Decoding Language Diffusion Models with Foreseeing Movement

Anonymous ACL submission

Abstract

Language Diffusion Models (LDMs) benefit from a flexible decoding mechanism that enables parallelized inference and controllable generations over autoregressive models. Yet such flexibility introduces a critical challenge: inference performance becomes highly sensitive to the decoding order of tokens. Existing heuristic methods, however, focus mainly on local effects while overlooking long-term impacts. To address this limitation, we propose the Foreseeing Decoding Method (FDM), a novel approach that integrates both local and global considerations to unlock the full potential, employing a search-based strategy to enable effective optimization in discrete spaces. Furthermore, by analyzing the consistency of chosen tokens in the full decoding process, we develop a variant, FDM with Acceleration (FDM-A), which restricts deep exploration to critical steps identified as the exploration and balance circumstances. Extensive experiments across diverse benchmarks and model architectures validate the scalability of FDM and demonstrate the superior efficiency-performance trade-off achieved by FDM-A. Our work might potentially provide a principled step toward more powerful decoding methods for LDMs.

1 Introduction

In recent years, diffusion models (Ho et al., 2020; Rombach et al., 2022) have emerged as a promising competitor in natural language modeling (Nie et al., 2025; DeepMind, 2025), challenging the dominance of auto-regressive generation (Jaech et al., 2024; Guo et al., 2025; Team et al., 2025). Instead of generating tokens sequentially from left to right and token by token, Language Diffusion Models (LDMs) can generate multiple tokens in parallel, making it highly efficient at the inference stage (Li et al., 2025; Khanna et al., 2025). In addition, owing to pipelines of processing denoising and the bidirectional modeling, LDMs can outperform the auto-regressive models in broad aspects such as reverse reasoning (Berglund et al., 2024), controllable generation (Xiong et al., 2025; Li et al., 2022), or multi-modal reasoning (Li et al., 2025).

Table 1: The performances of LLaDA with various decoding orders on the ARC (Clark et al., 2018) benchmark.

	Random	Margin	FDM-A
Accuracy (↑)	79.06	82.55	**86.30**
Tokens/Seconds (↑)	12.01	10.85	**38.20**

However, high flexibility is a double-edged sword, as it can lead to performance degradation if sampling paths are chosen inappropriately, attributing to the convergence issues (Li and Cai, 2025) and the non-convexity of the optimization goal (Kim et al., 2025; Peng et al., 2025; Ben-Hamu et al., 2025). For example, as shown in Table 1, compared to decoding with a random order, decoding answers with the order of the largest marginal probability yields an improvement on the ARC benchmark (Clark et al., 2018) from 79.06% to 82.55%, underscoring its critical importance. To address this issue, previous works have mainly developed strategies from a heuristic perspective. For example, pioneer works (Nie et al., 2025; Zheng et al., 2024) propose to decode by enlarging the predicted probability. Follow-up works argue that probability margins (Kim et al., 2025) and the entropy of the predicted distribution (Ben-Hamu et al., 2025) are better substitutes, considering that LDMs may be confused when the probabilities of the topmost tokens at the given position are near. However, though effective, previous heuristic approaches decode tokens with local information at each step, overlooking the long-term impacts underlying the full decoding paths without leveraging the full potentiality of LDMs at the inference stage.

Therefore, in this paper, with further analyzing the decoding formulation of LDMs, we find that

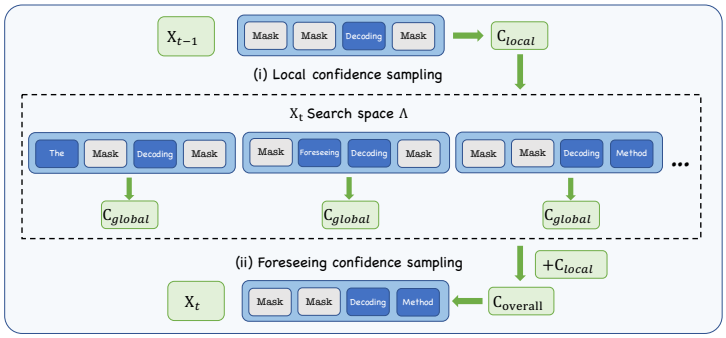


Figure 1: The pipeline of FDM. We first compress the search space into a small set Λ by filtering out candidates of lower local confidence *i.e.* C_{local}. In the final, we incorporate both local and global confidence to decide the ultimate choice at step t.

it can be divided into two components. The first one is termed as **local confidence** in Figure 1 (i), reflecting the uncertainty of models in predicting a given token and widely adopted by the heuristic methods. The other is the **global confidence**, capturing the future impact in decoding a specific token. It is not directly accessible and typically ignored by prior methods. However, we demonstrate that the training goal of LDMs provides a good approximation. By further combining both of them into considerations, we propose **F**oreseeing **D**ecoding **M**ethod (FDM), illustrated in Figure 1. To ensure the computation is affordable in reality, the decoding process of FDM is divided into two stages: We firstly rank the candidate tokens with their local confidence. In the second competition, the overall criterion of both local and global confidence is adopted to decide the final tokens chosen for decoding. We not only theoretically prove FDM will achieve lower errors compared to heuristic methods but also perform experiments across multiple benchmarks to illustrate that the benefit of it can further improve by enlarging the search space. It demonstrates FDM will serve as a test-time scaling (Snell et al., 2024; Bi et al., 2024) method specifically designed for LDMs.

Furthermore, by calculating the consistency ratio of FDM and the simple confidence-based decoding, we find that a global view is not always necessary at every step, especially when the context is sufficient for reliable decisions. Motivated by this observation, we propose an accelerated version of FDM (FDM-A). It will perform an adaptive exploration from the probability feedback, switching to FDM only when all candidates have low confidence or borderline scenarios. As shown in Table 1, FDM-A can not only accelerate the decoding process with over $3\times$ speed-up but also obtain notable performances. In summary, our contributions are listed as follows:

- We propose the **F**oreseeing **D**ecoding **M**ethod (FDM) to schedule the decoding orders of LDMs. Unlike heuristic approaches, it takes both the local and global confidence as the criterion for decision, guaranteeing a lower divergence with the natural distribution.
- Motivated by the agreement ratio in decoding, we further accelerate FDM with an adaptive strategy (FDM-A). It adaptively performs exploration when the context is rare and parallel decodes tokens if models have enough context for complete generation.
- The experiments across multiple benchmarks and variants of LDMs are performed, manifesting the scalability of FDM and the outstanding performance of FDM-A in balancing efficiency and performance.

2 Related Work

Language Diffusion Models (LDMs). As one of the most successful types of generative models, the popularity of diffusion models has been well-recognized in the vision domain (Ho et al., 2020; Peebles and Xie, 2023; Rombach et al., 2022). At the training stage, the model predicts the noises added to the natural inputs. Then, at the test time, by utilizing the model to denoise step by step, diffusion models recover the original data through iterative refinement. Inspired by their great success, early investigations such as D3PMs (Austin et al., 2021), DiffuSeq (Gong et al., 2023) have demonstrated the potential of the diffusion pipeline in language tasks. However, limited to the small scales, their performances are considerably inferior to those of their auto-regressive competitors. Recently, LLaDA (Nie et al., 2025; Zhu et al., 2025a) scales the model parameters to billions, considered as one of the most representative works in

Language Diffusion Models (LDMs). To save the computational costs, other architectures like Dream (Ye et al., 2025), DiffuGPT (Gong et al., 2024), and Dream-Coder (Xie et al., 2025) perform continual training on the pretrained weights of the auto-regressive counterparts. Built upon the outstanding performances in the language capacity, language diffusion models have also been applied to multi-modal applications such as biomedical understanding (Dong et al., 2025) and chart understanding (You et al., 2025). But in this paper, we propose FDM to improve the performance of LDMs at the inference stage.

Decoding Order of LDMs. With the rapid development of LDMs, their shortcomings are also disclosed. One of the most prominent ones is the vital significance of the decoding orders (Kim et al., 2025). This is because when the context is limited, models will fail to accurately predict all tokens. In other words, the uncertainty of predictions can be measured with the output probability. This motivates the heuristic-based approaches, including decoding with the largest probability value (Nie et al., 2025; Zheng et al., 2024), with the largest margin probability (Kim et al., 2025), and with the least entropy of distribution (Ben-Hamu et al., 2025). Although heuristic methods improve the performance of LDMs a lot compared to decoding with the random order, they make decisions based on local confidence, ignoring the sequential consequence to other tokens. This limitation may bring errors to the generated sequences. Thus, in this paper, we propose FDM and FDM-A, which incorporate the global confidence to depict the future cascading effect in the decoding process. .

Acceleration of LDMs. Although LDMs map to the complete answer at each step, they will suffer a quality-efficiency trade-off with a varying decoding step (Feng et al., 2025; Nie et al., 2025). In addition, to achieve the bidirectional attentions, the attention maps of LDMs are not causal, requiring model-agnostic methods like KV-Cache for adaptations. Therefore, in a series of works (Wu et al., 2025; Hu et al., 2025; Ma et al., 2025), they propose their own approximation acceleration for reusing the Key-Value, largely accelerating the inference speed. More related to our work, (Hong et al., 2025) and (Ben-Hamu et al., 2025) both propose samplers for dynamic decoding in LDMs. But (Hong et al., 2025) purely focus on the parallel decoding for acceleration and (Ben-Hamu et al., 2025) propose WINO for revoking suspicious error tokens. While in our paper, we propose FDM-A to balance the explorations and accelerations in decoding, achieving a better trade-off.

3 Preliminaries

Unlike auto-regressive models that generate answers in a sequential way, LDMs define a Markov chain that revises the answer step-by-step with a denoised process. Given the user query $\mathbf{q}$ and the vocabulary space $M = \{1, 2, ...m\}$, the generation of final answer $\mathbf{x}_T$ with length L under the data distribution p_{data} can be formulated as:

$$\begin{aligned}\mathbf{x}_T &= \arg\max_{\mathbf{x}_T} p_{data}(\mathbf{x}_{0:T}|\mathbf{q}) \\ &= \arg\max_{\mathbf{x}_T}\{p(\mathbf{x}_0)\prod_{\alpha=1}^{T-1} p_{data}(\mathbf{x}_\alpha|\mathbf{q}, \mathbf{x}_{0:\alpha-1})\},\end{aligned} \tag{1}$$

where $p(\mathbf{x}_0)$ is sampled from an initial noise distribution which is independent to $\mathbf{q}$. $\mathbf{x}_\alpha$ is a partially masked sequence at the α step. To analyze the influence of the intermediate variable $\mathbf{x}_t$, we can regroup the terms associated with step t as:

$$\begin{aligned}\mathbf{x}_T = \arg\max_{\mathbf{x}_T}\{&p_{data}(\mathbf{x}_{t+1:T}|\mathbf{q}, \mathbf{x}_{0:t}) \\ &\cdot p_{data}(\mathbf{x}_t|\mathbf{q}, \mathbf{x}_{t-1}) \cdot \prod_{\alpha=1}^{t-1} p_{data}(\mathbf{x}_\alpha|\mathbf{q}, \mathbf{x}_{0:\alpha-1})\}.\end{aligned} \tag{2}$$

Although the above equation shows that $\mathbf{x}_t$ can impact future variables from $\mathbf{x}_{t+1}$ to $\mathbf{x}_T$, heuristic decoding strategy, π_H, commonly simplify the procedure by focusing solely with the local partition:

$$\mathbf{x}_t = \arg\max_{\mathbf{x}_t} p_\theta(\mathbf{x}_t|\mathbf{q}, \mathbf{x}_{t-1}), \tag{3}$$

where θ is a parameterized neural network introduced by LDMs to model the natural distribution. According to (Nie et al., 2025), the optimization target of LDMs can be given as:

$$\mathbb{E}_{\substack{\mathbf{x}_T\sim p_{data}\\ t\in[0,T]}}\Big[\frac{1}{n}\sum_{j=1}^{n}\mathbf{1}[\mathbf{x}_t^{(j)} = \texttt{Mask}] \odot \log p_\theta(\mathbf{q}, \mathbf{x}_t^{(j)})[\mathbf{x}_T]\Big], \tag{4}$$

where $\odot$ is the Hadamard product and $[\mathbf{x}_T]$ represents the probability value of the tokens in $\mathbf{x}_T$. With a well-trained model, LDMs learn to approximate the conditional log-probability of future tokens in $\mathbf{x}_m$ given the current masked state, $\mathbf{x}_t$ $(m > t)$ and the user query $\mathbf{q}$. Specifically,

$$\begin{aligned}&\log p_\theta(\mathbf{x}_m|\mathbf{x}_t, \mathbf{q}) \\ &= \mathbf{1}\begin{bmatrix}\mathbf{x}_m \neq \texttt{Mask} \\ \&\&\ \ \mathbf{x}_t = \texttt{Mask}\end{bmatrix} \odot \log p_\theta(\mathbf{q}, \mathbf{x}_t)[\mathbf{x}_m].\end{aligned} \tag{5}$$

which follows from the fact that the model predicts the token distribution for any masked position independently.

4 Methodology

4.1 The Foreseeing Decoding Method

Although Equation 3 establishes a simple formulation in decoding $\mathbf{x}_t$, ignoring the contribution of $\mathbf{x}_t$ in $p_{data}(\mathbf{x}_{t+1:T}|\mathbf{q}, \mathbf{x}_{0:t})$ might mislead the decoding process, leading the unexpected errors in the generative response. Therefore, differing from previous works, our proposed Foreseeing Decoding Method (FDM) aims to decode tokens that achieve the largest value across the overall formulation and its decoding strategy, π_F can be given as:

$$\mathbf{x}_t = \arg\max_{\mathbf{x}_t} p_\theta(\mathbf{x}_T|\mathbf{q}, \mathbf{x}_t) \cdot p_\theta(\mathbf{x}_t|\mathbf{q}, \mathbf{x}_{t-1}). \quad (6)$$

Here we replace $p_{data}(\mathbf{x}_{t+1:T}|\mathbf{q}, \mathbf{x}_{0:t})$ with $p_\theta(\mathbf{x}_T|\mathbf{q}, \mathbf{x}_t)$ owing to the property of the Markov Chain and the modeling of LDMs. From the theoretical perspective, we also prove that under π_F, the generative distribution achieves lower KL divergence with the natural distribution p_{data} than that of π_H.

Theorem 1. *Let* $\Delta_{total} \triangleq \sum_{t=1}^{T} \mathbb{E}_{p_{data}(\mathbf{x}_{t-1})}\big[\mathcal{I}_{p_{data}}(\mathbf{x}_t; \mathbf{x}_T|\mathbf{x}_{t-1})\big]$, *where* $\mathcal{I}_{p_{data}}(\mathbf{x}_t; \mathbf{x}_T|\mathbf{x}_{t-1})$ *is the conditional mutual information under q. Then*

$$D_{\mathrm{KL}}(p_{\mathrm{data}}(\mathbf{x}) \,\|\, p_{\pi_F}) = D_{\mathrm{KL}}(p_{\mathrm{data}}(\mathbf{x}) \,\|\, p_{\pi_H}) - \Delta_{\mathrm{total}}. \quad (7)$$

For the complete proof, please refer to Appendix A for more details. Due to the fact that mutual information is non-negative, we derive that the generative distribution under π_F has lower KL divergence with the natural distribution p_{data} than π_H. By further applying the log transformation to Equation 6, we can obtain,

$$\arg\max_{\mathbf{x}_t} \{\log p_\theta(\mathbf{x}_T|\mathbf{q}, \mathbf{x}_t) + \log p_\theta(\mathbf{x}_t|\mathbf{q}, \mathbf{x}_{t-1})\}. \quad (8)$$

With deeper analysis, we find the first term captures how decoding a specific token influences the future (the **global** confidence, C_{global}), while the second term reflects the model's confidence in decoding it at step t (the **local** confidence, C_{local}). According to Equation 5, C_{global} can be estimated by:

$$\begin{aligned} C_{global} &= \log p_\theta(\mathbf{x}_T|\mathbf{x}_t, \mathbf{q}) \\ &= \mathbf{1}\begin{bmatrix} \mathbf{x}_m \neq \texttt{Mask} \\ \&\& \;\; \mathbf{x}_t = \texttt{Mask} \end{bmatrix} \odot \log p_\theta(\mathbf{q}, \mathbf{x}_t)[\mathbf{x}_T]. \end{aligned} \quad (9)$$

Regarding $\mathbf{x}_T$ is a full response without any mask, $\mathbf{x}_T \neq \texttt{Mask}$ can be omitted since it holds for every position. In addition, instead of greedily decoding $\mathbf{x}_T$ from the model output to calculate C_{global}. We compute the expectation over the entire model output distribution:

$$C_{global} = \mathbf{1}[\mathbf{x}_t = \texttt{Mask}] \odot \mathbb{E}_{p_\theta} \log p_\theta(\mathbf{q}, \mathbf{x}_t). \quad (10)$$

In addition, with Equation 5, we can formulate C_{local} with:

$$\begin{aligned} C_{\mathrm{local}} = \mathbf{1}\big[\mathbf{x}_t \neq \texttt{Mask} \;\&\&\; \mathbf{x}_{t-1} = \texttt{Mask}\big] \\ \odot \log p_\theta(\mathbf{q}, \mathbf{x}_{t-1})[\mathbf{x}_t]. \end{aligned} \quad (11)$$

Similar to the denotation in Equation 5, here $[\mathbf{x}_t]$ means the predicted probability of each token in $\mathbf{x}_t$. After combining both equations, we have:

$$\begin{aligned} &\arg\max_{\mathbf{x}_t} \; \mathbf{1}[\mathbf{x}_t = \texttt{Mask}] \odot \mathbb{E}_{p_\theta} \log p_\theta(\mathbf{q}, \mathbf{x}_t) \\ &+\mathbf{1}[\mathbf{x}_t \neq \texttt{Mask} \;\&\&\; \mathbf{x}_{t-1} = \texttt{Mask}] \odot \log p_\theta(\mathbf{q}, \mathbf{x}_{t-1})[\mathbf{x}_t]. \end{aligned} \quad (12)$$

Note that in this equation, $p_\theta(\mathbf{q}, \mathbf{x}_{t-1})$ is independent of $\mathbf{x}_t$, which can be accurately calculated by querying the model θ with the acquired sequence $\mathbf{x}_{t-1}$ and user prompt $\mathbf{q}$. In contrast, the $p_\theta(\mathbf{q}, \mathbf{x}_t)$ takes the discrete variable $\mathbf{x}_t$ as a part of the input. It means every evaluation involves a single forward pass. To resolve this problem, we introduce the hyperparameter K to compress the search space for efficiency. Specifically, we first obtain the candidate token set, *i.e.*, Cand, from the predictions of the model parameterized by θ at the masked positions:

$$\big\{\mathbf{1}[\mathbf{x}_{t-1} = \texttt{Mask}] \odot \arg\max_{\mathbf{x}} \; p_\theta(\mathbf{q}, \mathbf{x}_{t-1})\big\}. \quad (13)$$

Furthermore, to avoid being trapped in the local optima, we also incorporate a dynamic pruning strategy that retains only candidate tokens whose confidence exceeds the predefined threshold γ. Then, C_{local} is introduced as the metric to rank each possibility of $\mathbf{x}_t$ and we only keep the Top-K candidates for further decision. Thus, the search space Λ can be defined as:

$$\Lambda = \Big\{\mathbf{x}_t \in \text{Top-K}(\{\mathbf{x}_t\}) \;\Big|\; C_{\mathrm{local}}(\mathbf{x}_t) > \gamma\Big\}. \quad (14)$$

Defined on Λ, our proposed **F**oreseeing **D**ecoding **M**ethod (FDM) is formulated as follows,

$$\mathbf{x}_t = \begin{cases} \arg\max_{\mathbf{x}_t \in \{\mathbf{x}_t\}} C_{local}, & \Lambda = \varnothing, \\ \arg\max_{\mathbf{x}_t \in \Lambda} \{C_{local} + C_{global}\}, & \Lambda \neq \varnothing. \end{cases} \quad (15)$$

We also summarize the whole decoding process of FDM at the t step in Algorithm 1 in Appendix B.

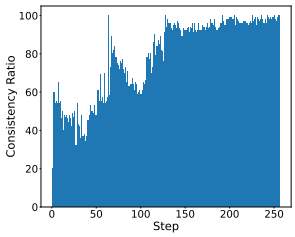


Figure 2: The consistency ratio of selecting the next decoding token using C_{local} alone versus both C_{local} and C_{global}. The decisions of both strategies are made based on the same $\mathbf{x}_{t-1}$ in each step. Peak points are observed on the steps of 64 and 128 because we follow the proposed semi-autoregressive pipeline in (Nie et al., 2025) with the block size 64.

4.2 Acceleration with the Foreseeing Decoding Method

Based on the analysis in Section 4.1, in this section, we further investigate a variant, *i.e.*, FDM-A to better balance the decoding speed and performance. In Figure 2, we calculate the consistency ratio in token selection with local confidence only (heuristic decoding) versus both local and global confidence. We average the results over the first 100 examples in the GSM8K (Cobbe et al., 2021) test set. The experiments are performed on LLaDA (Nie et al., 2025), with $\gamma = 0.6$ and $K = 2$.

In the early stages of decoding, due to limited context, the overlap between the two decoding strategies is relatively low, at approximately 50%. However, as decoding progresses, the degree of overlap gradually increases and finally exceeds 90%. It is consistent with the empirical observation in Figure 3, the final decoding sequence ($\mathbf{X}_T$) is largely shaped when t is small. But when t is approaching T, most unmasked tokens have been determined by the adequate contexts, indicating the marginal impact of decoding orders. This evidence demonstrates that more exploration from the global confidence is needed at the beginning, while at the final stage, we only need to adopt the local confidence to accelerate the decoding.

Motivated by this observation, we divide the entire process into three stages: **the exploration, balance and acceleration phases** by incorporating two hyperparameters, *i.e.* η_1 and η_2. Firstly, in the exploration phase, if there are no tokens whose prediction probability exceeds η_1, we will perform exploration with FDM. We denote it with $\texttt{FDM}_1$ ($\mathbf{x}_{t-1}$, $\mathbf{q}$, n), initialized with the K_1. n is the number of tokens decoded in this step and we set it to 1 for reliability. Thus $\mathbf{x}_t$ can be given as:

$$\texttt{FDM}_1(\mathbf{x}_{t-1}, \mathbf{q}, n = 1, \gamma = \gamma_1). \tag{16}$$

In the balance phase, we focus on the intermediate states of the decoding process. We combine the exploration and acceleration to achieve a good trade-off. Specifically, we refer to the tokens whose probability lies between η_1 and η_2 as the "Borderline Tokens" and those that have the prediction probability greater than η_1 as the "Qualified Tokens". By applying FDM, we weigh options within a collection comprising both sets. Let $\mathcal{Q}_{\eta_1} \triangleq [\mathbf{x}_{t-1} = \texttt{Mask}]p_\theta(\mathbf{q}, \mathbf{x}_{t-1}) > \eta_1$ denote the qualified-token condition, and let $\texttt{NUM}(\cdot)$ count the number of tokens satisfying it. The formulation of decoding $\mathbf{x}_t$ is:

$$\texttt{FDM}_1\Big(\mathbf{x}_{t-1}, \mathbf{q}, n = \texttt{NUM}(\mathcal{Q}_{\eta_1}), \gamma = \eta_2\Big). \tag{17}$$

As shown in Figure 3, different decoding methods perform almost the same at the tail, indicating the convergence in decoding. At this time, we shift the mode to the acceleration phase. The tokens are decoded with the local confidence only to achieve the fastest speed. The decoding function can be regarded as a special case of FDM. By initializing it with $K = 1$, we represent it by $\texttt{FDM}_2(\mathbf{x}_{t-1}, \mathbf{q}, n)$ and $\mathbf{x}_t$ can be decoded with:

$$\texttt{FDM}_2\Big(\mathbf{x}_{t-1}, \mathbf{q}, n = \min\big(\texttt{NUM}(\mathcal{Q}_{\eta_1}), N\big), \gamma = 1.0\Big). \tag{18}$$

N is the upper bound for clipping the decoding number of tokens.

We term this acceleration decoding pipeline as FDM-A. The algorithm of it is summarized in Algorithm 2 in Appendix B.

5 Experiment

5.1 Main Results

Benchmarks and Metrics: We evaluate FDM and FDM-A on four benchmarks: GSM8K (Cobbe et al., 2021), HumanEval (Chen et al., 2021), Countdown (Zhao et al., 2025), and ARC (Clark et al., 2018), covering math problem solving, code generation, logical reasoning, and commonsense knowledge. Following Hong et al. (2025), we add a system prompt before each input, with dataset-specific details summarized in Appendix C. All evaluations are conducted in the zero-shot setting. We use accuracy (ACC) to measure task performance and Tokens Per Second (TPS) to evaluate decoding efficiency.

Baselines and models: We compare FDM with three heuristic-based decoding methods: decoding with the highest probability (Probability) (Nie et al.,

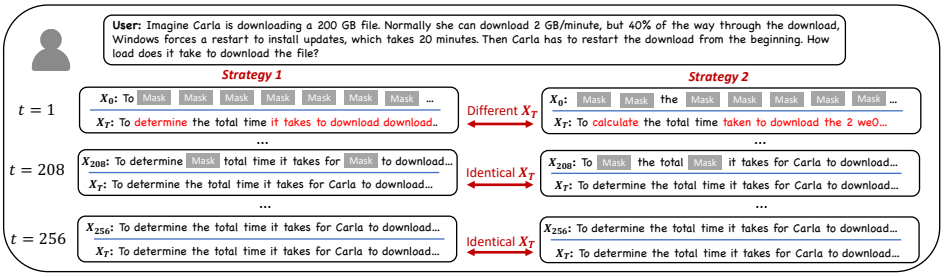


Figure 3: The effect of different decoding strategy to $\mathbf{x}_T$ at the step t given the identical $\mathbf{x}_{t-1}$. The influence gradually decreases as t increases from 0 to T.

2025; Zheng et al., 2024), with the highest marginal probability (Margin) (Kim et al., 2025) and with the lowest entropy (Entropy) (Ben-Hamu et al., 2025). For FDM-A, since it tries to balance between efficiency and performance, we also compare it with the latest dynamic decoding methods for LDMs: EB (Entropy Bounded Sampler) (Ben-Hamu et al., 2025) and WINO (Hong et al., 2025). To ensure the generalability of the observations, we cover four variants of LDMs: LLaDA-8B-Instruct (Nie et al., 2025), LLaDA-1.5 (Zhu et al., 2025a), LLaDA-MoE-7B-Instruct (Zhu et al., 2025b) and MMaDA-8B-MixCoT (Yang et al., 2025).

Hyperparameters: Consistent with the observation in Nie et al. (2025), we adopt the semi-autoregressive pipeline for instruction-tuned LDMs, using a generation length of 256 and a block size of 64. The number of decoding steps T is set to 256 when comparing heuristic decoding methods with FDM, and to 128 when comparing acceleration methods with FDM-A. For the baselines, the threshold in EB is set to 0.5, while τ_1 and τ_2 in WINO are set to 0.7 and 0.9, respectively. For our methods, the dynamic pruning threshold γ is set to 0.6 for FDM and 0.5 for FDM-A, since FDM-A performs fewer exploration steps. For stage division in FDM-A, we set $\eta_1 = 0.8$ and $\eta_2 = 0.7$. We vary the width K of FDM from 2 to 4 and set the default width of FDM-A to 2. All experiments are conducted on a single NVIDIA A100 80G GPU.

Results: Firstly, Table 2 shows that FDM consistently surpasses baseline methods across all models and benchmarks, demonstrating the importance of exploration with global confidence. For example, on LLaDA and ARC, the best heuristic method achieves 82.55%, while FDM with width $K = 2$ reaches 86.00%. Moreover, increasing K further improves performance, confirming the inference-time scaling ability of FDM; on LLaDA-MoE and GSM8K, accuracy increases from 77.48% to 78.32% as K grows from 2 to 4. Notably, these gains require neither a verifier nor parameter optimization, making FDM suitable for scenarios with sufficient computation and high accuracy requirements. Appendix D provides an example that FDM decodes correctly while all heuristic methods fail.

Secondly, Table 3 shows that FDM-A achieves a strong accuracy–speed trade-off, with only marginal performance loss compared with FDM. On LLaDA and GSM8K, FDM-A reaches 81.96% accuracy, only 0.07% lower than FDM ($K = 2$), while improving TPS from 8.28 to 42.65, yielding a 5.15× speedup. In contrast, heuristic methods degrade notably when the decoding steps are reduced; for example, probability decoding drops from 81.20% to 78.17% when T is halved. By effectively exploiting global confidence in the exploration and balance stages, FDM-A also outperforms dynamic decoding methods such as WINO.

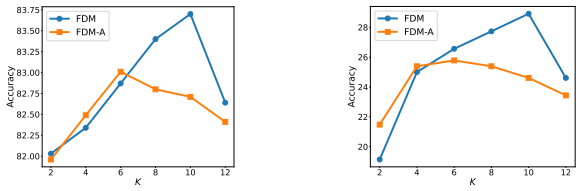
5.2 Ablation Studies

In this section, we investigate the influence of hyperparameters on the performance of FDM and FDM-A. In addition, in Appendix E, we explore the generalization of hyperparameters across generation lengths. We find that when the generation length is extended to 512, the scalability of FDM maintains, and FDM-A achieves better accuracy–efficiency trade-off than baseline methods.

Going Wider with K. In Section 5.1, we show that larger K will benefit the performance of FDM and a natural question is whether it will bring consistent improvement with the increase of K. Thus, here we set K with the value from $[2, 4, 6, 8, 10, 12]$ and perform experiments on the LLaDA model. As summarized in both Figure 4 and Figure 8 in Appendix F, the accuracy will reach the peak for both FDM-A and FDM. This is because although the training goal of the network is to fit the data distribution p_{data}, in realistic, the divergence between two distributions is not zero, leading to accumulating bias when we perform

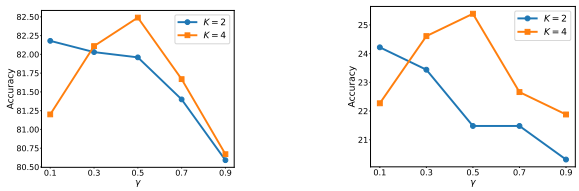
Table 2: Comparison (%) of FDM with the heuristic decoding strategies across four benchmarks. With the scale up of the wide K, we see that accuracy increases with the decrease of TPS, demonstrating that FDM serves as an inference-time scaling method.

Benchmark	Method	LLaDA		LLaDA-1.5		MMaDA-MixCoT		LLaDA-MoE	
		ACC	TPS	ACC	TPS	ACC	TPS	ACC	TPS
GSM8K	Probability (T=256)	81.20	11.51	82.10	11.89	56.18	11.26	76.50	4.13
	Margin (T=256)	80.14	11.23	82.18	11.39	56.18	10.97	76.80	4.04
	Entropy (T=256)	80.12	10.79	81.93	10.94	54.91	10.51	76.80	3.84
	FDM (K=2)	82.03	8.28	82.49	8.29	57.54	8.18	77.48	3.80
	FDM (K=3)	82.18	5.86	82.64	5.84	57.68	5.71	77.63	3.78
	FDM (K=4)	82.34	4.61	83.02	4.69	57.92	4.61	78.32	3.60
HumanEval	Probability (T=256)	42.68	9.24	42.68	9.20	12.80	8.91	56.71	3.81
	Margin (T=256)	43.29	8.76	42.68	8.83	12.80	8.65	58.54	3.81
	Entropy (T=256)	40.24	8.67	42.68	8.67	11.59	8.28	59.15	3.56
	FDM (K=2)	43.29	6.55	43.29	6.66	12.80	6.32	59.76	3.72
	FDM (K=3)	44.51	4.63	43.29	4.51	12.80	4.60	59.76	3.67
	FDM (K=4)	45.73	3.66	44.51	3.71	13.40	3.64	60.37	3.47
Countdown	Probability (T=256)	18.75	11.19	16.02	11.20	4.69	11.15	40.62	4.15
	Margin (T=256)	19.14	10.89	20.31	10.92	14.61	10.84	40.23	4.03
	Entropy (T=256)	18.44	10.39	20.31	10.39	7.03	10.31	38.67	3.95
	FDM (K=2)	19.14	8.21	20.70	8.62	14.45	8.13	40.62	3.73
	FDM (K=3)	21.09	6.11	21.88	6.48	16.40	6.10	46.10	3.71
	FDM (K=4)	25.00	5.28	23.43	5.25	17.58	4.97	46.88	3.70
ARC	Probability (T=256)	80.96	10.96	87.06	10.98	58.26	10.94	76.19	4.01
	Margin (T=256)	82.55	10.85	87.44	10.82	56.37	10.62	75.85	4.01
	Entropy (T=256)	78.13	10.43	87.38	10.38	58.33	10.18	71.84	3.98
	FDM (K=2)	86.00	7.72	88.18	7.71	59.43	7.68	82.89	3.85
	FDM (K=3)	86.46	5.58	88.45	5.59	59.61	5.58	83.45	3.73
	FDM (K=4)	86.68	4.58	88.59	4.56	59.76	4.37	83.51	3.64



(a) GSM8K (b) Countdown

Figure 4: The influence of K on model performance on GSM8K and Countdown benchmarks.



(a) GSM8K (b) Countdown

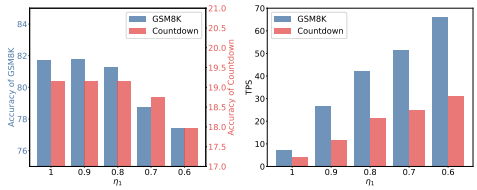
Figure 5: The influence of γ on model performance on GSM8K and Countdown benchmarks.

excessive searching operations, supported by the theoretical analysis in Appendix H. Intriguingly, when we compare the curves of FDM and FDM-A, we find that FDM outperforms FDM-A when K is larger. Conversely, FDM-A holds an advantage when K is small. This highlights the complementary role of both methods. FDM-A is suited for use when computational resources are limited. FDM, on the other hand, provides robust capabilities under conditions of abundant computational supply. We also explore scaling FDM by expanding the number of foreseeing depth in Appendix G. The results reveal that it is another scaling dimension except the width K.

The Setting of γ. When we foresee the future impact with the global confidence, we introduce a threshold γ to dynamically prune the candidate tokens of the low local confidence value. We further investigate its effect by tuning it between 0.1 and 0.9 with $K = 2$ and 4. Taking FDM-A as an example, the results on LLaDA of the GSM8K and Countdown datasets are shown in Figure 5. For more results, please refer to Figure 9 in Appendix F. We also observe a trade-off in its configuration: the results illustrate that a smaller γ, *e.g.* 0.1, achieves better accuracy in $K = 2$. This is because it provides a wider choice for selection, mitigating the issue of inadequate search. However, we find that it can not maintain its advantage when K increases to 4, since more tokens of lower C_{local} are chosen, reflecting the uncertainty of models in its prediction. In contrast, if we set γ too large, it will suppress the exploration in the foreseeing decoding. In total, γ near 0.5 can balance both in its application, which

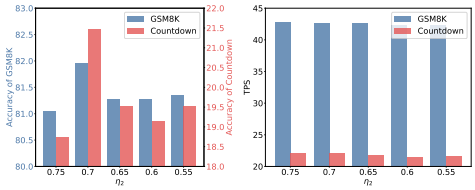
Table 3: Comparison (%) of FDM-A with methods that accelerate the decoding of LDMs. The best results are in **bold**. FDM-A achieves not only the highest accuracy, but also the fastest speed.

Benchmark	Method	LLaDA		LLaDA-1.5		MMaDA-MixCoT		LLaDA-MoE	
		ACC	TPS	ACC	TPS	ACC	TPS	ACC	TPS
GSM8K	Probability (T=128)	78.17	20.86	80.06	22.04	52.08	21.65	71.49	8.81
	Margin (T=128)	79.23	21.36	80.39	21.61	50.72	21.37	72.93	8.47
	Entropy (T=128)	77.94	20.69	80.22	20.71	49.13	20.51	70.66	8.39
	EB	79.61	36.46	81.35	37.74	53.83	36.39	74.53	13.98
	WINO	79.45	40.02	80.89	40.96	51.33	36.92	72.78	13.02
	FDM-A (Ours)	**81.96**	**42.65**	**82.87**	**43.98**	**55.12**	**40.96**	**76.72**	**14.89**
HumanEval	Probability (T=128)	35.37	16.75	39.63	17.04	7.93	16.59	51.22	7.99
	Margin (T=128)	37.20	16.25	37.20	16.58	12.80	16.15	49.39	7.74
	Entropy (T=128)	31.20	15.93	35.37	15.77	4.27	15.46	48.78	7.46
	EB	37.20	17.34	37.80	18.99	12.80	24.95	59.15	10.14
	WINO	39.02	18.33	40.24	20.59	12.20	29.90	56.71	9.31
	FDM-A (Ours)	**44.51**	**21.56**	**42.07**	**23.83**	**13.41**	**32.32**	**60.98**	**11.25**
Countdown	Probability (T=128)	19.53	21.19	**21.09**	22.01	12.01	21.97	44.53	8.80
	Margin (T=128)	20.31	20.65	19.92	21.76	8.59	21.57	42.58	8.33
	Entropy (T=128)	19.53	20.62	18.75	20.93	8.59	20.52	35.94	8.15
	EB	19.20	18.68	19.92	18.79	18.75	48.51	42.19	9.48
	WINO	19.14	20.52	19.14	20.38	20.70	52.39	25.00	8.23
	FDM-A (Ours)	**21.48**	**21.98**	**21.09**	**22.29**	**26.95**	**62.30**	**49.61**	**10.14**
ARC	Probability (T=128)	82.16	21.24	86.38	21.71	55.04	22.26	72.74	8.68
	Margin (T=128)	84.34	20.65	85.91	21.34	56.30	21.47	75.48	8.56
	Entropy (T=128)	77.20	19.91	84.58	20.48	55.61	20.58	69.08	8.12
	EB	73.55	32.01	85.68	34.89	56.86	32.89	71.50	8.98
	WINO	79.44	34.17	85.31	35.27	56.79	34.27	70.86	7.92
	FDM-A (Ours)	**86.30**	**38.20**	**87.66**	**38.43**	**59.50**	**37.07**	**83.33**	**12.62**



(a) Accuracy (b) TPS

Figure 6: The influence of η_1 on Accuracy and TPS on GSM8K and Countdown benchmarks.



(a) Accuracy (b) TPS

Figure 7: The influence of η_2 on Accuracy and TPS on GSM8K and Countdown benchmarks.

is chosen as the configuration for the experiments in Section 5.1.

The Choices of η_1 and η_2. In addition to K and γ, FDM-A also introduces η_1 and η_2 as hyperparameters for stage divisions. In the main text, we perform experiments on the GSM8K and Countdown benchmarks in Figure 6 and 7. For more results on HumanEval and ARC benchmarks, please refer to Figure 10 and 11 in Appendix F. We firstly fix η_2 at 0.6 and linearly decrease η_1 from 1.0 to 0.6. The results in Figure 6 (a) demonstrate that the accuracy remains stable at first, and then drops sharply in small η_1. In contrast, the TPS monotonically improves with the decrease of η_1 because more tokens are parallely decoded. By default, we set η_1 as 0.8 because it can not only maintain the model utility but also achieve high decoding speed. Secondly, we fix η_1 with 0.8 and set η_2 from [0.75, 0.7, 0.65, 0.6, 0.55]. Among all configurations, $\eta_2 = 0.7$ consistently achieves outstanding performances across all benchmarks. Because when η_2 is large, the exploration at the balance stage is insufficient. But if η_2 is set too small, uncertain tokens will interfere the correctness of decoding.

6 Conclusion

In this paper, we address token decoding orders in Language Diffusion Models (LDMs). We propose the Foreseeing Decoding Method (FDM), which combines local and global confidence to optimize decoding order, and theoretically prove that it reduces divergence from the natural distribution compared with heuristic decoding. We further introduce FDM-A, which applies deep exploration only at critical steps for better efficiency. Experiments show that FDM consistently outperforms strong baselines, while FDM-A achieves a superior speed–performance trade-off, providing a principled step toward more effective LDM decoding.

Limitations

Our paper addresses an important decoding problem of LDMs: the decoding order of masked tokens. However, there are still several limitations. First, both FDM and FDM-A improve the performance of LDMs from the perspective of decoding without modifying model parameters. Therefore, their performance gains are inherently bounded by the capability of the underlying model. Second, due to the gap between the fitting distribution learned by the model and the natural data distribution, excessively large values of γ may introduce noisy candidates into the search space. As a result, the accuracy of FDM and FDM-A can decrease when γ is set too large.

References

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

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. 2025. Accelerated sampling from masked diffusion models via entropy bounded unmasking. *arXiv preprint arXiv:2505.24857*.

Lukas Berglund, Meg Tong, Maximilian Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2024. The reversal curse: Llms trained on "a is b" fail to learn "b is a". In *ICLR*.

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

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

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

Google DeepMind. 2025. Gemini diffusion.

Xuanzhao Dong, Wenhui Zhu, Xiwen Chen, Zhipeng Wang, Peijie Qiu, Shao Tang, Xin Li, and Yalin Wang. 2025. Llada-medv: Exploring large language diffusion models for biomedical image understanding. *arXiv preprint arXiv:2508.01617*.

Guhao Feng, Yihan Geng, Jian Guan, Wei Wu, Liwei Wang, and Di He. 2025. Theoretical benefit and limitation of diffusion language model. *arXiv preprint arXiv:2502.09622*.

Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, and 1 others. 2024. Scaling diffusion language models via adaptation from autoregressive models. *arXiv preprint arXiv:2410.17891*.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. 2023. Diffuseq: Sequence to sequence text generation with diffusion models. In *ICLR*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In *NeurIPS*.

Feng Hong, Geng Yu, Yushi Ye, Haicheng Huang, Huangjie Zheng, Ya Zhang, Yanfeng Wang, and Jiangchao Yao. 2025. Wide-in, narrow-out: Revokable decoding for efficient and effective dllms. *arXiv preprint arXiv:2507.18578*.

Zhanqiu Hu, Jian Meng, Yash Akhauri, Mohamed S Abdelfattah, Jae-sun Seo, Zhiru Zhang, and Udit Gupta. 2025. Accelerating diffusion language model inference via efficient kv caching and guided diffusion. *arXiv preprint arXiv:2505.21467*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, Stefano Ermon, and 1 others. 2025. Mercury: Ultra-fast language models based on diffusion. *arXiv preprint arXiv:2506.17298*.

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

Gen Li and Changxiao Cai. 2025. A convergence theory for diffusion language models: An information-theoretic perspective. *arXiv preprint arXiv:2505.21400*.

Tianyi Li, Mingda Chen, Bowei Guo, and Zhiqiang Shen. 2025. A survey on diffusion language models. *arXiv preprint arXiv:2508.10875*.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. In *NeurIPS*.

Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025. dkv-cache: The cache for diffusion language models. *arXiv preprint arXiv:2505.15781*.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. *arXiv preprint arXiv:2502.09992*.

William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *CVPR*.

Fred Zhangzhi Peng, Zachary Bezemek, Sawan Patel, Jarrid Rector-Brooks, Sherwood Yao, Avishek Joey Bose, Alexander Tong, and Pranam Chatterjee. 2025. Path planning for masked diffusion model sampling. *arXiv preprint arXiv:2502.03540*.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *CVPR*.

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

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. 2025. Fast-dllm: Training-free acceleration of diffusion llm by enabling kv cache and parallel decoding. *arXiv preprint arXiv:2505.22618*.

Zhihui Xie, Jiacheng Ye, Lin Zheng, Jiahui Gao, Jingwei Dong, Zirui Wu, Xueliang Zhao, Shansan Gong, Xin Jiang, Zhenguo Li, and 1 others. 2025. Dreamcoder 7b: An open diffusion language model for code. *arXiv preprint arXiv:2509.01142*.

Zhen Xiong, Yujun Cai, Zhecheng Li, and Yiwei Wang. 2025. Unveiling the potential of diffusion large language model in controllable generation. *arXiv preprint arXiv:2507.04504*.

Ling Yang, Ye Tian, Bowen Li, Xinchen Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. 2025. Mmada: Multimodal large diffusion language models. *arXiv preprint arXiv:2505.15809*.

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

Zebin You, Shen Nie, Xiaolu Zhang, Jun Hu, Jun Zhou, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. 2025. Llada-v: Large language diffusion models with visual instruction tuning. *arXiv preprint arXiv:2505.16933*.

Siyan Zhao, Devaansh Gupta, Qinqing Zheng, and Aditya Grover. 2025. d1: Scaling reasoning in diffusion large language models via reinforcement learning. *arXiv preprint arXiv:2504.12216*.

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

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, and 1 others. 2025a. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*.

Fengqi Zhu, Zebin You, Yipeng Xing, Zenan Huang, Lin Liu, Yihong Zhuang, Guoshan Lu, Kangyu Wang, Xudong Wang, Lanning Wei, and 1 others. 2025b. Llada-moe: A sparse moe diffusion language model. *arXiv preprint arXiv:2509.24389*.

A Proof of Theorem 1

Define the single-step KL errors

$$\begin{aligned}\varepsilon_t^H &\triangleq D_{KL}(p_{data}(\mathbf{x}_t|\mathbf{x}_{t-1}), \pi_H(\mathbf{x}_t)),\\ \varepsilon_t^F &\triangleq D_{KL}(p_{data}(\mathbf{x}_t|\mathbf{x}_{t-1}), \pi_F(\mathbf{x}_t)).\end{aligned} \tag{19}$$

We first prove that

$$\varepsilon_t^F = \varepsilon_t^H - \mathcal{I}_{p_{data}}(\mathbf{x}_t; \mathbf{x}_T|\mathbf{x}_{t-1}), \tag{20}$$

where $\mathcal{I}_{p_{data}}(\mathbf{x}_t; \mathbf{x}_T|\mathbf{x}_{t-1})$ is the conditional mutual information under q.

Since $\mathbf{x}_{t-1}$ and $\mathbf{q}$ can be considered as fixed variables at the step t, we omit them in the notation for brevity. According to the definition of KL divergence, we have:

$$\begin{aligned}\varepsilon_t^F &\triangleq D_{KL}\big(p_{data}(\mathbf{x}_t), \pi_F(\mathbf{x}_t)\big)\\ &= \sum_{\mathbf{x}_t} p_{data}(\mathbf{x}_t)\Big[\log p_{data}(\mathbf{x}_t) - \log \pi_F(\mathbf{x}_t)\Big].\end{aligned} \tag{21}$$

By construction

$$\begin{aligned}\pi_F(\mathbf{x}_t) &= \frac{\exp(S(\mathbf{x}_t))}{Z_t},\\ S(\mathbf{x}_t) &= C_{\text{local}}(\mathbf{x}_t) + C_{\text{global}}(\mathbf{x}_t),\\ Z_t &= \sum_{\mathbf{x}_t'} \exp(S(\mathbf{x}_t')).\end{aligned} \tag{22}$$

Hence

$$\begin{aligned}\log \pi_F(\mathbf{x}_t) &= S(\mathbf{x}_t) - \log Z_t\\ &= C_{\text{local}}(\mathbf{x}_t) + C_{\text{global}}(\mathbf{x}_t) - \log Z_t.\end{aligned} \tag{23}$$

By inserting Equation 23 into 21 and split the sum,

$$\begin{aligned}\varepsilon_t^F &= \sum_{\mathbf{x}_t} p_{data}(\mathbf{x}_t)\Big[\log p_{data}(\mathbf{x}_t) - C_{\text{local}}(\mathbf{x}_t)\\ &\quad - C_{\text{global}}(\mathbf{x}_t) + \log Z_t\Big] \qquad (24)\\ &= \underbrace{\sum_{\mathbf{x}_t} p_{data}(\mathbf{x}_t)\Big[\log p_{data}(\mathbf{x}_t) - C_{\text{local}}(\mathbf{x}_t)\Big]}_{\text{Term A}} \qquad (25)\\ &\quad - \underbrace{\sum_{\mathbf{x}_t} p_{data}(\mathbf{x}_t)\Big[C_{\text{global}}(\mathbf{x}_t) - \log Z_t\Big]}_{\text{Term B}}. \qquad (26)\end{aligned}$$

By definition $C_{\text{local}}(\mathbf{x}_t) = \log p_\theta(\mathbf{x}_t)$, so

$$\begin{aligned}\text{Term A} &= \sum_{\mathbf{x}_t} p_{data}(\mathbf{x}_t)\Big[\log p_{data}(\mathbf{x}_t)\\ &\quad - \log p_\theta(\mathbf{x}_t)\Big] \qquad (27)\\ &= D_{KL}\big(p_{data}(\mathbf{x}_t), p_\theta(\mathbf{x}_t)\big) \triangleq \varepsilon_t^H.\end{aligned}$$

First recall

$$\begin{aligned}C_{\text{global}}(\mathbf{x}_t) &= \mathbb{E}_{x'\sim p_\theta} \log p_\theta(x' \mid \mathbf{x}_t)\\ &= \sum_{\mathbf{x}_T} p_\theta(\mathbf{x}_T \mid \mathbf{x}_t) \cdot \log p_\theta(\mathbf{x}_T \mid \mathbf{x}_t).\end{aligned} \tag{28}$$

Therefore

$$\begin{aligned}&\text{Term B} \qquad (29)\\ &= \sum_{\mathbf{x}_t} p_{data}(\mathbf{x}_t)\Big[C_{\text{global}}(\mathbf{x}_t) - \log Z_t\Big]\\ &= \sum_{\mathbf{x}_t} p_{data}(\mathbf{x}_t) \cdot \Big[\sum_{\mathbf{x}_T} p_\theta(\mathbf{x}_T \mid \mathbf{x}_t) \qquad (30)\\ &\quad \cdot \log p_\theta(\mathbf{x}_T \mid \mathbf{x}_t) - \log Z_t\Big]. \qquad (31)\end{aligned}$$

Next we use the identity $Z_t = p_\theta(\mathbf{x}_T)$ (marginal over $\mathbf{x}_t'$), which follows from

$$\begin{aligned}Z_t &= \sum_{\mathbf{x}_t'} \exp(S(\mathbf{x}_t'))\\ &= \sum_{\mathbf{x}_t'} p_\theta(\mathbf{x}_t', \mathbf{x}_T) = p_\theta(\mathbf{x}_T).\end{aligned} \tag{32}$$

Hence

$$\begin{aligned}\log Z_t &= \log p_\theta(\mathbf{x}_T)\\ &= \sum_{\mathbf{x}_T} p_\theta(\mathbf{x}_T \mid \mathbf{x}_t) \log p_\theta(\mathbf{x}_T),\end{aligned} \tag{33}$$

where the second equality holds because $p_\theta(\mathbf{x}_T)$ does not depend on $\mathbf{x}_t$ and $\sum_{\mathbf{x}_T} p_\theta(\mathbf{x}_T \mid \mathbf{x}_t) = 1$. Inserting this into Equation 29 gives

$$\begin{aligned}\text{Term B} &= \sum_{\mathbf{x}_t} p_{data}(\mathbf{x}_t) \sum_{\mathbf{x}_T} p_\theta(\mathbf{x}_T \mid \mathbf{x}_t)\\ &\quad \cdot \Big[\log p_\theta(\mathbf{x}_T \mid \mathbf{x}_t) - \log p_\theta(\mathbf{x}_T)\Big] \qquad (34)\\ &= \sum_{\mathbf{x}_t, \mathbf{x}_T} p_{data}(\mathbf{x}_t, \mathbf{x}_T)\\ &\quad \cdot \Big[\log \frac{p_\theta(\mathbf{x}_T \mid \mathbf{x}_t)}{p_\theta(\mathbf{x}_T)}\Big] \qquad (35)\\ &= \sum_{\mathbf{x}_t, \mathbf{x}_T} p_{data}(\mathbf{x}_t, \mathbf{x}_T)\\ &\quad \cdot \Big[\log \frac{p_{data}(\mathbf{x}_t, \mathbf{x}_T)}{p_{data}(\mathbf{x}_t) p_{data}(\mathbf{x}_T)}\Big] \qquad (36)\\ &\quad \big(\text{replace } p_\theta \text{ with } p_{data}\big)\\ &= \mathcal{I}_{p_{data}}(\mathbf{x}_t; \mathbf{x}_T). \qquad (37)\end{aligned}$$

Inserting Equation 27 and 34 into Equation 24 yields

$$\varepsilon_t^F = \varepsilon_t^H - \mathcal{I}_{p_{data}}(\mathbf{x}_t; \mathbf{x}_T), \tag{38}$$

which completes the proof of Equation 20. Using the chain rule for KL divergence over sequences we have

$$
\begin{aligned}
D_{KL}\big(p_{data}(\mathbf{x}), p_{\pi_F}\big) &= \sum_{t=1}^{T} \mathbb{E}_{p_{data}(\mathbf{x}_{t-1})}\big[\varepsilon_t^F\big] \\
&= \sum_{t=1}^{T} \mathbb{E}_{p_{data}(\mathbf{x}_{t-1})}\big[\varepsilon_t^H \\
&\quad - \mathcal{I}_{p_{data}}(\mathbf{x}_t; \mathbf{x}_T | \mathbf{x}_{t-1})\big].
\end{aligned} \tag{39}
$$

Recognising the definitions

$$
\begin{aligned}
&\sum_{t=1}^{T} \mathbb{E}_{p_{data}(\mathbf{x}_{t-1})} \varepsilon_t^H = D_{KL}\big(p_{data}(\mathbf{x}), p_{\pi_H}\big), \\
&\sum_{t=1}^{T} \mathbb{E}_{p_{data}(\mathbf{x}_{t-1})} \mathcal{I}_{p_{data}}(\mathbf{x}_t; \mathbf{x}_T | \mathbf{x}_{t-1}) = \Delta_{\text{total}},
\end{aligned} \tag{40}
$$

we obtain

$$
D_{KL}\big(p_{data}(\mathbf{x}), p_{\pi_F}\big) = D_{KL}\big(p_{data}(\mathbf{x}), p_{\pi_H}\big) - \Delta_{\text{total}}, \tag{41}
$$

which finishes the proof of Theorem 1. □

B Algorithms

Algorithm 1 Foreseeing Decoding Method (FDM)

Input: User query $\mathbf{q}$, the partially decoded sequence $\mathbf{x}_{t-1}$, pruning threshold γ, width K, well-trained models θ, the number of tokens for decoding n.
// Get the candidate tokens of the undecoded positions.
Cand $= [\mathbf{1}[\mathbf{x}_{t-1} = \texttt{Mask}] \odot \arg\max p_\theta(\mathbf{q}, \mathbf{x}_{t-1})]$
for x_t in Candidate **do**
  if $p_\theta(\mathbf{q}, \mathbf{x}_{t-1})[x_t] \leq \gamma$ **then**
    Delete x_t from Candidate.
  end if
end for
Priority Queue $\{\mathbf{x}_t\}$ of the priority C_{local} by foreseeing decoding n tokens from Candidate.
// Narrow the search space with the width K.
$\Lambda =$ Top-K$(\{\mathbf{x}_t\})$
if $\Lambda = \varnothing$ **then**
  $\mathbf{x}_t = \arg\max C_{local}(\mathbf{x}_t)$
else
  Calculate $C_{global}(\mathbf{x}_t)$ for each $\mathbf{x}_t$ in Λ
  $\mathbf{x}_t = \arg\max_{\mathbf{x}_t \in \Lambda} \{C_{local}(\mathbf{x}_t) + C_{global}(\mathbf{x}_t)\}$
end if
Output: The decoded answer $\mathbf{x}_t$ at step t

Algorithm 2 Acceleration with FDM (FDM-A)

Input: User query $\mathbf{q}$, fully masked sequence $\mathbf{x}_0$, pruning threshold γ_1, searching width K_1, stage division coefficients η_1 and η_2, upper bound for the decoding number N, well-trained models θ.
Initialize FDM with $K = K_1$ and get $\texttt{FDM}_1$.
Initialize FDM with $K = 1$ and get $\texttt{FDM}_2$.
$t = 1$
while Mask not in $\mathbf{x}_t$ **do**
  if $\texttt{NUM}\big([\mathbf{x}_{t-1} = \texttt{Mask}] p_\theta(\mathbf{q}, \mathbf{x}_{t-1}) > \eta_1\big) = 0$ **then**
    $\mathbf{x}_t = \texttt{FDM}_1(\mathbf{x}_{t-1}, \mathbf{q}, n = 1, \gamma = \gamma_1)$
  else if $\texttt{NUM}\big([\mathbf{x}_{t-1} = \texttt{Mask}] p_\theta(\mathbf{q}, \mathbf{x}_{t-1}) > \eta_1\big) \geq N$ **then**
    $\mathbf{x}_t = \texttt{FDM}_2(\mathbf{x}_{t-1}, \mathbf{q}, n = N, \gamma = 1.0)$
  else if $\texttt{NUM}\big(\eta_2 < [\mathbf{x}_{t-1} = \texttt{Mask}] p_\theta(\mathbf{q}, \mathbf{x}_{t-1}) \leq \eta_1\big) = 0$ **then**
    $\mathbf{x}_t = \texttt{FDM}_2(\mathbf{x}_{t-1}, \mathbf{q}, n = \texttt{NUM}([\mathbf{x}_{t-1} = \texttt{Mask}] p_\theta(\mathbf{q}, \mathbf{x}_{t-1}) > \eta_1), \gamma = 1.0)$
  else
    $\mathbf{x}_t = \texttt{FDM}_1(\mathbf{x}_{t-1}, \mathbf{q}, n = \texttt{NUM}([\mathbf{x}_{t-1} = \texttt{Mask}] p_\theta(\mathbf{q}, \mathbf{x}_{t-1}) > \eta_1), \gamma = \eta_2)$
  end if
  $t = t + 1$
end while
Output: Generated answer $\mathbf{x}_t$.

C System Prompts for Evaluation

GSM8K

You are a math expert. You will be given a question to solve. Solve it step by step. Wrap the final answer in a \\boxed{}.
Respond in the following format:
<reasoning >
Your reasoning here
</reasoning >
<answer>
\\boxed{...}
</answer>

ARC

You will be given a multiple choice question and a list of options. You need to first reason step by step, and then select the correct option (A, B, C, D). Wrap the single letter of the correct option in \\boxed{...}. Respond in the following format:
<reasoning>
Your reasoning here
</reasoning>
<answer>
\\boxed{...}
</answer>

Countdown

Using only the provided numbers, create an arithmetic expression that evaluates to exactly the provided target number. You may use the operations +, -, *, and / as needed, but each number must be used exactly once. Think step-by-step. After reasoning, provide only your final expression inside \\boxed{ }
tags without including an equals sign or the target number. For example: \\boxed{a + b * c}
Respond in the following format:
<reasoning>
Your reasoning here
</reasoning>
<answer>
\\boxed{...}
</answer>

HumanEval

Write a solution to the following problem and make sure that it passes the tests:
python

D The Example Case

Question in the ARC benchmark

Which of the following is the primary way that evergreen trees planted on the north and west sides of residences in Massachusetts help reduce winter heating costs?
A They insulate against cold air
B They optimize solar heating.
C They reduce radiant cooling.
D They act as a windbreak.

Decoding with the highest probability (Incorrect)

Evergreen trees planted on the north and west sides of residences in Massachusetts help reduce winter heating costs primarily by acting as a physical barrier against cold cold air. This trees' asymmetrical and dense foliage provide a a natural insulation, preventing cold air from entering the home, thereby reducing the need for heating heating.
</reasoning>
<answer>
\boxed{A}
</answer>

Decoding with the highest margin (Incorrect)

Evergreen trees planted on the north and west sides of residences in Massachusetts can help reduce winter heating costs by providing as much solar heat as possible. This is because the trees can capture and store sunlight, which can be used to heat the home during the winter months. By maximizing the amount of solar heat, the trees can help reduce the need for additional heating systems.
</reasoning>

<answer>
B

Decoding with the lowest entropy (Incorrect)

Evergreen trees are planted on the north and west sides of residences in Massachusetts to help reduce winter heating costs. The primary reason for this benefit is due to the insulation provided by the trees. The trees provide insulation against the cold, reducing the amount of heat escaping the house and keeping the interior temperature slightly warmer. This insulation reduces the need for heating, thereby making the home more efficient.
<answer>
\boxed{A}

Decoding with FDM (Correct)

Evergreen trees planted on the north and west sides of residences in Massachusetts can help reduce winter heating costs by acting as a windbreak. These trees can block cold air from entering the home, reducing the need for heating. Additionally, evergreen trees can also provide insulation against cold air, but the primary benefit is their role as a windbreak. </reasoning>
<answer>
\boxed{D}
</answer>

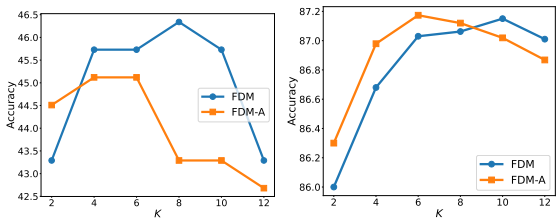
E Additional Evaluation with Longer Generation Length

We further evaluate the robustness of FDM and FDM-A under a longer generation length of 512. The results in Table 4 show that the hyperparameter configurations remain effective when the decoding horizon is extended.

Table 4: Evaluation results on LLaDA with generation length of 512 across benchmarks.

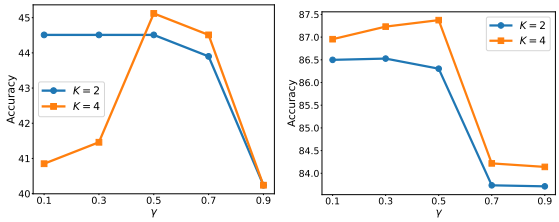
Method	GSM8K		HumanEval		Countdown		ARC	
	ACC	TPS	ACC	TPS	ACC	TPS	ACC	TPS
Probability ($T = 512$)	83.55	7.77	40.85	6.80	25.39	7.94	83.93	7.81
Margin ($T = 512$)	82.87	7.58	41.46	6.61	25.39	7.74	84.84	7.60
Entropy ($T = 512$)	83.40	7.24	37.80	6.29	24.61	7.38	74.02	7.28
FDM ($K = 2$)	83.93	5.33	41.46	4.53	25.78	5.50	86.11	5.28
FDM ($K = 3$)	84.31	3.81	42.07	3.04	26.95	3.99	87.09	3.80
FDM ($K = 4$)	84.84	2.95	42.07	2.57	26.95	3.23	87.24	2.88
Probability ($T = 256$)	79.83	15.30	34.76	13.45	20.70	15.69	83.82	15.52
Margin ($T = 256$)	80.44	14.92	38.41	13.07	24.61	15.12	84.16	14.99
Entropy ($T = 256$)	78.62	14.33	32.32	12.32	18.75	14.58	76.55	14.34
EB	82.03	40.26	38.41	20.07	22.66	18.81	74.78	30.56
WINO	82.18	42.23	35.37	22.55	24.22	23.74	84.13	42.50
FDM-A	**83.78**	**45.74**	**40.24**	**24.38**	**27.73**	**25.01**	**85.17**	**44.35**

F Ablation Studies on the HumanEval and ARC Benchmarks



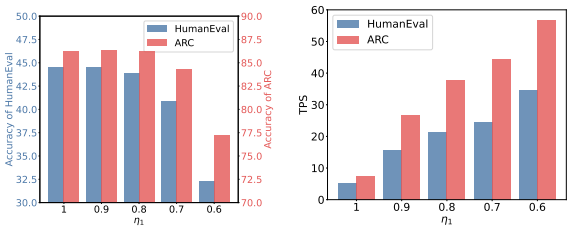
(a) HumanEval (b) ARC

Figure 8: The influence of K on model performance on HumanEval and ARC benchmarks.



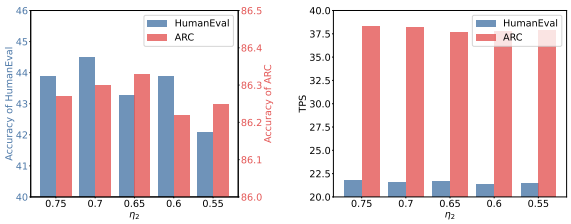
(a) HumanEval (b) ARC

Figure 9: The influence of γ on model performance on HumanEval and ARC benchmarks.



(a) Accuracy (b) TPS

Figure 10: The influence of η_1 on Accuracy and TPS on HumanEval and ARC benchmarks.



(a) Accuracy (b) TPS

Figure 11: The influence of η_2 on Accuracy and TPS on HumanEval and ARC benchmarks.

We perform experiments on the HumanEval and ARC benchmarks with the experimental settings in Section 5.2. We observe that the optimal hyperparameter has good transferability across datasets.

G Going Deeper with FDM

Table 5: Accuracy on the Countdown benchmark with different foreseeing depths M and widths K.

Accuracy	$K=1$	$K=2$	$K=3$	$K=4$
$M=1$	18.75	19.14	21.09	25.00
$M=2$	18.75	19.53	23.83	25.39
$M=3$	18.75	19.92	25.39	30.08

Recall that in Equation 6, the decoding objective for $\mathbf{x}_t$ is defined as:

$$\mathbf{x}_t = \arg\max_{\mathbf{x}_t} \; p_\theta(\mathbf{x}_{t+1:T}|\mathbf{q}, \mathbf{x}_t) p_\theta(\mathbf{x}_t|\mathbf{q}, \mathbf{x}_{t-1}). \tag{42}$$

We can further expand the conditional probability $p_\theta(\mathbf{x}_{t+1:T}|\mathbf{q}, \mathbf{x}_t)$. By introducing the hyperparameter M, FDM can be extended to a foreseeing depth larger than 1:

$$\begin{aligned}\mathbf{x}_t = &\arg\max_{\mathbf{x}_t} \; p_\theta(\mathbf{x}_{t+M:T}|\mathbf{q}, \mathbf{x}_{t+M-1}) \\ &\cdot \prod_{\alpha=t}^{t+M-1} p_\theta(\mathbf{x}_\alpha|\mathbf{q}, \mathbf{x}_{\alpha-1})\end{aligned} \tag{43}$$

By jointly varying the depth M and width K, we conduct experiments on the Countdown benchmark using LLaDA. The results in Table 5 show that increasing the foreseeing depth provides another dimension for inference-time scaling. However, this extension also increases the computational cost to $O(K^M)$, making the latency impractical for most real-world scenarios.

H Theoretical Analysis for Performance Degradation with a Extremely Large K

We next present a theoretical analysis to support that the degradation in performance is due to the consequence of noise-amplified selection in the mismatch between the model and natural distributions. At each decoding step, we have K candidate tokens whose true future returns are

$$\begin{aligned}s_1, \ldots, s_K \in \mathbb{R}, \quad s_* = \max_i s_i, \\ s_i \triangleq \log p_{\text{data}}(\mathbf{x}_t = \text{i-th candidate} \\ \text{token} \mid \mathbf{x}_{t-1}, \mathbf{q}).\end{aligned} \tag{44}$$

When K is large enough, we can assume that the token with the highest possibility in the real data distribution is already in the candidates. The model in our algorithm only observes confidence scores, which are noisy estimates of true future returns:

$$\hat{s}_i = s_i + \xi_i \tag{45}$$

Here we hypothesize that ξ_i is independent and identically distributed (i.i.d.) Gaussian random variables:

$$\xi_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma^2) \tag{46}$$

and $\hat{s}_i$ is equal to the sum of the local and global confidence of the i-th candidate token:

$$\begin{aligned}\hat{s}_i = &C_{local}(\text{i-th candidate token}) \\ &+ C_{global}(\text{i-th candidate token})\end{aligned} \tag{47}$$

According to Equation 8, our algorithm selects the candidate with the highest $\hat{s}_i$:

$$j = \underset{i \in [K]}{argmax}\, \hat{s}_i. \tag{48}$$

For any sub-optimal candidate i with confidence gap $\Delta_i = s_* - s_i > 0$, the probability that noise flips the ranking is

$$\Pr\big(\hat{s}_i > \hat{s}_*\big) = \Pr\big(\xi_i - \xi_* > \Delta_i\big) = \Phi\Big(-\frac{\Delta_i}{\sqrt{2}\sigma}\Big), \tag{49}$$

where $\Phi(\cdot)$ is the standard normal tail CDF(Cumulative Distribution Function). Union-bounding over the $K-1$ sub-optimal candidates yields

$$\Pr(\text{select non-optimal}) \leq \sum_{i \neq *} \Phi\Big(-\frac{\Delta_i}{\sqrt{2}\sigma}\Big). \tag{50}$$

The RHS(right hand side) increases monotonically with K whenever there exist $\Delta_i = O(\sigma)$. On the other hand, let $\xi_{(K)} = \max_{i \leq K} \xi_i$. The Extreme-value Theory gives

$$\mathbb{E}[\xi_{(K)}] \approx \sigma\sqrt{2 \ln K}. \tag{51}$$

Hence, the selected estimate satisfies

$$\mathbb{E}[\hat{s}_j] \approx s_* + \sigma\sqrt{2 \ln K}, \tag{52}$$

while the true return of the selected candidate satisfies:

$$\mathbb{E}[s_j] \leq s_* + \underbrace{\mathbb{E}[\xi_{(K)} - \xi_j]}_{\geq 0}. \tag{53}$$

Consequently, the expected regret is:

$$\mathbb{E}[\Delta_j] = \mathbb{E}[s_* - s_j] \geq C\sigma\sqrt{\ln K} \tag{54}$$

which grows with K, providing a quantitative lower bound for the "winner's curse" in the decoding process. Because each incorrect selection changes the subsequent context, the regret accumulates along the generation process. Early mistakes (where K is too large in our schedule) therefore have a negative effect, explaining the empirical observation in Figure 4 and 8.