

Plan for Speed: Dilated Scheduling for Masked Diffusion Language Models

Omer Luxembourg, Haim Permuter, Eliya Nachmani

School of Electrical and Computer Engineering, Ben-Gurion University of the Negev
omerlux@post.bgu.ac.il, haimp@bgu.ac.il, eliyana@bgu.ac.il

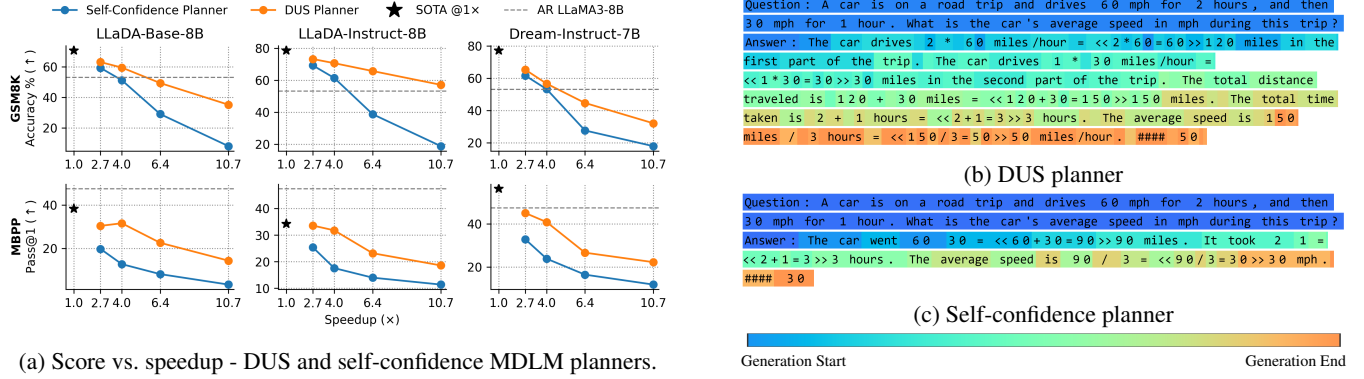


Figure 1: (a) Trade-off between inference speedup and task accuracy on math (GSM8K, MATH500) and code (HumanEval, MBPP) benchmarks. Solid orange curves show DUS planner results; solid blue curves show self-confidence planner results. The star (★) marks the single-token, token-by-token SOTA baseline, and dashed gray lines denote AR LLaMA3-8B performance under the same protocol (Nie et al. 2025b; Ye et al. 2025b; Grattafiori et al. 2024). (b), (c) Chain-of-thought on a GSM8K example with block size $B = 32$ (speedup 6.4×). Token shading (blue→orange) encodes the unmasking iteration. (b), DUS, generates a full, coherent reasoning trace; (c), self-confidence, truncates its chain-of-thought prematurely.

Abstract

Masked diffusion language models (MDLMs) promise fast, non-autoregressive text generation, yet existing samplers, which pick tokens to unmask based on model confidence, ignore interactions when unmasking multiple positions in parallel and effectively reduce to slow, autoregressive behavior. We propose the Dilated Unmasking Scheduler (DUS), an inference-only, planner-model-free method that partitions sequence positions into non-adjacent dilated groups and unmasked them in parallel so as to minimize an upper bound on joint entropy gain at each denoising step. By explicitly trading off the number of network calls against generation quality, DUS recovers most of the performance lost under traditional parallel unmasking strategies. Across math (GSM8K, MATH500), code (HumanEval, MBPP) and general-knowledge benchmarks (BBH, MMLU-Pro), DUS outperforms confidence-based planners, without modifying the underlying denoiser, and reveals the true speed-quality frontier of MDLMs.

1 Introduction

Diffusion-based language models have emerged as a promising alternative to traditional autoregressive (AR) large language models, offering potential advantages in parallel generation and controllability (Campbell et al. 2022; Lou,

Meng, and Ermon 2024; Sahoo et al. 2024). In the discrete diffusion paradigm for text, generation proceeds as an iterative denoising process: starting from a fully noised (e.g., masked) sequence, the model gradually reconstructs the original text over multiple timesteps. While this enables any-order generation, better quality requires one denoising pass per token, which leads to slow inference, as generating a sequence of length G typically entails $\mathcal{O}(G)$ denoiser calls.

AR LLMs have traditionally generated text token-by-token, leading to linear inference latency. Speculative decoding tackles this by first using a lightweight “draft” model to propose multiple tokens in parallel, then having the full AR model verify them - achieving speedups without retraining or architecture changes (Leviathan, Kalman, and Matias 2023; Xia et al. 2022). By contrast, diffusion-based LMs reconstruct the entire sequence simultaneously: although naive masked-denoising still incurs $\mathcal{O}(G)$ passes, it inherently supports any-order and fully parallel generation, laying the groundwork for inference schedules that reduce denoiser calls below linear time.

Existing strategies to accelerate diffusion sampling either introduce heuristic planners that select tokens to unmask based on confidence or entropy, or employ semi-AR block-

wise generation, which divides the sequence into contiguous spans and applies parallel diffusion within each block to preserve global coherence (Ye et al. 2025b; Arriola et al. 2025; Nie et al. 2025b). However, selecting a planner can be shortsighted and prone to error propagation, and semi-AR diffusion still incurs $\mathcal{O}(B)$ denoiser calls per block of size B . Recent work has demonstrated the effectiveness of framing unmasking as an inference-time planning paradigm (Peng et al. 2025), while others have incorporated planning mechanisms directly into the ELBO training objective (Liu et al. 2025). Additional research addresses the computational inefficiency of standard token-by-token denoising by proposing non-uniform schedules (Park et al. 2024) or denoiser’s confidence based unmasking schemes (Ben-Hamu et al. 2025) enabling multiple tokens to be revealed in parallel.

In this work, we introduce the *Dilated Unmasking Scheduler* (DUS), a purely inference-time, model-agnostic planner that partitions each block of length B into logarithmically many iterations by revealing tokens in a fixed dilation pattern. Under a first-order Markov assumption, this schedule minimizes the joint conditional entropy at each step, although reducing the number of denoiser calls from $\mathcal{O}(B)$ to $\mathcal{O}(\log B)$ without any retraining or extra planner modules. Our main contributions are:

- **Inference-Only, Model-Agnostic Decoding:** A drop-in strategy requiring zero modifications to model architecture or training.
- **Logarithmic Unmasking Schedule:** Deterministic, coarse-to-fine dilation that respects local context and minimizes joint entropy, in $\mathcal{O}(\log B)$ denoiser iterations.
- **Theoretical Guarantees:** We show that DUS approaches the joint entropy bound at each iteration under fast-mixing assumptions, compared to baselines with the same step budget.
- **Empirical Validation:** Extensive experiments on GSM8K, HumanEval, and MBPP with LLaDA-8B and Dream-7B demonstrate up to an order-of-magnitude reduction in denoiser calls and consistent quality improvements over confidence-based planners.

The remainder of this paper is organized as follows. Section 2 presents the fundamentals of masked diffusion language models. In Section 3, masked diffusion framework is formalized, DUS is introduced, and theoretical analysis under Markovian assumption is given. In Section 4, we present experimental results on mathematical reasoning, code generation, and general knowledge chain-of-thought (COT) benchmarks. Finally, Section 5 concludes and outlines future directions.

2 Related Work

Discrete diffusion sampling relies on a planner to decide which masked tokens to reveal at each reverse step (Liu et al. 2025). Early “self-planners” use the diffusion denoiser itself to rank positions by simple criteria: top- k highest probability (confidence), lowest conditional entropy, or top- k probability margin (the gap between the highest and second-highest

scores) (Campbell et al. 2022; Sahoo et al. 2024; Kim et al. 2025).

Path Planning (P2) extends beyond these self-planners by incorporating an external guiding model - a pretrained BERT - to evaluate candidate token sets according to their output probabilities (Peng et al. 2025). P2 compares denoiser-guided selection, random sampling, and BERT-scored planners, finding that the learned BERT planner yields better scores in various experiments at the cost of auxiliary model calls.

Semi-AR block diffusion segments a long sequence into contiguous spans of length B , runs denoiser iterations in parallel within each block, and reveals blocks sequentially (Arriola et al. 2025; Nie et al. 2025b). This approach does not reduce the total number of denoiser passes from $\mathcal{O}(G)$ to $\mathcal{O}(G/B)$ (where G is the generation length), it still incurs $\mathcal{O}(B)$ calls per block, and it typically requires a policy to decide block order, most often determined by data characteristics. While the first (Arriola et al. 2025) discuss both training and inference with semi-AR block diffusion, here we focus exclusively on inference.

At the 7-8 B-parameter scale, two masked diffusion language models-Dream-7B (Ye et al. 2025b) and LLaDA-8B (Nie et al. 2025b)-demonstrate the practical viability of fully parallel decoding. LLaDA-8B, trained on 2.3 T tokens, matches LLaMA3-8B (Grattafiori et al. 2024) across a broad suite of zero and few-shot evaluations, particularly on mathematical reasoning and Chinese, language benchmarks-while retaining any-order generation via its principled diffusion objective and efficient remasking schedules. Dream-7B, pretrained on 580B tokens with AR model initialization and context-adaptive token-level noise rescheduling, achieves strong performance on mathematics, coding, and planning tasks. Both models offer supervised fine-tuned (SFT) variants for instruction following and applying semi-AR decoding over predefined blocks.

DiffuCoder (Gong et al. 2025) is a 7B-parameter MDLM trained solely on 130 billion tokens of code, an effort that its authors claim frees the model from the strong left-to-right bias typical of text-trained diffusion LLMs. By adjusting sampling temperature, DiffuCoder dynamically controls its “causalness”, allowing fully parallel decoding without resorting to semi-AR blocks. Furthermore, the introduction of a coupled-GRPO reinforcement learning step after pretraining further reduces residual AR tendencies and yields an improvement on code benchmarks without any semi-AR inference fallbacks.

Inference efficiency can also be improved orthogonally via Key-Value (KV) caching, where stable key and value tensors from transformer layers are reused across denoising steps. Recent works such as *FreeCache* (Hu et al. 2025) and *dKV-Cache* (Ma et al. 2025) report up to $3\times$ speedups with minimal impact on generation quality. These approaches require additional memory to store cached activations, and in the case of FreeCache an auxiliary AR model is needed to guide diffusion.

Adaptive scheduling methods determine which tokens to unmask and when to do so during sampling, with the goal of improving inference speed while preserving generation

quality. On the timestep axis, Jump Your Steps (JYS) selects a non-uniform subset of noise levels by estimating KL divergences between adjacent diffusion kernels - trading a small ELBO loss for significantly fewer denoiser calls (Park et al. 2024). Common discrete masked diffusion models, however, are trained with cross-entropy objectives rather than continuous score-based losses (Sahoo et al. 2024), making direct application of log-likelihood schedules less straightforward. At the token granularity, EB-Sampler uses an entropy-bounded threshold to pick the largest set of tokens whose cumulative conditional entropy remains below a user-specified bound - achieving $2 - 3\times$ empirical speedups on coding and math benchmarks without retraining (Ben-Hamu et al. 2025). This per-token heuristic offers practical gains but lacks formal iteration-complexity guarantees and does not fully account for token dependencies when unmasking in parallel.

At scale, both open-source and commercial systems illustrate the practical adoption of diffusion LLMs. Dream-7B and LLaDA-8B employ planner heuristics and blockwise (semi-AR) diffusion to rival AR baselines on text and code benchmarks (Ye et al. 2025a,b; Nie et al. 2025b), and recent offerings - DeepMind’s Gemini Diffusion (Google DeepMind 2025) and Inception Labs’ Mercury (Labs 2025) - further underscore the growing throughput and practical adoption of diffusion-based language models.

3 Method

This section introduces the foundational concepts of MDLMs, reviews existing unmasking planners, presents our DUS, and provides theoretical guarantees of its optimality.

3.1 Notation

Let $\mathcal{X} = \{X_1, \dots, X_G\}$ be a sequence of random variables forming a stationary, ergodic, first-order Markov chain with fast-mixing properties. At each decoding iteration step t , a masked version of the sequence is denoted by $\mathcal{M} = \{M_1, \dots, M_G\}$, where M_i is replaced by X_i if token i has been unmasked, and $M_i = [\text{MASK}]$ otherwise. Let $\mathcal{S}_t \subset \{\mathcal{X}, \mathcal{M}\}$ denote the state at denoiser iteration t , i.e., the collection of all currently known (unmasked) tokens and remaining masked positions.

Define a *planner* \mathcal{P}_t that chooses $k \leq G$ candidate indices $\{i_1, \dots, i_k\} \subset \{1, \dots, G\}$ of masked tokens to be unmasked at iteration t , and a *denoiser* \mathcal{D} that maps each of those masked tokens $M_{i_j}, j \in \{1, \dots, k\}$ to a prediction of X_{i_j} conditioned on the current state

$$\hat{X}_{i_j} = \mathcal{D}(M_{i_j} \mid \mathcal{S}_t). \quad (1)$$

The objective is to minimize the joint entropy of the full sequence, i.e., $H(\mathcal{X})$. In this case, with conditioned generation on a prompt, which is an initial state S_0 , our minimization objective is

$$H(\mathcal{X} \mid S_0). \quad (2)$$

3.2 MDLM as Denoiser and Planner

Masked diffusion samplers consist of two core components:

1. **Denoiser.** A pretrained network \mathcal{D}_θ that, at each reverse diffusion step, takes a partially masked sequence and predicts the values of all masked tokens based on the current unmasked context (Campbell et al. 2022; Austin et al. 2023).
2. **Planner.** A strategy \mathcal{P}_t that, at each iteration t , selects which positions in the sequence to unmask next. Common planners include ancestral diffusion sampling (random selection), confidence-based selection, entropy-based selection, and pre-trained planners such as in (Peng et al. 2025; Liu et al. 2025).

Most large-scale discrete diffusion models perform inference in a semi-AR, block-wise fashion: the sequence is partitioned into consecutive blocks of size B , and each block is unmasked and denoised before moving to the next (Nie et al. 2025b). Block diffusion training masks entire spans of length B and learns to recover them jointly, yielding stronger local coherence, while other methods train on unmasking the whole sequence (Arriola et al. 2025).

Our DUS method slots in as a *inference-only* planner. It requires no retraining of the denoiser and can serve as a drop-in planning strategy for any standard discrete diffusion model - MDLM (Sahoo et al. 2024), MD4 (Shi et al. 2025), LLaDA (Nie et al. 2025b), Dream (Ye et al. 2025b,a), DiffuCoder (Gong et al. 2025), and others - reducing per-block complexity from $\mathcal{O}(B)$ to $\mathcal{O}(\log B)$.

3.3 Detailed Formulation

Because the denoiser’s training (recovering masked tokens per reverse step to minimize ELBO loss) is optimized for partial reconstruction, it cannot reveal all tokens at once. Consequently, inference unfolds over several unmasking-denoising iterations (Campbell et al. 2022; Nie et al. 2025a). Recent large-scale diffusion LLMs extend this by partitioning the sequence into semi-AR blocks of length $B \leq K$, applying several denoiser iterations within each block. Accordingly, at iteration t , define the state \mathcal{S}_t to include (1) all previously unmasked blocks, (2) the current block (with both masked and unmasked tokens), and (3) the remaining masked blocks. Under this formulation, inference reduces to minimizing the joint conditional entropy $H(X_b, \dots, X_{b+B} \mid \mathcal{S}_t)$, where $\{b, \dots, b+B\}$ are indices in the current inferred block, and parallel k -tokens unmasking seek to minimize the joint entropy gain

$$H(X_{i_1}, \dots, X_{i_k} \mid \mathcal{S}_t), \quad (3)$$

where $\{i_1, \dots, i_k\} \subseteq \{b, \dots, b+B\}$. Due to the nature of the model, each token is unmasked independently of the other $k-1$, hence the minimization term is practically

$$\sum_{j=1}^k H(X_{i_j} \mid \mathcal{S}_t). \quad (4)$$

The following sections compares multiple planner strategies including our new method and evaluates their effectiveness in minimizing the block conditional entropy across iterations of unmasking.

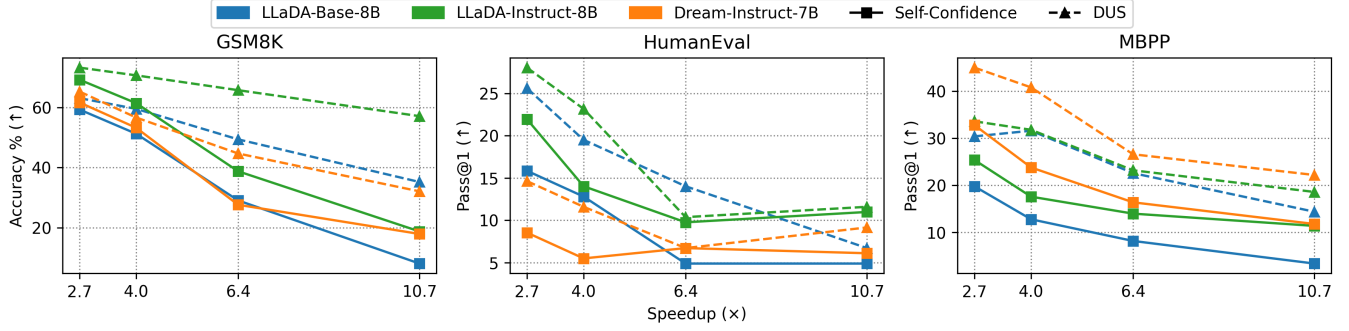


Figure 2: Experiments on GSM8K, HumanEval, and MBPP for variate speedup factors - defined by semi-AR inference block size. Higher score (Accuracy / Pass@1) is better. Each color represent a different model, while marker indicates the two planners tested - self-confidence ■; DUS ▲. Across all datasets and speedups, DUS achieves higher scores compared to the traditional planner.

3.4 Self-Planners Denoiser Confidence Guided

Next both planners are referred as self-planners - using the denoiser’s signal as a confidence measurement to chose the next tokens to be unmasked.

Self-Confidence by Probability. Define the output probabilities of the denoiser \mathcal{D} as $p_{\mathcal{D}}(X_{i_j} | \mathcal{S}_t)$ for the candidate token at sequence index i_j given state \mathcal{S}_t . At each timestep t , the planner selects the top- k tokens according to the denoiser’s confidence, i.e. $\arg \max p_{\mathcal{D}}(X_{i_j} | \mathcal{S}_t)$ for $j \in \{1, \dots, k\}$. This strategy tends to select tokens that are predictable, often resulting in unmasking tokens that are highly correlated with previously unmasked tokens (e.g., adjacent in the sequence), leading to redundancy and higher entropy gain:

$$\begin{aligned}
 & H(X_{i_1}, \dots, X_{i_k} | \mathcal{S}_t) \\
 &= \sum_{j=1}^k H(X_{i_j} | X_{i_1}, \dots, X_{i_{j-1}}, \mathcal{S}_t) \\
 &\stackrel{(a)}{\leq} \sum_{j=1}^k H(X_{i_j} | \mathcal{S}_t),
 \end{aligned} \tag{5}$$

where (a) valid since conditioning reduces entropy. Clearly, self-planner by confidence has a lower bound on the entropy gain, which is the joint entropy, while there is no guarantee for the bound equality.

Self-Confidence by Entropy. This variant selects the k tokens with the minimal individual conditional entropy $H(X_{i_j} | \mathcal{S}_t)$. While it avoids highly uncertain positions, it does not account for mutual information (MI) between candidates, leaving again the entropy gain suboptimal.

3.5 DUS as Predefined Planner

Introducing our DUS, a deterministic planner that, for a block of size B with exponent base a , predefines an unmasking strategy fixing which tokens are revealed at each of the $R = \lceil \log_a B \rceil$ iterations. Let

$$R = \lceil \log_a B \rceil, \quad s_t = \lfloor \frac{B}{a^t} \rfloor, \quad t = 1, \dots, R. \tag{6}$$

Hence, define the iterative planner \mathcal{P}_t and the incremental unmasked group at each iteration \mathcal{U}_t as,

$$\begin{aligned}
 \mathcal{U}_0 &= \emptyset, \\
 \mathcal{P}_t &= \{k \in \{1, \dots, B\} \setminus \mathcal{U}_{t-1} \mid (k-1) \bmod s_t = 0\}, \\
 \mathcal{U}_t &= \mathcal{U}_{t-1} \cup \mathcal{P}_t.
 \end{aligned} \tag{7}$$

If $|\mathcal{P}_R| < |\mathcal{P}_{R-1}|$, merge \mathcal{P}_R into \mathcal{P}_{R-1} to balance coverage. This completes in $R \approx \lceil \log_a B \rceil$ steps, where $|\mathcal{P}_t| = \max(a, a^{t-1})$ is an incremental series.

For example, let $B = 8$ and $a = 2$, then $R = 3, s_1 = 4, s_2 = 2, s_3 = 1$, and

$$\begin{aligned}
 \mathcal{P}_1 &= \{k \mid (k-1) \bmod 4 = 0\} = \{1, 5\}, \\
 \mathcal{P}_2 &= \{k \notin \{1, 5\} \mid (k-1) \bmod 2 = 0\} = \{3, 7\}, \\
 \mathcal{P}_3 &= \{k \notin \{1, 5, 3, 7\} \mid (k-1) \bmod 1 = 0\} = \{2, 4, 6, 8\}.
 \end{aligned}$$

3.6 Theoretical Analysis of DUS

Presented below is the main lemma showing that DUS achieves the joint-entropy bound equal to the sum of individual conditional entropies.

Lemma 1 *Under a fast-mixing first-order Markov chain, let $i_1 < \dots < i_k$ be the indices selected by DUS. Then*

$$H(X_{i_1}, \dots, X_{i_k} | \mathcal{S}_t) \approx \sum_{j=1}^k H(X_{i_j} | \mathcal{S}_t). \tag{8}$$

The proof of Lemma 1 hinges on the following auxiliary lemma, which bounds the MI in terms of the maximal correlation coefficient.

Lemma 2 *Define \mathcal{X} as a fast-mixing first-order Markov chain. Let ρ be the Hirschfeld–Gebelein–Rényi maximal correlation coefficient (Rényi 1959) between random variables, defined by*

$$\rho = \sup_{\substack{f \in L^2(P_{X_{i_n}}), g \in L^2(P_{X_{i_m}}); \\ \mathbb{E}[f(X_{i_n})] = 0, \mathbb{E}[f(X_{i_n})^2] = 1; \\ \mathbb{E}[g(X_{i_m})] = 0, \mathbb{E}[g(X_{i_m})^2] = 1}} \mathbb{E}[f(X_{i_n}) g(X_{i_m})]. \tag{9}$$

For two positions i_m, i_n at distance $d = |i_m - i_n|$, the MI satisfies

$$I(X_{i_m}; X_{i_n}) \leq -\frac{1}{2} \log(1 - \rho^{2d}) = O(\rho^{2d}). \quad (10)$$

The decay bound in Lemma 2 is proved in (Asoodeh, Alajaji, and Linder 2016), where exponential correlation decay is established under fast-mixing.

Proof of Lemma 1: Under fast-mixing and $\rho < 1$, Lemma 2 implies

$$I(X_{i_m}; X_{i_n}) \leq -\frac{1}{2} \log(1 - \rho^{2d}) \leq \frac{\rho^{2d}}{2(1 - \rho^2)}.$$

Let $\varepsilon > 0$. Since $\rho < 1$, there exists D such that for all $d \geq D$,

$$I(X_{i_m}; X_{i_n}) < \frac{\varepsilon}{k-1}. \quad (11)$$

Without loss of generality, let $i_1 < \dots < i_k$ be the indices of the k tokens chosen by DUS, each pair separated by at least D . The joint conditional entropy then decomposes as

$$\begin{aligned} H(X_{i_1}, \dots, X_{i_k} | \mathcal{S}_t) &= \\ &= \sum_{j=1}^k H(X_{i_j} | X_{i_1}, \dots, X_{i_{j-1}}, \mathcal{S}_t) \\ &= H(X_{i_1}) + \sum_{j=2}^k [H(X_{i_j} | \mathcal{S}_t) \\ &\quad - I(X_{i_j}; X_{i_1}, \dots, X_{i_{j-1}} | \mathcal{S}_t)] \\ &\stackrel{(a)}{=} H(X_{i_1}) + \sum_{j=2}^k [H(X_{i_j} | \mathcal{S}_t) - I(X_{i_j}; X_{i_{j-1}} | \mathcal{S}_t)] \\ &\stackrel{(b)}{=} H(X_{i_1}) + \sum_{j=2}^k [H(X_{i_j} | \mathcal{S}_t)] - \varepsilon \\ &\approx \sum_{j=1}^k H(X_{i_j} | \mathcal{S}_t). \end{aligned} \quad (12)$$

Transition (a) follows from the first-order Markov property, and (b) uses the bound from Equation (11), since our dilated pattern maximizes distances. \square

In summary, DUS minimizes the independent conditional entropy and achieves the joint entropy bound, whereas other planners are unbounded from below. Three core features are integrated in our planner:

1. **Maintained Spacing Across Iterations:** By preserving the same sparse dilation at every round, tokens distant is guaranteed - and thus the bound in (11) holds throughout all R iterations, unlike schedulers that cluster adjacent tokens and break the independence assumption, mainly for later iterations. Hence, (12) approximation preserved across the block iterations.
2. **Contextual Conditioning:** By revealing tokens spread across the sequence early, later predictions are conditioned on richer adjacent context, further reducing remaining entropy.

3. **Skip Mechanism:** As an interleavable feature atop our deterministic schedule, tokens whose denoiser signal (e.g. low confidence or low negative entropy) falls below a threshold are deferred to the next iteration \mathcal{P}_{t+1} - enabling DUS to preserve its fixed dilation while still adapting to model uncertainty and avoiding the exposure of fully uncurtained tokens.

Under a fast-mixing, first-order Markov process, DUS achieves lower cumulative conditional entropies (the sum of per-iteration entropy gains) than denoiser-guided planners in parallel unmasking - by explicitly reaching the joint entropy bound at each step and distributing informative context most efficiently across iterations.

4 Experiments

This section evaluates the generative and downstream capabilities of our new planner for MDLMs on various families of tasks: mathematical reasoning, code generation and general knowledge COT multiple-choice questions. Three pre-trained diffusion-based LLM - LLada, Dream, and DiffuCoder - under unified benchmarking protocols. Detailed information about datasets and experiments settings presented in Appendix A.1

4.1 Experimental Setup

For each model and dataset, decoding is applied with a semi-AR masked diffusion denoising process for different block sizes $B \in \{8, 16, 32, 64\}$. Decoding proceeds in $n_{\text{blocks}} = G/B$ rounds, in each of which the model predicts all currently masked tokens before moving to the next block. The Number of Function Evaluations (NFE) of a block is $\text{NFE}_{\text{block}} = \log_2 B$ on average - due to the nature of our DUS planner. Thus, k , the number of unmasked tokens in parallel in one iteration is set to $k = B/\log_2 B$ on average. The total number of denoiser calls - total NFE, is

$$\text{NFE} = n_{\text{blocks}} \text{NFE}_{\text{block}} = \frac{G}{B} \log_2 B, \quad (13)$$

hence larger blocks results in with more parallelism against fewer diffusion evaluations. Section 4.4 investigate further the effect on performance of fixed k versus incremental k (as in DUS), for self-confidence and random planners. Beside this particular experiment, all experiment's other planner but DUS feature fixed k .

We evaluate five masked diffusion LLMs. (1) LLaDA-Base-8B, (2) LLaDA-Instruct-8B (Nie et al. 2025b) - an 8B parameter mask-diffusion transformer pre-trained on a large mixed text + code corpus, and it's instruction-tuned version. (3) Dream-Instruct-7B (Ye et al. 2025b) - a 7B parameter instruction-tuned mask-diffusion model, supervised fine-tuned based on the base version. To streamline our analysis, we focus on the instruction-tuned variant and do not report results for the pre-trained Dream-Base model, whose out-of-the-box performance for our experiments is appreciably lower than that of its fine-tuned version. (4) DiffuCoder-Base-7B, (5) DiffuCoder-Instruct-7B (Gong et al. 2025) - a 7B parameter mask-diffusion transformer trained exclusively on 130B tokens of code with AR-initialization, and

Table 1: Math (GSM8K, MATH500) and Code (Humaneval, MBPP) benchmarks for self-confidence (Conf.) and DUS (ours). Tasks reported accuracy (%) for math and pass@1 for code, at block sizes $B = \{8, 16, 32, 64\}$, with corresponding speedup factor (\times). Diffucoder is tested only on code benchmarks; Humaneval, and MBPP. **Bold** marks better scores.

Model		B=8 $\times 2.7$		B=16 $\times 4$		B=32 $\times 6.4$		B=64 $\times 10.7$	
		Conf.	DUS	Conf.	DUS	Conf.	DUS	Conf.	DUS
GSM8K	LLaDA-Base	59.29	63.08	51.23	59.51	29.04	49.36	8.04	35.18
	LLaDA-Instruct	69.22	73.24	61.41	70.66	38.74	65.73	18.73	57.09
	Dream-Instruct	61.64	65.28	53.22	56.63	27.60	44.66	17.89	32.07
MATH500	LLaDA-Base	16.6	21.4	11.2	19.2	6.0	13.6	2.6	10.2
	LLaDA-Instruct	21.4	23.8	15.4	22.8	10.8	19.2	8.0	14.8
	Dream-Instruct	22.4	27.0	15.4	19.8	7.2	13.2	4.0	11.6
Humaneval	LLaDA-Base	15.85	25.61	12.8	19.51	4.88	14.02	4.88	6.71
	LLaDA-Instruct	21.95	28.05	14.02	23.17	9.76	10.37	10.98	11.59
	Dream-Instruct	8.54	14.63	5.49	11.59	6.71	6.71	6.10	9.15
	DiffuCoder-Base	17.07	28.66	6.71	38.41	2.44	21.95	0.61	6.10
	DiffuCoder-Instruct	7.93	22.56	14.02	20.12	13.41	12.80	11.59	8.54
MBPP	LLaDA-Base	19.8	30.4	12.8	31.6	8.2	22.6	3.4	14.4
	LLaDA-Instruct	25.4	33.6	17.6	31.8	14.0	23.2	11.4	18.6
	Dream-Instruct	32.8	45.0	23.8	40.8	16.4	26.6	11.8	22.2
	DiffuCoder-Base	29.2	48.6	17.4	43.0	10.2	27.4	3.4	17.2
	DiffuCoder-Instruct	31.8	46.4	25.6	43.6	21.0	26.6	13.0	18.2

it’s instruction fine-tuned model. Diffucoder authors claim that their results are given without semi-AR protocol, hence for their model without DUS semi-AR inference is off. Two planners are used:

1. **Self-confidence (baseline).** at each block the model’s top- k confident tokens are unmasked at each diffusion reverse iteration, while the others are masked. k is set to a fixed value that is dependent on block size of the semi-AR process, $k = \log_2 B$, unless stated otherwise. LLaDA and Dream models use maximum probability as their confidence while Diffucoder uses entropy (as in their original work).
2. **DUS (ours).** for each block length B the DUS is applied (as defined in Section 3.5), that unmask on average $k = \log_2 B$ tokens in denoiser iteration, across a block.

Experiments report block size B and relative inference speedup, $B/\log_2 B$ (originates from Equation (13)), calculated as the ratio of token-by-token total NFE to the experiment’s total NFE; both planners use the same B and NFE budget, and their final task scores are compared accordingly. A GSM8K problem, for $B = 32$, is visualized in Figure 1b and Figure 1c for DUS and self-confidence respectively.

4.2 Math and Coding Experiments

Models are evaluated on GSM8K, MATH500, HumanEval, and MBPP to assess performance trade-offs between inference speed and accuracy under semi-AR masked diffusion. Figure 1a and 2 plot accuracy vs. speedup for $2.7\times$, $4\times$, $6.4\times$, and $10.7\times$ (block sizes $B \in \{8, 16, 32, 64\}$). Smaller blocks yield higher accuracy at the cost of more denoiser rounds (higher NFE), while larger blocks enable up to $10\times$ fewer iterations but lower end accuracy. The

DUS planner consistently improves up to 27% over the self-confidence baseline at the same NFE budget. GSM8K shows a steady decline in scores as inference speed increases, whereas the code benchmarks (HumanEval and MBPP) exhibit less smooth trends-likely reflecting their already low performance (under 10 %) at higher speedups. Nonetheless, DUS delivers more consistent improvements across all settings.

Table 1 presents extensive results on all of the 4 math and code benchmarks. Consistently, DUS improves or matches the results of the baseline self-confidence planner, as it can achieve up to 40% improvement on math benchmarks and up to 20% on code benchmarks, while using less NFE compared to naive token-by-token inference.

Moreover, it can be observed that particularly on code benchmarks, LLaDA and Diffucoder base models outperform their instruct model while using DUS as their planner, compared to the superiority of instruction tuned models with self-confidence planner.

4.3 General Knowledge Experiments

To further demonstrate DUS’s superiority under accelerated inference budgets, experiments were conducted on the BBH and MMLU-Pro multiple-choice reasoning benchmarks using a few-shot chain-of-thought protocol. Table 2 reports block size B , corresponding speedup factor, and accuracy for LLaDA-Base and LLaDA-Instruct at $B \in \{16, 32\}$. Although absolute gains are smaller than in the math and code experiments, DUS consistently outperforms the self-confidence planner at the same $2.7\times$, $6.4\times$ speedups: BBH accuracy improves by 1–5%, and MMLU-Pro by 5–9%. These results confirm that DUS yields higher-quality reasoning across diverse question formats under constrained infer-

Table 2: General knowledge (BBH, MMLU-pro) benchmarks for self-confidence (Conf.) and DUS (ours). Both are few-shot, COT, multiple-choices dataset on general topics from various fields. Tasks reported accuracy (%) at block sizes $B = \{8, 16, 32, 64\}$, with corresponding speedup factor (\times). **Bold** marks better scores.

Model		B=16 $\times 4$		B=32 $\times 6.4$	
		Conf.	DUS	Conf.	DUS
BBH	LLaDA-Base	40.93	41.48	35.93	40.56
	LLaDA-Instruct	47.59	50.37	44.26	50.93
MMLU-Pro	LLaDA-Base	32.14	37.32	24.11	32.50
	LLaDA-Instruct	25.71	34.64	31.07	34.64

ence budgets.

4.4 Planners Parallelism Performance

To isolate the contribution of DUS’s incremental unmasking schedule from other components, an ablation study varies the number of tokens unmasked per iteration under two strategies: *fixed-k*, which unmasks a constant $k = \log_2 B$ tokens at every step, and *dilated-incremental*, which gradually increases the unmasking group so that the average remains $k = \log_2 B$ but early iterations reveal fewer tokens and later iterations more (as in the DUS planner). Table 3 reports task accuracy (%) on the first 300 samples of GSM8K for LLaDA-Base and LLaDA-Instruct at block sizes $B = 16, 32$, comparing both the self-confidence and random planners under each schedule.

It is observed that DUS attains the highest score across every model and block size configuration. Under self-confidence, however, *fixed-k* outperforms *dilated-incremental*, indicating that confidence-guided selection benefits from uniform parallelism. By contrast, random unmasking scores poorly under *fixed-k* but, when paired with *dilated-incremental* scheduling, especially at larger block sizes, it recovers most of its accuracy loss and even surpasses *fixed-k* self-confidence variants. This behavior follows from the theoretical analysis (Lemma 2): *dilated-incremental* scheduling, for random planner, spaces unmasked tokens to reduce their MI, allowing the joint entropy bound in Equation (12) to closely hold. As a result, even an unguided planner benefits dramatically from incremental unmasking, whereas self-confidence gains little from it.

Overall, DUS delivers substantial gains over the traditional self-confidence approach at every evaluated speedup across diverse domains. Although parallel unmasking of multiple tokens can degrade performance - since the denoiser lacks information from tokens revealed simultaneously - our planner effectively mitigates this issue, recovering a significant portion of the lost accuracy. These results establish a training-free method for accelerating MDLMs inference (both pre-trained and instruction-tuned) without sacrificing major generation quality.

Table 3: Planners parallelism ablation on GSM8K (300 samples, $G = 256$), comparing self-confidence (Conf.) vs. random planners under two unmasking schedules: *fixed-k* and *dilated-incremental* (Inc., as in DUS). Accuracy (%) is shown for both base and instruct LLaDA models at $B = \{16, 32\}$. Best scores are **bold**, second-best are underlined.

Model	Block	Conf.		Random		DUS
		fixed	inc.	fixed	inc.	
Base	16	55.00	49.33	24.67	43.67	61.33
	32	30.67	25.00	10.00	<u>39.00</u>	48.33
Instruct	16	<u>62.33</u>	47.67	40.00	61.33	71.67
	32	39.67	23.67	23.33	<u>53.67</u>	66.67

5 Conclusions

We have introduced the DUS, a purely inference-time, model-free agnostic planner for MDLMs. In extensive experiments on diffusion LLMs, DUS consistently outperforms the traditional denoiser-confidence planner, improving downstream task accuracy by up to 27% on challenging math and code benchmarks, while simultaneously reducing the number of denoising iterations by an order of magnitude. Unlike typical speed-quality trade-offs, our method both accelerates inference and enhances output quality. By unlocking the parallelism inherent in diffusion decoding without any modifications to model architecture or training, DUS demonstrates a new path toward diffusion-based LMs that surpass AR approaches in both efficiency and performance, and inspires the design of future inference-only planners to fully exploit this potential.

References

- Arriola, M.; Gokaslan, A.; Chiu, J. T.; Yang, Z.; Qi, Z.; Han, J.; Sahoo, S. S.; and Kuleshov, V. 2025. Block Diffusion: Interpolating Between Autoregressive and Diffusion Language Models. ArXiv:2503.09573 [cs].
- Asoodeh, S.; Alajaji, F.; and Linder, T. 2016. On Maximal Correlation, Mutual Information and Data Privacy. *IEEE Transactions on Information Theory*, 62(12): 7272–7282.
- Austin, J.; Johnson, D. D.; Ho, J.; Tarlow, D.; and Berg, R. v. d. 2023. Structured Denoising Diffusion Models in Discrete State-Spaces. ArXiv:2107.03006 [cs].
- BenHamu, H.; Gat, I.; Severo, D.; Nolte, N.; and Karrer, B. 2025. Accelerated Sampling from Masked Diffusion Models via Entropy Bounded Unmasking. *arXiv preprint arXiv:2505.24857*.
- Campbell, A.; Benton, J.; Bortoli, V. D.; Rainforth, T.; Deligiannidis, G.; and Doucet, A. 2022. A Continuous Time Framework for Discrete Denoising Models. ArXiv:2205.14987 [stat].
- Gong, S.; Zhang, R.; Zheng, H.; Gu, J.; Jaitly, N.; Kong, L.; and Zhang, Y. 2025. DiffuCoder: Understanding and Improving Masked Diffusion Models for Code Generation. *arXiv preprint arXiv:2506.20639*.
- Google DeepMind. 2025. Gemini Diffusion. [urlhttps://deepmind.google/models/gemini-diffusion/](https://deepmind.google/models/gemini-diffusion/).

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Hu, Z.; Meng, J.; Akhauri, Y.; Abdelfattah, M. S.; Seo, J.-s.; Zhang, Z.; and Gupta, U. 2025. Accelerating diffusion language model inference via efficient kv caching and guided diffusion. *arXiv preprint arXiv:2505.21467*.

Kim, J.; Shah, K.; Kontonis, V.; Kakade, S.; and Chen, S. 2025. Train for the Worst, Plan for the Best: Understanding Token Ordering in Masked Diffusions. *arXiv preprint arXiv:2502.06768*.

Labs, I. 2025. Mercury. [urlhttps://www.inceptionlabs.ai/introducing-mercury](https://www.inceptionlabs.ai/introducing-mercury).

Leviathan, Y.; Kalman, M.; and Matias, Y. 2023. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, 19274–19286. PMLR.

Lightman, H.; Kosaraju, V.; Burda, Y.; Edwards, H.; Baker, B.; Lee, T.; Leike, J.; Schulman, J.; Sutskever, I.; and Cobbe, K. 2023. Let’s Verify Step by Step. *arXiv preprint arXiv:2305.20050*.

Liu, S.; Nam, J.; Campbell, A.; Stärk, H.; Xu, Y.; Jaakkola, T.; and Gómez-Bombarelli, R. 2025. Think While You Generate: Discrete Diffusion with Planned Denoising. ArXiv:2410.06264 [cs].

Lou, A.; Meng, C.; and Ermon, S. 2024. Discrete Diffusion Modeling by Estimating the Ratios of the Data Distribution. ArXiv:2310.16834 [stat].

Ma, X.; Yu, R.; Fang, G.; and Wang, X. 2025. dkv-cache: The cache for diffusion language models. *arXiv preprint arXiv:2505.15781*.

Nie, S.; Zhu, F.; Du, C.; Pang, T.; Liu, Q.; Zeng, G.; Lin, M.; and Li, C. 2025a. Scaling up Masked Diffusion Models on Text. ArXiv:2410.18514 [cs].

Nie, S.; Zhu, F.; You, Z.; Zhang, X.; Ou, J.; Hu, J.; Zhou, J.; Lin, Y.; Wen, J.-R.; and Li, C. 2025b. Large Language Diffusion Models. ArXiv:2502.09992 [cs].

Park, Y.-H.; Lai, C.-H.; Hayakawa, S.; Takida, Y.; and Mit-sufuji, Y. 2024. Jump Your Steps: Optimizing Sampling Schedule of Discrete Diffusion Models. ArXiv:2410.07761 [cs].

Peng, F. Z.; Bezemek, Z.; Patel, S.; Rector-Brooks, J.; Yao, S.; Tong, A.; and Chatterjee, P. 2025. Path Planning for Masked Diffusion Model Sampling. ArXiv:2502.03540 [cs].

Rényi, A. 1959. On Measures of Dependence. *Acta Mathematica Academiae Scientiarum Hungaricae*, 10: 441–451.

Sahoo, S. S.; Arriola, M.; Schiff, Y.; Gokaslan, A.; Mar-roquin, E.; Chiu, J. T.; Rush, A.; and Kuleshov, V. 2024. Simple and Effective Masked Diffusion Language Models. ArXiv:2406.07524 [cs].

Shi, J.; Han, K.; Wang, Z.; Doucet, A.; and Titsias, M. K. 2025. Simplified and Generalized Masked Diffusion for Discrete Data. ArXiv:2406.04329 [cs].

Suzgun, M.; Scales, N.; Schärli, N.; Gehrmann, S.; Tay, Y.; Chung, H. W.; Chowdhery, A.; Le, Q. V.; Chi, E. H.; Zhou, D.; et al. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Wang, Y.; Ma, X.; Zhang, G.; Ni, Y.; Chandra, A.; Guo, S.; Ren, W.; Arulraj, A.; He, X.; Jiang, Z.; et al. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Xia, H.; Ge, T.; Wang, P.; Chen, S.-Q.; Wei, F.; and Sui, Z. 2022. Speculative decoding: Exploiting speculative execution for accelerating seq2seq generation. *arXiv preprint arXiv:2203.16487*.

Ye, J.; Gao, J.; Gong, S.; Zheng, L.; Jiang, X.; Li, Z.; and Kong, L. 2025a. Beyond Autoregression: Discrete Diffusion for Complex Reasoning and Planning. ArXiv:2410.14157 [cs].

Ye, J.; Xie, Z.; Zheng, L.; Gao, J.; Wu, Z.; Jiang, X.; Li, Z.; and Kong, L. 2025b. Dream 7B.

A Appendix

A.1 Datasets and Settings

All experiments were conducted on NVIDIA Tesla V100 GPUs. Evaluation was performed using *Language Model Evaluation Harness* repository (<https://github.com/EleutherAI/lm-eval-harness>) and our code is based on the LLaDA repository <https://github.com/ML-GSAI/LLaDA>, and report average NFEs per block (and overall for each generation length) to ensure a fair comparison across methods.

For the standard benchmarks GSM8K, MBPP, and HumanEval, we evaluated on the full datasets (1,319, 500, and 164 samples, respectively), with GSM8K ablations limited to the first 300 samples. To facilitate evaluation on MATH500 (Lightman et al. 2023), we implemented a new task class for the 500 cherry-picked samples from the MATH dataset (Hendrycks et al. 2021). Finally, to assess robustness on reasoning and professional-knowledge benchmarks, we sampled subsets from BBH (Suzgun et al. 2022) (20 examples from each of its 27 subgroups, 540 total) and MMLU-Pro (Wang et al. 2024) (40 examples from each of its 14 subgroups, 560 total). Datasets’ generation length G is set to 256, except the coding datasets, HumanEval and MBPP, where $G = 512$.

All evaluations use a few-shot, COT prompting framework: GSM8K and MBPP with 4-shot contexts, HumanEval with 0-shot, MATH500 with 4-shot, BBH with 3-shot, and MMLU-Pro with 5-shot.

Early stop for generation is implemented for all planners, which ends generation if [EOS] token is unmasked and all previous tokens are unmasked too, since generating text after this token will generate non-related content to the answer.

Formula	Definition
G	Generation length.
B	Block size.
k	Average number of unmasked tokens per iteration.
$n_{\text{blocks}} = \frac{G}{B}$	Number of blocks.
$\text{NFE}_{\text{block}} = \frac{B}{k}$	Number of function evaluations per block.
$\text{NFE} = n_{\text{blocks}} \cdot \text{NFE}_{\text{block}}$	Total number of function evaluations.
$\frac{\text{NFE}}{\text{NFE}_{\text{block}}} = \frac{B}{k}$	Speedup factor compared to token-by-token inference.

Table 4: Inference speedup conversion table.

Table 5: Speedup comparison on GSM8K (300 samples, $G = 256$) using DUS under an $8\times$ inference budget (total $\text{NFE} = 32$). Block sizes $B \in \{8, 16, 32\}$ correspond to average NFEs per block of $\{1, 2, 4\}$. Results for both base and instruct LLaDA models; best scores are **bold** and second-best are underlined.

Block Size	8	16	32
Avg NFEs@Block	1	2	4
Model	Score (% , \uparrow)		
Base	13.33	11.00	32.33
Instruct	12.00	<u>16.00</u>	56.67

Table 4 summarizes the notation and formulas used throughout this paper. In most of our experiments, the unmasking parameter is set as $k = \log_2 B$ unless otherwise specified.

A.2 Block-Size Effect on Speedups

In this experiment, the effect of block size B on generation accuracy was evaluated under a fixed total NFEs, corresponding to an $8\times$ speedup relative to token-by-token decoding. The DUS was configured to begin at a higher iteration $t_0 > 1$, resulting in larger unmasking group sizes k per step (cf. (6)). Block sizes $B = \{8, 16, 32\}$, corresponding to $\text{NFE}_{\text{block}} = \{1, 2, 4\}$, were tested on GSM8K (first 300 samples). Table 5 reports task accuracy (%) for LLaDA-Base and LLaDA-Instruct, revealing a monotonic increase in performance with B : the Base model rose from 13.33% at $B = 8$ to 32.33% at $B = 32$ ($\approx 2.4\times$ accuracy improvement), while the Instruct variant climbed from 12.00% to 56.67% ($\approx 4.7\times$ accuracy improvement).

These gains are attributed to the fact that larger block sizes, when combined with DUS, spread the initially predicted tokens farther apart. This spatial separation reduces MI among unmasked tokens in early iterations - consistent with our analysis in Lemma 2 - and allows subsequent itera-

tions to fill in the gaps and more effectively correct existing errors introduced by coarse-grained parallelism. Tuning B thus offers an additional lever to boost output quality without raising the compute budget, though excessively large B may force the model to predict tokens with insufficient nearby context, which can exceed the model capabilities.