, and math and GPQA are graded by answer match and code by the reference tests. Appendix D details the full experimental settings.

5.2 Main Results

Accuracy. Table 2 compares all methods with CoT at three model sizes. Since CoT is the baseline that training-free latent reasoning methods commonly aim to surpass, we focus on how each method changes the overall average relative to CoT. First, ST lowers the overall average of CoT at every size, by 3.3, 2.9, and 3.5 points, which agrees with the analysis in Section 3.1 that the soft input behaves like argmax decoding. SST adds Gumbel noise to the mixture and thereby departs from argmax-like behavior, which gives it a higher average than ST at every size. However, SST still changes the CoT average by $+0.8$, -0.2, and -1.9 points, so its effect depends on the model size. In contrast, TT consistently surpasses CoT at every size, raising the overall average from 69.5 to 70.5 on Qwen3-4B, from 71.4 to 72.6 on Qwen3-8B, and from 77.4 to 78.1 on Qwen3-14B. The gains are particularly large on mathematics and science, where TT raises the CoT average by 2.3, 2.2, and 0.5 points. On coding tasks, TT also stays within 0.6 points of CoT and surpasses it on Qwen3-14B, which shows that its gains are not limited to a single domain. Overall, TT is the only evaluated method that surpasses CoT at every size.

Efficiency. TT also shortens correct generations while improving accuracy. Specifically, its overall average length is 5.2, 6.5, and 5.4% shorter than that of CoT on Qwen3-4B, 8B, and 14B, and this reduction holds consistently across all three sizes and both task domains. By domain, the reduction on coding tasks, at 8.6, 8.0, and 10.4%, is larger than that on mathematics and science, at 4.1, 6.1, and 3.9%. ST and SST also shorten correct generations relative to CoT, but both end thinking with

Table 3: Truncated and finished generations on AIME24–26, 720 per decoder and model. A generation is truncated when it exhausts the token budget, and Repeated span counts generations that contain a repeated span as defined in Appendix E.

Decoder	*Qwen3-8B*				*Qwen3-14B*			
	Truncated (%)	Acc. when finished (%)	Repeated span	Overall acc. (%)	Truncated (%)	Acc. when finished (%)	Repeated span	Overall acc. (%)
CoT	6.9	75.2	19	70.0	5.8	80.5	13	75.6
Argmax	13.8	76.3	97	66.7	7.2	81.0	34	75.3
TT	**4.4**	75.0	38	**71.8**	**3.6**	78.8	19	**76.0**
w/o arbitration	6.5	74.6	59	70.1	3.8	74.9	25	72.1

Table 4: Arbitration layer, with the rest of TT fixed (avg@8). Depth is the layer index divided by the number of layers. Best results are highlighted in **bold**.

Depth	*Qwen3-8B*			*Qwen3-14B*		
	ℓ	AIME25	GPQA-Diamond	ℓ	AIME25	GPQA-Diamond
$\approx$0.50	18	67.1	58.6	20	66.7	59.5
ℓ^*	22	**72.9**	**61.9**	27	**71.7**	**60.5**
$\approx$0.72	26	68.8	56.9	29	**71.7**	59.8
$\approx$0.95	34	64.6	57.1	38	70.4	59.5

Cold Stop once the entropy stays low, and ST loses 2.9 to 3.5 points of average accuracy in the process. In contrast, TT does not use Cold Stop, yet it still produces shorter generations. Overall, TT is the only evaluated method that is both more accurate and shorter than CoT at every size.

Analysis. To understand where the gains of TT come from, Table 3 separates the AIME24–26 generations of Qwen3-8B and 14B into truncated and finished ones. First, finished argmax generations are as accurate as or more accurate than those of CoT on both models, at 76.3% and 81.0% against 75.2% and 80.5%. However, argmax falls into repeated spans far more often than CoT, so it exhausts the token budget in 13.8% of generations on Qwen3-8B, twice the 6.9% of CoT, and in 7.2% on Qwen3-14B against 5.8% for CoT. This indicates that the loss of argmax stems mainly from loops rather than from the quality of its reasoning, and the accuracy drop of ST discussed above largely shares this cause. In contrast, TT keeps the argmax trajectory and exits loops in place, so it shows the lowest truncation on both models, at 4.4% and 3.6%, and contains far fewer repeated spans than argmax. As a result, TT reaches the highest overall accuracy on both models. Interestingly, the role of arbitration differs between model sizes. On Qwen3-8B, arbitration further reduces truncation from 6.5% to 4.4% and complements loop recovery. In contrast, Qwen3-14B already keeps truncation low at 3.8% without arbitration, and arbitration there instead raises the accuracy of finished generations from 74.9% to 78.8%. This suggests that arbitration not only reduces loops but can also improve the reasoning itself by correcting the hesitant decisions of the final layer with the transition-layer signal. Moreover, fewer repeated spans also mean fewer unnecessary tokens, which explains why TT produces shorter correct generations without Cold Stop. Overall, loop recovery lets argmax generations reach the end, and arbitration improves the decisions along the way, so the two components together yield higher accuracy than CoT with shorter generations.

5.3 Ablation Study

Layer sweep. To verify that the label-free rule selects an appropriate transition layer, Table 4 moves the arbitration layer from ℓ^* to layers at about 50%, 72%, and 95% of depth while keeping the rest of TT fixed. On Qwen3-8B, ℓ^* achieves the highest accuracy on both benchmarks, at 72.9 on AIME25 and 61.9 on GPQA-Diamond. Moving the layer to half depth lowers them to 67.1 and 58.6, and moving it toward the output lowers them further to 64.6 and 57.1 at 95% of depth. This pattern agrees with the observation in Section 3.3. Layers shallower than ℓ^* rarely produce confident predictions and thus offer few alternatives for arbitration, whereas layers deeper than ℓ^* mostly repeat the final prediction and provide no new signal. On Qwen3-14B, ℓ^* also reaches the highest AIME25 accuracy at 71.7, tied with the 72% layer, and the highest GPQA-Diamond accuracy at 60.5 among all layers. In contrast, the half-depth layer falls to 66.7 on AIME25, which shows that arbitrating with a layer that lacks confident alternatives can even blur the decisions. Overall, the

Table 5: Components of TT (avg@8). CoT and argmax rows add loop control to each base decoder, and TT rows remove one component. Argmax + RP + ICE is TT without arbitration. Best results are highlighted in **bold**.

	Qwen3-8B				*Qwen3-14B*			
Decoder	AIME25	AIME26	GPQA-Diamond	Avg.	AIME25	AIME26	GPQA-Diamond	Avg.
CoT	66.7	69.6	58.2	64.8	**71.7**	72.1	59.6	67.8
+ RP	70.4	**70.0**	58.8	66.4	**71.7**	73.3	60.7	68.6
Argmax	65.4	62.1	58.9	62.1	70.0	72.1	**60.9**	67.7
+ RP	63.8	62.1	58.5	61.5	65.0	64.6	59.6	63.1
+ RP + ICE	69.6	64.6	58.8	64.3	67.5	67.5	59.5	64.8
TT (ours)	**72.9**	67.1	**61.9**	**67.3**	**71.7**	**73.8**	60.5	**68.7**
w/o RP	70.8	60.4	56.7	62.6	67.9	69.2	59.5	65.5
w/o ICE	70.0	63.8	60.0	64.6	**71.7**	70.0	60.3	67.3

label-free rule selects the best or tied-best layer in all four comparisons without any accuracy sweep, which supports the point of sharp entropy drop as a suitable layer for arbitration.

Component ablation. Table 5 examines the role of each component from two directions, adding loop control to CoT and argmax and removing one component at a time from TT. First, removing any component lowers the TT average on both models. Specifically, removing arbitration lowers it by 3.0 and 3.9 points on Qwen3-8B and 14B, removing the repetition penalty by 4.7 and 3.2 points, and removing ICE by 2.7 and 1.4 points. Without the penalty, loops form far more often and ICE's three escapes run out more often, so the penalty acts as a prerequisite for ICE to work. However, loop control alone does not explain the gains of TT. Adding only the penalty to argmax even lowers its average from 62.1 to 61.5 on Qwen3-8B and from 67.7 to 63.1 on Qwen3-14B. This indicates that the penalty suppresses repetition but can also disturb ordinary decisions that do not repeat. Adding ICE on top raises the average to 64.3 on Qwen3-8B, which is still below the 64.8 of CoT, and to 64.8 on Qwen3-14B, which remains below argmax. Adding arbitration then raises the average substantially to 67.3 and 68.7, surpassing CoT on both models. As seen in Table 3, arbitration corrects the decisions of generations that the penalty and ICE let reach the end, which reverses the side effect of loop control and yields an additional gain. Finally, adding the penalty to CoT raises its average to 66.4 and 68.6, which confirms that the penalty also benefits sampled decoding. Nevertheless, TT keeps the highest average on both models. Overall, the three components are complementary, as the penalty and ICE let the trajectory exit loops and arbitration steers it toward better decisions than sampling.

6 Conclusion

In this paper, we analyze what the soft input of training-free soft thinking actually carries and how intermediate layers of pretrained reasoning models can support thinking-time decoding. Through a geometric analysis, we first show that the departure of the soft input from the argmax embedding is set by the uncertainty of the top candidate, which is zero or small on most steps, so Soft Thinking behaves much like argmax decoding and shares its repetition loops. However, argmax generations that finish are slightly more accurate than sampled ones on average, which indicates that loops should be exited so that reasoning can continue rather than cut off as Cold Stop does. Moreover, a transition layer located without labels can be confident where the final layer hesitates but stays diffuse inside repetition loops. Therefore, hesitation and loops call for different mechanisms. Based on these observations, Transition Thinking arbitrates hesitant decisions with the transition-layer prediction along a single argmax trajectory and exits verified loops in place. Without additional training or forward passes, it achieves higher overall accuracy and shorter generations than CoT on Qwen3-series models, and arbitration adds gains on top of loop recovery. Overall, these findings suggest that intermediate layers of pretrained reasoning models hold decision signals that final-layer decoding does not use, and that using these signals only at the steps that need them can serve as an effective decoding strategy. Extending this analysis to other model families and to models trained for latent reasoning remains an important direction for future work.

AI USE STATEMENT

In this work, we used generative AI tools to design or provide feedback on research methodology or experiments, assist with translation, and interpret results. We have not used generative AI tools to generate synthetic data sets, provide critical ingredients for proving mathematical claims, or assist in the writing of proofs, and cleaning and reformatting datasets and supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to create or edit software code, discover research topics or identify gaps, identify relevant literature, edit the paper to improve readability, and for brainstorming. We have reviewed all AI-assisted work. We checked AI-suggested research ideas and gaps against prior work through a manual literature survey, and AI-suggested related work was verified by reading the cited papers. AI-generated code was verified and tested for correctness by the authors, and AI-assisted interpretations were checked against the stored result files. AI-assisted translations and edits were reviewed by the authors to ensure that they preserve the intended meaning. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

All experiments use publicly available Qwen3 checkpoints and public benchmarks, and the transition layer of each model is chosen from unlabeled AIME24 thinking traces before any evaluation. The selection rule appears in Section 3.3, and the resulting layers are listed in Table 7. The decoding rule and all of its constants are given in Section 4, and Algorithm 1 in Appendix C restates it step by step. Appendix A derives the payload decomposition of Eq. 1 and describes the measurement setup of Section 3.1. Appendix D describes the inference engine and hardware and lists the settings of every method, while Appendix E specifies the offline loop test used in our analysis. Since batched bf16 inference is not bit-exact, we report averages over eight generations and keep the completions, token counts, and grades of every run.

REFERENCES

Riccardo Ali, Francesco Caso, Christopher Irwin, and Pietro Liò. Entropy-Lens: Uncovering decision strategies in LLMs. *arXiv preprint arXiv:2502.16570*, 2025.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

Shuochen Chang, Tong Bai, Xiaofeng Zhang, Qianli Ma, Qingyang Liu, Zhaohe Liao, Yibo Miao, and Li Niu. Unlocking the black box of latent reasoning: An interpretability-guided approach to intervention. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34019–34032, 2026.

Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*, 2024.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R Glass, and Pengcheng He. DoLa: Decoding by contrasting layers improves factuality in large language models. In *International Conference on Learning Representations*, volume 2024, pp. 54158–54183, 2024.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul Smolensky, Vishrav Chaudhary, and Stuart Shieber. Implicit chain of thought reasoning via knowledge distillation. *arXiv preprint arXiv:2311.01460*, 2023.

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

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, et al. LayerSkip: Enabling early exit inference and self-speculative decoding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12622–12642, 2024.

Renyu Fu and Guibo Luo. SeLaR: Selective latent reasoning in large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7073–7087, 2026.

Zihao Fu, Wai Lam, Anthony Man-Cho So, and Bei Shi. A theoretical analysis of the repetition problem in text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 12848–12856, 2021.

Soham Gadgil, Chris Lin, and Su-In Lee. Where to steer: Input-dependent layer selection for steering improves LLM alignment. *arXiv preprint arXiv:2604.03867*, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

Naman Jain, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025.

Dylan Jayabahu and Tinuade Adeleke. The halt vector: Internalizing a causal steering intervention for efficient reasoning. *arXiv preprint arXiv:2608.28859*, 2026.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213, 2022.

Junyan Li, Wenshuo Zhao, Yang Zhang, and Chuang Gan. Steering LLM thinking with budget guidance. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 37437–37451, 2026.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in neural information processing systems*, 36:41451–41530, 2023a.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori B Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 12286–12312, 2023b.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? Rigorous evaluation of large language models for code generation. *Advances in neural information processing systems*, 36:21558–21572, 2023.

Mathematical Association of America. American invitational mathematics examination (AIME) 2024–2026, problems and answers. `https://huggingface.co/datasets/math-ai/aime24`, `https://huggingface.co/datasets/math-ai/aime25`, `https://huggingface.co/datasets/math-ai/aime26`, 2026.

Zibin Meng, Peng Xie, and Kani Chen. Depth-entropy guided sampling for training-free LLM reasoning. *arXiv preprint arXiv:2607.09693*, 2026.

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.

nostalgebraist. interpreting GPT: the logit lens. LessWrong, 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens.

Renuka Oladri, Niveda Jawahar, and Abdirisak Mohamed. Token budget saturation and mechanistic early detection of reasoning non-convergence in chain-of-thought models. *arXiv preprint arXiv:2607.21433*, 2026.

Charilaos Pipis, Shivam Garg, Vasilis Kontonis, Vaishnavi Shrivastava, Akshay Krishnamurthy, and Dimitris Papailiopoulos. Wait, wait, wait... why do reasoning models loop? *arXiv preprint arXiv:2512.12895*, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Michael Rizvi-Martel, Guillaume Rabusseau, and Marius Mosbach. The illusion of superposition? a principled analysis of latent thinking in language models. *arXiv preprint arXiv:2604.06374*, 2026.

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.

Dachuan Shi, Abedelkadir Asi, Keying Li, Xiangchi Yuan, Leyan Pan, Wenke Lee, and Wen Xiao. SwiReasoning: Switch-thinking in latent and explicit for Pareto-superior reasoning LLMs. In *International Conference on Learning Representations*, volume 2026, pp. 137060–137093, 2026.

Wenhui Tan, Jiaze Li, Jianzhong Ju, Zhenbo Luo, Ruihua Song, and Jian Luan. Think silently, think fast: Dynamic latent compression of LLM reasoning chains. *Advances in Neural Information Processing Systems*, 38:4646–4668, 2026a.

Wenhui Tan, Fiorenzo Parascandolo, Enver Sangineto, Jianzhong Ju, Zhenbo Luo, Qian Cao, Rita Cucchiara, Ruihua Song, and Jian Luan. Restoring exploration after post-training: Latent exploration decoding for large reasoning models. *arXiv preprint arXiv:2602.01698*, 2026b.

Yao Tang, Li Dong, Yaru Hao, Qingxiu Dong, Furu Wei, and Jiatao Gu. Multiplex thinking: Reasoning via token-wise branch-and-merge. *arXiv preprint arXiv:2601.08808*, 2026.

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

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

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

Junhong Wu, Jinliang Lu, Zixuan Ren, Gangqiang Hu, Zhi Wu, Dai Dai, et al. LLMs are single-threaded reasoners: Demystifying the working mechanism of soft thinking. In *International Conference on Learning Representations*, volume 2026, pp. 4533–4548, 2026.

Wenya Xie, Shaochen Zhong, Hoang Anh Duy Le, Zhaozhuo Xu, Jianwen Xie, and Zirui Liu. Word Salad Chopper: Reasoning models waste a ton of decoding budget on useless repetitions, self-knowingly. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 33576–33586, 2025.

Jin Xu, Xiaojiang Liu, Jianhao Yan, Deng Cai, Huayang Li, and Jian Li. Learning to break the loop: Analyzing and mitigating repetitions for neural text generation. *Advances in Neural Information Processing Systems*, 35:3082–3095, 2022.

Yige Xu, Xu Guo, Zhiwei Zeng, and Chunyan Miao. SoftCoT: Soft chain-of-thought for efficient reasoning with LLMs. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23336–23351, 2025a.

Zicheng Xu, Xiuyi Lou, Guanchu Wang, Yu-Neng Chuang, Feng Luo, Guangyao Zheng, Alexander S Szalay, Zirui Liu, and Vladimir Braverman. DTS: Enhancing large reasoning models via decoding tree sketching. *arXiv preprint arXiv:2511.00640*, 2025b.

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.

Aojie Yuan, Zhiyuan Julian Su, Haiyue Zhang, Yi Nian, and Yue Zhao. Hidden error awareness in chain-of-thought reasoning: The signal is diagnostic, not causal. *arXiv preprint arXiv:2605.09502*, 2026a.

Jiayi Yuan, Hao Li, Xinheng Ding, Wenya Xie, Yu-Jhe Li, Wentian Zhao, Kun Wan, Jing Shi, Xia Hu, and Zirui Liu. Understanding and mitigating numerical sources of nondeterminism in LLM inference. *Advances in Neural Information Processing Systems*, 38:169819–169851, 2026b.

Jianyi Zhang, Da-Cheng Juan, Cyrus Rashtchian, Chun-Sung Ferng, Heinrich Jiang, and Yiran Chen. SLED: Self logits evolution decoding for improving factuality in large language models. *Advances in Neural Information Processing Systems*, 37:5188–5209, 2024.

Jinyang Zhang, Hongxin Ding, Yue Fang, Weibin Liao, Muyang Ye, Junfeng Zhao, and Yasha Wang. The tell-tale norm: ℓ_2 magnitude as a signal for reasoning dynamics in large language models. *arXiv preprint arXiv:2606.06188*, 2026a.

Xuanming Zhang, Sining Zhoubian, Yuxuan Chen, Tianyi Tang, An Yang, Sean Du, Chujie Zheng, Fei Huang, Dayiheng Liu, Gao Huang, et al. Deeper is not always better: Mitigating the alignment tax via confident layer decoding. *arXiv preprint arXiv:2606.21906*, 2026b.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, and Xin Wang. Soft thinking: Unlocking the reasoning potential of LLMs in continuous concept space. *Advances in Neural Information Processing Systems*, 38:168990–169012, 2026c.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody H Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. SGLang: Efficient execution of structured language model programs. *Advances in neural information processing systems*, 37:62557–62583, 2024.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to AI transparency. *arXiv preprint arXiv:2310.01405*, 2023.

A THE SOFT-THINKING PAYLOAD AND ITS EFFECT ON GENERATION

This appendix supports Section 3.1, using Qwen3-8B together with Qwen3-1.7B. The payload is set by the uncertainty $1 - p_1$ and is small on most thinking steps (Appendices A.1 and A.3). Feeding it instead of the argmax embedding rarely changes the next generated tokens, and repeated generations share long prefixes as argmax generations do (Appendix A.3).

A.1 THEORETICAL ANALYSIS

At thinking step t, let $C_t = \{c_1, \ldots, c_k\}$ be the candidates retained by Soft Thinking, ordered by their normalized probabilities $p = (p_1, \ldots, p_k)$. Write $e_i = \mathrm{emb}(c_i)$ for the input embedding of candidate i. Soft Thinking feeds their probability-weighted mixture. Grouping the non-argmax candidates makes its top-1 decomposition explicit.

$$e_{\mathrm{soft}} = \sum_i p_i \, e_i \tag{6}$$

$$e_{\mathrm{rest}} = \sum_{i \geq 2} \frac{p_i}{1 - p_1} \, e_i \tag{7}$$

$$e_{\mathrm{soft}} = p_1 \, e_1 + (1 - p_1) \, e_{\mathrm{rest}} \tag{8}$$

Here $1 - p_1 = \sum_{i \geq 2} p_i$ is the uncertainty, the probability left for the non-top candidates, and e_{rest} is the corresponding probability-weighted mean of the remaining embeddings. The last line therefore expresses the soft input as a convex blend of the argmax embedding and this remaining-candidate mean.

The *payload*, the part of the input beyond discrete argmax feedback, is the resulting displacement from e_1.

$$r = e_{\mathrm{soft}} - e_1 = (1 - p_1)(e_{\mathrm{rest}} - e_1) \tag{9}$$

When $p_1 = 1$ the uncertainty is zero, and we define $e_{\mathrm{rest}} = e_1$ so that $r = 0$. To compare its magnitude across models, let $\bar{n} = \mathrm{median}_v \|\mathrm{emb}(v)\|$ be the median vocabulary-embedding norm. The normalized payload then factorizes into an uncertainty term and a geometry term.

$$\frac{\|r\|}{\bar{n}} = (1 - p_1) \, D_t \qquad D_t = \frac{\|e_{\mathrm{rest}} - e_1\|}{\bar{n}} \tag{10}$$

The payload thus points from e_1 toward e_{rest}, with magnitude given by the uncertainty $1 - p_1$ times the normalized distance D_t. Since e_{rest} is a convex combination of the remaining candidate embeddings, $D_t \leq \max_{i \geq 2} \|e_i - e_1\| / \bar{n}$. On the measured checkpoints D_t stays of order one, with medians of 0.94 and 1.32 for Qwen3-1.7B and Qwen3-8B, and a log–log fit of $\|r\|$ on $1 - p_1$ gives $R^2 = 0.985$ on checkpoints with a nonzero payload. This indicates that the payload size is almost entirely determined by the uncertainty $1 - p_1$.

A.2 MEASUREMENT SETUP

Measurements use Qwen3-1.7B and Qwen3-8B on the first 24 AIME24 problems. For each problem we run Soft Thinking for 1,500 thinking steps, feeding back the top-10 mixture at T=0.6. We take one checkpoint every 50 steps from step 50 to step 1,450, which gives 29 checkpoints per problem and 696 per model. The prefix-sharing paragraph of Appendix A.3 instead uses the main-experiment Qwen3-8B generations on AIME24–26. At each checkpoint we filter the candidates exactly as Soft Thinking does and renormalize the retained probabilities to sum to one. We report every embedding-space distance and norm in units of $\bar{n}$.

A.3 EMPIRICAL CHANNEL BEHAVIOR

The payload is small on most steps. The uncertainty $1 - p_1$ is exactly zero on 42–48% of checkpoints, where the cumulative curves of Figure 3a start, and small on most others. On 83–89% of checkpoints with a nonzero payload, the payload is below $0.4\,\bar{n}$, far from the distance of $1.38\,\bar{n}$ between random token embeddings marked by the dashed line. Moreover, as listed in Table 6, the nearest vocabulary embedding to e_{soft} is the argmax embedding on more than 99% of checkpoints. Therefore, the soft input is nearly indistinguishable from the argmax embedding on most steps.

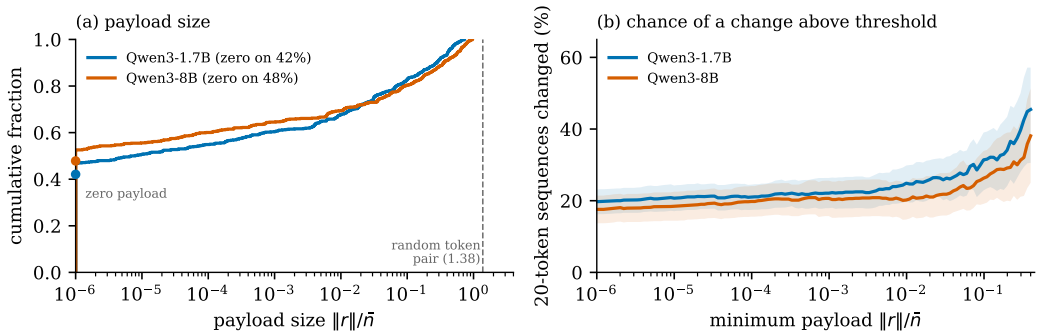


Figure 3: Payload size and its effect on the next 20 tokens, over 696 checkpoints per model. (a) Cumulative distribution of $\|r\|/\bar{n}$. (b) Change rate among checkpoints with $\|r\|/\bar{n} \geq x$.

Table 6: Channel statistics over 696 checkpoints per model, of which 403 and 363 have a nonzero payload. Percentages have 95% cluster-bootstrap intervals of at most ± 5 points over the 24 problems.

	Qwen3-1.7B	Qwen3-8B
Embedding-space proximity		
nearest embedding to e_{soft} is $\text{emb}(c_1)$	99.3%	99.1%
Payload direction (nonzero payload)		
median $\cos(r, \text{emb}(c_2) - \text{emb}(c_1))$	0.92	0.92
shuffle null	0.56	0.56
rank 2 is best-aligned	94%	93%
20-token sequence agreement		
same next 20 tokens (nonzero payload)	81.3%	83.9%
same next 20 tokens (all checkpoints)	89.2%	91.6%

The chance of a change grows with payload size. From each checkpoint we generate the next 20 tokens greedily twice, once with e_{soft} and once with the argmax embedding as the next input. The two 20-token sequences are identical on 89–92% of checkpoints. With zero payload, the two sequences always match. Figure 3b shows the change rate among checkpoints whose normalized payload is at least a threshold x, with 95% bootstrap intervals over the 24 problems. The curves stop at $x = 0.4$, where 44 and 63 checkpoints remain, and the rate rises from 26–32% at $x = 0.1$ to 38–45% at $x = 0.4$. Because D_t stays of order one, large payloads arise only where $1 - p_1$ is large. Therefore, the soft input changes the generation mainly at steps where the final layer hesitates.

Nonzero payloads point toward the runner-up. On checkpoints with a nonzero payload, the median cosine between the payload and $e_2 - e_1$ is 0.92, against 0.56 under a null that randomly repairs the same probabilities with the same candidates. Moreover, the runner-up is the best-aligned candidate on 93–94% of these checkpoints. The mixture therefore feeds back the final layer's own second choice rather than a new signal.

Repeated generations share long prefixes. For Qwen3-8B on AIME24–26 we measure the longest thinking prefix shared by all eight generations of a problem, as a fraction of their mean thinking length in characters. The median over the 90 problems is 18% for Soft Thinking and 17% for argmax, against 0.1% for sampled CoT and SST, whose shared prefix is only the opening words. Neither decoder samples during thinking, so in exact arithmetic these generations would coincide. A likely source of their separation is the small logit perturbation of batched bf16 inference, which can flip the argmax where the top two candidates are nearly tied (Yuan et al., 2026b). In contrast, the Gumbel noise of SST lets each generation leave the argmax path from the first steps, which is consistent with SST behaving like sampled CoT rather than like argmax. As shown in Table 10,

Soft Thinking also ends in literal repetition about as often as argmax, which is consistent with its argmax-like behavior.

A.4 Implication

The standard training-free mixture therefore re-encodes the final-layer distribution as a small displacement toward the runner-up, and the frozen model usually continues as it would from the argmax token. This agrees with Rizvi-Martel et al. (2026), who find that pretrained models collapse superposed inputs toward a dominant token unless they are trained from scratch on latent thoughts. Larger payloads, which are more likely to change the next tokens, arise where the final layer hesitates. As described in Section 4.1, Transition Thinking therefore uses a prediction from another depth at such steps to decide whether to switch the token, rather than blending in the final layer's runner-up. This does not rule out continuous reasoning in models trained to use continuous states.

B Locating the Transition Layer

This appendix details the AIME24 calibration of Section 3.3 for the three main models. The profiles are those of Figure 1a, computed with the logit lens of Eq. 2 on unlabeled AIME24 thinking traces. The calibration set $\mathcal{C}$ contains the first 768 thinking tokens of the first sampled-CoT generation for each of the first six AIME24 problems, 4,608 positions in total. We also describe the prediction at ℓ^* inside repetition loops, the change in the MLP update near ℓ^*, and the indexing convention.

Calibration profiles. Table 7 lists the three largest one-layer decreases of the mean entropy and the mean entropy just before ℓ^*, at ℓ^*, and at the final layer. It also reports how often the top-1 prediction at ℓ^* agrees with the final top-1. In all three models the entropy peaks early, at 8.3–9.2 nat, stays high through the middle layers, and falls most sharply at ℓ^*. On Qwen3-8B the drop at $\ell^* = 22$ is clearly the largest. On Qwen3-4B and Qwen3-14B the second-largest drop is nearly equal and occurs at the next layer, so the rule selects the shallower of the two adjacent layers. As shown in Table 4, the layer selected by this rule gives the best or tied-best accuracy among the compared depths.

Table 7: AIME24 calibration profiles of the main models (the profiles of Figure 1a). Entropies are means over the calibration positions in nat. For Qwen3-14B the two largest drops are equal at this precision, and the rule takes the shallower layer.

		Largest drops: layer (nat)			Mean entropy [nat]			Top-1 agree
Model (layers)	ℓ^* (depth)	1st	2nd	3rd	$\mathcal{H}_{\ell^*-1}$	$\mathcal{H}_{\ell^*}$	$\mathcal{H}_F$	at ℓ^*
Qwen3-4B (36)	22 (0.61)	22 (1.53)	23 (1.50)	25 (0.64)	6.42	4.89	0.49	0.13
Qwen3-8B (36)	22 (0.61)	22 (1.09)	23 (0.73)	25 (0.47)	4.99	3.90	0.26	0.10
Qwen3-14B (40)	27 (0.68)	27 (0.95)	28 (0.95)	23 (0.91)	5.47	4.52	0.52	0.11

What the transition layer is and is not. At ℓ^* the prediction is far from the final distribution. Its entropy is 3.9–4.9 nat against 0.26–0.52 at the final layer, and its top-1 token agrees with the final top-1 on only 10–13% of positions. The gate condition $H_{\ell^*} < 1$ holds on 6–10% of positions (Table 1). This indicates that ℓ^* marks where intermediate predictions begin to form rather than where the final decision is already known. Therefore, a confident prediction at ℓ^* can offer an alternative to the final decision, which hesitation arbitration uses at hesitant steps.

Inside a loop. We teacher-force the looping tails of 12 Qwen3-8B argmax generations that exhausted the budget. On loop positions the final-layer entropy is 0.011 nat, while the prediction at $\ell^* = 22$ has a mean entropy of 6.6 nat. The gate condition $H_{\ell^*} < 1$ holds on 3% of these positions, and even an equal-weight mix of the two logits changes the argmax token on only 0.1% of them, so arbitration cannot break the loop. This motivates the separate loop recovery of Section 4.2.

MLP update near the transition layer. The transition also coincides with a change in what the MLP adds to the residual stream. Each Qwen3 layer adds attention and MLP outputs to its hid-

den state, and Figure 1b shows the cosine between the MLP update and the normalized final state, averaged over the calibration positions $\mathcal{C}$.

$$h_{t,\ell} = h_{t,\ell-1} + \Delta^{\text{attn}}_{t,\ell} + \Delta^{\text{MLP}}_{t,\ell} \tag{11}$$

$$a_\ell = \frac{1}{|\mathcal{C}|} \sum_{t \in \mathcal{C}} \cos\big(\Delta^{\text{MLP}}_{t,\ell}, u_t\big) \qquad u_t = \frac{N(h_{t,L-1})}{\|N(h_{t,L-1})\|} \tag{12}$$

In all three models the projection is positive in the first few layers, near zero through the middle layers, and positive and rising with depth after ℓ^*. This indicates that after ℓ^* the MLP starts to write toward the final prediction, which is consistent with the entropy transition.

Index convention. Layer k denotes the output of the k-th decoder layer, counted from zero. Index 0 of the Hugging Face tensor `hidden_states` is the embedding, so `hidden_states[k]` is layer $k-1$. All figures and tables in this paper use layer indices.

C Decoding Algorithm

Algorithm 1 summarizes the decoding rule of Section 4, which combines hesitation arbitration with the repetition penalty and ICE along a single thinking trajectory. We write $R_\rho(z, Y)$ for the repetition penalty of Eq. 5, which divides the positive scores and multiplies the negative scores of every token already in the generated sequence Y by ρ. The predicate $\mathrm{RepeatingTail}(Y)$ tests periods $p \le 1{,}024$ with enough history. It returns true if at least 90% of the latest 256 or 512 tokens equal the tokens p positions earlier for some such p. During ICE, $\mathrm{RandomTop20}(z_F, Y)$ assigns fresh independent uniform scores to the final layer's top-20 tokens and returns the highest-scoring token after applying R_ρ.

Algorithm 1 Transition Thinking

Require: Transition layer ℓ^*; repetition penalty $\rho = 1.05$; token budget $B = 32{,}768$
Require: Low-entropy window $W = 256$; escape length $m = 64$; escape limit $K = 3$
$Y \leftarrow [\,]$ ▷ generated tokens, in order
$(c, f) \leftarrow (0, 0)$ ▷ low-entropy streak; failed checks
$(e, r) \leftarrow (0, 0)$ ▷ escapes used; escape steps left
Thinking phase
for $t = 1, \ldots, B$ **do**
  Run one Transformer pass and read $z_F = W_U N(h_{t,L-1})$ and $z_\ell = W_U N(h_{t,\ell^*})$
  $p_F \leftarrow \mathrm{softmax}(z_F)$; $p_\ell \leftarrow \mathrm{softmax}(z_\ell)$
  $g \leftarrow \frac{1}{2}(1 - \max_v p_F(v))\,\mathbf{1}[H(p_\ell) < 1]$
  $q \leftarrow R_\rho\big((1-g)z_F + g z_\ell, Y\big)$
Choose the next token
  if $r > 0$ **then** ▷ ICE step
    $x \leftarrow \mathrm{RandomTop20}(z_F, Y)$
  else
    $x \leftarrow \arg\max_v q(v)$
  end if
  Append x to Y
Check for a persistent loop
  if $r > 0$ **then**
    $r \leftarrow r - 1$; $c \leftarrow 0$ ▷ detector paused during an escape
  else if $f \le 3$ **then** ▷ detector remains active
    $c \leftarrow c + 1$ if $H(\mathrm{softmax}(q/0.6)) < 0.01$; otherwise $c \leftarrow 0$
    if $c = W$ **then**
      $c \leftarrow 0$
      if $\mathrm{RepeatingTail}(Y)$ **then**
        if $e = K$ **then**
          $x \leftarrow$ </think> ▷ fourth verified loop
          Replace the last token of Y with x
        else
          $e \leftarrow e + 1$; $r \leftarrow m$ ▷ escape on the next m steps
        end if
      else
        $f \leftarrow f + 1$ ▷ disable after a fourth failed check
      end if
    end if
  end if
  if $x =$ </think> **then**
    break
  end if
end for
if $x \neq$ </think> **then**
  return Y ▷ truncated at the budget, no answer
end if
Answer phase
Sample with temperature 0.6, top-p 0.95, top-k 20, without the repetition penalty

D Implementation and Run Configuration

Engine. All arms run in a shared fork of SGLang 0.4.6.post1 (Zheng et al., 2024), the version used by Soft Thinking. We patched it with components from SGLang 0.5.18 to support Blackwell GPUs and use the FlashInfer backend. The fork implements Soft Thinking's embedding-mixture input and SST's Gumbel-Softmax variant. For TT, it reads the transition-layer logits $z_{\ell^*} = W_U N(h_{\ell^*})$ with the model's final normalization N and LM head W_U (Eq. 2). It gates the mix with the final logits, applies the repetition penalty in the sampler, and runs the ICE loop detector and escape. The transition-layer prediction adds one RMSNorm and one LM-head multiplication per decoding step. Table 8 lists the decoding and evaluation settings of every method.

Table 8: Decoding and evaluation settings.

Shared setup	
Models	Qwen3-4B, Qwen3-8B, Qwen3-14B in thinking mode
Precision	bf16
Generations per problem	8 on every benchmark
Token budget	32,768 generated tokens
AIME grading	Rule-based exact match
GPQA grading	Reference answer match
Code grading	Reference tests, including the LiveCodeBench release-v5 evaluator
Hardware	One NVIDIA RTX PRO 6000 Blackwell (96 GB) per run
Thinking phase	
CoT sampling	$T = 0.6$, top-$p = 0.95$, top-$k = 30$
ST / SST input	Top-$k = 20$, Soft input embedding Top-$k = 20$
ST / SST Cold Stop	Entropy below 0.01 nat for 256 steps
SST noise	Gumbel temperature 0.5
TT layer ℓ^*	22 / 22 / 27 for 4B / 8B / 14B (zero-indexed, Section 3.3)
TT gate	$g = \frac{1}{2}(1 - \max p_F)$ when $H_{\ell^*} < 1$ nat, zero otherwise
TT penalty	$\rho = 1.05$ on generated tokens
TT loop trigger	Penalized entropy at $T = 0.6$ below 0.01 nat for 256 steps, followed by a check for a repeating period of at most 1,024 tokens with at least 90% matches over the latest 256 or 512 tokens
TT escape	64 steps drawn at random from the final top-20 with the penalty applied, at most 3 escapes, and a fourth verified loop ends thinking
Answer phase (all methods)	
Sampling	$T = 0.6$, top-$p = 0.95$, top-$k = 20$
Repetition penalty	Off

E LOOPS AND TRUNCATION UNDER ARGMAX DECODING

This appendix separates completion from accuracy under argmax decoding and compares how ST and argmax loops terminate. We call a generation truncated when it reaches the token budget without emitting `</think>` and finished when it emits `</think>`, whether by itself or through Cold Stop. We split accuracy into the fraction t of truncated generations and the accuracy c of finished generations. Truncated generations are graded on their partial text and are rarely correct, so overall accuracy is close to $(1-t)\,c$. Table 9 applies this decomposition to sampled CoT and argmax decoding on Qwen3-8B, over 240 generations per benchmark. The measured accuracy exceeds $(1-t)\,c$ only where a truncated generation already contains the correct answer, which happens for four argmax generations on AIME24 and one each on AIME25 and AIME26. Argmax is more accurate on finished generations on AIME24 and AIME25, but it exhausts the budget roughly twice as often, which erases this gain. Most of this extra truncation ends in a literal repeated tail.

Table 9: Accuracy decomposition for sampled CoT and argmax decoding on Qwen3-8B (%).

Benchmark	Decoder	Truncated t	Acc. when finished c	$(1-t)\,c$	Acc.
AIME24	CoT	6.7	79.0	73.8	73.8
	Argmax	14.2	**82.5**	70.8	72.5
AIME25	CoT	7.9	72.4	66.7	66.7
	Argmax	15.4	**76.8**	65.0	65.4
AIME26	CoT	6.2	**74.2**	69.6	69.6
	Argmax	11.7	69.8	61.7	62.1

Table 10 counts loops, generations whose thinking ends in a repeated tail. A tail counts as repeated when more than 30% of the word 8-grams in its last 3,000 characters are repeats, an offline test separate from the ICE detector of Section 4.2. On AIME24–26, ST and argmax loop about equally often, in 56 and 53 of their 720 generations, but the loops end differently. ST finishes 53 of its 56 loops, consistent with Cold Stop ending them, while argmax truncates 52 of its 53. All four decoders reach about 70% accuracy away from loops, against 21% for ST and 9% for argmax on loops. This indicates that Cold Stop ends loops but does not recover the reasoning they interrupt. Table 3 applies the same test to every 3,000-character window and therefore counts more generations.

Table 10: Loops in the 720 Qwen3-8B generations on AIME24–26. Each loop either finishes or is truncated, and accuracy is in percent.

Decoder	Loops	Finished	Truncated	Acc. on loops	Acc. off loops
CoT	9	0	9	0.0	70.9
ST	56	53	3	21.4	70.2
SST	9	7	2	44.4	70.1
Argmax	53	1	52	9.4	71.3