LOW-CONFIDENCE REMASKING TRAPS FLEXIBILITY: REALIZING ARBITRARY-ORDER POTENTIAL FOR DIVERSE ROLLOUTS IN DIFFUSION LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Masked diffusion language models support arbitrary-order generation, suggesting a natural way to produce diverse outputs. However, recent work argues that this flexibility reduces diversity by delaying high-uncertainty tokens that can lead to different generation paths. We trace this diversity loss not to arbitrary-order generation itself, but largely to *low-confidence remasking* (LCR), a widely used decoding rule. At each step, LCR samples a token at every masked position but commits only the sampled token with the highest probability, filtering out the rest. We show that this mechanism can exponentially suppress lower-probability tokens as more positions compete, and observe the same suppression in LLaDA. In contrast, *top-probability position selection* (TPP), which has often been conflated with LCR under the shared label *confidence-based decoding*, avoids this diversity loss. TPP first selects the position whose most likely token has the highest probability, then samples directly from that position's distribution. Replacing LCR with TPP restores diversity and yields Pass@k comparable to left-to-right decoding, suggesting that the reported diversity loss stems largely from LCR's filtering rather than from generating high-confidence positions first. To further exploit order flexibility, we introduce **Entropy-Guided Initialization** (EGI), which samples the first token at the highest-entropy position and then follows TPP. This simple modification further improves rollout diversity and solution coverage beyond left-to-right decoding, with gains extending to downstream policy optimization, highlighting the potential of arbitrary-order generation for diverse rollouts.

1 INTRODUCTION

Masked diffusion language models (MDMs) predict token distributions at all masked positions simultaneously (Austin et al., 2021a; Lou et al., 2024; Sahoo et al., 2024; Shi et al., 2024; Ou et al., 2025). This enables both parallel decoding and arbitrary-order generation (Wu et al., 2026; Kim et al., 2025a), unlike autoregressive models that generate tokens in a fixed left-to-right order (Radford et al., 2018; 2019). In particular, this order flexibility appears naturally suited to diverse rollout sampling, where multiple responses to the same prompt explore different solution paths (Gong et al., 2026; Ni et al., 2026). Such diversity can improve *solution coverage*, measured by Pass@k: the probability that at least one of k sampled responses is correct (Chen et al., 2021).

However, Ni et al. (2026) report lower rollout diversity and Pass@k under arbitrary-order decoding than under left-to-right decoding of the same MDM. Their main comparison uses a widely adopted arbitrary-order decoding rule, *low-confidence remasking* (LCR) (Chang et al., 2022; Nie et al., 2025). They attribute this *flexibility trap* to generating more certain tokens first and delaying high-uncertainty positions where token choices can lead to different reasoning paths. As context accumulates, their uncertainty decreases, leaving fewer paths to explore. Left-to-right decoding instead resolves uncertain positions as they arise, allowing sampling to explore different paths.

In this work, we challenge this interpretation of the flexibility trap: *the reported diversity loss largely stems from how LCR filters sampled tokens, not arbitrary-order generation itself*. For one-token decoding, LCR samples a token at every masked position but commits only the sampled token with the highest model probability. An alternative decoding rule, *top-probability position selection*

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.

(TPP) (Kim et al., 2025a), first chooses the position where the most likely token has the highest probability. It then samples and commits a token at that position without further filtering. Yet this distinction is often obscured in prior work, which refers to either rule as *confidence-based decoding* and sometimes pairs TPP-style descriptions or analyses with LCR experiments (see App. A.2).

Two toy models reveal how LCR's token filtering can *exponentially suppress* lower-probability tokens and outputs. A lower-probability sampled token survives only if *all* competing sampled tokens have equal or lower model probability. In an independent-token toy model, this event becomes exponentially unlikely as the number of remaining masked positions grows. TPP, in contrast, generates lower-probability tokens at the rate specified by the chosen temperature. A second toy model with non-overlapping solution sequences shows analogous suppression of lower-probability outputs at the sequence level. These examples demonstrate that LCR can severely restrict exploration even when temperature makes lower-probability tokens common among the initial samples.

We observe the same suppression in LLaDA (Nie et al., 2025): LCR predominantly commits the most likely token at the selected position even as temperature increases. Simply replacing LCR with TPP restores diversity and yields Pass@k comparable to left-to-right decoding across benchmarks on LLaDA and LLaDA-1.5 (Zhu et al., 2026). Crucially, TPP still prioritizes high-confidence positions and can therefore postpone uncertain ones. These results challenge the claim that delaying uncertain positions is the main cause of diversity loss and instead identify LCR's filtering of lower-probability sampled tokens across positions as a central limitation. More broadly, this substantial behavioral gap makes their conflation potentially misleading, underscoring the need to distinguish LCR from TPP.

Finally, we show that changing position selection at just one decoding step can further improve rollout sampling beyond left-to-right decoding. We introduce *Entropy-Guided Initialization* (EGI), a training-free modification that changes only the first decoding step of TPP. EGI first samples a token at the highest-entropy masked position, then follows TPP for the remaining steps. This modification improves rollout diversity and solution coverage over left-to-right decoding, including higher Pass@k. These benefits extend to downstream group-relative policy optimization, where replacing left-to-right rollouts with EGI rollouts improves performance across all three methods we evaluate (Wang et al., 2026a;b; Ni et al., 2026). Together, these results show that diverse rollout sampling need not be confined to left-to-right decoding and highlight the untapped potential of arbitrary-order generation as a design space for both inference and policy optimization.

Our contributions are summarized as follows:

- We distinguish two decoding rules often labeled *confidence-based decoding*: low-confidence remasking (LCR) and top-probability position selection (TPP). They coincide under greedy decoding at zero temperature but define fundamentally different samplers at positive temperature, making this distinction essential for rollout sampling and their conflation potentially misleading.
- Using two toy models, we show that LCR's token filtering can *exponentially suppress* lower-probability choices, and observe the same suppression in LLaDA. Replacing LCR with TPP largely closes the Pass@k gap to left-to-right decoding, pointing to LCR's token filtering, rather than confidence-prioritized ordering itself, as a central source of the *flexibility trap*.
- We introduce *Entropy-Guided Initialization* (EGI), which modifies only the first step of TPP to use order flexibility for exploration. EGI improves rollout diversity and solution coverage beyond left-to-right decoding, with gains extending to downstream group-relative policy optimization, highlighting the potential of arbitrary-order generation for stronger rollout sampling.

2 Preliminaries

2.1 Masked Diffusion Language Models and Decoding Rules

Discrete diffusion models learn to generate discrete data by reversing a stochastic corruption process (Hoogeboom et al., 2021; Austin et al., 2021a; Lou et al., 2024). Masked diffusion language models (MDMs) use a forward process that progressively replaces tokens with a special mask token (Sahoo et al., 2024; Shi et al., 2024; Ou et al., 2025). The model learns to reconstruct the original tokens from partially masked sequences and generates text by iteratively filling masked positions. LLaDA (Nie et al., 2025) scales masked diffusion to an 8B-parameter large language model.

Given a partially masked sequence $\mathbf{x}$, let M denote the set of masked positions and $p_i(v)$ the probability the model assigns to token $v \in \mathcal{V}$ at position $i \in M$. An MDM predicts these distributions simultaneously, allowing tokens to be committed in parallel or in arbitrary orders (Wu et al., 2026; Kim et al., 2025a). To isolate decoding-rule effects, we focus on one-token-per-step decoding.

Decoding Rules. An MDM can choose a masked position to generate next and then sample a token at that position. It can follow a left-to-right autoregressive (AR) order, always choosing the leftmost masked position, or select the next position adaptively based on its current predictions.[1] Common position-selection criteria favor the largest top-token probability, the largest margin between the two highest token probabilities (Kim et al., 2025a), or the lowest token entropy (Ye et al., 2025). We call the first rule *top-probability position selection* (TPP), which samples directly from a position's distribution after selecting it. With consistent model conditionals, this sampling preserves the common joint distribution regardless of position order (Kim et al., 2025a).

Another widely used decoding rule is *low-confidence remasking* (LCR) (Chang et al., 2022; Nie et al., 2025). In our one-token-per-step setting, LCR first samples a token proposal $\tilde{x}_i$ at every masked position $i \in M$. It then commits only the proposal with the highest model probability $p_i(\tilde{x}_i)$, leaving all other positions masked.[2] Thus, TPP selects a position before token sampling, whereas LCR selects among tokens that have already been sampled. Despite this distinction, the term *confidence-based decoding* has been used for both TPP and LCR across prior work. We formalize these rules and examine how they differ under rollout sampling in Sec. 3.

2.2 Diverse Rollout Sampling

Rollout sampling generates multiple responses to the same prompt and is one of the simplest forms of inference-time scaling. Its effectiveness depends on whether the samples explore diverse, viable solution paths rather than repeatedly follow the same trajectory. Such diversity can increase *solution coverage*, the chance of finding at least one correct solution across multiple attempts. This coverage is commonly measured by **Pass@k**, the probability that at least one of k sampled solutions is correct (Chen et al., 2021; Brown et al., 2024).

Rollout diversity is also important for Group-Relative Policy Optimization (GRPO), which samples multiple rollouts per prompt and learns from their relative rewards (Shao et al., 2024). For binary rewards, groups containing both correct and incorrect rollouts provide relative learning signal, whereas identical-reward groups provide little. By exploring different solution paths, diverse rollouts can increase the chance of within-group reward variation. Pass@k remains a common measure of such rollout coverage, while another useful metric is **Potential@k**, which measures how often problems missed by a single rollout are recovered through additional sampling (Yao et al., 2025).

Token temperature is one of the simplest and most common controls for rollout diversity. Given a token distribution $p(v)$, temperature $T > 0$ defines the tempered distribution

$$p^{(T)}(v) \propto p(v)^{1/T},$$

with $T = 1$ recovering the original distribution. As $T \to 0$, sampling becomes greedy and selects only the top-probability token, which we conventionally denote by $T = 0$. Increasing T gives lower-probability tokens more mass, approaching uniform sampling as $T \to \infty$.

3 Two Distinct Samplers Behind Confidence-based Decoding

As noted in Sec. 2, both low-confidence remasking (LCR) and top-probability position selection (TPP) have been referred to as confidence-based decoding or related terms, but they define different decoding rules. This conflation has led to mismatches in prior work, where descriptions or analyses of TPP are sometimes paired with experiments using LCR (see App. A.2 for details).

[1] Block-autoregressive decoding generates contiguous blocks from left to right (Arriola et al., 2025; Nie et al., 2025). The position-selection criteria discussed here apply within each block.

[2] Here, "remasking" discards newly sampled tokens rather than masking previously committed tokens, as in methods such as ReMDM (Wang et al., 2025).

To formalize the distinction, consider a single decoding step. For a fixed partially decoded sequence, let M denote the set of masked positions and $p_i(v)$ the model probability of token $v \in \mathcal{V}$ at position $i \in M$. Let $p_i^{(T)}(v) \propto p_i(v)^{1/T}$ denote the tempered distribution at temperature $T > 0$.

Low-confidence remasking (**LCR**) first independently samples a token proposal $\tilde{x}_i \sim p_i^{(T)}$ at every masked position $i \in M$. It scores each proposal by its probability $p_i(\tilde{x}_i)$ under the original model distribution and selects the highest-scoring proposal: $i^\star = \arg\max_{i \in M} p_i(\tilde{x}_i)$. The selected proposal $\tilde{x}_{i^\star}$ is then committed to the sequence at position $i^\star$, while all other proposals are rejected.

Top-probability position selection (**TPP**) instead scores each masked position by its top-token probability, $c_i := \max_{v \in \mathcal{V}} p_i(v)$. It selects the highest-scoring position, $i^\star = \arg\max_{i \in M} c_i$, and then samples and immediately commits a token $x_{i^\star} \sim p_{i^\star}^{(T)}$ at that position. All other positions in M remain masked. Notably, the committed token need not be the top token defining $c_{i^\star}$.

[LCR] Sample all, then select an index

1. Sample proposals: $\tilde{x}_i \sim p_i^{(T)},\ \forall i \in M$

2. Score and select: $i^\star = \arg\max_{i \in M} p_i(\tilde{x}_i)$

3. Commit: $x_{i^\star} = \tilde{x}_{i^\star}$

[TPP] Select a position, then sample

1. Score positions: $c_i := \max_{v \in \mathcal{V}} p_i(v),\ \forall i \in M$

2. Select a position: $i^\star = \arg\max_{i \in M} c_i$

3. Sample and commit: $x_{i^\star} \sim p_{i^\star}^{(T)}$

Under greedy decoding at $T = 0$, LCR and TPP coincide, committing the highest-probability token across all masked positions. This equivalence helps explain why the two procedures can be conflated under deterministic decoding. For rollout sampling at $T > 0$, however, their difference affects the sampling distribution, not merely the decoding order.

TPP preserves the model distribution in principle. At $T = 1$, TPP changes only which position is generated next, so if the model conditionals come from a common joint distribution, sampling order does not change the resulting joint distribution (Kim et al., 2025a). For any $T > 0$, TPP samples and commits directly from the tempered distribution at the selected position, following the token probabilities induced by the temperature without further rejection based on the sampled token.

LCR, by contrast, can exponentially suppress lower-probability tokens. A lower-probability proposal is committed only if *every* competitor has lower model probability. A single higher-probability proposal suffices to reject it, so its commitment probability rapidly decreases as more positions compete. LCR therefore biases commitment toward higher-probability proposals, severely suppressing the lower-probability tokens introduced by the requested temperature (Zhang et al., 2026).

Sec. 4 establishes this exponential suppression in toy models as the number of competing positions grows. Sec. 5 observes the corresponding pattern in large-scale MDMs (e.g., LLaDA), where commitments remain concentrated on top-probability tokens even as temperature increases.

4 Analyzing Diversity Suppression in Low-Confidence Remasking

We use two toy models to isolate how LCR's rejection of lower-probability proposals affects rollout diversity. The first considers i.i.d. tokens, where each decoding step leaves the remaining token distributions unchanged, and reveals the effect at the token level. This setting closely matches the independent-position analysis of Zhang et al. (2026), but yields a sharper result: beyond entropy reduction, non-top commitments are exponentially suppressed under LCR. The second considers fully non-overlapping sequences and extends the same mechanism to sequence-level outputs. In both settings, we reduce the analysis to top-probability versus non-top choices and compare how LCR suppresses non-top choices while TPP follows their tempered probability. See App. A.1 for detailed comparisons with prior work and App. B for proofs and additional results.

4.1 Toy Model I: Independent Tokens

Setup. We begin with the simplest i.i.d. setting, where every position independently follows the same strictly positive distribution p over a finite vocabulary $\mathcal{V}$, with a unique top-probability token. We generate a length-L sequence from a fully masked state, committing one token per step. Let $c_T := \max_{v \in \mathcal{V}} p^{(T)}(v)$ denote the top-token probability under the tempered distribution.

Table 1: **Concrete toy examples of diversity suppression under LCR.** For $L = 128$, Toy I reports the non-top token probability at the first decoding step and the expected non-top fraction in the final sequence. Toy II reports the non-top sequence probability for ten non-overlapping sequences. Target (TPP) denotes the corresponding values under the tempered distribution, which TPP preserves exactly, whereas LCR suppresses non-top choices through cross-position competition.

	Independent Token Generation (Toy I)				**Non-Overlapping Sequence Generation (Toy II)**	
T	$\mathbb{P}$(non-top token)		$\mathbb{E}[N_{\text{non-top}}/L]$		$\mathbb{P}$(non-top sequence)	
	LCR	**Target (TPP)**	**LCR**	**Target (TPP)**	**LCR**	**Target (TPP)**
1	0.000139%	90.0%	7.03%	90.0%	0.0000000000394%	80.0%
2	0.00801%	92.90%	10.22%	92.90%	0.000000270%	85.71%
∞	0.141%	95.0%	14.82%	95.0%	0.000139%	90.0%

Non-top tokens are exponentially suppressed under LCR. Consider a single decoding step with m masked positions. Under TPP, all positions have the same top-token probability, so any position can be selected and a token is sampled and committed directly from $p^{(T)}$. A non-top token is therefore committed with probability $1 - c_T$, directly reflecting the chosen temperature. Under LCR, however, a non-top proposal is committed if and only if all m positions propose non-top tokens, since a single top-token proposal is enough to reject it. Thus, the probabilities of committing a non-top token under TPP and LCR are

$$\boxed{\mathbb{P}_{\text{TPP}}(\text{non-top token}) = 1 - c_T, \quad \mathbb{P}_{\text{LCR}}(\text{non-top token}) = (1 - c_T)^m.}$$

TPP therefore commits non-top tokens at exactly the rate specified by temperature, whereas under LCR the same probability collapses exponentially as more positions compete.

The final non-top fraction vanishes under LCR. The stepwise suppression accumulates over the full sequence. The expected fraction of non-top tokens in the final sequence satisfies

$$\boxed{\mathbb{E}_{\text{TPP}}\left[\frac{N_{\text{non-top}}}{L}\right] = 1 - c_T, \quad \mathbb{E}_{\text{LCR}}\left[\frac{N_{\text{non-top}}}{L}\right] \le \frac{1 - c_T}{L c_T} = O\left(\frac{1}{L}\right),}$$

where $N_{\text{non-top}}$ is the number of non-top tokens in the final sequence. Under TPP, non-top tokens appear in the final sequence at exactly the expected fraction $1 - c_T$ specified by the tempered distribution. Under LCR, however, the same temperature drives the non-top fraction to zero as $O(1/L)$, even though the tempered distribution continues to assign non-top tokens a fixed probability $1 - c_T$.

Concrete Example. Tab. 1 (Left) illustrates the gap between TPP and LCR for $L = 128$ with 20 tokens, where the top-probability token has probability only 0.1 and the other 19 equally share the remaining 0.9. As $T \to \infty$, the tempered distribution assigns 95% probability to non-top tokens, which TPP preserves. Despite this extreme flattening, the probability that LCR commits a non-top token at the first step falls to 0.141%, and the final non-top fraction is 14.82%.

4.2 Toy Model II: Non-Overlapping Sequences

Setup. We next consider a sequence setting motivated by reasoning problems with distinct valid solution trajectories. Let $\{\mathbf{x}_{(1)}, \ldots, \mathbf{x}_{(K)}\}$ be a finite set of length-L sequences with positive probabilities, one of which we generate from a fully masked state one token per step. For simplicity, we assume that any two sequences differ at every position: $(\mathbf{x}_{(k)})_i \neq (\mathbf{x}_{(\ell)})_i$ for all $k \neq \ell$ and all i. Assume a unique top-probability sequence, and let π_T denote its probability at temperature T.

Non-top sequence probability is exponentially suppressed. By construction, the first committed token identifies the sequence and fixes its continuation. Under TPP, all positions have the same top-token probability, so any position can be selected. Because the first committed token determines the

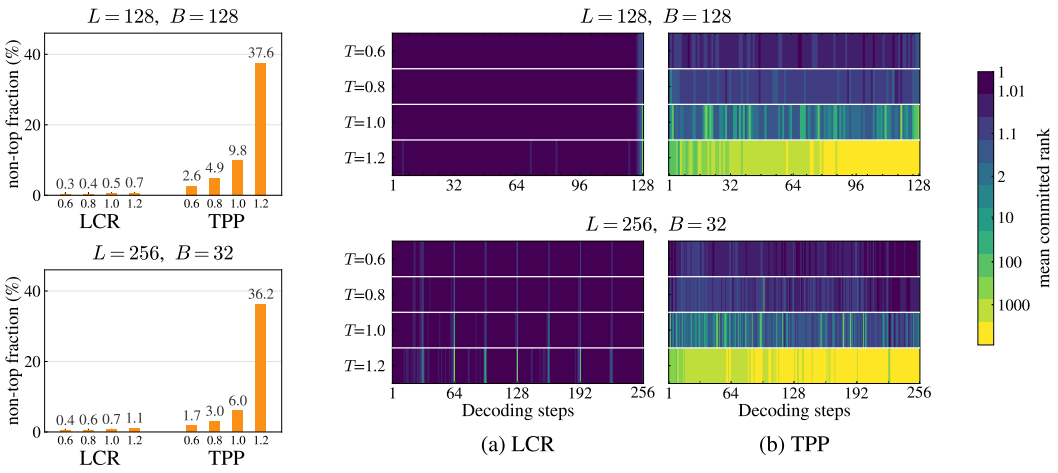

Figure 1: (**Left**) **Non-top commitment rate.** Higher temperature substantially increases non-top commitments under TPP but barely under LCR. Bars show the fraction of committed tokens that are non-top. (**Right**) **Committed-token rank over decoding.** LCR remains near rank one until the end of each sequence or block, whereas TPP commits higher-rank tokens throughout decoding as temperature increases. Heatmaps show the per-step mean local rank of committed tokens. Both panels use GSM8K with $L/B = 128/128$ and $256/32$. HumanEval results are provided in Fig. 5.

sequence, TPP generates a non-top sequence with probability $1 - \pi_T$. Under LCR, however, a non-top sequence is generated only when all L positions propose tokens from non-top sequences, since a single top-sequence proposal rejects every non-top proposal. Thus, the probabilities of generating a non-top sequence under TPP and LCR are

$$\boxed{\mathbb{P}_{\text{TPP}}(\text{non-top sequence}) = 1 - \pi_T, \quad \mathbb{P}_{\text{LCR}}(\text{non-top sequence}) = (1 - \pi_T)^L.}$$

TPP generates non-top sequences at exactly the probability specified by temperature, whereas under LCR that probability collapses exponentially with sequence length.

Concrete Example. Tab. 1 (Right) considers ten non-overlapping length-128 sequences, where the top-probability sequence has probability 0.2 and the other nine equally share 0.8. As $T \to \infty$, the tempered distribution assigns 90% probability to non-top sequences. TPP preserves this probability, whereas LCR suppresses it to 0.000139%.

Across both toy models, TPP preserves the non-top probability specified by temperature, whereas LCR can suppress it exponentially through competition across token proposals. By rejecting lower-probability proposals before commitment, LCR can therefore severely constrain the token and sequence choices that reach the final output, even at high temperature.

5 Low-Confidence Remasking Suppresses Diversity in Practice

Our analysis in Sec. 4 predicts strong suppression of lower-probability proposals under LCR before commitment. We test whether the same suppression occurs in a large-scale MDM, LLaDA-8B-Instruct (Nie et al., 2025), using GSM8K (Cobbe et al., 2021) and HumanEval (Chen et al., 2021). We generate 64 rollouts per prompt with generation length/block size $128/128$ and $256/32$. See App. C.1 for experimental details and additional results.

Non-top commitment rate. We compare LCR and TPP across four temperatures $T \in \{0.6, 0.8, 1.0, 1.2\}$ and measure the fraction of *non-top* commitments: tokens that are not the model's highest-probability token at their position at the time of commitment. Fig. 1 (Left) shows that increasing temperature sharply increases this fraction under TPP, reaching 37.6% and 36.2% at $T = 1.2$ for the $128/128$ and $256/32$ settings, whereas LCR remains at only 0.7% and 1.1%. Fig. 1 (Right) further tracks the mean local rank of the committed token at its position over decoding

steps. Under LCR, this rank remains almost entirely at **one** even as temperature increases, indicating near-greedy token selection. Ranks rise mainly near the end of each sequence or block, when only a few masked positions remain to compete. Under TPP, by contrast, higher temperature raises the committed-token rank throughout decoding. This pattern is consistent with Sec. 4.1: cross-position competition strongly suppresses non-top commitments under LCR until few competitors remain.

Proposal vs. commitment under LCR. Fig. 2 compares the untempered model probabilities of LCR's sampled proposals and committed tokens, pooled across decoding steps. As temperature increases, proposal probabilities shift downward, showing that LCR samples more lower-probability tokens. Yet committed-token probabilities remain concentrated at high values, typically above 0.9, across temperatures. Temperature thus changes what LCR proposes far more than what it commits. Lower-probability alternatives are sampled, then largely rejected before they can enter the generated sequence.

Despite this suppression, LCR can still benefit from multiple rollouts through variation in generation order and occasional non-top commitments near the end of each block (see App. C.2). However, these diagnostics show that most token-level exploration introduced by temperature is removed before commitment, substantially limiting the diversity contributed by token sampling.

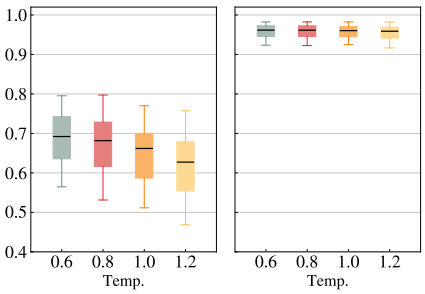

Figure 2: **LCR proposal/commitment.** Box plots show untempered model probabilities of sampled proposals (*proposal*) and committed tokens (*commitment*), pooled over decoding steps. GSM8K results with $L/B = 256/32$. HumanEval results in App. C.1.

6 Realizing the Potential of Arbitrary-Order Rollouts

Secs. 4 and 5 showed that LCR strongly suppresses lower-probability token commitments in both toy models and a large-scale MDM, revealing an inherent bias toward high-probability tokens. We now revisit the reported advantage of left-to-right (AR) over arbitrary-order generation in rollout coverage (Ni et al., 2026). We first examine whether this gap persists when LCR is replaced with TPP, and then whether even minimal use of arbitrary-order flexibility can further improve rollout diversity, solution coverage, and downstream policy optimization. See Apps. C and D for details.

6.1 Top-Probability Position Selection Closes the Rollout-Coverage Gap

Setup. We compare AR, LCR, and TPP using Pass@k on LLaDA-8B-Instruct (Nie et al., 2025) and LLaDA-1.5 (Zhu et al., 2026) across GSM8K (Cobbe et al., 2021), MATH-500 (Hendrycks et al., 2021), HumanEval (Chen et al., 2021), and MBPP (Austin et al., 2021b). For each prompt, we generate 64 rollouts and report Pass@k for $k \leq 64$. Following Ni et al. (2026), AR, LCR, and TPP use $T = 0.6$. Fig. 3 also includes the proposed method **EGI**, which we introduce in Sec. 6.2.

Results. Fig. 3 shows a consistent pattern across benchmarks and models: LCR yields substantially weaker Pass@k, whereas TPP achieves Pass@k comparable to AR. This indicates that the token suppression observed under LCR is associated with reduced solution coverage across rollouts.

These results revisit the conclusion of Ni et al. (2026), who attribute the higher Pass@k of AR over LCR to confidence-based decoding postponing uncertain positions. Crucially, TPP also prioritizes positions with high top-token probabilities and can therefore postpone uncertain positions, yet achieves Pass@k comparable to AR. These results indicate that the diversity loss arises from LCR's proposal-rejection mechanism rather than from confidence-prioritized position selection itself.

6.2 A Single Use of Order Flexibility Surpasses Autoregressive Rollouts

We next ask whether arbitrary-order flexibility can be leveraged further to improve rollout exploration. TPP prioritizes high-confidence positions and is not explicitly designed to promote diversity. We show that exploring this order flexibility only once, by changing the first committed position, can improve rollout diversity and solution coverage beyond AR.

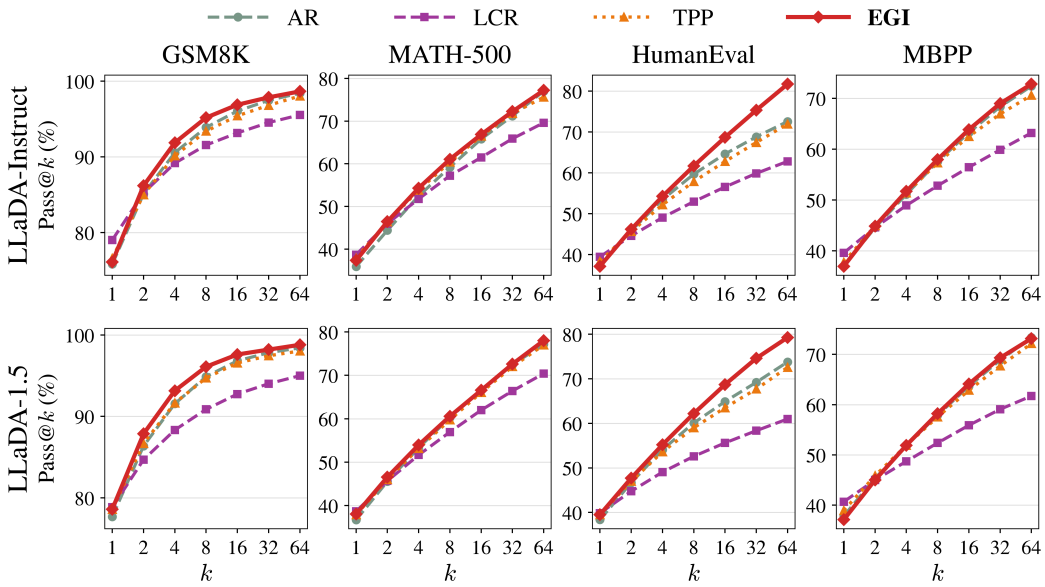

Figure 3: **Pass@k results.** Pass@k for AR and three arbitrary-order decoding rules (LCR, TPP, EGI) with LLaDA-8B-Instruct and LLaDA-1.5. LCR shows substantially weaker rollout coverage, TPP recovers performance comparable to AR, and EGI further improves Pass@k. For LCR, TPP, and EGI, the generation length/block size is set to $256/32$.

We propose **Entropy-Guided Initialization** (EGI), which selects the highest-entropy position at the first decoding step, $i^\star = \arg\max_{i \in M} -\sum_{v \in \mathcal{V}} p_i(v) \log p_i(v)$, samples and commits a token $x_{i^\star} \sim p_{i^\star}^{(T)}$, and follows TPP thereafter. In the rollout experiments in this subsection, the first commitment uses $T = 0.9$, while the subsequent TPP steps use $T = 0.6$. EGI thus uses order flexibility explicitly only at the first commitment, then follows regular TPP for the remaining steps.

Rollout diversity. Fig. 3 shows that EGI improves Pass@k over both TPP and AR, with particularly clear gains on GSM8K and HumanEval. We further compare EGI with AR using **Div-Self-BLEU** (Zhu et al., 2018; Yao et al., 2025), which measures lexical diversity across rollouts, and **Potential@k** (Yao et al., 2025), which measures how often problems missed by Pass@1 are recovered within k rollouts. Tab. 2 confirms that EGI also exceeds AR in both metrics, indicating that even a single use of order flexibility can improve rollout diversity and solution coverage.

Table 2: **Rollout diagnostics for EGI and AR.** Div-SB: Div-Self-BLEU; Pot.@k: Potential@k Higher values are better. HE denotes HumanEval. Metric definitions in App. C.4.

	Sampler	Div-SB	Pot.@8	Pot.@16	Pot.@64
GSM8K	AR	52.3	74.8	83.0	93.0
	EGI	**58.1**	**80.4**	**86.8**	**94.0**
HE	AR	57.2	34.5	41.8	54.5
	EGI	**58.8**	**38.5**	**49.1**	**69.7**

Effect of the first-step intervention. To analyze how EGI's single-token intervention affects TPP, we measure the mean pairwise normalized edit distance between the decoding-order permutations of different rollouts, with larger values indicating more divergent orders (see App. C.4 for metric definitions). Tab. 3 shows that EGI produces more diverse decoding orders than TPP despite differing only at the first step.

Table 3: **Decoding-order divergence.** Mean pairwise normalized edit distance between commitment orders across rollouts (mean $\pm$ SE). Higher values indicate more diverse decoding orders.

Sampler	GSM8K	HumanEval
TPP	0.45 $\pm$0.001	0.39 $\pm$0.006
EGI	**0.52** $\pm$0.0006	**0.52** $\pm$0.001

To test whether the gains come simply from increasing first-step temperature, we apply the same temperature increase to AR and observe much smaller gains (see App. C.5). These results suggest that selecting a high-entropy initial position through order flexibility contributes to EGI's gains beyond the temperature increase alone.

6.3 Entropy-Guided Rollouts Improve Policy Optimization

We finally show that the broader rollout exploration provided by EGI translates into improved downstream performance under group-relative policy optimization (GRPO) (Shao et al., 2024).

Setup. We compare AR and EGI for rollout generation using three recent group-relative policy optimization methods for large MDMs: SPG (Wang et al., 2026a), JustGRPO (Ni et al., 2026), and d2 (Wang et al., 2026b). For each method, we vary only the rollout decoding rule, using either AR or EGI, while keeping the policy optimization objective unchanged. We evaluate on GSM8K, MATH-500, HumanEval, and MBPP. For the coding tasks, training uses a subset of AceCoder-87K (Zeng et al., 2025), following Ni et al. (2026); Gong et al. (2026); Ou et al. (2026).

Table 4: **RL results with AR and EGI rollouts.** Best evaluation accuracy across checkpoints evaluated at fixed FLOP intervals (1×10^{19}) for SPG, JustGRPO, and d2. Within each method, only the training rollout sampler differs between AR and EGI. The better result is shown in **bold**.

	Sampler	GSM8K	MATH-500	HumanEval	MBPP
SPG	AR	82.11	37.8	38.4	42.8
	EGI	**84.08**	**39.6**	**40.2**	**43.4**
JustGRPO	AR	81.05	36.8	37.8	44.0
	EGI	**83.17**	**38.2**	**39.0**	**44.6**
d2	AR	81.73	37.2	37.2	43.6
	EGI	**85.29**	**38.8**	**38.4**	**44.2**

Following Wang et al. (2026b), we match total training compute across methods and evaluate checkpoints at fixed FLOP intervals, reporting the best evaluation accuracy. All training rollouts use group size $G = 8$ and a base temperature of $T = 0.6$, matching the best AR configuration in Ni et al. (2026). For EGI, the first commitment uses $T = 0.9$ for math and $T = 0.6$ for code. Evaluation uses deterministic LCR/TPP ($T = 0$) for all methods, which are equivalent under greedy decoding, consistent with prior post-RL evaluation protocols (Zhao et al., 2025; Tang et al., 2026; Ni et al., 2026). Experiment details in App. D.

Results. Tab. 4 shows that EGI consistently outperforms AR rollouts across benchmarks and all three policy optimization methods under matched training compute. This suggests that its broader rollout exploration translates into better downstream optimization outcomes.

Fig. 4 provides a closer look at the training dynamics. Compared with AR, EGI achieves faster gains in mean reward while maintaining higher within-group reward variance. Thus, EGI improves rollout reward while preserving the reward variation used for group-relative learning. Additional training dynamics across benchmarks in App. D.1.

Notably, EGI modifies only the first step before returning to standard TPP. That this limited use of order flexibility improves both rollout exploration and downstream policy optimization suggests that arbitrary-order generation offers a promising design space for diverse rollouts.

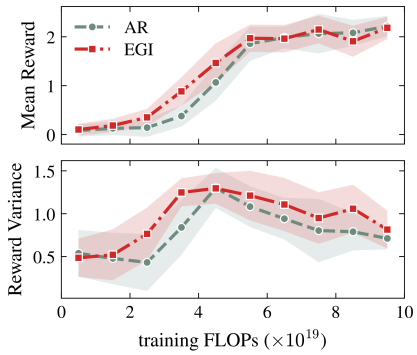

Figure 4: **Reward statistics during JustGRPO training on GSM8K.** Mean reward (top) and variance (bottom) for AR and EGI rollouts.

7 Conclusion

We distinguish two decoding rules that have often been conflated under the same label of confidence-based decoding: low-confidence remasking (LCR) and top-probability position selection (TPP). Through toy models and experiments with a large MDM, we show that LCR can severely suppress the lower-probability token exploration introduced by temperature, as cross-position rejection prevents most such proposals from reaching commitment and limits rollout diversity. In contrast, TPP directly follows the tempered distribution at the selected position and achieves Pass@k comparable to AR, while using order flexibility only at the first commitment to promote exploration further improves rollout diversity and downstream policy optimization. These results highlight the importance of distinguishing LCR from TPP and suggest that arbitrary-order flexibility offers a promising design space for diverse rollout generation beyond AR.

AI USE STATEMENT

We used generative AI tools to assist with writing and editing the manuscript and with implementing and debugging experimental code. All AI-assisted outputs were reviewed, revised, and cross-checked by the authors. We take responsibility for the final content of this work.

ETHICS STATEMENT

We are not aware of any specific ethical concerns raised by this work.

REPRODUCIBILITY STATEMENT

Complete proofs of all theoretical results are provided in App. B. We also provide the experimental settings and implementation details needed to reproduce our results in Apps. C and D.

REFERENCES

Marianne Arriola, Aaron Gokaslan, Justin T Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *ICLR*, 2025.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *NeurIPS*, 2021a.

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*, 2021b.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. In *NeurIPS*, 2025.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. Maskgit: Masked generative image transformer. In *CVPR*, 2022.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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

Liancheng Fang, Aiwei Liu, Henry Peng Zou, Yankai Chen, Enze Ma, Leyi Pan, Chunyu Miao, Wei-Chieh Huang, Xue Liu, and Philip S. Yu. Locally confident, globally stuck: The quality-exploration dilemma in diffusion language models. In *COLM*, 2026.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In *EMNLP-IJCNLP*, 2019.

Shansan Gong, Ruixiang Zhang, Huangjie Zheng, Jiatao Gu, Navdeep Jaitly, Lingpeng Kong, and Yizhe Zhang. Diffucoder: Understanding and improving masked diffusion models for code generation. In *ICLR*, 2026.

Satoshi Hayakawa, Yuhta Takida, Masaaki Imaizumi, Hiromi Wakaki, and Yuki Mitsufuji. Demystifying MaskGIT sampler and beyond: Adaptive order selection in masked diffusion. *TMLR*, 2026.

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

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *NeurIPS*, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham M Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In *ICML*, 2025a.

Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. Klass: Kl-guided fast inference in masked diffusion models. In *NeurIPS*, 2025b.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *ICML*, 2024.

Zanlin Ni, Shenzhi Wang, Yang Yue, Tianyu Yu, Weilin Zhao, Yeguo Hua, Tianyi Chen, Jun Song, Cheng Yu, Bo Zheng, and Gao Huang. The flexibility trap: Rethinking the value of arbitrary order in diffusion language models. In *ICML*, 2026.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *NeurIPS*, 2025.

Theo X. Olausson, Metod Jazbec, Xi Wang, Armando Solar-Lezama, Christian A. Naesseth, Stephan Mandt, and Eric Nalisnick. A tale of two temperatures: Simple, efficient, and diverse sampling from diffusion language models. In *COLM*, 2026.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *ICLR*, 2025.

Jingyang Ou, Jiaqi Han, Minkai Xu, Shaoxuan Xu, Jianwen Xie, Stefano Ermon, Yi Wu, and Chongxuan Li. Principled rl for diffusion llms emerges from a sequence-level perspective. In *ICLR*, 2026.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. *OpenAI Blog*, 2018.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 2019.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *NeurIPS*, 2024.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. In *NeurIPS*, 2024.

Xiaohang Tang, Rares Dolga, Sangwoong Yoon, and Ilija Bogunovic. wd1: Weighted policy optimization for reasoning in diffusion language models. In *ICLR*, 2026.

Chenyu Wang, Paria Rashidinejad, DiJia Su, Song Jiang, Sid Wang, Siyan Zhao, Cai Zhou, Shannon Zejiang Shen, Feiyu Chen, Tommi Jaakkola, Yuandong Tian, and Bo Liu. SPG: Sandwiched policy gradient for masked diffusion language models. In *ICLR*, 2026a.

Guanghan Wang, Yair Schiff, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. In *NeurIPS*, 2025.

Guanghan Wang, Gilad Turok, Yair Schiff, Marianne Arriola, and Volodymyr Kuleshov. d2: Improving reasoning in diffusion language models via trajectory likelihood estimation. In *ICML*, 2026b.

Yinjie Wang, Ling Yang, Bowen Li, Ye Tian, Ke Shen, and Mengdi Wang. Revolutionizing reinforcement learning framework for diffusion large language models. In *ICLR*, 2026c.

Gian Wiher, Clara Meister, and Ryan Cotterell. On decoding strategies for neural text generators. *TACL*, 2022.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. In *ICLR*, 2026.

Jian Yao, Ran Cheng, Xingyu Wu, Jibin Wu, and KC Tan. Diversity-aware policy optimization for large language model reasoning. In *NeurIPS*, 2025.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models. *arXiv preprint arXiv:2508.15487*, 2025.

Runpeng Yu, Xinyin Ma, and Xinchao Wang. Dimple: Discrete diffusion multimodal large language model with parallel decoding. *arXiv preprint arXiv:2505.16990*, 2025.

Huaye Zeng, Dongfu Jiang, Haozhe Wang, Ping Nie, Xiaotong Chen, and Wenhu Chen. Acecoder: Acing coder rl via automated test-case synthesis. In *ACL*, 2025.

Anthony Zhan. Simple policy gradients for reasoning with diffusion language models. In *ICML*, 2026.

Zeyang Zhang, Chengwei Liang, Xingyan Chen, Meiqi Gu, Minrui Luo, Jingzhao Zhang, and Tianxing He. Differences in text generated by diffusion and autoregressive language models. In *COLM*, 2026.

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. d1: Scaling reasoning in diffusion large language models via reinforcement learning. In *NeurIPS*, 2025.

Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. A reparameterized discrete diffusion model for text generation. In *COLM*, 2024.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. In *ACL*, 2026.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texygen: A benchmarking platform for text generation models. In *SIGIR*, 2018.

A Related Work

A.1 Prior Work on Diversity in Arbitrary-Order Decoding

Before discussing the closest prior work in detail, we summarize our positioning along three axes:

- **TPP, LCR, and their conflation.** The operational distinction between selecting among sampled tokens and selecting a position before token sampling is not itself new. Hayakawa et al. (2026); Fang et al. (2026) explicitly distinguish sample-then-choose from choose-then-sample procedures, while Zhang et al. (2026) show that resampling a token after LCR selects a position largely restores n-gram entropy. None of these works, however, focuses on the existing TPP and LCR rules being conflated under *confidence-based decoding*. Our focus is this conflation and why it matters for rollout sampling: the two rules define substantially different samplers when $T > 0$, so conflating them can lead to misleading conclusions.
- **Diversity suppression under LCR.** Prior work reports weak Pass@k and limited rollout coverage under LCR or closely related confidence-based decoding (Ni et al., 2026; Fang et al., 2026; Olausson et al., 2026), providing the empirical starting point for our analysis. Most directly, Zhang et al. (2026) theoretically establish entropy reduction under independent positions and empirically observe reduced n-gram entropy. We sharpen this analysis in an i.i.d. token model by showing exponential suppression of lower-probability choices and vanishing non-top fractions and normalized sequence entropy. We further show in an 8B LLaDA model that, even as sampling produces many lower-probability proposals, LCR commitments remain overwhelmingly concentrated on top tokens, and connect this suppression directly to reasoning rollout coverage.
- **Implications for arbitrary-order diversity.** Prior work responds to limited exploration in different ways, including favoring AR generation (Ni et al., 2026), introducing a separate position temperature (Olausson et al., 2026), and approximately targeting a globally tempered distribution (Fang et al., 2026). Our results show that much of the lost coverage can instead be recovered simply with TPP, which avoids LCR's proposal rejection and directly follows the requested token distribution at the selected position. EGI then uses order flexibility at only the first commitment to further improve coverage, highlighting arbitrary-order generation as a promising design space for diverse rollouts.

Ni et al. (2026) report lower Pass@k under arbitrary-order (AO) than AR decoding, using LCR as their main AO rule and entropy-based, margin-based, and random-order decoding as additional baselines. They explain this gap through *entropy degradation*, where confidence-based AO decoding postpones uncertain forking tokens until surrounding context narrows their possible branches. However, our results show that this conclusion is decoding-rule dependent. TPP still prioritizes high-top-probability positions and can postpone uncertain forks, yet achieves Pass@k close to AR, while EGI further improves rollout diversity and coverage over AR. These results indicate that the AR–LCR gap is driven primarily by LCR's rejection of lower-probability proposals rather than confidence-prioritizing ordering itself, and that AO flexibility remains a promising design space for diverse rollout sampling.

Zhang et al. (2026) provide the most directly related analysis of bias induced by LCR's proposal rejection. They attribute the bias to proposal-dependent selection favoring higher-probability outcomes, and prove the resulting entropy reduction under independent positions. They further show that resampling the token after LCR selects a position largely restores n-gram entropy, closely paralleling our observation that removing proposal rejection with TPP recovers rollout coverage. We quantify this suppression more sharply: in our i.i.d. model, non-top commitments are exponentially suppressed as more positions compete, and their final fraction vanishes as $O(1/L)$ (Prop. 2). At the entropy level, whereas Zhang et al. (2026) establish entropy reduction, we further show that the normalized sequence entropy itself vanishes as $O(\log L/L)$ (Prop. 4). We further observe in an 8B MDM that temperature produces many lower-probability proposals while LCR commitments remain overwhelmingly concentrated on top tokens, and connect this proposal rejection to rollout coverage and downstream policy optimization.

Hayakawa et al. (2026) explicitly distinguish the sample-then-choose structure of MaskGIT from choose-then-sample procedures and show through their moment-sampler analysis that proposal-dependent position selection can implicitly sharpen the token distribution. However, their analysis

of the MaskGIT sampler with position temperature α does not cover the $\alpha \to 0$ limit corresponding to LCR. Our work connects this operational distinction to the conflation of TPP and LCR under confidence-based decoding, and directly compares the two rules at the same token temperature to quantify the resulting suppression of non-top tokens under LCR.

Fang et al. (2026) analyze the quality–exploration trade-off using a TPP-style position-first setup and relate the resulting entropy bounds to the behavior of LCR (see App. A.2). Our analysis instead models LCR's proposal rejection directly and identifies cross-position rejection, rather than confidence-prioritized ordering itself, as the dominant source of diversity suppression in our toy settings. They improve Pass@k with an Independent Metropolis–Hastings sampler that approximately targets a globally tempered distribution. In contrast, we first remove LCR's proposal-rejection pathology with TPP and then obtain substantial additional gains from EGI, a one-step modification that highlights the untapped potential of AO flexibility.

Olausson et al. (2026) separate token and position temperature as controls over what to sample and where to decode, and use position temperature to improve rollout diversity. Their token-temperature analysis, however, uses a TPP-style position-first score and shows that, under its assumptions, all anchors precede the fork for every $T > 0$ (see App. A.2). Under LCR, token temperature changes the sampled proposals and substantially diversifies commitment order even while committed tokens remain overwhelmingly top tokens, providing much of LCR's remaining rollout variation (App. C.2). Thus, token and position temperature affect overlapping aspects of LCR, while neither generally eliminates the bias from proposal rejection at finite temperatures. Our analysis identifies this rejection as the dominant source of diversity suppression in our toy settings and removes it with TPP.

A.2 Conflation of TPP and LCR in Prior Work

TPP and LCR have both been described using terms such as *confidence-based decoding* or *confidence-based unmasking*. Position-first decoding as in TPP appears in Wu et al. (2026); Ben-Hamu et al. (2025), whereas proposal-based LCR is used under similar terminology in Wang et al. (2026a); Zhao et al. (2025); Tang et al. (2026); Zhan (2026). Note that these works either evaluate only at $T = 0$ or use implementations consistent with their descriptions, so the shared terminology is not problematic in practice. In several works, however, the rule described in the paper differs from that used in a released $T > 0$ implementation. The implementation comparisons below refer to publicly released code paths and do not necessarily reconstruct the exact code used for every reported experiment.

Wang et al. (2026c) describe TPP-style position selection using the maximum token probability at each position, while their released Dream rollout path at $T = 0.8$ uses LCR-style proposal rejection, ranking sampled proposals by their probabilities after temperature scaling and top-p/top-k filtering. Yu et al. (2025) describe TPP-style position selection based on proposal-independent, maximum-probability confidence in Alg. 1, while their released positive-temperature confidence path uses LCR-style proposal rejection based on the probabilities of sampled proposals. Wang et al. (2026b) describe d2-AnyOrder using TPP-style position selection before token sampling, whereas the released $T = 0.9$ rollout implementation uses LCR-style proposal rejection, committing only the two highest-scoring proposals. Kim et al. (2025b) define TPP-style position confidence by the maximum token probability in Def. 4.1, while the released Dream Top-k confidence baseline at $T = 0.2$ uses LCR-style proposal rejection based on sampled-proposal confidence. Under matched scoring and position-selection settings, these TPP-style and LCR-style decoding rules coincide at $T = 0$ but generally induce different sampling distributions at $T > 0$.

The same distinction is needed when interpreting theoretical analyses. Fang et al. (2026) derive an entropy bound under the condition $\max_{v \in \mathcal{V}} p_i(v) \geq 1 - \delta$, with the committed token at position i sampled from p_i. They distinguish sample-then-filter from rank-then-sample procedures, but the resulting entropy bound uses a TPP-style position-first abstraction that does not explicitly model LCR's proposal rejection. Similarly, Prop. 2 of Olausson et al. (2026) shows that, under an additional logit-gap assumption and with one token unmasked per step, all anchors precede the fork for every $T > 0$, but analyzes the position-first score $c_i = \max_v p_i^{(T)}(v)$ rather than LCR's sampled-proposal confidence. Thus, both analyses use position-first abstractions to interpret behavior under LCR, where proposal sampling and cross-position rejection define a different stochastic decoding

rule. By directly analyzing the LCR procedure and comparing it with TPP, we identify cross-position proposal rejection as a substantial contributor to diversity suppression in our settings.

A.3 Clarifying the Origins of TPP and LCR

We use Kim et al. (2025a) and Nie et al. (2025) as convenient reference formulations of TPP and LCR, respectively, rather than as claims about their independent origins. Kim et al. (2025a) study adaptive token ordering for MDMs, and their Top-K probability rule selects positions by maximum token probability before sampling tokens. For one-position commitment, this is the rule we call TPP. Nie et al. (2025) describe low-confidence remasking following MaskGIT (Chang et al., 2022). Their released $T > 0$ path first samples proposals and then selects positions using the probabilities of those proposals, which is the rule we call LCR.

Both procedures have broader precedents in iterative masked prediction and masked diffusion (Ghazvininejad et al., 2019; Chang et al., 2022; Zheng et al., 2024). Our terminology is therefore not intended to assign either decoding rule to a single originating work. It is intended to make explicit their operational difference at $T > 0$, where TPP selects a position before token sampling while LCR samples proposals before deciding which one to commit.

B Proofs and Additional Results

Both toy models follow the setups in Sec. 4 and use the TPP/LCR rules in Sec. 3. We start from a fully masked sequence and commit one token per step at temperature $T \in (0, \infty]$. LCR proposals are independently redrawn at each step conditional on the current sequence.

B.1 Toy Model I: Independent Tokens

Setup. We use the setting of Sec. 4.1, where every position independently follows the same strictly positive distribution p over a finite vocabulary $\mathcal{V}$ with a unique top-probability token. Recall that

$$c_T := \max_{v \in \mathcal{V}} p^{(T)}(v),$$

and let $N_{\text{non-top}}$ denote the number of non-top tokens in the final length-L sequence.

Proposition 1 (Non-top token probability)**.** *With m masked positions remaining,*

$$\mathbb{P}_{\text{TPP}}(\textit{non-top commitment}) = 1 - c_T, \qquad \mathbb{P}_{\text{LCR}}(\textit{non-top commitment}) = (1 - c_T)^m. \tag{1}$$

Proof. Under TPP, the selected position is sampled directly from $p^{(T)}$, so a non-top token is committed with probability $1 - c_T$.

Under LCR, any proposal of the unique top token rejects every non-top proposal. A non-top token is therefore committed if and only if all m positions propose non-top tokens. Since the proposals are independent draws from $p^{(T)}$,

$$\mathbb{P}_{\text{LCR}}(\text{non-top commitment}) = (1 - c_T)^m.$$

□

Proposition 2 (Expected non-top fraction)**.** *The expected non-top fractions satisfy*

$$\mathbb{E}_{\text{TPP}}\left[\frac{N_{\textit{non-top}}}{L}\right] = 1 - c_T, \qquad \mathbb{E}_{\text{LCR}}\left[\frac{N_{\textit{non-top}}}{L}\right] = \frac{(1 - c_T)[1 - (1 - c_T)^L]}{L c_T} \le \frac{1 - c_T}{L c_T} = O\left(\frac{1}{L}\right). \tag{2}$$

Proof. By Proposition 1, the probability of a non-top commitment is $1 - c_T$ under TPP and $(1 - c_T)^m$ under LCR when m positions remain. Summing over the L decoding steps gives

$$\mathbb{E}_{\text{TPP}}[N_{\text{non-top}}] = L(1 - c_T), \qquad \mathbb{E}_{\text{LCR}}[N_{\text{non-top}}] = \sum_{m=1}^{L} (1 - c_T)^m = \frac{(1 - c_T)[1 - (1 - c_T)^L]}{c_T}.$$

Dividing by L gives the result. □

B.2 Toy Model II: Non-Overlapping Sequences

Setup. We use the setting of Sec. 4.2, where the candidate length-L sequences differ at every position and there is a unique top-probability sequence. Recall that π_T denotes the probability of this sequence after tempering.

Proposition 3 (Non-top sequence probability)**.** *The probability of generating a non-top sequence is*

$$\mathbb{P}_{\text{TPP}}(\textit{non-top sequence}) = 1 - \pi_T, \qquad \mathbb{P}_{\text{LCR}}(\textit{non-top sequence}) = (1 - \pi_T)^L. \tag{3}$$

Proof. Because any two candidate sequences differ at every position, each token at any position identifies exactly one sequence. The first committed token therefore determines the entire generated sequence.

Before the first commitment, every position has the same top-token probability. Under TPP, any position may be selected and its token is sampled directly from the tempered distribution. The first token therefore identifies a non-top sequence with probability $1 - \pi_T$.

Under LCR, a proposal belonging to the top-probability sequence has higher model probability than any proposal belonging to a non-top sequence. A non-top sequence is therefore selected if and only if all L initial proposals belong to non-top sequences. Since these proposals are independent,

$$\mathbb{P}_{\mathrm{LCR}}(\text{non-top sequence}) = (1 - \pi_T)^L.$$

□

B.3 Additional Results

The token-level suppression in Sec. 4.1 also implies a stronger distributional consequence. Although TPP retains constant entropy per token, the entropy per token under LCR vanishes as the sequence length grows.

Proposition 4 (Final-sequence entropy)**.** *Let $X^{1:L}$ denote the final sequence in Toy Model I. Then*

$$H_{\mathrm{TPP}}(X^{1:L}) = LH(p^{(T)}), \qquad H_{\mathrm{LCR}}(X^{1:L}) = O(\log L),$$

so $H_{\mathrm{LCR}}(X^{1:L})/L = O(\log L/L) \to 0$.

Proof. Under TPP, every position is sampled directly from $p^{(T)}$, so

$$X^{1:L} \sim (p^{(T)})^{\otimes L},$$

which gives

$$H_{\mathrm{TPP}}(X^{1:L}) = LH(p^{(T)}).$$

Under LCR, let $r_i = \mathbb{P}(X^i \neq a)$. For each position,

$$H(X^i) \leq h(r_i) + r_i \log(|\mathcal{V}| - 1),$$

where h is the binary entropy function. By subadditivity of entropy and concavity of h,

$$H_{\mathrm{LCR}}(X^{1:L}) \leq Lh\left(\frac{\mathbb{E}_{\mathrm{LCR}}[N_{\text{non-top}}]}{L}\right) + \mathbb{E}_{\mathrm{LCR}}[N_{\text{non-top}}] \log(|\mathcal{V}| - 1).$$

Proposition 2 gives $\mathbb{E}_{\mathrm{LCR}}[N_{\text{non-top}}] = O(1)$, so the right-hand side is $O(\log L)$. □

Expanded numerical examples. Tab. 5 expands the examples in Tab. 1 across sequence lengths and temperatures. For reference, "TPP-equiv. T" is the TPP temperature that gives the same non-top probability as the corresponding LCR result. For K outcomes with top probability w and the remaining probability divided equally among the other $K - 1$ outcomes, it is computed as

$$T_{\text{TPP-equiv}}(u) = \frac{\log \frac{(K-1)w}{1-w}}{\log \frac{(K-1)(1-u)}{u}},$$

where u is the non-top probability to be matched. All entries in Tab. 5 are analytic evaluations rather than simulation estimates.

Table 5: **Expanded toy examples of diversity suppression under LCR.** Panel (a) uses the same 20-token distribution as Toy Model I, with top probability 0.1 and the remaining 0.9 shared equally among 19 tokens. Panel (b) uses the same ten non-overlapping sequences as Toy Model II, with top-sequence probability 0.2 and the remaining 0.8 shared equally among nine sequences. Probabilities and expected fractions are reported in percent. "TPP-equiv. T" is the TPP temperature matching the corresponding LCR result.

(a) Independent Token Generation (Toy I)

		$\mathbb{P}$(non-top token)			$\mathbb{E}[N_{\text{non-top}}/L]$		
L	T	LCR	Target (TPP)	TPP-equiv. T	LCR	Target (TPP)	TPP-equiv. T
32	0.5	0.1179	81.00	0.0771	13.31	81.00	0.1551
	1	3.434	90.00	0.1190	27.16	90.00	0.1901
	2	9.46	92.90	0.1436	37.00	92.90	0.2149
	∞	19.37	95.00	0.1710	47.87	95.00	0.2466
128	0.5	1.932×10^{-10}	81.00	0.0250	3.33	81.00	0.1184
	1	1.390×10^{-4}	90.00	0.0455	7.03	90.00	0.1352
	2	8.010×10^{-3}	92.90	0.0604	10.22	92.90	0.1460
	∞	0.1408	95.00	0.0786	14.82	95.00	0.1592
512	0.5	1.394×10^{-45}	81.00	0.0067	0.83	81.00	0.0967
	1	3.734×10^{-22}	90.00	0.0131	1.76	90.00	0.1072
	2	4.117×10^{-15}	92.90	0.0184	2.55	92.90	0.1135
	∞	3.931×10^{-10}	95.00	0.0256	3.71	95.00	0.1205

(b) Non-Overlapping Sequence Generation (Toy II)

		$\mathbb{P}$(non-top sequence)		
L	T	LCR	Target (TPP)	TPP-equiv. T
32	0.5	6.277×10^{-5}	64.00	0.0492
	1	0.07923	80.00	0.0869
	2	0.7206	85.71	0.1138
	∞	3.434	90.00	0.1465
128	0.5	1.553×10^{-23}	64.00	0.0137
	1	3.940×10^{-11}	80.00	0.0264
	2	2.697×10^{-7}	85.71	0.0370
	∞	1.390×10^{-4}	90.00	0.0517
512	0.5	5.810×10^{-98}	64.00	0.0035
	1	2.410×10^{-48}	80.00	0.0070
	2	5.287×10^{-33}	85.71	0.0100
	∞	3.734×10^{-22}	90.00	0.0144

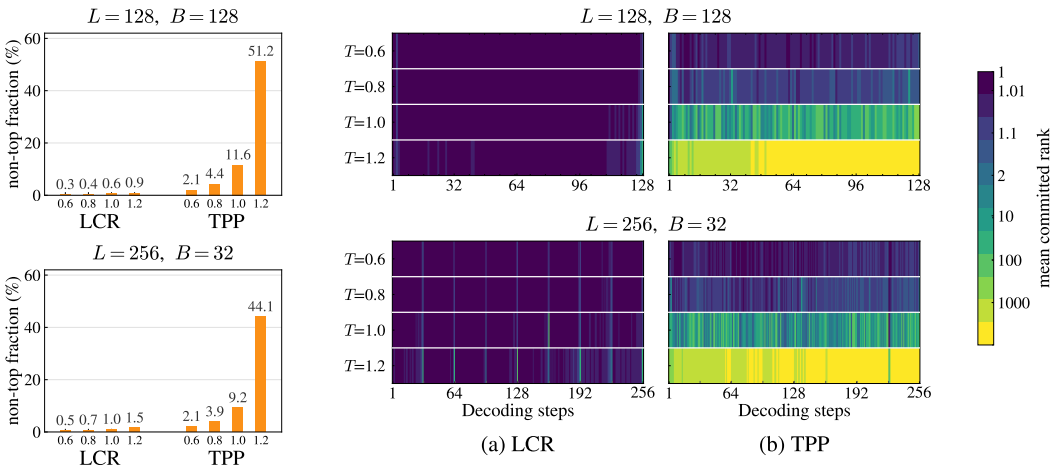

Figure 5: (**Left**) **Non-top commitment rate.** Higher temperature substantially increases non-top commitments under TPP but barely under LCR. Bars show the fraction of committed tokens that are non-top. (**Right**) **Committed-token rank over decoding.** LCR remains near rank one until the end of each sequence or block, whereas TPP commits higher-rank tokens throughout decoding as temperature increases. Heatmaps show the per-step mean local rank of committed tokens. Both panels use HumanEval with $L/B = 128/128$ and $256/32$.

C Experimental Details

C.1 Proposal and Commitment Diagnostics

We evaluate LLaDA-8B-Instruct on GSM8K and HumanEval using 64 rollouts per prompt, with generation length/block size configurations of $128/128$ and $256/32$. Across both benchmarks and configurations, LCR commits almost exclusively to the local argmax, the highest-probability token at each position at the time of commitment. Non-top commitments become appreciable only when very few masked positions remain, typically two or three (Figs. 1 and 5). Departures from the local argmax are thus concentrated near sequence or block boundaries, suggesting that cross-position competition suppresses them while many positions remain available.

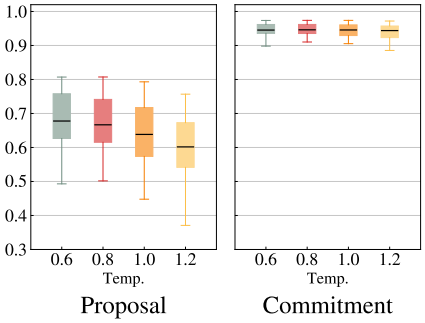

Figure 6: **LCR proposal/commitment.** Box plots show untempered model probabilities of proposed and committed tokens, pooled across decoding steps on HumanEval ($L/B = 256/32$).

The gap between proposals and commitments reveals how this behavior arises. LCR frequently proposes tokens with untempered model probabilities around 0.6–0.7, yet most lose to higher-probability proposals at competing positions. Regardless of the sampling temperature tested, committed tokens concentrate at much higher probabilities, often above 0.9 (Figs. 2 and 6). These observations suggest that LCR's near-argmax behavior arises from selection at the commitment stage: variability in sampled proposals need not translate into comparable variability in committed tokens.

C.2 Why LCR Still Gains from Multiple Rollouts

Although LCR rejects most lower-probability token proposals, its rollouts are not identical. Tab. 6 shows that increasing temperature increases both the normalized edit distance between generation orders and the token-level Hamming distance between final outputs. Here, normalized edit distance counts insertions and deletions divided by the combined sequence length, while normalized Hamming distance measures the fraction of token positions that differ, where higher distance indicates greater diversity.

Table 6: **Generation-order and output diversity under LCR.** For 64 rollouts per prompt, *Norm. Edit* and *Norm. Hamming* report the mean pairwise normalized edit distance between generation orders and the normalized Hamming distance between final token sequences, respectively, with values shown as mean ± SE. *Corr.* denotes the Pearson correlation between the two distances across pairs.

Dataset	T	Norm. Edit	Norm. Hamming	Corr.
GSM8K	0.6	0.32 ± 0.004	0.51 ± 0.007	0.94
	0.8	0.37 ± 0.003	0.59 ± 0.006	0.92
	1.0	0.42 ± 0.003	0.66 ± 0.006	0.86
	1.2	0.45 ± 0.002	0.71 ± 0.005	0.79
HumanEval	0.6	0.33 ± 0.009	0.56 ± 0.017	0.96
	0.8	0.38 ± 0.007	0.64 ± 0.014	0.94
	1.0	0.42 ± 0.006	0.72 ± 0.012	0.92
	1.2	0.45 ± 0.005	0.76 ± 0.010	0.91

The two distances are strongly correlated across rollout pairs (0.79–0.96). Different generation orders change the contexts available for later predictions, allowing order variation to propagate into different outputs.

Meanwhile, nearly all committed tokens remain the local argmax across temperatures (Figs. 1 and 5). Together, these observations suggest that generation order is a major source of LCR's remaining rollout variation, helping explain why Pass@k can improve with additional rollouts despite nearly greedy token commitments. However, this variation primarily explores different near-greedy trajectories, leaving alternatives that require lower-probability token commitments less explored.

C.3 Solution Coverage Comparisons

Following Ni et al. (2026), we compare AR, LCR, TPP, and EGI at a baseline temperature of $T = 0.6$ in Fig. 3. For EGI, only the first commitment uses $T = 0.9$; all subsequent commitments use $T = 0.6$. We generate $n = 64$ rollouts per prompt and report Pass@k following Chen et al. (2021):

$$\text{Pass@}\, k = \mathbb{E}_q \left[1 - \frac{\binom{n-c_q}{k}}{\binom{n}{k}} \right],$$

where c_q is the number of correct solutions among the n samples for prompt q.

C.4 Diversity and Potential Metrics

We report three metrics: Div-Self-BLEU measures surface-level diversity among generated solutions, Potential@k measures multi-rollout success with greater weight on problems with low single-rollout success, and normalized edit distance measures diversity in commitment orders.

Div-Self-BLEU. Following Yao et al. (2025), we report Div-Self-BLEU to measure inter-rollout diversity. For each prompt, each generated response is treated as a hypothesis while each of the remaining responses serves as a single reference in turn, and the resulting BLEU scores are averaged across response pairs. Since a lower Self-BLEU indicates greater diversity, we report

$$\text{Div-Self-BLEU} = 100 - \text{Self-BLEU}.$$

Div-Self-BLEU therefore ranges from 0 to 100, with higher values indicating greater diversity among the generated rollouts. Refer to Wiher et al. (2022) for additional details about BLEU metric.

Potential@k. We further report Potential@k (Yao et al., 2025), which measures the ability of additional rollouts to recover problems that are not solved by a single rollout. For N evaluation problems, it is defined as

$$\text{Potential@}K = \frac{\sum_{i=1}^{N} \text{Pass@}K(q_i)\,(1 - \text{Pass@}1(q_i))}{\sum_{i=1}^{N} (1 - \text{Pass@}1(q_i))},$$

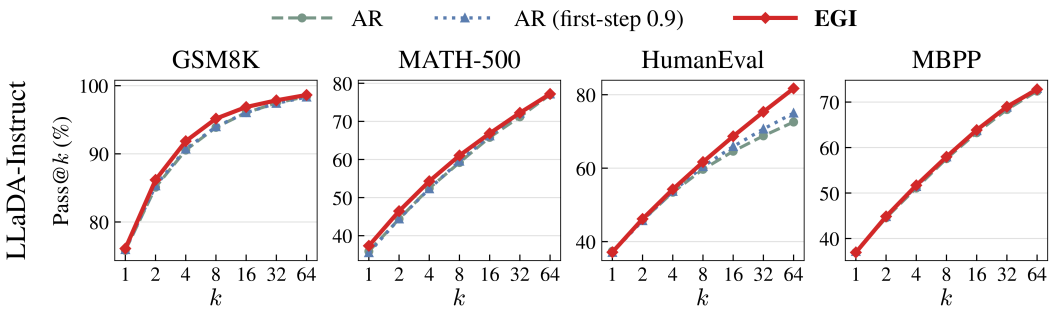

Figure 7: **Effect of first-step temperature on Pass@k.** Raising AR's first-step temperature to 0.9 yields performance close to standard AR, while EGI achieves substantially higher Pass@k.

where q_i denotes the i-th problem. Thus, Potential@k focuses on the additional solution coverage obtained from multiple rollouts beyond single-rollout performance.

Following Yao et al. (2025), we compute Pass@1 using deterministic LCR/TPP with $T = 0$, under which TPP and LCR produce the same decoding behavior.

Normalized edit distance. We measure decoding-order diversity by comparing the sequences of positions committed in different rollouts. For two commitment orders π and σ, we define

$$\mathrm{NED}(\pi, \sigma) = \frac{d_{\mathrm{ID}}(\pi, \sigma)}{|\pi| + |\sigma|},$$

where d_{ID} is the minimum number of insertions and deletions required to transform one sequence into the other; substitutions are not allowed. When both orders are permutations of the same n positions, this simplifies to

$$\mathrm{NED}(\pi, \sigma) = 1 - \frac{\mathrm{LCS}(\pi, \sigma)}{n},$$

where LCS denotes the length of the longest common subsequence. For each prompt, we average NED over all unordered pairs of rollouts, then average the resulting values across prompts. A value of zero indicates identical commitment orders, while larger values indicate greater order diversity.

For example, consider commitment orders $\pi = (1, 2, 3, 4)$ and $\sigma = (2, 3, 4, 1)$. Deleting position 1 from the beginning and inserting it at the end requires two edits, giving $\mathrm{NED} = 2/(4 + 4) = 0.25$. Equivalently, their longest common subsequence is $(2, 3, 4)$, so $\mathrm{NED} = 1 - 3/4 = 0.25$.

C.5 Effect of First-Step Temperature in AR

To test whether EGI's gains arise simply from increasing the first-step sampling temperature, we evaluate an AR variant that raises the temperature to 0.9 only at the first decoding step, matching EGI. All subsequent steps retain the standard AR temperature.

As shown in Fig. 7, this modification has only a limited effect, with Pass@k remaining close to that of standard AR. One possible explanation is that AR always begins at the fixed position immediately following the prompt, where predictive uncertainty may already be relatively low. Increasing the temperature at this position may therefore induce only limited additional variation. By comparison, EGI uses order flexibility to select a high-entropy initial position, where sampling at a higher temperature may have a larger effect on subsequent trajectories. This comparison suggests that the choice of the initial position may play an important role beyond the temperature increase alone.

D EXPERIMENTAL DETAILS ON GRPO

Training configuration. Following Zhao et al. (2025), we equip the attention and MLP projections of LLaDA-8B-Instruct with LoRA adapters (Hu et al., 2022), using rank 128, $\alpha = 64$, and dropout 0.05, while keeping the backbone frozen in 4-bit precision. This parameter-efficient setup has also been adopted or extended by Tang et al. (2026); Wang et al. (2026a;b). For consistency, we adapt the RL training pipeline of Ni et al. (2026) from full-parameter fine-tuning to this LoRA setup, preserving its likelihood definition and objective form.

Across all three GRPO objectives, we use a learning rate of 3×10^{-6} and four inner iterations per rollout batch. Each optimizer update uses eight prompts with eight rollouts per prompt, giving an effective batch size of 64 completions. Prompt tokens are randomly masked with probability 0.15.

The applicability of clipping and KL regularization depends on the objective. Following Zhao et al. (2025), we use a clipping range of $\epsilon = 0.5$ for objectives with importance-ratio clipping and a KL coefficient of $\beta = 0.04$ for objectives with a KL term.

Rollout configuration. We use a maximum generation length of 256, a block size of 32, and 8 rollouts per prompt in all training runs. Following the standard LLaDA-8B-Instruct generation protocol (Nie et al., 2025), we commit one masked token per decoding step, requiring 256 steps for a length-256 completion. AR and EGI share these settings and differ only in the sampling rule, requiring the same number of model evaluations and comparable computational costs per rollout.

Following the best-performing AR configuration in terms of Pass@k reported by Ni et al. (2026), we use a base temperature of $T = 0.6$ for both AR and EGI. For EGI's initial token sampling, we use $T = 0.9$ on math benchmarks and $T = 0.6$ on coding benchmarks.

Baselines. We compare three policy-optimization objectives, each crossed with both sampling rules. All three are group-relative: for a query q, a group of G responses is drawn from the old policy, and each response o_i is assigned an advantage $\hat{A}$ by standardizing its reward against the group statistics. They differ in how they handle the sequence likelihood. Two of them share the clipped surrogate

$$\mathcal{C}(\rho, \hat{A}) = \min\left(\rho\hat{A},\ \mathrm{clip}(\rho, 1-\epsilon, 1+\epsilon)\,\hat{A}\right), \tag{4}$$

which we write once here to keep the objectives below compact.

- **SPG-MIX.** Wang et al. (2026a) optimize bounds on the likelihood rather than an estimate of it, choosing the bound by the sign of the advantage:

$$\mathcal{J}_{\mathrm{SPG}}(\theta) = \mathbb{E}\Big[\tfrac{1}{G}\textstyle\sum_j \hat{A}^j \mathcal{L}^j\Big], \tag{5}$$

  where $\mathcal{L}^j = \mathcal{L}_{\mathrm{ELBO}}$ when $\hat{A}^j \geq 0$ and $\tilde{\mathcal{L}}_{\mathrm{EUBO}}$ otherwise, so a positively-advantaged trace raises a lower bound on its likelihood while a negatively-advantaged one lowers an upper bound — the sandwich the method is named for. Writing $w(t)$ for the weighting at diffusion time t, z_t for the noised sequence and $m_{t,i}$ for the indicator that position i is masked in z_t, the two bounds are

$$\mathcal{L}_{\mathrm{ELBO}} = \mathbb{E}_{t,z_t}\Big[\textstyle\sum_i w(t)\, m_{t,i} \log \pi_\theta(x_i \mid z_t)\Big], \tag{6}$$

$$\tilde{\mathcal{L}}_{\mathrm{EUBO}} = \tfrac{1}{\beta}\textstyle\sum_i \log \mathbb{E}_{t,z_t}\Big[w(t)\, m_{t,i}\, \pi_\theta^{\beta}(x_i \mid z_t)\Big], \tag{7}$$

  with β the bound temperature. Their mixed variant, which we use and refer to as SPG-MIX, replaces the negative branch with $\omega\tilde{\mathcal{L}}_{\mathrm{EUBO}} + (1-\omega)\mathcal{L}_{\mathrm{ELBO}}$. We set $\omega = 0.5$ and the bound temperature to 1.5, and estimate the likelihood at two diffusion times drawn from $[0, 1]$ under a block-random forward process.

- **JustGRPO.** Ni et al. (2026) forgo arbitrary order during RL training, sampling rollouts autoregressively so that the likelihood factorizes as

$$\log \pi_\theta^{\mathrm{AR}}(o \mid q) = \textstyle\sum_k \log \pi_\theta^{\mathrm{AR}}(o_k \mid o_{<k}, q), \tag{8}$$

  so that every generated position is scored exactly. This admits standard GRPO (Shao et al., 2024):

$$\mathcal{J}(\theta) = \mathbb{E}\Big[\tfrac{1}{G}\textstyle\sum_{i,k} \tfrac{1}{|o_i|}\mathcal{C}(\rho_{i,k}, \hat{A}_{i,k})\Big] - \beta D_{\mathrm{KL}}, \tag{9}$$

where $\rho_{i,k}$ is the token-level ratio of π_θ^{AR} to $\pi_{\theta_{\mathrm{old}}}^{\mathrm{AR}}$ at $o_{i,k}$ given $o_{i,<k}$ and q.

In our EGI condition, we change only the rollout sampler while retaining left-to-right likelihood evaluation and the same surrogate objective. This introduces a mismatch between the rollout distribution and the autoregressive likelihood used for optimization. Accordingly, we treat this setting as a JustGRPO-style surrogate with EGI rollouts, allowing us to study the effect of the rollout sampler while keeping the optimization rule fixed.

- **d2-StepMerge.** Wang et al. (2026b) decompose the trajectory likelihood over N merged segments rather than all T decoding steps, replacing $\prod_t \pi_\theta(x_t \mid x_{t+1})$ with $\prod_n \pi_\theta(x_{nT/N} \mid x_{(n+1)T/N})$, and optimize

$$\mathbb{E}\Big[\tfrac{1}{L}\textstyle\sum_{n,l} \mathbf{1}_{n,l}\, \mathcal{C}(\rho_n^l, \hat{A}^l)\Big] - \beta D_{\mathrm{KL}}, \tag{10}$$

where ρ_n^l is the ratio of π_θ to π_{old} at $x_{nT/N}^l$ given $x_{(n+1)T/N}^{1:L}$, and $\mathbf{1}_{n,l}$ selects the positions unmasked in segment n. Larger N tightens the decomposition at the cost of more model passes. Following the per-benchmark choice of Wang et al. (2026b) we use $N = 8$ for GSM8K and $N = 16$ for MATH-500 and code.

Benchmarks and rewards. We follow Zhao et al. (2025) for math prompts and rewards, and Ni et al. (2026) for coding, adjusting only the coding format reward to accept trailing text after the first code block, consistent with the evaluation harness. Prompts and rewards are identical across objectives within each benchmark.

Evaluation protocol. We compare all objectives and sampling rules under a matched training budget of 1×10^{20} accounted FLOPs, with evaluation every 1×10^{19} FLOPs. Each run therefore has ten evaluation checkpoints, and we report the highest accuracy among them. Evaluation uses each benchmark's full test set with the same deterministic decoding procedure across runs: LCR/TPP with temperature set to zero, which are equivalent under greedy decoding, consistent with prior post-RL evaluation protocols (Zhao et al., 2025; Tang et al., 2026; Ni et al., 2026). Training stops when the accumulated FLOP count reaches the budget; the number of optimizer steps is thus determined by each method's accounted compute cost rather than fixed in advance.

FLOP accounting. Following Wang et al. (2026b), we use a model-based FLOP accounting convention. We charge $2P$ per token for a no-gradient forward pass and $4P$ for an evaluation including activation backpropagation through the frozen backbone, where $P = 8 \times 10^9$. The additional gradient cost of the comparatively small LoRA adapters is neglected. These charges provide a hardware-independent compute proxy, not an exact measurement of implementation-level FLOPs.

The counter uses forward-row counts recorded by the implementation, multiplied by the full sequence length L (prompt plus completion) and the applicable per-token charge. Let G denote the group size across data-parallel processes, T the decoding steps per rollout, and M the per-completion likelihood evaluations defined above. The counted components are:

- **Rollout generation:** GT rows per generation batch, charged at $2P$ per token.
- **Old- and reference-policy likelihoods:** GM rows each time either policy's likelihood estimator is evaluated, charged at $2P$ per token.
- **Current-policy likelihood:** GM rows each time the likelihood estimator is evaluated with backpropagation, charged at $4P$ per token.

Costs accumulate whenever the corresponding computation occurs, following each objective's update schedule. Separate counters for rollout, old-policy, reference-policy, and current-policy costs are aggregated across data-parallel processes and logged at each training step, so the per-component breakdown is recorded for every run.

The budget excludes reward computation (including sandboxed code execution), optimizer arithmetic, tokenization, and periodic evaluation. It therefore measures accounted model computation during training rather than total training cost. The rollout protocol fixes $T = 256$. Likelihood costs depend on both the update schedule and M: $M = 2$ for SPG-MIX, $M = N$ for d2-StepMerge, and $M = 256$ for JustGRPO.

Hardware. Runs execute on a single node of four NVIDIA B200 GPUs under ZeRO-2 sharding with bfloat16 mixed precision. The number of prompts behind an optimizer update is fixed rather than derived from the device count, so the effective batch is identical across runs.

D.1 Additional Training Dynamics

We provide additional training dynamics beyond GSM8K on MATH-500 and coding tasks using a subset of AceCoder-87K. Fig. 8 compares the mean rollout reward and reward variance of AR and EGI under JustGRPO, plotted against accumulated training FLOPs.

Mean rollout reward tracks improvements in rollout quality, while within-group reward variance captures the reward differences that drive group-relative policy updates. Across both settings, EGI tends to achieve higher mean rollout rewards than AR at comparable training FLOPs while maintaining within-group reward variation. These results show a similar trend to that observed on GSM8K across additional mathematical reasoning and code generation task.

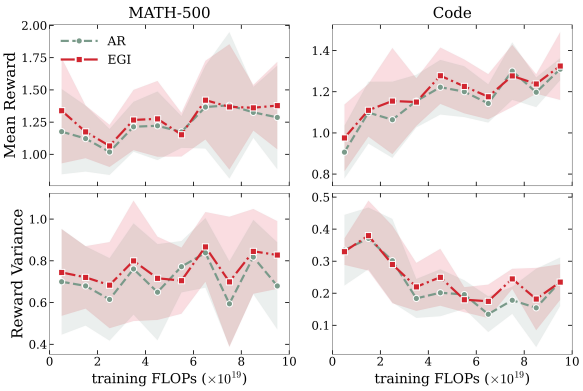

Figure 8: **Reward statistics during JustGRPO training.** Mean rollout reward and within-group reward variance for AR and EGI on MATH-500 and an AceCoder-87K subset, plotted against cumulative training FLOPs.