CONTENT BEFORE POSITION: A BAYESIAN DECISION FRAMEWORK FOR DIFFUSION LANGUAGE MODEL DECODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion language models predict tokens across a canvas in parallel, yet decoding typically prioritizes positions by local confidence. This can overlook collective evidence for a token distributed across several candidate positions. We introduce content-before-position decoding, a principle that separates what to generate from where to commit it. A one-step Bayesian formulation connects maximum-confidence position-first decisions to an aggregate-evidence, token-first endpoint. Building on this framework, we propose COLA (**C**ollective-evidence **O**rdering and **L**ocal **A**cceptance), a training-free decoder that uses token-first proposals and validates placement locally. It coordinates parallel allocations from each model prediction and regulates commitment and revision across denoising steps. On DiffusionGemma, COLA modestly improves on the already high PARALLELBENCH quality of the model's default Entropy Bound (EB) decoder, while reducing model evaluations per generated token by 39.4% and latency by 33.2%. Across six downstream tasks, aggregate model evaluations per generated token fall by 73.7% relative to EB, with higher accuracy on both mathematics benchmarks and pass@1 gains of 20.7 and 10.5 percentage points on MBPP and HumanEval, respectively. These results establish content selection as a practical design axis for efficient parallel diffusion decoding.

1 INTRODUCTION

Diffusion language models (DLMs) make generation order a decoding decision. They predict vocabulary distributions across a canvas, enabling multiple tokens to be committed in one forward (Austin et al., 2021a; Campbell et al., 2022; Sahoo et al., 2024). As large-scale DLMs extend to reasoning and code generation (Nie et al., 2025; Ye et al., 2025; Google DeepMind, 2026), a central question is how to convert these parallel predictions into accurate, efficient generation. The decoder must jointly resolve what content to generate and where to place it.

Most existing decoding approaches follow a *position-first* strategy, prioritizing locations through confidence, entropy, stability, or calibrated scheduling (Wu et al., 2026; Cai & Li, 2026; Li et al., 2026b; Xu et al., 2026; Wang et al., 2026a; Mohamed et al., 2026). Revision mechanisms reconsider commitments as context evolves (Wang et al., 2026b; Zhai et al., 2026; Frković et al., 2026). These rules center on local reliability, while *content support and placement certainty are distinct*. A token can receive evidence from several candidate positions even when no single position has a dominant peak. The distinction is especially natural in constrained generation, where required content can be clearer than its final arrangement.

We introduce *content-before-position* as an alternative decoding principle: use collective evidence to decide what to generate before resolving where to commit it. A Bayesian one-step model expresses this separation through token-identity and placement costs. Under a uniform location prior, the resulting score family connects maximum-confidence position-first decisions to the *token-first* endpoint, which prioritizes aggregate token evidence before placement. Factoring aggregate support into peak confidence and effective positional support explains how distributed evidence can outweigh a sharper local prediction.

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.

We instantiate this framework as COLA (Collective-evidence Ordering and Local Acceptance), a training-free decoder combining token-first proposals with residual coverage, selected-token discounting, and uncertainty budgeting within each forward, and commitment revision across steps. On DiffusionGemma, COLA scores 93.47% across all 1,700 PARALLELBENCH examples, with 33.2% lower latency than the default Entropy Bound (EB) decoder. Across six downstream tasks, it improves mathematics and code accuracy, including pass@1 gains of 20.7 and 10.5 percentage points on MBPP and HumanEval relative to EB. Code-output analysis links the code gains to fewer syntax and indentation failures, while six-task ablations characterize the quality–cost effects of commitment control and parallel packing.

Our contributions are threefold:

- We formulate a Bayesian decision framework connecting maximum-confidence position-first and token-first endpoints through explicit token-identity and placement costs, explaining how aggregate support combines local confidence with evidence distributed across positions.
- We develop COLA, a training-free decoder that couples global token selection with local acceptance, coordinates allocations within each model prediction, and regulates commitment and revision across denoising steps for reliable parallel generation.
- We evaluate COLA on 17 PARALLELBENCH tasks and six downstream benchmarks, with trace analysis, code-error diagnostics, and ablations characterizing quality–cost trade-offs.

2 RELATED WORK

Parallel generation and decoding. Parallel prediction makes decoding policy central to diffusion language modeling. Discrete and continuous-time formulations provide the foundation (Austin et al., 2021a; Campbell et al., 2022; Sahoo et al., 2024), while large-scale models extend it to instruction following and reasoning (Nie et al., 2025; Ye et al., 2025; Google DeepMind, 2026). The resulting decoding problem involves both assessing individual predictions and managing their interactions.

Reliability of commitments. A common strategy selects positions whose predictions appear reliable. DiffusionGemma's default decoder (EB) adapts Entropy Bound to the model, combining entropy-bounded updates with adaptive stopping (Google DeepMind, 2026). Confidence-NFE instantiates an entropy-sum budget (Cai & Li, 2026), whereas Prophet uses top-two confidence gaps to trigger early all-in commitment (Li et al., 2026b). Temporal stability further informs commitment in KLASS and Stability-Weighted Decoding (SWD) (Kim et al., 2025; Wu & Huang, 2026). These approaches determine how much of the current prediction to trust before another forward.

Coordination and revision. Reliable individual predictions can still conflict when committed together. Mean-Field addresses this by refining position scores through pairwise interactions between predictive distributions (Zoabi et al., 2026); DAPD and DEMASK use attention-derived or learned dependencies to select compatible position sets (Kim et al., 2026; Ringel et al., 2026). When errors emerge after commitment, Reversible Diffusion Decoding (RDD) enables backtracking and confidence-guided remasking (Wang et al., 2026b). Coordination thus acts on simultaneous updates, while revision responds to their consequences as context develops.

Content cues in decoding. Existing methods incorporate content cues into candidate scoring or generation planning. UNCODE uses corpus-frequency calibration to reweight position-wise candidates (Huang et al., 2026), while PVF prioritizes semantic anchors when constructing a generation plan (Li et al., 2026a). We instead make token identity the primary decision variable. COLA selects a vocabulary item by aggregating evidence across positions, then determines and locally validates its placement.

3 CONTENT BEFORE POSITION

At denoising state x_t, let $p_i(v)$ be the model's predictive marginal and U the $n > 0$ eligible unresolved positions. Position-first rules rank locations by predictive certainty, using entropy or peak confidence,

then select their tokens. Content-before-position instead pools support for a token across locations before choosing its placement.

3.1 IDENTITY BEFORE PLACEMENT

To prioritize token identity, we pool evidence equally across eligible positions. Let Y_i have marginal p_i and independently sample an auxiliary location I with $\pi_i = 1/n$, conditional on x_t. Then $V = Y_I$ represents content drawn from this pool:

$$q(i, v) = \pi_i p_i(v), \qquad q(v) = \sum_{i \in U} \pi_i p_i(v). \tag{1}$$

Prioritizing identity over placement gives costs $c_{\text{tok}} \geq c_{\text{pos}} > 0$:

$$\begin{aligned} \mathcal{L}((\hat{v}, \hat{i}), (V, I)) = c_{\text{tok}} \mathbf{1}[\hat{v} \neq V] \\ + c_{\text{pos}} \mathbf{1}[\hat{v} = V, \hat{i} \neq I], \end{aligned} \tag{2}$$

This auxiliary loss expresses a proposal preference, while $1 - p_i(v)$ measures predicted error at the actual placement.

Proposition 1 (One-step Bayes action)**.** *For $\rho = c_{\text{tok}}/c_{\text{pos}}$, the Bayes action maximizes*

$$S_\rho(v, i) = (\rho - 1)q(v) + q(i, v). \tag{3}$$

Its expected loss is

$$\mathcal{E}(v, i) = c_{\text{tok}} - (c_{\text{tok}} - c_{\text{pos}})q(v) - c_{\text{pos}} q(i, v), \tag{4}$$

giving the stated score by removing the action-independent term.

Define aggregate and peak evidence as

$$A(v) = \sum_{i \in U} p_i(v), \qquad P(v) = \max_{i \in U} p_i(v). \tag{5}$$

Under the uniform prior, maximizing over placement gives

$$G_\rho(v) = (\rho - 1)A(v) + P(v). \tag{6}$$

At $\rho = 1$, peak evidence alone recovers maximum-confidence selection; increasing ρ emphasizes aggregate evidence. The identity-priority limit $\rho \to \infty$ maximizes $A(v)$, breaks ties by $P(v)$, then selects the highest-probability placement. For $p(a) = (.2, .6, .6)$ and $p(b) = (.8, .1, .1)$, $A(a) = 1.4 > A(b) = 1$ but $P(a) = .6 < P(b) = .8$, so a outranks b when $\rho > 1.5$. COLA adopts this token-first limit, adding residual support and discounting for successive allocations.

By linearity, $A(v)$ is expected token multiplicity, with no independence assumption. For $P(v) > 0$, effective support decomposes this evidence as

$$R(v) = A(v)/P(v) \in [1, n], \qquad A(v) = P(v)R(v). \tag{7}$$

3.2 EVIDENCE ACROSS SUCCESSIVE COMMITMENTS

Alternative candidates can support content whose placement remains unresolved. Residual support prevents a position allocation from immediately erasing this evidence before predictions refresh. This is an allocation choice, not posterior conditioning on the commitment.

For selection scores $\tilde{p}$ initialized to p and positions C committed to x_i, define

$$A_\beta(u) = \sum_{j \in U} \tilde{p}_j(u) + \beta \sum_{\substack{j \in C \\ x_j \neq u}} p_j(u). \tag{8}$$

The unchanged predictions can still favor content just allocated. Accepting (v, i) therefore moves i to C and discounts only v at remaining positions by $w_{ij} \in (0, 1]$. With raw predictions fixed, the score changes are

$$\begin{aligned} \Delta A_\beta(u) &= \beta p_i(u) - \tilde{p}_i(u), && u \neq v, \\ \Delta A_\beta(v) &= -\tilde{p}_i(v) - \sum_{j \in U \setminus \{i\}} (1 - w_{ij}) \tilde{p}_j(v). \end{aligned} \tag{9}$$

With $\tilde{p} \leq p$ and $\beta = 1$, unselected scores cannot decrease; the selected score decreases if $\tilde{p}_i(v) > 0$. Without discounting, unselected scores are conserved exactly. Section 4 specifies the KL-based discount factors.

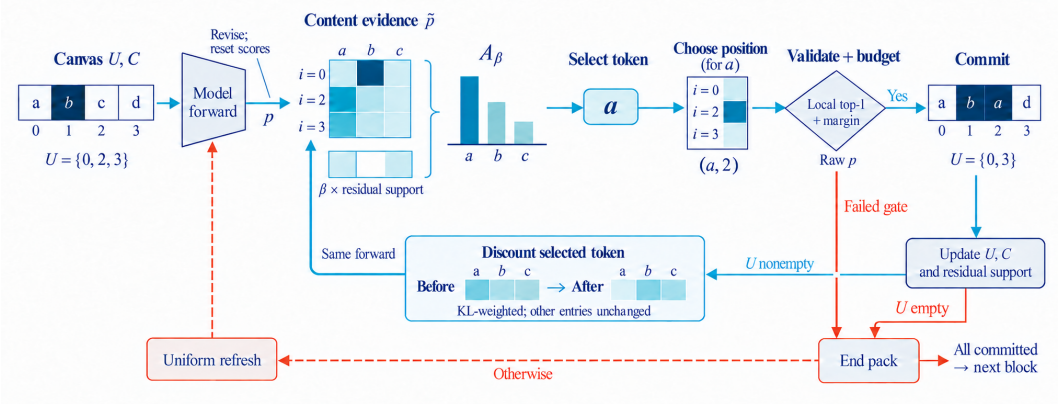


Figure 1: COLA's nested decoding loops. Residual-augmented evidence A_β selects a token before placement validation and uncertainty budgeting. Commitments update residual support and discount the selected token within the same forward. A failed gate or empty eligible set ends packing; an incompletely committed canvas is refreshed for the next forward. Dark canvas cells are committed; light cells contain unresolved inputs. Scores are schematic.

3.3 Local Reliability of Parallel Commitments

Local acceptance tests relative separation (Section 4.1), whereas $1-p_i(v)$ measures absolute predicted uncertainty. For a distinct-position pack B, let $r(B) = \sum_{(v,i)\in B}(1-p_i(v))$. If p_i are true conditional marginals and B is determined by x_t, the error count M_B satisfies

$$\mathbb{E}[M_B \mid x_t] = r(B), \qquad \Pr(M_B \geq 1 \mid x_t) \leq \min\{1, r(B)\}, \tag{10}$$

by linearity and the union bound, without position independence. With model predictions, $r(B)$ is an uncertainty proxy. The controller admits the first locally admissible pair (v_1, i_1), then budgets additions by η, yielding $r(B) \leq \max\{\eta, 1 - p_{i_1}(v_1)\}$ for nonempty packs. Thus η limits accumulation after the first acceptance, whose cost can exceed it.

4 COLA: Collective-Evidence Ordering and Local Acceptance

COLA separates content selection from placement validation and coordinates multiple commitments from each model forward. It maintains raw probabilities for local decisions and adjustable scores for content priority. The resulting pack updates the canvas; a fresh forward then determines which commitments survive and which positions need revision. Figure 1 summarizes these nested loops.

4.1 Token-First Proposals and Local Acceptance

For each forward, let C contain surviving commitments and U the eligible unresolved positions, as determined in Section 4.3. Initialize selection scores $\tilde{p}_i(v) = p_i(v)$ and use the residual-support score A_β from Equation 8:

$$v^* = \arg\max_v A_\beta(v), \qquad i^* = \arg\max_{i\in U} \tilde{p}_i(v^*). \tag{11}$$

Every vocabulary item competes before a position is chosen. The score combines unresolved-position evidence with weight-β support from committed positions for identities other than their committed tokens. Setting $\beta = 0$ recovers the unresolved-position aggregate.

Let $p_1(i)$ and $p_2(i)$ be the largest and second-largest probabilities at position i. A pair is locally admissible when v^* is the raw local top-one token and its ratio margin reaches τ_{cm}:

$$m(i^*) = \frac{p_1(i^*) - p_2(i^*)}{\max(p_2(i^*), 10^{-12})}. \tag{12}$$

Thus adjusted scores rank proposals, while raw probabilities validate their placements. The margin threshold τ_{cm} controls admission, independently of the theoretical cost ratio ρ. Commitment also

respects the packing budget in Section 4.3. Appendix J derives a sufficient probability bound for acceptance.

4.2 Coordinating parallel commitments

Committing (v^*, i^*) moves i^* from U to C and adds its alternative-token evidence to residual support. Because predictions remain fixed within the forward, they can continue favoring the content just allocated. We discount that token at competing locations using directional distribution similarity (Zoabi et al., 2026):

$$s_{i^*,j} = \exp\big[-\gamma D_{\mathrm{KL}}\big(p_{i^*} \,\|\, p_j\big)\big]\,. \tag{13}$$

For each remaining $j \in U$, update the current selection score:

$$\tilde{p}_j(v^*) \leftarrow \tilde{p}_j(v^*)\max(\epsilon, 1 - \lambda s_{i^*,j}). \tag{14}$$

Here $\lambda \geq 0$ controls suppression strength, $\gamma \geq 0$ its sensitivity to KL divergence, and $0 < \epsilon \leq 1$ the floor. Similar distributions induce stronger suppression, reducing the selected token's priority at competing placements. Other token entries remain unchanged, preserving their relative scores. The scores stay unnormalized; both KL and local validation use the raw distributions.

Equation 11 is then reapplied with updated U, C, and selection scores, without another model forward. A failed local test or budget check ends packing rather than trying lower-ranked candidates; an empty U also ends the pack. Discounts accumulate within this loop and reset at the next forward.

4.3 Commitment control across denoising steps

The controller determines when to refresh predictions and which commitments to retain under the refreshed context. Within a forward, it accumulates the pack uncertainty $r(B)$ defined in Section 3.3. After admitting the first locally admissible pair, it stops before an addition would make $r(B) > \eta$. This first-pair exception permits progress even when that pair exceeds the budget; all pairs still pass the same local gates. The budget resets at every forward.

Before forming a new pack, the fresh predictions re-evaluate existing commitments. A position reopens if its committed token is no longer local top one or its ratio margin falls below τ_{rb}. Surviving positions form C; unresolved positions form U, excluding those just reopened. This exclusion defers recommitment until the next forward has processed their refreshed inputs. Residual support is recomputed from the current raw probabilities over C, then updated as the pack grows.

After packing, committed tokens remain fixed and all unresolved positions receive uniformly sampled vocabulary tokens for the next forward. Previous processed logits pass through the model's self-conditioning interface and reset at block boundaries. A fully committed canvas completes the block; otherwise denoising continues up to the configured forward cap. Algorithm 1 summarizes the loop, and Appendix I records its settings.

5 Experimental Setup

Backbone. We use DiffusionGemma 26B-A4B (Google DeepMind, 2026), a diffusion language model with uniform token corruption, as the primary evaluation backbone. We also evaluate Nemotron-Labs-Diffusion-8B, with results reported in Appendix L.

Tasks. We evaluate on ParallelBench (Kang et al., 2026), covering waiting-line manipulation, text generation, and puzzles. The downstream suite spans mathematical reasoning with GSM8K and MATH-500 (Cobbe et al., 2021; Hendrycks et al., 2021), code generation with HumanEval and MBPP (Chen et al., 2021; Austin et al., 2021b), instruction following with IFEval (Zhou et al., 2023), and summarization with XSum (Narayan et al., 2018).

Baselines. Both suites compare COLA with DiffusionGemma's default decoder, the official model-specific adaptation of Entropy Bound (EB) (Google DeepMind, 2026), and four alternatives: Confidence-NFE (Conf.) for entropy-budgeted commitment (Cai & Li, 2026), Prophet for early stopping (Li et al., 2026b), Mean-Field (MF) for parallel coordination (Zoabi et al., 2026), and RDD for reversible decoding (Wang et al., 2026b). The four alternatives use adapters to the same backbone interface. Further experimental details appear in Appendices E, F, and I.

Table 1: Decoding quality and computational cost on PARALLELBENCH. EB denotes DiffusionGemma's default decoder, the official model-specific adaptation of Entropy Bound. Conf., Prophet, MF, and RDD denote Confidence-NFE, Prophet, Mean-Field, and Reversible Diffusion Decoding. Waiting-line and puzzle rows report accuracy; text rows report LanguageTool-clean outputs (%). Overall score averages the 17 task scores; NFE/100 counts forwards per 100 generated tokens. Bold marks the best value, including ties.

Task	EB	Conf.	Prophet	MF	RDD	**COLA**
Waiting-line						
Copy	**100**	**100**	85	99	**100**	**100**
Reverse	**100**	**100**	67	77	83	**100**
Sort	**7**	6	6	**7**	**7**	**7**
Shuffle	**100**	98	5	93	82	**100**
Insert by index	**98**	93	71	73	77	**98**
Insert random	**100**	98	20	83	86	**100**
Remove by index	**100**	**100**	83	94	98	**100**
Remove random	**100**	98	36	96	96	**100**
Replace by index	**100**	**100**	91	**100**	**100**	**100**
Replace random	**100**	92	28	68	71	**100**
Text generation						
Paraphrasing	93	92	39	79	91	**95**
Summarization	**99**	85	35	63	73	98
Words to sentence (easy)	99	93	16	81	89	**100**
Words to sentence (medium)	**99**	91	49	58	68	98
Words to sentence (hard)	97	77	62	45	58	**99**
Puzzles						
Latin square	**100**	78	82	70	92	**100**
Sudoku	91	84	19	62	77	**94**
Overall score ↑	93.1	87.4	46.7	73.4	79.3	**93.5**
Overall NFE/100 tokens ↓	69.3	115.7	43.7	**34.1**	40.4	42.0

6 MAIN RESULTS

6.1 QUALITY AND EFFICIENCY ON PARALLELBENCH

COLA modestly improves on EB's already high quality while substantially reducing decoding cost (Table 1). In the paired comparison, the overall score rises from 93.12% to 93.47%, while NFE/100 falls from 69.28 to 41.96 (39.4%). Measured mean latency also falls by 33.2%, from 3.105 to 2.074 seconds per example. The small quality gain comes from text generation and Sudoku, including a rise from 97% to 99% on Words to sentence (hard). Scores remain unchanged on all ten waiting-line tasks and perfect on Latin squares. MF and RDD achieve lower NFE/100 than COLA, but their overall scores are substantially lower, at 73.4% and 79.3%, respectively.

6.2 PERFORMANCE ACROSS DOWNSTREAM TASKS

COLA reduces aggregate NFE/100 by 73.7% relative to EB across the six downstream tasks, while achieving the highest scores among the evaluated decoders on all four mathematics and code benchmarks (Table 2). The largest gains over EB occur in code generation: pass@1 improves by 20.7 percentage points on MBPP and 10.5 points on HumanEval. Mathematical accuracy also rises, by 1.6 points on GSM8K and 2.6 points on MATH-500. Its aggregate NFE/100 of 10.5 is the lowest among the evaluated decoders, compared with EB's 40.1.

On XSum, COLA achieves the highest BERTScore F1, narrowly ahead of EB at 88.49 versus 88.44; the paired 95% interval for this difference includes zero. IFEval instruction-level strict accuracy reaches 87.8% versus EB's 89.7%. Paired bootstrap comparisons yield positive 95% intervals for the GSM8K and MBPP gains; the MATH-500 and HumanEval intervals include zero. Table 10 in Appendix H reports the paired counts and intervals at the fixed generation seeds.

Table 2: Downstream performance on mathematical reasoning (GSM8K and MATH; accuracy), code generation (HumanEval and MBPP; pass@1), instruction following (IFEval; instruction-level strict accuracy), and summarization (XSum; BERTScore F1). Baselines and abbreviations follow Table 1. Quality scores are scaled by 100; NFE/100 aggregates forward evaluations per 100 generated tokens. Bold marks column-best values.

Decoder	GSM8K Acc.	MATH Acc.	HumanEval pass@1	MBPP pass@1	IFEval Inst. strict	XSum BERTScore	NFE/100 ↓
EB	94.9	42.6	71.1	57.1	**89.7**	88.44	40.1
Conf.	93.4	42.8	34.2	36.2	80.1	88.38	50.5
Prophet	89.1	40.0	43.9	45.1	66.1	88.11	62.6
MF	91.9	40.8	47.4	39.3	78.3	88.02	25.8
RDD	92.6	41.4	51.8	43.8	83.9	88.23	33.4
COLA	**96.5**	**45.2**	**81.6**	**77.8**	87.8	**88.49**	**10.5**

Table 3: Decision-order comparisons on PARALLELBENCH. Simplified serial controls commit one token per forward; the COLA controller enables parallel commitment.

Commitment setting	Proposal rule	Score (%) ↑	NFE/100 ↓
Simplified serial	Position-first (entropy)	93.00	218.95
Simplified serial	Token-first	93.12	221.07
COLA controller	Position-first (entropy)	93.06	90.80
COLA controller	Token-first (COLA)	93.47	41.96

7 ANALYSIS

7.1 DECISION ORDER AND COMMITMENT SUPPORT

Table 3 compares token-first with position-first (entropy), which selects the lowest-entropy position and then its top token. In simplified serial decoding, each commitment is followed by a fresh model evaluation. The rules reach similar accuracy despite different commitment sequences on every example. Their forward counts are identical; the small NFE/100 difference reflects output length.

The distinction becomes pronounced with parallel commitment. Under the COLA controller, token-first reduces NFE/100 by 53.8% at modestly higher quality. This contrast suggests that collective evidence is especially useful for coordinating multiple commitments from a shared prediction: broad support identifies content that can be allocated across positions, while the controller checks each placement. The benefit appears primarily in how efficiently reliable commitments are assembled, rather than in serial accuracy alone.

Full-decoder traces support this interpretation. Selected tokens differ from those of the globally highest-probability pairs in 50.69% of commitments, with mean effective support of 8.33 positions and 5.32 accepted tokens per forward. Thus, aggregate evidence frequently changes content selection while enabling multi-token updates. Appendix D details the protocols and paired outcomes.

7.2 CHARACTERIZING THE CODE-GENERATION GAINS

The clearest code-level change is a reduction in syntax and indentation failures. We analyze all 564 paired outputs from Table 2 using the original extractor, test outcomes, and error logs, without repairing or rescoring outputs. On MBPP, these failures fall from 139/450 for EB to 22/450 for COLA; 102 of the 111 paired wins correspond to an EB syntax or indentation failure. On HumanEval, failures fall from 27/114 to 15/114, with the same correspondence in 22 of the 26 wins. Most paired code gains therefore coincide with avoiding malformed programs.

Among examples where both extracted programs compile, wins and losses are 9/8 on MBPP (300 examples) and 4/3 on HumanEval (76 examples). This output-conditioned subset characterizes the observed gains rather than separately measuring semantic reasoning. Residual Markdown fences occur in eight of the 18 MBPP regressions and ten of the 14 HumanEval regressions under the

Table 4: Text diagnostics from the Table 1 predictions. LanguageTool cleanliness uses all 500 text examples; requested-word coverage uses 300 inclusion examples, including 100 hard examples. Adjacent repeats count events over the 500 outputs.

Metric	EB	COLA
LanguageTool clean (%)	97.4	98.0
Requested-word coverage (%)	95.7	96.3
Hard requested-word coverage (%)	94.0	94.0
Adjacent repeated-word events	1	0

unchanged extractor, identifying output formatting as a remaining source of failures. Section 8 examines how component removal changes correctness and forward cost.

7.3 Task-dependent efficiency

Waiting-line tasks provide the largest wall-clock savings on ParallelBench. Accuracy remains at 905/1,000 while mean forwards per example fall from 18.883 to 4.751 and mean latency from 3.987 to 1.238 seconds. On puzzles, correctness rises from 191/200 to 194/200 while latency falls from 1.752 to 1.517 seconds. Text generation trades more computation for its small quality gain: LanguageTool-clean outputs increase from 487/500 to 490/500, while mean forwards per example rise from 7.632 to 17.232 and latency from 1.881 to 3.968 seconds. The overall wall-clock reduction is therefore driven primarily by waiting-line tasks. These per-example measurements come from the uninstrumented paired run and complement the token-normalized costs in Table 1.

7.4 Text quality and instruction constraints

Table 4 separates grammatical cleanliness, requested-content coverage, and repetition. COLA improves overall requested-word coverage alongside grammatical quality, with unchanged hard-word coverage and fewer adjacent repeats. The gains therefore extend beyond grammatical cleanliness, although the hardest content constraints remain equally challenging.

On IFEval, EB and COLA pass 689 and 674 of 768 instruction checks. The shortfall spans seven families, none contributing more than three checks; change-case and language totals are unchanged. At prompt level, COLA has 12 paired wins and 28 losses, with a 95% interval of $[-5.7, -0.8]$ percentage points. These results identify distributed instruction-compliance errors rather than a single dominant constraint family. XSum repetition is rare for both methods: mean adjacent-word repetition is 0.0239% for EB and 0.0156% for COLA, and repeated four-gram rates are 0.0042% for both, using case-folded alphabetic words. This complements their closely matched BERTScore F1 values.

8 Ablation Studies

Table 5 uses a progressive, cumulative ablation to examine how COLA sustains quality under parallel commitment. We first remove the pack-level risk budget, then relax the commit/rollback ratio margins, separating aggregate uncertainty control from pair-level acceptance and revision. Next, we remove residual coverage and selected-token KL to examine allocation coordination with these controls relaxed. Finally, one-token commitment tests the cost of refreshing predictions after each allocation. Each stage retains earlier removals; token-first proposals and local top-one checks remain throughout. Appendix I provides the protocol.

Progressively removing these controls lowers NFE/100 across all tasks, while quality follows task-dependent trade-offs. Risk-budget removal weakens code generation and GSM8K; relaxing margins further helps instruction following but hurts mathematics. Coordination is similarly conditional: removing residual coverage degrades both code tasks but improves MATH, whereas removing selected-token KL particularly harms MBPP and GSM8K while helping HumanEval. Each component therefore supports quality on particular tasks within this sequence. Together, they regulate how aggressively shared predictions are converted into commitments, balancing computational savings against task-specific sensitivity to acceptance, revision, and allocation order.

Table 5: Ablation results across six downstream tasks. Removals accumulate across rows; the final row restricts commitment to one token per forward. Cells report Score ↑ / NFE per 100 generated tokens ↓, using the metrics in Table 2. *One of 450 examples did not converge and is counted as incorrect.

Configuration	GSM8K	MATH	HumanEval	MBPP	IFEval	XSum
Full COLA	96.5/8.2	45.2/6.2	81.6/6.7	77.8/12.1	87.8/16.6	88.49/36.7
− Risk budget	96.0/7.1	45.4/4.9	75.4/6.4	75.1/9.6	86.2/12.7	88.48/25.6
− Ratio margins	96.3/6.8	43.2/4.5	76.3/6.2	75.3/8.5	87.4/10.8	88.44/21.8
− Residual coverage	95.4/6.5	44.2/4.4	66.7/6.1	71.8/8.0	87.4/9.7	88.39/20.3
− Selected-token KL	87.3/5.9	44.6/4.2	71.1/4.9	67.1/7.6	82.3/8.4	88.17/18.9
One token per forward	96.6/92.2	44.8/85.2	78.9/106.9	76.0*/105.6	88.4/101.2	88.49/133.1

The one-token control recovers quality across tasks by refreshing predictions more frequently, but incurs 7.1–21.9× the NFE/100 of the preceding parallel variant. It still trails full COLA on both code tasks despite its much higher forward cost. These results highlight the role of control and coordination in making parallel commitment effective: preserving strong generation quality without paying for a model refresh after every allocation.

9 Discussion

Content-before-position decoding reframes parallel generation as two distinct decisions: what content is supported by the current canvas and where it can be committed reliably. The Bayesian score family formalizes this separation through token-identity and placement costs. COLA realizes this principle through token-first proposals, coordinated parallel allocation, and local acceptance. Empirically, its selected tokens often differ from those of the globally strongest position–token pairs, showing that cross-position support provides information beyond local peak confidence. This supports a division of labor: aggregate evidence guides proposal order, while local reliability governs commitment.

Practically, this separation yields a favorable quality–efficiency trade-off. On PARALLELBENCH, COLA reduces model evaluations and latency while maintaining the already high benchmark quality; across downstream tasks, it combines substantially lower normalized forward cost with stronger mathematics and code performance. The code gains are associated mainly with fewer syntax and indentation failures. Progressive ablations further show that parallel commitment provides the main forward savings, while commitment control and allocation coordination help preserve quality when multiple tokens share one model prediction. Together, these results support COLA as a practical realization of content-before-position decoding.

Several natural extensions remain. The Bayesian analysis concerns a single commitment at a fixed denoising state, whereas a full diffusion trajectory couples decisions across successive evaluations. Extending the framework to sequential decisions could connect the theory more directly to long-horizon decoding. The main quantitative study centers on DiffusionGemma and its uniform-corruption interface, so broader evaluation across diffusion model families would further test generality. Finally, the preferred quality–cost operating point is task dependent, motivating adaptive commitment control that responds to task structure and generation context.

10 Conclusion

We introduced the content-before-position principle, with a Bayesian formulation that makes collective token support a first-class criterion for generation order. COLA realizes this principle through token-first proposals and locally validated parallel commitments. On DiffusionGemma, it substantially reduces decoding cost alongside a modest quality gain on ParallelBench and stronger mathematics and code generation. These results highlight content selection as a central design axis for diffusion decoding, alongside placement reliability and commitment scheduling.

AI USE STATEMENT

Generative AI tools were used in the preparation of this work to assist with language editing and polishing, the generation and debugging of experimental code, and the statistical summarization of experimental results. All AI-assisted outputs were carefully reviewed, verified, and, where necessary, revised by the authors. The authors take full responsibility for the content, methodology, experimental results, and conclusions presented in this paper.

REFERENCES

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, volume 34, 2021a. URL https://proceedings.neurips.cc/paper/2021/hash/958c530554f78bcd8e97125b70e6973d-Abstract.html.

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. URL https://arxiv.org/abs/2108.07732.

Changxiao Cai and Gen Li. Confidence-based decoding is provably efficient for diffusion language models. *arXiv preprint arXiv:2603.22248*, 2026. URL https://arxiv.org/abs/2603.22248.

Andrew Campbell, Joe Benton, Valentin De Bortoli, Tom Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/b5b528767aa35f5b1a60fe0aaeca0563-Abstract.html.

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

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. URL https://arxiv.org/abs/2110.14168.

Stipe Frković, Metod Jazbec, Dan Zhang, Christian A. Naesseth, Ilija Bogunovic, and Eric Nalisnick. Re-evaluating confidence remasking in masked diffusion language models. In *Advances in Neural Information Processing Systems*, 2026. URL https://arxiv.org/abs/2606.12232. Accepted for publication.

Google DeepMind. DiffusionGemma model card. Google AI for Developers, 2026. URL https://ai.google.dev/gemma/docs/diffusiongemma/model_card. Accessed 2026-09-25.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Pengcheng Huang, Tianming Liu, Zhenghao Liu, Yukun Yan, Shuo Wang, Tong Xiao, Zulong Chen, and Maosong Sun. Empirical analysis of decoding biases in masked diffusion models. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6853–6876, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.311. URL https://aclanthology.org/2026.acl-long.311/.

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. URL https://arxiv.org/abs/2510.04767.

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*, 2026. URL https://arxiv.org/abs/2603.12996.

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*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/8514c1f12dcdd061ac8521feb78f28ca-Paper-Conference.pdf.

Miao Li, Hanyang Jiang, Sikai Cheng, Hengyu Fu, Yuhang Cai, Baihe Huang, Tinghan Ye, Xuanzhou Chen, and Pascal Van Hentenryck. Plan, verify and fill: A structured parallel decoding approach for diffusion language models, 2026a. URL https://arxiv.org/abs/2601.12247.

Pengxiang Li, Yefan Zhou, Dilxat Muhtar, Lu Yin, Shilin Yan, Li Shen, Yi Liang, Soroush Vosoughi, and Shiwei Liu. Diffusion language model knows the answer before it decodes. In *International Conference on Learning Representations*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/daadbff4d4ea884ca3d9d389a1dfc61c-Abstract-Conference.html.

Amr Mohamed, Yang Zhang, Michalis Vazirgiannis, and Guokan Shang. Fast-decoding diffusion language models via progress-aware confidence schedules. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 35793–35807, 2026. doi: 10.18653/v1/2026.findings-acl.1782. URL https://aclanthology.org/2026.findings-acl.1782/.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1797–1807, 2018. doi: 10.18653/v1/D18-1206. URL https://aclanthology.org/D18-1206/.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/48b383b24230e0e6e649d9c98dae4d8c-Abstract-Conference.html.

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

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/eb0b13cc515724ab8015bc978fdde0ad-Abstract-Conference.html.

Kangyu Wang, Zhiyun Jiang, Haibo Feng, Weijia Zhao, Lin Liu, Jianguo Li, Zhenzhong Lan, and Weiyao Lin. CreditDecoding: Accelerating parallel decoding in diffusion large language models with trace credit. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11105–11123, 2026a. doi: 10.18653/v1/2026.acl-long.509. URL https://aclanthology.org/2026.acl-long.509/.

Xinyun Wang, Min Zhang, Sen Cui, Zhikang Chen, Bo Jiang, Kun Kuang, and Mingbao Lin. Reversible diffusion decoding for diffusion language models. *arXiv preprint arXiv:2602.00150*, 2026b. URL https://arxiv.org/abs/2602.00150.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/5d8d4e6061c3ba96c240b7fa1ae3471d-Abstract-Conference.html.

Yue Wu and Jian Huang. Stability-Weighted Decoding for diffusion language models, 2026. URL https://arxiv.org/abs/2604.17068.

Fan Xu, Huixuan Zhang, and Xiaojun Wan. DecoCal: Decoding with calibration in diffusion large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11866–11880, 2026. doi: 10.18653/v1/2026.acl-long.545. URL https://aclanthology.org/2026.acl-long.545/.

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

Kevin Zhai, Sabbir Mollah, Zhenyi Wang, and Mubarak Shah. CORE: Context-robust remasking for diffusion language models. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.04096.

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. URL https://arxiv.org/abs/2311.07911.

Tamim Zoabi, Ameen Ali, Liran Ringel, and Lior Wolf. Mean-field parallel decoding for discrete diffusion language models. *arXiv preprint arXiv:2606.15805*, 2026. URL https://arxiv.org/abs/2606.15805.

A Decoding Algorithm

Algorithm 1 gives the DiffusionGemma decoder in Section 4.3. Each forward refreshes the raw probabilities used for acceptance and revision; within-forward score updates coordinate subsequent proposals.

Algorithm 1 COLA on one DiffusionGemma canvas

Initialize uniform canvas x, committed set $C = \varnothing$, self-conditioning $h = \varnothing$
while forward budget remains **do**
  if active canvas complete **then**
    break
  end if
  Compute processed logits using (x, h); obtain raw probabilities p
  Save processed logits as next-step h; initialize $\tilde{p} \leftarrow p$
  Reopen $R \subseteq C$ by raw top-one and ratio-margin checks; $C \leftarrow C \setminus R$
  $U \leftarrow$ eligible unresolved positions excluding R; $B \leftarrow \varnothing$
  while $U \neq \varnothing$ **do**
    Form $A_\beta(v)$ from unresolved scores and residual alternative-token support
    $v^* \leftarrow \arg\max_v A_\beta(v)$; $i^* \leftarrow \arg\max_{i \in U} \tilde{p}_i(v^*)$
    if raw local acceptance fails, or $B \neq \varnothing$ and $r(B) + 1 - p_{i^*}(v^*) > \eta$ **then**
      break
    end if
    Commit (v^*, i^*); update C, U, B and residual support
    Discount $\tilde{p}_j(v^*)$ for $j \in U$ (Eq. 14)
  end while
  Keep committed tokens fixed; uniformly refresh unresolved positions, including R
end while
Return the completed canvas, or report budget exhaustion

The first locally admissible pair is accepted even when its uncertainty exceeds η. Reopened positions are excluded from the current pack and become eligible after the next forward. A new canvas resets commitments and self-conditioning. The supplementary Nemotron instantiation instead uses permanent masked-block commitments with $\beta = 0$, followed by the companion and fallback paths specified in Appendix I.

B Proofs

Proof of Proposition 1. For action (v, i), the content-error event has probability $1 - q(v)$ and the correct-token, wrong-location event has probability $q(v) - q(i, v)$. The posterior expected loss is therefore $c_{\text{tok}}(1 - q(v)) + c_{\text{pos}}(q(v) - q(i, v))$, which rearranges to Equation 4. Its first term is action independent; dividing the remaining terms by c_{pos} yields Equation 3.

Expected multiplicity. Linearity of expectation gives $\mathbb{E}[N_v \mid x_t] = \sum_i \Pr(Y_i = v \mid x_t) = A(v)$ without assuming independence among positions. Under the uniform selector, $\Pr(V = v \mid x_t) = A(v)/n$ and Bayes' rule gives $\Pr(I = i \mid V = v, x_t) = p_i(v)/A(v)$, whose maximizer is the conditional placement rule.

Evidence transfer within a forward. After accepting (v, i), an unselected identity $u \neq v$ loses $\tilde{p}_i(u)$ from the unresolved-position sum and gains $\beta p_i(u)$ in residual support, giving the first line of Equation 9. The selected identity gains no residual term at i and is discounted at each remaining position, giving the second line. Since every discount lies in $(0, 1]$, $\tilde{p} \leq p$ throughout the pack. Hence $\beta = 1$ preserves or increases unselected scores while decreasing the selected score. This is allocation under fixed predictions; the next forward supplies a new probability distribution.

Pack uncertainty. Write $M_B = \sum_{(v,i) \in B} \mathbf{1}[Y_i \neq v]$. Conditional linearity gives $\mathbb{E}[M_B \mid x_t] = r(B)$, and the union bound gives Equation 10 without an independence assumption. After the first accepted pair, every addition is checked against η. A pack with further additions therefore has

Table 6: Serial decision-order controls on 1,700 PARALLELBENCH examples in BF16. Local acceptance, revision, packing, selected-token discount, residual coverage, and uncertainty budgets are removed; all other settings are shared.

Proposal rule	Score (%)	NFE/100[†]
Position-first (entropy)	93.00	218.95
Token-first	93.12	221.07
Paired outcome and trajectory diagnostics		Examples
Both correct / both incorrect		1,576 / 112
Token-first wins / position-first (entropy) wins		7 / 5
Identical / different final texts		1,445 / 255
Identical / different commitment sequences		0 / 1,700

[†]NFE/100 is $100\times$ total forwards / total generated tokens, as in Table 1. Both rules use 76,800 forwards to fill all canvas positions, including those beyond EOS. Their returned-token totals (35,076 position-first (entropy); 34,740 token-first) account for the different normalized costs.

$r(B) \le \eta$; if the first pair already exceeds η, it remains a singleton. These cases give $r(B) \le \max\{\eta, 1 - p_{i_1}(v_1)\}$.

C PACKING–ROLLBACK ACCOUNTING

For a canvas completed in T forwards, let K_t count accepted commitments and B_t count reopened commitments at forward t. Starting from zero committed positions and ending with canvas length L gives $\sum_t (K_t - B_t) = L$. With $\bar{K} = T^{-1} \sum_t K_t$ and $r = (\sum_t B_t)/(\sum_t K_t)$, this yields

$$T = \frac{L}{\bar{K}(1 - r)}. \tag{15}$$

The identity separates pack size from revision overhead: computation decreases when each forward makes more net commitments. The identity applies to completed canvases, including executed positions beyond a returned EOS. NFE/100 instead normalizes by generated output tokens; latency additionally includes model execution and selection overhead.

D DECISION-ORDER DIAGNOSTICS

Simplified serial controls. Position-first (entropy) selects the uncommitted position with minimum Shannon entropy of the raw model distribution, then its top-probability token. Token-first selects the token with greatest aggregate probability across uncommitted positions, then its strongest position. Both commit irreversibly once per forward. Their commitment sequences differ on every example, while token-first has seven paired wins and five losses (Table 6).

Controlled parallel comparison. The controller comparison in Table 3 keeps selected-token KL discounting, raw local-top-one checks, commit/rollback ratio margins, the pack-level risk budget, and revision. For position-first (entropy), each position's KL-adjusted scores are normalized to a distribution; the lowest-entropy position and its adjusted top token form the proposal. Entropy is recomputed after each accepted commitment. This ranking replaces both aggregate and residual token support; residual state is maintained but does not affect position ranking. All four runs share BF16, prompts, seeds, canvases, self-conditioning, and uniform refresh. All examples converge. With parallel control, token-first has 12 paired wins and five losses, using 14,564 forwards versus 31,901 for position-first (entropy).

Full-decoder support statistics. The instrumented COLA run reproduces every Table 1 output, output-token count, and NFE, with 77,517 accepted commitments and 717 reopenings across 14,564 forwards. Support statistics include commitments later reopened. At each acceptance, raw support is $R(v) = \sum_{i \in U} p_i(v) / \max_{i \in U} p_i(v)$ over the current eligible positions. We average first over commitments within each example, then over the 1,700 examples. Mean support is 8.33 and

Table 7: Main evaluation datasets. All methods use the same frozen sample within each benchmark. Here, n denotes the number of evaluated test items.

Benchmark	Capability	Primary metric	Test n
PARALLELBENCH	Parallel dependency	Mean task score	1700
GSM8K	Math reasoning	Exact answer	1000
MATH-500	Math reasoning	Exact answer	500
HumanEval	Code generation	pass@1	114
MBPP	Code generation	pass@1	450
IFEval	Instruction following	Instruction strict	491
XSum	Summarization	BERTScore F1	1000

mean normalized support $R(v)/|U|$ is 0.382. The selected token matches the token of the globally highest-probability eligible pair in 49.31% of commitments under this averaging; agreement compares identities, not positions or complete pairs. Instrumented timing is excluded from the latency comparison. These traces connect aggregate-evidence proposals to the full decoder's actual commitments.

E BENCHMARK INVENTORY

Table 7 records the final evaluation inventory. PARALLELBENCH contains all 17 tasks with 100 examples each. The downstream suite totals 3,555 examples per decoder after excluding 50 development items from each of HumanEval, MBPP, and IFEval; Appendix I specifies the partitions. The main downstream metrics are exact-answer accuracy for mathematics, executable pass@1 for code, instruction-level strict accuracy for IFEval, and BERTScore F1 for XSum. On PARALLELBENCH, waiting-line and puzzle tasks use accuracy, while text tasks report the percentage of LanguageTool-clean outputs. XSum articles are sampled from nonempty test documents of at most 6,000 characters, using seed 20260810 and excluding ten development items. Every XSum prompt requests a single concise summary sentence.

Instruction and summary metrics. IFEval instruction-level strict accuracy micro-averages the original strict decisions over 768 checks in 491 prompts. Prompt-level accuracy instead requires every check in a prompt to pass and is reported separately as a diagnostic. XSum uses `bert-score` 0.3.13 with RoBERTa-large, layer 17, FP32, the slow case-sensitive tokenizer, no IDF weighting, and no baseline rescaling. We strip only outer whitespace from the saved summaries and references, score each pair, and report mean F1 multiplied by 100. BERTScore is a similarity score, not the percentage of correct summaries. None of the 6,000 pairs in Table 2 is truncated by the scorer. Table 5 uses the same scoring settings on its ablation outputs.

Development selection used prompt-level strict accuracy for IFEval and ROUGE-L F1 for XSum. The instruction-level and BERTScore results in Tables 2 and 5 were computed from unchanged outputs, without further decoder tuning. The supplementary Nemotron evaluation in Appendix L retains prompt-level strict accuracy and ROUGE-L F1, explicitly identified in its table.

F BASELINE ADAPTERS

Table 8 summarizes the selection, coordination, and revision mechanisms. All adapters share their backbone's forward path and corruption interface. DiffusionGemma's EB baseline is its official model-specific adaptation of Entropy Bound; Nemotron's released baseline uses a confidence threshold. Confidence-NFE (Conf.) sorts positions by entropy and commits a prefix within its entropy-sum budget, with at least one commitment for progress. Prophet applies its phase-dependent top-two logit-gap test to the trailing five positions of the active block, since the general prompts do not supply an answer span. Mean-Field (MF) forms pairwise distribution interactions and runs two fixed-point iterations. RDD adds its reversible update with a rollback budget of one. These settings define the transferred baselines reported in the main tables.

Table 8: Decoder mechanisms. DG and Nem. denote the two backbone adapters. Forward execution and caching are held fixed within a backbone; all listed methods are training-free.

Decoder	Proposal order	Parallel coord.	Revision
EB (DG)	Position first	Entropy bound	Denoising
Threshold (Nem.)	Position first	Confidence	No
Conf.	Position first	Entropy-sum budget	No
Prophet	Position first	All-in stop	No
MF	Position first	Pairwise	No
RDD	Position first	Backbone rule	Yes
COLA (DG)	Token first	Residual + token KL	Yes
COLA (Nem.)	Token first	Token KL + auxiliary	No

Table 9: Ratio-margin sensitivity on PARALLELBENCH with the full BF16 configuration of Table 1. Only the rollback and commitment margins change; the default 8/1 result is reused. PB is accuracy (%).

Rollback ratio	Commit ratio	PB ↑	Mean NFE ↓	NFE/100 ↓
7.0	0.5	93.41	**8.50**	**41.65**
8.0	0.75	93.41	8.54	41.84
8.0	1.0	**93.47**	8.57	41.96
9.0	1.0	**93.47**	8.57	42.00
10.0	1.5	**93.47**	8.65	42.36

G REVISION-MARGIN SENSITIVITY

Across these neighboring operating points, accuracy varies by one example out of 1,700 and NFE/100 ranges from 41.65 to 42.36. Relative to the default 8/1 setting, 7/0.5 and 8/0.75 each lose one correct example, while 9/1 and 10/1.5 preserve every per-example correctness outcome. All runs converge within the configured forward cap. The settings jointly vary commitment and revision margins, with self-conditioning, suppression, residual coverage, and the uncertainty budget held fixed.

H PAIRED DOWNSTREAM COMPARISONS

GSM8K and MBPP intervals exclude zero in favor of COLA; the MATH-500 and HumanEval intervals include zero. The supplementary IFEval prompt-level interval favors EB. Its primary instruction-level comparison is 674/768 versus 689/768 checks (Appendix K).

For XSum, the paired BERTScore F1 difference is $+0.0463$ points against EB, with a 95% interval of $[-0.0009, 0.0954]$, and $+0.1130$ against Conf., with $[0.0413, 0.1864]$. These intervals use 20,000 paired resamples of the 1,000 summary pairs with seed 20260925 and no multiplicity correction. Full-precision per-example scores are used before rounding the displayed means.

I REPRODUCIBILITY CONFIGURATION

Execution and evaluation. Use the evaluation pools and metrics in Appendix E, with identical prompts, per-example seeds, and output budgets across methods. Generation uses batch size one, non-thinking prompts, and deterministic token selection conditional on the corrupted canvas. For Tables 1 and 2, DiffusionGemma 26B-A4B IT runs in bfloat16 on three RTX 3090 GPUs without offload; the supplementary Nemotron-Labs-Diffusion-8B runs in bfloat16 on one RTX 3090. All runs use Docker. NFE/100 is 100 times total model evaluations divided by total generated tokens; budget-truncated outputs remain in scoring. For latency comparisons, alternate EB and COLA per example on the same loaded DiffusionGemma model and GPUs. ParallelBench text scoring uses LanguageTool 6.6 for DiffusionGemma and 6.8 for Nemotron, retaining the original benchmark prompts and references.

Table 10: Paired DiffusionGemma comparisons on the Table 2 pools. Math and code rows use the primary metrics; the IFEval row is a supplementary prompt-level strict diagnostic. Counts are successful examples or prompts. Differences and 95% paired-bootstrap intervals are percentage points (20,000 resamples, seed 20260923), reflecting example variation at fixed generation seeds.

Task	n	EB	COLA	Δ	95% interval	Wins / losses
GSM8K	1000	949	965	+1.6	$[0.6, 2.6]$	21 / 5
MATH-500	500	213	226	+2.6	$[-0.2, 5.4]$	32 / 19
HumanEval	114	81	93	+10.5	$[0.0, 21.1]$	26 / 14
MBPP	450	257	350	+20.7	$[16.0, 25.3]$	111 / 18
IFEval (prompt)	491	419	403	-3.3	$[-5.7, -0.8]$	12 / 28

DiffusionGemma on PARALLELBENCH. Use seeds $730001 +$ ordinal and canvas lengths/output limits of 32 for waiting-line tasks and 64 for text and puzzles. Set $(\tau_{\mathrm{cm}}, \tau_{\mathrm{rb}}) = (1, 8)$, $\eta = 0.1$, $\beta = 1$, $\lambda = 0.5$, $\gamma = 1$, and $\epsilon = 10^{-6}$. The first locally admissible pair is permitted regardless of the remaining uncertainty budget. Retain uniform refresh and pass previous processed logits through self-conditioning in decoder embedding dtype, resetting them at block boundaries. COLA uses no native companion or fallback. EB uses entropy bound 0.1; Conf. uses entropy bound 0.005. Prophet uses phase-wise top-two logit gaps 7.5/5.0/2.5, MF uses threshold 0.9 with two interaction iterations, and RDD uses scaling 0.9, remask exponent 1.0, and rollback budget one.

Nemotron configuration. Use canvas 32, seeds $730001 +$ ordinal on PARALLELBENCH, and the same task-specific output limits. Set commitment ratio 7, minimum token probability 0.95, minimum position ratio 0.5, $\beta = 0$, $\lambda = \gamma = 1$, and $\epsilon = 10^{-6}$. The position ratio is $(s_1 - s_2)/s_2$ for the two highest adjusted placement scores of the selected token, or infinity when one position remains. Local top-one, probability, and token-margin gates use raw model probabilities. After each token-first pack, accept native top-one companions above 0.95 without a token cap. If neither path commits, accept up to four highest-confidence native predictions above 0.90, or the single most confident prediction if none qualifies. Preserve native causal-boundary and cache handling; accepted tokens remain fixed, without an additional uncertainty budget or rollback.

Downstream configuration Use the same evaluation identities for every method: 1,000 GSM8K, 500 MATH, 114 HumanEval, 450 MBPP, 491 IFEval, and 1,000 XSum examples. Output limits are 512 tokens for mathematics and code, 1,024 for IFEval, and 128 for XSum. For DiffusionGemma, the seed is 730001 XOR the low 31 bits of the SHA-256 integer of the item ID. For Nemotron, it is 730001 XOR the big-endian integer of the first eight SHA-256 bytes, reduced modulo $2^{31} - 1$.

Ablation settings. Keep evaluation identities, prompts, seeds, output budgets, and starting hyperparameters fixed. Accumulate removals in table order: disable η, set $\tau_{\mathrm{cm}} = \tau_{\mathrm{rb}} = 0$, set $\beta = 0$, set $\lambda = 0$, then limit each pack to one token. Local top-one acceptance, non-top-one reopening, self-conditioning, and uniform refresh remain active. Zero margins therefore remove margin-based gates rather than all revision. Retain capped or truncated outputs in the scored denominator and include their computation in NFE/100.

J PROPOSAL ACCEPTANCE AND WITHIN-FORWARD SCORES

The content-before-position framework specifies proposal priority, whereas acceptance tests whether the selected location can support that proposal. The two controls have distinct meanings: ρ weights token identity against auxiliary placement in the one-step loss, while τ_{cm} specifies local separation from competing tokens. COLA builds on the aggregate-evidence, token-first endpoint, coordinates successive proposals through residual support and discounting, and uses τ_{cm} for local acceptance.

Consider the first proposal from unmodified probabilities with $\beta = 0$. Let $v^* = \arg\max_v A(v)$ and $i^* = \arg\max_i p_i(v^*)$. If v^* is locally top one and its ratio margin clears τ_{cm}, acceptance preserves the token-first endpoint action exactly. A sufficient condition is

$$P(v^*) > \frac{1 + \tau_{\mathrm{cm}}}{2 + \tau_{\mathrm{cm}}}. \tag{16}$$

Table 11: Current DiffusionGemma IFEval strict checks on the 491 prompts in Table 2. Counts are satisfied checks, not complete prompts.

Instruction family	Checks	EB	COLA
Format	146	140	137
Length	133	112	111
Combination	59	42	40
Keywords	152	132	130
Start/end	56	53	51
Detectable content	49	48	46
Language	28	28	28
Letter case	84	74	74
Punctuation	61	60	57
Total	768	689	674

Every competing probability is at most $1 - P(v^*)$, so this condition implies both local top one and $(P(v^*) - p_2(i^*))/p_2(i^*) > \tau_{\mathrm{cm}}$. Nemotron's probability floor 0.95 already exceeds the sufficient bound $8/9$ for commit ratio 7; its position-separation test additionally checks whether placement is distinctive. After a packing update, the same local acceptance criterion uses raw probabilities even though proposal priority uses adjusted scores.

Directional KL makes the accepted position the reference for subsequent allocation. For a remaining position j, repeated updates have the form

$$\widetilde{p}_j(v) = p_j(v) \prod_{k:\, v_k = v} w_{kj}, \qquad w_{kj} = \max\{\epsilon, 1 - \lambda e^{-\gamma D_{\mathrm{KL}}(p_{i_k} \| p_j)}\}. \tag{17}$$

Here (v_k, i_k) ranges over earlier commitments in the current pack, with $\lambda \geq 0$ and $0 < \epsilon \leq 1$. Each factor lies in $[\epsilon, 1]$. Thus scores can decrease but cannot exceed their original probabilities, and tokens not yet selected retain their original entries. Leaving rows unnormalized preserves this allocation discount; normalization would also rescale unrelated token entries. A new forward resets the product. The implementation computes one selected-to-remaining KL vector at a time, requiring $O(Kn|\mathcal{V}|)$ arithmetic for a pack of K commitments and $O(n|\mathcal{V}|)$ workspace. The workspace holds the forward distributions and adjusted scores plus a length-n KL vector, so memory grows linearly with canvas length and vocabulary size. For K proportional to n, arithmetic grows quadratically in canvas length; incremental evaluation keeps the workspace linear in n.

K Output Diagnostics

Instruction following. The 491-prompt DiffusionGemma evaluation contains 768 individual checks. EB satisfies 689 and COLA 674. At prompt level, 391 examples pass with both decoders, 28 only with EB, 12 only with COLA, and 60 with neither. Table 11 reports the current frozen strict-check decisions. These counts distinguish individual instruction checks from whole-prompt success.

Summary repetition. On the current 1,000 XSum examples, mean per-example adjacent-word repetition rates are 0.0239% for EB and 0.0156% for COLA; repeated-fourgram rates are 0.0042% for both. We lowercase Unicode alphabetic word sequences, divide adjacent-repeat events by available word transitions and duplicate fourgram occurrences by available fourgrams, and average rates over examples (empty denominators contribute zero). These lexical diagnostics complement BERTScore by measuring surface repetition rather than semantic similarity.

Code-error classification. The analysis in Section 7 uses all 564 paired code outputs and the original scorer's extraction function, saved test outcomes, and error logs. Static compilation checks classify syntax and indentation failures; saved stderr identifies the first observed test termination. Outputs are neither repaired nor re-executed. Syntax/indentation failures account for 102/111 MBPP wins and 22/26 HumanEval wins, but also 10/18 and 11/14 regressions, respectively. Residual Markdown fences are overlapping flags, not additional failure counts. The both-compilable comparison

Table 12: Supplementary Nemotron quality and computational cost on all 1,700 PARALLELBENCH examples. Threshold is the released confidence-threshold decoder. Conf., Prophet, MF, and RDD denote Confidence-NFE, Prophet, Mean-Field, and Reversible Diffusion Decoding. Waiting-line and puzzle rows report accuracy; text rows report LanguageTool-clean outputs (%). Overall score averages the 17 task scores; NFE/100 counts forwards per 100 generated tokens. Bold marks the best value, including ties.

Task	Threshold	Conf.	Prophet	MF	RDD	**COLA**
Waiting-line						
Copy	**100**	**100**	**100**	**100**	**100**	**100**
Reverse	**100**	**100**	**100**	**100**	**100**	**100**
Sort	22	21	1	**25**	22	22
Shuffle	88	85	1	82	**89**	**89**
Insert by index	89	86	73	90	84	**91**
Insert random	83	88	24	89	88	**91**
Remove by index	81	**88**	80	84	86	82
Remove random	91	**98**	41	87	92	97
Replace by index	92	95	84	92	**97**	96
Replace random	96	96	25	96	96	**97**
Text generation						
Paraphrasing	98	98	67	**100**	98	98
Summarization	87	90	83	**91**	87	87
Words to sentence (easy)	**100**	**100**	99	99	**100**	**100**
Words to sentence (medium)	**100**	**100**	99	99	**100**	**100**
Words to sentence (hard)	**100**	**100**	99	94	**100**	**100**
Puzzles						
Latin square	88	**94**	88	86	91	**94**
Sudoku	40	33	5	**42**	40	41
Overall score ↑	85.6	86.6	62.9	85.6	86.5	**87.4**
Overall NFE/100 tokens ↓	79.3	152.2	106.7	**65.3**	99.7	88.1

conditions on both generated outputs and retains the original test outcomes. Missing EB truncation flags are left unknown rather than interpreted as zero.

L SUPPLEMENTARY NEMOTRON EVALUATION

Nemotron evaluates the token-first proposal rule in a masked-block interface. Its COLA adaptation combines vocabulary-wide aggregate evidence and selected-token suppression with permanent commitments, high-confidence companions, and bounded fallback. Tables 12 and 13 report this fixed-commitment configuration; the following diagnostics explain its execution paths and computational cost. On Nemotron, COLA scores 1,485/1,700 (87.35%) against the released confidence-threshold decoder's 1,455/1,700 (85.59%), a gain of 1.76 points. The improvement is concentrated in structured tasks: waiting-line correctness rises from 842 to 865 out of 1,000, and puzzle correctness from 128 to 135 out of 200. All five text-task totals tie.

These gains use fixed commitments, with no masked-review forwards or reopened positions. COLA requires 18.57 forwards per example, or 88.1 NFE/100 versus the released decoder's 79.3.

On Nemotron, COLA matches Threshold on GSM8K and MATH-500, and achieves the best MBPP score among the evaluated decoders, 286/450 (63.6%). It solves one additional HumanEval problem and two additional IFEval prompts relative to Threshold, reaching 78.9% and 59.1%, respectively; XSum remains close at 17.50 versus 17.49 ROUGE-L. Aggregate NFE/100 is 60.4 compared with Threshold's 56.1, using the same fixed-commitment configuration across all six tasks.

L.1 COMMITMENT PATHS

Table 14 separates token-first packing, high-confidence companions, and fallback over the 3,555 downstream examples. Token-first packing initiates each step; fallback supplies progress when

Table 13: Nemotron downstream results on the Table 2 sample identities. IFEval uses prompt-level strict accuracy and XSum ROUGE-L F1; math and code metrics are unchanged. Scores are scaled by 100; NFE/100 is total forwards per 100 generated tokens. Bold marks column-best displayed values. Settings are in Appendix I.

Backbone	Decoder	GSM8K Acc.	MATH Acc.	HumanEval pass@1	MBPP pass@1	IFEval Prompt strict	XSum ROUGE-L	NFE/100 ↓
Nemotron	Threshold	92.6	**40.0**	78.1	63.3	58.7	17.49	56.1
	Conf.	**92.7**	39.2	**79.8**	60.7	**60.3**	**17.51**	74.1
	Prophet	90.9	36.8	62.3	55.3	56.8	17.50	83.8
	MF	91.5	39.6	69.3	54.7	55.2	**17.51**	**48.1**
	RDD	92.6	39.8	78.9	62.9	59.7	**17.51**	67.4
	COLA	92.6	**40.0**	78.9	**63.6**	59.1	17.50	60.4

Table 14: Nemotron commitment shares (%) on the current downstream evaluation pool. Counts cover executed canvas positions, excluding boundary seeds.

Task	Token-first	Companion	Fallback
GSM8K	16.4	52.7	30.8
MATH-500	17.1	52.5	30.4
HumanEval	16.9	54.0	29.0
MBPP	17.0	42.4	40.6
IFEval	12.4	40.8	46.8
XSum	10.9	33.2	55.9

neither earlier path commits. Counts include all executed block positions, including those beyond EOS, but exclude autoregressive boundary seeds. The substantial companion and fallback shares characterize this masked-block adaptation; no forced completion occurs.

L.2 Selection overhead

We profile both decoders on ten GSM8K and ten MBPP examples sampled with seed 20260905, using one RTX 3090, batch size one, 32-token blocks, and a 512-token limit. All 40 texts reproduce the saved outputs. CUDA events time model forwards (including causal prefill), packing, and nested KL updates, with synchronization at generation end. Table 15 subtracts KL from packing to avoid double counting; the remainder covers other control and synchronization. Forwards account for about 94% of COLA time, versus 0.28–0.33% for KL. Its higher latency accompanies more forwards, rather than substantial KL overhead.

L.3 Stability-weighted comparison

SWD discounts confidence using previous-to-current distribution KL (Wu & Huang, 2026). Following its Algorithm 1 and Figure 6, we multiply confidence by the exponential negative KL weight with $\lambda_{\mathrm{SWD}} = 1$ and uniform initial history, reset per block. The adapter retains threshold 0.9, at-least-one-position progress, causal boundary handling, and caching.

Table 16 uses 100 GSM8K and 100 MBPP examples sampled with seed 20260905. All methods share prompts, 32-token blocks, deterministic generation, and a 512-token limit; Threshold and COLA outputs are retrieved by exact IDs. SWD leads on arithmetic, while COLA leads on code. The task-dependent rankings distinguish temporal stability from collective content evidence.

L.4 Selected-token KL sensitivity

Table 17 varies one KL parameter at a time on ten examples per ParallelBench task (seed 20260905), holding other settings fixed. All six configurations, including $\lambda = 0$, produce identical text and NFE, with 150/170 correct. Token-first allocations vary slightly, but their sum with companion commitments remains 4,440; fallback contributes 1,991 throughout. High-confidence companions

Table 15: Instrumented Nemotron timing on ten examples per task. Time is mean seconds/example; phase percentages use total generation wall time.

Task	Decoder	Time	Fwd (%)	KL (%)	Pack† (%)
GSM8K	Threshold	3.77	95.8	–	–
	COLA	4.40	94.0	0.33	3.45
MBPP	Threshold	7.40	95.8	–	–
	COLA	8.35	94.1	0.28	3.41

†Packing excluding KL. Mean NFE is 101.5/117.0 for Threshold/COLA on GSM8K and 200.1/222.3 on MBPP.

Table 16: Matched-subset Nemotron comparison: 100 examples per task. GSM8K uses exact-answer accuracy and MBPP executable pass@1 (%); these are separate from the supplementary full-suite results in Table 13.

	GSM8K		MBPP	
Decoder	Acc.	NFE/100	Acc.	NFE/100
Threshold	95	47.28	66	61.02
SWD	97	54.24	54	63.15
COLA	95	52.84	67	65.97

can therefore absorb changes to the token-first pack under this acceptance regime. Together with Table 5, this illustrates how allocation effects depend on the surrounding commitment controller.

Table 17: Nemotron KL sensitivity on a stratified 170-example slice. The last row disables the penalty. Commit counts cover token-first allocations over all executed blocks; all six rows have identical text and NFE.

λ	γ	PB (%)	NFE/100	Token-first commits
1	1	88.24	86.10	565
0.5	1	88.24	86.10	563
2	1	88.24	86.10	566
1	0.5	88.24	86.10	566
1	2	88.24	86.10	563
0	1	88.24	86.10	563