INTER-TRAJECTORY DEPENDENCE LIMITS ANSWER COVERAGE IN LOCAL DLLM REPAIR

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time compute scaling improves large language model performance by allocating more computation during inference. For diffusion large language models (dLLMs), sequential repair offers a natural instantiation: it iteratively remasks low-confidence tokens and regenerates them while retaining the surrounding completion. However, additional repair rounds often yield limited performance gains. While recent work has studied inference dynamics within individual trajectories, our focus is inter-trajectory dependence: whether successive repair candidates explore distinct answer states. We introduce a diagnostic framework that combines answer-level metrics, hyperparameter sweeps, and remasking interventions to characterize this dependence. We assess pairwise answer agreement, unique-answer counts, oracle pass@k, and correction/regression rates across four dLLMs and three verifiable tasks. We find that sequential local repair produces highly correlated candidates with substantially lower answer coverage than independent sampling. The evaluated local-remasking variants do not reliably close this gap. These findings suggest prioritizing trajectory-level diversification when allocating test-time compute for answer coverage, with local repair used for refinement. Our study adds an inter-trajectory perspective to the analysis of dLLM inference dynamics and provides practical guidance for allocating inference compute.

1 INTRODUCTION

Test-time compute scaling is a central approach for improving large language model performance (Wang et al., 2023; Cobbe et al., 2021; Lightman et al., 2024; Snell et al., 2024; Muennighoff et al., 2025). In diffusion large language models (dLLMs) (Austin et al., 2021a; Nie et al., 2025), extra inference can be spent on trajectory-level resampling or sequential repair. Resampling starts each candidate from a fresh masked sequence; repair remasks and regenerates part of an existing completion. Prior work has developed several remasking-based refinement strategies (Wang et al., 2025; Zhai et al., 2026; Yao, 2026; Hong et al., 2025), while answer-level coverage across successive local repairs remains less characterized. We study this coverage by comparing candidates from retained-state repair with independent restart trajectories under the same model, using budget and generation-schedule controls to interpret the observed differences.

Recent studies describe commitment and uncertainty within individual dLLM trajectories, including proximity effects, anchor tokens, confusion zones, and irreversible block decisions (Kim et al., 2026; Guo & Tan, 2026; Chen et al., 2025a; Wang et al., 2026). These phenomena motivate a complementary question at the candidate-set level: when a completion is repeatedly repaired, how much does the resulting chain expand the set of answer hypotheses? We refer to this quantity as inter-trajectory answer coverage across the candidate set.

Figure 1 contrasts the two regimes. Repeated repair reuses a partially denoised sequence, so regenerated tokens condition on the context that survives remasking. Confidence-based remasking preserves many high-confidence answer-bearing tokens, and bidirectional attention propagates this context to newly generated tokens. The resulting chain can vary in wording while remaining concentrated around a small set of answer states.

We characterize this coverage with four complementary quantities: pairwise answer agreement $\hat{\rho}$, unique-answer count, oracle pass@k, and wrong-to-correct/correct-to-wrong transitions. Across four models and three tasks, independent restart trajectories expand answer coverage, whereas sequential

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

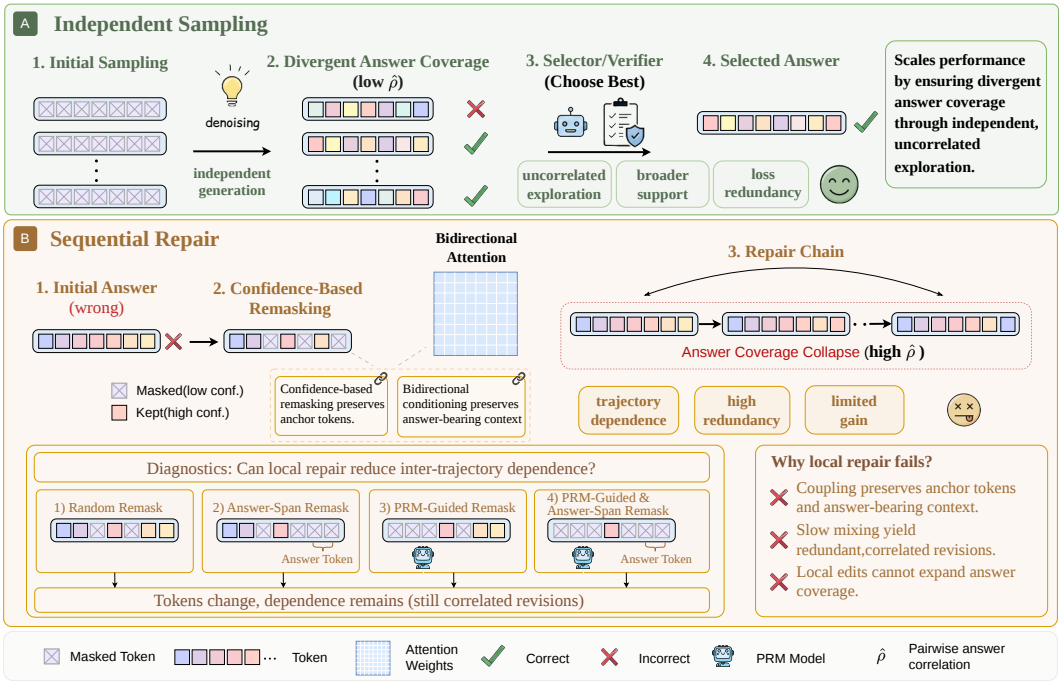


Figure 1: Inter-trajectory answer coverage collapse under dLLM repair. Independent sampling allocates test-time compute to separate denoising trajectories and expands the set of candidate answers. Sequential repair conditions each new candidate on the retained completion; in the evaluated settings, its candidates exhibit high answer agreement and limited answer-level coverage.

repair remains concentrated around a few answer states. Semantic canonicalization preserves the coverage gap. Parameter sweeps and random, answer-span, and process-guided remasking variants show that local changes do not reliably produce broader coverage. An attention-blocking intervention lowers agreement, supporting retained-context conditioning as one contributor. Finally, a full-remask restart that uses the independent sampler's blockwise kernel recovers coverage close to independent sampling, showing that the generation schedule accounts for the full-remask deficit.

The main contributions of this work are summarized as follows:

1. We characterize high answer agreement and limited candidate-set coverage under retained-state dLLM repair across models and tasks.
2. We introduce a diagnostic methodology that combines answer-level metrics, budget comparisons, parameter sweeps, semantic canonicalization, and controlled remasking interventions.
3. We identify a practical compute-allocation principle: generate independent answer hypotheses before extending a single repair chain when coverage is the objective.

2 Related Work

AR test-time scaling and self-correction. Inference-time scaling in autoregressive models typically improves reasoning by drawing near-independent samples, applying self-consistency, or selecting with verifiers (Wang et al., 2023; Cobbe et al., 2021; Lightman et al., 2024; Snell et al., 2024; Yao et al., 2023; Muennighoff et al., 2025), with pass@k capturing the resulting oracle upper bound (Chen et al., 2021). Iterative self-correction in AR models (Madaan et al., 2023; Shinn et al., 2023; Chen et al., 2024) feeds the previous answer back as text the model may overwrite, and its reported obstacle is self-evaluation (Huang et al., 2024). dLLM repair instead operates on a retained decoding state: confidence-based remasking leaves high-confidence answer tokens unmasked and bidirectional attention conditions regeneration on what was kept, so the candidate set is constrained before any selection is applied. Our diagnostics (Sections 5.1, 5.2) evaluate PRM-guided local edits and oracle selection over repair chains, finding substantially broader coverage under independent restarts.

dLLM test-time compute methods. Recent dLLM inference methods spend compute through remasking-based refinement (Wang et al., 2025; Zhai et al., 2026; Yao, 2026; Hong et al., 2025; Huang et al., 2025), diversity or reward-guided generation (Lamont et al., 2026; Lee et al., 2025; Chen et al., 2025b; Miles et al., 2026), search-based exploration (Bilal et al., 2026; Bai et al., 2026; Zhang et al., 2025; Huang et al., 2026), or adaptive unmasking and early stopping (Ben-Hamu et al., 2025; Li et al., 2025; Jazbec et al., 2026). These methods improve candidate quality, diversify trajectories, or reduce unnecessary steps. Our work complements these decoders by diagnosing when repeated local repair fails to convert extra compute into answer coverage, with metrics that track whether a strategy diversifies answer hypotheses.

Inference dynamics in dLLMs. A growing line of work analyzes commitment phenomena within individual dLLM trajectories. Early unmasking choices can shape the entire trajectory through proximity bias (Kim et al., 2026); attention analyses show dynamic sinks or floating anchors across denoising steps (Rulli et al., 2025; Dai et al., 2026); step-level uncertainty concentrates in dynamic confusion zones (Chen et al., 2025a); and irreversible commitments can cause stagnation that motivates backtracking (Wang et al., 2026). Hallucination and factuality studies further connect early commitments to context intrusion, commitment uncertainty, and self-consistent but incorrect outputs (Guo & Tan, 2026; Shah et al., 2026).

Positioning. Prior work characterizes intra-trajectory dynamics in dLLMs or uses trajectory-level signals to improve decoding (Kim et al., 2026; Guo & Tan, 2026; Chen et al., 2025a; Wang et al., 2026; Shah et al., 2026). We study a complementary scale: how answer states remain dependent across candidates produced by repeated repair. At this inter-trajectory scale, the limitation appears as answer coverage collapse; we quantify it with unique-answer count and pairwise agreement, and validate trajectory-level resampling as a lightweight intervention.

3 PRELIMINARIES

Masked diffusion language models. A masked diffusion language model (Austin et al., 2021a; Nie et al., 2025) corrupts a token sequence by replacing positions with `[MASK]` tokens and trains p_θ to predict masked tokens using bidirectional attention over the partially observed sequence. At inference, generation runs for T denoising steps: the model scores each masked position by $p_\theta(x_i \mid \mathbf{x}_t)$ and unmasks the highest-confidence fraction. Unlike autoregressive models, every non-masked token can attend to every other non-masked position. For confidence-based BoN selection, we use the mean predicted probability over response tokens of the fully denoised output.

Test-time inference strategies. Fix a total denoising budget B measured in forward passes. We compare three strategies parameterised by a shared temperature t and a total candidate budget.

Greedy. Greedy runs a single deterministic trajectory ($t{=}0$), costing T forward passes.

Independent sampling / Best-of-N (BoN). BoN draws N independent trajectories at temperature t, followed by task-specific selection. Total cost: NT forward passes.

Repair. We use repair to denote the natural sequential-remasking baseline: start from a single candidate $\hat{\mathbf{x}}^{(0)}$ drawn at temperature t and apply R revision rounds. Let $\mathbf{c}_i^{(r)} = p_\theta\big(\hat{x}_i^{(r)} \mid \hat{\mathbf{x}}^{(r)}\big)$ denote the per-token confidence after round r, and let $\mathcal{M}_\alpha(\hat{\mathbf{x}}, \mathbf{c})$ remask the α-fraction of positions with lowest confidence. Each round executes

$$\tilde{\mathbf{x}}^{(r)} = \mathcal{M}_\alpha\big(\hat{\mathbf{x}}^{(r-1)}, \mathbf{c}^{(r-1)}\big), \qquad \hat{\mathbf{x}}^{(r)} \sim p_\theta\big(\mathbf{x} \mid \tilde{\mathbf{x}}^{(r)}; T'\big), \tag{1}$$

with T' denoising steps per round, capped at $T/2$ by default. A round cannot take more denoising steps than it has masked positions, so $T' = \min\big(T/2, \lfloor \alpha L \rfloor\big)$ for response length L. Recomputing $\mathbf{c}^{(r)}$ costs one additional forward pass, giving per-round cost $C_{\text{rep}}{=}T'{+}1$. At the default $\alpha{=}0.2$, the mask count binds ($\lfloor \alpha L \rfloor{=}25$ for $L{=}128$ and 51 for $L{=}256$), giving $C_{\text{rep}}{\approx}0.81\,(T/2)$. Total cost: $T{+}RC_{\text{rep}}$ forward passes. All budget multipliers reported here are measured, not nominal.

Resampling-then-repair. As a lightweight intervention, we draw multiple independent trajectories and then apply limited local repair to each, decoupling answer-space exploration from refinement.

ODD (Orthogonal Diverse Diffusion). ODD (Lamont et al., 2026) generates N candidates jointly and adds a diversity-repulsion term α_{ODD} against tokens favored by other candidates. Its cost is NT forward passes, matching BoN, and it serves as a trajectory-level diversity baseline in Section 5.1.

Evaluation quantities. For k candidates $\{\hat{\mathbf{x}}^{(j)}\}_{j=1}^{k}$, we use four complementary quantities.

Coverage. pass@$k = \Pr[\exists\, j \le k : \hat{\mathbf{x}}^{(j)} \text{ is correct}]$ is the oracle accuracy achievable with a perfect selector; it upper-bounds the accuracy of any selector over the same candidate pool.

Dependence. Pairwise answer agreement

$$\hat{\rho} \;=\; \binom{k}{2}^{-1} \sum_{1 \le i < j \le k} \mathbf{1}\Big[\text{ans}(\hat{\mathbf{x}}^{(i)}) = \text{ans}(\hat{\mathbf{x}}^{(j)})\Big], \tag{2}$$

where $\text{ans}(\cdot)$ extracts the final answer. This statistic describes answer concentration and serves as a dependence proxy; independent draws from a concentrated distribution can also agree. When $\hat{\rho}{=}1$, all candidates share the same extracted answer. Lower agreement indicates more distinct answers, with potential for pass@k gains at comparable candidate quality.

Answer diversity. Unique answer count

$$u_k = \mathbb{E}\left[\left|\left\{\text{ans}(\hat{\mathbf{x}}^{(j)})\right\}_{j=1}^{k}\right|\right] \tag{3}$$

counts distinct extracted answers, with parse failures pooled in one bucket. Low u_k indicates limited answer diversity even when outputs differ in surface form.

Correction and regression. For consecutive rounds $r{-}1, r$ of a repair chain, let $y_i^{(r)} \in \{0,1\}$ mark whether problem i is answered correctly at round r. The wrong-to-correct and correct-to-wrong rates are

$$\text{W}{\to}\text{C}_r = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \mathbf{1}[y_i^{(r-1)}{=}0 \wedge y_i^{(r)}{=}1], \quad \text{C}{\to}\text{W}_r = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \mathbf{1}[y_i^{(r-1)}{=}1 \wedge y_i^{(r)}{=}0], \tag{4}$$

with per-round drift $\text{Net}_r = \text{W}{\to}\text{C}_r - \text{C}{\to}\text{W}_r$. A chain can change answers frequently ($\text{W}{\to}\text{C}_r + \text{C}{\to}\text{W}_r > 0$) while $\text{Net}_r \approx 0$, the signature of undirected local editing rather than directional improvement in answer accuracy across repair rounds.

4 Experiments

The main experiments concentrate on three representative tasks: **Countdown** (Katz et al., 2025), which evaluates constrained symbolic reasoning; **MATH500** (Hendrycks et al., 2021), which evaluates mathematical reasoning; and **HumanEval** (Chen et al., 2021), which evaluates code generation. Appendix D.5 further extends the evaluation to additional benchmarks for code generation, arithmetic reasoning, and multiple-choice question answering, including MBPP (Austin et al., 2021b), GSM8K (Cobbe et al., 2021), ARC-Challenge (Clark et al., 2018), OpenBookQA (Mihaylov et al., 2018), and CommonsenseQA (Talmor et al., 2019). The main experiments ask whether additional compute improves answer coverage when spent on independent trajectories or sequential repair. The primary model in this study is LLaDA-8B-Instruct (Nie et al., 2025). In addition, we evaluate LLaDA-1.5 (Zhu et al., 2025), LLaDA-MoE-7B-A1B (inclusionAI, 2025), and Dream-Instruct-7B (Ye et al., 2025). Full details of the experimental setup are provided in Appendix A.

Independent trajectories convert compute into coverage. Independent sampling provides the strongest accuracy gains across the evaluated model–task pairs, as summarized in Table 1. Repair is reported at $r{=}8$, 18, and 38 rounds, with measured budgets of $4.3\times$, $8.3\times$, and $16.4\times$ greedy decoding, within 10% of the corresponding BoN-$4/8/16$ budgets (Appendix A). The repair selector can inspect $r{+}1 \in \{9, 19, 39\}$ states, compared with $N \in \{4, 8, 16\}$ BoN candidates, so this comparison gives repair a larger candidate pool. Even with that advantage, the best repair setting reaches BoN-4 in only one of twelve model–task pairs: Dream on Countdown (50.0 at $16.4\times$ versus 45.8 at $4\times$). It remains 23.7 points below BoN-16. Repair accuracy is nearly flat in r, while independent trajectories continue to broaden coverage.

Table 1: Approximately matched-budget accuracy (%) across four dLLM models and three benchmarks. Repair uses t=0.4, α=0.2. Both strategies use the task-specific selection rules of Appendix A—symbolic verification (Countdown), unit-test execution (HumanEval), answer self-consistency (MATH500)—applied to BoN's N independent candidates and to the r+1 states of the repair chain; code-task selection uses the evaluation tests. Appendix C.9 describes repair fallback and the unselected variant. Red underlined repair entries fall below greedy; teal bold marks the best accuracy result for each corresponding model–task pair.

Model	Dataset	Greedy (1×)	Repair			BoN		
			r=8 (4.3×)	r=18 (8.3×)	r=38 (16.4×)	N=4 (4×)	N=8 (8×)	N=16 (16×)
LLaDA-8B	Countdown	25.4	27.2	27.7	27.7	57.9	72.5	**83.2**
	MATH500	33.4	33.3	33.3	33.5	36.9	39.5	**40.1**
	HumanEval	26.8	33.5	33.7	34.2	36.0	42.1	**46.7**
LLaDA-1.5	Countdown	19.1	24.1	24.3	24.3	55.2	74.5	**83.5**
	MATH500	33.2	34.9	34.9	34.9	37.1	40.0	**41.1**
	HumanEval	27.4	32.3	32.5	32.5	36.8	42.3	**48.6**
LLaDA-MoE	Countdown	43.4	40.6	40.9	41.3	69.5	82.0	**87.8**
	MATH500	35.4	34.8	34.7	34.8	40.5	43.8	**45.3**
	HumanEval	42.1	46.1	46.1	46.1	51.2	57.7	**60.8**
Dream-7B	Countdown	19.5	44.4	47.0	50.0	45.8	61.7	**73.7**
	MATH500	20.0	18.2	18.3	18.3	18.9	23.3	**26.8**
	HumanEval	34.8	22.0	22.4	23.0	37.4	48.6	**55.9**

Sequential repair fails to expand answer coverage. Repair gains on LLaDA are concentrated on HumanEval. Paired bootstrap intervals for repair minus greedy exclude zero for LLaDA-8B and LLaDA-1.5 at r=38 on HumanEval, while the MoE HumanEval gain and all Countdown/MATH500 intervals include zero. The significant HumanEval gains remain expensive: LLaDA-8B reaches 34.2 at 16.4× greedy compute versus 36.0 for BoN-4 at 4×; LLaDA-1.5 reaches 32.5 versus 36.8. On Countdown, BoN-16 gains at least 40 points, while the largest LLaDA repair gain is +5.2 and is not significant. The chain selector can recover only answers visited during repair, so its ceiling is set by the chain's answer coverage (Appendix C.10). This pattern motivates the answer-level diagnostics in Section 5.2: repair can retain high-confidence answer tokens while revising their surrounding rationale, and successive candidates may revisit similar conclusions with limited expansion of the answer pool across successive repair rounds.

Dream: Task-dependent repair gains. Dream exhibits local repair gains on the Countdown task, where accuracy more than doubles over greedy by r=38; this is the only repair-minus-greedy effect among all four models that is both large and statistically significant (paired bootstrap $[+24.6, +36.3]$), and the only cell where repair overtakes BoN-4—at four times its compute. However, these gains plateau well below BoN-16 and fail to generalize: on MATH500 and HumanEval repair instead *degrades* performance, significantly so on HumanEval ($[-18.3, -5.3]$). Dream therefore provides a task-dependent case of positive local drift. Section 5.2 examines answer-token exposure alongside correction and regression rates; the benefit of changing an answer varies across the evaluated tasks.

5 Diagnosing Inter-Trajectory Dependence

Diagnostic framing. Section 4 shows that additional repair rounds produce limited accuracy and motivates a more direct analysis of candidate-set coverage. We (i) fix the candidate count and use *oracle pass@k* to separate coverage from selector quality, (ii) test whether resampling before repair preserves coverage, and (iii) examine the dependence under masking interventions and parameter sweeps. Matched-compute comparisons provide the corresponding efficiency view.

Table 2: Candidate-count comparison at fixed temperature. Independent sampling and repair average seeds 42–44; ODD is a seed-42 reference. pass@1 scores the first candidate for independent sampling and repair, and a uniformly drawn candidate for ODD. Teal and red highlight the best and worst values, respectively, within each task and metric column.

Dataset	Strategy	pass@1	pass@16	Unique/16	$\hat{\rho}$
Countdown	Independent sampling	**25.2%**	**82.8%**	**11.5**	**0.08**
	ODD	**6.5%**	47.0%	5.3	0.48
	Sequential repair	24.5%	**29.0%**	**1.3**	**0.93**
MATH500	Independent sampling	34.0%	**63.8%**	**6.8**	**0.35**
	ODD	**19.4%**	49.0%	5.7	0.38
	Sequential repair	**35.0%**	**37.0%**	**1.3**	**0.94**
HumanEval	Independent sampling	**28.7%**	**53.1%**	**8.8**	**0.30**
	ODD	**22.3%**	43.3%	7.7	0.43
	Sequential repair	27.2%	**35.6%**	**2.3**	**0.80**

Table 3: Resampling-then-repair on Countdown with LLaDA-8B (k=8). The first two rows are baselines; +1/+3 apply one or three repair rounds after resampling. W→C, C→W, and Net compare correctness before and after repair. Teal/red mark best/worst values for pass@8, Unique/8, and $\hat{\rho}$.

Setting	pass@8	Unique/8	$\hat{\rho}$	W→C	C→W	Net
Resampling only	72.3	**6.67**	**0.067**	–	–	–
Sequential repair	**23.8**	**1.27**	**0.914**	2.73	0.39	+2.34
+ 1 repair round	74.2	6.52	0.075	1.95	0.29	+1.66
+ 3 repair rounds	**75.0**	6.50	0.077	3.08	0.54	+2.54

5.1 Characterizing Inter-Trajectory Dependence

Table 1 compares BoN and repair under a common task-specific selector. The two strategies still differ in trajectory dependence and per-candidate denoising budget, so we use the following candidate-count-matched experiment to probe coverage directly. To characterize answer dependence, we fix LLaDA-8B and temperature t=0.4, generate N=16 candidates under each strategy, and evaluate coverage using oracle pass@k rather than learned or heuristic selection. We compare three regimes for generating candidates: **Independent sampling**, where each candidate starts from a fully masked sequence; **ODD**, where all candidates are generated in one batched run with diversity repulsion strength α_{ODD}=5; and **sequential repair**, where one initial candidate is followed by 15 repair rounds. The initial repair candidate is a stochastic t=0.4 sample drawn with the same generation procedure as one independent candidate, so the two arms start from the same distribution and the observed dependence cannot be attributed to a deterministic starting point. Oracle evaluation measures candidate coverage independently of selector quality. This design controls candidate count, temperature, and evaluation while leaving total forward-pass counts unequal: sequential repair at α=0.2 uses approximately 454 forward passes on Countdown, compared with 1,024 for independent sampling. We use this experiment to measure dependence structure; Appendix D.3 provides the corresponding comparison under approximately matched generation budgets.

Table 2 shows a consistent relationship between answer agreement and pass@16. Table 25 reports a sweep over α_{ODD}. Repair and independent sampling begin with similar pass@1, but repair has much higher agreement and lower pass@16. ODD exhibits intermediate agreement and coverage, with lower single-candidate accuracy. Figure 2 visualizes this quality–dependence trade-off together with its coverage and diversity coordinates.

The divergence at pass@16 despite similar initial accuracy shows that successive repair rounds add few new answer states. Figure 3 shows how unique-answer coverage and pass@k grow with candidate count and how the remasking sweep changes agreement; corresponding results across the other evaluated models appear in Appendix D.2.

Resampling-then-repair validation. As a lightweight actionable check of the diagnosis, we resample k=8 independent Countdown trajectories with LLaDA-8B, then apply one or three local repair rounds per candidate. Table 3 shows that the hybrid preserves the coverage of resampling:

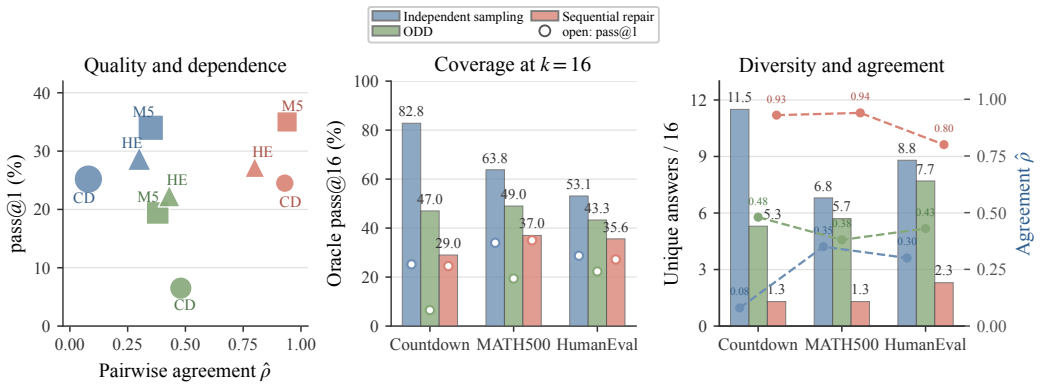


Figure 2: The comparison in Table 2. CD, M5, and HE denote Countdown, MATH500, and HumanEval. Left: agreement versus pass@1, with task markers and bubble area indicating pass@16. Center: pass@16 bars and pass@1 circles. Right: unique-answer bars and dashed agreement curves. The same strategy colors apply consistently across all three panels.

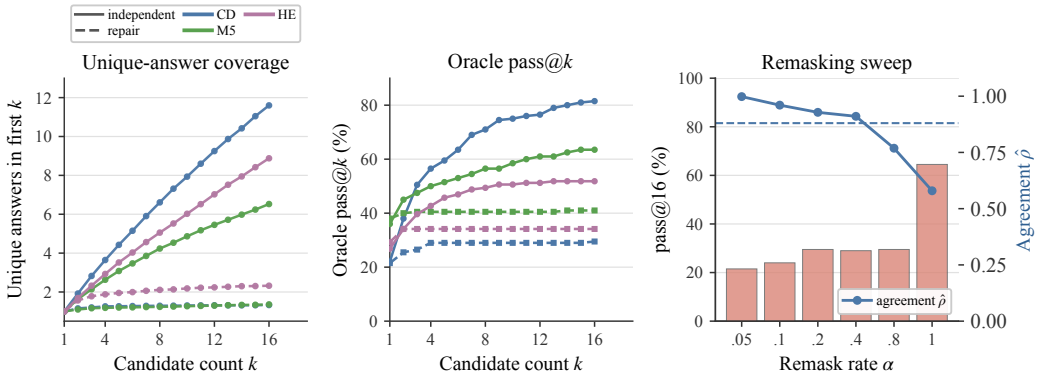


Figure 3: Single-seed diagnostics for LLaDA-8B (seed 42, N=16, t=0.4). Left and center: cumulative unique answers and oracle pass@k, with task colors and strategy line styles. Right: Countdown remasking sweep, showing pass@16 bars, an agreement curve, and the independent reference dashed; α=1 uses a joint-update restart. Table averages can differ because they include additional seeds.

it preserves the low-dependence regime established by independent initialization, while standard sequential repair remains concentrated around roughly one answer per problem. Its W→C and C→W rates show a small positive net drift, but increasing from one to three rounds changes pass@8 by less than one percentage point. These results suggest that test-time compute is better spent on generating independent answer hypotheses first, with local repair used only for refinement.

5.2 Probing Inter-Trajectory Dependence

In masked diffusion repair, the current answer and its surrounding rationale become part of the conditioning context for the next denoising step. Confidence-based remasking retains many high-confidence answer tokens, and bidirectional attention lets regenerated tokens condition on the preserved answer-bearing context. The resulting chain can change surface text while remaining concentrated around the same answer states.

Semantic canonicalization preserves most of the coverage gap: grouping answers by rational value (Countdown), mathematical equivalence (MATH500), or test-input behaviour (HumanEval) retains 81.6%, 94.4%, and 67.5% of the exact-match gap (Appendix C.7). Attention blocking provides a complementary intervention. On LLaDA-8B, blocking the retained answer span lowers $\hat{\rho}$ from 0.826 to 0.779, while blocking an equal number of randomly chosen retained positions lowers it to 0.691; unique answers rise from 1.80 to 1.99 and 2.55, respectively. The same direction appears

on LLaDA-1.5 and LLaDA-MoE, although all conditions remain far from independent sampling ($\hat{\rho}$=0.073, unique answers = 11.56). These results associate access to retained context with answer concentration; the small coverage changes and limited replication support a descriptive interpretation (Appendix C.8). Pairwise agreement remains high across rounds, and the right panel of Figure 5 (Appendix C.1) shows small later-round accuracy changes. Deterministic and stochastic chains exhibit the same persistent dependence (Appendix B.1), while lag-d agreement shows persistent answer similarity across rounds (Appendix B.2). LLaDA-MoE provides a clear qualitative example. Output text often changes substantially after repair, while correctness transitions vanish: rationales are rewritten around persistent conclusions. On Countdown, LLaDA-8B places answer-span tokens in the remasked set at only $0.557\times$ the uniform rate (Table 9). For LLaDA-MoE under deterministic denoising, 88.7% of output texts change at r=1, yet W→C and C→W rates are 0% for every round $r=1,\ldots,7$, showing how surface variation can coexist with stable answer states.

Answer exposure and repair benefit vary separately. Across four models and two tasks, the answer-exposure ratio correlates with repair agreement ($\rho_s = -0.881$), but has weak correlations with the coverage gap ($+0.071$) and repair gain (-0.095). These are descriptive associations across eight model–task cells (Appendix B.3). Dream exposes answer tokens frequently on both Countdown and MATH500, yet only Countdown shows positive net correction. Exposure characterizes opportunities for answer change; the correctness of those changes requires a separate evaluation.

Remasking strategy ablation. We evaluate four alternatives to confidence-based remasking: random remasking, forced answer-span remasking, PRM-guided suffix remasking, and their combination. For PRM-guided variants, Qwen2.5-Math-PRM-7B selects the first low-scoring segment in textual order. Table 11 (Appendix C.1) reports accuracy at r=3 and r=7. All variants remain in a high-dependence, low-coverage regime, with pass@16 close to standard repair and below independent sampling. Forced answer-span remasking exposes the answer directly, while PRM guidance targets a plausible local error; neither substantially broadens the answer distribution. Appendix C.2 extends these diagnostics to additional tasks and LLaDA-family models.

Uniform random remasking selects answer tokens at approximately rate α and also gives limited gains (Appendix C.3). A task-verifier-guided Countdown variant reaches 30.5% pass@16 versus 81.5% for independent sampling (Appendix C.2), extending the comparison to task-defined correctness feedback with heuristic error targeting.

Parameter sweeps do not restore coverage. The observed pattern remains consistent across remask ratios and temperatures. Increasing α lowers $\hat{\rho}$, but unique-answer counts remain low and pass@16 gains plateau well below independent sampling for $\alpha \leq 0.8$ (left panel of Figure 5). The chain becomes less correlated by pairwise agreement yet still fails to produce answer coverage comparable to independent sampling across the evaluated problems.

At α=1.0, the previous completion is fully removed and the operator becomes a restart. Coverage then depends on the regeneration schedule: the original joint-update variant remains concentrated, while a restart using the independent sampler's blockwise kernel reaches 80.0 pass@16 versus 83.0 for independent sampling on Countdown, with matched unique-answer counts (Appendix C.6). This control identifies a schedule effect at full remasking. For partial remasking, retained state, chain construction, and denoising allocation remain coupled in the evaluated procedures (Appendix C).

Taken together, these diagnostics show that sequential refinement can change surface phrasing and occasionally correct an answer while retaining strong dependence between successive answer states. Independent restart trajectories convert the same model's stochasticity into broader answer coverage.

6 Discussion

The restart controls show why the generation schedule matters alongside forward-pass budgets. With no retained response state, joint-update restarts reach 46.8% pass@16 on Countdown, while blockwise restarts reach 80.0% and independent sampling reaches 83.0% (Appendix C.6). The large full-remask deficit therefore reflects a schedule mismatch. The observed partial-repair deficit combines retained context, chain construction, denoising allocation, and regeneration schedule; their separate contributions remain unresolved by these controls.

Task structure modulates the magnitude of the effect. Coverage and selection are separate quantities: oracle selection can use a diverse candidate set, while a concentrated set offers few alternatives even to a perfect selector operating over the same candidate pool.

Trajectory-level guidance and search are natural complements to local token-level repair. Methods such as RFG (Chen et al., 2025b), Prism (Bai et al., 2026), S^3 (Bilal et al., 2026), and diffusion search frameworks (Zhang et al., 2025) can be evaluated with the same coverage and agreement diagnostics to determine whether they broaden the answer distribution.

Implications for future dLLM test-time scaling. Three practical lessons follow. (1) *Report answer coverage directly.* Unique answers per problem measures candidate-set growth, while $\hat{\rho}$ measures answer agreement; together they show whether extra samples explore new hypotheses. (2) *Report coverage alongside selection.* Oracle pass@k measures the available answer pool, while verifier-guided selection (Lightman et al., 2024; Snell et al., 2024) measures how effectively a system uses that pool. (3) *Allocate compute to cross-trajectory exploration when coverage is the objective.* In the tested settings, independent initialization preserves coverage more reliably than extending one repair chain, motivating parallel sampling, diversity objectives, and latent-space exploration.

Beyond verifiable answers. Our diagnosis is sharpest on tasks with a well-defined extracted answer. Appendix E reports instruction-following (IFEval) and creative-writing (Arena-Hard) results: repair is locally competitive with a fresh sample on some Arena-Hard comparisons, scoring 58.5% $[53.0, 64.0]$ against an independent sample for LLaDA-8B and 46.6% for Dream, while IFEval strict-prompt accuracy changes by at most 1.9 points. These results show that repair can alter open-ended responses, while the answer-level coverage diagnostic is most informative on verifiable tasks.

The controlled comparison separates candidate quality from coverage. On Countdown, repair retains a similar pass@1 to independent sampling but reaches only 29.0% oracle pass@16 with 1.3 unique answers, compared with 82.8% and 11.5 for independent sampling. The same ordering on MATH500 and HumanEval shows how the candidate pool can limit selection despite comparable accuracy for the initial candidates produced before any additional repair rounds.

The interventions separate changes in the candidate pool from improvements in correctness. Blocking retained positions or increasing the remask ratio can lower agreement and increase unique answers while leaving a large oracle-coverage gap. Answer-span exposure produces both W$\rightarrow$C and C$\rightarrow$W transitions, so the direction of the resulting accuracy change depends on their balance. The cross-model exposure audit and task-verifier intervention make this distinction explicit: access to new answer states can be measured separately from the ability to reach correct ones. Reporting both quantities helps assess which part of candidate generation improves with additional inference compute.

For practical inference, independent trajectories should provide the outer search budget, with repair, verification, or reranking reserved for candidates that need refinement. Evaluation should report pass@1, unique-answer count, agreement, oracle pass@k, and forward-pass cost at matched candidate counts. This separates gains from better candidates, broader pools, and more effective selection, while clarifying cross-task comparisons because denoising-step counts do not directly measure answer-space exploration. Combining broad candidate generation with targeted refinement therefore offers a direct way to spend test-time compute where it improves coverage and final selection.

7 Conclusion

Additional test-time compute is most useful when it produces new answer hypotheses. Across the evaluated dLLMs and verifiable tasks, sequential repair changes local text while retaining strong answer-level dependence, whereas independent restart trajectories broaden candidate coverage. Sweeps, budget comparisons, and remasking interventions characterize this dependence across repair settings, and full-remask controls show that matching the blockwise generation schedule restores independent-sampling coverage. These results motivate coverage-aware allocation of inference compute; the scope and limitations of these findings are discussed in Appendix G.

Reproducibility statement

All models and benchmarks are public, and no model is retrained. Appendix A specifies the decoding configurations, prompts, selection rules, seed protocol, and measured forward-pass accounting. Section 3 defines the answer-level diagnostics. Appendices C and D describe the intervention and budget comparisons, and Appendix E specifies the open-ended evaluation protocol.

AI use statement

Generative AI tools assisted manuscript editing, literature and reference checks, figure preparation, implementation and debugging of evaluation code, and feedback on methodology and interpretation. The evaluated dLLMs generate the experimental responses, and Qwen2.5-Math-PRM-7B supplies guidance for the remasking variants (Section 5.2). GPT-5.5 is used as an external judge for the PRM localization audit and Arena-Hard comparisons (Appendices C.2 and E); these scores are model-based judgments. Reported measurements are computed from the recorded experimental outputs. The authors take responsibility for the final text, claims, code, and artifacts, including all AI-assisted contributions to manuscript preparation and experimental analysis.

References

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021a.

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

Jinbin Bai, Yixuan Li, Yuchen Zhu, Yi Xin, Qingyu Shi, Aosong Feng, Xiaohong Liu, Molei Tao, Jianru Xue, Xiangtai Li, et al. Prism: Efficient test-time scaling via hierarchical search and self-verification for discrete diffusion language models. *arXiv preprint arXiv:2602.01842*, 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 D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 55981–56007. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/510e22f4a2da5212a64f1591736e2eaf-Paper-Conference.pdf.

Ahsan Bilal, Muhammad Ahmed Mohsin, Muhammad Umer, Asad Aali, Muhammad Usman Khanzada, Muhammad Usman Rafique, Zihao He, Emily Fox, and Dean F. Hougen. s^3: Stratified scaling search for test-time in diffusion language models, 2026. URL https://arxiv.org/abs/2604.06260.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Ranfei Chen, Ming Chen, and Kaifei Wang. Reasoning in diffusion large language models is concentrated in dynamic confusion zones, 2025a. URL https://arxiv.org/abs/2511.15208.

Tianlang Chen, Minkai Xu, Jure Leskovec, and Stefano Ermon. Rfg: Test-time scaling for diffusion large language model reasoning with reward-free guidance. *arXiv preprint arXiv:2509.25604*, 2025b.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. In *The Twelfth International Conference on Learning Representations*, 2024.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

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

Xin Dai, Pengcheng Huang, Zhenghao Liu, Shuo Wang, Yukun Yan, Chaojun Xiao, Yu Gu, Ge Yu, and Maosong Sun. Revealing the attention floating mechanism in masked diffusion models. *arXiv preprint arXiv:2601.07894*, 2026.

Zhengnan Guo and Fei Tan. Lost in diffusion: Uncovering hallucination patterns and failure modes in diffusion large language models. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026.

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. *arXiv preprint arXiv:2103.03874*, 2021.

Feng Hong, Geng Yu, Yushi Ye, Haicheng Huang, Huangjie Zheng, Ya Zhang, Yanfeng Wang, and Jiangchao Yao. Wide-in, narrow-out: Revokable decoding for efficient and effective dllms. *arXiv preprint arXiv:2507.18578*, 2025.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*, 2024.

Zemin Huang, Yuhang Wang, Zhiyang Chen, and Guo-Jun Qi. Don't settle too early: Self-reflective remasking for diffusion language models. *arXiv preprint arXiv:2509.23653*, 2025.

Zheng Huang, Kiran Ramnath, Yueyan Chen, Aosong Feng, Sangmin Woo, Balasubramaniam Srinivasan, Zhichao Xu, Kang Zhou, Shuai Wang, Haibo Ding, et al. Diffusion language model inference with monte carlo tree search. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 3493–3512, 2026.

inclusionAI. LLaDA-MoE-7B-A1B-Instruct. Hugging Face model card, 2025. URL https://huggingface.co/inclusionAI/LLaDA-MoE-7B-A1B-Instruct.

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

Michael Katz, Harsha Kokel, and Sarath Sreedharan. Seemingly simple planning problems are computationally challenging: The countdown game. *arXiv preprint arXiv:2508.02900*, 2025.

Jiyeon Kim, Sungik Choi, Yongrae Jo, Moontae Lee, and Minjoon Seo. Early decisions matter: Proximity bias and initial trajectory shaping in non-autoregressive diffusion language models, 2026. URL https://arxiv.org/abs/2604.10567.

Sean Lamont, Christian Walder, Paul Montague, Amir Dezfouli, and Michael Norrish. Free lunch for pass@ k? low cost diverse sampling for diffusion language models. *arXiv preprint arXiv:2603.04893*, 2026.

Jihoon Lee, Hoyeon Moon, Kevin Zhai, Arun Kumar Chithanar, Anit Kumar Sahu, Soummya Kar, Chul Lee, Souradip Chakraborty, and Amrit Singh Bedi. Test-time scaling in diffusion llms via hidden semi-autoregressive experts. *arXiv preprint arXiv:2510.05040*, 2025.

Pengxiang Li, Yefan Zhou, Dilxat Muhtar, Lu Yin, Shilin Yan, Li Shen, Soroush Vosoughi, and Shiwei Liu. Diffusion language models know the answer before decoding. *arXiv preprint arXiv:2508.19982*, 2025.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*, 2024.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The twelfth international conference on learning representations*, 2024.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2381–2391, 2018.

Roy Miles, Aysim Toker, Andreea-Maria Oncescu, Songcen Xu, Jiankang Deng, and Ismail Elezi. Test-time scaling with diffusion language models via reward-guided stitching. *arXiv preprint arXiv:2602.22871*, 2026.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

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 *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Maximo Eduardo Rulli, Simone Petruzzi, Edoardo Michielon, Fabrizio Silvestri, Simone Scardapane, and Alessio Devoto. Attention sinks in diffusion language models. *arXiv preprint arXiv:2510.15731*, 2025.

Yash Shah, Abhijit Chakraborty, Naresh Kumar Devulapally, Vishnu Lokhande, and Vivek Gupta. Oscar: Orchestrated self-verification and cross-path refinement. *arXiv preprint arXiv:2604.01624*, 2026.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pp. 8634–8652, 2023.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4149–4158, 2019.

Guanghan Wang, Yair Schiff, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. *arXiv preprint arXiv:2503.00307*, 2025.

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

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Lin Yao. Remask, don't replace: Token-to-mask refinement in diffusion language models. *arXiv preprint arXiv:2604.18738*, 2026.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: deliberate problem solving with large language models. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pp. 11809–11822, 2023.

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.

Kevin Zhai, Sabbir Mollah, Zhenyi Wang, and Mubarak Shah. Core: Context-robust remasking for diffusion language models. *arXiv preprint arXiv:2602.04096*, 2026.

Xiangcheng Zhang, Haowei Lin, Haotian Ye, James Zou, Jianzhu Ma, Yitao Liang, and Yilun Du. Inference-time scaling of diffusion models through classical search. *arXiv preprint arXiv:2505.23614*, 2025.

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.

Fengqi Zhu, Rongzhen Wang, Shen Nie, Xiaolu Zhang, Chunwei Wu, Jun Hu, Jun Zhou, Jianfei Chen, Yankai Lin, Ji-Rong Wen, et al. Llada 1.5: Variance-reduced preference optimization for large language diffusion models. *arXiv preprint arXiv:2505.19223*, 2025.

APPENDIX CONTENTS

- **Appendix A: Experimental Setup** 15
- **Appendix B: Additional Evidence for Commitment Dynamics** 17
  - Appendix B.1: Deterministic and Stochastic Repair Accuracy 17
  - Appendix B.2: Lag-d Agreement 17
  - Appendix B.3: Dream Repair: Cross-Task Mechanism Analysis 19
- **Appendix C: Robustness and Alternative Repair Operators** 20
  - Appendix C.1: Remasking Strategy Ablation 20
  - Appendix C.2: Answer-Span and Guided-Repair Interventions 20
  - Appendix C.3: Random Remasking 23
  - Appendix C.4: Alpha Sweep 24
  - Appendix C.5: Temperature Sensitivity 25
  - Appendix C.6: Blockwise Restart at Full Remasking 25
  - Appendix C.7: Semantic Canonicalization of Answers 26
  - Appendix C.8: Attention-Blocking Intervention 27
  - Appendix C.9: Repair Under the Main-Table Selector 27
  - Appendix C.10: Selection Rules versus Coverage 28
- **Appendix D: Extended Strategy and Model Comparisons** 28
  - Appendix D.1: Strategy Space of Controlled Runs 28
  - Appendix D.2: Cross-Model Replication of Trajectory Dependence 29
  - Appendix D.3: Compute-Matched Comparison 29
  - Appendix D.4: ODD Diversity-Repulsion Sweep 29
  - Appendix D.5: Additional Benchmark Snapshot 29
- **Appendix E: Open-Ended Evaluation** 30
- **Appendix F: Case Studies** 32
- **Appendix G: Limitations** 32
- **Appendix H: Broader Impacts** 32

A EXPERIMENTAL SETUP

Models. Our primary model is LLaDA-8B-Instruct (Nie et al., 2025). We additionally evaluate LLaDA-1.5 (Zhu et al., 2025), LLaDA-MoE-7B-A1B-Instruct (inclusionAI, 2025), and Dream-Instruct-7B (Ye et al., 2025), covering two dLLM families at varied scales. Qwen3-8B-Instruct (Yang et al., 2025) is included as an autoregressive reference. All models are publicly released pretrained or instruction-tuned checkpoints; no models are retrained in this work.

Decoding configuration. Table 4 lists the generation configuration used for every benchmark. All four dLLMs use the task-specific full-trajectory response length L, denoising step count T, and block length listed below. Repair retains the same response length and regenerates the selected positions using the local operator in Section 3. A greedy trajectory costs T forward passes and defines the $1\times$ budget unit used throughout; the per-round repair cost C_{rep} follows the expression in Section 3. These budgets are smaller than the configurations used in the LLaDA release (which evaluates HumanEval at L=512), so our greedy numbers are below the published single-trajectory results for these checkpoints. Because every strategy is run under the identical configuration, the repair-versus-BoN comparison remains internally matched, but absolute accuracies should not be read as reproductions of the model cards.

Table 4: Decoding configuration per benchmark, shared by all models and all strategies. L is the response length and T the number of denoising steps of a single full trajectory, which defines the $1\times$ forward-pass unit. Full trajectories use blockwise decoding with block length 32 and low-confidence remasking; local repair regenerates the remasked positions jointly.

Benchmark	L	T
Countdown	128	64
MATH500, GSM8K	256	128
HumanEval, MBPP	256	128
ARC-C, OpenBookQA, CommonsenseQA	512	256

Seeds. Each arm of Table 1 follows a fixed protocol. Greedy decoding is deterministic (t=0) and is run once per model–task pair. Every BoN cell is the mean of three independent runs at seeds 42, 43, and 44, without exception across all 36 cells. The repair columns are likewise the mean of three independent chains at seeds 42, 43, and 44; within each seed, r=8, 18, and 38 are three budgets read off the same chain, which is what the chain-level diagnostics ($\hat{\rho}$, unique-answer count, lag-d agreement) are defined on. Uncertainty is reported as a paired bootstrap over per-problem correctness (Section 4), resampling problem indices jointly for greedy and repair after averaging each problem's correctness across the three seeds. The headline greedy and BoN values are checked against stored run summaries, and the repair values against saved per-problem selector outputs. Not every diagnostic block was run with all three seeds. Independent-sampling references use the seed set of their experimental block and can differ slightly between tables: Countdown oracle pass@16 appears as 82.8 or 83.0 against a per-seed range of 81.5–84.5, with the corresponding unique-answer counts 11.46 and 11.56 and agreements 0.076 and 0.073. These differences reflect seed and subset variation; the seed-42 ODD reference is identified separately in Table 2.

Tasks. **Countdown** (n=256): a constrained symbolic arithmetic task in which the model must combine the given numbers using basic operations to reach a target value; correctness is verified by exact symbolic evaluation. **MATH500** (n=500): a diverse mathematical reasoning benchmark; correctness is determined by benchmark-standard answer extraction and equivalence checking. **HumanEval** (n=164): a Python code-generation benchmark evaluated by execution-based unit tests. Appendix D.5 extends the evaluation to further code-generation, arithmetic-reasoning, and multiple-choice benchmarks, including MBPP, GSM8K, ARC-Challenge, OpenBookQA, and CommonsenseQA, as detailed in that appendix.

Evaluation subsets. The headline accuracy tables (Table 1 and Table 26) use the full test sets listed above. The diagnostic analyses, which require storing 16 or 18 candidates per problem, instead use a fixed 200-problem subsample of Countdown and of MATH500; HumanEval is small enough that all

164 problems are used. This is the "reduced evaluation subset" referred to in the controlled, sweep, and guided-repair sections, and it applies to Tables 2, 3, 11, 12, 15, 17, 18, 19, 20, 22, 23, and 24. The subsample is drawn without replacement using the run seed, so seeds 42, 43, and 44 evaluate three different 200-problem subsets. Cross-seed spread in the diagnostic tables therefore reflects both decoding randomness and subset composition and should not be read as a pure decoding-variance estimate. Diagnostic comparisons use the candidate count and seed set specified for each experiment; single-seed curves in Figures 3 and 5 can therefore differ from multi-seed table averages. Within each independent-sampling/repair run, both strategies use the identical subset. The ODD rows of Table 2 are single-seed references, as noted in its caption.

Candidate selection protocol. BoN always generates N independent full trajectories; the selector depends on what feedback the task makes available. Countdown uses exact symbolic verification: the first candidate whose expression uses the provided numbers exactly once and evaluates to the target is selected, with confidence fallback only when no candidate verifies. HumanEval and MBPP use execution-based unit-test selection: the first candidate passing the supplied tests is selected. MATH500 and GSM8K use self-consistency over parsed final answers, grouping mathematically equivalent answers when possible, and fall back to model confidence when no answer has majority support. Multiple-choice benchmarks use the analogous parsed-choice self-consistency with confidence fallback. In the repair chains, missing per-round confidence scores require a final-round fallback when self-consistency has no winner (Appendix C.9). On HumanEval and MBPP, selection uses the evaluation tests themselves, so these numbers describe test-oracle selection. The selected BoN columns combine candidate coverage and task-specific selection. The oracle pass@k analyses remove this selector component and isolate coverage.

Matched-budget accounting. We organize comparisons by denoising forward-pass budget, using greedy decoding as the $1\times$ baseline. For BoN-N, budget scales linearly with the number of independently denoised full candidates. For Repair-R, budget includes the initial denoising trajectory and the additional passes used by remasking rounds. Every repair budget we report is the measured forward-pass count logged per run, not the nominal $T+RT/2$; Section 3 gives the per-round cost, which is smaller than $T/2$ whenever $\lfloor \alpha L \rfloor$ is. Relative to the BoN column it accompanies in Table 1, repair at r=8, 18, and 38 costs a measured $4.25\times$, $8.31\times$, and $16.44\times$ greedy—6.2%, 3.9%, and 2.7% *more* forward passes than BoN-4/8/16—so the repair columns are compared under a marginally larger budget than the independent-sampling columns they trail. Within the dLLM family, this gives a unified axis for comparing exploration-heavy and revision-heavy inference. This accounting covers generation forward passes; it excludes verifier execution, PRM scoring, and external judge calls, and therefore does not equate wall-clock time or total FLOPs.

Compute resources. All experiments were run on NVIDIA A800 GPUs (80 GB HBM2e). The primary model (LLaDA-8B-Instruct) fits on a single GPU for inference; Large-scale runs (e.g., Repair-R=38 on MATH500 n=500) took roughly 4–8 hours per configuration; additional sweeps and replications, including alpha sweeps, temperature sweeps, and multi-seed replication, took 1–4 hours each. The total compute across all reported experiments is estimated at approximately 300–400 GPU-hours. Additional preliminary runs not reported in the paper consumed a comparable amount of compute during exploratory evaluation.

Autoregressive reference. Table 5 reports Qwen3-8B-Instruct as an external autoregressive reference. It is included only to contextualize the absolute capability gap; its decode-step counts are not compute-equivalent to dLLM denoising passes and are not a matched-budget comparison.

Table 5: Qwen3-8B-Instruct autoregressive reference results. BoN uses independent autoregressive samples at t=0.4; greedy uses deterministic decoding. The remasking-based repair operator is not applicable to causal autoregressive models.

Dataset	Greedy	BoN-4	BoN-8
Countdown	89.1	98.4	98.4
MATH500	62.4	66.8	68.8
HumanEval	84.1	90.2	93.3

Temperature protocol. Greedy decoding is deterministic. In the main comparison, both repair and BoN use stochastic decoding at t=0.4, ensuring that exploration and revision are evaluated under the same temperature regime. Repair at t=0.0 and the deterministic fixed-point analysis are discussed separately in Appendix B.1. Temperature sensitivity across $t \in \{0.4, 0.7, 1.0\}$ for the two stochastic generation strategies is reported in Appendix C.5.

Statistical reporting. We quantify binomial-proportion uncertainty using Wilson 95% confidence intervals. Near p=0.4, the corresponding half-widths are approximately ±6.0 pp on Countdown, ±4.3 pp on MATH500, and ±7.5 pp on HumanEval. These Wilson intervals describe single-run proportions. The headline table reports three-seed means; the paired bootstrap averages each problem's correctness across those seeds before resampling problems. Its intervals are conditional on the evaluated seed set and do not separately estimate variation across new decoding seeds. Small differences without a reported paired interval are descriptive. Paired comparisons, such as repair vs. greedy, use paired bootstrap over per-problem correctness. A difference is labeled not statistically significant when the 95% CI for the paired difference includes zero.

Prompt templates. Figure 4 shows the exact system prompts used for all benchmarks. MATH500 and GSM8K share the same mathematical reasoning prompt; ARC-Challenge, OpenBookQA, and CommonsenseQA share the multiple-choice prompt.

B Additional Evidence for Commitment Dynamics

B.1 Deterministic and Stochastic Repair Accuracy

Table 6 compares deterministic and stochastic repair accuracy at r=7 for LLaDA-8B. Across all three tasks, neither repair regime approaches BoN-16 accuracy, indicating that iterative repair does not recover the coverage obtained by independent candidate generation. Repair does not consistently improve over greedy decoding, with gains on some tasks but regressions or only marginal changes on others. These results show that changing the denoising temperature alone does not yield a robust accuracy improvement or close the gap to BoN-16.

Table 6: Accuracy comparison at r=7 for deterministic and stochastic repair on LLaDA-8B. BoN-16 is a higher-budget reference with 16 candidates; the repair chain contains 8 states. Both repair columns report the *unselected* accuracy of round r=7 itself, averaged over seeds 42–44, whereas Table 1 applies the chain selector over rounds $0..r$; this read-out difference is why the stochastic Countdown entry here (22.7) sits below the selected r=8 value (27.2) in Table 1. Table 21 reports both read-outs side by side to make the effect of chain selection explicit.

Task	Greedy	Repair (t=0.0)	Repair (t=0.4)	BoN-16
Countdown	25.4	28.9	22.7	83.2
MATH500	33.4	33.0	34.0	40.1
HumanEval	26.8	33.5	29.5	46.7

B.2 Lag-d Agreement

Table 7: Lag-d agreement (LLaDA-8B, N=16, t=0.4).

d	Countdown	MATH500	HumanEval
1	0.934	0.947	0.838
2	0.927	0.945	0.847
4	0.915	0.950	0.824
8	0.896	0.941	0.799
15	0.730	0.875	0.433

For a repair chain of length N, the lag-d agreement ρ_d is the probability that the answers at rounds k and $k+d$ coincide, averaged over all valid pairs in the chain. Table 7 reports measured values for

Countdown — Symbolic Arithmetic

```
Using only the provided numbers, create an arithmetic expression that
evaluates to exactly the provided target number. You may use the
operations +, −, ×, and ÷ as needed, but each number must be used
exactly once. Think step-by-step. After reasoning, provide only your
final expression inside \boxed{} tags without including an equals
sign or the target number. For example: \boxed{a + b × c}

Respond in the following format:
<reasoning>
Your reasoning here
</reasoning>
<answer>
\boxed{...}
</answer>
```

MATH500 / GSM8K — Mathematical Reasoning

```
You are a math expert. You will be given a question to solve.
Solve it step by step. Wrap the final answer in a \boxed{}.

Respond in the following format:
<reasoning>
Your reasoning here
</reasoning>
<answer>
\boxed{...}
</answer>
```

HumanEval — Code Generation

```
Complete the following Python function. Continue the code after
the prompt. Return only valid Python code. Do not add markdown
fences, explanations, or prose.
```

MBPP — Code Generation

```
Write a Python function to solve the following problem.
Return only the function definition. Do not add markdown
fences, tests, or explanations.
```

ARC-Challenge / OpenBookQA / / CommonsenseQA — Multiple Choice

```
Answer the following multiple-choice question by reasoning
step by step. End your response with 'The answer is (X)'
where X is the letter of the correct choice.

Respond in the following format:
<reasoning>
Your reasoning here
</reasoning>
<answer>
The answer is (X)
</answer>
```

Figure 4: System prompts used for all benchmarks.

LLaDA-8B (N=16, t=0.4). Agreement decays slowly: on Countdown it remains above 0.80 through lag 8 before dropping to 0.73 at lag 15; on MATH500 it stays above 0.87 throughout; on HumanEval it remains near 0.80 through lag 8 before dropping more sharply at long lags. The persistence of agreement over multiple lags indicates slow exploration of distinct answer states over the repair horizon covered by the evaluated sequence of candidates.

B.3 Dream Repair: Cross-Task Mechanism Analysis

Table 8 shows that Dream exhibits task-dependent repair dynamics: corrective drift on Countdown, near-neutral drift on MATH500, and degradative drift on HumanEval. This pattern is consistent with the interaction between task structure and local remasking: short arithmetic outputs can be corrected by small edits, whereas long mathematical derivations and executable code are more vulnerable to disruptions that extend beyond the locally remasked span.

Table 8: Dream repair correctness-transition rates (greedy → repair-r=3). The net drift, W→C − C→W, summarizes the balance between corrections and regressions.

Task (n)	W→C	C→W	Net drift	Regime
Countdown (256)	21.1%	7.0%	+14.1 pp	Corrective
MATH500 (500)	3.8%	3.6%	+0.2 pp	Neutral walk
HumanEval (164)	6.7%	19.5%	−12.8 pp	Degradative

Answer exposure across models and tasks. We extend the confidence comparison to four models on Countdown and MATH500. For up to 100 deterministic (t=0) responses per model–task pair, we measure mean confidence within the extracted answer span and its exposure to confidence-based remasking. Define the exposure ratio as the fraction of extracted answer tokens selected by the lowest-confidence $\lceil \alpha L \rceil$ mask, divided by α=0.2; values below one indicate relative protection and values above one indicate relative exposure. This audit rounds the mask count upward, while repair uses $\lfloor \alpha L \rfloor$. The "Any" rate counts responses with at least one answer token in that set. Valid span counts range from 58 to 100, with only 58 valid Dream-Countdown responses, so the confidence and exposure summaries cover responses with successfully extracted answer spans.

Table 9 pairs these measurements with outcomes from separate stochastic diagnostic runs (t=0.4): repair agreement and the independent-minus-repair pass@16 gap at N=16, and the unselected round-7 accuracy minus greedy accuracy. These outcomes belong to the diagnostic audit and differ from the multi-seed, chain-selected headline results in Table 1.

Table 9: Answer confidence, exposure, and repair outcomes across four models and two tasks. CD denotes Countdown and M5 denotes MATH500. n counts valid answer spans; "Any" is a percentage. Gap and Gain are percentage-point differences under the diagnostic protocols specified in the text.

Model	Task	n	Conf.	Exposure	Any (%)	$\hat{\rho}$	Gap	Gain
LLaDA-8B	CD	98	0.963	0.557	93.9	0.928	52.0	-2.7
LLaDA-8B	M5	97	0.986	0.376	52.6	0.943	22.5	+0.6
LLaDA-1.5	CD	97	0.952	0.502	91.8	0.951	57.7	+4.1
LLaDA-1.5	M5	100	0.986	0.250	47.0	0.959	28.2	+0.8
LLaDA-MoE	CD	94	0.688	0.390	80.9	0.973	49.7	+0.5
LLaDA-MoE	M5	90	0.599	0.871	76.7	0.869	35.0	-0.7
Dream-7B	CD	58	0.014	4.253	100.0	0.656	26.0	+11.9
Dream-7B	M5	94	0.008	3.986	98.9	0.857	35.5	-0.5

Exposure is associated with agreement ($\rho_s = -0.881$), while its correlations with the coverage gap (+0.071) and repair gain (−0.095) are weak (Table 10). The task-centered agreement correlation is −0.833; the uncentered correlation has exact model-block p=0.042 across the 24 permutations. These small-sample, cross-protocol comparisons are descriptive. A separate problem bootstrap gives $[-0.952, -0.833]$ for the exposure–agreement correlation, conditional on the fixed outcome summaries; it captures uncertainty in the exposure means, not variation across new model families.

Table 10: Descriptive Spearman correlations across the eight model–task cells. Task-centered correlations subtract each task's mean from both quantities. The two-sided exact permutation test exchanges four model blocks while preserving the two tasks within each block; the eight cells represent four models evaluated on two tasks each.

Measurement	Outcome	ρ_s	Task-centered	Block p
Answer confidence	Agreement $\hat{\rho}$	+0.619	+0.548	0.167
Answer confidence	Coverage gap	-0.143	+0.024	0.583
Answer confidence	Repair gain	+0.048	-0.095	0.917
Exposure ratio	Agreement $\hat{\rho}$	-0.881	-0.833	0.042
Exposure ratio	Coverage gap	+0.071	+0.048	0.750
Exposure ratio	Repair gain	-0.095	+0.071	0.792
Any answer remasked	Agreement $\hat{\rho}$	-0.667	-0.714	0.042
Any answer remasked	Coverage gap	+0.286	+0.214	0.542
Any answer remasked	Repair gain	+0.048	-0.095	0.917

Dream has high exposure on both tasks, but positive repair gain only on Countdown. Thus answer accessibility characterizes how readily answers can change, while the usefulness of those changes must be assessed through the resulting correctness and coverage of the candidate pool.

C Robustness and Alternative Repair Operators

C.1 Remasking Strategy Ablation

Table 11 reports the remasking strategy ablation discussed in the main text. Figure 5 shows the remasking sweep and per-round accuracy changes on Countdown: early corrections are followed by small changes across later rounds.

Table 11: Remasking strategy ablation on MATH500 with LLaDA-8B. Accuracy at r=3 and r=7 is the unselected accuracy of that round, matching the read-out of Table 6; $\hat{\rho}$, Unique/16, and pass@16 are computed over 16 candidates from a single repair chain. Independent sampling serves as a trajectory-level reference for comparison with all five local remasking strategies.

Strategy	Acc r=3	Acc r=7	$\hat{\rho}$	Unique/16	pass@16
Independent sampling	–	–	*0.348*	*6.77*	*63.8%*
Low-confidence	34.1%	34.0%	0.941	1.34	37.0%
Random	32.2%	33.6%	0.938	1.28	37.5%
Answer-span (forced)	34.2%	32.2%	0.924	1.46	42.0%
PRM (low-score suffix)	34.3%	33.8%	0.960	1.25	40.5%
PRM + answer-span	35.2%	34.8%	0.914	1.53	42.0%

C.2 Answer-Span and Guided-Repair Interventions

This section collects local remasking interventions used to test whether changing the edited token subset can reduce inter-trajectory dependence. In addition to standard low-confidence remasking, we evaluate random remasking, forced answer-span remasking, PRM-guided suffix remasking, and a combination of PRM guidance with answer-span remasking. These interventions target plausible local failure sources, but they do not restore independent answer coverage. The main remasking ablations use the stochastic setting t=0.4. Unless otherwise specified, results in this subsection use this stochastic setting; the deterministic Answer-Span experiment below is reported separately as a mechanism-isolation diagnostic of local repair.

Confidence-based remasking selects the α fraction of tokens with lowest predicted confidence for remasking. For mathematical outputs containing a `\boxed{}` answer token, the model often assigns high predicted probability to the answer token early in denoising, so confidence-based remasking tends to exclude it across repair rounds. When answer tokens are frozen and reasoning tokens are

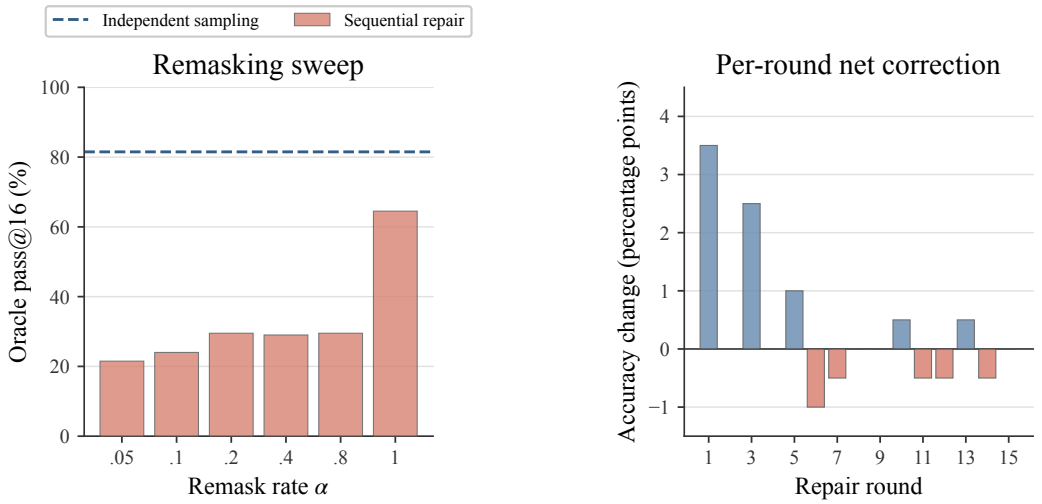


Figure 5: Countdown diagnostics (LLaDA-8B, seed 42, n=200, t=0.4). The left panel shows the remasking sweep; α=1 uses a joint-update restart. The right panel shows per-round accuracy changes at α=0.2, computed as unconditional W→C minus C→W rates with the common denominator n, following the accuracy-drift definition in Section 3.

remasked, bidirectional attention can regenerate reasoning conditioned on both the prompt and the frozen answer: the model may rationalize the existing answer rather than re-derive it, so correctness changes are unlikely when the answer token is never remasked.

Answer-state preservation varies across model families. LLaDA-MoE provides the clearest evidence: under deterministic denoising, 88.7% of output texts change at r=1, yet W→C and C→W transition rates are 0% for all r from 1 to 7, consistent with answer tokens remaining fixed. LLaDA-8B's answer tokens receive somewhat lower confidence scores, so occasional remasking occurs in early rounds, producing small but non-zero transition rates that remain approximately balanced.

We additionally run a deterministic Answer-Span intervention as a focused test of the fixed-point mechanism. Unlike the main stochastic ablations, this diagnostic uses t=0.0: answer-span tokens are forced into the remask pool at every repair round, so the answer is always regenerated rather than protected by confidence. This setting isolates whether answer-span exposure can break deterministic answer preservation without the confound of stochastic drift. Figure 6 compares the resulting per-round W→C and C→W rates against a matched deterministic standard-repair baseline.

Figure 7 summarizes conditional correction and regression rates for the five stochastic remasking strategies on MATH500. These rates use different denominators: previously incorrect responses for W→C and previously correct responses for C→W. Their difference therefore differs from the unconditional accuracy drift defined in Section 3. Together with Table 11, they show that exposing local errors permits both corrections and regressions while coverage remains below the coverage achieved by independent candidate generation.

Guided-repair diagnostics. Beyond answer-span exposure, we also test guided local edits that target plausible reasoning errors. Table 12 reports selected guided-repair diagnostics on answer-structured tasks for LLaDA-1.5 and LLaDA-MoE. We restrict these appendix-only runs to Countdown and MATH500 because the interventions rely on identifiable answer spans or PRM step scores; HumanEval involves code-level syntactic validity and is not directly targeted by these guidance operators. These runs use the same stochastic repair setting on the same reduced evaluation subset as the controlled analyses. Guided repair can help in some model and task pairs, but the gains are uneven and remain well below matched-budget BoN.

No single guided variant dominates across models, tasks, and repair rounds. Combining PRM guidance with answer-span remasking helps in selected configurations, but PRM-only or even standard repair is stronger in others. Guided variants can improve some settings, yet the gains remain

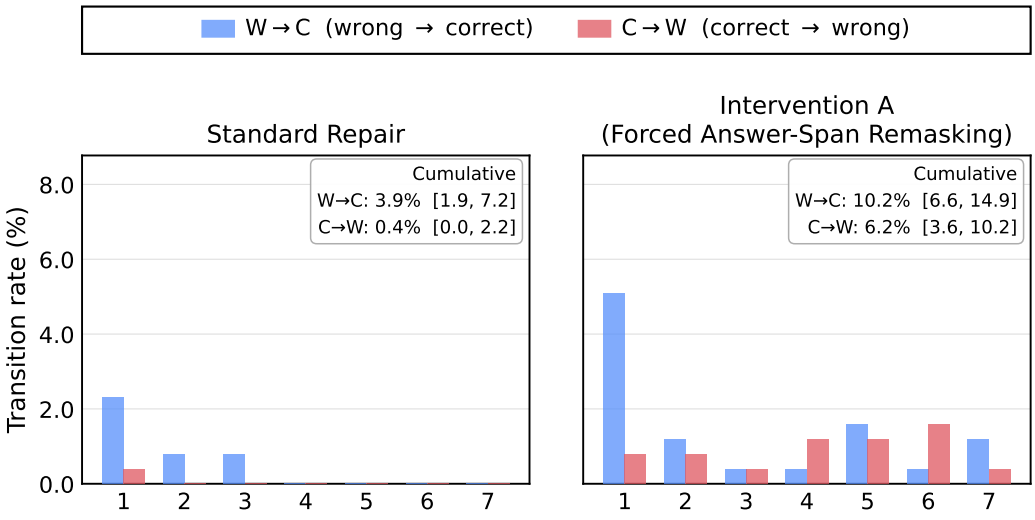


Figure 6: Per-round W→C and C→W transition rates for matched deterministic standard repair and the deterministic Answer-Span intervention. Forcing answer-span remasking breaks the fixed point but increases both repairs and corruptions (LLaDA-8B, Countdown, α=0.2, t=0.0, n=256).

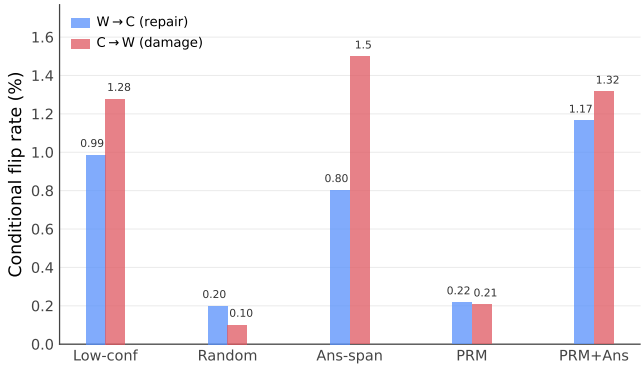


Figure 7: Mean conditional flip rates (W→C and C→W) across repair rounds for five remasking strategies (LLaDA-8B, MATH500, α=0.2, t=0.4). Each direction is normalized by the number of responses eligible for that transition.

Table 12: Cross-model guided repair variants. Standard repair and guided variants are evaluated on the same reduced controlled-analysis subset; guided variants are appendix-only runs.

Model	Task	Round	Standard	Answer span	PRM	PRM+ans
LLaDA-1.5	Countdown	r=3	21.9%	26.5%	29.5%	19.0%
		r=7	23.2%	24.0%	24.5%	26.5%
	MATH500	r=3	33.4%	37.5%	35.0%	38.5%
		r=7	34.0%	34.0%	32.0%	38.5%
LLaDA-MoE	MATH500	r=3	34.9%	35.0%	34.5%	34.5%
		r=7	34.7%	36.0%	40.5%	36.0%
	Countdown	r=3	41.9%	33.0%	38.5%	41.0%
		r=7	43.9%	35.5%	43.5%	44.5%

uneven and do not overturn the broader compute-allocation advantage of independent exploration over repeated local refinement of the same initial candidate.

PRM-guided suffix selection. Qwen2.5-Math-PRM-7B scores a completed response after deterministic segmentation: explicit "Step" markers take precedence, followed by blank-line blocks and then nonempty lines. We select the first segment in textual order with score below 0.5, then collect tokens from that segment and its successors up to the $\lfloor \alpha L \rfloor$ mask budget. If the suffix is too short, the remaining positions are filled by globally lowest-confidence tokens. If no segment falls below the threshold, segmentation fails, or PRM scoring/alignment fails, selection falls back to global low confidence. Unselected tokens, including later reasoning and answer tokens, remain visible during regeneration. The combined PRM+answer variant additionally remasks the detected answer span, so its mask count can exceed the nominal budget. "First" specifies textual order; the low score is a heuristic signal for choosing an edit location.

On the role of PRM localization. Qwen2.5-Math-PRM-7B provides a heuristic remasking location. We audited 100 LLaDA-8B MATH500 traces, sampling at most one repair round per problem. GPT-5.5 independently identified the first substantively invalid segment from the problem, reference answer, and segmented reasoning, with PRM scores and selections hidden. Error-presence agreement was 91.0% (problem-bootstrap 95% CI: $[85.0, 96.0]$), while exact first-segment agreement was 76.0% ($[67.0, 84.0]$). Thus, detecting whether a trace contains an error is easier than locating its first erroneous segment. These estimates measure agreement with a model judge; the guided-repair results should be interpreted with this localization uncertainty.

Task-verifier-guided repair. On Countdown, we also evaluate LLaDA-8B with a deterministic arithmetic verifier and a text-based rule for selecting repair positions (n=200, seed 42, N=16, t=0.4, α=0.2). The verifier checks that the final expression uses the supplied numbers exactly once and evaluates to the target within a 10^{-5} numerical tolerance. Verified-correct responses are frozen. For an incorrect response, the rule finds the first explicitly false arithmetic equality, or falls back to the invalid final expression. Repair prioritizes the detected final-answer span, then fills the fixed mask budget from the targeted error and its textual suffix; any remaining mask positions are filled in ascending order of token confidence.

On the 200 initial responses, verifier correctness agrees with the stored evaluation labels in all 200 cases. Among the 157 incorrect responses, the rule identifies 28 false local equalities and 124 invalid final expressions; five use the parsing fallback. The resulting $152/157{=}96.8\%$ targeting rate describes the availability of a rule-selected span, and does not establish whether that span contains the first underlying reasoning error.

Table 13: Task-verifier-guided repair on Countdown with LLaDA-8B (seed 42, 200 problems). All strategies provide 16 candidate states. Agreement is recomputed from parsed answers, pooling parse failures as one symbol. Costs count generation-model forwards and exclude verifier computation; freezing correct responses reduces the repair cost.

Strategy	pass@16 (%)	Unique/16	$\hat{\rho}$	Forwards
Confidence repair	29.5	1.33	0.928	454.0
Task-verifier repair	30.5	1.60	0.878	350.2
Independent sampling	81.5	11.60	0.071	1024.0

Task-verifier repair reaches 30.5% pass@16 and 1.60 unique answers (Table 13), compared with 29.5% and 1.33 for confidence repair. The one-point coverage difference is descriptive in this single-seed comparison. Candidate counts are matched while compute differs; task-defined correctness feedback and heuristic error targeting leave a large gap to independent sampling under the evaluated sequential masking and regeneration procedure.

C.3 Random Remasking

Under uniform random remasking, positions are drawn uniformly at each round, so answer tokens are remasked with probability $\approx\alpha$ rather than being systematically protected. Table 14 shows that

accuracy remains near greedy throughout, ruling out confidence-token avoidance as the sole source of persistent dependence between successive candidates in the repair chain.

Table 14: Per-round accuracy (%) under uniform random remasking (α=0.2, t=0.4, LLaDA-8B). BoN-16 is an independent-sampling reference at higher compute.

Dataset	Greedy	r=1	r=3	r=7	r=15	BoN-16
Countdown	25.4	24.2	27.3	22.3	21.9	**83.2**
MATH500	33.4	34.4	32.2	33.6	33.4	**40.1**
HumanEval	26.8	26.2	29.3	29.9	30.5	**46.7**

C.4 Alpha Sweep

Table 15 reports $\hat{\rho}$, unique answers per 16 draws, and oracle pass@16 accuracy as a function of remask ratio α for sequential repair on Countdown (LLaDA-8B, t=0.4, N=16). Independent sampling at t=0.4 is included as a reference row. Because repair budget varies with α, the pass@16 numbers here should be read as trend diagnostics rather than matched-budget efficiency comparisons.

Table 15: Alpha sweep: sequential repair on Countdown; independent sampling included as a reference. Repair budget varies with α; independent sampling uses 1024 passes. pass@16 is oracle, included as a diagnostic trend only.

Strategy	α	pass@16	$\hat{\rho}$	Unique/16
Repair	0.05	21.5	0.998	1.01
Repair	0.10	24.0	0.960	1.16
Repair	0.20	29.0	0.930	1.31
Repair	0.40	29.0	0.910	1.44
Repair	0.80	29.5	0.769	2.21
Repair	1.00	64.5	0.579	3.77
Repair (fairsteps)	1.00	41.5	0.666	3.14
Repair (fresh-seed)	1.00	46.0	0.652	3.21
Independent sampling	–	82.8	0.076	11.46

To isolate the effect of α under comparable compute, Table 16 reports a second sweep in which repair rounds are adjusted so that all rows use approximately the same total forward-pass budget. The α=0.40 row uses full denoising steps (T) rather than half steps ($T/2$); this is an empirical adjustment based on pilot runs showing that half-step denoising becomes unstable when remasking 40% of tokens, whereas lower α values remain stable under the reduced budget. For $\alpha \leq 0.8$, increasing the remask ratio lowers $\hat{\rho}$ but yields only small gains in unique answers. The apparent jump at α=1.0 requires additional controls: the fairsteps variant gives each repair round a full T denoising steps, matching the per-candidate denoising budget of independent sampling, while the fresh-seed variant resets the RNG state at each repair round to remove cross-round stochastic accumulation. Both of these controls remain below independent sampling in pass@16 and unique-answer coverage. However, both still regenerate the response by updating all positions jointly, which differs from independent sampling's blockwise procedure. Appendix C.6 isolates this factor and shows that once the update schedule is also matched, a genuine full-remask restart recovers independent-sampling coverage; the residual in the fairsteps/fresh-seed rows is therefore attributable to the update schedule rather than to persistent dependence after a true restart.

Because the full-remask results are scattered across three tables that use different regeneration settings, we state the mapping explicitly. All of them are joint-update restarts at α=1 on Countdown with LLaDA-8B, and they differ only in per-round denoising steps, RNG handling, and how many runs are averaged: (i) Table 15, α=1.0 row, pass@16 64.5—the original joint update with $T/2$ steps per round and no RNG reset (559 forwards); (ii) Table 15, the fairsteps (41.5) and fresh-seed (46.0) rows—the same joint update with a full T steps per round (1,039 forwards), the fresh-seed row additionally resetting the RNG each round; (iii) Table 18, the Joint column, pass@16 46.8—the fresh-seed full-T joint update of (ii), averaged over two complete runs (46.0 and 47.5), so the 46.0 in (ii) is one of the two runs that compose 46.8 rather than independent corroboration of it. The spread from 41.5 to

64.5 among joint restarts is itself part of the finding: with no retained state, coverage is set by the regeneration schedule rather than by dependence, and no joint variant approaches the 80.0 of the matched blockwise restart using the independent sampler's regeneration schedule.

Table 16: Approximately matched-budget alpha sweep on Countdown (LLaDA-8B, t=0.4). Repair rounds are chosen so that all settings use roughly the same total forward-pass budget per sample.

α	Rounds	Repair steps	Avg. passes	Accuracy
0.05	64	$T/2$	512	19.5
0.10	34	$T/2$	506	24.6
0.20	17	$T/2$	506	21.9
0.40	8	T	480	21.1
0.80	6	T	454	17.6
1.00	6	T	454	5.5

C.5 Temperature Sensitivity

Table 17 extends the controlled comparison to t from 0.4 to 1.0 for both sequential repair and independent sampling at the same temperature settings.

Table 17: Temperature sweep: sequential repair and independent sampling on Countdown, MATH500, and HumanEval (LLaDA-8B, N=16). Independent sampling is a reference; repair-16 uses ≈45% of the compute of independent sampling-16 (454 forward passes on Countdown, 1024 for independent sampling under the same Countdown configuration).

Dataset	Strategy	t	$\hat{\rho}$	Unique/16	pass@1	pass@16
Countdown	Repair	0.4	0.930	1.31	24.5%	29.0%
		0.7	0.916	1.43	17.0%	20.0%
		1.0	0.878	1.65	19.0%	25.5%
	Independent	0.4	0.076	11.46	25.2%	82.8%
		0.7	0.050	12.53	19.5%	81.0%
		1.0	0.041	13.10	17.0%	84.0%
MATH500	Repair	0.4	0.941	1.34	35.0%	37.0%
		0.7	0.934	1.36	33.5%	34.5%
		1.0	0.925	1.59	33.0%	35.0%
	Independent	0.4	0.348	6.77	34.0%	63.8%
		0.7	0.333	7.16	36.5%	65.5%
		1.0	0.306	7.51	32.0%	69.5%
HumanEval	Repair	0.4	0.800	2.29	27.2%	35.6%
		0.7	0.774	2.48	26.8%	34.8%
		1.0	0.645	3.74	26.8%	36.6%
	Independent	0.4	0.296	8.78	28.7%	53.1%
		0.7	0.169	11.57	27.4%	55.5%
		1.0	0.092	13.34	23.2%	57.3%

C.6 Blockwise Restart at Full Remasking

At α=1, repair retains no tokens from the previous completion and is therefore an independent restart. What such a restart recovers depends on how the response is regenerated. To separate the update schedule from any residual dependence, we compare two full-remask restarts against independent sampling on LLaDA-8B. Both restarts start from the prompt plus a fully masked response, use fresh randomness, and match independent sampling's nominal denoising allocation. The *Joint* variant lets all masked response positions compete at every step (and adds one forward pass per round to recompute confidence, giving 1,039 vs. 1,024 measured forwards on Countdown and 2,063 vs. 2,048 on MATH500/HumanEval). The *Blockwise* variant instead reuses independent sampling's exact

blockwise procedure, completing one 32-token block before opening the next and revealing the same number of tokens per denoising step as in the independent-sampling reference.

Table 18: Full-remask restart at α=1 on LLaDA-8B (pass@16, %). Values average two complete runs. Joint updates all response positions jointly; Blockwise uses independent sampling's blockwise restart kernel. Matching the blockwise generation procedure recovers coverage close to independent sampling across each of the three evaluated tasks.

Task	Independent	Joint	Blockwise
Countdown	83.0	46.8	**80.0**
MATH500	63.0	45.5	**64.8**
HumanEval	52.1	43.3	**53.4**

Table 18 shows that with no retained state, the blockwise restart is close to independent sampling (Countdown 80.0 vs. 83.0, with 11.65 vs. 11.56 unique answers and 0.071 vs. 0.073 agreement; HumanEval 8.73 vs. 8.82 unique, 0.297 vs. 0.296 agreement), whereas the joint update is far more concentrated (3.21 unique answers and 0.649 agreement on Countdown, 5.93 and 0.498 on HumanEval). The full-remask comparison identifies the generation schedule as the source of the large restart deficit. At α<1, the evaluated procedures jointly change retained state, chain construction, mask ratio, denoising allocation, and regeneration schedule. Their separate contributions to the partial-repair coverage gap remain unresolved. For Countdown, MATH500, and HumanEval, Blockwise minus Joint is $33.2/19.3/10.1$ points; Joint minus the seed-42 partial repair is $17.3/4.5/9.2$ points. These differences compare point estimates from distinct protocols and cannot be added as a causal decomposition. They show why generation procedure must be considered when interpreting the high agreement and limited coverage of local repair.

C.7 Semantic Canonicalization of Answers

Because $\hat{\rho}$ and unique-answer counts operate on extracted answers, surface-form variation could in principle inflate the diversity gap. We therefore recompute the diagnostics after conservative, task-specific semantic canonicalization on LLaDA-8B: Countdown expressions are evaluated with exact rational arithmetic and grouped by value; MATH500 answers are grouped with the same mathematical-equivalence checker used by the correctness evaluator; and HumanEval programs are grouped by their return/exception signatures for calls whose arguments can be parsed from the benchmark tests. Each parsed call is executed once. These signatures establish agreement on those calls only; they do not prove full program equivalence. Cases without a usable execution signature are grouped using exact identity of their extracted program text.

Table 19: Semantic canonicalization on LLaDA-8B (N=16, t=0.4). "Classes" is the mean number of equivalence classes among 16 candidates (Independent / Repair); "Gap retained" is the Independent−Repair difference in classes as a fraction of the corresponding difference in distinct texts; $\hat{\rho}$ is the semantic pairwise agreement (Independent / Repair).

Dataset	Classes (Ind./Rep.)	Gap retained	$\hat{\rho}$ (Ind./Rep.)
Countdown	9.69 / 1.31	81.6%	0.128 / 0.931
MATH500	6.24 / 1.36	94.4%	0.381 / 0.943
HumanEval	6.11 / 1.61	67.5%	0.446 / 0.900

Table 19 shows that canonicalization lowers absolute diversity, most strongly for independent sampling, but the Independent−Repair unique-answer gap is largely preserved (81.6%, 94.4%, and 67.5% of the exact-match gap on Countdown, MATH500, and HumanEval). Semantic pairwise agreement remains far higher for repair than for independent sampling on every task. Oracle pass@16 is unchanged because the task evaluators already score correctness by execution or mathematical equivalence. Surface variation therefore explains part of the absolute unique-answer difference but not the answer-coverage collapse.

C.8 Attention-Blocking Intervention

We test how access to retained context changes answer concentration with a two-stage Countdown intervention (N=16, 200 problems per run) on three LLaDA-family models. Forward counts match across the three repair arms within each model; independent sampling uses its standard full-trajectory budget. Stage 1 regenerates low-confidence tokens outside the detected answer span while varying which retained positions the regeneration may attend to; Stage 2 identically remasks and regenerates the answer span in all arms. We compare no blocking, blocking the retained positions that contain the answer, and blocking an equal number of randomly chosen retained positions outside the answer span. Span detection succeeds on 96.0%–98.5% of cases; undetected cases remain in their arm and receive no span-specific blocking or regeneration. Stage 2 uses $T/2$ steps for LLaDA-8B and T for LLaDA-1.5 and LLaDA-MoE. LLaDA-8B values average two complete runs (seeds 42 and 43); the other models each have one complete seed-42 run. Thus cross-model effect sizes reflect different stage-2 allocations as well as model differences.

Table 20: Attention-blocking intervention on Countdown (N=16). Blocking retained positions (answer or random) reduces agreement and increases unique answers across all three models, but neither condition approaches independent sampling.

Model	Condition	pass@16 (%)	Unique/16	$\hat{\rho}$
LLaDA-8B	No blocking	31.0	1.80	0.826
	Block answer	32.0	1.99	0.779
	Block random	34.8	2.55	0.691
	Independent sampling	**83.0**	**11.56**	**0.073**
LLaDA-1.5	No blocking	25.0	1.74	0.851
	Block answer	27.5	1.99	0.801
	Block random	31.5	2.65	0.710
	Independent sampling	**84.0**	**12.24**	**0.056**
LLaDA-MoE	No blocking	42.0	1.53	0.869
	Block answer	44.0	1.67	0.837
	Block random	44.0	1.99	0.770
	Independent sampling	**89.0**	**8.80**	**0.174**

Table 20 shows that both blocking conditions reduce agreement and increase unique answers in all three models, consistent with attention to the retained completion contributing to dependence. The effects are modest and neither condition approaches independent sampling, and blocking the answer span is not systematically stronger than blocking equally many random retained positions. Random blocking gives higher unique-answer counts in all three models; its pass@16 exceeds answer blocking by 2.8, 4.0, and 0.0 points, respectively. With only one run for two models, repeated-run uncertainty is unavailable for those models. We treat the small coverage differences as descriptive; the intervention shows sensitivity to retained-context access without identifying the answer span as a privileged source of the coverage gap.

C.9 Repair Under the Main-Table Selector

Table 1 scores repair with the same task-specific selector used for BoN, applied to the $r+1$ successive states of the repair chain. This removes an asymmetry that would otherwise favour BoN, which inspects N candidates before committing. Table 21 reports the three quantities this choice sits between: the unselected final round, which is what a deployed repair loop would actually return; the selected value from Table 1; and the oracle over the same chain, which upper-bounds any selector. Per-round confidences are not stored in the chains, so the selector's confidence fallback is replaced by the final round, i.e. when the selector has no signal the cell reverts to the unselected value. On Countdown and HumanEval the selector is the correctness criterion itself, so selected and oracle coincide; on MATH500 self-consistency leaves between 0.6 and 3.9 oracle points unused.

Selection over the chain moves repair by at most 1.3 points in every LLaDA cell, leaving the gap to BoN-4 essentially unchanged. The one substantial shift is Dream on Countdown, where the chain visits correct expressions it does not retain: the final round gives 32.4 while selection recovers 50.0.

Even there the oracle over 39 repair states remains 23.7 points below BoN-16. Repair's deficit is therefore a property of which answers the chain produces, not of the rule used to read one off.

Table 21: Repair accuracy (%) under three read-out rules. "Final" is the unselected last round; "Selected" applies the main-table selector over rounds $0..r$ and is what Table 1 reports; "Oracle" is the coverage upper bound over the same 39 states. BoN-4 is the cheapest independent-sampling column ($4\times$ versus $16.4\times$ greedy compute). Bold marks the larger of Selected $r{=}38$ and BoN-4.

			Final	Selected			Oracle	BoN-4
Model	Dataset	Greedy	$r{=}38$	$r{=}8$	$r{=}18$	$r{=}38$	$r{=}38$	($4\times$)
LLaDA-8B	Countdown	25.4	26.4	27.2	27.7	27.7	27.7	**57.9**
	MATH500	33.4	33.3	33.3	33.3	33.5	34.8	**36.9**
	HumanEval	26.8	33.9	33.5	33.7	34.2	34.2	**36.0**
LLaDA-1.5	Countdown	19.1	23.2	24.1	24.3	24.3	24.3	**55.2**
	MATH500	33.2	34.8	34.9	34.9	34.9	35.5	**37.1**
	HumanEval	27.4	31.9	32.3	32.5	32.5	32.5	**36.8**
LLaDA-MoE	Countdown	43.4	40.1	40.6	40.9	41.3	41.3	**69.5**
	MATH500	35.4	34.7	34.8	34.7	34.8	35.5	**40.5**
	HumanEval	42.1	45.3	46.1	46.1	46.1	46.1	**51.2**
Dream-7B	Countdown	19.5	32.4	44.4	47.0	**50.0**	50.0	45.8
	MATH500	20.0	18.1	18.2	18.3	18.3	22.2	**18.9**
	HumanEval	34.8	19.5	22.0	22.4	23.0	23.0	**37.4**

C.10 Selection Rules versus Coverage

To separate coverage from selection, we apply three offline selection rules to the same LLaDA-8B candidate sets: exact symbolic verification for Countdown, the most frequent math-equivalence class for MATH500, and the most frequent return/exception signature for HumanEval. This is not the paper's main selected-accuracy evaluation; only Countdown shares its main selector.

Table 22: Coverage versus selection on LLaDA-8B ($N{=}16$). "After rule" applies the offline selector above; oracle pass@16 is the coverage upper bound. Selection recovers all coverage on Countdown but leaves oracle points unused on MATH500 and HumanEval.

Dataset	Strategy	First cand. (%)	After rule (%)	Oracle pass@16 (%)
Countdown	Independent	25.2	**82.8**	**82.8**
	Repair	24.5	29.0	29.0
MATH500	Independent	34.0	43.2	**63.8**
	Repair	35.0	35.3	37.0
HumanEval	Independent	28.7	34.2	**53.1**
	Repair	27.2	32.2	35.6

Table 22 confirms that coverage and selection are distinct bottlenecks: the Countdown verifier converts all available coverage into selected accuracy, whereas the MATH500 and HumanEval rules leave roughly 20 and 19 oracle points unused. In all cases repair trails independent sampling under both the offline rule and oracle coverage.

D Extended Strategy and Model Comparisons

D.1 Strategy Space of Controlled Runs

Figure 2 replots the controlled comparison of Table 2 in the plane spanned by cross-candidate dependence ($\hat{\rho}$) and per-candidate quality (pass@1), with bubble area encoding pass@16, and adds the corresponding coverage and diversity coordinates. Sequential repair lies far to the right: its initial candidates have competitive accuracy, but successive candidates remain strongly dependent,

and the small bubbles indicate limited pass@16 growth alongside this concentration. ODD reduces dependence by enforcing diversity but moves downward, incurring a per-candidate quality cost. Independent sampling occupies the upper-left region, combining high-quality candidates with low dependence. The geometry makes explicit that the bottleneck is the conjunction of quality and independence rather than either candidate quality or candidate dependence considered in isolation.

D.2 Cross-Model Replication of Trajectory Dependence

Table 23 extends the controlled sampling-vs-repair comparison of Table 2 to LLaDA-1.5 and LLaDA-MoE. All runs use N=16, t=0.4, α=0.2. Across model families, sequential repair consistently shows higher dependence and lower pass@16 than independent sampling.

Table 23: Cross-model controlled comparison: independent sampling and sequential repair. Gap measures the coverage difference, computed as ind pass@16 − rep pass@16.

Model	Task	ind $\hat{\rho}$	rep $\hat{\rho}$	ind pass@16	rep pass@16	Gap (pp)
LLaDA-8B	Countdown	0.076	0.930	82.8%	29.0%	53.8
	MATH500	0.348	0.941	63.8%	37.0%	26.8
	HumanEval	0.296	0.800	53.1%	35.6%	17.5
LLaDA-1.5	Countdown	0.057	0.951	84.3%	26.6%	57.7
	MATH500	0.370	0.959	63.5%	35.3%	28.2
	HumanEval	0.280	0.796	51.2%	38.2%	13.0
LLaDA-MoE	Countdown	0.163	0.973	89.7%	40.0%	49.7
	MATH500	0.337	0.869	69.5%	34.5%	35.0
	HumanEval	0.370	0.795	53.7%	43.9%	9.8
Dream-7B	Countdown	0.054	0.656	76.0%	50.0%	26.0
	MATH500	0.164	0.857	55.5%	20.0%	35.5
	HumanEval	0.016	0.558	56.7%	25.0%	31.7

D.3 Compute-Matched Comparison

Table 24 is the matched-budget counterpart to the candidate-count-matched controlled comparison in Table 2. Rather than fixing N=16 and probing dependence structure, it approximately matches total forward-pass budget across independent sampling, ODD, and repair, and should be read as an efficiency check. Independent sampling uses N=8 candidates; ODD uses N=8 jointly generated sequences; repair uses N=18 candidates. t=0.4. ODD is defined for the LLaDA denoising loop, so this comparison covers the three LLaDA-family models.

D.4 ODD Diversity-Repulsion Sweep

Table 25 further characterises ODD's behavior with a sweep over repulsion strength α_{ODD}. Increasing ODD repulsion lowers agreement and increases unique-answer coverage in this sweep. Per-candidate accuracy remains around 6%, below independent sampling; even at α_{ODD}=5, both quality and unique-answer coverage remain lower.

D.5 Additional Benchmark Snapshot

Table 26 reports results on additional benchmarks.

The three multiple-choice benchmarks show a much larger repair deficit than MBPP or GSM8K, and this deficit is an answer-formatting effect rather than a coverage effect. On these tasks the scorer requires an explicit option label, and repair frequently destroys it: the responses degenerate into end-of-text padding or into a declarative sentence that never names a choice. Table 27 separates the two components. The fraction of responses from which no option can be extracted roughly triples under repair, while accuracy computed over the responses that *are* parseable is essentially unchanged from greedy. Independent sampling is affected in the opposite direction, because selecting among 16 candidates almost always finds one that is well formed (1–3% unparseable at N=16). These

Table 24: Compute-matched controlled comparison. Independent sampling and ODD use $N{=}8$ candidates ($\approx$512 passes on Countdown, $\approx$1024 on MATH500/HumanEval); repair uses $N{=}18$ ($\approx$506 to 1012 measured passes). **Bold**: best accuracy and best unique-answer count per model and task group across the three evaluated generation strategies.

Model	Dataset	Strategy	Passes	Accuracy	$\hat{\rho}$	Unique
LLaDA-8B	Countdown	Indep. sampling	512	**71.0%**	0.071	**5.66**
		ODD	512	37.5%	0.427	3.67
		Repair	506	29.5%	0.929	1.34
	MATH500	Indep. sampling	1024	**56.5%**	0.357	3.26
		ODD	1024	44.0%	0.307	**4.31**
		Repair	1012	41.0%	0.943	1.36
	HumanEval	Indep. sampling	1024	**49.4%**	0.290	4.34
		ODD	1024	40.9%	0.360	**4.81**
		Repair	1012	34.1%	0.806	2.36
LLaDA-1.5	Countdown	Indep. sampling	512	73.4%	0.058	**5.93**
		ODD	512	**75.8%**	0.149	5.65
		Repair	506	27.7%	0.949	1.26
	MATH500	Indep. sampling	1024	**59.5%**	0.377	3.05
		ODD	1024	45.5%	0.387	**3.79**
		Repair	1012	35.0%	0.962	1.29
	HumanEval	Indep. sampling	1024	**47.0%**	0.283	4.31
		ODD	1024	46.3%	0.302	**5.25**
		Repair	1012	39.0%	0.785	2.43
LLaDA-MoE	Countdown	Indep. sampling	512	**83.0%**	0.173	4.25
		ODD	512	71.5%	0.129	**5.78**
		Repair	506	41.5%	0.974	1.11
	MATH500	Indep. sampling	1024	**60.5%**	0.328	**4.48**
		ODD	1024	52.5%	0.288	4.43
		Repair	1012	34.5%	0.869	2.09
	HumanEval	Indep. sampling	1024	50.6%	0.370	3.48
		ODD	1024	**62.2%**	0.315	**4.84**
		Repair	1012	43.9%	0.795	2.42

Table 25: ODD α-sweep on Countdown (LLaDA-8B, $N{=}16$, $t{=}0.4$). Rand. draw: mean per-candidate accuracy. pass@16: oracle coverage over the 16 candidates, which on Countdown coincides with the symbolic-verifier selection used in Table 1; it is the same quantity reported in the pass@16 column of Table 2, and the Restart row is that table's independent-sampling reference.

α_{ODD}	Rand. draw	pass@16	$\hat{\rho}$	Unique
1	6.0%	36.3%	0.717	3.0
2	6.0%	43.0%	0.697	3.4
3	5.9%	44.1%	0.684	3.6
5	6.5%	47.0%	0.480	5.3
Restart	25.2%	**82.8%**	0.08	11.5

three rows therefore measure format stability, not answer coverage, and we exclude them from the diagnostic analysis in the main text.

E Open-Ended Evaluation

The diagnostics in the main text rely on a well-defined extracted answer. To probe whether the same conclusions hold on open-ended generation, we evaluate two additional settings: all 541 prompts of **IFEval** (Zhou et al., 2023) with the official deterministic instruction-following verifiers, and the

Table 26: Additional benchmark results for LLaDA-8B. Repair uses t=0.4, α=0.2; BoN-N uses the task-specific selection protocol described in Appendix A. Unlike Table 1, the repair columns here report the unselected final round, because per-round chains were not retained for these benchmarks; Table 21 shows that chain-level selection moves repair by at most 1.3 points on the three main benchmarks evaluated with the same LLaDA-8B checkpoint.

Dataset	Greedy	Repair			BoN		
	(1×)	r=8	r=18	r=38	N=4	N=8	N=16
MBPP	32.3	37.4	38.1	37.0	40.5	42.0	45.5
GSM8K	74.1	75.3	76.7	75.7	82.0	82.6	83.4
ARC-Challenge	72.8	50.9	50.2	51.2	82.7	84.0	85.1
OpenBookQA	68.4	41.2	41.2	44.2	84.8	87.2	87.6
CommonsenseQA	64.9	43.7	45.8	45.0	75.1	78.8	79.5

Table 27: Decomposition of the multiple-choice repair deficit (LLaDA-8B). "Unparseable" is the fraction of responses containing no extractable option label; "Acc. if parsed" is accuracy restricted to the remainder. The accuracy drop in Table 26 is accounted for almost entirely by the first column pair.

	Unparseable (%)			Acc. if parsed (%)		
Dataset	Greedy	r=8	r=38	Greedy	r=8	r=38
ARC-Challenge	12.9	36.5	38.1	83.5	80.1	82.8
OpenBookQA	16.0	49.0	46.2	81.4	80.8	82.2
CommonsenseQA	13.8	37.8	38.2	75.3	70.4	72.7

250 creative-writing prompts of **Arena-Hard v2.0** (Li et al., 2024) judged pairwise by an external LLM judge. In both, "Initial" is the original response, "Repair" applies two confidence-based 20% remasking rounds to it, and "Independent" is a separately generated response. For LLaDA models, only positions before the first EOS token can be remasked; Dream applies repair across all positions, so cross-model magnitudes are descriptive. Mean generation compute is lower for repair than for a fresh independent response. IFEval measures instruction compliance, while Arena measures pairwise preference; both complement the answer-coverage diagnostics under their own evaluation criteria.

For Arena, GPT-5.5 judges each pair in both answer orders with method identities hidden, assessing instruction following, relevance, clarity, concision, and creativity. We use high reasoning effort and a maximum completion length of 2,048 tokens. Disagreement between the two orderings is counted as a tie. We retain only prompts with two valid verdicts, leaving 246–248 of the 250 prompts per model, and compute percentile confidence intervals by resampling prompts 100,000 times (seed 20260724). Counting every prompt without two valid verdicts as an independent-sampling win gives preference scores of 57.6%, 57.0%, 49.6%, and 46.0% over all 250 prompts for LLaDA-8B, LLaDA-1.5, LLaDA-MoE, and Dream, respectively. These point estimates give the lower bound obtained by assigning all missing comparisons to independent sampling.

Table 28: IFEval strict-metric accuracy (%) under official deterministic evaluation. Changes from the initial response are small and mixed, and neither repair nor independent sampling uniformly dominates across all evaluated models and instruction-following metrics.

Model	Metric	Initial	Repair	Independent
LLaDA-8B	Strict prompt	53.05	**53.23**	50.28
	Strict instr.	62.95	**64.15**	61.51
LLaDA-1.5	Strict prompt	58.41	**59.70**	56.93
	Strict instr.	67.63	**68.94**	67.27
LLaDA-MoE	Strict prompt	57.12	57.30	**59.15**
	Strict instr.	67.99	68.59	**69.54**
Dream-7B	Strict prompt	**46.77**	44.92	44.92
	Strict instr.	56.95	56.47	**57.31**

Table 29: Arena-Hard v2.0 creative-writing prompts (250), judged blindly in both orders. Reported score is $(\text{wins} + 0.5\,\text{ties})/n$ for Repair versus Independent, with 95% bootstrap CIs; only LLaDA-8B and LLaDA-1.5 exclude 50%, while LLaDA-MoE and Dream show no detectable preference.

Model	Both orders	Repair wins	Ties	Ind. wins	Score (95% CI)
LLaDA-8B	246	121	46	79	**58.5** [53.0, 64.0]
LLaDA-1.5	247	117	51	79	**57.7** [52.2, 63.2]
LLaDA-MoE	248	81	86	81	50.0 [45.0, 55.0]
Dream-7B	247	86	58	103	46.6 [41.1, 52.0]

Tables 28 and 29 show task-dependent effects of local refinement. LLaDA-8B and LLaDA-1.5 repair responses are preferred to fresh samples on Arena creative writing, while IFEval changes from the initial completion are small and mixed. The unequal generation costs leave the compute-matched ranking unresolved. These evaluations measure instruction compliance and response preference, complementing the verifiable-task analysis and providing evidence on local refinement beyond the three tasks evaluated through final-answer correctness.

F Case Studies

The following examples are drawn from the controlled N=18, t=0.4 runs. They are intended to illustrate the answer-state patterns measured by unique-answer count, one per task: Figure 8 for Countdown, Figure 9 for MATH500, and Figure 10 for HumanEval.

To make the repair mechanism itself more concrete, Figure 11 shows one complete repair trajectory. The figure includes the initialization stage followed by three repair rounds, and highlights which token positions are remasked before regeneration together with the resulting answer token sequence. The example illustrates that the trajectory is not a fixed-point repeat: later rounds can change the answer state even when most surrounding tokens remain stable.

G Limitations

This paper characterizes retained-state repair through answer-level coverage and dependence diagnostics. The evidence covers confidence-based repair and local variants (random, forced answer-span, and PRM-guided remasking) on LLaDA and Dream models; effect sizes vary with model, task, and remasking schedule. At α=1, the operator becomes a restart, and the blockwise control recovers coverage close to independent sampling. The task-verifier diagnostic uses Countdown-specific feedback and a heuristic edit rule; broader verifier-guided search and tree-search decoders remain outside the evaluation, and most direct diagnostics use tasks with extractable answers; open-ended results are therefore descriptive. The aggregate agreement statistic $\hat{\rho}$ should be paired with semantic clustering and task-specific equivalence checks, as illustrated in Appendix C.7. The autoregressive reference in Appendix A provides capability context rather than a cross-architecture budget comparison.

H Broader Impacts

This work is primarily an empirical and mechanistic study of test-time inference behavior in open-source dLLMs. A positive impact of the work is that it helps practitioners avoid spending inference compute on strategies that produce little effective search. In particular, the results suggest that some forms of sequential revision can consume substantial additional compute without producing proportional gains, which has implications for both research efficiency and energy use.

The work also has potential negative implications. Better understanding of inference-time scaling may help improve the efficiency or effectiveness of generative models, including systems that could be misused for deceptive or harmful generation. However, this paper does not introduce a new high-capability base model, does not release new training data, and does not propose a new generation system that materially expands misuse capabilities beyond currently available open models.

The broader scientific value of the work is to sharpen how the community reasons about extra inference compute. Rather than treating all additional test-time compute as equally useful, the

Countdown input: numbers [49, 44, 7], target 86.
Repeated final answers are collapsed into one row with their counts.

Sequential repair

0/18 correct, 3 unique final answers

Unique final answers

x13	wrong	`7 + 49`
x4	wrong	`7 * 49`
x1	wrong	`7 * 44`

Independent sampling

12/18 correct, 9 unique final answers

Unique final answers

x9	correct	`49 + 44 - 7`
x2	correct	`(49 + 44) - 7`
x1	wrong	`49 - 44 + 49 / 7 + 44`
x1	wrong	`49 + (7*(5-44))`
x1	wrong	`49 - 7 + 49`
x1	wrong	`None`
x1	wrong	`49 -44 + 7`
x1	wrong	`56 - 44`
x1	correct	`(49 + 44)  - 7`

Figure 8: Answer distributions for a representative Countdown problem (numbers $[49, 44, 7]$, target 86) over 18 candidates. Sequential repair yields no correct answer and only 3 distinct expressions, 13 of the 18 candidates repeating the same one; independent sampling produces 9 distinct answers and is correct in 12 of the 18 generated candidate responses.

paper argues that the quality of that compute depends on whether it creates effectively independent candidates or merely produces correlated local revisions. We expect this perspective to be useful in designing safer and more efficient future inference procedures.

MATH500 input: Simplify $\frac{1}{5} \cdot \frac{8}{7} \div \frac{12}{20}$. Express as a common fraction.

Sequential repair

2/18 correct, 2 unique answers

Unique answers

x16	wrong	`\frac{4}{21}`
x2	correct	`\frac{8}{21}`

Independent sampling

7/18 correct, 7 unique answers

Unique answers

x7	correct	`\frac{8}{21}`
x5	wrong	`\frac{1}{3}`
x2	wrong	`1`
x1	wrong	`\frac{4}{35}`
x1	wrong	`\frac{4}{7}`
x1	wrong	`4`
x1	wrong	`\frac{4}{3}`

Figure 9: Answer distributions for a representative MATH500 fraction-simplification problem over 18 candidates. Sequential repair produces the wrong answer $\frac{4}{21}$ in 16 of 18 candidates and reaches the correct answer $\frac{8}{21}$ only twice; independent sampling reaches the correct answer 7 times while also exploring several other answers.

Input: def is_equal_to_sum_even(n): """...can n be written as sum of 4 positive even numbers?..."""

Generated body (identical across all 18 samples — only the final return differs):

```
    if n < 8:          return False
    if n % 2 != 0:     return False
    if n == 8: return True  ···  if n == 28: return True
    return  ???   ← only this line varies across all 18 samples
```

Sequential repair

2/18 passing, 4 unique final returns

Unique final return statements

x15	fail	return True if n >= 8
x1	pass	return n == n - 8
x1	pass	return True and n >= 8
x1	fail	return False if n >= 8

Independent sampling

11/18 passing, 8 unique final returns

Unique final return statements

x8	pass	return n == n - 8
x4	pass	return False
x1	fail	return True
x1	pass	return False and n > 8
x1	fail	return quotient % 4 == 0
x1	fail	return quotient == 4
x1	fail	return n % n == 0
x1	fail	return False if n == 8

Figure 10: Final-return distributions for a representative HumanEval problem over 18 candidates. The displayed prefix is model-generated and identical across all samples; only the final return statement differs. Sequential repair concentrates on the failing `return True if n >= 8` variant, whereas independent sampling produces more passing final-return variants.

Question

If $1+12+123+1234+12345+123456+1234567+12345678$ is congruent to n modulo 9, where $0 \leq n < 9$, what is the value of n?

Initial Response

To solve the problem, we need to find the value of the sum modulo 9. We can use the property that the sum of a number is congruent to the sum of its digits modulo 9. Therefore, we can simplify the problem by finding the sum of the digits of each number in the sequence and then summing these results modulo 9.

Answer: $\boxed{6}$

Repair 1

To solve this problem, we need to find the sum of the given sequence and then take the result modulo 9. We know that a number is congruent to the sum of its digits modulo 9. Therefore, we can simplify the problem by finding the sum of the digits of each number in the sequence and then taking the result modulo 9.

Answer: $\boxed{6}$

Repair 2

To solve this problem, we need to find the sum of the given numbers and then take the result modulo 9. We know that a number is congruent to the sum of its digits modulo 9. Therefore, we can simplify the problem by finding the sum of the digits of each number in the sequence and then taking the result modulo 9.

Answer: $\boxed{5}$

Repair 3

To solve this problem, we need to find the sum of the given numbers and then take the result modulo 9. We know that a number is congruent to the sum of its digits modulo 9. Therefore, we can simplify the problem by finding the sum of the digits of each number in the sequence and then taking the result modulo 9.

Answer: $\boxed{3}$

Figure 11: A representative four-stage repair trajectory: one initialization stage followed by three repair rounds. The MATH500 prompt asks for the value of n modulo 9, with correct answer n=3. Each stage shows the tokens that are remasked before regeneration and the resulting decoded answer. The example is selected to show a nontrivial repair path, where the answer changes across rounds rather than remaining frozen at the answer chosen by the initial generation.