WHAT MAKES A GOOD DECODING PATH? SELECTING PARALLEL-DECODABLE TRAJECTORIES FOR DIFFUSION LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion large language models (DLLMs) allow flexible decoding trajectories, but their factorized parallel predictions can produce jointly incompatible tokens even when individual predictions are confident. Existing trajectory-selection methods emphasize marginal predictability or immediate dependency risk, leaving less explored how decoding decisions reshape the dependency structure encountered by subsequent updates. In this paper, we derive an information-theoretic identity linking the current and expected residual factorization costs to the information that revealed tokens provide about remaining positions, and propose Factorization-Aware Candidate Transition Selection (FACTS), a training-free sampler that ranks candidate transitions by combining future uncertainty reduction with a joint-compatibility signal obtained by comparing sequential and factorized scores of the same proposed tokens. Experiments on Dream and LLaDA across mathematical reasoning and code generation show that FACTS is particularly effective under wider parallel decoding.

1 INTRODUCTION

Diffusion large language models (DLLMs) have emerged as an alternative to autoregressive (AR) language models by replacing fixed left-to-right generation with iterative masked denoising, enabling bidirectional context, flexible token filling orders, and parallel token prediction (Austin et al., 2021a; He et al., 2023; Sahoo et al., 2024; Nie et al., 2026; Ye et al., 2025; Bie et al., 2025). The flexible generation also introduces a new problem in deciding which tokens to reveal at each step and therefore which trajectory to follow (Kim et al., 2025; Xu et al., 2026; Ni et al., 2026). Finding good trajectories has consequently become important for both training and inference, where favorable denoising trajectories can be sampled directly to improve inference (Shen et al., 2026; Yang et al., 2026; Lee et al., 2026; Liu et al., 2026) and also can provide better supervision for distillation or few-step generation (Peng et al., 2026; Zhang et al., 2026b; Li et al., 2026; Qian et al., 2026).

Early samplers mainly evaluate a decoding decision from the current state. They prioritize masked positions with high confidence (Nie et al., 2026; Wu et al., 2026), low entropy (Ye et al., 2025), or related token-wise scores so that the tokens committed at each step are likely to be correct (Ben-Hamu et al., 2026; Kim et al., 2026). However, each reveal can change the context for all remaining positions, and a locally reliable choice may therefore lead to an unfavorable continuation (Lee et al., 2026; Yang et al., 2026). To address this phenomenon of nearsightedness, some works optimize downstream utility instead. They evaluate candidate actions by the uncertainty reduction they induce in future masked positions (Yang et al., 2026; Fu et al., 2026; Xu et al., 2025), or explicitly look ahead over several candidate trajectories and prefer paths with lower future uncertainty (Lee et al., 2026; Huang et al., 2026). These approaches substantially improve trajectory planning, but future states are still evaluated primarily through token-wise or marginal predictability. This becomes insufficient when multiple tokens are predicted together because tokens that each appear highly confident on their own may still form an unreliable joint prediction under the factorized parallel prediction used by DLLMs (Kang et al., 2026; Ringel et al., 2026; Yao, 2026).

Recent analyses of parallel decoding reveal that the true conditional distribution over a group of masked tokens can contain substantial statistical dependence, whereas a DLLM predicts these to-

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.

kens in parallel through a factorized reverse distribution, and the discrepancy between the joint distribution and its factorized approximation creates an irreducible factorization error (Lavenant & Zanella, 2025; Zhang et al., 2026a; Kang et al., 2026). These results make clear that trajectory quality cannot be understood from marginal uncertainty alone, because a trajectory that quickly reduces marginal uncertainty may still repeatedly leave strongly coupled tokens for the same parallel updates and therefore accumulate large factorization errors. Recent methods have begun to account for this issue by explicitly modeling inter-token dependencies during decoding. They test contextual independence through intervention-induced changes in token distributions (Azangulov et al., 2026), extract dependency signals from attention maps to guide token ordering or construct conflict graphs (Luo et al., 2026; Zhou et al., 2026), learn pairwise conditional influences from hidden states to constrain the cumulative dependency of simultaneously updated tokens (Ringel et al., 2026), and correct parallel-rollout estimates of future Path Log-Likelihood using an entropy-derived upper bound on cumulative total correlation (Liu et al., 2026). Despite these different implementations, existing approaches primarily use dependency to identify unreliable parallel updates or correct their estimated cost. This makes dependency largely a source of local decoding risk to be mitigated, while overlooking how decoding decisions can reshape the dependency structure of subsequent updates. We argue that dependency should therefore be considered beyond its immediate factorization cost, as its evolution can fundamentally affect the quality of the entire decoding trajectory.

In this work, we study trajectory selection under factorized parallel decoding. Our key observation is that each decoding decision not only incurs an immediate factorization cost, but also reshapes the dependency structure faced by subsequent updates. Trajectory quality should therefore be determined by how factorization difficulty evolves throughout decoding rather than by the local risk of individual parallel updates alone. Building on classical information-theoretic analyses and recent dependency-aware decoding methods (Watanabe, 1960; Lavenant & Zanella, 2025; Liu et al., 2025), we use total correlation to quantify this difficulty and develop a training-free sampler that jointly considers the factorization cost of the current update and the factorization difficulty left for future decoding. Our contributions are summarized as follows:

- Conceptually, we identify the evolution of factorization difficulty as a missing consideration in DLLM trajectory selection. Existing criteria largely characterize the quality or risk of individual updates, but do not capture how these decisions reshape the dependency structure encountered by subsequent parallel decoding.
- Methodologically, we establish an information-theoretic view of trajectory selection based on total correlation and develop a training-free sampler under this principle. Our formulation evaluates decoding decisions by jointly considering their assignment-level positive interaction evidence and the factorization difficulty they leave to future updates.
- Empirically, we show that explicitly accounting for the evolution of factorization difficulty yields better decoding trajectories. Across different DLLMs and decoding budgets, our method consistently improves the quality–parallelism trade-off.

2 PRELIMINARIES

2.1 DIFFUSION LARGE LANGUAGE MODELS

Let $\overline{\mathcal{V}}$ denote the model vocabulary, let $[\mathtt{M}] \in \overline{\mathcal{V}}$ be the mask token, and let $\mathcal{V} = \overline{\mathcal{V}} \setminus \{[\mathtt{M}]\}$ be the clean-token vocabulary. A clean sequence is denoted by $\mathbf{X} = (X_1, \ldots, X_L) \in \mathcal{V}^L$. DLLMs typically corrupt clean sequences through an absorbing-mask process and learn to recover masked tokens from bidirectional context (Austin et al., 2021a; Sahoo et al., 2024; Nie et al., 2026; Ye et al., 2025). At inference time, let $\mathbf{z}$ denote the current partially decoded token state and let $\mathcal{M}$ denote the eligible masked positions in the current active block. A candidate action $a = (\mathcal{S}, \mathbf{x}_{\mathcal{S}})$ selects positions $\mathcal{S} \subseteq \mathcal{M}$ and proposes one clean token at each selected position, producing the successor state $\mathbf{z}_a = \mathbf{z} \oplus a$. Different actions expose different contexts to subsequent predictions, so their sequence determines the decoding trajectory.

Although every masked position can attend to the full visible state, tokens committed within the same transition are predicted through separate position-wise distributions. The parallel prediction

over $\mathcal{S}$ therefore has the factorized form

$$q_\theta^{\mathrm{fac}}(\mathbf{x}_\mathcal{S} \mid \mathbf{z}) = \prod_{i \in \mathcal{S}} q_{\theta,i}(x_i \mid \mathbf{z}). \tag{1}$$

Thus, DLLMs are flexible in deciding which variables become visible across transitions but cannot directly represent statistical dependence among variables predicted together within one transition.

2.2 Factorization Gap of Parallel Prediction

Fix a state $\mathbf{z}$ and a set $\mathcal{S}$ of positions to be predicted in parallel, and let $p(\mathbf{x}_\mathcal{S} \mid \mathbf{z})$ denote the underlying conditional joint distribution. The expected negative log-likelihood of a predictor is its cross-entropy with p. The unrestricted optimum is attained by p itself, whereas under the factorization constraint each factor is independently minimized by the corresponding marginal p_i:

$$\begin{aligned} \mathcal{E}^\star_{\mathrm{joint}}(\mathcal{S} \mid \mathbf{z}) &= \min_q H(p(\mathbf{X}_\mathcal{S} \mid \mathbf{z}), q(\mathbf{X}_\mathcal{S} \mid \mathbf{z})) = H(\mathbf{X}_\mathcal{S} \mid \mathbf{z}), \\ \mathcal{E}^\star_{\mathrm{fac}}(\mathcal{S} \mid \mathbf{z}) &= \min_{\{q_i\}_{i \in \mathcal{S}}} H\left(p(\mathbf{X}_\mathcal{S} \mid \mathbf{z}), \prod_{i \in \mathcal{S}} q_i(X_i \mid \mathbf{z})\right) = \sum_{i \in \mathcal{S}} H(X_i \mid \mathbf{z}). \end{aligned} \tag{2}$$

Their difference is

$$\begin{aligned} \mathcal{E}^\star_{\mathrm{fac}}(\mathcal{S} \mid \mathbf{z}) - \mathcal{E}^\star_{\mathrm{joint}}(\mathcal{S} \mid \mathbf{z}) &= \sum_{i \in \mathcal{S}} H(X_i \mid \mathbf{z}) - H(\mathbf{X}_\mathcal{S} \mid \mathbf{z}) \\ &= \mathrm{KL}\left(p(\mathbf{X}_\mathcal{S} \mid \mathbf{z}) \,\middle\|\, \prod_{i \in \mathcal{S}} p_i(X_i \mid \mathbf{z})\right) \\ &=: \mathrm{TC}(\mathcal{S} \mid \mathbf{z}). \end{aligned} \tag{3}$$

The resulting conditional total correlation (TC) (Watanabe, 1960) is therefore exactly the irreducible excess cross-entropy induced by the factorization constraint.

A learned DLLM may additionally incur marginal modeling error. Inserting the marginals of p into its joint KL divergence separates these two sources:

$$\begin{aligned} &\mathrm{KL}\left(p(\mathbf{X}_\mathcal{S} \mid \mathbf{z}) \,\middle\|\, \prod_{i \in \mathcal{S}} q_{\theta,i}(X_i \mid \mathbf{z})\right) \\ &= \mathbb{E}_p\left[\log p(\mathbf{X}_\mathcal{S} \mid \mathbf{z}) - \sum_{i \in \mathcal{S}} \log p_i(X_i \mid \mathbf{z})\right] + \sum_{i \in \mathcal{S}} \mathbb{E}_{p_i}[\log p_i(X_i \mid \mathbf{z}) - \log q_{\theta,i}(X_i \mid \mathbf{z})] \\ &= \mathrm{TC}(\mathcal{S} \mid \mathbf{z}) + \sum_{i \in \mathcal{S}} \mathrm{KL}(p_i(X_i \mid \mathbf{z}) \,\|\, q_{\theta,i}(X_i \mid \mathbf{z}))\,. \end{aligned} \tag{4}$$

The second term is marginal modeling error, whereas the first remains even with perfect position-wise marginals whenever the reference conditional joint is not factorized (Lavenant & Zanella, 2025; Zhang et al., 2026a; Kang et al., 2026). Because $\mathrm{TC}(\mathcal{S} \mid \mathbf{z})$ depends on the visible state $\mathbf{z}$, each action changes the context and hence the factorization difficulty faced by later parallel predictions. This state dependence provides a mechanism through which candidate selection can change the structural difficulty faced by later parallel predictions.

3 Method

We propose Factorization-Aware Candidate Transition Selection (FACTS), a training-free sampler that uses the pretrained DLLM itself to select decoding transitions.

3.1 Factorization-Aware Transition Utility

At state $\mathbf{z}$, a candidate reveal set $\mathcal{S} \subseteq \mathcal{M}$ partitions the currently masked variables into those predicted now and the remaining set $\mathcal{R} = \mathcal{M} \setminus \mathcal{S}$. From Eq. 3, predicting $\mathcal{S}$ in the current parallel transition incurs the structural error $\mathrm{TC}(\mathcal{S} \mid \mathbf{z})$. Once their values become visible, the remaining variables are instead governed by the conditional joint $p(\mathbf{X}_\mathcal{R} \mid \mathbf{X}_\mathcal{S}, \mathbf{z})$, whose distance from its factorized marginals is $\mathrm{TC}(\mathcal{R} \mid \mathbf{X}_\mathcal{S}, \mathbf{z})$. The latter summarizes the factorization difficulty of the

post-transition state before a later reveal schedule is chosen. Applying the same structural error to the current and post-transition states and expanding both terms gives

$$\begin{aligned}
&\mathrm{TC}(\mathcal{S} \mid \mathbf{z}) + \mathrm{TC}(\mathcal{R} \mid \mathbf{X}_{\mathcal{S}}, \mathbf{z}) \\
&= \left[\sum_{i\in\mathcal{S}} H(X_i \mid \mathbf{z}) - H(\mathbf{X}_{\mathcal{S}} \mid \mathbf{z})\right] + \left[\sum_{j\in\mathcal{R}} H(X_j \mid \mathbf{X}_{\mathcal{S}}, \mathbf{z}) - H(\mathbf{X}_{\mathcal{R}} \mid \mathbf{X}_{\mathcal{S}}, \mathbf{z})\right] \\
&= \left[\sum_{u\in\mathcal{M}} H(X_u \mid \mathbf{z}) - H(\mathbf{X}_{\mathcal{M}} \mid \mathbf{z})\right] - \sum_{j\in\mathcal{R}} [H(X_j \mid \mathbf{z}) - H(X_j \mid \mathbf{X}_{\mathcal{S}}, \mathbf{z})] \\
&= \mathrm{TC}(\mathcal{M} \mid \mathbf{z}) - \sum_{j\in\mathcal{R}} I(X_j; \mathbf{X}_{\mathcal{S}} \mid \mathbf{z}).
\end{aligned} \tag{5}$$

$\mathrm{TC}(\mathcal{M} \mid \mathbf{z})$ is constant for all candidates considered from the same state, a reveal set therefore reduces this current-plus-residual structural burden exactly in expectation over $\mathbf{X}_{\mathcal{S}} \sim p(\cdot \mid \mathbf{z})$ through the entropy reductions $H(X_j \mid \mathbf{z}) - H(X_j \mid \mathbf{X}_{\mathcal{S}}, \mathbf{z})$, which are the conditional mutual informations $I(X_j; \mathbf{X}_{\mathcal{S}} \mid \mathbf{z})$. This directly motivates preferring reveals that provide more information about the variables that remain masked. More details can be seen in Appendix A.1.

Eq. 5 concerns the usefulness of revealing a set of variables, but an actual action $a = (\mathcal{S}, \mathbf{y}_{\mathcal{S}})$ must also choose their token values. This distinction matters because the DLLM proposes these values through the factorized distribution in Eq. 1. A collection of tokens can therefore be individually plausible while forming an unlikely joint assignment. Under the ideal distribution, the difference between the joint log probability of a candidate and its factorized reference is

$$c(a \mid \mathbf{z}) = \log p(\mathbf{y}_{\mathcal{S}} \mid \mathbf{z}) - \sum_{i\in\mathcal{S}} \log p_i(y_i \mid \mathbf{z}) = \log \frac{p(\mathbf{y}_{\mathcal{S}} \mid \mathbf{z})}{\prod_{i\in\mathcal{S}} p_i(y_i \mid \mathbf{z})}. \tag{6}$$

Positive $c(a \mid \mathbf{z})$ means that dependence among the selected variables provides positive interaction evidence for this concrete assignment relative to the product of its marginals. The two derivations therefore serve different purposes. The mutual-information terms in Eq. 5 determine whether the selected positions provide useful context for what remains, whereas c evaluates whether the concrete values proposed at those positions are jointly compatible.

3.2 Practical Scoring

The ideal quantities above depend on the unknown joint distribution, so FACTS replaces them with local scores computed from the pretrained DLLM. Let K denote the maximum number of tokens revealed in one decoding step. At state $\mathbf{z}$, the actual candidate size is $k = k(\mathbf{z}) := \min\{K, |\mathcal{M}|\}$. FACTS draws N proposal attempts, each containing exactly k position–token pairs. After exact deduplication, the candidate pool is $\mathcal{A}(\mathbf{z}) = \{a_1, \ldots, a_{N_c}\}$, where $N_c = |\mathcal{A}(\mathbf{z})| \le N$. For each candidate $a = \{(s_r, y_r)\}_{r=1}^{k}$, the pairs are indexed in increasing position order, and $\mathcal{R}(a) = \mathcal{M} \setminus \{s_1, \ldots, s_k\}$ denotes the eligible positions that remain masked. We use

$$\Delta H(a) = \frac{1}{|\mathcal{R}(a)|} \sum_{j\in\mathcal{R}(a)} [h_j(\mathbf{z}) - h_j(\mathbf{z}_a)] \tag{7}$$

as a model-based *Future Uncertainty Reduction* proxy, with $\Delta H(a) = 0$ when $\mathcal{R}(a)$ is empty. The before and after entropies are evaluated on exactly the same candidate-specific remaining positions. Under a consistent joint distribution, averaging the corresponding ideal entropy reduction over reveal outcomes recovers $|\mathcal{R}|^{-1} \sum_{j\in\mathcal{R}} I(X_j; \mathbf{X}_{\mathcal{S}} \mid \mathbf{z})$ from Eq. 5.

To approximate $c(a \mid \mathbf{z})$, FACTS keeps the proposed tokens fixed and asks how their scores change when they are conditioned on one another. Let $[k] = \{1, \ldots, k\}$ and let $\Pi_k \subseteq \mathfrak{S}_k$ denote the common non-empty set of reveal orders used for all candidates at the current transition. For an order $\pi = (\pi_1, \ldots, \pi_k) \in \Pi_k$, define the prefix action $a_{\pi_{<r}} := \{(s_{\pi_u}, y_{\pi_u}) : 1 \le u < r\}$. The factorized score evaluates every proposed token from the current state, whereas the sequential score conditions each token on the preceding prefix specified by π:

$$\ell_{\mathrm{fac}}(a) = \sum_{r=1}^{k} \log q_{\theta, s_r}(y_r \mid \mathbf{z}), \quad \ell_{\mathrm{seq}}^{\pi}(a) = \sum_{r=1}^{k} \log q_{\theta, s_{\pi_r}}\left(y_{\pi_r} \mid \mathbf{z} \oplus a_{\pi_{<r}}\right). \tag{8}$$

If all state-wise model predictions were conditionals of one consistent joint distribution, $\ell^{\pi}_{\mathrm{seq}}(a) - \ell_{\mathrm{fac}}(a)$ would equal the ideal correction in Eq. 6 for every order. Real DLLM predictions need not satisfy global joint consistency, so FACTS treats each reveal order as a sequential probe of within-action interaction and defines the *Positive Interaction Gain* which exploits dependence that provides additional predictive support

$$C(a) = \frac{1}{|\Pi_k|} \sum_{\pi \in \Pi_k} \left[\frac{\ell^{\pi}_{\mathrm{seq}}(a) - \ell_{\mathrm{fac}}(a)}{k} \right]_+ . \tag{9}$$

FACTS enumerates all $k!$ orders for small k and otherwise samples a reproducible subset of unique orders. Appendix A.3 discusses the consistency assumptions, positive-part operation, and order averaging in detail.

FACTS combines the future-state and assignment-level signals through

$$S_{\mathrm{FACTS}}(a) = \lambda_H \Delta H(a) + \lambda_C C(a), \quad a^\star = \arg\max_{a \in \mathcal{A}(\mathbf{z})} S_{\mathrm{FACTS}}(a). \tag{10}$$

Once $a^\star$ is selected, all of its position–token pairs are committed simultaneously through the standard DLLM parallel transition.

3.3 Efficient Sequential Evaluation

Computing $C(a)$ independently for every reveal order repeats many model evaluations. For a fixed candidate $a_n = \{(s_{n,r}, y_{n,r})\}_{r=1}^{k}$, any sequential prefix is completely determined by the subset $U \subseteq [k]$ of candidate pairs already revealed. We denote the corresponding state by $\mathbf{z}_{n,U} = \mathbf{z} \oplus \{(s_{n,r}, y_{n,r}) : r \in U\}$. Two reveal orders that reach the same U produce exactly the same model input under identical auxiliary conditioning, regardless of how that subset was reached. FACTS represents the local rollouts as paths through this subset lattice and evaluates each distinct candidate-local prefix state once, while routing its logits back to every logical order that requires them.

Without state reuse, N_c candidates and $|\Pi_k|$ reveal orders require $N_c|\Pi_k|(k-1)$ additional sequential model rows because the first token of each order is already scored by the current-state logits. Let D_k be the number of distinct non-empty proper prefix subsets appearing in Π_k. FACTS instead evaluates

$$R_{\mathrm{seq}} = N_c D_k, \quad D_k \le \min\{|\Pi_k|(k-1), 2^k - 2\}. \tag{11}$$

With all $k!$ permutations, every non-empty proper subset appears, so sequential evaluation decreases from $N_c\, k!(k-1)$ to $N_c(2^k - 2)$. For $k = 4$, this reduces the sequential work from 72 to 14 model rows per candidate. Appendix A.5 gives the exact equivalence proof, total row-count analysis, micro-batching strategy, and logit-reuse conditions.

4 Experiment

We evaluate FACTS against several trajectory-selection baselines across reasoning and code-generation tasks, while Appendix B provides the complete benchmark descriptions, evaluation prompts, baseline implementations, and decoding hyper-parameters.

4.1 Experimental Setup

Models, benchmarks, and evaluation. We evaluate our sampler on two representative instruction-tuned DLLMs, Dream-v0-Instruct-7B (Ye et al., 2025) and LLaDA-8B-Instruct (Nie et al., 2026). We consider four reasoning and code-generation benchmarks. GSM8K (Cobbe et al., 2021) contains 1,319 grade-school math problems and is evaluated by final-answer accuracy. MATH-500 (Hendrycks et al., 2021) contains 500 competition-level mathematics problems and is evaluated by symbolic equivalence of the extracted final answer. For code generation, we use the standard 164-problem HumanEval (Chen et al., 2021) and the 257-example `sanitized/test` split of MBPP (Austin et al., 2021b), and report execution-based pass@1 on both benchmarks. The maximum generation length is 256 tokens for GSM8K and 512 tokens for MATH-500, HumanEval, and MBPP. GSM8K, MATH-500, and MBPP are evaluated under both zero-shot and fixed few-shot prompting, while HumanEval follows its standard zero-shot setting.

Table 1: Main results on **Dream-v0-Instruct-7B**. We report final-answer accuracy (%) on GSM8K and MATH-500 and pass@1 (%) on HumanEval and MBPP under zero-shot and few-shot prompting. The best result in each column is shown in **bold**, and the second-best result is underlined. Avg. denotes the macro-average over four tasks for zero-shot and over the three supported tasks for few-shot; HumanEval does not use few-shot prompting.

Method	GSM8K		MATH-500		HumanEval		MBPP		Avg.	
	K=2	K=4	K=2	K=4	K=2	K=4	K=2	K=4	K=2	K=4
Zero-shot										
UNIFORM	31.1	12.2	12.6	1.4	6.7	2.4	18.7	3.5	17.3	4.9
CONFIDENCE	47.6	11.9	19.4	1.2	25.6	1.8	26.1	5.1	29.7	5.0
ENTROPY	43.8	8.0	15.8	0.0	21.3	1.2	22.6	5.8	25.9	3.8
MARGIN	46.9	13.3	21.4	0.8	20.1	0.6	28.0	7.0	29.1	5.4
KLASS	46.2	11.9	19.4	1.4	25.0	1.8	27.2	5.1	29.5	5.0
LOOKUM	63.7	52.1	42.4	27.4	42.1	21.9	51.4	32.3	49.9	33.4
INFO-GAIN	65.7	46.6	43.2	26.8	46.3	22.6	50.6	28.0	51.4	31.0
FACTS (Ours)	**65.9**	**57.8**	**45.4**	**31.6**	**47.6**	**34.8**	**59.1**	**44.0**	**54.5**	**42.0**
Few-shot										
UNIFORM	45.1	22.1	13.0	2.0	—	—	26.9	9.3	28.3	11.2
CONFIDENCE	64.1	35.6	21.6	2.2	—	—	34.2	12.8	40.0	16.9
ENTROPY	58.2	31.2	20.6	1.6	—	—	30.0	14.8	36.2	15.9
MARGIN	60.9	37.5	23.2	3.6	—	—	35.0	16.7	39.7	19.3
KLASS	64.1	35.6	20.4	2.2	—	—	35.8	12.4	40.1	16.8
LOOKUM	75.3	66.3	**45.6**	29.8	—	—	63.0	48.6	**61.3**	48.3
INFO-GAIN	**77.8**	64.7	43.2	26.0	—	—	61.1	38.9	60.7	43.2
FACTS (Ours)	76.2	**69.0**	43.4	**32.0**	—	—	**63.8**	**59.1**	61.1	**53.4**

Baselines. We compare against two groups of trajectory-selection methods. First, we include four current-state heuristics that select reveal positions using UNIFORM (Nie et al., 2026), CONFIDENCE (Chang et al., 2022), ENTROPY (Wang et al., 2026), or MARGIN (Kim et al., 2025) scores. These methods represent common local criteria that do not explicitly evaluate the effect of a reveal on later decoding states. Second, we compare with stronger dependency- or lookahead-aware samplers, including KLASS (Kim et al., 2026), LOOKUM (Lee et al., 2026), and INFO-GAIN (Yang et al., 2026). We evaluate all methods under the same decoding settings to ensure a fair comparison.

Decoding settings. Following Yang et al. (2026), our primary comparison fixes the block length to $B = 16$ for all methods. We evaluate $K = 2$ and $K = 4$ to test stronger parallel decoding, and use a token sampling temperature of 0.7. Candidate-based methods use $N = 8$ proposal attempts and a position temperature of 0.1. For a generation length of L tokens, decoding processes L/B blocks with B/K transitions per block, resulting in L/K decoding transitions in total. All methods are evaluated without KV caching.

4.2 MAIN RESULTS: FACTS SELECTS BETTER DECODING TRAJECTORIES

Tables 1 and 2 show that FACTS is most effective under wider parallel decoding. At $K = 4$, it achieves the best average performance on both Dream and LLaDA under both zero-shot and few-shot prompting. On Dream, the gains over the strongest baseline average are 8.6 percentage points in zero-shot evaluation and 5.1 points in few-shot evaluation, with improvements across both reasoning and code generation tasks. At $K = 2$, FACTS remains competitive in every setting.

The comparison highlights the value of considering both future uncertainty and parallel compatibility. LookUM and Info-Gain substantially outperform local token-scoring heuristics by evaluating how a reveal affects subsequent predictions. FACTS additionally accounts for the assignment-level positive interaction evidence and the factorization difficulty left for future updates, rather than relying on marginal predictability alone. Its stronger relative performance at $K = 4$ is consistent with this motivation: committing more tokens from the same state makes dependencies within parallel updates a more consequential consideration. These results support accounting for the evolution of factorization difficulty in trajectory selection, particularly at higher parallelism.

4.3 FACTORIZATION AWARENESS MATTERS UNDER PARALLEL DECODING

Table 2: Main results on **LLaDA-8B-Instruct**.

Method	GSM8K		MATH-500		HumanEval		MBPP		Avg.	
	K=2	K=4	K=2	K=4	K=2	K=4	K=2	K=4	K=2	K=4
Zero-shot										
UNIFORM	50.3	24.7	19.0	3.8	4.3	1.2	16.0	3.9	22.4	8.4
CONFIDENCE	63.4	33.1	26.6	5.0	3.0	2.4	27.2	6.2	30.1	11.7
ENTROPY	59.3	22.3	21.8	3.0	2.4	1.8	26.5	5.1	27.5	8.0
MARGIN	65.2	36.5	26.4	6.4	3.7	3.0	24.9	6.2	30.0	13.1
KLASS	63.2	33.1	27.2	4.8	3.7	1.8	26.9	5.8	30.2	11.4
LOOKUM	77.3	74.7	36.0	**35.0**	3.7	**7.9**	44.4	34.6	40.3	38.1
INFO-GAIN	**77.6**	73.2	36.0	32.2	6.7	3.0	**48.2**	29.6	**42.1**	34.5
FACTS (Ours)	76.0	**76.5**	**37.0**	32.6	**7.9**	7.3	45.1	**39.7**	41.5	**39.0**
Few-shot										
UNIFORM	53.5	24.3	18.6	5.8	—	—	13.2	4.7	28.5	11.6
CONFIDENCE	67.9	28.6	28.6	5.4	—	—	19.8	4.3	38.8	12.8
ENTROPY	62.3	22.4	22.0	4.0	—	—	12.1	4.3	32.1	10.2
MARGIN	67.9	35.8	31.2	6.6	—	—	18.3	4.3	39.1	15.6
KLASS	66.4	28.8	28.4	5.6	—	—	19.5	3.9	38.1	12.8
LOOKUM	78.6	77.3	40.4	37.6	—	—	42.0	26.5	53.7	47.1
INFO-GAIN	79.2	74.5	39.2	34.4	—	—	39.3	18.3	52.6	42.4
FACTS (Ours)	**80.1**	**79.2**	**41.6**	**37.8**	—	—	**45.1**	**31.1**	**55.6**	**49.4**

Combining future uncertainty reduction and interaction gain outperforms either component alone. Table 3 ablates FACTS into ΔH-only and C-only variants. Although C-only outperforms ΔH-only in all tested settings, adding ΔH yields further gains, reaching 6.61 percentage points over C-only on few-shot MBPP at $K = 4$. These results support the complementary roles of current assignment compatibility and future predictability, with larger benefits at the wider parallel setting.

Table 3: Component ablation: MATH-500 accuracy and MBPP pass@1 (%). Δ is the gain over the best single component (pp).

Method	MATH-500		MBPP	
	K=2	K=4	K=2	K=4
Zero-shot				
ΔH-only	42.40	27.40	55.64	36.19
C-only	43.40	29.40	58.75	42.80
FACTS	**45.40**	**31.60**	**59.14**	**43.97**
Δ (pp)	+2.00	+2.20	+0.39	+1.17
Few-shot				
ΔH-only	42.40	31.60	61.87	49.81
C-only	43.00	31.80	63.04	52.53
FACTS	**43.40**	**32.00**	**63.81**	**59.14**
Δ (pp)	+0.40	+0.20	+0.77	+6.61

FACTS gains a larger advantage as parallel decoding widens. Fig. 1 extends the comparison to $K = 8$ alongside $K = 2, 4$. On few-shot MBPP, its advantage over the strongest competing method grows from 0.77 to 10.50 and 12.06 percentage points at $K = 2$, 4, and 8, respectively. On few-shot MATH-500, FACTS trails the strongest baseline at $K = 2$ but leads by 2.20 and 3.40 percentage points at $K = 4$ and 8. The growing advantage at larger widths suggests that current compatibility and future predictability provide complementary guidance when more tokens must be committed together.

Prefix-state reuse accelerates FACTS without changing its decoding decisions. Table 4 evaluates the reuse strategy from Section 3.3 against naive sequential evaluation of the same FACTS candidates and reveal orders. With the default eight sampled orders, reuse halves the sequential model rows and yields a 1.91× generation speedup; the table also reports savings under full order enumeration. These results confirm that prefix-state reuse removes redundant sequential evaluations without changing the FACTS objective or its resulting trajectory.

4.4 CANDIDATE-RANKING DIAGNOSTICS FOR UNDERSTANDING FACTS

To isolate ranking quality, we let FACTS, Info-Gain, and LookUM rank a shared candidate pool with a target size of $N = 8$ at each frozen source state. We obtain these states with Dream using CONFIDENCE decoding ($B = 16$, $K = 4$) on few-shot MATH-500 and MBPP. We independently force each candidate action and complete its successor state with a $K = 1$ CONFIDENCE continuation, sharing the resulting downstream outcomes across scorers.

FACTS selects actions with higher downstream value. Fig. 2 measures this value by the correctness of the common continuation from each scorer's top-ranked action. Since the three scorers share

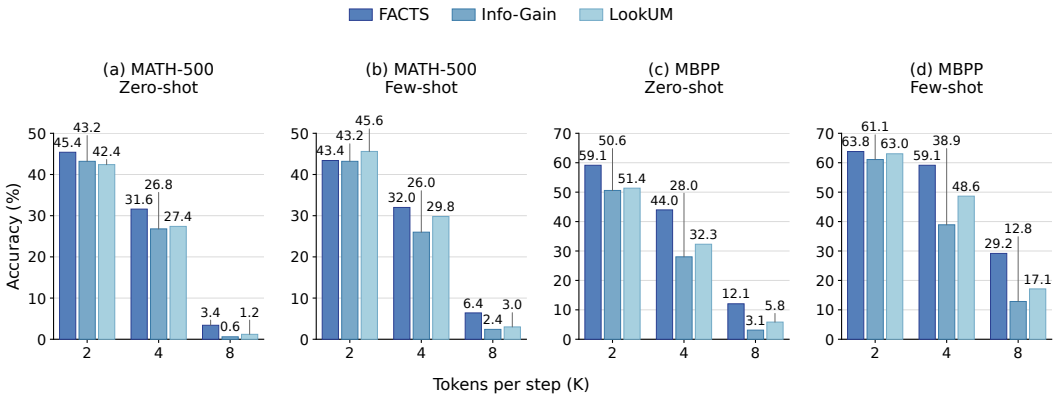

Figure 1: Effect of parallel decoding width on Dream-v0-Instruct-7B. We compare FACTS, Info-Gain, and LookUM at $K \in \{2, 4, 8\}$ on MATH-500 (accuracy) and MBPP (pass@1) under zero-shot and few-shot prompting.

Table 4: Efficiency of exact prefix-state reuse on Dream-v0-Instruct-7B. We use zero-shot decoding with $K = 4$, $B = 16$, and $N = 8$. Sequential rows denote the model evaluations required by the interaction term. Speedup is relative to FACTS without prefix-state reuse.

Benchmark	Orders	Sequential rows	Reduction	Speedup
MATH-500	8 sampled	36,360 → 18,180	50.0%	1.91×
MBPP	8 sampled	36,624 → 18,312	50.0%	1.91×
MATH-500	24 exact	109,080 → 21,210	80.6%	4.85×
MBPP	24 exact	109,944 → 21,378	80.6%	4.88×

the source state, candidate actions, and continuations, these differences isolate candidate ranking from the other parts of decoding. FACTS therefore more often gives highest priority to transitions from which the common continuation policy obtains a correct solution.

Positive Interaction Gain changes the selected action in 12.84% of source states. We compare the FACTS winner with the candidate maximizing ΔH to examine whether FACTS simply reproduces the ranking induced by Future Uncertainty Reduction ΔH. The selected action differs in 12.87% MATH-500 states and 12.76% MBPP states, respectively. The nearly identical rate across the two tasks indicates a systematic effect on candidate ranking rather than a task-specific exception. Together with Fig. 2, this shows that FACTS is not merely a rescaled information-gain criterion and that incorporating assignment compatibility can change which transition is preferred.

Eight sampled reveal orders closely reproduce full-enumeration top-1 choices at $K = 4$. Positive Interaction Gain depends on the reveal order used to sequentially evaluate the tokens within a $K = 4$ candidate. Since $K = 4$ admits $4! = 24$ possible reveal orders, we additionally evaluate all 24 orders as a higher-budget reference. As shown in Table 5, we compare the default eight-order estimate (q8) with all 24 reveal orders (q24) on the shared candidate pools. Top-1 agreement reaches 97.60% on MATH-500 and 97.91% on MBPP, indicating that eight sampled reveal orders are sufficient to approximate full-enumeration candidate selection in the evaluated $K = 4$ settings.

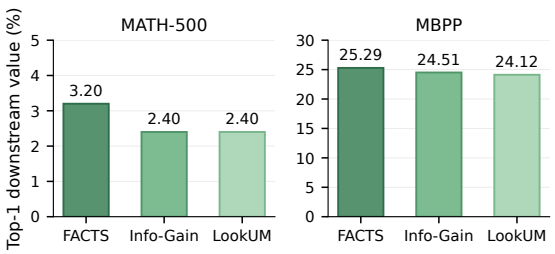

Figure 2: Downstream value of the top-ranked candidate action on Dream under few-shot prompting.

Table 5: Stability of FACTS with eight sampled reveal orders (q8) relative to all 24 orders (q24). Order-gap std. measures the variation of the sequential-versus-factorized interaction gap across the eight sampled reveal orders

Task	Top-1 agree.	Order-gap std.
MATH-500	97.60%	0.00936
MBPP	97.91%	0.11296

5 Related Work

5.1 Development of Diffusion Language Models

Diffusion-based text generation developed along continuous and discrete formulations. Continuous methods learned embeddings or representations. Diffusion-LM (Li et al., 2022) introduced embedding-space Gaussian diffusion and gradient-based control, DiffuSeq (Gong et al., 2023) adapted it to conditional generation, and SeqDiffuSeq (Yuan et al., 2024) and TESS (Karimi Mahabadi et al., 2024) improved the architecture, noise schedule, and output representation.

Discrete methods instead diffuse categorical token states. D3PM (Austin et al., 2021a) established categorical transition kernels, including the absorbing-mask process later adopted by DiffusionBERT (He et al., 2023) and MDLM (Lou et al., 2024), while SEDD (Sahoo et al., 2024) developed a score-based objective for discrete spaces. This masked formulation subsequently scaled to general-purpose DLLMs. LLaDA (Nie et al., 2026) combined from-scratch pretraining with instruction tuning, Dream (Ye et al., 2025) supported arbitrary-order generation and infilling, and LLaDA 2.0 (Bie et al., 2025) extended the paradigm to 100B parameters. Despite this progress, these models predict masked positions with factorized token-wise distributions and do not themselves specify which positions should be revealed together or in what order; decoding policy therefore remains an inference-time problem (Sahoo et al., 2024; Kim et al., 2025; Zhang et al., 2026a; Xu et al., 2026).

5.2 Trajectory Selection in Diffusion Language Models

Masked diffusion decoding must choose the reveal order, parallel update width, and, in search-based methods, token assignments (Kim et al., 2025; Shen et al., 2026). Early samplers made these decisions from the current state: LLaDA (Nie et al., 2026) and Dream (Ye et al., 2025) ranked positions by confidence, entropy, or margin; Fast-dLLM (Wu et al., 2026) and EB-Sampler (Ben-Hamu et al., 2026) adapted the update width through confidence or entropy criteria; and KLASS (Kim et al., 2026) additionally measured prediction stability across steps.

Later methods evaluated what a decision enables next. Information-Gain (Yang et al., 2026), ETE (Fu et al., 2026), and LoPA (Xu et al., 2025) score candidate successor states through the entropy or confidence of the remaining masked positions. Lee et al. (2026), Shen et al. (2026), and Huang et al. (2026) extend this idea to multi-step paths or joint order–token search; POKE-SMC instead optimizes future Path Log-Likelihood with sequential Monte Carlo and an entropy-based bound on cumulative total correlation (Liu et al., 2026). Trajectory selection has also entered training. Planner-aligned objectives (Peng et al., 2026) and learned scheduling policies (Xu et al., 2026) favor useful paths, while d3LLM (Qian et al., 2026), T3D (Zhang et al., 2026b), and IDLM (Li et al., 2026) distill trajectory supervision into few-step generators.

A complementary line asks which tokens can be predicted reliably in parallel. Recent analyses (Liu et al., 2025; Lavenant & Zanella, 2025; Zhang et al., 2026a) connect the irreducible error of product-form denoisers to conditional total correlation, while Kang et al. (2026) demonstrates that confident marginals can violate joint constraints. Dependency-aware samplers operationalize this insight locally. PUNT (Azangulov et al., 2026) tests contextual independence, DAWN (Luo et al., 2026) and DOS (Zhou et al., 2026) extract dependency signals from attention, and DEMASK (Ringel et al., 2026) learns pairwise conditional influence to avoid incompatible simultaneous updates.

6 Conclusion

In this paper, we argue that a decoding decision affects not only the reliability of the tokens committed at the current step, but also the factorization difficulty faced by subsequent updates. Consequently, trajectory selection should account for both the compatibility of the current parallel assignment and the dependency structure left for future decoding. Based on this view, we proposed FACTS, a training-free sampler that combines Future Uncertainty Reduction with Positive Interaction Gain to select more parallel-decodable trajectories. Experiments on Dream and LLaDA across mathematical reasoning and code generation consistently show the benefit of this criterion, with the strongest gains emerging as the parallel decoding width increases. These results suggest that dependency in DLLMs should be treated not only as a local constraint on simultaneous prediction, but as a state-dependent quantity that can be actively shaped through trajectory selection.

AI USE STATEMENT

We used generative AI tools, primarily OpenAI ChatGPT, to assist with refining and verifying detailed mathematical derivations, implementation, and feedback on experimental design. We also used these tools to support literature search and summarization and to edit portions of the manuscript for clarity and readability.

The research idea, theoretical claims, and high-level derivational framework were developed independently by the authors without generative AI assistance. All experimental designs, implementations, and empirical conclusions included in the final paper were reviewed and verified by the authors. Literature references suggested by generative AI were checked against the original sources, and all reported experimental results were obtained from our own implementations and experiments.

We take full responsibility for the final content of this work, including any text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 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.

Iskander Azangulov, Teodora Pandeva, Niranjani Prasad, Javier Zazo, and Sushrut Karmalkar. Parallel sampling from masked diffusion models via conditional independence testing. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=XjcHRIu0iF.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas S Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. *Advances in Neural Information Processing Systems*, 38:55981–56007, 2026.

Tiwei Bie, Maosong Cao, Kun Chen, Lun Du, Mingliang Gong, Zhuochen Gong, Yanmei Gu, Jiaqi Hu, Zenan Huang, Zhenzhong Lan, et al. Llada2. 0: Scaling up diffusion language models to 100b. *arXiv preprint arXiv:2512.15745*, 2025.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11305–11315. IEEE, 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.

Hengyu Fu, Baihe Huang, Virginia Adams, Charles Wang, Junkeun Yi, Mohammad Mahdi Kamani, Venkat Krishna Srinivasan, and Jiantao Jiao. From bits to rounds: Parallel decoding with exploration for diffusion language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=CTrA4eNQfv.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and Lingpeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=jQj-_rLVXsj.

Zhengfu He, Tianxiang Sun, Qiong Tang, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. DiffusionBERT: Improving generative masked language models with diffusion models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4521–4534, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.248. URL https://aclanthology.org/2023.acl-long.248/.

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 *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

Zheng Huang, Kiran Ramnath, Yueyan Chen, Aosong Feng, Sangmin Woo, Balasubramaniam Srinivasan, Zhichao Xu, Kang Zhou, Shuai Wang, Haibo Ding, and Lin Lee Cheong. Diffusion language model inference with Monte Carlo tree search. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 3493–3512, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-386-9. doi: 10.18653/v1/2026.findings-eacl.180. URL https://aclanthology.org/2026.findings-eacl.180/.

Wonjun Kang, Kevin Galim, Seunghyuk Oh, Minjae Lee, Yuchen Zeng, Shuibai Zhang, Coleman Richard Charles Hooper, Yuezhou Hu, Hyung Il Koo, Nam Ik Cho, and Kangwook Lee. Parallelbench: Understanding the trade-offs of parallel decoding in diffusion LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=OsZr5T7Cd0.

Rabeeh Karimi Mahabadi, Hamish Ivison, Jaesung Tae, James Henderson, Iz Beltagy, Matthew Peters, and Arman Cohan. TESS: Text-to-text self-conditioned simplex diffusion. In Yvette Graham and Matthew Purver (eds.), *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2347–2361, St. Julian's, Malta, March 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.eacl-long.144. URL https://aclanthology.org/2024.eacl-long.144/.

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 *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=DjJmre5IkP.

Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. Klass: Kl-guided fast inference in masked diffusion models. *Advances in Neural Information Processing Systems*, 38:92267–92301, 2026.

Hugo Lavenant and Giacomo Zanella. Error bounds and optimal schedules for masked diffusions with factorized approximations. *arXiv preprint arXiv:2510.25544*, 2025.

Sanghyun Lee, Seungryong Kim, Jongho Park, and Dongmin Park. Lookahead unmasking elicits reliable decoding in diffusion language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=OdM5JRVEiK.

David Li, Nikita Gushchin, Dmitry Abulkhanov, Eric Moulines, Ivan Oseledets, Maxim Panov, and Alexander Korotin. IDLM: Inverse-distilled diffusion language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=bS8qU2wJxu.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in neural information processing systems*, 35: 4328–4343, 2022.

Anji Liu, Oliver Broadrick, Mathias Niepert, and Guy Van den Broeck. Discrete copula diffusion. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 88953–88979, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/dd1fef536655685898a6602bfbf16857-Paper-Conference.pdf.

Xuejie Liu, Vit Chun Yap, Yitao Liang, and Anji Liu. Lookahead path likelihood optimization for diffusion LLMs. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=X8jXqDDaOT.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32819–32848. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/lou24a.html.

Lizhuo Luo, Zhuoran Shi, Jiajun Luo, Zhi Wang, Shen Ren, Wenya Wang, and Tianwei Zhang. Dawn: Dependency-aware fast inference for diffusion llms. *arXiv preprint arXiv:2602.06953*, 2026.

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 *Proceedings of the 43rd International Conference on Machine Learning (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. *Advances in Neural Information Processing Systems*, 38:50608–50646, 2026.

Fred Zhangzhi Peng, Zachary Bezemek, Jarrid Rector-Brooks, Shuibai Zhang, Michael M. Bronstein, Anru Zhang, Alexander Tong, and Joey Bose. Planner aware path learning in diffusion language models training. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=lAlI5FuIf7.

Yu-Yang Qian, Junda Su, Lanxiang Hu, Peiyuan Zhang, Zhijie Deng, Peng Zhao, and Hao Zhang. d3LLM: Ultra-fast diffusion LLM using pseudo-trajectory distillation. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=rzBAQT2Fkg.

Liran Ringel, Ameen Ali, and Yaniv Romano. Dependency-guided parallel decoding in discrete diffusion language models. *arXiv preprint arXiv:2604.02560*, 2026.

Subham S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Yangyi Shen, Tianjian Feng, Jiaqi Han, Wen Wang, Tianlang Chen, Chunhua Shen, Jure Leskovec, and Stefano Ermon. Improving diffusion language model decoding through joint search in generation order and token space. *arXiv preprint arXiv:2601.20339*, 2026.

Yinjie Wang, Ling Yang, Bowen Li, Ye Tian, Ke Shen, and Mengdi Wang. Revolutionizing reinforcement learning framework for diffusion large language models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=KNAyc9DMe3.

Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of research and development*, 4(1):66–82, 1960.

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 *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=3Z3Is6hnOT.

Chenkai Xu, Yijie Jin, Jiajun Li, Yi Tu, Guoping Long, Dandan Tu, Mingcong Song, Hongjie Si, Tianqi Hou, Junchi Yan, et al. Lopa: Scaling dllm inference via lookahead parallel decoding. *arXiv preprint arXiv:2512.16229*, 2025.

Jiawei Xu, Minghui Liu, Aakriti Agrawal, Yifan Chen, and Furong Huang. Scheduling thoughts: Learning the order of thought in diffusion language models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=j4UhHaxCpq.

Kaisen Yang, Jayden Teoh, Kaicheng Yang, Yitong Zhang, and Alex Lamb. Improving sampling for masked diffusion models via information gain. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=1HPTFpwktA.

Lin Yao. Don't commit alone: Joint token commitment in diffusion large language models, 2026. URL https://arxiv.org/abs/2607.04469.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7b: Diffusion large language models, 2025. URL https://arxiv.org/abs/2508.15487.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Fei Huang, and Songfang Huang. Text diffusion model with encoder-decoder transformers for sequence-to-sequence generation. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 22–39, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.2. URL https://aclanthology.org/2024.naacl-long.2/.

Shaorong Zhang, Longxuan Yu, Rob Brekelmans, Luhan Tang, Salman Asif, and Greg Ver Steeg. Generation order and parallel decoding in masked diffusion models: An information-theoretic perspective. *arXiv preprint arXiv:2602.00286*, 2026a.

Tunyu Zhang, Xinxi Zhang, Ligong Han, Haizhou Shi, Xiaoxiao He, Zhuowei Li, Hao Wang, Kai Xu, Akash Srivastava, Chengzhi Mao, et al. Few-step diffusion language models via trajectory self-distillation. *arXiv preprint arXiv:2602.12262*, 2026b.

Xueyu Zhou, Yangrong Hu, and Jian Huang. DOS: Dependency-oriented sampler for masked diffusion language models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17404–17419, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.861. URL https://aclanthology.org/2026.findings-acl.861/.

A Theoretical and Implementation Details of FACTS

A.1 Full Derivation of the Transition Factorization Identity

For disjoint $\mathcal{S}$ and $\mathcal{R}$ with $\mathcal{M} = \mathcal{S} \cup \mathcal{R}$, conditional total correlation after revealing $\mathbf{X}_\mathcal{S}$ is $\mathrm{TC}(\mathcal{R} \mid \mathbf{X}_\mathcal{S}, \mathbf{z}) = \sum_{j\in\mathcal{R}} H(X_j \mid \mathbf{X}_\mathcal{S}, \mathbf{z}) - H(\mathbf{X}_\mathcal{R} \mid \mathbf{X}_\mathcal{S}, \mathbf{z})$, where the conditional entropies already average over $\mathbf{X}_\mathcal{S} \sim p(\cdot \mid \mathbf{z})$. Starting from the two structural terms used in Sec. 3.1,

$$\begin{aligned}
&\mathrm{TC}(\mathcal{S} \mid \mathbf{z}) + \mathrm{TC}(\mathcal{R} \mid \mathbf{X}_\mathcal{S}, \mathbf{z}) \\
&= \sum_{i\in\mathcal{S}} H(X_i \mid \mathbf{z}) - H(\mathbf{X}_\mathcal{S} \mid \mathbf{z}) + \sum_{j\in\mathcal{R}} H(X_j \mid \mathbf{X}_\mathcal{S}, \mathbf{z}) - H(\mathbf{X}_\mathcal{R} \mid \mathbf{X}_\mathcal{S}, \mathbf{z}) \\
&= \sum_{i\in\mathcal{S}} H(X_i \mid \mathbf{z}) + \sum_{j\in\mathcal{R}} H(X_j \mid \mathbf{X}_\mathcal{S}, \mathbf{z}) - H(\mathbf{X}_\mathcal{M} \mid \mathbf{z}) \\
&= \sum_{u\in\mathcal{M}} H(X_u \mid \mathbf{z}) - H(\mathbf{X}_\mathcal{M} \mid \mathbf{z}) - \sum_{j\in\mathcal{R}} [H(X_j \mid \mathbf{z}) - H(X_j \mid \mathbf{X}_\mathcal{S}, \mathbf{z})] \\
&= \mathrm{TC}(\mathcal{M} \mid \mathbf{z}) - \sum_{j\in\mathcal{R}} I(X_j; \mathbf{X}_\mathcal{S} \mid \mathbf{z}).
\end{aligned} \tag{12}$$

Rearranging yields the equivalent decomposition $\mathrm{TC}(\mathcal{M} \mid \mathbf{z}) = \mathrm{TC}(\mathcal{S} \mid \mathbf{z}) + \mathrm{TC}(\mathcal{R} \mid \mathbf{X}_\mathcal{S}, \mathbf{z}) + \sum_{j\in\mathcal{R}} I(X_j; \mathbf{X}_\mathcal{S} \mid \mathbf{z})$. This form makes the connection to FACTS explicit: the action-dependent

reduction in current-plus-residual structural error is the sum of marginal information gains received by the remaining variables.

The same argument extends from two stages to a complete fixed decoding schedule. Let $\mathcal{P} = (\mathcal{S}_1, \ldots, \mathcal{S}_T)$ be an ordered partition of $\mathcal{M}$ and let $\mathcal{P}_{<t} = \bigcup_{r<t} \mathcal{S}_r$ denote the positions revealed before transition t. The expected cumulative factorization cost along this schedule is

$$\begin{aligned} \mathcal{F}(\mathcal{P} \mid \mathbf{z}) &:= \sum_{t=1}^{T} \mathrm{TC}(\mathcal{S}_t \mid \mathbf{X}_{\mathcal{P}_{<t}}, \mathbf{z}) \\ &= \sum_{t=1}^{T} \sum_{i \in \mathcal{S}_t} H(X_i \mid \mathbf{X}_{\mathcal{P}_{<t}}, \mathbf{z}) - H(\mathbf{X}_{\mathcal{M}} \mid \mathbf{z}) \\ &= \mathrm{TC}(\mathcal{M} \mid \mathbf{z}) - \sum_{t=1}^{T} \sum_{i \in \mathcal{S}_t} I(X_i; \mathbf{X}_{\mathcal{P}_{<t}} \mid \mathbf{z}). \end{aligned} \tag{13}$$

The second equality follows because the blockwise joint entropies telescope to $H(\mathbf{X}_{\mathcal{M}} \mid \mathbf{z})$ by the probability chain rule. When $\mathcal{S}_1 = \mathcal{S}$, the information sum in Eq. 13 contains the direct contribution $\sum_{j \in \mathcal{R}} I(X_j; \mathbf{X}_{\mathcal{S}} \mid \mathbf{z})$ from the first reveal, together with additional information contributed by later blocks under their chosen schedule. Thus, Eq. 12 is the $T = 2$ special case, while the one-step FACTS score greedily approximates the first-reveal contribution.

A different but also valid identity is $\mathrm{TC}(\mathcal{M} \mid \mathbf{z}) = \mathrm{TC}(\mathcal{S} \mid \mathbf{z}) + \mathrm{TC}(\mathcal{R} \mid \mathbf{z}) + I(\mathbf{X}_{\mathcal{S}}; \mathbf{X}_{\mathcal{R}} \mid \mathbf{z})$. The group mutual information $I(\mathbf{X}_{\mathcal{S}}; \mathbf{X}_{\mathcal{R}} \mid \mathbf{z})$ is generally not equal to $\sum_{j \in \mathcal{R}} I(X_j; \mathbf{X}_{\mathcal{S}} \mid \mathbf{z})$. FACTS uses the latter because its practical future score measures the reduction of each remaining token's marginal entropy. The conditional-TC decomposition in Eq. 12 is therefore the theoretical identity that matches the implemented scoring rule.

A.2 Connection between Future Uncertainty Reduction and Mutual Information

For a fixed reveal set $\mathcal{S}$ and a remaining position $j \in \mathcal{R}$, the conditional mutual information can be written as

$$\begin{aligned} I(X_j; \mathbf{X}_{\mathcal{S}} \mid \mathbf{z}) &= H(X_j \mid \mathbf{z}) - H(X_j \mid \mathbf{X}_{\mathcal{S}}, \mathbf{z}) \\ &= H(X_j \mid \mathbf{z}) - \mathbb{E}_{\mathbf{Y}_{\mathcal{S}} \sim p}[H(X_j \mid \mathbf{z}, \mathbf{X}_{\mathcal{S}} = \mathbf{Y}_{\mathcal{S}})] \\ &= \mathbb{E}_{\mathbf{Y}_{\mathcal{S}} \sim p}[H(X_j \mid \mathbf{z}) - H(X_j \mid \mathbf{z}, \mathbf{X}_{\mathcal{S}} = \mathbf{Y}_{\mathcal{S}})]\,. \end{aligned} \tag{14}$$

Summing over $\mathcal{R}$ shows that the expected realized reduction of the remaining marginal entropies equals $\sum_{j \in \mathcal{R}} I(X_j; \mathbf{X}_{\mathcal{S}} \mid \mathbf{z})$. FACTS replaces the unknown ideal conditional distributions with the DLLM distributions before and after a concrete candidate assignment and normalizes by $|\mathcal{R}(a)|$. This yields Eq. 7.

A.3 Properties of Joint Compatibility

For a fixed reveal set $\mathcal{S}$, the ideal compatibility correction from Eq. 6 satisfies

$$\begin{aligned} \mathbb{E}_{\mathbf{Y}_{\mathcal{S}} \sim p}[c((\mathcal{S}, \mathbf{Y}_{\mathcal{S}}) \mid \mathbf{z})] &= \mathbb{E}_p\left[\log \frac{p(\mathbf{Y}_{\mathcal{S}} \mid \mathbf{z})}{\prod_{i \in \mathcal{S}} p_i(Y_i \mid \mathbf{z})}\right] \\ &= \mathrm{KL}\left(p(\mathbf{X}_{\mathcal{S}} \mid \mathbf{z}) \,\middle\|\, \prod_{i \in \mathcal{S}} p_i(X_i \mid \mathbf{z})\right) \\ &= \mathrm{TC}(\mathcal{S} \mid \mathbf{z}). \end{aligned} \tag{15}$$

The expectation is non-negative, but individual values of c can be negative. FACTS uses compatibility at the assignment level to rerank concrete proposals; it does not use Eq. 15 as an objective for choosing sets with large TC.

Suppose that the current marginal predictions and every prefix-conditioned prediction are all conditionals of one common joint distribution. For any reveal order π, the probability chain rule gives

$$\sum_{r=1}^{k} \log p\left(X_{s_{\pi_r}} = y_{\pi_r} \mid \mathbf{z}, \mathbf{X}_{s_{\pi_{<r}}} = \mathbf{y}_{\pi_{<r}}\right) = \log p(\mathbf{y}_{\mathcal{S}} \mid \mathbf{z}). \tag{16}$$

Subtracting $\sum_{r=1}^{k} \log p_{s_r}(y_r \mid \mathbf{z})$ therefore recovers $c(a \mid \mathbf{z})$ independently of π. A DLLM queried at different partially masked states is not guaranteed to define one globally consistent joint distribution, so practical sequential scores can differ across orders. Averaging over Π_k reduces sensitivity to any particular reveal order.

A.4 Candidate Construction and Probability Support

Candidate construction follows the same proposal-based structure for every FACTS transition. At each eligible position, a non-mask token is proposed from Gumbel-perturbed logits. The probability of that proposed token under the clean-renormalized distribution is then used as its position score, optionally with an additional position-level Gumbel perturbation, and the top K positions form the candidate action. Candidates are sorted by position and deduplicated using their exact position–token pairs. Duplicate attempts are not replenished because they represent the same deterministic successor state and receive the same FACTS score.

The raw model distribution may assign probability mass to [M], but [M] is not a valid committed action. FACTS therefore conditions every entropy and token log probability on the clean vocabulary $\mathcal{V}$. The same mask-excluded normalization is used for candidate position scores, $\Delta H(a)$, $\ell_{\mathrm{fac}}(a)$, and $\ell^{\pi}_{\mathrm{seq}}(a)$. This keeps all quantities on the same action support and prevents differences in mask probability from being interpreted as differences in clean-token uncertainty or compatibility.

A.5 Exact Prefix-State Reuse

For candidate $a_n = \{(s_{n,r}, y_{n,r})\}_{r=1}^{k}$, define $\mathbf{z}_{n,U} = \mathbf{z} \oplus \{(s_{n,r}, y_{n,r}) : r \in U\}$ for every subset $U \subseteq [k]$. If two logical reveal orders have the same revealed subset U at some depth, they contain exactly the same committed position–token pairs and therefore induce the same model input. Under deterministic evaluation with identical timestep, attention, positional information, and model hooks, their logits are identical. FACTS consequently evaluates $\mathbf{z}_{n,U}$ once and reuses its target-position logits for every logical request associated with that state. Each reveal order still receives the same sequence of conditional log probabilities it would receive under independent rollout evaluation, so prefix reuse leaves $\ell^{\pi}_{\mathrm{seq}}(a)$, $C(a)$, and $S_{\mathrm{FACTS}}(a)$ unchanged up to ordinary floating-point variation.

For $\pi \in \Pi_k$ and $1 \le r < k$, let $U_{\pi,r} = \{\pi_1, \ldots, \pi_r\}$ be the subset revealed before scoring the $(r+1)$-st token. Define $\mathcal{U}(\Pi_k) = \{U_{\pi,r} : \pi \in \Pi_k,\ 1 \le r < k\}$ and $D_k = |\mathcal{U}(\Pi_k)|$. For N_c unique candidates, naive evaluation requires $N_c|\Pi_k|(k-1)$ sequential rows, whereas prefix-state reuse requires only $N_c D_k$ rows, where

$$D_k \le \min\{|\Pi_k|(k-1), 2^k - 2\}. \tag{17}$$

If all $k!$ orders are evaluated, every non-empty proper subset occurs and $D_k = 2^k - 2$. The exhaustive sequential cost therefore decreases from $N_c\, k!(k-1)$ to $N_c(2^k - 2)$. Future Uncertainty Reduction additionally requires N_c complete successor-state rows, so the total additional model-row work beyond the shared current-state forward is $N_c(1 + D_k)$ when $\mathcal{R} \neq \varnothing$.

Model-row counts measure computational work. FACTS batches candidate successor states and collects distinct prefix states into a shared execution queue. If the prefix queue is evaluated with micro-batch size B, its model evaluations require approximately $\lceil N_c D_k / B \rceil$ forward calls while bounding peak activation memory. Only target-position logits are converted to full precision for the clean log-softmax used in compatibility scoring. When the selected successor state is exactly the state used at the next decoding iteration and all auxiliary conditioning is unchanged, its previously computed logits can also be reused. Candidate deduplication, prefix-state reuse, batching, micro-batching, and valid logit reuse preserve the FACTS objective; sampling only a subset of the $k!$ reveal orders changes the order average and is therefore the only approximation in the sequential evaluation itself.

Table 6: Benchmark configurations and evaluation procedures used in our experiments.

Benchmark	Configuration	Split	# Examples	Metric
GSM8K	main	test	1,319	Final-answer accuracy
MATH-500	–	test	500	Symbolic equivalence
HumanEval	–	test	164	Execution pass@1
MBPP	sanitized	test	257	Execution pass@1

B Detailed Experimental Setup

B.1 Benchmarks and Evaluation

Table 6 summarizes the dataset splits and evaluation procedures used in our experiments. All experiments are conducted on the complete evaluation split without subsampling.

GSM8K. We use the `main/test` split of GSM8K[1]. The generated response is reduced to a final numerical answer using a fixed extraction rule, after which correctness is determined by exact agreement with the reference answer. The extraction procedure follows the GSM8K evaluation convention used by `lm-evaluation-harness` and is kept fixed for all methods.

MATH-500. We evaluate on the complete 500-example test split of MATH-500[2]. We first extract the final answer from the generated response and then compare it with the reference answer using `Math-Verify`[3]. The parsed prediction and reference are verified under strict symbolic equivalence rather than string equality.

HumanEval. For HumanEval[4], the model generates a continuation of the official function stub. We append the extracted continuation to the original stub and execute the official task tests. A sample is counted as correct only when all tests complete successfully. We generate one completion for each problem in each run and therefore report empirical pass@1.

MBPP. We use the `sanitized/test` split of MBPP[5] containing 257 examples. The generated Python program is evaluated against the public `test_list` associated with each example. As with HumanEval, correctness is determined through execution in a bounded subprocess and we report empirical pass@1.

B.2 Evaluation Prompts

We use the same task-specific prompts for all decoding methods. For few-shot evaluation, the demonstrations are fixed rather than sampled independently for each example. The demonstrations and the test query are concatenated into a single user message, which is then wrapped by the native chat template of Dream or LLaDA. We do not introduce an additional system message. The resulting shot settings are 4-shot for GSM8K, 4-shot for MATH-500, 3-shot for MBPP, and 0-shot only for HumanEval.

GSM8K. For zero-shot evaluation, the prompt contains only this query and no explicit chain-of-thought instruction.

```
Q: <question>

A:
```

For few-shot evaluation, we prepend four fixed worked examples before the test query. The examples are the first four demonstrations, in their original order, from the GSM8K CoT configuration

[1] https://huggingface.co/datasets/openai/gsm8k
[2] https://huggingface.co/datasets/HuggingFaceH4/MATH-500
[3] https://pypi.org/project/math-verify/
[4] https://huggingface.co/datasets/openai/openai_humaneval
[5] https://huggingface.co/datasets/google-research-datasets/mbpp

of `lm-evaluation-harness` [6]. Their complete rationales and the final "The answer is ..." statements are retained. The test query itself remains unchanged and does not contain an explicit chain-of-thought instruction.

```
Q: <fixed demonstration 1>

A: <worked rationale 1> The answer is <answer 1>.

Q: <fixed demonstration 2>

A: <worked rationale 2> The answer is <answer 2>.

Q: <fixed demonstration 3>

A: <worked rationale 3> The answer is <answer 3>.

Q: <fixed demonstration 4>

A: <worked rationale 4> The answer is <answer 4>.

Q: <question>

A:
```

MATH-500. For zero-shot evaluation, no demonstration is included.

```
Problem:
<problem>

Solution:
```

For few-shot evaluation, we prepend the four fixed worked examples returned by `list_fewshot_samples()` in the Minerva MATH utilities of `lm-evaluation-harness`. The complete worked solutions, including their final-answer statements, are retained. The evaluation set remains MATH-500; only the prompt format and demonstrations are inherited from Minerva MATH.

[6] https://github.com/eleutherai/lm-evaluation-harness

```
Problem:
<fixed demonstration 1>

Solution: <worked solution 1>

Problem:
<fixed demonstration 2>

Solution: <worked solution 2>

Problem:
<fixed demonstration 3>

Solution: <worked solution 3>

Problem:
<fixed demonstration 4>

Solution: <worked solution 4>

Problem:
<problem>

Solution:
```

HumanEval. HumanEval uses an instruction-style user prompt together with a fixed assistant-side prefill. The user content is

```
Write a solution to the following problem and make sure that it
  passes the tests:
```<official function stub>
```

After the model-specific chat template introduces the assistant generation marker, we append

```
Here is the completed function:
```python
<official function stub>
```

The masked generation region therefore continues from the original function stub rather than regenerating it from scratch.

MBPP. The zero-shot query is

```
You are an expert Python programmer, and here is your task: <problem>
Your code should pass these tests:

<test 1>
<test 2>
<test 3>
[BEGIN]
```

For few-shot evaluation, we prepend exactly three fixed demonstrations with official MBPP task IDs 2, 3, and 4, in this order. Each demonstration contains its task description, three public tests, the reference program between `[BEGIN]` and `[DONE]`, and no natural-language chain-of-thought. The demonstrations are drawn from the dedicated MBPP prompt examples and are disjoint from the scored `sanitized/test` split.

```
You are an expert Python programmer, and here is your task:
<demonstration problem>
Your code should pass these tests:

<demo test 1>
<demo test 2>
<demo test 3>
[BEGIN]
<reference program>
[DONE]

... two more fixed demonstrations ...

You are an expert Python programmer, and here is your task:
<test problem>
Your code should pass these tests:

<test 1>
<test 2>
<test 3>
[BEGIN]
```

B.3 Baseline Implementations

All baselines operate on the same active block and commit exactly K positions at each transition whenever at least K masked positions remain. They share the same model predictions, generation budget, and token sampling procedure whenever applicable. Ties between position scores are resolved deterministically by position index.

For clarity, we distinguish the model's full output distribution $p_{\theta,i}$ from its clean-token conditional distribution

$$\widetilde{p}_{\theta,i}(v) = \frac{p_{\theta,i}(v)}{1 - p_{\theta,i}([\mathtt{M}])}, \qquad v \in \mathcal{V}. \tag{18}$$

Different published criteria naturally operate on different probability supports; we retain these semantics rather than forcing every baseline to use the same renormalization.

Uniform (Nie et al., 2026). Uniform samples K masked positions uniformly without replacement from the current active block. The token committed at each selected position is sampled from the corresponding non-mask token distribution.

Confidence (Chang et al., 2022). Confidence ranks positions by

$$s_i = \max_{v \in \mathcal{V}} \widetilde{p}_{\theta,i}(v \mid \mathbf{z}) \tag{19}$$

and reveals the K positions with the largest scores.

Entropy (Wang et al., 2026). Entropy ranks positions by increasing entropy of the full model distribution, including the mask token,

$$s_i = -H(p_{\theta,i}(\cdot \mid \mathbf{z}))\,, \tag{20}$$

and selects the top K positions.

Margin (Kim et al., 2025). Margin uses the difference between the two largest probabilities under the clean-renormalized distribution,

$$s_i = \widetilde{p}^{(1)}_{\theta,i} - \widetilde{p}^{(2)}_{\theta,i}, \tag{21}$$

where (1) and (2) denote the largest and second-largest clean-token probabilities.

KLASS (Kim et al., 2026). We use a fixed-K adaptation of KLASS. The stability score is computed from the KL divergence between the current distribution and the previous prediction of the same position, using $D_{\mathrm{KL}}(p_{\mathrm{current}} \| p_{\mathrm{previous}})$, a history length of one, and $\epsilon = 5 \times 10^{-4}$. The K most stable eligible positions are committed at each transition.

Table 7: Default decoding hyperparameters used in the controlled experiments. Candidate-specific parameters apply only to candidate-based methods.

Hyperparameter	Value
Block length B	16
Tokens per step K	$\{2, 4\}$
Token temperature	0.7
Position temperature	0.1
Proposal attempts N	8
Decoding transitions	L/K
KV cache	None
Confidence bypass	Disabled
λ_H	1.0
λ_C	1.0
Exhaustive order enumeration	$K \leq 3$
Sampled orders for $K > 3$	8
TC order seed	42
TC forward micro-batch size	16

LookUM (Lee et al., 2026). Our controlled LookUM baseline performs one-step entropy lookahead within the current active block and selects candidate actions according to their resulting future uncertainty. This corresponds to the active-block adaptation used in our comparison and does not include the complete SMC/NIS path-pool machinery of the original system.

Info-Gain (Yang et al., 2026). We use the candidate-based Information Gain sampler as the strongest direct lookahead baseline. At each transition it constructs N candidate actions and ranks them according to their effect on the uncertainty of the subsequent masked state. For the controlled fixed-K experiments, confidence-threshold bypass is disabled so that all methods reveal the same number of tokens per step.

B.4 Decoding and Method-specific Hyperparameters

Unless otherwise specified, we use the hyperparameters in Table 7 for both Dream and LLaDA. We evaluate using the same set of random seeds across methods so that comparisons are paired at the benchmark-example and seed levels. The reported results are averaged across three seeds, $\{42, 3407, 2026\}$.

For $K = 2$, both possible reveal orders are enumerated exactly. They contain two distinct one-token prefix states, giving $D_2 = 2$.

For $K = 4$, we sample eight unique reveal orders from the $4! = 24$ permutations using the fixed TC order seed. These eight orders contain 24 logical conditional queries per candidate before reuse, but repeated prefix subsets are evaluated only once. Under the fixed order set used in our experiments, they induce $D_4 = 12$ unique prefix states per candidate. Hence the sequential compatibility term requires at most $12N_c \leq 96$ physical model rows per scored transition rather than $24N_c \leq 192$ logical rows.