ACE: Attention Coalition Envelopes for Parallel Decoding in Diffusion Language Models

Anonymous authors
Paper under double-blind review

Abstract

Masked diffusion large language models (dLLMs) enable parallel generation by predicting multiple unresolved positions in a single denoiser pass, but individually confident predictions need not be jointly compatible. We study this tension through a cooperative-game view of parallel commitment. Under a coherent reference distribution, the KL risk of a parallel commit decomposes into individual prediction errors and a supermodular dependence cost. This structure leads to an ordered modular envelope that assigns each candidate a fixed charge before selection, reducing coalition formation to sorting and a cumulative sum. We instantiate this principle as the *Attention Coalition Envelope* (ACE), which uses native attention to form compatibility-aware token coalitions without training or additional denoiser evaluations; its formal guarantee applies to the attention-induced surrogate game. Across six benchmarks on LLaDA and Dream, ACE reaches the empirical quality–efficiency Pareto frontier in all 12 benchmark–backbone settings, with 30/36 evaluated operating points being Pareto-optimal. It also attains the highest measured accuracy in 10/12 settings, including 5 statistically supported leads. On two matched LLaDA reasoning controls, ACE reduces denoiser evaluations by 32–36% at comparable accuracy.

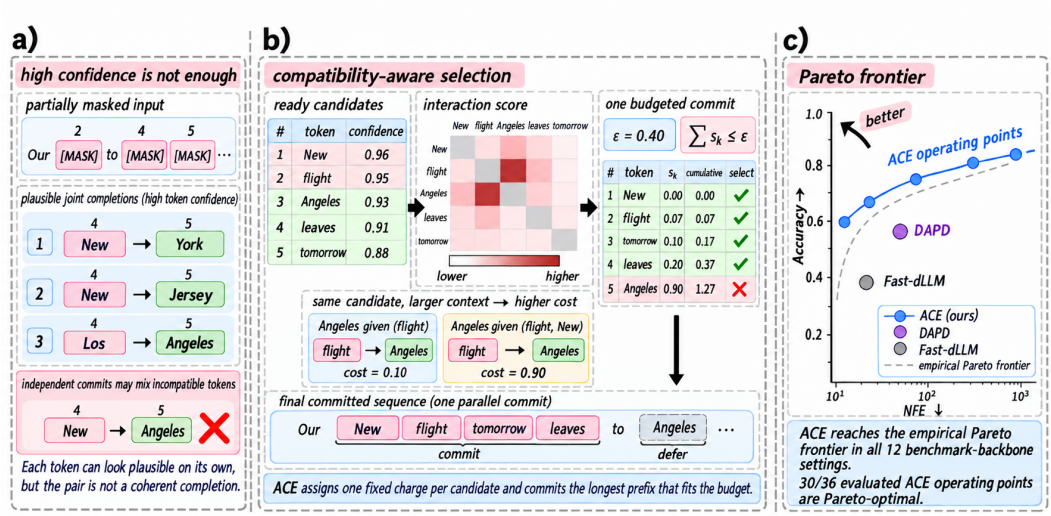

Figure 1: (a) **Problem:** confident tokens can be jointly incompatible; (b) **Method:** ACE assigns fixed scores and commits the largest coalition within a shared budget. (c) **Result:** ACE offers Pareto-optimal quality–efficiency trade-offs in all twelve benchmark–backbone settings; the curve is schematic.

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.

1 INTRODUCTION

Autoregressive language models generate a response one token at a time, making the number of serial model evaluations grow with the output length Vaswani et al. (2017); Brown et al. (2020). Masked diffusion language models (dLLMs) offer a more efficient decoding paradigm: each denoiser evaluation predicts every unresolved position, allowing several tokens to be committed in parallel Sahoo et al. (2024); Nie et al. (2025); Ye et al. (2025). This parallelism, however, is not free. Standard multi-token transitions sample selected predictions independently even though language tokens are coupled by syntax, semantics, arithmetic, program state, and long-range constraints Wu et al. (2026); Kang et al. (2026). Committing too aggressively can therefore combine individually plausible tokens into a jointly implausible sequence, while conservative selection leaves parallelism ineffective. **The central challenge is to identify large, mutually compatible groups of tokens without turning selection itself into a sequential bottleneck.**

Parallel decoding thus involves two distinct questions: whether an individual prediction is reliable enough to commit, which we call *readiness*, and whether a set of ready predictions can safely be committed together, which we call *compatibility*. Confidence, entropy, and temporal stability provide useful readiness signals Wu et al. (2026); Kim et al. (2025); Bao et al. (2026), but they do not capture the dependence lost when several predictions are committed independently. Dependency-aware decoders instead model compatibility through attention graphs, learned pairwise influence, or contextual independence tests Kim et al. (2026); Ringel et al. (2026); Azangulov et al. (2026). While effective, these approaches either reduce dependence to hard or pairwise conflicts, require an auxiliary predictor, or introduce additional model evaluations.

Different form existing approaches, we take a set-level view of parallel commitment and formulate it as a cooperative game, where unresolved positions are players and a simultaneous commit forms a coalition. Under a coherent reference conditional, the KL risk of a parallel commit decomposes exactly into individual prediction error and conditional total correlation Watanabe (1960), cleanly separating readiness from compatibility. The resulting dependence cost is supermodular: the additional cost of a token cannot decrease as the coalition grows. Consequently, the safe commit size depends not only on the decoding state but also on which tokens are committed together. This perspective also provides a common view of existing decoders through their readiness rules, compatibility surrogates, and selection procedures.

The remaining difficulty is computational. Evaluating the exact dependence cost after every addition would require growing the coalition sequentially and recomputing the remaining costs at each step. To avoid this bottleneck, we derive an *ordered modular envelope*: each token is charged once for joining all higher-priority predecessors, and the resulting fixed charges upper-bound the cost of any subsequently selected coalition. Budgeted selection then reduces to one sort and one cumulative sum. The proposed *Attention Coalition Envelope* (ACE) turns this construction into a practical decoder using native attention, without training or additional denoiser evaluations. Its formal guarantee applies to the attention-induced surrogate game; whether native attention tracks the underlying probabilistic dependence is evaluated empirically.

Across six benchmarks on LLaDA and Dream, ACE reaches the empirical quality–efficiency Pareto frontier in all 12 benchmark–backbone settings, with 30/36 evaluated operating points being Pareto-optimal. It attains the highest measured accuracy in 10/12 settings, including 5 statistically supported leads. Matched LLaDA controls further show that the ordering recovers reasoning accuracy, while compatibility-aware selection reduces denoiser evaluations by 32–36% on two reasoning tasks at comparable accuracy. Our contributions are:

1. **A unified game-theoretic framework.** We decompose the parallel commit risk into individual prediction errors and supermodular dependence under a coherent reference distribution, and characterize the maximum commit size under a dependence budget along a greedy chain. This view unifies existing parallel decoders through readiness, compatibility, and selection.
2. **One-shot coalition selection with ACE.** We derive an ordered modular envelope that upper-bounds coalition costs using fixed per-token charges. ACE instantiates the envelope with native attention, requiring neither training nor additional denoiser evaluations; its guarantee applies to the attention-induced surrogate game.

3. **Pareto gains with mechanism attribution.** Across six benchmarks on LLaDA and Dream, ACE achieves Pareto-optimal quality–efficiency trade-offs across both backbones. Matched controls further separate the roles of ordering and compatibility, showing that ordering recovers reasoning accuracy while ACE reduces evaluation cost at comparable quality.

2 Related Work

Autoregressive language models. Most contemporary language models use causal Transformers and factorize a response from left to right Vaswani et al. (2017); Brown et al. (2020). This gives each token the complete generated prefix as context but imposes a serial generation path. Draft-and-verify methods such as speculative decoding can accept several autoregressive tokens per target-model call while preserving the target distribution Leviathan et al. (2023). They accelerate causal generation; parallel dLLM decoding instead decides which unresolved positions may be committed jointly.

Masked and diffusion language models. Discrete diffusion replaces left-to-right generation with iterative denoising of a corrupted sequence Austin et al. (2021a); Lou et al. (2024); Sahoo et al. (2024). LLaDA Nie et al. (2025) and Dream Ye et al. (2025) scale masked denoising to instruction-tuned language models, while block diffusion Arriola et al. (2025) interpolates between full-sequence denoising and autoregressive block order. These models expose position-level parallelism because one denoiser call predicts every active mask, but exploiting it requires an explicit commitment policy.

Parallel decoding for diffusion language models. Training-free decoders commonly commit tokens using fixed schedules, confidence thresholds, or temporal stability, while learned policies predict when a token is ready Wu et al. (2026); Kim et al. (2025); Bao et al. (2026). Dependency-aware methods use hard attention graphs, attention-derived orders or clusters, learned pairwise influence, and contextual conditional-independence tests Kim et al. (2026); Luo et al. (2026); Qi et al. (2026); Zhou et al. (2026); Ringel et al. (2026); Azangulov et al. (2026). Kang et al. (2026); Chen et al. (2026); Zhang et al. (2026) show that the generation order and safe parallelism are inseparable. ACE differs by treating native attention as a continuous internal-coalition cost and converting it into one-shot budgeted selection. Appendix B provides the detailed comparison.

3 Masked Diffusion Language Models and Parallel Decoding

3.1 Masked-diffusion training

Let $\mathcal{V}$ be a finite token vocabulary, let $[\texttt{MASK}] \notin \mathcal{V}$ be a special mask symbol, and let $[L] := \{1, \dots, L\}$ index a response of length L. We write x for an optional prompt or other visible context and $y = (y_1, \dots, y_L) \in \mathcal{V}^L$ for the clean response.

A masked diffusion language model corrupts y by replacing response tokens with $[\texttt{MASK}]$. At noise level $t \in [0, 1]$, each position is masked according to the noise schedule, producing $y_t \in (\mathcal{V} \cup \{[\texttt{MASK}]\})^L$ with masked set $M_t := \{i \in [L] : (y_t)_i = [\texttt{MASK}]\}$. For every $i \in M_t$, the denoiser predicts $q_{\theta,i}(v \mid x, y_t, t)$ over $v \in \mathcal{V}$ from the prompt, visible response tokens, and optional noise level. Modern masked diffusion objectives reduce to a time-weighted masked-token cross-entropy of the form

$$\mathcal{L}_{\mathrm{MDM}}(\theta) = \mathbb{E}_{y,x,t,y_t}\left[w(t) \sum_{i \in M_t} -\log q_{\theta,i}(y_i \mid x, y_t, t)\right], \tag{1}$$

where $w(t) \geq 0$ is determined by the diffusion parameterization and noise schedule. This formulation covers the objectives used by MDLM-style models and current diffusion large language models Sahoo et al. (2024); Nie et al. (2025); Ye et al. (2025).

3.2 Iterative unmasking at inference

At any decoding round, whose index we suppress, partition $[L]$ into revealed positions U and masked positions $M = [L] \setminus U$. Let y_U be the revealed response tokens and write $z := (x, y_U, M)$ for the current state, augmented with the noise level when required. One denoiser pass returns the marginal

$q_i(v) := q_{\theta,i}(v \mid z)$ for every $i \in M$. The active set $M^{\text{act}} \subseteq M$ contains the positions eligible for commitment: it equals M for full-sequence diffusion and the masked portion of the current block for block diffusion Arriola et al. (2025).

A parallel decoder selects a nonempty set $S \subseteq M^{\text{act}}$ and samples from the product distribution $q_z^S(Y_S) := \prod_{i \in S} q_i(Y_i)$. The selected values are committed simultaneously by updating $U \leftarrow U \cup S$ and $M \leftarrow M \setminus S$. Thus one denoiser call commits $|S|$ tokens. Existing samplers choose S using fixed schedules, top-k confidence, confidence thresholds, temporal stability, or explicit dependency estimates Wu et al. (2026); Kim et al. (2025; 2026); Ringel et al. (2026).

4 Commitment as a Cooperative Game

Our goal is to commit as many tokens as possible per denoiser call while controlling parallel commit risk. **We treat unresolved positions as players and a set selected for one parallel commit as a coalition**. The analysis applies to all masked positions M and to any restricted active set $M^{\text{act}} \subseteq M$.

Let $p_\star(Y \mid x)$ be a fixed reference distribution over complete responses, such as the data distribution targeted by training. The joint distribution over unresolved tokens is $p_z(Y_M) := p_\star(Y_M \mid x, Y_U = y_U)$. The reference is *coherent*: all coalition distributions are marginals of this same joint, allowing us to separate individual prediction errors from token dependence (Appendix A.1).

Proposition 1 (Parallel commit risk decomposition)**.** *For every nonempty $S \subseteq M$, let $\mathcal{R}_z(S) := D_{\mathrm{KL}}\big(p_z(Y_S) \,\big\|\, q_z^S(Y_S)\big)$ be its* parallel commit risk*. Then it decomposes into unary prediction errors $e_z(i)$ and a dependence cost $c_z(S)$:*

$$\mathcal{R}_z(S) = \sum_{i \in S} e_z(i) + c_z(S), \tag{2}$$

$$e_z(i) := D_{\mathrm{KL}}\big(p_z(Y_i) \,\|\, q_i(Y_i)\big), \qquad c_z(S) := D_{\mathrm{KL}}\big(p_z(Y_S) \,\|\, \textstyle\prod_{i \in S} p_z(Y_i)\big).$$

Individually plausible predictions need not be jointly compatible, as illustrated by "New Angeles" (Figure 1a). Even perfect individual predictions leave the dependence cost $c_z(S)$, the minimum excess expected log loss of any factorized predictor relative to the reference joint Gneiting & Raftery (2007). This conditional total correlation Watanabe (1960) measures dependence that individual prediction accuracy cannot remove. The risk compares distributions, not the correctness of an individual decoded sequence (Appendix A.7).

We model compatibility through the cooperative game (M, c_z), with $c_z(\varnothing) := 0$, and handle unary readiness separately. This view unifies existing decoders through their readiness rules, surrogate coalition costs, and selection procedures (Appendix B).

To form a coalition, we need to know how its cost changes when another token joins it. For $S \subseteq M$ and $i \in M \setminus S$, the additional cost is the token's mutual information with the group under p_z (Appendix A.2):

$$\Delta c_z(i \mid S) := c_z(S \cup \{i\}) - c_z(S) = \mathrm{I}_z(Y_i; Y_S). \tag{3}$$

The next result shows that the same token's additional cost cannot decrease as the coalition grows (Appendix A.3).

Proposition 2 (**Supermodularity** of coalition dependence)**.** *The dependence cost c_z is supermodular: for every $A \subseteq B \subseteq M$ and $i \in M \setminus B$,*

$$\Delta c_z(i \mid A) \leq \Delta c_z(i \mid B). \tag{4}$$

Figure 1b illustrates this increasing-cost intuition: adding "Angeles" becomes more costly when the group already includes "New." To find how far a coalition can grow within budget, consider a *greedy chain*: start empty and repeatedly add the token with the smallest additional dependence cost. Supermodularity then links the dependence budget to the maximum commit size along this chain.

Proposition 3 (**Maximum commit size**, informal)**.** *Along this greedy chain, each successive addition costs at least as much as the previous one. Once the total dependence cost exceeds a budget $\varepsilon \geq 0$, no later coalition in the chain is feasible. The last budget-feasible coalition therefore determines the* maximum commit size *along this chain, not over all subsets.*

The formal statement and proof are provided separately (Appendix A.4). The **maximum commit size** depends on the decoding state and dependence budget: a fixed commit size can exceed the budget or leave available parallelism unused.

Practical decoders avoid evaluating the reference dependence cost: unary gates omit interactions; DAPD uses binary conflicts, DEMASK learned pairwise costs, and PUNT contextual feasibility tests Wu et al. (2026); Kim et al. (2025; 2026); Ringel et al. (2026); Azangulov et al. (2026). The ideal greedy chain also requires sequential cost updates. Section 5 replaces these updates with fixed upper-bound scores, then constructs practical surrogate scores from native attention.

5 METHOD

The greedy chain in proposition 3 requires reference costs and sequential updates. We first replace these updates with fixed upper-bound scores, then construct practical surrogate scores from native attention.

Let $a_i := \max_{v \in \mathcal{V}} q_i(v)$ denote confidence. Order at most b highest-confidence eligible positions in M^{act} as $E_z = (e_1, \ldots, e_m)$, for example by descending confidence. The candidate cap b equals the block size in block mode (Appendix C.2). To bound each candidate's cost independently of which predecessors are selected, score it against all its predecessors: define $P_k := \{e_1, \ldots, e_{k-1}\}$ and ideal score $u_k := \Delta c_z(e_k \mid P_k) = \mathrm{I}_z(Y_{e_k}; Y_{P_k})$, with $u_1 = 0$.

Theorem 1 (Ordered modular upper envelope)**.** *For any candidate order E_z with scores u_k defined above, every subset $S \subseteq E_z$ satisfies*

$$c_z(S) \le \sum_{e_k \in S} u_k. \tag{5}$$

By supermodularity, joining any subset of predecessors costs no more than joining all predecessors. Summing these fixed scores gives an additive, or modular, upper bound for every coalition, even when some predecessors are excluded (Appendix A.5). For a per-round budget $\varepsilon \ge 0$, the envelope gives the selection problem

$$\max_{S \subseteq E_z} |S| \quad \text{subject to} \quad \sum_{e_k \in S} u_k \le \varepsilon. \tag{6}$$

Each token contributes one unit of progress, so taking the longest budget-feasible prefix of ascending scores maximizes commit size under this constraint. The candidate order fixes the scores; ascending-score selection then admits four tokens and defers "Angeles" in Figure 1b. Denote this rule for practical nonnegative scores $s = (s_1, \ldots, s_m)$ by $S^*(s, \varepsilon)$.

For training-free, head-free selection, ACE replaces the ideal scores u_k with attention-based predecessor sums. Let $\overline{A}$ be the current pass's head-averaged final-layer attention. Extract masked-position interactions without renormalizing, and set $w_z(i,j) := (\overline{A}_{ij} + \overline{A}_{ji})/2$, with $w_z(i,i) := 0$. For an interaction threshold $\tau \ge 0$, the surrogate scores are

$$s_k^{\mathrm{ACE}}(\tau) := \sum_{\ell < k} w_z(e_k, e_\ell)\, \mathbf{1}[w_z(e_k, e_\ell) > \tau], \qquad s_1^{\mathrm{ACE}}(\tau) = 0. \tag{7}$$

The threshold τ removes interactions at or below its value; $\tau = 0$ retains the original weights. All scores are computed simultaneously. The risk decomposition also motivates a unary readiness penalty; an additional order prior discourages committing ahead of unresolved earlier tokens:

$$R_k := r_z(e_k) + s_k^{\mathrm{ACE}}(\tau) + \lambda \frac{\left|\{j \in M^{\mathrm{act}} : j < e_k\}\right|}{|M^{\mathrm{act}}|}. \tag{8}$$

Here $r_z(i) \ge 0$ penalizes insufficient confidence in the individual prediction (Appendix D.1). The order term is the fraction of active masked positions preceding e_k, scaled by $\lambda \ge 0$. Applying $S^*((R_1, \ldots, R_m), \varepsilon)$ budgets readiness, attention, and order scores together.

Predecessor scores depend on candidate order, so we compare confidence, symmetric degree, directed anchor, and left-to-right orders, retaining the coalition maximizing utility J (Appendix C.2):

$$J(S) := |S| + g_{\mathrm{out}}(S), \qquad g_{\mathrm{out}}(S) := \frac{1}{2} \sum_{i \in S} \sum_{j \in M \setminus S} w_z(i,j). \tag{9}$$

The bonus heuristically rewards potential context for future commits, using unthresholded attention to positions left masked, including later blocks. Except for fallbacks, the winning coalition satisfies the budget under the scores used to select it. All four orders reuse one denoiser pass and its attention, with a score sort and cumulative sum per order. Selected tokens are committed simultaneously at their original positions using the most likely values in our experiments; others remain masked. Singleton fallbacks ensure progress but can override the budget (Appendix C.1).

The guarantee covers the thresholded attention game; it also controls true dependence if practical scores upper-bound ideal scores on the selected set (Appendices A.7 and A.8). Diagnostics support attention as a dependence-ranking signal beyond unary confidence, not a calibrated cost (Appendix E.1).

6 Experiments

6.1 Experimental setup

We evaluate LLaDA-8B-Instruct and Dream-v0-Instruct-7B Nie et al. (2025); Ye et al. (2025) on six benchmarks: HumanEval Chen et al. (2021) and MBPP Austin et al. (2021b) for code generation, GSM8K Cobbe et al. (2021) and MATH-500 Hendrycks et al. (2021); Lightman et al. (2024) for mathematical reasoning, IFEval Zhou et al. (2023) for instruction following, and BBH Suzgun et al. (2023) with three-shot chain of thought for multi-step reasoning. We compare against our reproductions of Fast-dLLM Wu et al. (2026), EB-Sampler Ben-Hamu et al. (2025), KLASS Kim et al. (2025), and both DAPD variants Kim et al. (2026). Every row is our own rerun on one H200 in bfloat16 with greedy decoding, a 256-token generation length, and matched prompts. We measure efficiency using both wall-clock latency and the number of function evaluations (NFE), where NFE is the mean number of denoiser forward passes per example.

6.2 Main quality–efficiency results

Table 1: Quality–efficiency trade-offs on LLaDA and Dream, reported as accuracy (%) / seconds per sample; gray marks the Pareto frontier and bold the balanced operating point. The displayed ACE operating point lies on the accuracy–latency Pareto frontier in all twelve settings and has the lowest measured latency in ten of twelve.

Method	HumanEval	MBPP	GSM8K	MATH-500	IFEval	BBH
LLaDA-8B-Instruct						
Fast-dLLM (4-block)	36.6 / 1.70	20.2 / 0.91	79.4 / 1.68	27.0 / 2.00	53.8 / 2.38	52.4 / 1.03
EB-Sampler (4-block)	36.6 / 2.59	19.2 / 2.34	77.5 / 5.01	28.4 / 4.76	55.2 / 2.90	57.7 / 3.50
KLASS (4-block)	36.6 / 3.59	18.6 / 3.17	77.3 / 7.41	27.6 / 6.41	55.6 / 3.40	57.5 / 4.19
DAPD-Staged (1-block)	31.7 / 1.21	41.2 / 1.84	69.5 / 3.22	28.0 / 2.67	**62.5 / 1.22**	53.3 / 1.73
DAPD-Direct (1-block)	31.7 / 0.57	36.6 / 1.06	70.5 / 2.32	26.8 / 2.07	57.4 / 0.98	50.0 / 1.04
ACE (balanced, ours)	**37.8 / 0.53**	**37.4 / 0.74**	**76.3 / 1.24**	**30.2 / 1.36**	64.9 / 2.48	**53.3 / 0.86**
Dream-v0-Instruct-7B						
Fast-dLLM	**43.3 / 1.98**	30.4 / 1.15	43.1 / 1.85	17.4 / 3.03	18.8 / 0.86	45.9 / 1.25
EB-Sampler	46.3 / 4.17	31.4 / 7.33	41.6 / 6.87	13.2 / 7.94	7.7 / 2.15	50.1 / 4.61
KLASS	48.2 / 4.59	31.6 / 3.42	41.7 / 7.18	13.2 / 8.53	7.4 / 2.02	49.9 / 5.00
DAPD-Staged	46.9 / 3.78	47.8 / 2.55	55.5 / 3.99	27.2 / 3.34	36.3 / 2.30	45.0 / 3.89
DAPD-Direct	46.9 / 3.94	51.0 / 1.33	56.9 / 3.86	31.6 / 3.48	37.8 / 0.53	48.3 / 3.27
ACE (balanced, ours)	59.2 / 2.04	**52.6 / 0.50**	**53.4 / 1.24**	**32.8 / 2.35**	**41.5 / 0.37**	**56.4 / 0.90**

Table 1 summarizes deployment-level accuracy–latency trade-offs across both backbones. The displayed ACE configuration lies on the empirical accuracy–latency Pareto frontier in all twelve benchmark–backbone settings and has the lowest measured latency in ten of twelve. Complementarily, across the complete uncached accuracy–NFE grid, ACE reaches the empirical Pareto frontier in all twelve settings, as sketched in Figure 1c, with 30/36 evaluated ACE operating points being Pareto-optimal (Appendix E.2). Thus, quality–efficiency gains persist in both deployment latency and denoiser-evaluation counts.

The benefit remains task-dependent: quality-oriented settings may require more computation, and aggressive DAPD configurations can remain cheaper at some operating points. The resulting curves

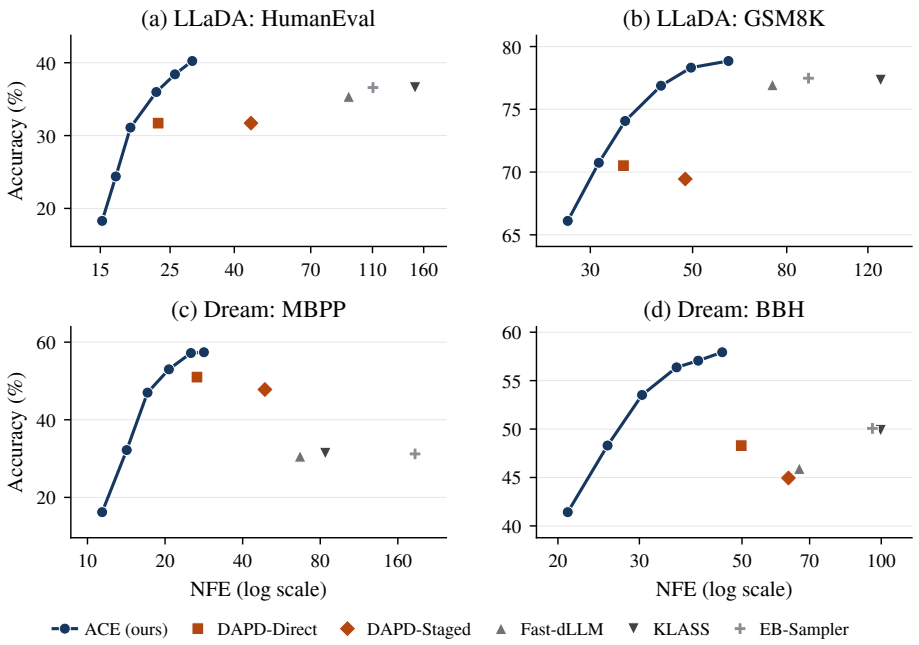

Figure 2: Accuracy versus denoiser evaluations across representative ACE operating points on coding and reasoning benchmarks for LLaDA and Dream. Every displayed baseline is strictly dominated by a measured ACE setting with higher accuracy and fewer evaluations.

therefore characterize a family of quality–efficiency trade-offs rather than a universal speed advantage (Figure 2 and Appendix E.4).

Beyond the balanced choices in Table 1, a fixed LLaDA configuration attains the highest measured accuracy on four of six tasks, while the same ACE selector leads all six Dream tasks with task-specific readiness gates (Appendix E.2). Thus, in uncached comparisons, ACE achieves the highest measured accuracy in 10/12 settings; five of these leads have paired-bootstrap 95% intervals excluding zero against the strongest reproduced baseline (Appendix E.5).

Choosing an operating point. The balanced row selects a nondominated ACE configuration per task using an equal-weight quality–efficiency criterion (Appendix D.2); it is a post-hoc summary rather than a fixed recipe. In deployment, choose the lowest-cost nondominated configuration meeting the target accuracy, and use measured latency when response time is the primary constraint.

6.3 What produces the gains?

Table 2: Dependence-detection AUROC for LLaDA and Dream on constructed examples (Appendix E.1). Attention is more informative overall than readiness or proximity, but can mistake shared context for dependence.

	LLaDA				Dream			
Score	All	Interleaved	Nested	Same-list controls	All	Interleaved	Nested	Same-list controls
Attention weight	**0.96**	0.95	0.93	0.72	**0.77**	0.76	0.77	0.42
Token proximity	0.50	0.24	0.41	0.50	0.50	0.24	0.41	0.50
Readiness	0.47	0.47	0.48	0.32	0.47	0.47	0.46	0.34
Random	0.50	–	–	–	0.50	–	–	–
KL, predicted tokens	0.91	0.91	0.91	0.64	0.94	0.93	0.94	0.66
KL, averaged over tokens	0.96	0.96	0.96	0.81	0.98	0.97	0.98	0.77

Validating the compatibility signal. On constructed dependency examples, attention distinguishes dependent pairs more reliably than readiness or proximity, including interleaved and nested layouts (Table 2). This separation suggests that attention captures compatibility information beyond individual confidence or token adjacency. On LLaDA, its single-pass discrimination is comparable to a KL diagnostic that requires additional model evaluations, supporting its use as an inexpensive selection signal. Shared context can nevertheless mimic dependence, especially on Dream: attention is useful for ranking candidate interactions, but its magnitude is not a calibrated dependence-cost estimate (Appendix E.1).

One-shot versus sequential coalition selection. The sequential construction in proposition 3 motivates testing whether ACE's one-shot envelope can retain the benefits of coalition-dependent selection. We compare ACE's envelope and four-order arbitration with a position-ordered greedy selector that checks each candidate against the growing coalition. The Dream comparison matches attention weights, interaction thresholds, readiness gates, and termination policies (Appendix E.4).

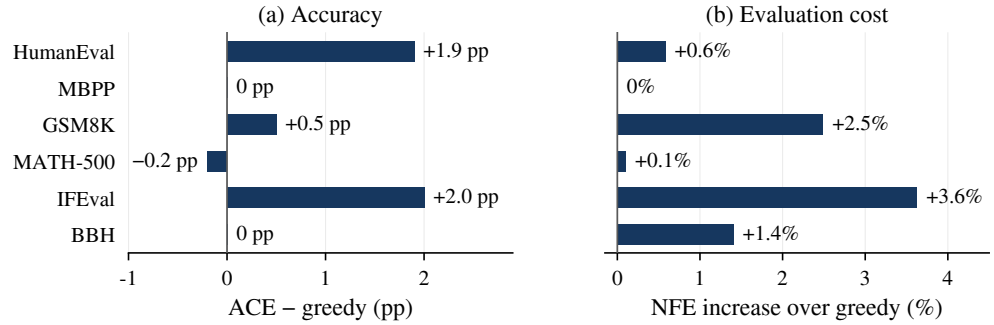

Figure 3: Dream accuracy and NFE changes for ACE versus matched greedy sequential selection. ACE retains comparable accuracy with negligible additional evaluations.

ACE retains comparable accuracy with negligible additional evaluations across six benchmarks (Figure 3). Despite replacing sequential compatibility checks with fixed scores, the envelope and four-order arbitration preserve the greedy selector's measured quality–efficiency trade-off. The key distinction is between modeling interactions and processing them sequentially: these results show that effective coalition formation can retain the former without requiring the latter.

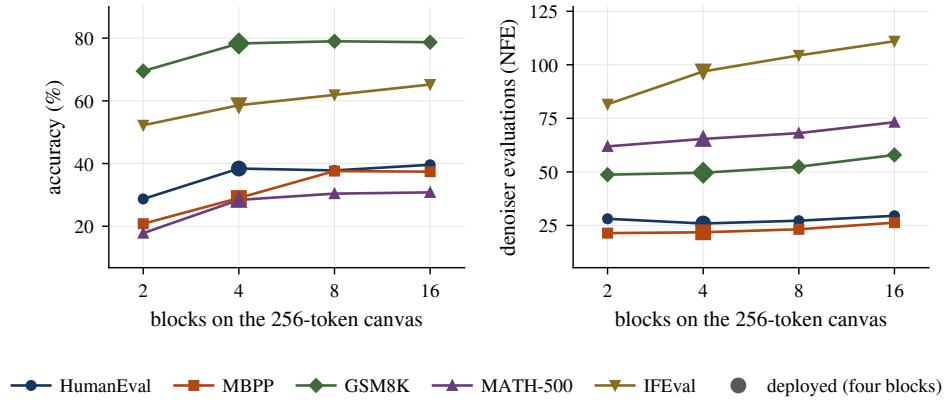

Figure 4: Accuracy and NFE versus block count on LLaDA. Smaller blocks enforce stronger left-to-right order, while larger blocks allow larger parallel commits.

Complementary roles of order and compatibility. Matched controls on GSM8K and MATH-500 separate the effects of ordering and coalition selection. Adding the order prior to confidence decoding recovers most of the full-canvas reasoning accuracy gain. With region and ordering matched, ACE retains comparable accuracy while reducing denoiser evaluations by 32–36% (Appendix E.3). This supports complementary roles: ordering improves readiness decisions, while compatibility-aware selection recovers parallelism at that quality. The matched four-block ACE configuration

also improves both accuracy and NFE throughout the corresponding confidence audit. At fixed readiness, Dream's interaction threshold adjusts the quality–efficiency trade-off: tighter compatibility constraints can improve accuracy, but further restriction adds evaluations without consistent gains (Appendix E.3).

Block size couples ordering and parallelism. Figure 4 shows that finer blocks are an accuracy option for selected tasks, not a uniform quality–efficiency improvement. A smaller decoding region both limits commit size and imposes stronger blockwise order, so the sweep measures their combined effect rather than commit size alone. We retain four blocks as the cross-task reference (Appendix E.4).

6.4 Why adapt commit size?

Within a chosen region, the decoder must still determine how many candidates to commit in each round. Proposition 3 motivates making this decision from the current state and budget. We isolate its benefit by holding candidate-pool construction, scoring, ranking, and decoding protocol fixed, comparing imposed commit sizes with a matched single-order ACE selector stopped at its budget.

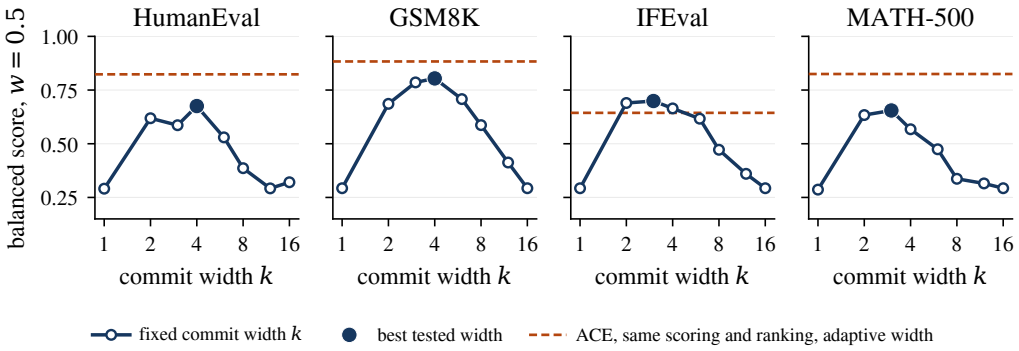

Figure 5: Equal-weight quality–efficiency scores for fixed and adaptive commit sizes on LLaDA. Adaptive ACE outperforms the best tested fixed size on three of four tasks.

Under an equal-weight score combining normalized accuracy and NFE, intermediate fixed sizes maximize the score on each task, while adaptive stopping outperforms the best fixed size on three of four tasks; IFEval favors a fixed size (Figure 5). The preferred size also changes with the quality–efficiency objective. These gains support the budget-based selection principle behind proposition 3. Model-based dependence diagnostics likewise show that one budget admits different commit sizes across states (Appendix E.1).

7 Conclusion

We formulate parallel token commitment of dLLM inference as a cooperative game that separates individual readiness from joint compatibility, which also provides a unified view of existing decoders. The resulting ordered envelope assigns fixed per-token costs, reducing coalition selection to sorting and a cumulative sum while allowing the commit size to adapt to token dependence and budget. ACE instantiates this principle with native attention, requiring neither training nor additional denoiser evaluations; its guarantee applies to the attention-induced surrogate game. Across LLaDA and Dream, ACE improves the quality–efficiency trade-off at multiple operating points, showing that compatibility-aware parallel decoding need not rely on sequential coalition formation.

Compatibility control need not improve on tuned confidence gates, so dependence alone does not predict gains. Settings depend on checkpoint and output structure; robustness across model scales remains unestablished. Main comparisons use greedy commitment; limited stochastic checks show task-dependent accuracy changes at stable NFE, leaving matched stochastic comparisons open.

AI USE STATEMENT

Generative AI tools were used to improve the readability of the manuscript and assist with limited code editing. All AI-assisted text and code were carefully reviewed and verified by the authors. Generative AI tools were not used to develop the research idea, design the methodology, interpret the experimental results, or derive the main scientific claims. The authors take full responsibility for the content and conclusions of this work.

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 (ICLR)*, 2025.

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

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. arXiv preprint arXiv:2108.07732, 2021b.

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

Wenrui Bao, Zhiben Chen, Dan Xu, and Yuzhang Shang. Learning to parallel: Accelerating diffusion large language models via learnable parallel decoding. In *International Conference on Learning Representations*, 2026.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas S Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, Anthony DiPofi, Julen Etxaniz, Benjamin Fattori, Jessica Zosa Forde, Charles Foster, Jeffrey Hsu, Mimansa Jaiswal, Wilson Y. Lee, Haonan Li, Charles Lovering, Niklas Muennighoff, Ellie Pavlick, Jason Phang, Aviya Skowron, Samson Tan, Xiangru Tang, Kevin A. Wang, Genta Indra Winata, François Yvon, and Andy Zou. Lessons from the trenches on reproducible evaluation of language models. arXiv preprint arXiv:2405.14782, 2024.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374, 2021.

Sitan Chen, Kevin Cong, and Jerry Li. Optimal inference schedules for masked diffusion models. In *Proceedings of the Thirty-Ninth Conference on Learning Theory*, 2026.

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

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007. ISSN 1537-274X. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

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*, 2021.

Wonjun Kang, Kevin Galim, Seunghyuk Oh, Minjae Lee, Yuchen Zeng, Shuibai Zhang, Coleman Hooper, Yuezhou Hu, Hyung Il Koo, Nam Ik Cho, and Kangwook Lee. ParallelBench: Understanding the trade-offs of parallel decoding in diffusion LLMs. In *International Conference on Learning Representations*, 2026.

Bumjun Kim, Dongjae Jeon, Moongyu Jeon, and Albert No. DAPD: Dependency-aware parallel decoding via attention for diffusion LLMs. In *International Conference on Machine Learning (ICML)*, 2026.

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 (NeurIPS)*, 2025.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning (ICML)*, 2023.

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 (ICLR)*, 2024.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

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

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 (NeurIPS)*, 2025.

Heqiang Qi, Wei Huang, Mingyuan Bai, and Xiangming Meng. Cluster-level attention-guided parallel decoding for masked diffusion language models. *arXiv preprint arXiv:2605.29607*, 2026.

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

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*, 2024.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-Bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, 2023. URL https://aclanthology.org/2023.findings-acl.824/.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of Research and Development*, 1960.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. In *International Conference on Learning Representations*, 2026.

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.

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

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. arXiv preprint arXiv:2311.07911, 2023.

Yuyan Zhou, Kai Syun Hou, Weiyu Chen, and James Kwok. Attention-based sampler for diffusion language models. *arXiv preprint arXiv:2604.08564*, 2026.

Appendix Contents

A Proofs **14**

A.1 Parallel commit risk decomposition . . . 14

A.2 Entropy form and marginal cost . . . 14

A.3 Supermodularity . . . 15

A.4 Maximum commit size . . . 15

A.5 Proof of the ordered modular upper envelope . . . 16

A.6 Conditional KL and mutual information . . . 16

A.7 Estimated scores and oracle safety . . . 16

A.8 ACE attention game, envelope, and slack . . . 17

B Existing Parallel Decoders as Cooperative Games **18**

B.1 A common formulation of readiness and coalition selection . . . 18

B.2 Unary, pairwise, and contextual models of compatibility . . . 18

B.3 From coalition costs to one-shot envelope selection . . . 19

C Algorithms and Decoding Procedures **20**

C.1 Budgeted coalition selection and progress guarantees . . . 20

C.2 One-round ACE algorithm . . . 20

C.3 Complete generation and termination policies . . . 21

D Experimental Setup Details **22**

D.1 Configurations used in the main results . . . 22

D.2 Evaluation metrics and reporting protocol . . . 23

E Additional Experimental Results **24**

E.1 Dependence signals and adaptive commit size . . . 24

E.2 Denoising Efficiency of ACE . . . 27

E.3 Contributions of ordering and the ACE selector . . . 29

E.4 Numerical results supporting the main experiments . . . 31

E.5 Statistical support for accuracy gains . . . 33

A PROOFS

This appendix follows the argument from parallel commit risk to one-shot coalition selection. We derive the dependence cost and its increasing marginals, use a fixed order to bound it, and then distinguish the ideal risk guarantee from the guarantee for ACE's attention-induced game.

A.1 PARALLEL COMMIT RISK DECOMPOSITION

In population analysis, $p_\star$ is the clean-data distribution and $p_z(Y_M) = p_\star(Y_M \mid x, Y_U = y_U)$, assuming $p_\star(Y_U = y_U \mid x) > 0$. Under token-value-independent masking, the mask pattern supplies no further information about the missing values.

On states with positive training weight, the Bayes-optimal predictor for the masked-token cross-entropy in Equation (1) therefore returns $p_z(Y_i)$ at each masked position. The learned q_i may differ from these marginals, and predictions across masked states need not be conditionals of one joint. The unary terms $e_z(i)$ account for their mismatch with the reference.

With the reference fixed, we can now separate individual prediction error from dependence.

Proof of proposition 1. For a nonempty $S \subseteq M$, expand the definition of $\mathcal{R}_z(S)$:

$$\mathcal{R}_z(S) = \mathbb{E}_{p_z(Y_S)}\left[\log \frac{p_z(Y_S)}{\prod_{i\in S} q_i(Y_i)}\right]. \tag{10}$$

Insert the product of reference marginals:

$$\log \frac{p_z(Y_S)}{\prod_{i\in S} q_i(Y_i)} = \log \frac{p_z(Y_S)}{\prod_{i\in S} p_z(Y_i)} + \sum_{i\in S} \log \frac{p_z(Y_i)}{q_i(Y_i)}. \tag{11}$$

Taking expectation under $p_z(Y_S)$ gives

$$\mathcal{R}_z(S) = D_{\mathrm{KL}}\left(p_z(Y_S) \,\middle\|\, \prod_{i\in S} p_z(Y_i)\right) + \sum_{i\in S} D_{\mathrm{KL}}(p_z(Y_i)\,\|q_i(Y_i)) \tag{12}$$

$$= c_z(S) + \sum_{i\in S} e_z(i). \tag{13}$$

□

Alternatively, a stochastic decoder may define the reference joint, measuring fidelity to that decoder rather than the data; independent restarts from masked states need not yield its conditionals. The reference is an analysis object, not an additional model evaluated by ACE.

With $c_z(\varnothing) := 0$, the compatibility cost vanishes for singletons and is positive whenever the selected variables are not mutually independent. The players in (M, c_z) are token variables, not strategic agents: the decoder chooses which positions to commit, and each position's dependence cost is determined by the coalition it joins. Each added position provides one token of parallel progress, formalized in Appendix A.4.

A.2 ENTROPY FORM AND MARGINAL COST

The decomposition isolates the dependence term; its entropy form reveals the cost of adding one candidate.

Lemma 1 (Marginal cost equals mutual information)**.** *For every $S \subseteq M$ and $i \in M \setminus S$,*

$$\Delta c_z(i \mid S) = \mathrm{I}_z(Y_i; Y_S),$$

where mutual information is evaluated under the state-conditioned reference p_z.

Proof. Let H_z denote entropy under p_z. For finite discrete variables,

$$c_z(S) = \mathbb{E}_{p_z}\left[\log p_z(Y_S) - \sum_{i\in S} \log p_z(Y_i)\right] \tag{14}$$

$$= \sum_{i\in S} \mathrm{H}_z(Y_i) - \mathrm{H}_z(Y_S). \tag{15}$$

For $i \notin S$,

$$c_z(S\cup\{i\}) - c_z(S) = \mathrm{H}_z(Y_i) - \mathrm{H}_z(Y_{S\cup\{i\}}) + \mathrm{H}_z(Y_S) \tag{16}$$

$$= \mathrm{H}_z(Y_i) - \mathrm{H}_z(Y_i \mid Y_S) \tag{17}$$

$$= \mathrm{I}_z(Y_i; Y_S). \tag{18}$$

□

Combining proposition 1 and lemma 1, the marginal parallel commit risk is

$$\mathcal{R}_z(S\cup\{i\}) - \mathcal{R}_z(S) = e_z(i) + \mathrm{I}_z(Y_i; Y_S). \tag{19}$$

When the unary prediction errors $e_z(i)$ are small, the remaining decision quantity is the dependence of i on the current coalition, not a context-free score attached to i.

A.3 Supermodularity

The mutual-information form also shows how this marginal cost changes as the coalition grows.

A set function $f : 2^M \to \mathbb{R}$ is supermodular if, for all $A \subseteq B \subseteq M$ and $i \notin B$,

$$f(A\cup\{i\}) - f(A) \le f(B\cup\{i\}) - f(B). \tag{20}$$

Using lemma 1,

$$[c_z(B\cup\{i\}) - c_z(B)] - [c_z(A\cup\{i\}) - c_z(A)] \tag{21}$$

$$= \mathrm{I}_z(Y_i; Y_B) - \mathrm{I}_z(Y_i; Y_A) \tag{22}$$

$$= \mathrm{I}_z(Y_i; Y_{B\setminus A} \mid Y_A) \ge 0. \tag{23}$$

Hence c_z is supermodular, proving proposition 2.

A.4 Maximum commit size

Each added position contributes one token of parallel progress; proposition 3 identifies how far a greedy chain can grow within a dependence budget.

Proposition 3 (**Maximum commit size**, formal)**.** *For nonempty M, start from $S_0 := \varnothing$ and define $S_k := S_{k-1}\cup\{i_k\}$, where*

$$i_k \in \operatorname*{arg\,min}_{i\in M\setminus S_{k-1}} \Delta c_z(i \mid S_{k-1}), \qquad g_k := \Delta c_z(i_k \mid S_{k-1}). \tag{24}$$

Then $0 = g_1 \le \cdots \le g_{|M|}$, so $c_z(S_k) = \sum_{j\le k} g_j$ is convex and nondecreasing in k. For a dependence budget $\varepsilon \ge 0$, the maximum commit size *is the largest number of tokens committed together along this chain without exceeding the budget:*

$$k^*_\varepsilon = \max\{k \in \{0,\dots,|M|\} : c_z(S_k) \le \varepsilon\}. \tag{25}$$

Proof of proposition 3. For $1 \le k < |M|$ and any remaining candidate $j \in M \setminus S_k$,

$$\Delta c_z(j \mid S_k) \ge \Delta c_z(j \mid S_{k-1}) \ge g_k.$$

The first inequality follows from supermodularity; the second follows from the greedy choice at step k in Equation (24). Minimizing over the remaining candidates gives $g_{k+1} \ge g_k$, regardless of how ties are broken. The first increment is $g_1 = 0$ by lemma 1.

Partial sums of nonnegative, nondecreasing increments are nondecreasing and convex, so the feasible chain prefixes under $\varepsilon \ge 0$ form $\{0,\dots,k^*_\varepsilon\}$, proving Equation (25). The empty prefix is feasible because $c_z(\varnothing) = 0$; if $M = \varnothing$, the maximum commit size is zero. □

The maximum commit size depends on the state: a fixed size can exceed the budget when marginal costs rise early or leave parallelism unused when they remain small. Independent tokens permit the entire pool. The guarantee applies only to this greedy chain and controls dependence, not the correctness of individual predictions.

The ideal greedy procedure requires sequential marginal-cost updates against the selected tokens. Section 5 replaces these adaptive costs with an ordered upper bound computed once.

A.5 Proof of the ordered modular upper envelope

We first decompose dependence along a fixed order, then enlarge each selected predecessor set to obtain the envelope.

Proof of theorem 1. Let $E_z = (e_1, \dots, e_m)$ and $P_k = \{e_1, \dots, e_{k-1}\}$. For $S \subseteq E_z$, define $P_k^S = P_k \cap S$. Telescoping in the inherited order and applying lemma 1 gives

$$c_z(S) = \sum_{e_k \in S} \left[c_z(P_k^S \cup \{e_k\}) - c_z(P_k^S) \right] \tag{26}$$

$$= \sum_{e_k \in S} \mathrm{I}_z(Y_{e_k}; Y_{P_k^S}). \tag{27}$$

The first term is zero because its predecessor set is empty; for $S = \varnothing$, both sums are zero. Because $P_k^S \subseteq P_k$, the chain rule gives

$$\mathrm{I}_z(Y_{e_k}; Y_{P_k}) = \mathrm{I}_z(Y_{e_k}; Y_{P_k^S}) + \mathrm{I}_z(Y_{e_k}; Y_{P_k \setminus P_k^S} \mid Y_{P_k^S}) \tag{28}$$

$$\geq \mathrm{I}_z(Y_{e_k}; Y_{P_k^S}). \tag{29}$$

Therefore,

$$c_z(S) \leq \sum_{e_k \in S} \mathrm{I}_z(Y_{e_k}; Y_{P_k}) = \sum_{e_k \in S} u_k. \tag{30}$$

□

A.6 Conditional KL and mutual information

The preceding envelope uses ideal mutual-information scores. The following identity explains when conditional-KL diagnostics recover these scores in expectation (Appendix E.1).

For $P \subseteq M$ and $i \in M \setminus P$, let z_P be the state after filling P with values Y_P, and define the conditional KL

$$d(i \mid P, z) := D_{\mathrm{KL}}(q_i(\cdot \mid z_P) \,\|\, q_i(\cdot \mid z)) \,. \tag{31}$$

Assume that $q_i(\cdot \mid z) = p_z(Y_i)$ and $q_i(\cdot \mid z_P) = p_z(Y_i \mid Y_P)$. If $Y_P \sim p_z(Y_P)$, then

$$\mathbb{E}_{Y_P \sim p_z}[d(i \mid P, z)] = \sum_{y_i, y_P} p_z(y_i, y_P) \log \frac{p_z(y_i, y_P)}{p_z(y_i) p_z(y_P)} \tag{32}$$

$$= \mathrm{I}_z(Y_i; Y_P). \tag{33}$$

For $P = P_k$ and $i = e_k$, this expectation equals the ideal ordered score u_k. Predicted tokens do not provide this reference average, and model samples alone do not establish the assumed reference conditionals.

A.7 Estimated scores and oracle safety

This diagnostic identity does not ensure that practical scores upper-bound the ideal costs. The following condition is sufficient to transfer the envelope's risk control to estimated scores.

Corollary 1 (Oracle risk control)**.** *If the practical scores satisfy $s_k \geq u_k$ for every $e_k \in S^*(s, \varepsilon)$, then*

$$c_z(S^*(s, \varepsilon)) \leq \varepsilon. \tag{34}$$

Proof. By construction, $\sum_{e_k \in S^*} s_k \le \varepsilon$. Therefore, theorem 1 gives $c_z(S^*) \le \sum_{e_k \in S^*} u_k \le \sum_{e_k \in S^*} s_k \le \varepsilon$. □

Correlation does not establish that every selected score upper-bounds its ideal counterpart. The empirical coverage tests use model-based KL and a modified combined score whose first pool candidate's contribution is zero (Appendix E.1); they do not establish this premise for true dependence.

Distributional risk and greedy commitment. Proposition 1 compares the reference joint distribution with the product distribution $\prod_{i \in S} q_i$, which treats token predictions as independent. This distribution-level comparison does not guarantee a particular greedy completion. The deployed decoder commits each selected token's most likely value, while readiness and compatibility are evaluated from the same denoiser pass before commitment. Appendix E.1 evaluates conditional sensitivity along these greedy trajectories.

A.8 ACE ATTENTION GAME, ENVELOPE, AND SLACK

The attention scores are exact ordered marginals of the thresholded attention game, so the same envelope construction applies. The following proof also identifies its slack.

For fixed $\tau \ge 0$, abbreviate the summand of Equation (7) as $\tilde{w}_z(i,j) := w_z(i,j)\,\mathbf{1}[w_z(i,j) > \tau]$. These weights remain nonnegative and symmetric with zero diagonal, and $\tau = 0$ recovers $\tilde{w}_z = w_z$. The corresponding attention-induced coalition game is

$$c_z^{\mathrm{att}}(S;\tau) := \sum_{\{i,j\} \subseteq S} \tilde{w}_z(i,j), \qquad S \subseteq E_z. \tag{35}$$

For $i \notin S$,

$$c_z^{\mathrm{att}}(S \cup \{i\};\tau) - c_z^{\mathrm{att}}(S;\tau) = \sum_{j \in S} \tilde{w}_z(i,j). \tag{36}$$

This marginal cost is nonnegative and increases with S, so the attention game is monotone and supermodular for every fixed τ.

Theorem 2 (Attention coalition envelope)**.** *For every fixed $\tau \ge 0$ and every subset $S \subseteq E_z$,*

$$c_z^{\mathrm{att}}(S;\tau) \le U_z^{\mathrm{ACE}}(S;\tau) := \sum_{e_k \in S} s_k^{\mathrm{ACE}}(\tau). \tag{37}$$

Proof. Give S the order inherited from E_z. Each internal pair contributes once to both the game cost and the envelope; only the envelope includes excluded predecessors. Thus Equations (7) and (35) give

$$U_z^{\mathrm{ACE}}(S;\tau) - c_z^{\mathrm{att}}(S;\tau) = \sum_{e_k \in S} \sum_{\ell < k} \tilde{w}_z(e_k, e_\ell) - \sum_{e_k \in S} \sum_{\substack{\ell < k \\ e_\ell \in S}} \tilde{w}_z(e_k, e_\ell) \tag{38}$$

$$= \sum_{e_k \in S} \sum_{\substack{\ell < k \\ e_\ell \notin S}} \tilde{w}_z(e_k, e_\ell) \ge 0. \tag{39}$$

This proves Equation (37) and gives its exact slack. □

The slack comes from retained interactions between selected tokens and predecessors excluded from the final coalition. Changing candidate order can therefore change the bound's conservatism, motivating ACE's comparison of candidate orders.

At $\varepsilon = 0$, a budget-feasible coalition has zero thresholded cost and thus no internal pair with $w_z(i,j) > \tau$; this does not imply a maximum-cardinality independent set. This conservatism within the attention game is distinct from its empirical alignment with probabilistic dependence, examined in Appendix E.1. Appendix B places this surrogate alongside the coalition costs of other decoders.

B Existing Parallel Decoders as Cooperative Games

Appendix A establishes the ideal dependence cost and its ordered envelope. We compare practical decoders through three choices: candidate eligibility, the penalty assigned to a group, and the procedure used to select it.

B.1 A common formulation of readiness and coalition selection

At state z, let $E_z \subseteq M^{\mathrm{act}}$ be a method's candidate positions, possibly all active masks, and let k_z be a prescribed commit-size cap. A common objective maximizes commit size:

$$\max_{\substack{S \subseteq E_z \\ |S| \le k_z}} |S| \quad \text{subject to} \quad \widehat{C}_z(S) \le \varepsilon, \tag{40}$$

where $\widehat{C}_z$ is a selection penalty, not necessarily the true compatibility cost c_z of proposition 1; $k_z = |E_z|$ removes the commit-size cap.

This objective provides a common comparison framework, not a claim that every decoder solves it exactly. Selection procedures may impose prefix restrictions, use greedy heuristics, or invoke progress fallbacks. Order-dependent costs are identified below.

Block diffusion as formulated by Arriola et al. (2025) restricts the active player set before selection. A shared block matches the selection domain, not necessarily attention extraction, kernels, or caching.

B.2 Unary, pairwise, and contextual models of compatibility

With the selection problem specified, the central distinction is what information each method uses to assess a group.

Unary readiness and ordering rules. Fixed-size commit rules use $\widehat{C}_z = 0$ and a scheduled cap k_z, with unary priorities breaking ties. The confidence rule of Wu et al. (2026) forms $E_z^{\mathrm{conf}} = \{i : a_i \ge \gamma\}$ and commits eligible positions. The KLASS rule of Kim et al. (2025) additionally requires small temporal KL changes across a recent history, with a confidence-ranked fallback. These rules do not estimate within-coalition interaction.

Entropy-based dependence control. EB-Sampler, introduced by Ben-Hamu et al. (2025), orders positions by a model-error proxy and takes the largest prefix satisfying $\sum_{i \in S} H(q_i) - \max_{i \in S} H(q_i) \le \varepsilon$, where H denotes entropy and the empty set has zero cost. Subtracting the largest entropy gives singletons zero cost and prevents the penalty from being a sum of fixed per-token costs. This dependence bound uses unary statistics without estimating which pairs interact. It is valid for reference marginals; substituting model marginals relies on low model error.

Relational signals reduced to token priorities. Attn-Sampler, introduced by Zhou et al. (2026), reduces attention to column-sum priorities and combines confidence with a dynamic attention-score threshold. Its scores use relational context, but its selector does not accumulate an internal-coalition interaction cost. The following methods retain such interactions explicitly.

Hard pairwise conflicts from attention. DAPD, introduced by Kim et al. (2026), thresholds attention into a conflict graph $G_z = (E_z, \mathcal{E}_z)$. Its graph-selection core admits sets with zero cost under

$$\widehat{c}_z^{\mathrm{DAPD}}(S) := \sum_{\{i,j\} \subseteq S} \mathbf{1}[\{i,j\} \in \mathcal{E}_z]. \tag{41}$$

A degree-prioritized greedy scan produces a conflict-free set that cannot be enlarged by adding another candidate, but need not be the largest conflict-free set. Practical variants also use confidence shortcuts.

DAWN, introduced by Luo et al. (2026), combines attention conflicts with committed anchors. CLAD, introduced by Qi et al. (2026), moves the selection unit to confidence-induced contiguous clusters. Token progress then corresponds to cluster-size weights rather than cluster count.

Learned pairwise costs with adaptive construction. DEMASK, introduced by Ringel et al. (2026), predicts directed influences $\widehat{D}_{i,j}$ of revealing j on the prediction at i. Starting from the leftmost mask, it repeatedly adds a confidence-qualified candidate with minimum accumulated influence from selected positions. For the resulting order $\pi = (\pi_1, \ldots, \pi_{|S|})$, the budgeted dependency is

$$\widehat{c}_z^{\mathrm{DEMASK}}(S; \pi) := \sum_{k=2}^{|S|} \sum_{\ell<k} \widehat{D}_{\pi_k, \pi_\ell}. \tag{42}$$

A global order inherited by every subset assigns a fixed cost to each pair. DEMASK instead constructs the order adaptively, so its accumulated cost depends on both the selected set and its construction order.

Contextual feasibility from additional model evaluations. PUNT, introduced by Azangulov et al. (2026), uses confidence-ordered anchor/test splits to prune candidates whose predictions change excessively after anchors are revealed. Conditional on candidate values and order, acceptance and rejection can be encoded as costs zero and $+\infty$, without asserting supermodularity or exact optimization. The tests obtain set-specific information through additional denoiser evaluations during subset discovery.

B.3 From coalition costs to one-shot envelope selection

The same interaction model can support different selectors. For any pairwise game with fixed nonnegative interaction weights, the ordered-prefix argument in Appendix A.5 gives a modular upper bound. For ACE's thresholded attention game, this gives

$$c_z^{\mathrm{att}}(S; \tau) \le U_z^{\mathrm{ACE}}(S; \tau) := \sum_{e_k \in S} s_k^{\mathrm{ACE}}(\tau). \tag{43}$$

In the attention game, marginal cost depends on selected peers; in the envelope, scores are fixed before selection and can include excluded predecessors (Appendix A.8). Sorting the scores and taking the longest feasible prefix solves the modular commit-size maximization problem exactly, not the original pairwise problem. Theorem 2 bounds the attention game, not the true probabilistic cost.

With $r_z(i) = 0$, $\lambda = 0$, and $\varepsilon = 0$, the envelope selector admits all zero-score candidates in each order. The resulting coalition has no internal interaction above τ, enforcing the hard-conflict family's zero-conflict constraint in one shot (Appendix C.2). Both procedures enforce zero conflicts, but need not select the same set.

More generally, the combined score in Equation (8) adds nonnegative readiness and order-prior terms, preserving the attention-game bound for budget-feasible selections. Singleton fallbacks can override the combined-score budget, and per-order selections are compared using the configured utility (Appendix D.1).

Table 3: Readiness, compatibility, and selection in representative decoders. Among these methods, only DEMASK requires selector-specific training and only PUNT uses additional denoiser calls, whose count depends on the candidate set.

Method	Readiness / priority	Compatibility model	Selection procedure
Fixed commit size	unary priority	none	top-k_z
Confidence	confidence	none	gate and fallback
KLASS	confidence; temporal KL	none	gate and fallback
EB-Sampler	model-error proxy	entropy-only bound	feasible ordered prefix
Attn-Sampler	confidence; attention priority	no set-specific cost	dynamic threshold
DAPD	degree and confidence priority	binary attention conflicts	greedy independent set
DEMASK	leftmost seed; confidence gate	learned directed pairs	adaptive greedy budget
PUNT	confidence order	contextual non-influence	recursive testing
ACE	confidence gate or unary penalty	thresholded attention game	score sort; budgeted prefix

Table 3 shows that compatibility information and coalition construction are separate design choices: attention-based interactions can support either sequential selection or a fixed-score envelope. This conceptual comparison extends beyond the reproduced baselines listed in Section 6.1. Appendix C implements the shared budgeted selector and its deployed configurations.

C Algorithms and Decoding Procedures

Following the coalition-cost formulation in Appendix B, we specify the decoder at three levels: selecting a coalition from fixed scores, running one ACE round, and repeating rounds until termination. Deployed settings and length policies are specified separately (Appendix D).

C.1 Budgeted coalition selection and progress guarantees

Given $m \geq 1$ candidates $e_1, \ldots, e_m$ with nonnegative scores, sort them so that $s_{(1)} \leq \cdots \leq s_{(m)}$, where $e_{(j)}$ is the candidate with score $s_{(j)}$. Preserve the input candidate order on ties. For a budget $\varepsilon \geq 0$, let

$$r^* := \max \left\{ r \in \{0, 1, \ldots, m\} : \sum_{j=1}^{r} s_{(j)} \leq \varepsilon \right\} \tag{44}$$

with the empty sum equal to zero, and define the budget-feasible selection

$$S^*(s, \varepsilon) := \{e_{(1)}, \ldots, e_{(r^*)}\}. \tag{45}$$

This prefix maximizes commit size under the additive budget. For each candidate order, ACE applies this selector to the combined scores in Equation (8):

$$S_z^{\mathrm{ACE}} := S^*((R_1, \ldots, R_m), \varepsilon)\,, \tag{46}$$

where ε budgets readiness, attention, and order scores together. Setting $r_z(i) = 0$ and $\lambda = 0$ recovers attention-only selection.

Progress fallbacks. Deployment handles two empty outcomes separately:

- *Empty candidate pool.* If no candidate passes readiness, select the highest-confidence EOS-eligible active position.
- *Empty budget-feasible prefix.* If candidates exist but none fits the budget, select the cheapest singleton $\{e_{(1)}\}$, even if its score exceeds ε.

These deployment overrides remain separate from the budget-feasible selector S^* used in corollary 1. With a nonempty pool, attention-only selection has $s_1^{\mathrm{ACE}}(\tau) = 0$ and needs no budget fallback; combined scores can require it.

We next instantiate this rule with ACE's scores and candidate orders.

C.2 One-round ACE algorithm

Algorithm 1 uses one denoiser pass to form a candidate pool, score it under several candidate orders, and return a selected coalition with its original predictions. Both backbones follow this pipeline; readiness, scoring, and the utility for choosing among orders are configuration settings (Appendix D.1).

Candidate formation. Let $F \subseteq M^{\mathrm{act}}$ contain the EOS-admissible positions, with $F = M^{\mathrm{act}}$ when the frontier is disabled. Form $C \subseteq F$ from at most b highest-confidence positions passing the configured readiness gate (Appendix D.1). The cap is the block size in block mode and configurable for full-canvas decoding. Marginals use each backbone's native logit alignment, including Dream's right shift.

Readiness and the order prior. The combined scores in Equation (8) add the configured unary readiness contribution $r_z(i)$ and an order prior before applying the same budgeted selector. Confidence surprisal uses $r_z(i) = -\log a_i$; threshold-only readiness is the special case $r_z(i) = 0$ on

eligible candidates. Let $n_i := |\{j \in M^{\text{act}} : j < i\}|$ count unresolved active positions preceding i. These counts are computed by a cumulative sum before EOS filtering; $\lambda = 0$ removes the prior (Appendix E.3).

The nonnegative added terms preserve the attention-game bound for non-fallback selections: $c_z^{\text{att}}(S;\tau) \le \sum_{e_k \in S} R_k \le \varepsilon$. A singleton fallback may exceed the combined-score budget, but has zero attention-game cost (Appendix C.1).

Attention domain. Only EOS-admissible active positions can be selected, but degree, anchor, and g_{out} use unthresholded attention over all unresolved positions M, including later blocks. Eligibility and the order prior use M^{act}. Extracted entries retain full-sequence normalization. The threshold τ affects only coalition scores and their game.

Candidate orders. The orders are considered in the sequence confidence, symmetric degree, directed anchor, then position. The first three sort descending by a_i, $\sum_{j \in M} w_z(i,j)$, and $\sum_{j \in M \setminus \{i\}} (\overline{A}_{ji} - \overline{A}_{ij})$, respectively; position sorts left to right.

Choosing a coalition. Each order changes predecessor sets while reusing the same attention and produces one proposal. We keep the proposal maximizing the configured utility J. Each non-fallback proposal satisfies the budget under its own order's scores.

Ties are resolved deterministically: initial confidence ties favor smaller positions, subsequent sorts preserve input order on ties, and utility ties favor the earlier order in the sequence.

Algorithm 1 Selecting tokens for one simultaneous ACE commit

Require: State $z = (x, y_U, M)$; nonempty active set $M^{\text{act}} \subseteq M$; denoiser q_θ.
Require: Candidate cap b; readiness and score settings from Appendix D.1; threshold $\tau \ge 0$; budget $\varepsilon \ge 0$; candidate orders $\mathcal{O}$; utility J; EOS-frontier policy.

Predict and form candidates

Run q_θ once for marginals q_i and head-averaged final-layer attention $\overline{A}$
$\widehat{y}_i \leftarrow \arg\max_v q_i(v)$, $a_i \leftarrow q_i(\widehat{y}_i)$ for $i \in M^{\text{act}}$
$n_i \leftarrow \left|\{j \in M^{\text{act}} : j < i\}\right|$ for $i \in M^{\text{act}}$ ▷ before EOS filtering
$F \leftarrow$ EOS-admissible positions in M^{act}, or M^{act} if the frontier is disabled
$C \leftarrow$ the at most b highest-confidence positions in F passing the configured readiness gate
if $C = \varnothing$ **then**
    $i^* \leftarrow \arg\max_{i \in F} a_i$; **return** $(\{i^*\}, \widehat{y}_{i^*})$ ▷ readiness fallback
end if

Score and select under each order

$w_z(i,j) \leftarrow \frac{1}{2}(\overline{A}_{ij} + \overline{A}_{ji})$ for $i, j \in M$; $w_z(i,i) \leftarrow 0$
for all orders $o \in \mathcal{O}$ **do**
    Order C as $(e_1, \ldots, e_m)$ by o; compute all $s_k^{\text{ACE}}(\tau)$ using Equation (7)
    $R_k \leftarrow r_z(e_k) + s_k^{\text{ACE}}(\tau) + \lambda n_{e_k} / |M^{\text{act}}|$ using Equation (8)
    $S_o \leftarrow S^*((R_1, \ldots, R_m), \varepsilon)$; if empty, use the cheapest singleton
end for

Choose and return a coalition

$o^* \leftarrow \arg\max_{o \in \mathcal{O}} J(S_o)$; $S \leftarrow S_{o^*}$
return $(S, \widehat{y}_S)$ for one simultaneous commit of the original predictions

The round algorithm returns selected positions and their original predictions. The generation loop commits those predictions, applies termination policies, and updates the decoding state.

C.3 Complete generation and termination policies

In Algorithm 2, the EOS frontier filters candidate eligibility before selection. After commitment, EOS-tail filling and stop-sequence termination can fill masks without another denoiser call. Final truncation determines the returned text (Appendix D.1).

Algorithm 2 Complete generation with the configured selector and length policies

Require: Prompt x; response length L; full-canvas or block mode; denoiser q_θ; configuration from Appendix D.1; its settings and enabled length policies.

Initialize an all-mask response canvas; $U \leftarrow \varnothing$, $M \leftarrow [L]$, $z \leftarrow (x, y_U, M)$
while $M \neq \varnothing$ **do**
  $M^{\mathrm{act}} \leftarrow M$, or the masked positions in the first unfinished block
  Update any scheduled gate using global progress $1 - |M|/L$
  $(S, y_S) \leftarrow$ one round as in Algorithm 1, with the configured settings
  Write y_S into the canvas simultaneously
  Apply enabled EOS-tail filling after a committed terminator
  if stop-sequence termination is enabled and a stop string occurs in the resolved prefix **then**
    Fill remaining masks with the terminator ▷ no further denoiser calls
  end if
  Refresh M from the canvas; $U \leftarrow [L] \setminus M$; $z \leftarrow (x, y_U, M)$
end while
Apply the backbone's EOS and harness suffix-truncation conventions
return the truncated response

Whenever the active set is nonempty, the EOS frontier leaves at least one eligible position. Selection or a fallback therefore removes at least one mask per round, giving at most L denoiser calls; tail filling can only reduce this count. Both backbones use the same fixed-score computation, budgeted selection, and simultaneous commitment, without sequential coalition expansion or additional denoiser calls. Appendix D specifies the experimental configurations.

D Experimental Setup Details

Appendix C describes the decoding procedure; here we specify the experimental configurations and how the reported results are measured and summarized.

D.1 Configurations used in the main results

Shared selector settings. Both backbones use the combined score in Equation (8), the same envelope selector, and four candidate orders: confidence, symmetric degree, directed anchor, and position (Appendix C.2). LLaDA uses minimum confidence $f = 0.1$, unary readiness score $r_z(i) = -\log a_i$, interaction threshold $\tau = 0$, and arbitration by $J(S) = |S| + g_{\mathrm{out}}(S)$. Dream uses minimum confidence γ, $r_z(i) = 0$, the task-specific τ, budget $\varepsilon = 0$, $\lambda = 0$, and arbitration by $J(S) = |S|$. All runs have a 256-token output canvas; the candidate cap is 64 for four-block LLaDA and 256 for full-canvas LLaDA and Dream.

Settings behind the balanced main-table cells. For each task, we select an ACE tier using the uncached accuracy–NFE comparison, then evaluate that configuration with prefix caching for Table 1. Table 4 lists the settings that vary across tasks. These are post-hoc selections, not one fixed cross-task recipe; Appendix D.2 gives the selection criterion. The throughput, default, and quality tiers are defined beside their complete uncached results (Appendix E.2).

Table 4: Settings for the balanced ACE cells in Table 1, all with the prefix KV cache; one block spans the full 256-token canvas, and Dream always uses one block. The arrow denotes a readiness gate changed after half the output positions are committed.

	LLaDA			Dream	
Task	Blocks	ε	λ	τ	γ
HumanEval	4	0.75	0	0.05	0.8
MBPP	1	0.75	1	0.10	0.7
GSM8K	4	1.0	0	0.05	0.6
MATH-500	4	1.0	0	0.10	$0.9 \rightarrow 0.8$
IFEval	1	0.75	1	0.05	0.2
BBH	4	0.75	0	0.05	0.8

Settings behind the frontier curves. Figure 2 reports uncached ACE accuracy–NFE trade-offs across additional measured settings, not just the balanced cells. The LLaDA HumanEval and GSM8K panels sweep $\varepsilon \in \{0.75, 1.0, 1.5, 2.0, 3.0\}$ in four blocks with $\lambda = 0$, and add the full-canvas setting $(\varepsilon, \lambda) = (0.75, 1)$. Dream keeps the full canvas and $\varepsilon = 0$: MBPP sweeps $\gamma \in \{0.4, 0.5, 0.6, 0.7, 0.8, 0.85\}$ at $\tau = 0.1$, and BBH sweeps $\gamma \in \{0.5, 0.6, 0.7, 0.8, 0.85, 0.9\}$ at $\tau = 0.05$. The curves connect nondominated measured settings; the LLaDA curve therefore changes region and the order prior as well as budget.

Region and length policies. Fast-dLLM, EB-Sampler, and KLASS use four blocks on LLaDA; both DAPD variants use the full canvas. All Dream methods use the full canvas.

Three policies govern candidate eligibility and termination (Algorithm 2). The *EOS frontier* admits only the earliest active EOS prediction, and only after all preceding positions are resolved. *Early stopping* fills the unscored tail after a committed terminator without further evaluations. *Stop-sequence termination* checks the contiguous committed prefix after each round and stops at the first harness stop string, preserving the scored response because the harness truncates there.

The reported ACE settings enable early EOS stopping, with the EOS frontier enabled for full-canvas decoding. Stop-sequence termination is enabled on HumanEval and BBH for LLaDA, additionally on MBPP in its full-canvas configuration, and on HumanEval, MBPP, GSM8K, and BBH for Dream. Baselines retain their published runners; matched stopping controls are reported in Appendix E.2.

Prefix KV cache. Table 1 runs ACE with a prefix KV cache in the style of Wu et al. (2026). A full forward pass builds the cache at each block start and refreshes it after every four cached forward passes within a block, including full-canvas decoding. Between refreshes, only the uncached suffix is forwarded; every masked position and its selector inputs are recomputed using the cached prefix. The refresh interval was chosen on twenty-problem pilots where it recovered the uncached accuracy and NFE.

On LLaDA the stock modeling code refuses a cache and is patched at load time to accept one. Dream uses its native cache, with the forwarded suffix beginning one token before the block to preserve logit alignment. Because the bidirectional denoiser sees a stale prefix between refreshes, caching can change outputs; the cached cells of Table 1 and uncached results in Appendix E.2 are therefore reported separately.

The other decoders in Table 1 retain their shipped cache settings: Fast-dLLM uses its own prefix cache on both backbones, while DAPD, EB-Sampler, and KLASS run without a cache, as in their published implementations.

D.2 Evaluation metrics and reporting protocol

Metrics and prompts. All methods use matched prompts and lm-eval scoring Biderman et al. (2024). HumanEval and MBPP report execution-based pass@1, and MATH-500 reports answer exact match. GSM8K uses five-shot prompts and `flexible-extract` exact match; BBH uses three-shot chain-of-thought prompts and `get-answer` exact match, averaged over the 6,511 problems across 27 subtasks. IFEval reports instruction-level loose accuracy: the total number of satisfied

instructions divided by the total number of instructions, rather than an average of prompt-level ratios. All task scores are percentages.

Balanced selection and highlighting. For each backbone and task, gray marks the joint Pareto frontier: five reproduced baselines and three ACE tiers on accuracy against NFE in Table 9, and the five baselines plus the single displayed ACE configuration on accuracy against seconds per sample in Table 1. Within each table, we min-max normalize the two axes separately over its frontier using unrounded values. Bold marks the point minimizing equally weighted squared Euclidean distance to maximum accuracy and minimum cost; ties favor the lower cost.

Selecting the displayed ACE tier and highlighting the main table are separate steps. We first select the tier minimizing this distance among frontier ACE points in the uncached accuracy–NFE comparison, then evaluate that fixed configuration with prefix caching. We recompute the main table's gray and bold marks using the resulting accuracy and latency of its displayed rows. Thus, the row named balanced ACE is not necessarily the bold point on the latency axis. This is a post-hoc summary, not a held-out selection; Table 4 lists the selected settings.

Timing and NFE. Seconds per sample in Table 1 are wall-clock times per problem from prompt tokenization to the decoded output, CUDA-synchronized, on one H200. Methods run sequentially within each timing run. The main table combines cached runs for ACE and Fast-dLLM with uncached runs for the other baselines, using a repeated Fast-dLLM reference to check timing drift between runs. BBH accuracy uses all 6,511 problems; its latency uses the first 25 problems from each of the 27 subtasks, 675 in total.

The ACE rows use the prefix KV cache of Appendix D.1, so their accuracies differ slightly from the uncached decodes in Appendix E.2. NFE is the mean number of denoiser evaluations per example under the configured length policies (Appendix D.1); both full and cached forward passes count as evaluations.

E Additional Experimental Results

We expand the main experiments by examining the dependence signal and adaptive commit size, efficiency without prefix caching, and the contributions of ordering and selection. Complete numerical results and statistical uncertainty follow (Appendices E.4 and E.5).

E.1 Dependence signals and adaptive commit size

We first examine whether attention identifies dependent tokens and whether its scores cover measured dependence costs, extending the signal validation in Table 2. We then test how these costs vary across decoding states and what this implies for adaptive commit size.

Dependence ranking and cost coverage. For 1,004 deployed four-block LLaDA rounds, we measure KL divergence after filling earlier masks, in ACE's chosen order, with predicted tokens. Expected conditional KL equals mutual information under matched reference conditionals and reference-distributed samples (Appendix A.6); here it serves as a model-based diagnostic.

The recorded-round diagnostics in Tables 5 and 6 use KL in nats, from the conditioned distribution to the original one. Pairwise KL conditions on one token; ordered token KL conditions on all earlier candidates in the selected order. Coalition cost sums token KL divergences conditioned only on earlier committed members. Ranking is measured by mean within-round Spearman correlation ρ: pairwise correlations compare pair scores with pairwise KL, while ordered correlations compare attention scores with ordered token KL. The pairwise readiness score is $-\log a_i - \log a_j$.

Coverage tests whether scores are large enough, rather than merely well ranked. Coalition coverage is the fraction of rounds whose summed scores over committed tokens cover their coalition cost, reported over all rounds and over those with cost above 0.01 nats. Per-token coverage compares each score with its ordered token KL and averages within rounds over committed members. The combined score adds $-\log a_i$ to the attention score, with the first pool candidate's contribution set to zero for this diagnostic. Within-budget rates use $\varepsilon = 0.75$; p90 denotes the 90th percentile of coalition cost.

Attention ranks pairwise dependence better than readiness alone, but often underestimates its cost; adding the readiness term substantially improves coverage (Table 5). Rescaling attention alone to improve coverage reduced estimated commit sizes toward one token. A separate check found that most of the 576 stored subsets of three to five tokens had model-based cost no greater than the pairwise sum.

Table 5: Dependence ranking and coverage of model-based KL costs on recorded LLaDA decoding rounds (Appendix E.1). Attention ranks dependence better than readiness alone, while combining both terms substantially improves empirical cost coverage.

(a) Ranking and cost

Task	Mean candidates / committed tokens	Pairwise ρ attention / readiness / random	Median token KL attention deciles 1 / 5 / 10
HumanEval	36.5 / 3.5	0.47 / 0.36 / -0.01	0.31 / 1.55 / 1.68
MBPP	36.3 / 3.7	0.49 / 0.36 / 0.00	0.23 / 1.17 / 1.61
GSM8K	29.3 / 3.1	0.55 / 0.32 / 0.00	0.32 / 1.40 / 1.50
MATH-500	22.8 / 2.5	0.55 / 0.37 / 0.01	0.35 / 1.43 / 1.57
IFEval	23.2 / 2.1	0.55 / 0.30 / 0.00	0.17 / 1.17 / 1.64

(b) Coverage (%)

Task	Attention score all / cost > 0.01	Per-token attention score	Combined score all / cost > 0.01	Cost within budget
HumanEval	71.5 / 52.6	83.3	96.5 / 94.0	98.0
MBPP	53.5 / 38.4	66.5	96.5 / 95.4	97.5
GSM8K	78.5 / 54.0	83.3	98.5 / 96.6	99.0
MATH-500	82.7 / 50.7	82.3	98.0 / 94.4	98.0
IFEval	73.0 / 41.9	76.1	97.5 / 94.6	99.0

Conditioning on predicted or sampled tokens. To test whether the ranking and coverage findings depend on using predicted tokens, we compare predictions with sequential model samples on 606 separate rounds, averaging six sequences per round and three samples per pair.

Sampling preserves the ranking result but increases the largest measured costs, reducing combined-score coverage (Table 6). Mean within-round correlations between predicted and sampled conditioning are $\rho = 0.71$ for ordered token costs and 0.78 for pairwise costs. Thus, good coverage when conditioning on predicted tokens does not imply equally good coverage over sampled alternatives.

Table 6: Dependence ranking and cost coverage when conditioning on predicted or sampled tokens on the same LLaDA rounds. Sampling preserves the ranking signal but exposes larger dependence costs and reduces score coverage.

Conditioning tokens	Pairwise attention ρ	Ordered attention ρ	Pairwise readiness ρ	Cost median / p90	Combined coverage (%) all / cost > 0.01	Within budget (%)
Predictions	0.53	0.57	0.35	0.015 / 0.31	97.9 / 95.9	98.7
Samples	0.57	0.62	0.34	0.019 / 0.94	84.7 / 70.9	86.5

Validation with known dependence. The preceding checks measure changes in the model's own probabilities. To test dependence detection against rules defined independently of the model, we construct 576 examples per backbone, each with four one-token blanks in two independent groups, yielding 3,456 pairs. Within each group, the instruction requires a copy, a country and its capital, a word and its opposite, two different words from a list, or two independent draws from that list.

Blank orders are AABB (adjacent), ABAB (interleaved), and ABBA (nested), with matching letters denoting a group; together they balance proximity so that it cannot distinguish dependent pairs overall. Same-list controls compare dependent pairs only against independent draws from the same list. The KL scores condition on a predicted token or average over listed tokens, weighted by model probabilities renormalized over that list. We report dependence-detection AUROC, with chance at 0.50.

On examples with known dependence, attention separates dependent from independent pairs better overall than proximity, readiness, or random scores (Table 2). Attention's overall 95% bootstrap intervals, resampling examples, are 0.95–0.97 on LLaDA and 0.75–0.79 on Dream. LLaDA's single-pass attention score is comparable in overall discrimination to KL averaged over possible tokens, which requires extra model calls. Separation weakens for two independent draws from the same list, falling below chance on Dream: attention can confuse shared context with actual dependence.

Dream's lower score also depends on how attention is mapped to token positions. Dream predicts position p from the hidden state at $p-1$, whereas the deployed weights use mask positions. A post-hoc check shifting both attention indices one position earlier raises overall AUROC from 0.770 to 0.897, although same-list controls remain near chance. This suggests that attention indexing contributes to the gap. It diagnoses the input signal rather than directly testing ACE's coalition-formation rule; the shifted mapping has not been validated in end-to-end decoding.

Attention helps identify which tokens depend on each other, but does not reliably measure how strong that dependence is. Adding readiness improves cost coverage in the tested LLaDA rounds. The conditioning checks still use the model's own probabilities, and the known-dependence examples are short, constructed tasks. Neither establishes a bound on the true dependence cost: the guarantee remains specific to the attention-based surrogate game (theorem 2).

State-dependent commit size along a greedy chain. We next examine how measured costs accumulate as more tokens are committed. The greedy-chain diagnostic in Section 6.4 and Figure 6 uses 200 recorded LLaDA rounds each on HumanEval, GSM8K, and IFEval, and 202 on MATH-500. At each step, we add the candidate with the smallest model-based KL change after conditioning on the preceding predicted tokens; the first increment is zero because there are no preceding tokens. The 0.75-nat threshold applies to these proxy costs, not the deployed combined score or a calibrated measure of true dependence; their increments need not satisfy the theorem's exact monotonicity.

Each chain is recorded to the smaller of 24 positions and the candidate-pool size. If the recorded prefix remains within the threshold while candidates remain, its feasible commit size is only a lower bound. In Figure 6, the median and interquartile band at each size use only chains reaching that size; 120 faint traces show individual rounds. Marks above the threshold summarize the distribution of feasible commit sizes, pooling sizes of 12 or more at the last tick.

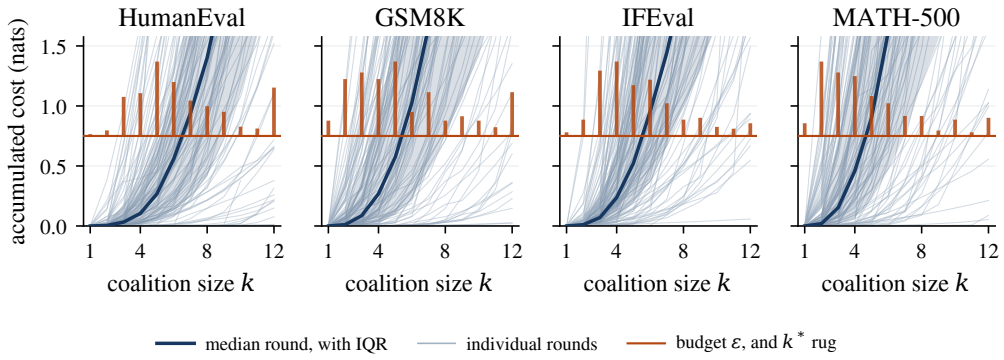

Figure 6: Accumulated proxy dependence costs along greedy chains in recorded LLaDA decoding rounds. A common diagnostic budget admits different commit sizes across states, motivating adaptive rather than fixed-size commitment.

Fixed versus adaptive commit size. To test whether adapting commit size improves end-to-end performance, the study in Section 6.4 and Figure 5 holds candidate-pool construction, scoring, and ranking fixed and replaces budget-based stopping with an imposed commit size $k \in \{1, 2, 3, 4, 6, 8, 12, 16\}$. Both arms use a single confidence order, the same LLaDA readiness settings, and the same deployed task protocol; the adaptive reference is this matched control, not the multi-order default tier. The fixed-size rule takes the first k ranked candidates, or the whole pool if fewer remain. We summarize the quality–efficiency trade-off using

$$\mathrm{Score}_w(k) = 1 - \sqrt{w(1-\hat{a}_k)^2 + (1-w)\hat{n}_k^2}, \tag{47}$$

where $\hat{a}_k$ and $\hat{n}_k$ are accuracy and NFE min-max normalized over all tested fixed sizes and the adaptive control within each task, and w weights accuracy. Figure 5 uses equal weights, $w = 0.5$.

The score in Equation (47) uses the same distance criterion as Table 1, but its normalization bounds cover all eight tested fixed sizes and the matched adaptive control within each task, rather than only the Pareto frontier. Table 7 changes the accuracy weight under these fixed bounds and separately recomputes the equal-weight optimum using bounds from the nondominated subset.

Table 7: Best tested commit sizes and fixed/adaptive scores under alternative accuracy weightings and normalizations; bold marks the higher score. The preferred size changes with the objective, so the measured optimum is not a universal constant.

	Best fixed k All-tested normalization			Best fixed k Frontier normalization	Score at $w = 0.5$ All-tested normalization	
Task	$w = 0.25$	$w = 0.50$	$w = 0.75$	$w = 0.50$	Best fixed	Adaptive
HumanEval	4	4	2	6	0.68	**0.82**
GSM8K	4	4	3	8	0.80	**0.88**
IFEval	4	3	2	4	**0.70**	0.64
MATH-500	3	3	2	6	0.65	**0.82**

The matched adaptive control achieves the higher equal-weight quality–efficiency score on three tasks, with IFEval as the exception. The preferred fixed size changes with accuracy weighting and normalization, showing that its selection depends on the task and evaluation objective.

E.2 Denoising Efficiency of ACE

Table 1 reports deployment latency with prefix KV caching for ACE. Here we first test whether its quality–efficiency gains persist without that cache, then isolate the contribution of termination and examine the cost of each evaluation.

Three operating tiers. The throughput, default, and quality tiers are three configurations of the same ACE selector. The three LLaDA recipes are fixed across tasks: throughput uses four blocks with $(\varepsilon, \lambda) = (1.0, 0)$, default uses four blocks with $(0.75, 0)$, and quality uses the full canvas with $(0.75, 1)$. All share the readiness and arbitration settings in Appendix D.1.

Dream keeps the full canvas and $\varepsilon = 0$, with $\tau = 0.1$ on MBPP and MATH-500 and 0.05 elsewhere. Its task-specific readiness gates are listed in Table 8; each arrow switches the gate after half the output positions are committed, while all other gates are constant. These gates and interaction thresholds were selected task by task using separate independent-set sweeps, with the IFEval quality point chosen for its quality–efficiency trade-off rather than maximum accuracy. The reported tiers use the shared envelope selector (Table 17).

Table 8: Dream readiness gates γ for the three operating tiers. The tiers vary readiness within the same full-canvas ACE selector.

Tier	HumanEval	MBPP	GSM8K	MATH-500	IFEval	BBH
Throughput	0.7	0.5	0.6	0.7	0.1	0.7
Default	0.8	0.7	0.7	0.8	0.2	0.8
Quality	$0.9 \rightarrow 0.8$	0.8	0.8	$0.9 \rightarrow 0.8$	0.3	0.9

Quality–efficiency trade-offs without caching. Table 9 reports accuracy / NFE for these three ACE tiers with prefix KV caching disabled. The reproduced baselines retain their evaluated cache settings: Fast-dLLM is uncached on LLaDA and prefix-cached on Dream, while the other baselines are uncached. Gray marks the empirical accuracy–NFE Pareto frontier within each backbone and task; bold marks the balanced operating point under equal metric weighting (Appendix D.2).

Without prefix caching, ACE offers a nondominated quality–efficiency trade-off in every setting, including operating points that improve both accuracy and NFE over reproduced baselines on each backbone. These gains therefore do not require KV reuse; the cached latency results in Table 1 evaluate an additional deployment optimization.

Table 9: Accuracy (%) / NFE for all three ACE tiers without prefix KV caching and reproduced baselines on LLaDA and Dream. Uncached ACE reaches the accuracy–NFE Pareto frontier in all twelve benchmark–backbone settings.

Method	HumanEval	MBPP	GSM8K	MATH-500	IFEval	BBH
LLaDA-8B-Instruct						
Fast-dLLM (4-block)	35.4 / 92.4	19.4 / 42.0	77.0 / 74.5	28.2 / 96.9	55.8 / 127.6	56.8 / 41.4
EB-Sampler (4-block)	36.6 / 110.4	19.2 / 52.4	77.5 / 89.2	28.4 / 114.1	55.2 / 136.4	57.7 / 64.5
KLASS (4-block)	36.6 / 150.2	18.6 / 68.7	77.3 / 127.9	27.6 / 147.5	55.6 / 157.0	57.5 / 75.0
DAPD-Staged (1-block)	31.7 / 45.2	41.2 / 37.7	69.5 / 48.2	28.0 / 58.9	**62.6 / 50.2**	53.3 / 26.7
DAPD-Direct (1-block)	31.7 / 22.9	**36.6 / 22.4**	70.5 / 35.4	26.8 / 46.2	57.4 / 43.7	50.0 / 16.1
ACE (throughput, ours)	36.0 / 22.6	24.2 / 20.6	**76.9 / 42.7**	**28.0 / 56.3**	57.1 / 85.4	53.9 / 25.7
ACE (default, ours)	**38.4 / 25.9**	29.0 / 21.8	78.3 / 49.6	28.4 / 65.4	58.6 / 96.9	**55.5 / 29.5**
ACE (quality, ours)	40.2 / 29.4	39.2 / 28.2	78.8 / 59.8	34.0 / 84.7	64.5 / 132.7	55.1 / 41.2
Dream-v0-Instruct-7B						
Fast-dLLM	43.3 / 111.5	30.6 / 66.8	43.1 / 98.8	17.4 / 158.2	18.8 / 53.3	45.9 / 66.5
EB-Sampler	46.3 / 155.2	31.2 / 186.6	41.6 / 131.7	13.2 / 190.7	7.7 / 114.3	50.1 / 95.7
KLASS	48.2 / 164.5	31.4 / 83.7	41.7 / 134.5	13.2 / 200.3	7.4 / 104.8	49.9 / 99.7
DAPD-Staged	46.9 / 109.5	47.8 / 48.8	55.5 / 56.4	27.2 / 60.6	36.3 / 82.6	45.0 / 63.0
DAPD-Direct	46.9 / 113.4	51.0 / 26.6	**56.9 / 55.4**	**31.6 / 63.3**	37.8 / 18.6	48.3 / 49.8
ACE (throughput, ours)	50.0 / 90.4	32.2 / 14.2	54.2 / 48.9	29.8 / 80.7	37.4 / 7.5	53.5 / 30.4
ACE (default, ours)	**59.8 / 103.4**	**53.0 / 20.7**	55.8 / 61.7	31.0 / 96.0	**43.5 / 20.0**	**56.4 / 36.1**
ACE (quality, ours)	63.4 / 112.4	57.2 / 25.2	58.5 / 74.3	31.8 / 106.1	47.7 / 49.5	57.9 / 45.3

Call savings from stop-sequence termination. Having established the uncached trade-offs, we isolate how stopping contributes to the call count. We stop decoding once the committed prefix contains an evaluation stop string, avoiding further generation of text that the evaluator would discard. Each pair changes only this stop-sequence termination check; the selector, prompts, and other termination settings remain fixed (Appendix D.1). The accuracy column applies to both settings, and calls saved is the percentage reduction from the NFE without this check.

Table 10: Accuracy and denoiser evaluations with and without stop-sequence termination, holding each decoder's other settings fixed. Accuracy is unchanged, while ACE saves substantially more evaluations than DAPD in the matched LLaDA HumanEval comparison.

Decoder	Accuracy (%) both settings	NFE without stopping	NFE with stopping	Calls saved
LLaDA / HumanEval				
ACE (throughput)	36.0	49.1	22.6	54%
ACE (default)	38.4	59.1	25.9	56%
ACE (quality)	40.2	86.7	29.4	66%
DAPD-Direct	31.7	22.9	22.8	$< 1\%$
DAPD-Staged	31.7	45.2	43.8	3%
LLaDA / MBPP				
ACE (quality)	39.2	57.7	28.2	51%
Dream / MBPP: independent-set controls, $\tau = 0.05$				
Independent set, $\gamma = 0.8$	57.6	97.2	27.1	72%
Independent set, $\gamma = 0.7$	55.2	76.2	22.0	71%

The same stop check preserves every score while saving 51–66% of calls for the representative LLaDA ACE recipes and 71–72% for the Dream independent-set controls, versus at most about 3% for DAPD (Table 10). Frontier-disciplined decoding exposes a resolved stopping prefix while much of the suffix remains masked; DAPD's scattered commits expose it nearer completion. Stopping and selection therefore interact, rather than contribute independent savings. Consistently, quadrupling the canvas in two code-task diagnostics changes calls far less than proportionally, with scores within 1.2 points.

Per-call cost without caching. Call counts alone do not determine runtime, so we also measure the cost of each uncached evaluation. The breakdown in Table 11 uses one H200 and uncached ACE and DAPD with the recipes and termination policies of Table 9.

DAPD rows come from matched same-job rounds. The ACE rows were re-timed alone after an output-neutral selector optimization that reproduces every score and NFE; their denoiser time per call is within 3% of the matched rounds.

The breakdown logs ACE components over each full benchmark, using the default tier on LLaDA and the quality tier on Dream. *Other* comprises stop-sequence and EOS checks plus loop bookkeeping. DAPD runs as shipped without instrumentation, so its per-call total is seconds per sample divided by NFE, converted to milliseconds.

Table 11: Per-call runtime components for uncached ACE and total per-call runtime for uncached DAPD on LLaDA and Dream, in milliseconds. ACE calls are cheaper across the tested tasks, with denoiser computation accounting for most of their runtime.

		ACE					DAPD total	
Backbone	Task	Denoiser	Attention	Selection	Other	Total	Direct	Staged
LLaDA	HumanEval	21.7	0.15	1.79	0.27	23.9	24.9	26.8
LLaDA	MBPP	40.2	0.70	1.85	0.35	43.1	46.9	48.5
LLaDA	GSM8K	50.9	1.38	1.88	0.28	54.4	66.1	67.4
LLaDA	MATH-500	37.0	0.53	1.79	0.29	39.6	45.0	46.0
LLaDA	IFEval	19.0	0.12	1.65	0.19	20.9	22.2	23.9
Dream	HumanEval	22.9	0.23	1.76	0.54	25.4	35.4	34.9
Dream	MBPP	32.6	0.50	1.97	0.67	35.7	50.8	52.7
Dream	GSM8K	42.7	1.25	1.92	0.53	46.4	69.7	71.1
Dream	MATH-500	34.6	0.66	1.81	0.23	37.3	55.5	56.3
Dream	IFEval	16.2	0.14	1.80	0.26	18.4	28.5	27.8

Without caching, ACE's denoiser forward accounts for about nine tenths of each call; reconstruction, selection, and bookkeeping together take about two to four milliseconds (Table 11). Its calls are cheaper than DAPD's on every tested task and backbone. The larger gap on Dream is consistent with reconstructing one attention layer rather than DAPD's ten. One-shot selection and lightweight reconstruction thus keep the per-call overhead small independently of prefix caching, complementing the accuracy–NFE trade-offs above.

E.3 Contributions of ordering and the ACE selector

We separate three questions: whether ACE improves on confidence decoding under matched deployment conditions, what ordering and selector choice contribute to reasoning performance, and, on Dream, what the interaction term alone contributes. In the LLaDA controls, ACE uses $\varepsilon = 0.75$ and confidence decoding uses a fixed $\gamma = 0.9$. Prompts, generation length, and termination settings are matched between selectors within each task and region; the EOS frontier is enabled only for full-canvas decoding (Appendices D.1 and D.2).

Gains under matched deployment conditions. Table 12 first compares the selectors in four-block decoding with the order prior disabled ($\lambda = 0$). The separate full-canvas confidence row tests region choice, not selector choice, and also uses $\lambda = 0$.

Table 12: LLaDA accuracy (%) / NFE under matched four-block decoding, with a full-canvas confidence control. ACE improves both metrics on all five tasks in the matched comparison.

Selector	HumanEval	MBPP	GSM8K	MATH-500	IFEval
Matched selector comparison: four blocks					
Confidence	34.8 / 38.0	19.8 / 34.9	76.6 / 74.0	27.8 / 96.3	57.7 / 126.1
ACE (default)	38.4 / 25.9	29.0 / 21.8	78.3 / 49.6	28.4 / 65.4	58.6 / 96.9
Confidence region control: full canvas					
Confidence	36.6 / 43.0	40.2 / 40.8	68.2 / 78.4	28.0 / 113.9	65.2 / 142.8

Under matched four-block policies, ACE achieves higher measured accuracy on all five tasks with 23–38% fewer evaluations, so its gains are not explained solely by deployment conditions. Changing the region also affects confidence-only accuracy in task-dependent ways; full-canvas confidence exceeds four-block ACE's accuracy on MBPP and IFEval.

Ordering and selector efficiency. With deployment conditions matched, we next separate the roles of ordering and selector choice. Table 13 places confidence and ACE side by side at each fixed region and order-prior setting; the full-canvas rows isolate the accuracy contribution of ordering from the evaluation savings of selector choice. The confidence control uses $-\log a_i + \lambda n_i/|M^{\mathrm{act}}| \le -\log \gamma$; ACE includes the same positional term in its combined budget (Equation (8)).

Table 13: LLaDA reasoning accuracy (%) / NFE for confidence and ACE at matched region and order-prior settings. Ordering improves full-canvas accuracy for both selectors, while ACE retains that quality with 32–36% fewer evaluations.

Region	λ	GSM8K		MATH-500	
		Confidence	ACE	Confidence	ACE
Four blocks	0	76.6 / 74.0	78.3 / 49.6	27.8 / 96.3	28.4 / 65.4
Full canvas	0	68.2 / 78.4	68.5 / 50.8	28.0 / 113.9	26.6 / 75.8
Full canvas	1	78.4 / 93.5	78.8 / 59.8	33.8 / 124.4	34.0 / 84.7

Ordering substantially improves full-canvas reasoning accuracy for both selectors. With ordering matched, ACE retains comparable accuracy while reducing evaluations by 32–36%. This is a selector-level comparison, not an isolated attention-term ablation: the unary readiness contribution and multi-order arbitration also differ from threshold decoding.

The order prior penalizes commitments whose preceding context is unresolved. A post-hoc 100-prompt GSM8K audit finds that the prior nearly eliminates far-ahead commitments under both selectors. Those commitments disproportionately contain numerals, consistent with fixing intermediate results before their derivations, though the audit does not establish that failure mechanism causally. Restoring order also raises NFE: stronger readiness discipline can protect quality while narrowing the available parallelism.

Interaction threshold on Dream. To isolate compatibility strength within the same selector, we now hold readiness and multi-order arbitration fixed. Table 14 varies only τ at each task's fixed default-tier readiness gate, with the same full-canvas selector, budget $\varepsilon = 0$, and termination policies (Appendix E.2). Lower thresholds retain more interactions and impose stricter compatibility constraints; $\tau = 1$ removes those constraints, giving a readiness-only control within ACE's decoding stack. These diagnostic settings do not reselect the balanced configurations in Table 1.

Table 14: Dream accuracy (%) / NFE across interaction thresholds at fixed readiness gates; asterisks mark the default-tier settings. Stricter compatibility control can improve accuracy at greater evaluation cost, but further restriction does not consistently help.

τ	HumanEval	MBPP	GSM8K	MATH-500	IFEval	BBH
0.02	61.0 / 117.0	55.0 / 27.4	57.4 / 73.0	30.2 / 125.1	47.1 / 27.6	56.4 / 43.5
0.05	59.8 / 103.4*	55.6 / 22.6	55.8 / 61.7*	30.6 / 108.6	43.5 / 20.0*	56.4 / 36.1*
0.1	58.5 / 96.3	53.0 / 20.7*	56.0 / 55.2	31.0 / 96.0*	41.5 / 16.9	56.4 / 33.8
0.2	58.5 / 96.1	53.0 / 20.6	55.6 / 55.1	31.2 / 95.5	42.1 / 16.6	56.2 / 33.5
1.0	58.5 / 96.1	53.0 / 20.6	55.6 / 55.1	31.2 / 95.5	42.4 / 16.5	56.2 / 33.5

At the default thresholds, measured accuracy remains close to the readiness-only control. Stronger compatibility constraints improve accuracy on most tasks at increased evaluation cost, but further tightening does not consistently help. Compatibility strength therefore remains a task-dependent quality–efficiency choice even after readiness is fixed, rather than a setting whose most restrictive value is uniformly best.

E.4 Numerical results supporting the main experiments

We first expand the operating-point frontiers, then report the full matched selector comparison and block-count sweep supporting Figures 2 to 4. These grids cover settings and benchmarks omitted from the main plots; tier definitions and the complete uncached accuracy–NFE comparison are in Appendix E.2.

The extended frontiers (Figures 7 and 8) use filled circles for measured ACE settings and solid lines to connect nondominated ACE points. Baselines match Table 9, and each point uses one deterministic seed under its deployed protocol.

LLaDA region and budget grid. Table 15 varies the active region, order prior, and budget under Equation (8), with the EOS frontier at full canvas. Termination policies differ from the main comparisons: Table 9 gives the deployed-termination NFE for HumanEval and quality-tier MBPP at unchanged accuracy. The extended LLaDA frontier combines four-block budget sweeps with the full-canvas quality recipe at $\lambda = 1$ (Figure 7).

Table 15: LLaDA accuracy (%) / NFE across active regions, order priors, and coalition budgets at the deployed readiness rule; bold marks each task's accuracy maximum. The accuracy benefit of the order prior depends on the decoding region and task.

Region, λ	ε	HumanEval	MBPP	GSM8K	MATH-500	IFEval
full, λ=0	0.75	38.4 / 71.6	36.8 / 47.7	68.5 / 50.8	26.6 / 75.8	63.9 / 113.0
full, λ=1	0.75	**40.2** / 86.7	**39.2** / 57.7	**78.8** / 59.8	**34.0** / 84.7	**64.5** / 132.7
4blk, λ=0	0.75	38.4 / 59.1	29.0 / 21.8	78.3 / 49.6	28.4 / 65.4	58.6 / 96.9
4blk, λ=1	0.75	36.6 / 102.0	38.0 / 37.0	**78.8** / 75.5	30.8 / 90.7	63.9 / 124.7
4blk, λ=0	1.0	36.0 / 49.1	24.2 / 20.6	76.9 / 42.7	28.0 / 56.3	57.1 / 85.4
4blk, λ=1	1.0	37.2 / 85.4	38.0 / 32.6	78.2 / 64.6	30.8 / 78.1	62.8 / 113.2

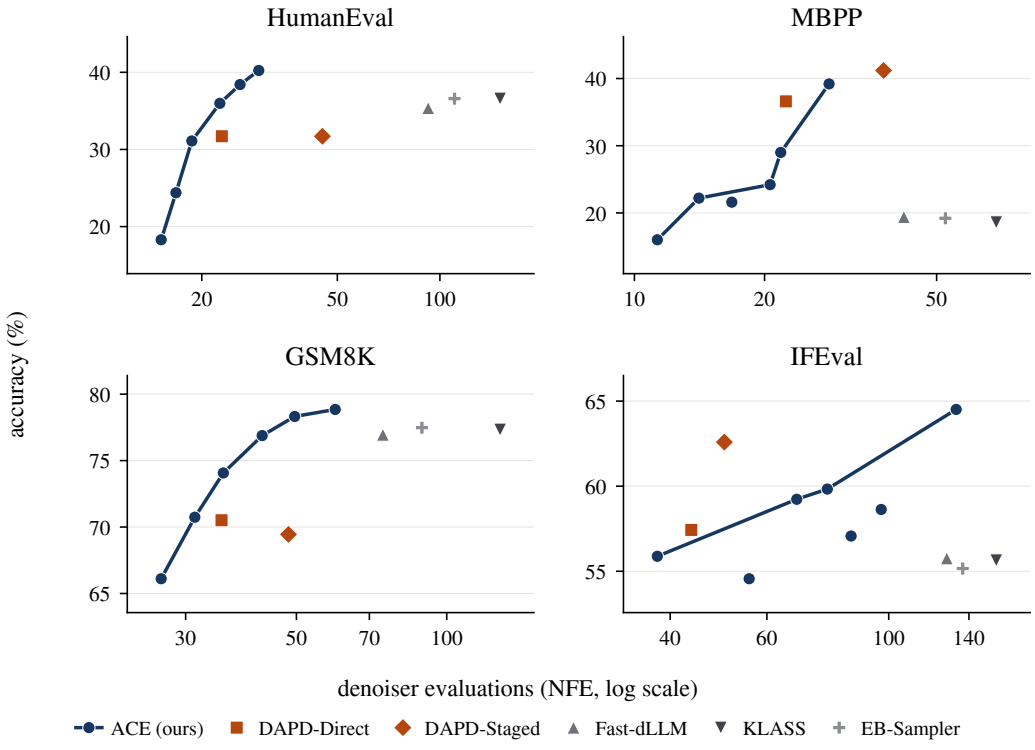

Figure 7: LLaDA quality–efficiency trade-offs across ACE settings and reproduced baselines. ACE improves both accuracy and evaluation cost on several tasks, while some baselines remain more accurate or use fewer calls.

Dream readiness grid. Complementing the fixed-readiness interaction sweep (Appendix E.3), Table 16 varies readiness at fixed task-specific interaction thresholds. Every listed gate is measured on all six benchmarks using the full-canvas ACE selector with $\varepsilon = 0$, multi-order arbitration, and deployed termination, prompts, and metrics; IFEval uses instruction-level loose accuracy. The threshold is $\tau = 0.1$ on MBPP and MATH-500 and 0.05 elsewhere.

Gates in the grid are constant, unlike the scheduled quality points defined in Appendix E.2 and included in Table 9. The extended Dream frontier varies readiness and also includes HumanEval's half-progress $\gamma = 0.9 \to 0.8$ schedule (Figure 8).

Table 16: Dream accuracy (%) / NFE across five readiness gates on all six benchmarks under the shared ACE selector. Stricter readiness generally buys accuracy with additional evaluations, but the returns flatten at different gates, so no single gate is best across benchmarks.

γ	HumanEval	MBPP	GSM8K	MATH-500	IFEval	BBH
0.4	30.5 / 54.9	16.2 / 11.4	39.1 / 25.2	13.2 / 41.7	45.0 / 78.6	37.3 / 16.9
0.5	39.0 / 62.4	32.2 / 14.2	48.7 / 36.9	21.0 / 54.7	49.2 / 97.5	41.4 / 21.0
0.6	48.8 / 79.2	47.0 / 17.1	54.2 / 48.9	25.0 / 65.9	49.3 / 114.3	48.3 / 25.6
0.7	50.0 / 90.4	53.0 / 20.7	55.8 / 61.7	29.8 / 80.7	50.5 / 126.0	53.5 / 30.4
0.8	59.8 / 103.4	57.2 / 25.2	58.5 / 74.3	31.0 / 96.0	50.4 / 135.9	56.4 / 36.1

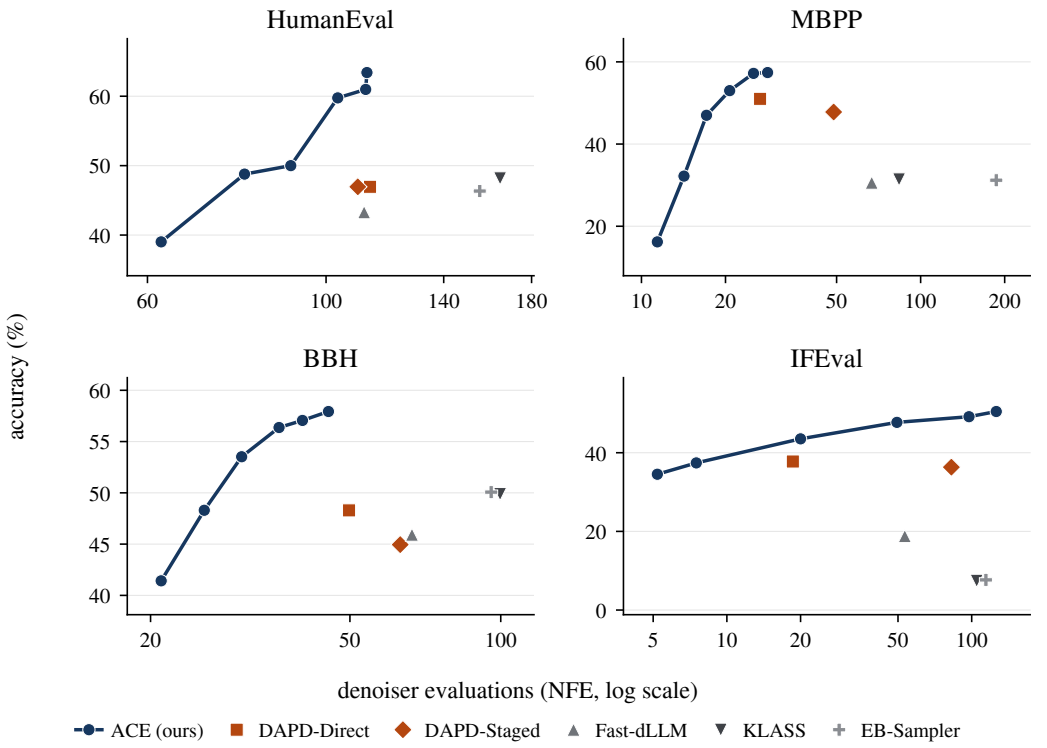

Figure 8: Dream quality–efficiency trade-offs across ACE readiness settings and reproduced baselines. ACE dominates the baselines on the displayed coding and reasoning tasks, while IFEval retains competing trade-offs.

Matched selector comparison. With interaction information fixed, this comparison tests one-shot versus sequential coalition construction across all three tiers (Table 17). Dream's shared ACE envelope at $\varepsilon = 0$ with four-order arbitration is compared against position-ordered greedy independent-set selection, matching interaction weights, readiness gates, thresholds, and termination. Tier settings are defined in Appendix E.2.

The table expands the six default rows plotted in Figure 3. Each comparison uses one deterministic seed; agreement is the share of problems with the same pass/fail or exact-match outcome. The main

figure reports accuracy differences in percentage points and relative NFE changes from the greedy control.

Table 17: Dream accuracy (%) / NFE for ACE and greedy independent-set selection under matched gates, interaction thresholds, and length policies. At the quality tier, ACE retains nearly identical accuracy with only modest additional evaluations.

Task	Tier	Independent set	ACE, $\varepsilon = 0$	Calls	Agreement
HumanEval	quality	63.4 / 112.3	63.4 / 112.4	+0.1%	100%
HumanEval	default	57.9 / 102.8	59.8 / 103.4	+0.6%	95.7%
HumanEval	throughput	53.0 / 90.2	50.0 / 90.4	+0.2%	93.3%
MBPP	quality	57.2 / 25.2	57.2 / 25.2	0	100%
MBPP	default	53.0 / 20.7	53.0 / 20.7	0	100%
MBPP	throughput	32.2 / 14.2	32.2 / 14.2	0	100%
GSM8K	quality	59.1 / 72.8	58.5 / 74.3	+2.1%	97.6%
GSM8K	default	55.3 / 60.2	55.8 / 61.7	+2.5%	94.8%
GSM8K	throughput	53.4 / 47.8	54.2 / 48.9	+2.3%	90.8%
MATH-500	quality	31.8 / 106.0	31.8 / 106.1	+0.1%	100%
MATH-500	default	31.2 / 95.9	31.0 / 96.0	+0.1%	99.8%
MATH-500	throughput	29.8 / 80.7	29.8 / 80.7	0	100%
IFEval	quality	47.7 / 46.9	47.7 / 49.5	+5.5%	90.2%
IFEval	default	41.5 / 19.3	43.5 / 20.0	+3.6%	90.8%
IFEval	throughput	36.8 / 6.4	37.4 / 7.5	+17.2%	94.5%
BBH	quality	57.8 / 44.8	57.9 / 45.3	+1.1%	96.5%
BBH	default	56.4 / 35.6	56.4 / 36.1	+1.4%	92.9%
BBH	throughput	53.1 / 29.9	53.5 / 30.4	+1.7%	88.1%

Block-count results and paired contrasts. Block count controls both blockwise order and the maximum candidate-pool size. Table 18 reports the values plotted in Figure 4 at $(\varepsilon, \lambda) = (0.75, 0)$, plus the throughput-budget comparison. Within each budget, only block count changes; $\lambda = 0$ and b is the block size on the 256-token canvas.

Table 18: LLaDA accuracy (%) / NFE as block count varies under fixed recipes. Finer blocks improve MBPP and IFEval at modest evaluation cost, but gains are not uniform across tasks.

Blocks (b)	ε	HumanEval	MBPP	GSM8K	MATH-500	IFEval
2 (128)	0.75	28.7 / 28.1	20.8 / 21.4	69.5 / 48.7	17.8 / 61.9	52.2 / 81.6
4 (64), deployed	0.75	38.4 / 25.9	29.0 / 21.8	78.3 / 49.6	28.4 / 65.4	58.6 / 96.9
8 (32)	0.75	37.8 / 27.2	37.6 / 23.2	79.0 / 52.4	30.4 / 68.1	61.9 / 104.4
16 (16)	0.75	39.6 / 29.5	37.4 / 26.3	78.7 / 57.9	30.8 / 73.2	65.2 / 111.0
4 (64), deployed	1.0	36.0 / 22.6	24.2 / 20.6	76.9 / 42.7	28.0 / 56.3	57.1 / 85.4
8 (32)	1.0	35.4 / 24.1	36.0 / 21.9	77.6 / 45.7	27.8 / 59.6	62.0 / 92.8

Paired comparisons support the accuracy gains from four to eight blocks on MBPP and IFEval at both budgets; further refinement benefits only IFEval significantly. These gains require additional evaluations, and matched timings confirm that eight blocks are slower despite similar per-call cost.

E.5 Statistical support for accuracy gains

We assess the uncached accuracy margins against two comparator groups: DAPD and the strongest reproduced baseline (Figure 9). The figure uses the quality tier throughout, conservatively on LLaDA BBH where the default tier scores higher. It compares both DAPD variants and, wherever an ACE tier holds the highest point estimate in Table 9, the strongest reproduced baseline.

Paired-bootstrap intervals support accuracy gains over DAPD across both backbones, with strictly positive 95% intervals in nine of the twelve displayed DAPD comparisons (Figure 9). Five of the ten accuracy leads also remain statistically supported against the strongest reproduced baseline. The distinction matters: LLaDA HumanEval has a supported gain over DAPD-Direct but an unresolved lead over the strongest unary baseline. The remaining five strongest-baseline leads are uncertain, including Dream MATH-500's one-problem edge; Fast-dLLM retains a significant advantage on LLaDA BBH.

We use 10,000 paired problem-level resamples and percentile intervals, preserving each metric's aggregation; for IFEval, each resample sums instruction-level numerators and denominators rather than averaging prompt ratios. These intervals describe evaluation-set uncertainty, not variation across decoding seeds. Contrasts are ordered by effect size within each backbone; dark-blue intervals exclude zero.

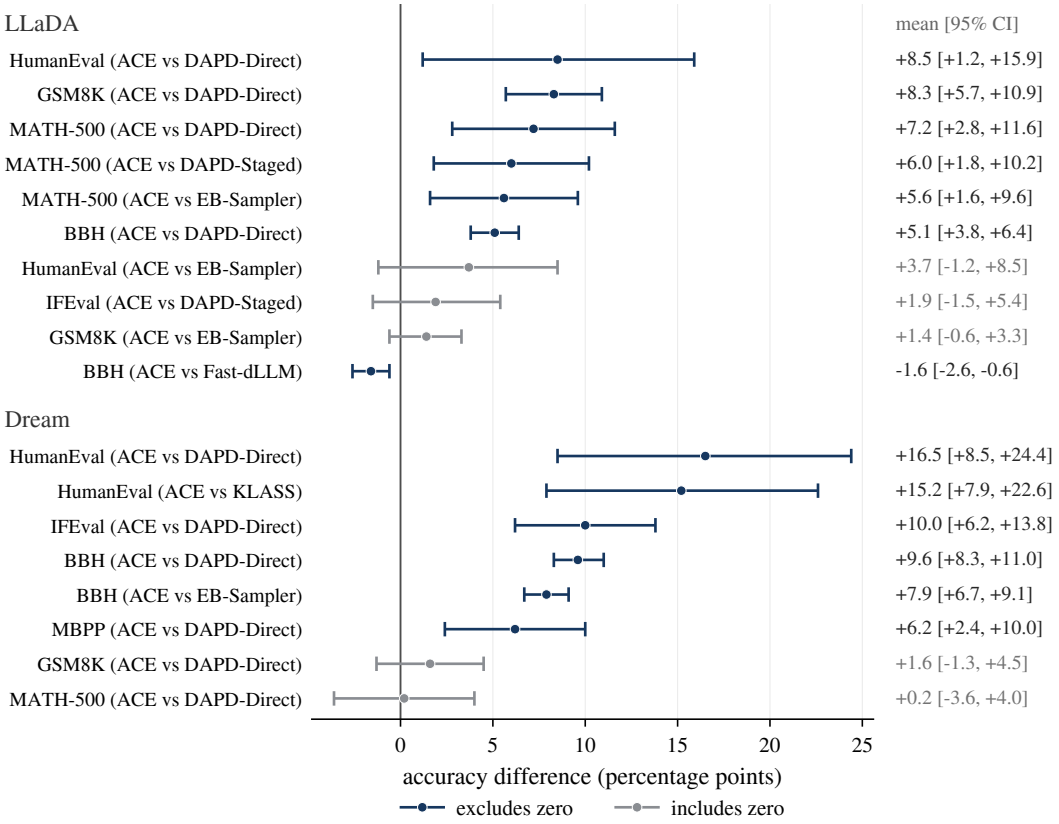

Figure 9: Paired-bootstrap 95% accuracy-margin intervals for quality-tier ACE against DAPD and the strongest reproduced baselines. ACE achieves statistically supported gains over DAPD across both backbones and over the strongest reproduced baseline in five benchmark–backbone settings.