TAPER: TOKEN ADMISSION WITH PRICED EARLY RELEASE FOR MASKED DIFFUSION DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Without a key–value cache, every call of a blockwise masked diffusion decoder predicts all masked positions, yet only the active block may be committed. We study TAPER (Token Admission with Priced Early Release), a training-free rule that also commits low-risk positions in a bounded window beyond the block by reusing the active block's price. Released positions never count toward the block's completion quota, and the rule adds no model call, history buffer or calibration table. An exact per-output ledger, $\mathrm{NFE} = L - E - S$, shows why early commits are not saved calls: they mostly replace later within-block batching, so the aggregate call reduction per early commit is only 0.09–0.12. On MATH500, GSM8K, HumanEval and MBPP, windows chosen on these benchmarks cut calls by 6.1% (LLaDA-8B) and 7.9% (Dream-7B) at the same price, with paired accuracy changes of 0.00 $[-0.90, +0.87]$ and $+0.28$ $[-0.37, +0.96]$ points; the reductions replicate across hardware, prices, lengths and 1,142 development problems. Controls locate the saving: with end-of-sequence tail completion applied to both policies, release still removes about 6% of calls, and at equal admission a confidence-and-history gate with AHD's released thresholds saves little because of its history condition, while confidence alone recovers most of the saving, which limits claims for the price gate itself. Release is a second knob rather than a better operating curve: a fixed price does not fix an accuracy target, we establish neither accuracy non-inferiority nor a reproducible advantage over retuning the admission price, and at length 128 accuracy point estimates fall by up to about one point.

1 INTRODUCTION

Masked diffusion language models such as LLaDA and Dream generate text by repeatedly predicting all masked positions and committing some of them (Austin et al., 2021a; Sahoo et al., 2024; Shi et al., 2024; Nie et al., 2025; Ye et al., 2025). At these scales, decoding is usually semi-autoregressive: the continuation is split into blocks that are completed left to right (Arriola et al., 2025; Nie et al., 2025). The cost of decoding is the number of denoiser calls. Without a key–value cache, each call runs the model on the whole canvas, so it returns predictions for every masked position, including those in later blocks.

A decoding policy makes two different choices. *Admission* decides which eligible positions to commit at the current call; confidence, margin, entropy and temporal-stability rules all act here (Chang et al., 2022; Kim et al., 2025a; Koh et al., 2025; Wu et al., 2026; Kim et al., 2025b; Mohamed et al., 2026; Wu & Huang, 2026; Ben-Hamu et al., 2025). *Eligibility* decides which positions may be committed at all; in blockwise decoding it is fixed by the block boundary. The boundary has a visible cost. Under admission alone in our main setting, 61% (LLaDA) and 68% (Dream) of denoiser calls commit a single token, and on 24% of calls at least one position in the next 32 already has a risk below the price the active block is paying. Those predictions are computed but excluded from commitment (Figure 1a,b).

Relaxing the boundary is not new. D2F trains a model to pipeline blocks (Wang et al., 2026), AdaBlock-dLLM moves block boundaries to semantic delimiters (Lu et al., 2026), and AHD releases future positions whose predictions have stayed consistent over a stored history (Zou et al., 2026). We study a bounded, history-free member of this family: a release rule that looks only at the current call, reuses the admission rule's own price, and cannot replace the active block's required commit. The rule

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.

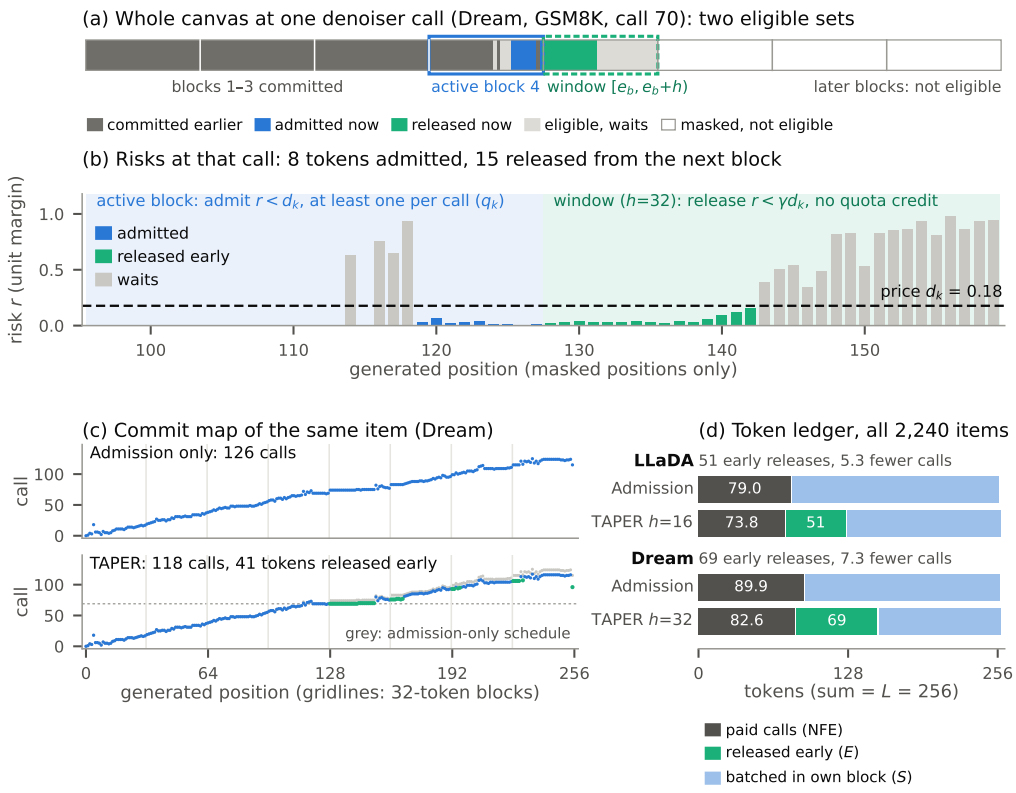

Figure 1: Selected settings on saved traces ($L = 256$, $B = 32$; LLaDA $h = 16$, Dream $h = 32$). (a) The canvas at one Dream call on a GSM8K item: only the active block (admission) and the next h positions (release) are eligible. (b) Risks of the eligible masked positions; those below the price are committed (the released tokens are answer content here). (c) Call at which each position is committed (dotted: the call of (a,b)); the item is the lowest-index GSM8K item both policies answer correctly with the median saving (8 calls). (d) Over 2,240 items, $L = \mathrm{NFE} + E + S$: 51 and 69 early releases accompany only 5.3 and 7.3 fewer calls. Positions and calls in (c) are zero-based, numbers in (a) one-based (call 70 is index 69).

is deliberately minimal: our main question, and our main contribution, is a controlled measurement of what such a rule recovers, and why.

We introduce TAPER (Token Admission with Priced Early Release) and make three contributions.

- **A minimal release rule.** Release reuses the active block's scheduled price for a bounded future window, so it adds no threshold, denoiser call, buffer or table, and a future position can never replace the active block's required commit. Writing it as the exact minimizer of a separable price objective makes the rule and its completion guarantee precise; the objective is a surrogate and carries no accuracy guarantee (Section 3). The reuse is a simplification, not a new selection principle: a fixed confidence threshold recovers most of the saving.
- **An exact token ledger and what it measures.** Every complete output satisfies $\mathrm{NFE} = L - E - S$; the aggregate call reduction per early commit is only 0.09–0.12, so counts of early commits overstate call savings about tenfold (Section 4.4).
- **Controls that locate the saving, and its limits.** At fixed prices release reduces calls across the evaluated settings, with paired intervals; the reduction survives end-of-sequence tail completion on both policies and a development evaluation. At equal admission and AHD's released thresholds, the history condition, not the confidence threshold, limits AHD-style release. At a fixed price, release keeps the admission criterion, not an accuracy target, and it establishes neither accuracy non-inferiority nor a reproducible advantage over retuning the price (Section 4).

2 PRELIMINARIES

Let $x_0 = (x_0^1, \ldots, x_0^L) \in \mathcal{V}^L$ be a clean continuation. At corruption level $t \in [0, 1]$, the absorbing-mask process is

$$q_t(x_t^i \mid x_0^i) = (1 - t)\mathbf{1}\{x_t^i = x_0^i\} + t\mathbf{1}\{x_t^i = \texttt{[MASK]}\}. \tag{1}$$

At decision index n, a denoiser call on canvas x_{t_n} and prompt c returns $p_{n,i}(v) = p_\theta(x_0^i = v \mid x_{t_n}, c)$ and proposal $a_{n,i} = \arg\max_v p_{n,i}(v)$. The corruption level t and decision index n are distinct. Starting from all masks, a policy commits the modes at a nonempty set A_n of masked positions and never remasks them. Its compute cost is the number of calls, NFE. Predictive probabilities need not be calibrated to task correctness.

Admission base. Let $\mathcal{M}_n$ be the masked positions before call n and $T_{\max}$ the total call budget. The continuation is split into L/B blocks $I_b = \{(b-1)B + 1, \ldots, bB\}$ of length B. Block b is active until all its positions are committed and has a budget of $H = T_{\max}B/L$ calls. With $C_n = I_b \cap \mathcal{M}_n$ the active masks and $k \in \{0, \ldots, H-1\}$ the call index inside the block, admission uses

$$r_{n,i} = 1 - \big(p_{n,i,(1)} - p_{n,i,(2)}\big), \qquad d_k = \frac{\sigma}{H}\Big(1 + \frac{k}{H-1}\Big), \qquad q_k = \Big\lceil \frac{|C_n|}{H-k} \Big\rceil, \tag{2}$$

where $p_{(1)} \geq p_{(2)}$ are the two largest probabilities. The risk r is the one minus the top-two margin of the current mode, d_k is a price that rises linearly within a block, and q_k guarantees completion. Admission commits every active position with $r < d_k$ and, if fewer than q_k qualify, the lowest-risk remaining ones; ties go to the smaller index. We call the result A_n^C. For the admission and release arms, $T_{\max} = L$, so $H = B$ and $q_k = 1$: each call commits at least one active token. We assume integer L/B and integer $H \geq 2$ throughout.

This admission rule is our own: a scheduled margin threshold, the margin analogue of the Fast-dLLM confidence threshold (Wu et al., 2026) with a price that rises inside each block. It is not published elsewhere, and it is the $h = 0$ member of TAPER below. We chose $\sigma = 4$ during earlier development on the same benchmarks. With $B = 32$ and $\sigma = 4$ the price rises from 0.125 to 0.25 within each block.

3 METHOD: PRICED EARLY RELEASE

Rule. Let $e_b = bB + 1$ be the first position after block b and h the window length; intervals below denote integer positions intersected with $\mathcal{M}_n$. TAPER extends eligibility to the masked positions of the next h canvas slots and prices them with the active block's price:

$$F_n = \mathcal{M}_n \cap [e_b,\, e_b + h), \qquad \mathcal{R}_n = \{\, j \in F_n : r_{n,j} < \gamma\, d_k \,\}, \qquad A_n = A_n^C \cup \mathcal{R}_n. \tag{3}$$

Here $\gamma > 0$ scales the release price. The risks of F_n come from the same call as those of C_n. Released positions do not count toward q_k, so the active block still commits at least one of its own tokens per call. We use selected windows $h = 16$ for LLaDA and $h = 32$ for Dream, chosen from $h \in \{8, 16, 32\}$ on these benchmarks: each is the window with the highest four-dataset macro accuracy of the three (Table 14), so its reported accuracy change is the most favourable of three benchmark-evaluated candidates. The common $h = 32$ control and all measured windows are reported. This is one parameterized rule, not two model-specific algorithms. We fix $\gamma = 1$: the stricter $\gamma = 1/2$, which keeps γd_k below d_0, the price at a released position's own first block call, saves fewer calls (5.9% and 5.1% against 9.4% and 7.8%) without a measurable accuracy gain (Section O), and none of three fixed release thresholds spanning the price range dominates the scheduled price in point estimates (Section N), which does not show that scheduling beats a well-chosen fixed one.

Proposition 1 (Local rule, progress and cost)**.** *Fix an observed state. (i) A_n minimizes $\sum_{i \in A \cap C_n}(r_{n,i} - d_k) + \sum_{j \in A \cap F_n}(r_{n,j} - \gamma d_k)$ over $A \subseteq C_n \cup F_n$ with $|A \cap C_n| \geq q_k$, with minimal cardinality among minimizers; $h = 0$ recovers admission. (ii) Every block finishes within H calls, including blocks partly filled by earlier releases, so decoding uses at most $T_{\max}$ calls. (iii) Computing $\mathcal{R}_n$ requires no denoiser call beyond the one that produced A_n^C.*

The proof (Section A) is elementary. The proposition makes the rule and its guarantees exact; it does not say that the objective predicts correctness, nor that a larger window lowers total calls, since every release changes all later predictions.

Algorithm 1 TAPER decoding

Require: canvas of L masks, block length B, per-block budget H, window h, price scale σ, γ

for $b = 1, \dots, L/B$ **do**
  for $k = 0, \dots, H-1$ **do**
    $C \leftarrow I_b \cap \mathcal{M}$; **if** $C = \varnothing$ **then break**
    $F \leftarrow \mathcal{M} \cap [e_b, e_b + h)$; one denoiser call gives modes and risks on $C \cup F$
    $A^C \leftarrow \{i \in C : r_i < d_k\}$, filled to q_k by lowest risk ▷ admission
    $R \leftarrow \{j \in F : r_j < \gamma d_k\}$ ▷ release; no progress credit
    commit the modes at $A^C \cup R$; $\mathcal{M} \leftarrow \mathcal{M} \setminus (A^C \cup R)$
  end for
end for

Ledger. Let u_b be the number of positions of block b committed before it becomes active, N_b its calls and $w_{b,k} = |A^C_{b,k}|$ the number of its own positions committed at its k-th call, $k = 0, \dots, N_b - 1$.

Proposition 2 (Token ledger)**.** *For a decoding path without remasking that commits each of the L positions exactly once, in which every call has one active block and commits at least one of its positions, $N_b = B - u_b - \sum_{k=0}^{N_b-1}(w_{b,k} - 1)$ and $\mathrm{NFE} = L - E - S$, with $E = \sum_b u_b$ and $S = \sum_{b,k}(w_{b,k} - 1)$.*

Corollary 3. *Relative to admission only ($E = 0$), $\Delta\mathrm{NFE} = -E - \Delta S$. For $E > 0$, with $\eta = -\Delta\mathrm{NFE}/E$ and $\rho = -\Delta S/E$, $\eta + \rho = 1$: η is the aggregate call reduction per early release and ρ the aggregate drop in within-block batching per early release.*

Both rates compare aggregate counts of two trajectories that diverge after the first release; they do not say which token caused which saving. Proposition 2 is bookkeeping, but it changes what should be measured: the number of tokens a cross-block rule decodes early, E, is not its saving. The saving is ηE, and η depends on the model and the schedule.

4 Experiments

4.1 Setup

We evaluate LLaDA-8B-Instruct and Dream-v0-Instruct-7B in bfloat16 with greedy proposals on the full MATH500, GSM8K test, HumanEval and MBPP evaluation sets (500/1,319/164/257 items) (Hendrycks et al., 2021; Lightman et al., 2024; Cobbe et al., 2021; Chen et al., 2021; Austin et al., 2021b). The main geometry is $L = T_{\max} = 256$, $B = 32$, $\sigma = 4$, without a key–value cache. Math tasks use exact-answer accuracy and code tasks pass@1, both from one project scorer. Macro averages weight the four datasets equally. We count physical denoiser calls. Every paired comparison aligns identical prompts and uses a stratified bootstrap (5,000 draws, 95% percentile intervals).

Hardware and uncertainty. Each paired comparison shares one accelerator type; baselines, transfer studies and replications also use other types with the same implementation (Table 4), and cross-cohort differences (Section B) are not a significance threshold. Intervals condition on observed runs, exclude selection uncertainty and are unadjusted for multiplicity; an interval containing zero establishes neither equivalence nor non-inferiority, and no loss margin was prespecified.

Baselines. All baselines run in our prompt, block and scoring pipeline with $B = 32$ and their default thresholds, which we did not sweep. Confidence 0.90 commits every active position with top probability at least 0.9 (at least one per call); KLASS uses the authors' dataset-specific thresholds (Kim et al., 2025b); Fast-dLLM commits the m most confident positions for the largest m with $(m+1)(1 - c_{(m)}) < 1$, $c_{(m)}$ the m-th largest confidence (Wu et al., 2026); LESS uses its block implementation for LLaDA and a block-restricted adaptation for Dream (Mohamed et al., 2026); ADAS reorders the Fast-dLLM candidates by attention to selected positions (Sahin et al., 2026), run with Instruct checkpoints. LESS and ADAS are concurrent work. These are rule-level comparisons, not reproductions of each system's caching and native geometry; further references are in Section C.

Table 1: Full-test accuracy (%) and denoiser calls at $L = T_{\max} = 256$, $B = 32$. Macro weights datasets equally. Selected windows are $h = 16$ (LLaDA) and $h = 32$ (Dream), each the most accurate of $h \in \{8, 16, 32\}$ on these benchmarks; every window is reported in Table 14. Baseline thresholds were not swept. Bold: highest accuracy and lowest calls within each model (ties included), not significance.

Method	**MATH**		**GSM8K**		**HumanEval**		**MBPP**		**Macro**	
	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓
LLaDA-8B-Instruct										
Confidence 0.90	32.00	98.81	75.44	77.38	36.59	54.01	47.47	85.63	47.87	78.96
KLASS	30.00	135.95	75.21	109.44	**38.41**	98.49	46.69	127.55	47.58	117.86
LESS	32.00	111.73	75.36	91.60	37.80	72.34	47.86	99.36	48.26	93.76
Fast-dLLM	31.80	**77.50**	**76.35**	**60.37**	34.76	**44.80**	46.69	**66.42**	47.40	**62.27**
ADAS + Fast-dLLM	32.00	79.27	75.44	61.44	36.59	45.37	**49.42**	66.70	48.36	63.19
TAPER, $h = 0$ (admission only)	34.20	91.38	75.44	76.77	37.80	55.08	47.08	81.74	**48.63**	76.24
TAPER, $h = 32$ (common window)	33.60	84.05	74.75	68.52	35.98	47.76	47.47	75.69	47.95	69.00
TAPER, $h = 16$ (ours)	**34.40**	86.21	74.83	71.04	37.80	51.48	47.47	77.73	**48.63**	71.62
Dream-v0-Instruct-7B										
Confidence 0.90	**44.40**	110.92	82.26	92.41	57.32	40.50	63.04	94.62	61.75	84.61
KLASS	42.20	175.67	**83.70**	171.36	57.93	85.26	62.26	140.40	61.52	143.17
LESS	42.40	132.63	82.03	117.08	**60.37**	66.93	61.87	113.21	61.67	107.46
Fast-dLLM	43.40	**86.78**	82.64	**71.19**	53.66	**34.29**	59.53	**74.37**	59.81	**66.65**
ADAS + Fast-dLLM	44.20	88.12	82.87	72.41	54.27	34.43	58.75	76.16	60.02	67.78
TAPER, $h = 0$ (admission only)	43.40	108.26	83.02	90.14	57.93	40.19	**63.81**	84.72	62.04	80.83
TAPER, $h = 16$	42.20	103.17	82.94	84.91	58.54	37.76	63.42	82.27	61.78	77.03
TAPER, $h = 32$ (ours)	43.60	101.59	83.32	81.90	58.54	34.52	**63.81**	79.86	**62.32**	74.47

4.2 Main results

Table 1 compares external samplers with the selected windows. LLaDA $h = 16$ reduces macro calls from 76.24 to 71.62 (-6.1%), with an accuracy change of 0.00 [-0.90, $+0.87$] points; Dream $h = 32$ reduces 80.83 to 74.47 (-7.9%), with $+0.28$ [-0.37, $+0.96$] points (Table 19). For Dream, admission at $\sigma = 4$ is itself dominated: $\sigma = 5$ reaches the same macro accuracy (62.04) with 76.52 calls, 5.3% fewer, and against it $h = 32$ uses 2.7% fewer calls at $+0.28$ points (single runs). Against KLASS, these use 39.2% and 48.0% fewer calls; against LESS, 23.6% and 30.7% fewer, much of it from admission alone. Section 4.3 asks whether release adds anything over retuning the price. Against one-token-per-call decoding (confidence $K = 1$, 256 calls, another cohort), they are within 0.71 and 0.01 points of macro accuracy with 72% and 71% fewer calls (Section C).

The common $h = 32$ LLaDA control is faster (69.00 calls) but less accurate (47.95%): its paired change is -0.68 [-1.72, $+0.33$] points. An independent replication cohort reproduces the call reductions (-6.1% LLaDA, -8.2% Dream) with accuracy changes of -0.19 [-1.05, $+0.69$] and $+0.17$ [-0.62, $+1.00$] points (Table 8). In all three hardware cohorts the common window cuts 9.4–9.6% (LLaDA) and 7.8–8.2% (Dream) of calls (Table 19). On 1,142 development problems per model, disjoint from the test sets and not used to choose TAPER's settings (some were used in earlier work on other decoding rules; Section E), the frozen settings cut calls by 6.2% (LLaDA) and 7.4% (Dream) with accuracy changes of -0.25 [-1.13, $+0.70$] and -0.19 [-1.32, $+0.95$] points (Table 11): the call reduction transfers; accuracy preservation is not shown. Fast-dLLM and ADAS use fewer calls but are less accurate by 2.5 and 2.3 points on Dream and 1.23 and 0.27 on LLaDA, operating-point differences rather than matched-compute comparisons (examples: Section Q).

4.3 Release versus retuning the price

Figure 2 shows every measured setting; external thresholds were not swept, whereas our price and windows were developed on these benchmarks. The envelope joins the non-dominated admission prices (linear between them, flat beyond); the stricter hull is the best accuracy that any mixture of measured prices reaches at the same calls, equal to the envelope at $L = 256$ near the selected windows.

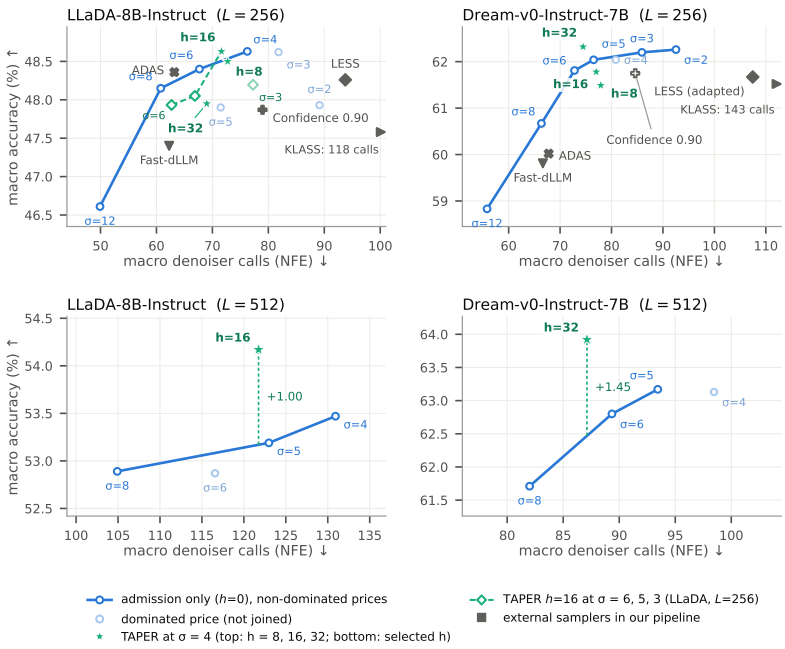

Figure 2: Accuracy against calls ($B = 32$, four-dataset macro, single runs). Top, $L = 256$: blue, admission sweeping σ (line: non-dominated prices; light circles: dominated); green stars, TAPER with $h = 8, 16, 32$; green diamonds, LLaDA $h = 16$ at other prices; grey, external samplers; open cross, a replication cohort. Bottom, $L = 512$, test benchmarks only: the dotted segment is the gap to the admission envelope, which the development evaluation does not reproduce (Table 2). Values: Tables 1, 9, 14 and 15.

Table 2: Key paired contrasts (95% intervals, single runs). $L = 512$: TAPER against the hull of the measured admission prices (joint bootstrap) and against the nearest price ($\sigma = 5$ LLaDA, $\sigma = 6$ Dream), on the test benchmarks (four tasks, or the three development tasks) and on development problems (dev., three tasks, contrasts fixed in advance). $L = 256$: end-of-sequence controls and same-base gate ablations. Full tables: Tables 9, 10, 12 and 18.

	LLaDA		Dream	
Contrast	Δ accuracy (points)	Δ calls	Δ accuracy (points)	Δ calls
vs. hull, $L = 512$, test	$+0.91$ [-0.38, $+1.80$]	matched	$+1.45$ [$+0.36$, $+2.46$]	matched
vs. hull, $L = 512$, test (3 tasks)	$+0.85$ [-0.31, $+1.67$]	matched	$+1.15$ [-0.07, $+2.14$]	matched
vs. hull, $L = 512$, dev.	$+0.24$ [-0.95, $+1.20$]	matched	$+0.01$ [-1.39, $+1.28$]	matched
vs. nearest price, $L = 512$, test	$+0.98$ [-0.19, $+2.15$]	-1.0%	$+1.12$ [$+0.06$, $+2.20$]	-2.5%
vs. nearest price, $L = 512$, dev.	$+0.45$ [-0.77, $+1.58$]	-1.2%	-0.09 [-1.48, $+1.30$]	-0.7%
tail completion on both, TAPER − admission, $L = 256$	$+0.05$ [-0.83, $+0.93$]	-6.1%	-0.06 [-0.83, $+0.70$]	-6.2%
EOS-only release − admission, $L = 256$	-0.10 [-0.25, 0.00]	-0.1%	$+0.17$ [-0.57, $+0.92$]	-2.7%
history − price gate, $L = 256$	$+0.10$ [-0.78, $+1.00$]	$+6.3\%$	-0.22 [-0.89, $+0.42$]	$+7.4\%$
confidence only − price gate, $L = 256$	-0.14 [-0.67, $+0.37$]	$+0.5\%$	-0.15 [-0.69, $+0.30$]	$+1.2\%$

Release versus price retuning at $L = 256$. Admission at $\sigma = 6$ (48.40/67.68) is more accurate than LLaDA with the common window $h = 32$ (47.95/69.00) at fewer calls, and admission at $\sigma = 8$ beats Fast-dLLM on both models at fewer calls. At matched calls the selected windows sit $+0.12$ (LLaDA) and $+0.41$ (Dream) points above the admission envelope, the other windows between -0.57 and -0.04, and ADAS at $+0.12$ and -0.90 (Gap columns of Tables 14 and 15); interpolating through the dominated $\sigma = 5$ would inflate the LLaDA gap to $+0.71$ (Section M). Release also composes with the price, removing 5.6 to 7.4% of LLaDA's calls at each of four prices (Table 13), though three of these points are dominated by admission at another price. Release is thus a second knob, not a better curve. At a fixed admission price it reduces calls while preserving the admission criterion; this does not preserve a deployment accuracy target, which must be revalidated because

Table 3: Token ledger at $L = 256$, pooled over all 2,240 items (Table 1 weights datasets equally); each output satisfies $L = \mathrm{NFE} + E + S$ (Proposition 2). $\eta = -\Delta\mathrm{NFE}/E$ and $\rho = -\Delta S/E$ are aggregate rates per early release. Terminal: share of early releases that are end-of-sequence tokens after the last content token ($h = 32$).

Model	Policy	NFE	E	S	η	ρ	Terminal
LLaDA	$h = 0$	79.01	0.00	176.99	–	–	–
	$h = 8$	75.13	32.02	148.85	0.121	0.879	–
	$h = 16$	73.76	50.66	131.58	0.104	0.896	–
	$h = 32$	71.29	66.10	118.61	0.117	0.883	14.8%
Dream	$h = 0$	89.91	0.00	166.09	–	–	–
	$h = 8$	86.33	32.46	137.21	0.110	0.890	–
	$h = 16$	85.23	54.36	116.41	0.086	0.914	–
	$h = 32$	82.59	69.16	104.25	0.106	0.894	58.3%

release changes outputs; the fixed-price contrast only isolates the effect of eligibility at an unchanged price.

Test-set and development comparisons at $L = 512$. We ran admission at $\sigma \in \{5, 6, 8\}$, fixed before these runs (the $\sigma = 4$ and TAPER runs existed; a 32-item replay reproduces both bit for bit), with the windows chosen at $L = 256$ (Table 9). On the test benchmarks no measured price dominates the selected windows, which lie $+0.91$ [-0.38, $+1.80$] (LLaDA) and $+1.45$ [$+0.36$, $+2.46$] (Dream) points above the hull in a joint paired bootstrap over all five policies. Before observing any outcome of a development evaluation on the 1,142 problems of Section 4.2, we fixed its policies and its primary contrasts, TAPER against the price of nearest test-set calls ($\sigma = 5$ LLaDA, $\sigma = 6$ Dream): they are $+0.45$ [-0.77, $+1.58$] and -0.09 [-1.48, $+1.30$] points, and the secondary hull gaps $+0.24$ [-0.95, $+1.20$] and $+0.01$ [-1.39, $+1.28$] (Tables 2 and 12). Restricted to the same three tasks, the test-set hull gaps are $+0.85$ [-0.31, $+1.67$] and $+1.15$ [-0.07, $+2.14$], so the two evaluations differ in both items and task mix. The fixed-price call reduction reproduces (-7.0% and -10.1% against $\sigma = 4$), but we do not establish a reproducible gain beyond retuning. The price sets the admission criterion; release keeps that criterion and its quota while widening eligibility, though it can still change later states and hence later admission decisions.

4.4 What release commits

Early is not saved. Table 3 decomposes each output with Proposition 2. With the selected windows, release commits 50.7 (LLaDA $h = 16$) and 69.2 (Dream $h = 32$) tokens per output before their blocks open, yet pooled calls fall by only 5.3 and 7.3 (LLaDA $h = 32$: 66.1 and 7.7). Across all windows, the saving per early release, η, is between 0.086 and 0.121, and within-block batching falls by the remaining 0.88–0.91 per early release. Treating each early commit as a saved call would therefore overestimate the observed reduction by a factor of 8 to 12 in these runs (about 9.5 for the selected windows), an empirical property of these policies rather than a general bound. The rate depends on what is released: end-of-sequence releases alone give $\eta = 0.016$ (LLaDA) and 0.057 (Dream), as admission already commits the tail in wide batches; with tail completion on both policies, the aggregate call reduction per remaining early commit is 0.11 and 0.15 (Section I). A one-block window saves 0.7 to 1.2 calls per block boundary (Section H).

The saving survives end-of-sequence handling. On Dream, 58% of early releases are end-of-sequence tokens after the last content token (LLaDA $h = 32$: 15%), so release might only finish the answer tail early. We ran online tail completion, which commits the whole remaining tail once a call proposes end-of-sequence at every masked position after the first committed one, and computed exact prefix-complete termination from saved traces (Section I). These rules alone save admission 0.8% (LLaDA) and 2.8–3.2% (Dream) of calls. With either rule applied to both policies, release still removes 6.1% [-6.6, -5.7] and 6.2–6.3% of calls, with accuracy changes under tail completion of $+0.05$ [-0.83, $+0.93$] and -0.06 [-0.83, $+0.70$] points (Table 18); releasing only end-of-sequence proposals removes 0.1% and 2.7%. These are separate contrasts, not an additive causal decomposition

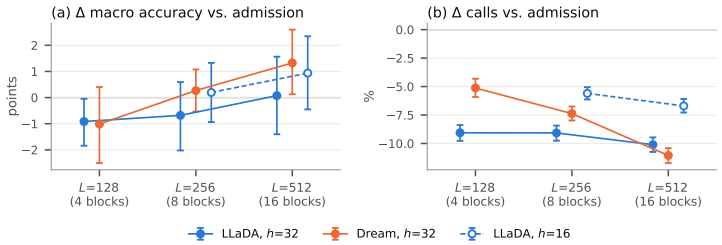

Figure 3: Release against admission only as the number of blocks grows ($B = 32$, $T_{\max} = L$, all parameters fixed). Solid lines: $h = 32$ (Dream's selected window and LLaDA's common-window control); dashed: LLaDA's selected $h = 16$. Three-dataset macro (MATH, HumanEval, MBPP); bars are paired 95% intervals. Four-dataset results at every length are in Tables 16 and 19.

of the saving: the tested end-of-sequence rules do not remove it, and about 6% of calls are still saved when both policies use either rule.

On Dream, the gate matters. A blind control releases, at each call, as many window positions as pass the priced gate, $|\mathcal{R}_n|$, but those nearest the active block, not the lowest-risk (Table 8). It loses 8.57 [-10.50, -6.74] points on Dream while saving only 3.6% more calls; on LLaDA ($h = 16$) the difference, -0.42 [-1.27, $+0.43$] points at nearly the same calls, is inconclusive.

A history gate at equal admission. To separate the gate from the admission base, we ported AHD's anchored-history gate with its released settings into our decoder, with admission, quota and window fixed (Tables 2 and 10): a window position is released if its confidence is at least 0.9 and the decay-weighted KL divergence of its current distribution from its previous five is below 0.01 (0.02 on HumanEval; AHD's released LLaDA settings, also used for Dream). On LLaDA, where the price-gate run in the same code reproduces TAPER exactly, this gate commits 5.1 tokens early per output against 50.7 and removes 0.1% of calls against 6.1%; on Dream it commits 32.2 against 69.2 and removes 1.4% against 8.2%. Six previous distributions with 0.01 throughout, or requiring both gates, change little. Keeping the confidence condition but dropping the history condition releases 49.4 and 66.5 tokens early and removes 5.6% and 7.1% of calls: under the tested admission, windows and released thresholds, the history condition, not the confidence level, limits release. This concerns AHD's released thresholds transferred to our admission and windows (Dream uses the LLaDA settings); it does not show that history is uninformative or that a tuned history gate could not match the price gate. The price gate uses 0.5% and 1.2% fewer calls than confidence alone, at accuracy changes of $+0.14$ [-0.37, $+0.67$] and $+0.15$ [-0.30, $+0.69$] points, so confidence alone recovers about 87–93% of the saving: the price gate is a simplification that reuses the admission criterion ($\gamma = 1$) instead of a separate threshold, not a new optimization principle with a distinctive accuracy advantage.

4.5 Transfer across lengths and block sizes

Figure 3 holds each window fixed and changes only the length. The absolute saving grows with the number of boundaries; in percent (three-dataset macro) it rises on Dream from 5.1% to 7.4% to 11.1% at $L = 128, 256, 512$ and is nearly flat on LLaDA (9.1%, 9.1%, 10.1%). Point estimates of the accuracy change rise with length ($h = 32$, three datasets: LLaDA -0.92 [-1.84, -0.05], -0.68, $+0.07$; Dream -1.01 [-2.51, $+0.40$], $+0.27$, $+1.33$; with GSM8K at $L = 128$, -0.53 and -0.40), but the intervals overlap and we did not test for a trend.

Changing B with $T_{\max} = L$ also rescales the price ($d_k \propto \sigma/B$): the selected windows save 2.2–2.4% of calls with $B = 64$ and 18.9–19.3% with $B = 16$, where Dream loses 1.04 points (single-run differences; Table 17 and Section G).

4.6 RUNTIME AND PLACEMENT

Wall-clock time follows calls without a cache. In repeated RTX4090 timing on the first 64 items of each dataset, the selected windows cut decoder time by 6.9% (LLaDA) and 8.8% (Dream), matching their call reductions, at unchanged peak memory; in the same cache-free $B = 32$ pipeline TAPER takes 43% and 29% less time than KLASS and LESS on LLaDA and 52% and 31% less on Dream (Table 20). Each policy runs at its own accuracy, so this is not an accuracy-matched speed comparison. A table-based placement reranker lowered accuracy at $L = 512$ by about 2.5 points (Section J), so TAPER has no table.

5 RELATED WORK AND DISCUSSION

Unmasking policies. Training-free decoders rank or threshold masked positions by confidence, margin, entropy or temporal stability (Chang et al., 2022; Kim et al., 2025a; Koh et al., 2025; Wu et al., 2026; Ben-Hamu et al., 2025; Kim et al., 2025b; Mohamed et al., 2026; Wu & Huang, 2026). Others add future-aware or dependency-aware signals such as information gain, lookahead or conditional-independence tests (Yang et al., 2026; Lee et al., 2026; Azangulov et al., 2026; Ringel et al., 2026), rerank candidates by attention (Sahin et al., 2026), learn the policy (Jazbec et al., 2026; Chen et al., 2026) or consult an auxiliary autoregressive model (Israel et al., 2025). All of them decide which eligible positions to commit; release changes which positions are eligible and keeps the admission rule, so it could wrap any admission rule that commits within an active block, although we test it only with ours. We train no denoiser (foundations: Section P).

Closest block-relaxation work. Within block diffusion (Arriola et al., 2025), D2F trains pipelined block decoding (Wang et al., 2026), AdaBlock-dLLM places block boundaries at semantic delimiters (Lu et al., 2026) and AHD releases future positions with a stable history (Zou et al., 2026). Early cross-block commitment is not new; we contribute a minimal current-call instance that reuses the admission price and keeps a separate active quota, an exact ledger, and controls that locate its saving. We did not evaluate AdaBlock-dLLM; more related work is in Section P.

Limitations. Prices, γ and windows were selected on the test benchmarks; the development problems share their task families and were partly used in earlier work on other decoding rules; intervals are unadjusted for selection and multiplicity, and external baselines kept their default thresholds. We evaluate two checkpoints, greedy decoding and four reasoning/code tasks, and release only on our admission with fixed windows; other admission rules, adaptive windows and stochastic decoding are untested. Release can lower accuracy at $L = 128$ with the common window and for Dream with $B = 16$; the gate ablations use AHD's released thresholds, not an optimized history-gate frontier. With a key–value cache, window predictions may come from stale states and calls differ in cost, so the ledger would count calls but not time. Equivalence and non-inferiority are not established.

6 CONCLUSION

TAPER extends blockwise eligibility to a bounded future window, using predictions already computed by the current call and the active block's admission price. For the evaluated models and tasks, our measurements support four conclusions. (i) At a fixed price, release reduces calls with small changes in accuracy point estimates, and the reduction survives end-of-sequence handling on both policies and a development evaluation. (ii) Early commits are not saved calls: they largely replace later within-block batching, so early-decoding methods should report early commits and call reductions together with their ratio η. (iii) Under the tested admission, windows and released thresholds, the history condition of an AHD-style gate, not its confidence threshold, limits release, and a current-call gate recovers most of the saving, so the price gate is a simplification. (iv) Release does not establish accuracy non-inferiority or a reproducible advantage over retuning the admission price, so cross-block rules should also be compared with a retuned admission sweep and an end-of-sequence control.

AI USE STATEMENT

Language model assistants were used for literature retrieval, code generation and debugging, analysis scripts and manuscript editing, at the author's direction. The author ran all experiments and takes full responsibility for the proofs, implementations, reported numbers, citations and claims.

ETHICS STATEMENT

This work studies inference for existing models and introduces no new model or human-subject data. Lower decoding cost does not mitigate hallucination, bias, unsafe code or other deployment risks. Model and dataset licenses apply.

REPRODUCIBILITY STATEMENT

Sections 2 and 3 and Algorithm 1 specify the rule, including tie handling; Section B lists the evaluation protocol. Source code, run configurations and per-item outputs will be released, with the scripts that recompute every paired interval, hull comparison, ledger value and end-of-sequence control in the paper from those outputs. The supplementary material lists the development-split record identifiers with their prior-use flags, the frozen evaluation plans with their hashes, and the pre-specified primary contrasts.

REFERENCES

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

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, pp. 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. URL https://arxiv.org/abs/2108.07732.

Iskander Azangulov, Teodora Pandeva, Niranjani Prasad, Javier Zazo, and Sushrut Karmalkar. Parallel sampling from masked diffusion models via conditional independence testing. In *International Conference on Learning Representations*, 2026.

Parikshit Bansal and Sujay Sanghavi. Enabling approximate joint sampling in diffusion LMs. *arXiv preprint arXiv:2509.22738*, 2025.

Richard Bellman. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60(6):503–515, 1954.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. In *Advances in Neural Information Processing Systems*, volume 38, pp. 55981–56007, 2025.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, 35:28266–28279, 2022.

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. MaskGIT: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11315–11325, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Chang_MaskGIT_Masked_Generative_Image_Transformer_CVPR_2022_paper.html.

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. URL https://arxiv.org/abs/2107.03374.

Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. dParallel: Learnable parallel decoding for dLLMs. In *International Conference on Learning Representations*, 2026.

Yuan Shih Chow, Herbert Robbins, and David Siegmund. *Great Expectations: The Theory of Optimal Stopping*. Houghton Mifflin, 1971.

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.

Justin Deschenaux and Caglar Gulcehre. Beyond autoregression: Fast LLMs via self-distillation through time. In *International Conference on Learning Representations*, 2025.

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 *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

Chunsan Hong, Seonho An, Min-Soo Kim, and Jong Chul Ye. Improving discrete diffusion unmasking policies beyond explicit reference policies. In *International Conference on Learning Representations*, 2026.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In *Advances in Neural Information Processing Systems*, volume 34, pp. 12454–12465, 2021.

Pengcheng Huang, Tianming Liu, Zhenghao Liu, Yukun Yan, Shuo Wang, Tong Xiao, Zulong Chen, and Maosong Sun. Empirical analysis of decoding biases in masked diffusion models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6853–6876, 2026. URL https://aclanthology.org/2026.acl-long.311/.

Daniel Mingyi Israel, Guy Van den Broeck, and Aditya Grover. Accelerating diffusion LLMs via adaptive parallel decoding. In *Advances in Neural Information Processing Systems*, 2025.

Metod Jazbec, Theo X. Olausson, Louis Béthune, Pierre Ablin, Michael Kirchhof, João Monteiro, Victor Turrisi, Jason Ramapuram, and Marco Cuturi. Learning unmasking policies for diffusion language models. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2512.09106.

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 *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 30749–30768, 2025a. URL https://proceedings.mlr.press/v267/kim25ah.html.

Seo Hyun Kim, Sunwoo Hong, Hojung Jung, Youngrok Park, and Se-Young Yun. KLASS: KL-guided fast inference in masked diffusion models. In *Advances in Neural Information Processing Systems*, 2025b. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/8514c1f12dcdd061ac8521feb78f28ca-Paper-Conference.pdf.

Hyukhun Koh, Minha Jhang, Dohyung Kim, Sangmook Lee, and Kyomin Jung. Conditional [MASK] discrete diffusion language model. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 8899–8923, 2025. URL https://aclanthology.org/2025.emnlp-main.450/.

Sanghyun Lee, Seungryong Kim, Jongho Park, and Dongmin Park. Lookahead unmasking elicits reliable decoding in diffusion language models. In *International Conference on Machine Learning*, 2026. URL https://icml.cc/virtual/2026/poster/64337.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=CNicRIVIPA.

Guanxi Lu, Hao Mark Chen, Yuto Karashima, et al. AdaBlock-dLLM: Semantic-aware diffusion LLM inference via adaptive block size. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2509.26432.

Ziwei Luo, Ziqi Jin, Lei Wang, Lidong Bing, and Thomas B. Schön. Self-rewarding sequential monte carlo for masked diffusion language models, 2026. URL https://arxiv.org/abs/2602.01849.

Amr Mohamed, Guokan Shang, and Michalis Vazirgiannis. LESS is more: Mutual-stability sampling for diffusion language models. *arXiv preprint arXiv:2606.16908*, 2026. URL https://arxiv.org/abs/2606.16908.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 50608–50646, 2025.

Hunter Nisonoff, Junhao Xiong, Stephan Allenspach, and Jennifer Listgarten. Unlocking guidance for discrete state-space diffusion and flow models. In *International Conference on Learning Representations*, 2025.

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

Yong-Hyun Park, Chieh-Hsin Lai, Satoshi Hayakawa, Yuhta Takida, and Yuki Mitsufuji. Jump your steps: Optimizing sampling schedule of discrete diffusion models. In *International Conference on Learning Representations*, 2025.

Dhruvesh Patel, Tahira Naseem, Gaurav Pandey, Md Arafat Sultan, Andrew McCallum, and Ramón Fernandez Astudillo. Improved sampling from masked diffusion models with position contrastive guidance. In *NeurIPS 2025 Workshop on Structured Probabilistic Inference & Generative Modeling*, 2025.

Goran Peskir and Albert Shiryaev. *Optimal Stopping and Free-Boundary Problems*. Birkhäuser, Basel, 2006.

Jarrid Rector-Brooks, Mohsin Hasan, Zhangzhi Peng, Zachary Quinn, Chenghao Liu, Sarthak Mittal, Nouha Dziri, Michael Bronstein, Yoshua Bengio, Pranam Chatterjee, Alexander Tong, and Avishek Joey Bose. Steering masked discrete diffusion models via discrete denoising posterior prediction. In *International Conference on Learning Representations*, 2025.

Yinuo Ren, Haoxuan Chen, Yuchen Zhu, Wei Guo, Yongxin Chen, Grant M Rotskoff, Molei Tao, and Lexing Ying. Fast solvers for discrete diffusion models: Theory and applications of high-order algorithms. In *Advances in Neural Information Processing Systems*, 2025.

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

Yusuf Sahin, Ahmed Rockey Saikia, Volkan Cevher, and Paolo Favaro. Attention-discounted adaptive sampler for masked diffusion language models. *arXiv preprint arXiv:2606.10829*, 2026. URL https://arxiv.org/abs/2606.10829.

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

Shreshth Saini, Avinab Saha, Balu Adsumilli, Neil Birkbeck, Yilin Wang, and Alan C. Bovik. TABES: Trajectory-aware backward-on-entropy steering for masked diffusion models, 2026. URL https://arxiv.org/abs/2602.00250.

Yair Schiff, Subham Sekhar Sahoo, Hao Phung, Guanghan Wang, Sam Boshar, Hugo Dalla-torre, Bernardo P de Almeida, Alexander Rush, Thomas Pierrot, and Volodymyr Kuleshov. Simple guidance mechanisms for discrete diffusion models. In *International Conference on Learning Representations*, 2025.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis Titsias. Simplified and generalized masked diffusion for discrete data. *Advances in Neural Information Processing Systems*, 37: 103131–103167, 2024.

Albert N. Shiryaev. *Optimal Stopping Rules*, volume 8. Springer Science & Business Media, 2007.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Xu Wang, Chenkai Xu, Yijie Jin, Jiachun Jin, Hao Zhang, Kai Yu, and Zhijie Deng. Diffusion LLMs can do faster-than-AR inference via discrete diffusion forcing. In *International Conference on Learning Representations*, 2026.

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

Yue Wu and Jian Huang. Stability-weighted decoding for diffusion language models. *arXiv preprint arXiv:2604.17068*, 2026. URL https://arxiv.org/abs/2604.17068.

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

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

Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. In *International Conference on Learning Representations*, 2025.

Shun Zou, Yong Wang, Zehui Chen, Lin Chen, Chongyang Tao, Feng Zhao, and Xiangxiang Chu. Breaking block boundaries: Anchor-based history-stable decoding for diffusion large language models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026. URL https://arxiv.org/abs/2604.08964.

A PROOFS

Proposition 1. (i) The feasible sets factor into subsets of C_n with at least q_k elements and arbitrary subsets of F_n; the two are disjoint because F_n starts at e_b, the first position after the active block. On F_n the objective is minimized with minimal cardinality by taking exactly the negative coefficients $r_{n,j} - \gamma d_k$, which is $\mathcal{R}_n$. On C_n every negative coefficient must be included; if fewer than q_k are negative, adding the smallest remaining coefficients minimizes the sum, since exchanging a selected larger coefficient for an unselected smaller one does not increase it (feasibility holds because $q_k \le |C_n|$). Ties follow the stated order. With $h = 0$, $F_n = \varnothing$. (ii) Let m_k be the active masks before local call k. The call commits at least $q_k = \lceil m_k/(H-k) \rceil$ active tokens, so $m_{k+1} \le m_k - q_k$, and at $k = H - 1$ all remaining masks are committed. Releases only reduce m_0 of later blocks and never contribute to q_k. Summing H over blocks gives $T_{\max}$. (iii) $C_n \cup F_n$ are rows of one full-canvas forward pass; this holds for bidirectional denoisers that score the whole canvas in one call. □

Proposition 2 and Corollary 3. Each of the L positions is committed exactly once. Of the B positions of block b, u_b are committed before activation and the remaining ones over its N_b calls, so $\sum_k w_{b,k} = B - u_b$; subtracting $N_b = \sum_k 1$ gives the block identity, and summing over blocks gives $\mathrm{NFE} = L - E - S$. For admission only $E = 0$, hence $\Delta\mathrm{NFE} = -E - \Delta S$; when the release path has $E > 0$, division gives $\eta + \rho = 1$. The identity requires a complete path in which each call belongs to one active block and commits at least one of its positions; a comparator with auxiliary calls would add them to NFE separately, and a path stopped early leaves masked positions and is reported by its calls only (Section I). □

B PROTOCOL AND HARDWARE PROVENANCE

Protocol. Models load once per worker. Current and future positions use the same normalised predictive distribution with the mask token excluded; Dream reads position-shifted logits as its implementation requires. Proposals are argmax with deterministic position tie-breaking, so the nominal seed does not create independent trials. At $L \in \{128, 512\}$ and $B \in \{16, 64\}$ we keep $T_{\max} = L$, $\sigma = 4$ and $\gamma = 1$; the price therefore scales as σ/B. Configurations, prompt hashes and scorer versions are stored with every result; paired comparisons verify prompt and input identity per item.

Table 4: Hardware provenance, consolidated for reproducibility. Accuracy and call comparisons are paired or controlled within a hardware cohort unless explicitly described as cross-cohort.

Experiment group	Accelerator
Primary results, price/window sweeps, and most ablations	H200
Fixed-width top-K baselines	RTX4090
Dream Confidence 0.90 baseline	RTX3090
Independent controls and replication cohort 1	RTX4090
Release-price strictness, $B = 64$ transfer, replication cohort 2	RTX3090
Development validation ($L = 256$)	LLaDA: RTX3090; Dream: RTX4090
Same-base history-gate ablation	LLaDA: H200; Dream: RTX4090
Same-base confidence-only gate	LLaDA: H200; Dream: RTX4090 and H200
Repeated wall-clock timing	RTX4090

Cross-GPU variation. Deterministic decoding is not bitwise stable across GPU types, because floating-point reductions differ and near-tied decisions flip. We took 22 (model, policy) configurations that ran with identical code, prompts and scorer on at least two of H200, RTX4090 and RTX3090. For each run we computed its four-dataset macro accuracy minus the linear interpolation of the admission price sweep measured on the same card (at least three prices per card), at the run's call count. Across cards, this residual differs by a median of 1.10 points (maximum 2.36). Macro calls of identical configurations differ across cards by a median of 0.33% (maximum 0.74%, 32 configurations). These descriptive cross-hardware residuals also depend on interpolation of each card's sweep. They are not standard errors, confidence intervals, or a one-point noise floor. Within-hardware paired uncertainty must be assessed separately; a sub-one-point difference need not be noise, and a larger difference need not be significant.

C ADDITIONAL BASELINES

Tables 5 and 6 report fixed-width and other references, and Table 7 reports LESS with its native geometry.

Table 5: LLaDA: additional references ($L = 256$, $B = 32$). Fixed-width rows commit K tokens per call. ADAS, concurrent work, runs in our pipeline with Instruct checkpoints: ADAS + Fast-dLLM is its adaptive and ADAS ($K = 2$) its fixed-width configuration; Fast-dLLM++ is a rule-level port. Bold: column extrema, not significance.

Method	**MATH**		**GSM8K**		**HumanEval**		**MBPP**		**Macro**	
	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓
Fixed width										
Confidence ($K = 2$)	31.60	128.00	72.86	128.00	32.93	128.00	37.35	128.00	43.68	128.00
Margin ($K = 2$)	31.00	128.00	73.31	128.00	29.27	128.00	38.91	128.00	43.12	128.00
Entropy ($K = 2$)	31.20	128.00	72.48	128.00	25.61	128.00	40.08	128.00	42.34	128.00
Confidence ($K = 1$)	34.20	256.00	**76.35**	256.00	37.80	256.00	49.03	256.00	49.34	256.00
Margin ($K = 1$)	33.40	256.00	73.69	256.00	37.20	256.00	45.91	256.00	47.55	256.00
Entropy ($K = 1$)	**35.60**	256.00	74.15	256.00	**39.63**	256.00	**50.19**	256.00	**49.89**	256.00
Other references										
KLASS	30.00	135.95	75.21	109.44	38.41	98.49	46.69	127.55	47.58	117.86
Fast-dLLM++ (rule port)	30.40	**61.18**	73.16	**44.32**	35.37	**37.58**	43.97	**50.71**	45.72	**48.45**
ADAS + Fast-dLLM	32.00	79.27	75.44	61.44	36.59	45.37	49.42	66.70	48.36	63.19
ADAS ($K = 2$)	29.40	128.00	73.62	128.00	30.49	128.00	46.30	128.00	44.95	128.00
TAPER, $h = 16$ (ours)	34.40	86.21	74.83	71.04	37.80	51.48	47.47	77.73	48.63	71.62

Table 6: Dream: additional references ($L = 256$, $B = 32$). Fixed-width rows commit K tokens per call. ADAS, concurrent work, runs in our pipeline with Instruct checkpoints: ADAS + Fast-dLLM is its adaptive and ADAS ($K = 2$) its fixed-width configuration; Fast-dLLM++ is a rule-level port. Bold: column extrema, not significance.

Method	**MATH**		**GSM8K**		**HumanEval**		**MBPP**		**Macro**	
	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓
Fixed width										
Confidence ($K = 2$)	39.00	128.00	79.98	128.00	45.73	128.00	49.03	128.00	53.44	128.00
Margin ($K = 2$)	41.00	128.00	80.74	128.00	49.39	128.00	52.53	128.00	55.92	128.00
Entropy ($K = 2$)	38.80	128.00	78.85	128.00	42.68	128.00	51.36	128.00	52.92	128.00
Confidence ($K = 1$)	42.60	256.00	83.32	256.00	**60.37**	256.00	63.04	256.00	62.33	256.00
Margin ($K = 1$)	44.00	256.00	82.56	256.00	57.32	256.00	**64.59**	256.00	62.12	256.00
Entropy ($K = 1$)	**45.00**	256.00	83.24	256.00	59.15	256.00	63.81	256.00	**62.80**	256.00
Other references										
KLASS	42.20	175.67	**83.70**	171.36	57.93	85.26	62.26	140.40	61.52	143.17
Fast-dLLM++ (rule port)	41.00	**69.46**	80.82	**55.05**	47.56	**27.14**	60.31	**63.23**	57.42	**53.72**
ADAS + Fast-dLLM	44.20	88.12	82.87	72.41	54.27	34.43	58.75	76.16	60.02	67.78
ADAS ($K = 2$)	37.60	128.00	80.06	128.00	51.22	128.00	55.64	128.00	56.13	128.00
TAPER, $h = 32$ (ours)	43.60	101.59	83.32	81.90	58.54	34.52	63.81	79.86	62.32	74.47

Table 8: Independent controls ($L = 256$, $B = 32$, four-dataset macro; E: early releases per output; pairing as in Table 19). *: one card; unmarked rows: a second card of the same type (both in replication cohort 1 of Table 4); †paired across the two cards. Blind controls (Section 4.4) are paired with TAPER on their own card. The AHD row ports AHD's gate (six previous distributions, KL < 0.01; Table 10) with its confidence-0.9 base; it is not the upstream system.

Model	Policy	Acc	Calls	E	Peak mem.	Paired change
LLaDA	Admission only*	47.92	76.08	0.0	15.4 GiB	reference
	AHD port*	48.28	78.52	11.0	18.1 GiB	$+0.37$ [-1.60, $+2.21$] pt, $+3.2\%$ calls
	TAPER $h = 32$*	47.44	68.80	66.5	15.4 GiB	-0.48 [-1.42, $+0.44$] pt, -9.6% calls
	TAPER $h = 16$	47.72	71.42	51.0	15.3 GiB	-0.19 [-1.05, $+0.69$] pt, -6.1% calls†
	Blind release, $h = 16$	47.31	71.33	50.9	15.3 GiB	-0.42 [-1.27, $+0.43$] pt vs. TAPER $h = 16$
Dream	Admission only	61.73	80.85	0.0	14.8 GiB	reference
	TAPER $h = 32$*	61.90	74.25	69.2	14.8 GiB	$+0.17$ [-0.62, $+1.00$] pt, -8.2% calls†
	Blind release, $h = 32$*	53.33	71.60	72.1	14.8 GiB	-8.57 [-10.50, -6.74] pt vs. TAPER $h = 32$

Table 7: LESS with its native block geometry and a fixed 256-token generation length in our evaluator. LLaDA uses $B = 64$; Dream is global ($B = 256$). These are context results, not matched-B32 comparisons or a reproduction of the complete published evaluation protocol.

Model	**MATH**		**GSM8K**		**HumanEval**		**MBPP**		**Macro**	
	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓
LLaDA	30.00	103.81	75.59	82.07	36.59	61.76	47.08	91.54	47.31	84.79
Dream	41.80	123.16	81.80	106.58	59.15	50.20	61.87	101.03	61.15	95.24

Single-position decoding ($K = 1$) is a high-compute quality reference rather than a matched-compute competitor. Relative to Confidence $K = 1$, TAPER changes macro accuracy by -0.71 points on LLaDA and -0.01 points on Dream while using 72.0% and 70.9% fewer model calls, respectively. Relative to Entropy $K = 1$, the corresponding accuracy differences are -1.26 and -0.48 points. Because the fixed-width and adaptive results come from different hardware cohorts, these are descriptive point estimates rather than paired evidence. More generally, fixed-width references show the quality available at 128 or 256 calls and are not matched-compute competitors to adaptive policies. ADAS + Fast-dLLM has macro 48.36/63.19 on LLaDA and 60.02/67.78 on Dream; the selected TAPER settings have 48.63/71.62 and 62.32/74.47. These comparisons trade accuracy against calls and do not establish equivalence. The Fast-dLLM++ port is a different speed–quality point; its call count is outside the measured admission sweep, so matched-call interpolation would require extrapolation.

D INDEPENDENT CONTROLS

E PRICE SWEEP AT LENGTH 512, GATE ABLATION AND VALIDATION ON DEVELOPMENT PROBLEMS

Table 9: Admission price sweep at $L = 512$, $B = 32$ (four-dataset macro) against TAPER with the selected windows. $\sigma = 4$ and TAPER are saved runs, which a 32-item replay with the current decoder reproduces bit for bit; $\sigma \in \{5, 6, 8\}$ was fixed before these three runs and run with the same admission rule. Gap: TAPER minus the envelope (non-dominated polyline) and minus the upper hull of the measured prices at matched calls; the hull interval resamples items jointly for all five policies (Section E). A development evaluation with prospectively specified contrasts is in Table 12. The windows are those selected at $L = 256$. [d]: dominated by another admission price. Paired changes are TAPER minus admission, with the protocol of Table 19. Single runs; intervals are unadjusted.

Model	Policy	Acc / Calls	Δ accuracy (points)	Δ calls (%)
LLaDA	TAPER ($h = 16$)	54.17 / 121.75	reference	reference
	admission $\sigma = 4$	53.47 / 130.92	$+0.70$ [-0.36, $+1.75$]	-7.0 [-7.5, -6.5]
	admission $\sigma = 5$	53.19 / 122.98	$+0.98$ [-0.19, $+2.15$]	-1.0 [-1.6, -0.4]
	admission $\sigma = 6$[d]	52.87 / 116.56	$+1.30$ [$+0.12$, $+2.42$]	$+4.5$ [$+3.8$, $+5.1$]
	admission $\sigma = 8$	52.89 / 104.91	$+1.28$ [-0.16, $+2.73$]	$+16.1$ [$+15.1$, $+17.0$]
	gap (envelope; hull)	$+1.00$; $+0.91$ [-0.38, $+1.80$] points		
Dream	TAPER ($h = 32$)	63.92 / 87.12	reference	reference
	admission $\sigma = 4$[d]	63.13 / 98.46	$+0.79$ [-0.13, $+1.71$]	-11.5 [-12.0, -11.0]
	admission $\sigma = 5$	63.17 / 93.44	$+0.74$ [-0.22, $+1.68$]	-6.8 [-7.3, -6.2]
	admission $\sigma = 6$	62.80 / 89.35	$+1.12$ [$+0.06$, $+2.20$]	-2.5 [-3.1, -1.9]
	admission $\sigma = 8$	61.71 / 82.00	$+2.20$ [$+0.84$, $+3.60$]	$+6.2$ [$+5.4$, $+7.1$]
	gap (envelope; hull)	$+1.45$; $+1.45$ [$+0.36$, $+2.46$] points		

Table 10: Same-base release-gate ablation ($L = 256$, $B = 32$; LLaDA $h = 16$, Dream $h = 32$): admission, quota and window are fixed and only the release gate changes. *History* ports AHD's anchored-history rule (confidence ≥ 0.9 and a decay-weighted KL divergence below a threshold): *released* uses AHD's released LLaDA settings, also for Dream; *six previous* compares six distributions with KL < 0.01; *confidence only* drops the history condition. E: early commits per output (pooled). Paired changes are against the price gate (protocol of Table 19); details in Section E.

Model	Release gate	Acc / Calls	E	Δ accuracy (points)	Δ calls (%)
LLaDA	Admission only	48.63 / 76.24	0.0	–	–
	price gate (TAPER)	48.63 / 71.62	50.7	reference	reference
	confidence only	48.49 / 71.95	49.4	-0.14 [-0.67, $+0.37$]	$+0.5$ [$+0.1$, $+0.7$]
	history, released	48.73 / 76.15	5.1	$+0.10$ [-0.78, $+1.00$]	$+6.3$ [$+5.9$, $+6.8$]
	history, six previous	48.73 / 76.17	4.7	$+0.10$ [-0.78, $+1.00$]	$+6.4$ [$+5.9$, $+6.8$]
	price ∧ history (six)	48.73 / 76.17	4.7	$+0.10$ [-0.78, $+1.00$]	$+6.4$ [$+5.9$, $+6.8$]
Dream	Admission only	61.73 / 80.85	0.0	–	–
	price gate (TAPER)	61.90 / 74.25	69.2	reference	reference
	confidence only	61.75 / 75.11	66.5	-0.15 [-0.69, $+0.30$]	$+1.2$ [$+0.8$, $+1.5$]
	history, released	61.68 / 79.74	32.2	-0.22 [-0.89, $+0.42$]	$+7.4$ [$+6.9$, $+8.0$]
	history, six previous	61.61 / 79.93	30.4	-0.29 [-0.97, $+0.36$]	$+7.6$ [$+7.1$, $+8.2$]

Table 11: Validation of the frozen settings of Table 1 ($L = 256$, $B = 32$, $\sigma = 4$, LLaDA $h = 16$, Dream $h = 32$) on development problems disjoint from the test sets: 384 MATH problems from the MATH training split, 384 GSM8K training problems and 374 MBPP problems outside the MBPP test set, none used to choose TAPER's settings; parts were used in earlier work on other decoding rules (Section E). Both policies of each dataset run within the same hardware cohort; exact provenance is in Table 4. Accuracy (%) / calls; the macro weights the three datasets equally. Paired change: TAPER minus admission (stratified bootstrap, 5,000 draws).

Model	Policy	MATH	GSM8K	MBPP	Macro (3)
LLaDA	Admission only	37.24 / 92.86	82.29 / 75.16	43.32 / 86.16	54.28 / 84.73
	TAPER	35.94 / 87.37	82.03 / 69.40	44.12 / 81.62	54.03 / 79.46
	paired change: -0.25 [-1.13, $+0.70$] points, -6.2% [-6.7, -5.7] calls				
Dream	Admission only	45.31 / 107.04	85.42 / 87.51	56.95 / 90.02	62.56 / 94.86
	TAPER	46.61 / 99.97	84.90 / 79.15	55.61 / 84.46	62.38 / 87.86
	paired change: -0.19 [-1.32, $+0.95$] points, -7.4% [-7.9, -6.9] calls				

Table 12: Development evaluation with prospectively specified contrasts at $L = 512$ ($B = 32$) on the 1,142 development problems of Table 11: TAPER with the selected windows against admission at the prices of Table 9, three-dataset macro. Its policies and primary contrasts (TAPER against the price of nearest test-set calls: $\sigma = 5$ LLaDA, $\sigma = 6$ Dream) were specified before any outcome; the hull gap is a secondary analysis. Paired changes are TAPER minus admission (stratified bootstrap, 5,000 draws); the hull gap uses the joint bootstrap described below.

Model	Policy	Acc / Calls	Δ accuracy (points)	Δ calls (%)
LLaDA	TAPER ($h = 16$)	57.33 / 121.68	reference	reference
	admission $\sigma = 4$	57.58 / 130.89	-0.25 [-1.30, $+0.80$]	-7.0 [-7.5, -6.6]
	admission $\sigma = 5$ (primary)	56.88 / 123.19	$+0.45$ [-0.77, $+1.58$]	-1.2 [-1.8, -0.7]
	admission $\sigma = 6$	56.80 / 116.21	$+0.53$ [-0.77, $+1.83$]	$+4.7$ [$+4.1$, $+5.3$]
	admission $\sigma = 8$	56.01 / 104.79	$+1.32$ [-0.18, $+2.81$]	$+16.1$ [$+15.3$, $+16.9$]
	gap (envelope; hull)	$+0.46$; $+0.24$ [-0.95, $+1.20$] points		
Dream	TAPER ($h = 32$)	65.09 / 103.48	reference	reference
	admission $\sigma = 4$	65.70 / 115.10	-0.61 [-1.84, $+0.59$]	-10.1 [-10.7, -9.5]
	admission $\sigma = 5$	65.53 / 109.36	-0.44 [-1.76, $+0.87$]	-5.4 [-6.0, -4.7]
	admission $\sigma = 6$ (primary)	65.18 / 104.17	-0.09 [-1.48, $+1.30$]	-0.7 [-1.4, $+0.0$]
	admission $\sigma = 8$	63.96 / 95.70	$+1.13$ [-0.46, $+2.71$]	$+8.1$ [$+7.2$, $+9.0$]
	gap (envelope; hull)	$+0.01$; $+0.01$ [-1.39, $+1.28$] points		

The validation problems are the development partitions of our evaluation pipeline: 384 MATH problems from the MATH training split, 384 GSM8K training problems and 374 MBPP problems, none of which overlaps the test sets by record identifier or content hash, and none of which was used to choose TAPER's window, price or γ. They are not an untouched holdout: 288 of the MBPP problems were marked as seen in our exposure registry and 112 of them calibrated other decoding rules; 100 problems of each dataset were in an earlier screening run of other rules, including a variant of the admission rule; and training-split problems may have been seen by the models. This leaves the pairing of the two policies on identical inputs intact but limits how independent the problems are of our wider development. Without MBPP the call reductions are 6.7% (LLaDA) and 7.9% (Dream). HumanEval has no held-out split. The $L = 512$ development evaluation (Table 12) uses the same problems and the windows, prices, decoder and scorer of Table 9. Its policies and primary contrasts were specified, and its frozen plans hashed, before any of its outcomes; the joint-bootstrap hull comparison is a secondary analysis. Its first 32 outputs replayed test items and reproduced the saved $L = 512$ rows bit for bit.

The hull in Tables 2, 9 and 12 is the upper concave hull of the measured admission prices: an item-level mixture of two prices has linearly mixed expected accuracy and calls, so the hull is the best expected accuracy reachable at TAPER's calls. Its interval comes from a paired bootstrap that resamples the same items for all five policies (5,000 draws, seed 3535, stratified by dataset) and recomputes the hull in each draw; it conditions on the measured policies and ignores the selection of

the windows. At $L = 512$ the hull exceeds the envelope only for LLaDA, where $\sigma = 5$ lies below the chord to $\sigma = 4$ (from $\sigma = 8$ on the test benchmarks, from $\sigma = 6$ on the development problems).

Gate ablation (Table 10). The history rule uses decay 0.5 and AHD's released LLaDA settings (five previous distributions, KL < 0.01, < 0.02 on HumanEval), also for Dream, for which AHD gives no setting; it is a component ablation, not an official AHD run. On LLaDA the price-gate run in the ablation code reproduces the saved TAPER outputs exactly (2,240 of 2,240); on Dream the price row is the saved replication-cohort TAPER run of Table 8. Every item on which the released gate commits nothing early reproduces the saved admission output exactly (1,712 of 1,712 LLaDA and 16 of 16 Dream items), and each confidence-only worker first replayed price-gate items and reproduced the saved TAPER rows of its card type (20 of 20). Peak memory is 15.4 and 14.8 GiB for TAPER's own implementation (the admission-only level); the ablation code computes posteriors for every masked position and peaks at 16.0 and 15.4 GiB for confidence only, 18.0 and 17.8 GiB for released and 18.2 and 18.1 GiB for six previous (LLaDA, Dream), so the history itself costs about 2 GiB. A second Dream confidence-only run in the main cohort, paired with the TAPER run of Table 1, gives 61.99 / 75.21 (E 66.8), -0.33 [-0.82, $+0.10$] points and $+1.0\%$ [$+0.6$, $+1.4$] calls against the price gate.

F WINDOW AND PRICE SWEEPS

Table 13: Release on top of each admission price (LLaDA, $h = 16$, $L = 256$, $B = 32$, four-dataset macro): release minus admission at the same σ, paired with the protocol of Table 19. Release removes calls at every price; no accuracy change is distinguishable from zero. The admission runs at $\sigma = 3, 5, 6$ come from earlier runs of the same admission rule.

σ	Admission	Release ($h = 16$)	Δ accuracy (points)	Δ calls (%)
6	48.40 / 67.68	47.93 / 62.70	-0.47 [-1.46, $+0.52$]	-7.4 [-7.9, -6.8]
5	47.90 / 71.48	48.05 / 66.87	$+0.15$ [-0.79, $+1.07$]	-6.4 [-6.9, -6.0]
4	48.63 / 76.24	48.63 / 71.62	0.00 [-0.90, $+0.87$]	-6.1 [-6.5, -5.6]
3	48.62 / 81.82	48.20 / 77.26	-0.42 [-1.26, $+0.36$]	-5.6 [-6.0, -5.2]

The following tables report every measured window and price setting in the primary-cohort sweep.

Table 14: Release-window sweep at $L = 256$, $B = 32$, $\sigma = 4$. Full-test accuracy (%) and calls. $h = 0$ is admission only. Gap: macro accuracy minus the envelope of non-dominated admission prices at the same calls (Table 15; single runs, descriptive). Bold marks the highest accuracy and lowest calls within each model, including ties; it implies no significance.

Window	**MATH**		**GSM8K**		**HumanEval**		**MBPP**		**Macro**		Gap
	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	
LLaDA											
$h = 0$	34.20	91.38	**75.44**	76.77	**37.80**	55.08	47.08	81.74	**48.63**	76.24	–
$h = 8$	34.20	87.58	75.13	72.60	37.20	52.17	**47.47**	78.54	48.50	72.72	-0.04
$h = 16$	**34.40**	86.21	74.83	71.04	**37.80**	51.48	**47.47**	77.73	**48.63**	71.62	$+0.12$
$h = 32$	33.60	**84.05**	74.75	**68.52**	35.98	**47.76**	**47.47**	**75.69**	47.95	**69.00**	-0.49
Dream											
$h = 0$	43.40	108.26	83.02	90.14	57.93	40.19	**63.81**	84.72	62.04	80.83	–
$h = 8$	42.80	104.31	82.79	86.18	57.32	38.37	63.04	82.76	61.49	77.91	-0.57
$h = 16$	42.20	103.17	82.94	84.91	**58.54**	37.76	63.42	82.27	61.78	77.03	-0.27
$h = 32$	**43.60**	**101.59**	**83.32**	**81.90**	**58.54**	**34.52**	**63.81**	**79.86**	**62.32**	**74.47**	$+0.41$

Table 15: Admission-price sweeps at $L = 256$, $B = 32$: four-dataset macro. All measured prices are listed. Gap: accuracy minus the envelope of non-dominated admission settings at the same calls (linear between adjacent settings, flat beyond the last; descriptive only). Bold marks highest accuracy and lowest calls within each model/window group, not a jointly optimal setting.

Model	Window	σ	Acc↑	Calls↓	Gap
LLaDA	$h = 0$	12	46.61	**49.91**	–
LLaDA	$h = 0$	8	48.15	60.77	–
LLaDA	$h = 0$	6	48.40	67.68	–
LLaDA	$h = 0$	5	47.90	71.48	–
LLaDA	$h = 0$	4	**48.63**	76.24	–
LLaDA	$h = 0$	3	48.62	81.82	–
LLaDA	$h = 0$	2	47.93	89.14	–
Dream	$h = 0$	12	58.83	**55.80**	–
Dream	$h = 0$	8	60.67	66.37	–
Dream	$h = 0$	6	61.81	72.80	–
Dream	$h = 0$	5	62.04	76.52	–
Dream	$h = 0$	4	62.04	80.83	–
Dream	$h = 0$	3	62.20	85.91	–
Dream	$h = 0$	2	**62.26**	92.53	–
LLaDA	$h = 16$	6	47.93	**62.70**	-0.29
LLaDA	$h = 16$	5	48.05	66.87	-0.32
LLaDA	$h = 16$	4	**48.63**	71.62	$+0.12$
LLaDA	$h = 16$	3	48.20	77.26	-0.43
External samplers (thresholds not swept; Table 1)					
LLaDA	Confidence 0.90	–	47.87	78.96	-0.76
LLaDA	KLASS	–	47.58	117.86	-1.05
LLaDA	LESS	–	48.26	93.76	-0.37
LLaDA	Fast-dLLM	–	47.40	62.27	-0.80
LLaDA	ADAS + Fast-dLLM	–	48.36	63.19	$+0.12$
Dream	Confidence 0.90	–	61.75	84.61	-0.43
Dream	KLASS	–	61.52	143.17	-0.74
Dream	LESS	–	61.67	107.46	-0.59
Dream	Fast-dLLM	–	59.81	66.65	-0.91
Dream	ADAS + Fast-dLLM	–	60.02	67.78	-0.90

G TRANSFER ACROSS LENGTHS AND BLOCK SIZES

Length and block-size comparisons are grouped by backbone and geometry; missing cells remain unfilled.

Table 16: Length transfer under a shared execution environment with $B = 32$, $T_{\max} = L$, fixed price and window per row family. Macro (3) excludes GSM8K; Macro (4) includes all four datasets. GSM8K at $L = 128$ and for LLaDA $h = 32$ at $L = 512$ ran later with the same decoder, which reproduced saved rows bit for bit. Bold extrema are within each model/length group.

Model	L	Window	MATH		GSM8K		HumanEval		MBPP		Macro (3)		Macro (4)	
			Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓
LLaDA	128	$h = 0$	**26.80**	49.33	71.19	47.17	**30.49**	32.48	**45.53**	39.33	**34.27**	40.38	**43.50**	42.08
LLaDA	128	$h = 32$	26.00	**46.23**	**71.80**	**41.98**	**30.49**	**28.16**	43.58	**35.77**	33.36	**36.72**	42.97	**38.04**
LLaDA	256	$h = 0$	34.20	91.38	**75.44**	76.77	**37.80**	55.08	47.08	81.74	39.70	76.06	**48.63**	76.24
LLaDA	256	$h = 16$	**34.40**	86.21	74.83	71.04	**37.80**	51.48	**47.47**	77.73	**39.89**	71.81	**48.63**	71.62
LLaDA	256	$h = 32$	33.60	**84.05**	74.75	**68.52**	35.98	**47.76**	**47.47**	**75.69**	39.02	**69.16**	47.95	**69.00**
LLaDA	512	$h = 0$	36.80	151.39	**82.26**	103.56	46.95	129.53	**47.86**	139.22	43.87	140.05	53.47	130.92
LLaDA	512	$h = 16$	**39.00**	140.06	**82.26**	95.01	**47.56**	120.89	**47.86**	131.04	**44.81**	130.66	**54.17**	121.75
LLaDA	512	$h = 32$	37.80	**136.04**	82.18	**89.19**	46.95	**114.99**	47.08	**126.65**	43.94	**125.89**	53.50	**116.72**
Dream	128	$h = 0$	37.00	64.39	66.41	58.85	**50.61**	29.95	**64.98**	70.51	**50.86**	54.95	**54.75**	55.92
Dream	128	$h = 32$	**37.40**	**61.18**	**67.85**	**54.86**	47.56	**26.45**	64.59	**68.76**	49.85	**52.13**	54.35	**52.81**
Dream	256	$h = 0$	43.40	108.26	83.02	90.14	57.93	40.19	**63.81**	84.72	55.05	77.72	62.04	80.83
Dream	256	$h = 32$	**43.60**	**101.59**	**83.32**	**81.90**	**58.54**	**34.52**	**63.81**	**79.86**	**55.32**	**71.99**	**62.32**	**74.47**
Dream	512	$h = 0$	47.00	152.05	**85.90**	100.71	58.54	50.13	61.09	90.95	55.54	97.71	63.13	98.46
Dream	512	$h = 32$	**49.20**	**138.67**	85.06	**87.72**	**59.15**	**40.62**	**62.26**	**81.46**	**56.87**	**86.91**	**63.92**	**87.12**

Table 17: Block-size transfer at $L = 256$, $T_{\max} = 256$, $\sigma = 4$, parameters fixed. Rows within each model and block-size group share an execution environment; exact provenance is in Table 4. With $B = 16$, $h = 16$ is one block and $h = 32$ spans two blocks. Differences between rows are single-run point differences without paired intervals. Bold marks the highest accuracy and lowest calls within each model and geometry, including ties; it implies no significance.

Policy	**MATH**		**GSM8K**		**HumanEval**		**MBPP**		**Macro**	
	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓
$B = 64$, *LLaDA*										
$h = 0$	31.40	103.79	74.75	89.13	37.20	60.73	**45.91**	93.73	47.31	86.85
$h = 16$	30.60	101.58	**75.66**	86.92	37.80	58.62	**45.91**	91.98	47.49	84.78
$h = 64\ (= B)$	**32.20**	**100.05**	**75.66**	**85.12**	**39.02**	**56.55**	45.53	**90.38**	**48.10**	**83.02**
$B = 64$, *Dream*										
$h = 0$	**44.60**	125.36	81.12	106.52	48.17	46.38	**62.26**	96.39	59.04	93.66
$h = 32$	43.40	122.75	81.20	103.20	**49.39**	45.49	**62.26**	94.89	**59.06**	91.58
$h = 64\ (= B)$	44.40	**121.85**	**81.35**	**99.66**	45.12	**43.75**	61.87	**94.43**	58.18	**89.92**
$B = 16$, *LLaDA*										
$h = 0$	**33.00**	81.07	74.53	67.27	**37.80**	50.96	46.30	71.68	**47.91**	67.75
$h = 16\ (= B)$	32.60	**67.03**	**74.60**	**53.20**	36.59	**40.46**	**46.69**	**59.07**	47.62	**54.94**
$B = 16$, *Dream*										
$h = 0$	**40.60**	94.94	**82.49**	78.97	**54.88**	39.60	**62.65**	81.60	**60.15**	73.78
$h = 32\ (= 2B)$	40.40	**80.31**	81.12	**63.44**	53.05	**26.57**	61.87	**67.76**	59.11	**59.52**

In an independent $L = 128$ cohort with $h = 16$ (four datasets), release changes macro accuracy by $+0.24$ points (43.90 vs. 43.66) with 4.6% fewer calls on LLaDA, and by -0.31 points (54.43 vs. 54.74) with 3.8% fewer calls on Dream; the AHD port reaches 43.14 at 42.07 calls on LLaDA. In a second $L = 128$ cohort with $h = 32$ (four datasets), Dream changes by -0.61 points (53.79 vs. 54.40) with 6.1% fewer calls.

H MECHANISM DETAILS

Per block boundary (three-dataset macro call saving of the one-block window divided by the $L/B-1$ boundaries), release saves 1.22, 0.99 and 0.94 calls on LLaDA and 0.94, 0.82 and 0.72 on Dream at $L = 128, 256, 512$ (Table 16). Over the admission-only trajectories at $L = 256$, 61.4% (LLaDA) and 68.0% (Dream) of calls commit exactly one token. On 24.5% and 23.9% of calls at least one of the next 32 positions has risk below the current price, and 7.0% and 10.1% of all observed future-window positions do. With $h = 32$, early releases per output and the share that are end-of-sequence tokens after the last content token are: LLaDA MATH 57.5 (24.9%), GSM8K 64.4 (10.4%), HumanEval 93.9 (16.5%), MBPP 73.9 (18.1%); Dream MATH 49.5 (33.3%), GSM8K 70.2 (56.6%), HumanEval 106.2 (78.4%), MBPP 78.5 (79.6%).

To measure the tail's share of the call saving, we call a block a tail block if all its positions are end-of-sequence tokens after the last content token of that trajectory's output. Pooling MATH, HumanEval and MBPP with $h = 32$, tail blocks account for 13.6%, 5.5% and 9.5% of LLaDA's call saving at $L = 128, 256, 512$, and for 8.0%, 19.7% and 34.3% of Dream's. Admission alone spends 0.3, 2.2 and 7.8 calls per output on Dream's tail blocks at these lengths. Tail membership is recomputed separately from each generated output and can differ between trajectories. The reported shares are differences of those assigned counts, not effects of an intervention; interventions that change only end-of-sequence handling are in Section I.

I END-OF-SEQUENCE CONTROLS

Table 18: End-of-sequence controls ($L = 256$, $B = 32$, main cohort; LLaDA $h = 16$, Dream $h = 32$). Accuracy (%) / calls per dataset and four-dataset macro; E: early releases per output (pooled); Fill: positions committed per output by tail completion. Termination rows are computed exactly from the saved per-call traces of the first two rows and leave accuracy unchanged; they stop before every position is committed, so only calls are shown. Rules and paired intervals: Section I.

Model	Policy	MATH	GSM8K	HumanEval	MBPP	Macro	E	Fill
LLaDA	Admission only	34.20 / 91.38	75.44 / 76.77	37.80 / 55.08	47.08 / 81.74	48.63 / 76.24	0.0	–
	TAPER	34.40 / 86.21	74.83 / 71.04	37.80 / 51.48	47.47 / 77.73	48.63 / 71.62	50.7	–
	Admission + tail completion	34.40 / 90.65	75.44 / 76.61	37.80 / 54.16	47.08 / 80.97	48.68 / 75.60	0.0	11.5
	TAPER + tail completion	34.80 / 85.47	74.83 / 70.85	37.80 / 50.63	47.47 / 76.89	48.73 / 70.96	46.4	10.8
	EOS-only release	33.80 / 91.26	75.44 / 76.66	37.80 / 55.00	47.08 / 81.75	48.53 / 76.17	5.8	–
	Admission + termination	– / 90.72	– / 76.63	– / 54.12	– / 81.13	48.63 / 75.65	–	–
	TAPER + termination	– / 85.53	– / 70.87	– / 50.55	– / 77.13	48.63 / 71.02	–	–
Dream	Admission only	43.40 / 108.26	83.02 / 90.14	57.93 / 40.19	63.81 / 84.72	62.04 / 80.83	0.0	–
	TAPER	43.60 / 101.59	83.32 / 81.90	58.54 / 34.52	63.81 / 79.86	62.32 / 74.47	69.2	–
	Admission + tail completion	43.80 / 107.63	82.94 / 88.72	57.93 / 36.01	63.81 / 81.80	62.12 / 78.54	0.0	57.0
	TAPER + tail completion	43.80 / 101.52	83.32 / 81.84	57.32 / 32.68	63.81 / 78.55	62.06 / 73.65	40.9	52.2
	EOS-only release	44.60 / 107.47	82.49 / 87.19	57.93 / 37.12	63.81 / 82.87	62.21 / 78.66	41.3	–
	Admission + termination	– / 107.63	– / 88.65	– / 35.66	– / 81.11	62.04 / 78.26	–	–
	TAPER + termination	– / 101.36	– / 81.29	– / 32.44	– / 78.21	62.32 / 73.32	–	–

All arms run in the main cohort with the decoder of Table 1; with both options off it reproduces the saved admission and TAPER rows bit for bit (2 items in each of 16 model, policy and dataset cells). *Tail completion* adds no call and uses no oracle: after the commits of a call, let $p^\star$ be the first committed end-of-sequence position; if the same call proposes an end-of-sequence token at every still-masked position after $p^\star$, those proposals are committed, and blocks with no masked position cost no call. It can change later predictions of positions before $p^\star$, and hence accuracy. *Prefix-complete termination* stops after the first call at which some end-of-sequence token is committed and every position before the first one is committed; since committed tokens are never remasked, the output up to its first end-of-sequence token equals the saved output, and it is evaluated exactly on the saved per-call traces (one LLaDA HumanEval output has text after its first end-of-sequence token; its scored code is unchanged). Terminated paths leave positions masked, so Proposition 2 does not apply to them and Table 18 reports their calls only. *EOS-only release* keeps TAPER's price and window but releases only window positions whose current proposal is an end-of-sequence token.

Paired changes (accuracy in points, calls in %, protocol of Table 19), LLaDA then Dream: tail completion on admission, $+0.05$ [0.00, $+0.15$] and -0.8 [-1.0, -0.7]; $+0.08$ [-0.36, $+0.54$] and -2.8 [-3.0, -2.6]. TAPER with tail completion against admission with it, $+0.05$ [-0.83, $+0.93$] and -6.1 [-6.6, -5.7]; -0.06 [-0.83, $+0.70$] and -6.2 [-6.7, -5.8]. TAPER without it against admission with it, -0.05 [-0.97, $+0.83$] and -5.3 [-5.7, -4.8]; $+0.20$ [-0.58, $+1.00$] and -5.2 [-5.6, -4.7]. EOS-only release against admission, -0.10 [-0.25, 0.00] and -0.09 [-0.14, -0.05]; $+0.17$ [-0.57, $+0.92$] and -2.7 [-2.9, -2.4]. Termination on admission, -0.8 [-0.9, -0.7] and -3.2 [-3.3, -3.0] calls; TAPER against admission with termination on both, -6.1 [-6.6, -5.7] and -6.3 [-6.8, -5.8] calls, at unchanged accuracy.

The tail is cheap. On Dream tail completion fills 57 tail positions per output yet saves 1.6 pooled calls, because admission commits the tail in wide batches; releasing only end-of-sequence proposals commits 41 positions early per output and saves 2.3 pooled calls ($\eta = 0.057$; LLaDA 5.8 and 0.09, $\eta = 0.016$). With tail completion on both policies, TAPER's remaining releases (46.4 and 40.9 per output) save 5.3 and 6.0 pooled calls, $\eta = 0.114$ and 0.148, above the rates of Table 3. These rates compare aggregate counts of diverging trajectories, like those of Corollary 3.

J PLACEMENT RERANKING

Let i_0 be the single token chosen by admission and P_W the W lowest-risk active positions. Placement replaces i_0 by $j^\star = \arg\max_{j \in \mathcal{J}_n} \sum_{i \in C_n \setminus \{j\}} \widehat{K}(r_i, |i-j|)$ over $\mathcal{J}_n = \{j \in P_W : \widehat{\phi}(r_j) \leq \widehat{\phi}(r_{i_0}) + \delta\}$, where $\widehat{\phi}$ and $\widehat{K}$ are displacement and receiver-change tables estimated offline from reference trajectories and $\delta = 0.02$. The random control draws j uniformly from $\mathcal{J}_n$; the exclusive control disables placement at calls where release acts. Added with $W = 5$ to the common $h = 32$ release control at $L = 256$, placement is cheaper on both models, nominally more accurate on LLaDA and less accurate on Dream. Four-dataset macro accuracy/calls at $L = 256$ (primary cohort, $h = 32$): LLaDA admission 48.63/76.24, placement W5 48.46/72.48, release 47.95/69.00, placement W5 + release 48.82/65.40, placement W3 + release 48.75/65.88, random W3 + release 48.46/67.63, exclusive W3 + release 48.65/65.96; Dream admission 62.04/80.83, placement W5 61.86/76.65, release 62.32/74.47, placement W5 + release 61.41/70.10, placement W3 + release 61.92/70.38, random W3 + release 63.04/72.20, exclusive W3 + release 61.50/70.52. At $L = 512$ (three datasets), placement W3 on top of release changes accuracy by -2.73 [-4.99, -0.42] (LLaDA) and -2.46 [-4.65, -0.37] (Dream) points and calls by a further 5.2% and 5.1%.

K PAIRED UNCERTAINTY AND OUTPUT CHANGES

Table 19: Release versus admission only ($h = 0$), paired on identical prompts. Stratified bootstrap within datasets (5,000 draws, seed 3535), 95% percentile intervals of the change in macro accuracy (points) and macro calls (%). "Sets" is the number of datasets in the macro; three-dataset rows (without GSM8K) match Figure 3. Replication cohorts are named in the Setting column; exact hardware provenance is in Table 4; †paired with an admission run on a second card of the same type (Table 8). Intervals are unadjusted for multiplicity and for choosing windows on these benchmarks; they do not establish non-inferiority.

Model	Setting	Sets	Δ accuracy (points)	Δ calls (%)
LLaDA	$L = 256$, $h = 16$	4	0.00 [-0.90, $+0.87$]	-6.1 [-6.5, -5.6]
LLaDA	$L = 256$, $h = 32$	4	-0.68 [-1.72, $+0.33$]	-9.5 [-10.0, -9.0]
LLaDA	$L = 256$, $h = 8$	4	-0.13 [-0.96, $+0.66$]	-4.6 [-5.0, -4.2]
Dream	$L = 256$, $h = 32$	4	$+0.28$ [-0.37, $+0.96$]	-7.9 [-8.3, -7.4]
Dream	$L = 256$, $h = 16$	4	-0.26 [-0.87, $+0.36$]	-4.7 [-5.1, -4.3]
Dream	$L = 256$, $h = 8$	4	-0.55 [-1.20, $+0.03$]	-3.6 [-3.9, -3.3]
LLaDA	$L = 256$, $h = 16$, replication 1†	4	-0.19 [-1.05, $+0.69$]	-6.1 [-6.6, -5.7]
LLaDA	$L = 256$, $h = 32$, replication 1	4	-0.48 [-1.42, $+0.44$]	-9.6 [-10.1, -9.0]
Dream	$L = 256$, $h = 32$, replication 1†	4	$+0.17$ [-0.62, $+1.00$]	-8.2 [-8.6, -7.7]
LLaDA	$L = 256$, $h = 32$, replication 2	4	-0.05 [-0.97, $+0.85$]	-9.4 [-9.9, -8.9]
Dream	$L = 256$, $h = 32$, replication 2	4	$+0.34$ [-0.44, $+1.15$]	-7.8 [-8.2, -7.4]
LLaDA	$L = 128$, $h = 32$	3	-0.92 [-1.84, -0.05]	-9.1 [-9.8, -8.4]
LLaDA	$L = 128$, $h = 32$	4	-0.53 [-1.29, $+0.20$]	-9.6 [-10.1, -9.1]
Dream	$L = 128$, $h = 32$	3	-1.01 [-2.51, $+0.40$]	-5.1 [-5.9, -4.3]
Dream	$L = 128$, $h = 32$	4	-0.40 [-1.61, $+0.74$]	-5.6 [-6.2, -5.0]
LLaDA	$L = 512$, $h = 32$	3	$+0.07$ [-1.40, $+1.56$]	-10.1 [-10.7, -9.5]
LLaDA	$L = 512$, $h = 32$	4	$+0.04$ [-1.14, $+1.14$]	-10.9 [-11.4, -10.3]
LLaDA	$L = 512$, $h = 16$	4	$+0.70$ [-0.36, $+1.75$]	-7.0 [-7.5, -6.5]
Dream	$L = 512$, $h = 32$	4	$+0.79$ [-0.13, $+1.71$]	-11.5 [-12.0, -11.0]

Pooling items, release with the selected windows changes correctness on 80 of 2,240 LLaDA items (37 improved, 43 harmed) and 64 Dream items (35 improved, 29 harmed); the common LLaDA window $h = 32$ improves 45 and harms 59. Release is not pathwise monotone: with the selected windows, calls increase on 120 of 2,240 items for each model. Outputs change at the token level on 37% of LLaDA items ($h = 16$; 51% with $h = 32$) and 50% of Dream items ($h = 32$), so token changes are far more common than correctness changes. Intervals in this appendix condition on saved trajectories and do not account for the extensive benchmark-guided search. Containing zero is not a non-inferiority or equivalence test.

L RUNTIME

Table 20: Repeated decoder timing on RTX4090 at $L = 256$, $B = 32$: all policies run in the same pipeline without a key–value cache, each at its own operating point, so this is not an accuracy-matched speed comparison and does not reproduce the baselines' native caching. First 64 items of each dataset, two unrecorded warm-ups per task, model loading and scoring excluded. Seconds per output and calls are four-dataset means over three complete repeats per model; the repeats agree within 0.2%. Δ time is relative to admission only.

Model	Policy	Seconds / output	Calls	Δ time
LLaDA	Admission only	3.807	75.35	reference
	TAPER (selected h)	3.545	70.18	-6.9%
	KLASS	6.240	117.07	$+63.9\%$
	LESS	4.961	94.19	$+30.3\%$
Dream	Admission only	4.032	80.80	reference
	TAPER (selected h)	3.676	73.66	-8.8%
	KLASS	7.744	143.80	$+92.1\%$
	LESS	5.312	107.50	$+31.7\%$

The repeated timing in Table 20 runs the four policies in alternating order, with two unrecorded warm-ups per task; on both models the three complete repeats agree to within 0.2%. Earlier timing covered the decoder loop including trace preparation, on the same card and items. Single runs give 3.808 to 3.419 seconds per output for LLaDA with $h = 32$ (calls 75.35 to 67.68) and 4.030 to 3.671 seconds for Dream with $h = 32$ (calls 80.80 to 73.66). LLaDA with $h = 16$ against admission, in three alternating repeats after warm-up, gives 3.82 to 3.56 seconds (calls 75.35 to 70.18). Peak allocated memory is 15.3 GiB (LLaDA) and 14.7 GiB (Dream) with and without release.

M INTERPRETING OPERATING-POINT CURVES

The admission line in Figure 2 joins only settings that no other measured price dominates; dominated settings are drawn but not joined. A line through every setting in price order would pass through dominated points and is not a frontier. At LLaDA's selected 71.62 calls, the nondominated admission neighbors are approximately (67.68,48.40) and (76.24,48.63), yielding 48.51% under linear interpolation. The difference from 48.63% is approximately 0.12 points. Interpolation through the dominated (71.48,47.90) setting instead yields a 0.71-point difference. An independent randomized mixture of two policies has linearly mixed expected accuracy and calls, but these segments are neither evaluated deterministic policies nor confidence bands. Formal frontier comparisons would require uncertainty for paired policy contrasts and a treatment of selection. Neither these interpolations nor cross-GPU variation establish frontier equality. The envelope is formed from single runs and keeps the favourable ones, while the selected windows were also chosen on these benchmarks; both sides carry selection effects of unknown net sign.

N FIXED RELEASE THRESHOLDS

Table 21: Scheduled release price against fixed release thresholds spanning its range ($L = 256$, $B = 32$, four-dataset macro). Admission, quota and window are unchanged (LLaDA $h = 16$, Dream $h = 32$); only the release threshold differs. Paired changes are TAPER minus the fixed threshold, with the protocol of Table 19; intervals are unadjusted and do not establish superiority.

Model	Release threshold	Acc / Calls	Δ accuracy (points)	Δ calls (%)
LLaDA	scheduled d_k (TAPER)	48.63 / 71.62	reference	reference
	fixed 0.125	48.45 / 72.38	$+0.18$ [-0.19, $+0.62$]	-1.1 [-1.4, -0.7]
	fixed 0.1875	48.09 / 70.87	$+0.54$ [$+0.03$, $+1.10$]	$+1.1$ [$+0.9$, $+1.2$]
	fixed 0.25	47.80 / 69.15	$+0.83$ [$+0.08$, $+1.66$]	$+3.6$ [$+3.1$, $+4.1$]
Dream	scheduled d_k (TAPER)	62.32 / 74.47	reference	reference
	fixed 0.125	61.97 / 75.55	$+0.35$ [0.00, $+0.71$]	-1.4 [-1.7, -1.1]
	fixed 0.1875	62.19 / 74.12	$+0.13$ [-0.31, $+0.63$]	$+0.5$ [$+0.2$, $+0.7$]
	fixed 0.25	61.76 / 72.24	$+0.56$ [-0.22, $+1.36$]	$+3.1$ [$+2.6$, $+3.6$]

Table 21 keeps admission, quota and window and replaces only the release price γd_k by a fixed threshold. Against the lowest threshold, 0.125 (the price at a block's first call), TAPER is more accurate and uses fewer calls on both models in point estimates; the call reductions are resolved (1.1% and 1.4%), the accuracy changes are not ($+0.18$ [-0.19, $+0.62$] and $+0.35$ [0.00, $+0.71$] points). Against 0.1875 and 0.25 it is more accurate but uses 0.5–3.6% more calls, a trade-off rather than dominance. None of the three fixed thresholds dominates the scheduled price in point estimates, so the scheduled price is a non-dominated operating point among those tested; the data do not show that the schedule is better than a well-chosen fixed threshold.

O STRICTER RELEASE PRICE

Table 22: Release-price strictness in an independent cohort ($h = 32$, $L = T_{\max} = 256$, $B = 32$, $\sigma = 4$): accuracy (%) / calls with all three policies on the same card type. GSM8K admission comes from a separate run in this cohort that reproduces this cohort's admission outputs exactly on the other three datasets. Paired changes are in Section O. This cohort is separate from the primary cohort of Table 1.

Model	Policy	MATH		GSM8K		HumanEval		MBPP		Macro	
		Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓	Acc↑	Calls↓
LLaDA	Admission ($h = 0$)	32.40	91.06	74.83	76.59	39.02	54.28	46.30	82.56	48.14	76.12
	Release, $\gamma = 1$	32.60	83.56	75.44	68.41	38.41	47.43	45.91	76.41	48.09	68.95
	Release, $\gamma = 1/2$	31.60	86.65	75.13	71.99	38.41	49.32	46.69	78.51	47.96	71.62
Dream	Admission ($h = 0$)	44.20	109.42	83.02	90.38	58.54	39.62	64.59	84.93	62.59	81.09
	Release, $\gamma = 1$	44.80	102.25	83.17	82.20	59.15	34.27	64.59	80.23	62.93	74.73
	Release, $\gamma = 1/2$	43.80	105.21	83.24	85.39	59.15	35.39	64.20	81.67	62.60	76.91

For $\gamma = 1/2$, the release price never exceeds d_0, the price a released position would face at its own block's first call. This is an exact price comparison, not a guarantee about correctness or future context. In the independent cohort of Table 22, paired on identical prompts with the protocol of Table 19, $\gamma = 1$ changes accuracy by -0.05 [-0.97, $+0.85$] (LLaDA) and $+0.34$ [-0.44, $+1.15$] (Dream) points at -9.4% and -7.8% calls, and $\gamma = 1/2$ by -0.18 [-1.08, $+0.74$] and $+0.01$ [-0.78, $+0.81$] points at -5.9% and -5.1% calls. Against $\gamma = 1$, $\gamma = 1/2$ changes accuracy by -0.13 [-0.87, $+0.61$] and -0.33 [-0.97, $+0.28$] points while using 3.9% and 2.9% more calls. The stricter price therefore removes part of the saving without a measurable accuracy gain. The cohort and accelerator differ from the primary study, so we make no cross-card comparison.

P ADDITIONAL METHODOLOGICAL CONTEXT

Masked diffusion language models build on discrete diffusion (Sohl-Dickstein et al., 2015; Hoogeboom et al., 2021; Campbell et al., 2022; Lou et al., 2024) and simplified masked objectives (Ou et al., 2025; Zheng et al., 2025). Global sequential decision problems can be formulated using dynamic programming and optimal stopping (Bellman, 1954; Chow et al., 1971; Peskir & Shiryaev, 2006; Shiryaev, 2007). With task target z, the objective $\mathbb{E}[\ell(\widehat{x}_0^\pi, z) + \lambda \mathrm{NFE}_\pi]$ charges once per shared denoiser call. Its continuation values are unavailable at inference; the separable objective of Proposition 1 is a specified local surrogate, not a derived exact Bellman policy or Bayes error guarantee.

Guidance and particle methods change other components of inference, including discrete guidance (Schiff et al., 2025; Nisonoff et al., 2025), posterior steering (Rector-Brooks et al., 2025), entropy-based steering of the unmasking trajectory (Saini et al., 2026), sequential Monte Carlo (Luo et al., 2026), and approximate joint sampling (Bansal & Sanghavi, 2025). Position-contrastive guidance (Patel et al., 2025), improved unmasking-policy objectives (Hong et al., 2026), and empirical decoding-bias analysis (Huang et al., 2026) offer further perspectives on prediction and selection. Schedule optimization, higher-order solvers and temporal self-distillation address complementary compute costs (Park et al., 2025; Ren et al., 2025; Deschenaux & Gulcehre, 2025). These are relevant context, not methods for which this paper claims matched empirical superiority.

Q SELECTED GENERATION EXAMPLES

We show four selected cases in which TAPER passes the existing evaluator and both KLASS and LESS fail. These are illustrative positive cases selected after inspecting the outputs, not a representative sample or evidence that early release caused a particular reasoning improvement. All three methods use identical tokenized prompts for each case, the same backbone, and $L = 256$, $B = 32$. KLASS uses its published thresholds in the shared evaluation pipeline; LESS uses the B32 implementation, including the stated Dream adaptation. TAPER uses the selected windows, $h = 16$ for LLaDA and $h = 32$ for Dream. Each panel reports physical NFE. Only marked excerpts are shortened; their contents are verbatim. Full outputs, record identifiers and full-test discordance counts will be released with the code.

The reverse outcome also occurs. For the four displayed model–dataset pairs, the numbers of examples correct only under TAPER versus correct only under KLASS are $54/32$ (LLaDA MATH), $48/41$ (Dream MATH), $14/12$ (LLaDA MBPP), and $8/7$ (Dream HumanEval). Against LESS, the corresponding counts are $44/32$, $56/50$, $13/14$, and $14/17$. These full-test counts use the saved evaluator labels; the selected examples additionally underwent manual inspection.

Selecting the larger root. LLaDA / MATH / `124`

Prompt: If $|x+5|-|3x-6| = 0$, find the largest possible value of x. Express your answer as an improper fraction.

All three responses obtain the two roots $11/2$ and $1/4$ before the displayed final selection.

KLASS
Incorrect 132 NFE
Verbatim excerpt

The largest possible value of x is $\frac{1}{4}$.

LESS
Incorrect 97 NFE
Verbatim excerpt

The largest possible value of x is $\boxed{\frac{1}{4}}$.

TAPER (ours)
Correct 79 NFE
Verbatim excerpt

The largest possible value of x is the solution from Case 1.

The final answer is $\boxed{\frac{11}{2}}$.

Figure 4: LLaDA on MATH, record 124. Both baselines find $11/2$ and $1/4$ in the omitted preceding calculations, then select the smaller value. TAPER selects the requested larger root with 79 evaluations. Panels quote the end of each saved response; they do not attribute this difference to a particular token commitment.

Counting the overlap once. Dream / MATH / `75`

Prompt: Two fair, 6-sided dice are thrown. What is the probability that the product of the two numbers is a multiple of 5? Express your answer as a common fraction.

KLASS
Incorrect 235 NFE
Verbatim excerpt

The first die can show a multiple of 5 in 1 way (the number 5), and the second die can show a multiple of 5 in 1 way (the number 5). So, there are 1 + 1 = 2 outcomes where the product is a multiple of 5.
However, we need to consider the case where both dice show a multiple of 5, which is only 1 outcome (5, 5). So, the total number of favorable outcomes is 2 + 1 = 3.
The probability is the number of favorable outcomes divided by the total number of outcomes.
The final answer is 3/36 = 1/12.

LESS
Incorrect 170 NFE
Verbatim excerpt

The first die can show a multiple of 5 in 1 way (the number 5), and the second die can show a multiple of 5 in 1 way (the number 5). So, there are 2 favorable outcomes.
However, we need to subtract the case where both dice show a multiple of 5, which is only 1 outcome (5, 5). So, the total number of favorable outcomes is 2 - 1 = 1.
Therefore, the probability that the product of the two numbers is a multiple of 5 is 1/36.
The final answer is $\boxed{1/36}$.

TAPER (ours)
Correct 132 NFE
Verbatim excerpt

There is only 1 multiple of 5 on a 6-sided die, which is 5 itself. So, there are 6 possible outcomes where the first die shows a 5 and 6 possible outcomes where the second die shows a 5. However, we need to subtract the case where both dice show a 5, which is counted twice. So, the total number of favorable outcomes is 6 + 6 - 1 = 11.
Therefore, the probability that the product of the two numbers is a multiple of 5 is 11/36.
The final answer is 11/36.

Figure 5: Dream on MATH, record 75. TAPER counts the six outcomes for each die showing 5 and subtracts the double-counted $(5, 5)$ outcome. The baseline excerpts miscount the favorable outcomes. The correct probability is $11/36$; TAPER uses 132 evaluations versus 235 and 170.

Sorting the intended iterable. LLaDA / MBPP / `64`

Prompt excerpt: Write a function to sort a list of tuples using the second value of each tuple.

KLASS
Incorrect 73 NFE
Verbatim excerpt

```
def subject_marks(tuples):
    return sorted(tuples.
     sort, key=lambda x: x
     [1])
```

LESS
Incorrect 51 NFE
Verbatim excerpt

```
def subject_marks(
     tuples_list):
    return sorted(tuples,
     key=lambda x: x[1])
```

TAPER (ours)
Correct 38 NFE
Verbatim excerpt

```
def subject_marks(
     tuples_list):
    return sorted(
     tuples_list, key=lambda
      x: x[1])
```

Figure 6: LLaDA on MBPP, task 64. Verbatim function excerpts; generated prose and tests are omitted. KLASS passes a bound method to `sorted`; LESS references an undefined variable. TAPER sorts the supplied list. Rechecking the prompt's three visible tests yields a `TypeError`, a `NameError`, and three passes, respectively, consistent with the saved evaluator labels.

Returning the second distinct minimum. Dream / HumanEval / `HumanEval/90`

Prompt summary: `next_smallest` returns the second smallest distinct element, or `None` if absent. The prompt includes `next_smallest([1, 1]) == None`.

KLASS
Incorrect 43 NFE
Full code

```
def next_smallest(lst):
    if len(lst) < 2:
        return None
    lst.sort()
    return lst[1]
```

LESS
Incorrect 41 NFE
Full code

```
def next_smallest(lst):
    if len(lst) < 2:
        return None
    lst.sort()
    return lst[1]
```

TAPER (ours)
Correct 22 NFE
Full code

```
def next_smallest(lst):
    if len(lst) < 2:
        return None
    lst = set(lst)
    if len(lst) < 2:
        return None
    return sorted(lst)[1]
```

Figure 7: Dream on HumanEval/90. Complete saved code outputs. For the prompt's `[1, 1]` case, both baselines return 1, while TAPER removes duplicate values and returns `None`. All four visible examples pass under TAPER; the saved benchmark evaluator also labels only TAPER correct.