BREAKING DIFFUSION: PARALLEL REASONING WITH DRAFT REFINEMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Drafting an answer allows a reasoning model to reconsider each decision in the context of the others. Yet repeated diffusion denoising does not ensure that individually plausible token updates form a better answer together. We introduce **Draft Refinement Language Models (DRLMs)**, which use a single denoiser to propose parallel revisions and evaluate a candidate draft. We train the denoiser to reconstruct every token from uniformly corrupted drafts, teaching both preservation and repair. Its predictions specify revisions directly, without a diffusion reverse transition. Evaluation requires a different use of these predictions: confidence in an existing token reflects both contextual support and evidence from observing the token itself. Using the known corruption process, we correct for this self-token evidence to estimate how well each token is supported by the prompt and the remaining draft. Aggregating these estimates yields a *cavity energy* computed in one forward pass, without an additional scoring model. Each round proposes a revision, reads the completed candidate in its revised context, and retains it only if its energy is lower. Across Countdown, Sudoku-Extreme, and Maze-Hard, our method outperforms autoregressive and discrete diffusion baselines. With 6M parameters, DRLM reaches 44.4% on Countdown-5 and 93.9% on Sudoku-Extreme. The results establish complete-draft parallel refinement as an effective alternative to reverse diffusion for complex reasoning.

1 INTRODUCTION

A draft can make its own mistakes visible. Consider rereading a draft argument: its conclusion may conflict with an earlier claim, prompting us to reconsider a step that seemed plausible on its own. The draft is not merely a record of thought, but something to reason with–its parts can be inspected, compared, and revised (Flower & Hayes, 1981; Kirsh, 2010). This suggests a strategy for machine reasoning: construct a complete but provisional answer early, then use the draft itself as context for refinement. A complete draft makes the whole answer available as context, allowing earlier decisions to be reconsidered in light of later ones.

Standard left-to-right autoregressive models (Radford et al., 2019; Brown et al., 2020; Achiam et al., 2023; Touvron et al., 2023) commit earlier output tokens before a complete candidate is available (Yao et al., 2023; Dziri et al., 2023). Discrete diffusion (Austin et al., 2021; Lou et al., 2024; Sahoo et al., 2024; Nie et al., 2025; Ye et al., 2025) predicts positions in parallel, and existing non-autoregressive refinement methods (Lee et al., 2018; Savinov et al., 2021) allow complete drafts to be revised. Two issues nevertheless need to be distinguished. In standard ancestral diffusion sampling, a predefined reverse transition schedule bounds how model predictions become token changes. Removing that transition permits direct revision, but does not determine whether a revised candidate should replace the current draft. Can a single model revise a complete draft in parallel and assess how well its parts support one another in one read?

We introduce Draft Refinement Language Models (DRLMs), which use a single denoiser to propose and evaluate revisions of complete answers. We train the model to reconstruct clean sequences from drafts corrupted by uniform token replacement, supervising every position. Because incorrect tokens are unmarked, the model must learn both what to preserve and what to repair. At inference, initialized from random tokens, its clean-token predictions directly generate a revised draft: all positions are reconsidered in parallel using the same input context, without a diffusion reverse transition.

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.

Using the same denoiser for evaluation requires separating contextual support from evidence supplied by the observed token itself. For example, in a draft such as $2+3=6$, the denoiser may favor the observed 6 because, under the corruption channel, observing a token is far more likely when it is correct than when it is any particular alternative–even when its prediction from the remaining context, $2+3=_$, favors 5. Our correction removes this self-token advantage, estimating how well the context supports the draft's token rather than rewarding its mere presence. For a Bayes-optimal denoiser, we show that dividing out the known self-token likelihood and renormalizing recovers the token's cavity conditional (Mezard et al., 1987): its distribution given the prompt and all other noisy draft tokens. Applying this correction to the trained model yields estimates for every position from a single full-draft read. Summing the negative log-probabilities assigned to the draft's tokens defines its *cavity energy*, a learned, noise-smoothed pseudo-likelihood score (Besag, 1975).

DRLM uses this score in a *propose–read–retain* generative loop. Each round partially re-noises a copy of the retained draft to prevent a draft converging to a local optimum while preserving some of its context, then constructs a candidate through direct parallel token predictions. A fresh forward pass through the same model then scores the candidate at a fixed read level, evaluating it in its revised context rather than reusing the predictions that produced it. The candidate replaces the retained draft only if its cavity energy is lower. All proposal and scoring passes count toward a fixed inference budget, after which the decoder returns the lowest-energy draft. Our ablations show the benefits of retaining the best-scoring draft, refining it rather than restarting independently, and correcting the evaluation score for self-token evidence.

Our contributions are as follows:

- **One-pass contextual draft evaluation.** We derive a cavity decomposition of the full clean-token posterior and use it to correct denoiser predictions for self-token evidence. The resulting contextual draft score requires only one forward pass, without architectural changes or separate leave-one-out evaluations.
- **Energy-controlled parallel refinement.** We introduce a *propose–read–retain* loop that uses a single denoiser to refine a complete draft from random initialization. Each round constructs a candidate through parallel token updates to a partially re-noised copy of the retained draft. A fresh forward pass scores the candidate, which replaces the retained draft only if its cavity energy is lower.
- **Improved reasoning performance.** With 6M parameters, DRLM outperforms the evaluated autoregressive and discrete diffusion baselines on Countdown, Sudoku-Extreme, and Maze-Hard. It reaches 87.6% on Countdown-4, 44.4% on Countdown-5, 93.9% on Sudoku-Extreme, and 73.0% on Maze-Hard, exceeding the strongest diffusion baseline on each task by $+14.3$, $+34.1$, $+28.4$, and $+33.7$ percentage points.

2 Background

We consider conditional generation of a sequence $x = (x^1, \ldots, x^L)$ over a vocabulary of size N given a condition c. Interpolating discrete diffusion corrupts each position independently toward a prior π (Austin et al., 2021; Sahoo et al., 2024; Shi et al., 2024; Von Rütte et al., 2025). Using one-hot token representations and suppressing the position index, the forward channel is

$$q(\mathbf{z}_t \mid x) = \mathrm{Cat}\big(\mathbf{z}_t;\ \alpha_t x + (1-\alpha_t)\pi\big), \tag{1}$$

where the keep probability α_t decreases from $\alpha_0 = 1$ to $\alpha_1 = 0$. With $\pi = \mathbf{m}$, a special mask symbol, corruption either preserves a token or masks it. With $\pi = \mathbf{u} = \mathbf{1}/N$, corruption instead replaces tokens uniformly over the ordinary vocabulary. Uniformly corrupted sequences are therefore complete drafts: every position contains an ordinary token, and replacements are unmarked and may coincide with the original tokens (Hoogeboom et al., 2021; Schiff et al., 2024; Sahoo et al., 2025).

A denoiser $x_\theta^\ell(c, \mathbf{z}_t, t)$ uses the condition and the full noisy sequence to predict clean-token probabilities at each position. In the MDLM and UDLM plug-in parameterizations, these predictions are substituted into the categorical parameter of the exact reverse posterior $q(\mathbf{z}_s^\ell \mid \mathbf{z}_t^\ell, x^\ell)$ for $s < t$, yielding a factorized learned transition $p_\theta(\mathbf{z}_s \mid c, \mathbf{z}_t)$ (Sahoo et al., 2024; Schiff et al., 2024). Standard diffusion training minimizes a negative evidence lower bound (NELBO) (Sohl-Dickstein et al., 2015; Ho et al., 2020; Kingma et al., 2021; Campbell et al., 2022), whose diffusion term compares

exact and learned reverse transitions. The denoiser is thus trained through the transitions induced by its parameterization.

Ancestral Decoding and Token Revision. Given a schedule $1 = t_T > \cdots > t_0 = 0$, standard ancestral decoding initializes each position independently from π and samples successive states from the learned reverse transitions. In standard masked diffusion, the schedule determines the probability of revealing a masked position, while the model determines its revealed value. Once revealed, a token is copied unchanged by subsequent reverse steps, even as its surrounding context changes (Sahoo et al., 2024). Remasking samplers relax this restriction by allowing visible tokens to return to the mask state (Wang et al., 2025b; Zheng et al., 2023; Ye et al., 2024).

Uniform-noise diffusion permits repeated token revisions (Schiff et al., 2024), but the reverse kernel still constrains how predictions translate into changes. Under its plug-in parameterization, the probability of changing a position is bounded by a schedule-dependent ceiling, strictly below one for interior steps. Thus, in both standard ancestral decoders, clean-token predictions are mediated by a diffusion transition rather than specifying the next sequence directly. Appendix C gives the precise parameterizations and derives the masked revelation probability and uniform revision bound.

From Draft Revision to Draft Evaluation. These constraints motivate refining complete drafts directly, letting clean-token predictions determine the next draft without a reverse diffusion transition. Direct revision, however, does not decide whether a revision should be kept: proposing a candidate and evaluating it are separate decisions. Evaluation needs care, because a full-posterior prediction treats each position's own token as evidence alongside the condition and the rest of the draft, so agreement with the current token does not by itself show contextual support.

3 Draft Refinement Language Modeling

We introduce draft refinement language models (DRLMs), in which one denoiser both proposes and evaluates revisions of complete sequences. Trained with all-position cross-entropy on uniformly corrupted drafts, it predicts full clean-token posteriors for direct parallel revision. Dividing out the known self-token likelihood turns the same predictions into one-pass cavity estimates of each token's contextual support. Each round re-noises the current draft, builds a complete candidate, and reads it in its revised context; the candidate replaces the incumbent only if its cavity energy is lower. After a fixed round budget, the decoder returns the lowest-energy candidate.

3.1 Learning Complete-Draft Prediction

We train a single denoiser to reconstruct clean sequences from complete, uniformly corrupted drafts (Lee et al., 2018; Savinov et al., 2021), then use its predictions to propose parallel revisions. Given a condition c and a clean sequence $x \in [N]^L$, we sample a keep probability $\alpha \sim U(0,1)$ and independently corrupt each position through the uniform replacement channel:

$$q_\alpha(s \mid x) = \prod_{i=1}^{L} k_\alpha(s_i, x_i),$$
$$k_\alpha(u, v) = \alpha \mathbb{1}[u = v] + \frac{1-\alpha}{N}. \tag{2}$$

Each position either preserves its clean token or draws uniformly from the vocabulary, possibly drawing the same token. The resulting draft is complete: every position contains an ordinary token, and incorrect tokens are unmarked. This channel defines the denoiser's training inputs, not a diffusion trajectory that generation must reverse.

The predictor $p_\theta(x_i \mid c, s, \alpha)$ uses the condition, the entire draft, and its corruption level to predict a clean-token distribution at each position. We train it with all-position cross-entropy:

$$\mathcal{L}(\theta) = \mathbb{E}_{(c,x)} \mathbb{E}_{\alpha \sim U(0,1)} \mathbb{E}_{s \sim q_\alpha(\cdot \mid x)} \left[-\frac{1}{L} \sum_{i=1}^{L} \log p_\theta(x_i \mid c, s, \alpha) \right]. \tag{3}$$

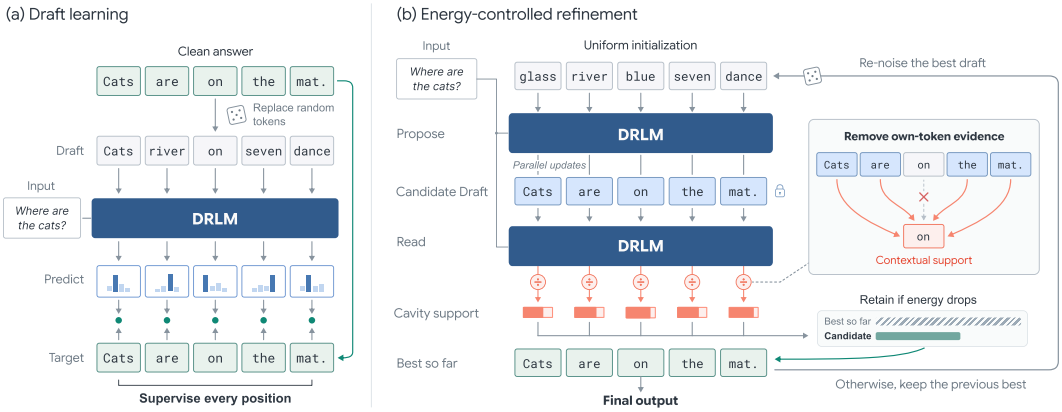


Figure 1: **Overview of Draft Refinement Language Models (DRLMs).** (a) All-position supervision on uniformly corrupted drafts teaches preservation and repair. (b) Starting from uniform initialization, the decoder proposes parallel revisions and reads each completed candidate using the same denoiser, at a fixed read level and without changing its tokens. Dividing out the self-token likelihood yields cavity estimates for contextual scoring. A candidate replaces the best-so-far draft only when its cavity energy decreases; otherwise, the incumbent draft is retained. Subsequent rounds re-noise the retained draft, and decoding returns the best-energy draft after a fixed round budget.

Every position contributes to the loss, including those whose observed token already matches the clean token (Devlin et al., 2019; Clark et al., 2020). The objective therefore supervises both preservation and repair.

With sufficient model capacity, the population minimizer recovers the clean-token posterior marginals $\Pr(x_i \mid c, s, \alpha)$ under the joint distribution of clean data and the corruption channel (Bengio et al., 2013). These are *full* posterior marginals: the position's own observed token s_i contributes evidence alongside the condition and the remaining draft.

We use these predictions directly to construct a revised draft. For a fixed condition c, define the parallel prediction map

$$[F_\alpha(s)]_i = \arg\max_{v \in [N]} p_\theta(x_i = v \mid c, s, \alpha), \tag{4}$$

with deterministic tie-breaking. One forward pass reconsiders all positions synchronously from the same input draft. The predicted modes determine which tokens are retained or revised, specifying the next complete draft without a diffusion reverse transition.

The training target matches this direct use of clean-token predictions. In contrast, under the plug-in reverse parameterization, the continuous-time uniform-noise NELBO targets a leave-one-out posterior, with the observed token's likelihood supplied by the reverse kernel (Gourevitch et al., 2026). Our cross-entropy objective instead trains the full-posterior probabilities used directly by F_α.

3.2 From Denoiser Predictions to Cavity Energy

The prediction map specifies how to propose a revised draft, but not whether the resulting candidate should replace the current one. To evaluate complete drafts, we distinguish contextual support from the evidence supplied by each observed token itself. Full-posterior predictions incorporate both sources of evidence, as appropriate for denoising under the reconstruction training channel. For candidate evaluation, however, agreement with the input token does not by itself establish support from the condition and the remaining draft. We therefore separate the known self-token likelihood from the predictor's output.

Under the channel equation 2, observing token $u = s_i$ supplies a likelihood ratio

$$\kappa_\alpha := \frac{k_\alpha(u,u)}{k_\alpha(u,v)} = 1 + \frac{N\alpha}{1-\alpha}, \qquad u \neq v, \quad 0 \leq \alpha < 1, \tag{5}$$

in favor of clean token u over any alternative v. This ratio equals one at $\alpha = 0$ and increases with the keep probability. For example, a 10:1 channel advantage can outweigh contextual odds of 1:5, leaving full-posterior odds of 2:1 in favor of the observed token. A posterior preference for the draft's existing token can therefore persist even when its context favors an alternative.

Under the joint distribution of clean data and the corruption channel, define the *cavity conditional* (Mezard et al., 1987) at position i as

$$e_i^*(v \mid c, s_{-i}, \alpha) = \Pr(x_i = v \mid c, s_{-i}, \alpha). \tag{6}$$

This conditional excludes the observation s_i while retaining the condition and all other draft tokens as noisy context. The following decomposition separates this contextual contribution from the self-token likelihood.

Proposition 1 (Cavity decomposition)**.** *For $0 \le \alpha < 1$, the full clean-token posterior satisfies*

$$\Pr(x_i = v \mid c, s, \alpha) = \frac{k_\alpha(s_i, v)\, e_i^*(v \mid c, s_{-i}, \alpha)}{\sum_{u \in [N]} k_\alpha(s_i, u)\, e_i^*(u \mid c, s_{-i}, \alpha)}. \tag{7}$$

The decomposition follows from Bayes' rule and independent corruption across positions. Since k_α is strictly positive for $\alpha < 1$, we can invert the decomposition by dividing out the channel likelihood and renormalizing. Applied to the trained predictor, this gives

$$\widehat{e}_{\theta,i}(v \mid c, s, \alpha) = \frac{p_\theta(x_i = v \mid c, s, \alpha)/k_\alpha(s_i, v)}{\sum_{u \in [N]} p_\theta(x_i = u \mid c, s, \alpha)/k_\alpha(s_i, u)}. \tag{8}$$

When the predictor equals the full posterior, this correction recovers e_i^* exactly. All estimates are obtained from the same full-draft forward pass, without an architectural change, such as the hollow networks that output leave-one-out conditionals (Sun et al., 2022; Zhao et al., 2026), or separate evaluations with individual positions removed (Salazar et al., 2020; Goyal et al., 2021).

We use these estimates to score a complete candidate draft. Fix a read keep probability $\alpha_s \in (0, 1)$, shared across candidates so that every draft is evaluated by the same scoring function. We supply the candidate s itself as the predictor's input at α_s; this read neither corrupts nor updates the candidate. Writing $d_i = p_\theta(x_i = s_i \mid c, s, \alpha_s)$, define the *cavity energy*

$$\begin{aligned} E_\theta(s) &= -\sum_{i=1}^{L} \log \widehat{e}_{\theta,i}(s_i \mid c, s, \alpha_s) \\ &= -\sum_{i=1}^{L} \log \frac{d_i}{d_i + \kappa_{\alpha_s}(1 - d_i)}. \end{aligned} \tag{9}$$

We write E_θ without the fixed condition and read level. Each term measures how strongly the condition and the rest of the noisy draft support the candidate's token once that token's own channel likelihood is divided out, so lower energy means greater estimated contextual support (LeCun et al., 2006). Because every term comes from the same read of the candidate, scoring a complete draft takes one forward pass and no additional scoring model. The correction is used only for evaluation: the proposal map F_α continues to use full-posterior predictions.

For the Bayes-optimal predictor, as $\alpha_s \uparrow 1$, the cavity conditionals approach the clean-data leave-one-out conditionals, and the energy approaches negative log pseudo-likelihood (Besag, 1975), provided every conditioning context has positive clean-data probability given c. At finite noise, the remaining draft tokens are instead treated as noisy observations. We therefore interpret E_θ as a learned, noise-smoothed pseudo-likelihood score, not a correctness certificate.

3.3 Refinement by Proposing, Reading, and Retaining

We refine a single incumbent draft through three stages: proposing, reading, and retaining (Algorithm 1). Each round re-noises the incumbent and applies the parallel prediction map F_α equation 4 for $J - 1$ passes along an inner keep-probability schedule. The final pass reads the completed candidate at the fixed level α_s to compute its cavity energy, without updating its tokens. This evaluates

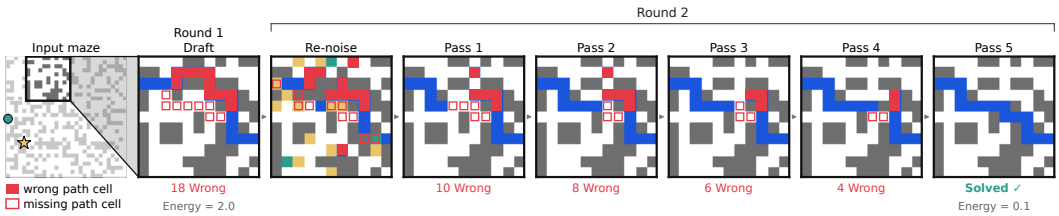


Figure 2: **Generative process on a Maze-Hard instance.** Given a maze with a start (circle) and a goal (star), DRLM generates a draft in the first round, then refines it into a valid shortest path. Each later round re-noises the draft, updates all cells in parallel, and keeps the result if its energy is lower. Panels after the input show the boxed region.

the candidate under its revised context, rather than reusing predictions conditioned on earlier drafts. The candidate replaces the incumbent only if its energy is strictly lower. Otherwise, the incumbent and its cached energy remain unchanged, and subsequent rounds refine the retained draft. This rule maintains the *best-so-far energy*. For a fixed predictor, condition, and read level, assume finite cavity energies and let s'_r and s_r denote the proposal and incumbent after round r, respectively. Then

$$E_\theta(s_r) = \min_{1 \le k \le r} E_\theta(s'_k), \qquad r \ge 1. \tag{10}$$

The first proposal establishes the incumbent, and every subsequent replacement strictly lowers its energy. Incumbent energy is therefore non-increasing, and the incumbent is the earliest proposal attaining the minimum.

Initialization uses the same procedure with $\rho = 0$: q_0 supplies a uniform random canvas independently of the initial placeholder draft. Later rounds use ρ_e with fresh, independent channel noise. Each round uses $J-1$ proposal passes and one candidate read, so R rounds require RJ model evaluations. The incumbent provides a complete best-so-far output after every round, available for return or further refinement without reevaluating its cached energy.

Algorithm 1: Energy-controlled refinement

Require: Condition c, rounds $R \ge 1$, passes per round $J \ge 2$, $0 < \rho_e < \alpha_s < 1$
Initialize any $s \in [N]^L$; $E \leftarrow +\infty$
for $r = 1, \dots, R$ **do**
  $\rho \leftarrow 0$ if $r = 1$, else ρ_e
  $(\beta_j)_{j=1}^{J} \leftarrow \mathrm{Linspace}(\rho, \alpha_s, J)$
  $s' \sim q_\rho(\cdot \mid s)$ ▷ Re-noise
  for $j = 1, \dots, J-1$ **do**
    $s' \leftarrow F_{\beta_j}(s')$ ▷ Propose
  end for
  $E' \leftarrow E_\theta(s')$ ▷ Read at α_s; no update
  if $E' < E$ **then**
    $(s, E) \leftarrow (s', E')$ ▷ Retain
  end if
end for
return s ▷ Best-so-far draft

4 Experiments

In our experiments, we evaluate our approach on three parallel reasoning tasks: Countdown, Sudoku-Extreme (Wang et al., 2025a), and Maze-Hard (Wang et al., 2025a), in which many constraint-coupled variables must be decided jointly and remain mutually consistent. These tasks require a solver to chain deductions and revise tentative assignments when they lead to contradictions. Because a single wrong position invalidates the entire solution, they test global planning rather than local pattern completion. See Appendix G for experimental details.

4.1 Countdown

Countdown is a mathematical reasoning task that generalizes the classic "24 Game" to arbitrary targets. Given a set of input numbers, the goal is to construct an expression that evaluates to a specified target using the operations $\{+, -, \times, \div\}$. The task tests both arithmetic accuracy and planning: a solver must identify useful intermediate quantities and combine them into a valid solution. For example, given the numbers $\{97, 38, 3, 17\}$ and target 14, one valid solution is $97 - 38 = 59$, $59 - 17 = 42$, $42 \div 3 = 14$.

Setup. We use the Countdown splits released by Ye et al. (2024): 500k training instances with targets in 10–100, and a held-out test set of 1,000 instances. A valid solution requires the full chain to be consistent, and multiple valid answers exist per instance, so the model must commit to one complete derivation. A solution is considered correct if every equation is arithmetically valid, every

operand is either an input number or an earlier result, and the final result equals the target. We study three settings of growing complexity by varying the number of inputs: CD-3, CD-4, and CD-5.

Baselines. We compare against autoregressive (AR) models at several scales: GPT-2 (Radford et al., 2019) trained from scratch with 6M, 85M, and 303M parameters, and pretrained LLaMA (Touvron et al., 2023) with 7B and 13B parameters, fine-tuned with LoRA (Hu et al., 2022) on the same training data. We also evaluate four paramter-matched discrete diffusion methods: the mask-absorbing models MGDM (Ye et al., 2024) and LDDM (Jo et al., 2026), and the uniform-noise models SUNDAE (Savinov et al., 2021) and UDLM (Schiff et al., 2024).

Results. Table 1 reports Pass@1, the fraction of problems solved by a single generated candidate. DRLM achieves the highest score across all three settings. On CD-3, it reaches near-perfect 99.5%, exceeding UDLM by $+0.8$ percentage points. The gains are substantially larger on CD-4: DRLM achieves 87.6%, outperforming LLaMA-13B SFT by $+36.5$ points and UDLM by $+14.3$ points. It also exceeds LDDM by $+31.3$ points.

On the more challenging CD-5 task, DRLM reaches 44.4% accuracy, compared with 7.4% for LLaMA-13B and 10.3% for LDDM. These are gains of 37.0 and 34.1 percentage points, corresponding to $6.0\times$ and $4.3\times$ their respective success rates. These results support complete-draft refinement as an effective approach to parallel arithmetic reasoning.

Table 1: Results on Countdown (CD). CD-k uses k input numbers. Best results are in bold.

Model	Params	CD-3	CD-4	CD-5
Autoregressive				
GPT-2 Scratch	6M	94.1	31.9	4.3
	85M	95.9	45.8	5.1
	303M	96.4	41.3	4.5
LLaMA SFT	7B	95.7	41.1	6.7
	13B	96.5	51.1	7.4
Discrete diffusion				
SUNDAE	6M	98.3	56.2	6.0
UDLM	6M	98.7	73.3	6.7
MGDM	6M	97.7	43.7	5.8
LDDM	6M	97.6	56.3	10.3
DRLM (ours)	6M	**99.5**	**87.6**	**44.4**

4.2 Sudoku and Maze-Hard

Sudoku-Extreme. Sudoku-Extreme (Wang et al., 2025a) requires completing a 9×9 grid so that each row, column, and 3×3 box contains the digits 1–9 exactly once. To assess how well generative models learn from data, we train on the full split of 3.8M puzzles rather than the small-sample (1K) setting (Wang et al., 2025a; Jolicoeur-Martineau, 2025), where performance on this unique-solution task may depend largely on architectural inductive biases. Accuracy measures the fraction of puzzles for which all 81 predicted cells match the unique solution.

Maze-Hard. Maze-Hard (Wang et al., 2025a) requires marking a shortest 4-connected path from a designated start S to a goal G in a 30×30 grid. We use 1,000 training and 1,000 test mazes, augmenting the training examples with the eight symmetries of the square. Unlike Sudoku-Extreme, every test maze admits multiple shortest paths. The input maze and its path-marked output are each flattened in row-major order into 900 tokens, yielding 1,800 tokens in total. A prediction is correct only if it preserves all walls and endpoints and its marked cells, together with S and G, form a single path of optimal length, verified by breadth-first search.

Baselines. For both tasks, we compare our method with autoregressive (AR) and discrete diffusion models: LLaMA (Touvron et al., 2023) 13B LoRA fine-tuned on the same training data and three parameter-matched diffusion models: UDLM (Schiff et al., 2024), MGDM (Ye et al., 2024), and LDDM (Jo et al., 2026). These models use the same training splits and evaluation criteria as DRLM.

Results. Table 2 shows that our method outperforms all autoregressive and discrete diffusion baselines on both tasks. On Sudoku-Extreme, DRLM achieves 93.9% exact-solution accuracy, a $+28.4$ percentage point improvement over the strongest diffusion baseline, UDLM (65.5%), and a $+69.8$ point improvement over LDDM

Table 2: Results on Sudoku-Extreme and Maze-Hard. Best results are in bold.

Model	Sudoku-Extreme	Maze-Hard
Autoregressive		
LLaMA SFT (13B)	1.6	1.5
Discrete diffusion		
UDLM (6M)	65.5	0.0
MGDM (6M)	12.3	10.4
LDDM (6M)	24.1	39.3
Specialized recurrent models		
HRM (27M)	55.0†	74.5

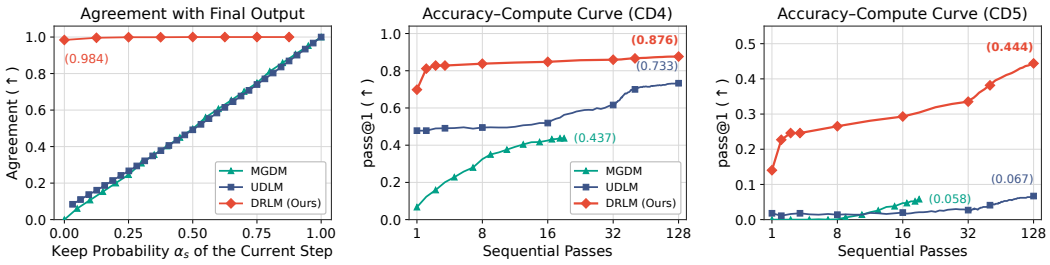


Figure 3: **Analysis. (a)** Agreement between each intermediate state and the decoder's own final output versus the current keep probability α_s. DRLM reaches 98.4% agreement after one round, while diffusion trajectories closely track their noise schedules. **(b)** Pass@1 of the current output after each sequential pass on Countdown-4 and Countdown-5, up to each method's default budget. DRLM exceeds both baselines' final accuracy after two passes on CD-4 and after one on CD-5.

(24.1%). The advantage is larger on Maze-Hard: DRLM reaches 73.0% optimal-path accuracy, $+33.7$ points above LDDM (39.3%). Notably, UDLM struggles on Maze-Hard. We hypothesize that this gap from the NELBO's weighting, which concentrates training signal on nearly clean canvases and thus provides little supervision for sampling $1{,}800$ tokens at once from random noise. Our results demonstrate that enabling complete-draft refinement on a standard transformer backbone unlocks substantive gains in parallel reasoning.

5 Analysis

Model-bounded versus schedule-bounded generation. Figure 3(a) compares decoding trajectories on held-out Countdown-4 problems using identical backbone trained on the same data. At each step, we measure the fraction of positions agreeing with the decoder's own final output. MGDM and UDLM closely track the schedule-dependent profiles α_s and $\alpha_s + (1-\alpha_s)/N$, respectively: agreement increases with the prescribed keep probability rather than saturating early. In contrast, DRLM already agrees with its final output at 98.4% of positions after the first round, while every position remains eligible for revision. This shows that trust schedule conditions predictions rather than imposing a reverse transition: the model directly determines which tokens to retain or replace.

Accuracy–Compute curve. Figure 3(b) tracks the Pass@1 of each decoder's current output after every model call on Countdown-4 and Countdown-5, up to each method's default budget. On CD-4, DRLM reaches 69.8% after one call and 81.1% after two, already above the final accuracy of UDLM (73.3%) and MGDM (43.7%), and later rounds raise it to 87.6%. UDLM instead stays near 50% for its first 16 steps and reaches its final accuracy only at the end of its schedule. On CD-5 the gap widens: DRLM's first call (14.0%) already exceeds the final accuracy of UDLM (6.7%) and MGDM (5.8%), and refinement keeps improving through all 32 rounds, to 44.4%. DRLM thus yields usable answers from the first call and converts further calls into accuracy, most visibly on the harder task, rather than relying on a schedule to complete a solution.

Table 3: Objective–Decoder ablation (Pass@1).

Training objective	Reverse-diffusion sampler (128)	Refinement decoder (128)
NELBO (UDLM)	6.9	36.6
All-position CE (DRLM)	22.5	**44.4**

Objective and decoder contribute separately. Table 3 crosses the training objective and decoder on Countdown-5 at a matched budget of 128 model evaluations, accounting for their different pre-

diction targets. Under our refinement decoder, the UDLM model's leave-one-out predictions are converted to full posteriors. DRLM under the ancestral sampler uses the inverse conversion before applying the plug-in reverse kernel. Replacing only the decoder raises accuracy from 6.9% to 36.6%, while replacing only the objective gives 22.5%. Combining both reaches 44.4%: decoder gains appear on both networks (+29.7 and +21.9 percentage points), and objective gains under both decoders (+15.6 and +7.8); all four effects exclude zero in paired bootstrap intervals (Appendix H.4). Thus, both the training objective and direct refinement contribute to the observed improvement, and their benefits persist after accounting for the different prediction targets.

Table 4: Retention ablation (Pass@1).

Decoder	CD-4	CD-5	Sudoku-Extreme	Maze-Hard
Accept every candidate	82.9	24.0	68.6	70.3
Restart from noise each round	85.3	30.6	77.0	66.2
DRLM (energy-controlled retention)	**87.6**	**44.4**	**93.9**	**73.0**

Energy-controlled retention drives refinement. Table 4 isolates the read-and-retain step at the same budget. Accepting every completed candidate, so that the read plays no role, leaves accuracy near its first-round level on Countdown and Sudoku-Extreme: Pass@1 falls from 87.6% to 82.9% on CD-4, from 44.4% to 24.0% on CD-5, and from 93.9% to 68.6% on Sudoku-Extreme. Restarting every round from a uniform canvas and keeping the lowest-energy draft recovers only part of the gain on every task (85.3%, 30.6%, 77.0%, and 66.2%), and drops to 66.2% on Maze-Hard. On Maze-Hard, re-noising and denoising again rarely degrades a draft, while refining the retained draft remains essential. This shows that the gains come from refining a retained draft under energy control, rather than from selecting among independent drafts.

6 Related Work

Discrete Diffusion. Discrete diffusion learns reverse transitions for categorical corruption processes (Austin et al., 2021; Hoogeboom et al., 2021; Campbell et al., 2022; Lou et al., 2024), including masking (Sahoo et al., 2024; Shi et al., 2024; Ou et al., 2025) and uniform replacement (Schiff et al., 2024; Sahoo et al., 2025), and has been scaled to billion-parameter language models (Nie et al., 2025; Ye et al., 2025; 2026). MGDM adapts masked diffusion to parallel reasoning (Ye et al., 2024), adaptive unmasking orders improve masked diffusion on Sudoku (Kim et al., 2025b), and LDDM propagates continuous information alongside sampled tokens (Jo et al., 2026). Remasking and correction methods enable revision through remasking (Zheng et al., 2023; Wang et al., 2025b), mixed mask–uniform corruption (Von Rütte et al., 2025), explicit training on incorrect visible tokens (Zhang et al., 2025; Schiff et al., 2026), learned per-token quality scores that select tokens to remask (Kim et al., 2025a), or corrector steps that resample tokens from leave-one-out conditionals (Zhao et al., 2026). These methods retain generation based on diffusion transitions, remasking, or interleaved correction steps. Our method instead uses uniform corruption only to construct training drafts without sampling a diffusion reverse transition, and it uses the channel likelihood to score complete candidates rather than to resample individual tokens.

Non-Autoregressive Iterative Refinement. Complete-sequence refinement predates discrete diffusion: deliberation networks polish a first-pass draft (Xia et al., 2017), and non-autoregressive translation refines drafts by iterative denoising (Lee et al., 2018) or by insertion and deletion edits (Gu et al., 2019). Mask-Predict (Ghazvininejad et al., 2019) and DisCo (Kasai et al., 2020) develop confidence- and context-dependent parallel updates. SUNDAE (Savinov et al., 2021) learns clean-sequence reconstruction from uniformly corrupted inputs with its self-generated trajectories, and alternating corruption and denoising also defines generative Markov chains (Bengio et al., 2013). Unlike these methods, which keep every revision, DRLM treats each revision as a proposal: it keeps the revised draft only if the same model assigns it lower energy. Recent structured reasoners scale test-time computation through weight-tied recurrence: HRM uses hierarchical recurrent states (Wang et al., 2025a), while TRM (Jolicoeur-Martineau, 2025) and PTRM (Sghaier et al., 2026) update both answer and latent states. DRLM instead uses the complete discrete draft itself as the recurrent state, making each round's candidate directly interpretable.

7 Conclusion

We introduced Draft Refinement Language Models (DRLMs), which use a single denoiser to propose and evaluate parallel revisions of complete drafts. All-position cross-entropy training on uniformly corrupted sequences teaches preservation and repair, enabling direct updates without a diffusion reverse transition. Correcting predictions for the known self-token likelihood yields cavity estimates for one-pass contextual evaluation, supporting a *propose–read–retain* generative loop that returns the lowest-energy evaluated draft within a fixed budget. Across Countdown, Sudoku-Extreme, and Maze-Hard, our method outperforms the evaluated autoregressive and discrete diffusion baselines. These results demonstrate the effectiveness of combining parallel revision with whole-draft evaluation for complex parallel reasoning.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

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, 2021.

Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. Generalized denoising auto-encoders as generative models. *Advances in neural information processing systems*, 26, 2013.

Julian Besag. Statistical analysis of non-lattice data. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 24(3):179–195, 1975.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

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

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*, 2020. URL https://openreview.net/pdf?id=r1xMH1BtvB.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 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. 4171–4186, 2019.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jian, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D Hwang, et al. Faith and fate: Limits of transformers on compositionality. *arXiv preprint arXiv:2305.18654*, 2023.

Linda Flower and John R. Hayes. A cognitive process theory of writing. *College Composition and Communication*, 32(4):365–387, 1981. ISSN 0010096X. URL http://www.jstor.org/stable/356600.

Kanishk Gandhi, Denise Lee, Gabriel Grand, Muxin Liu, Winson Cheng, Archit Sharma, and Noah D Goodman. Stream of search (sos): Learning to search in language. 2024.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 6112–6121, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1633. URL https://aclanthology.org/D19-1633/.

Samson Gourevitch, Yazid Janati, Dario Shariatian, Umut Simsekli, Eric Moulines, Eric P Xing, and Alain Durmus. Uniform diffusion models revisited: Leave-one-out denoiser and absorbing state reformulation. 2026.

Kartik Goyal, Chris Dyer, and Taylor Berg-Kirkpatrick. Exposing the implicit energy networks behind masked language models via metropolis–hastings. *arXiv preprint arXiv:2106.02736*, 2021.

Jiatao Gu, Changhan Wang, and Junbo Zhao. Levenshtein transformer. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 11179–11189. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/9297-levenshtein-transformer.pdf.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in neural information processing systems*, 34:12454–12465, 2021.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Mingyu Jo, Jaesik Yoon, Justin Deschenaux, Caglar Gulcehre, and Sungjin Ahn. Loopholing discrete diffusion: Deterministic bypass of the sampling wall. In *International Conference on Learning Representations*, volume 2026, pp. 93890–93918, 2026.

Alexia Jolicoeur-Martineau. Less is more: Recursive reasoning with tiny networks, 2025. URL https://arxiv.org/abs/2510.04871.

Jungo Kasai, James Cross, Marjan Ghazvininejad, and Jiatao Gu. Non-autoregressive machine translation with disentangled context transformer. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5144–5155. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/kasai20a.html.

Jaeyeon Kim, Seunggeun Kim, Taekyun Lee, David Z Pan, Hyeji Kim, Sham Kakade, and Sitan Chen. Fine-tuning masked diffusion for provable self-correction. *arXiv preprint arXiv:2510.01384*, 2025a. Jaeyeon Kim and Seunggeun Kim contributed equally; Taekyun Lee is also a co–first author.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. *arXiv preprint arXiv:2502.06768*, 2025b.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

David Kirsh. Thinking with external representations. *AI Soc.*, 25(4):441–454, November 2010. ISSN 0951-5666.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1173–1182, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1149. URL https://aclanthology.org/D18-1149/.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2024.

M. Mezard, G. Parisi, and M. Virasoro. *Spin Glass Theory and Beyond: An Introduction to the Replica Method and Its Applications*. World Scientific Lecture Notes in Physics. World Scientific Publishing Company, 1987. ISBN 9789813103917. URL https://books.google.co.kr/books?id=DwY8DQAAQBAJ.

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. *arXiv preprint arXiv:2502.09992*, 2025.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *International Conference on Learning Representations*, volume 2025, pp. 64972–65009, 2025.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. volume 1, pp. 9, 2019.

Subham Sekhar Sahoo, Marianne Arriola, Aaron Gokaslan, Edgar Mariano Marroquin, Alexander M Rush, Yair Schiff, Justin T Chiu, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=L4uaAR4ArM.

Subham Sekhar Sahoo, Justin Deschenaux, Aaron Gokaslan, Guanghan Wang, Justin T Chiu, and Volodymyr Kuleshov. The diffusion duality. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=9P9Y8FOSOk.

Julian Salazar, Davis Liang, Toan Q Nguyen, and Katrin Kirchhoff. Masked language model scoring. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 2699–2712, 2020.

Nikolay Savinov, Junyoung Chung, Mikolaj Binkowski, Erich Elsen, and Aaron van den Oord. Step-unrolled denoising autoencoders for text generation. *arXiv preprint arXiv:2112.06749*, 2021.

Yair Schiff, Subham Sekhar Sahoo, Hao Phung, Guanghan Wang, Sam Boshar, Hugo Dalla-torre, Bernardo P de Almeida, Alexander Rush, Thomas Pierrot, and Volodymyr Kuleshov. Simple guidance mechanisms for discrete diffusion models. *arXiv preprint arXiv:2412.10193*, 2024.

Yair Schiff, Omer Belhasin, Roy Uziel, Guanghan Wang, Marianne Arriola, Gilad Turok, Michael Elad, and Volodymyr Kuleshov. Learn from your mistakes: Self-correcting masked diffusion models. *arXiv preprint arXiv:2602.11590*, 2026.

Amin Sghaier, Ali Parviz, and Alexia Jolicoeur-Martineau. Probabilistic tiny recursive model. *arXiv preprint arXiv:2605.19943*, 2026.

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

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

Haoran Sun, Lijun Yu, Bo Dai, Dale Schuurmans, and Hanjun Dai. Score-based continuous-time discrete diffusion models. In *The Eleventh International Conference on Learning Representations*, 2022.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. 2023.

Dimitri Von Rütte, Janis Fluri, Yuhui Ding, Antonio Orvieto, Bernhard Schölkopf, and Thomas Hofmann. Generalized interpolating discrete diffusion. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 61810–61843. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/von-rutte25a.html.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. Hierarchical reasoning model, 2025a. URL https://arxiv.org/abs/2506.21734.

Guanghan Wang, Yair Schiff, Subham Sahoo, and Volodymyr Kuleshov. Remasking discrete diffusion models with inference-time scaling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=IJryQAOy0p.

Yingce Xia, Fei Tian, Lijun Wu, Jianxin Lin, Tao Qin, Nenghai Yu, and Tie-Yan Liu. Deliberation networks: Sequence generation beyond one-pass decoding. *Advances in neural information processing systems*, 30, 2017.

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, 2023.

Jiacheng Ye, Jiahui Gao, Shansan Gong, Lin Zheng, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Beyond autoregression: Discrete diffusion for complex reasoning and planning. *arXiv preprint arXiv:2410.14157*, 2024.

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.

Mengyu Ye, Keito Kudo, Wataru Ikeda, Ryosuke Matsuda, Keisuke Sakaguchi, and Jun Suzuki. Sumi: Open uniform diffusion language model from scratch, 2026. URL https://arxiv.org/abs/2606.19005.

Shuibai Zhang, Fred Zhangzhi Peng, Yiheng Zhang, Jin Pan, and Grigorios G. Chrysos. Corrective diffusion language models, 2025. URL https://arxiv.org/abs/2512.15596.

Yixiu Zhao, Jiaxin Shi, Feng Chen, Shaul Druckmann, Lester Mackey, and Scott Linderman. Informed correctors for discrete diffusion models. *Advances in Neural Information Processing Systems*, 38:125510–125538, 2026.

Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. A reparameterized discrete diffusion model for text generation. *arXiv preprint arXiv:2302.05737*, 2023.

A Appendix

B Use of Large Language Models

During the preparation of this manuscript, we utilized a large language model to improve grammar, clarity, and overall readability.

C Diffusion Kernels and Token-Revision Constraints

This appendix makes precise the statements about standard ancestral decoding in Section 2: the plug-in reverse kernels of MDLM and UDLM, the schedule-controlled revelation of masked diffusion, and the revision ceiling of uniform-noise diffusion. Throughout, $\tau < t$ are two diffusion times with keep probabilities $b = \alpha_\tau > a = \alpha_t \geq 0$. These times are unrelated to the read level α_s of Section 3.2. We fix one position, the condition, and the current state $\mathbf{z}_t$, and write u for the current token at that position and v for a possible next token.

C.1 Plug-in Reverse Kernels

Under the forward channel equation 1, the transition between the two times is

$$q(\mathbf{z}_t = u \mid \mathbf{z}_\tau = v) = \gamma \mathbb{1}[u = v] + (1-\gamma)\pi_u, \qquad \gamma = \frac{a}{b} \in [0, 1), \tag{11}$$

which composes with $q(\mathbf{z}_\tau \mid x)$ to give $q(\mathbf{z}_t \mid x)$. For a clean token with one-hot vector $\mathbf{e}_x$, Bayes' rule gives the exact reverse posterior

$$q(\mathbf{z}_\tau = v \mid \mathbf{z}_t = u, x) = \frac{[\gamma \mathbb{1}[v = u] + (1-\gamma)\pi_u]\,[b(\mathbf{e}_x)_v + (1-b)\pi_v]}{a(\mathbf{e}_x)_u + (1-a)\pi_u}. \tag{12}$$

Definition 1 (Plug-in kernel). *For a probability vector f, the denoiser's prediction at the position, the plug-in kernel substitutes f for $\mathbf{e}_x$ in equation 12:*

$$B_f^\pi(v \mid u) = \frac{[\gamma \mathbb{1}[v = u] + (1-\gamma)\pi_u]\,[bf_v + (1-b)\pi_v]}{af_u + (1-a)\pi_u}, \qquad af_u + (1-a)\pi_u > 0. \tag{13}$$

Applied independently at every position, it is the factorized learned transition $p_\theta(\mathbf{z}_\tau \mid c, \mathbf{z}_t)$ of MDLM and UDLM (Sahoo et al., 2024; Schiff et al., 2024).

Lemma 1. *(i) $\sum_v B_f^\pi(v \mid u) = 1$. (ii) $B_{\mathbf{e}_x}^\pi$ is the exact posterior equation 12. (iii) In general, B_f^π differs from the average $\sum_x f_x\, q(\mathbf{z}_\tau = \cdot \mid \mathbf{z}_t = u, x)$ of exact posteriors.*

Proof. (i) Summing the numerator over v gives $\gamma[bf_u + (1-b)\pi_u] + (1-\gamma)\pi_u = af_u + (1-a)\pi_u$, because $\gamma b = a$. (ii) is immediate. (iii) Take the uniform prior and the clean endpoint $b = 1$. Each exact posterior is then the point mass at x, so the average equals f, whereas equation 13 gives $B_f^{\mathbf{u}}(v \mid u) \propto k_a(u, v)\, f_v$ with k_a from equation 2: the plug-in kernel reweights the prediction by the likelihood of the current token. □

The substitution thus includes the normalizer, rather than averaging separately normalized posteriors.

C.2 Masked Revelation and Carry-Over

Let $\pi = \mathbf{m}$ be the mask, and adopt MDLM's parameterization: the prediction assigns no probability to the mask, $f_{\mathbf{m}} = 0$, and visible tokens are carried over (Sahoo et al., 2024).

Proposition 2 (Schedule-controlled revelation). *(i) At a masked position,*

$$B_f^{\mathbf{m}}(v \mid \mathbf{m}) = \frac{1-b}{1-a}\mathbb{1}[v = \mathbf{m}] + \frac{b-a}{1-a} f_v, \qquad \Pr_\theta(\mathbf{z}_\tau \neq \mathbf{m} \mid c, \mathbf{z}_t) = \frac{\alpha_\tau - \alpha_t}{1-\alpha_t}, \tag{14}$$

and a revealed token is distributed as f. (ii) At a visible position $u \neq \mathbf{m}$ with $af_u > 0$, $B_f^{\mathbf{m}}(\cdot \mid u)$ is the point mass at u, and MDLM's carry-over imposes this for every f. Hence a revealed token is never changed by later reverse steps.

Proof. (i) With $\pi = \mathbf{m}$ and $u = \mathbf{m}$, the numerator of equation 13 is $1-b$ for $v = \mathbf{m}$ and $(1-\gamma)bf_v = (b-a)f_v$ otherwise, and the denominator is $1-a$. Summing over ordinary tokens gives the revelation probability. (ii) For $u \neq \mathbf{m}$, $\pi_u = 0$, so the numerator vanishes unless $v = u$, where it equals $\gamma bf_u = af_u$, the denominator. Under the forward law, a visible token at time t requires $a > 0$. □

The revelation probability is set by the schedule, and the model determines only the revealed value. The proposition concerns this standard kernel. Remasking samplers (Wang et al., 2025b; Zheng et al., 2023; Ye et al., 2024) and confidence-ordered decoders modify it and are outside its scope.

C.3 Uniform-Noise Revision Ceiling

Let $\pi = \mathbf{u} = \mathbf{1}/N$ over the N ordinary tokens, and write $\nu = bf + (1-b)\mathbf{u}$ and $D = af_u + (1-a)/N > 0$.

Proposition 3 (Redraw form). *The uniform plug-in kernel mixes staying with a redraw from ν:*

$$B_f^{\mathbf{u}}(\cdot \mid u) = (1-\lambda_f)\,\delta_u + \lambda_f\,\nu, \qquad \lambda_f = \frac{1-\gamma}{ND} = \frac{b-a}{b\,(1-a+Naf_u)} \in (0,1]. \tag{15}$$

For $b < 1$, every token has redraw probability $\nu_v \geq (1-b)/N > 0$, so visible tokens are not absorbing.

Proof. The numerator of equation 13 is $[\gamma\mathbb{1}[v=u]+(1-\gamma)/N]\,\nu_v$. Since $\gamma\nu_u = af_u+\gamma(1-b)/N$, we have $D-\gamma\nu_u = (1-\gamma)/N$. Hence $1-\lambda_f = \gamma\nu_u/D \in [0,1)$, and the two expressions for λ_f agree. □

Proposition 4 (Revision ceiling). *Since a redraw may return the current token,*

$$\begin{aligned}\Pr_\theta(\mathbf{z}_\tau \neq u \mid c, \mathbf{z}_t) = \lambda_f(1-\nu_u) &= \frac{(b-a)\,[1-bf_u-(1-b)/N]}{b\,(1-a+Naf_u)} \\ &\leq C_{\tau,t}\Big(1-\frac{1-b}{N}\Big) \leq C_{\tau,t}, \qquad C_{\tau,t} = \frac{b-a}{b(1-a)}.\end{aligned} \tag{16}$$

The first bound is attained exactly when $f_u = 0$, strictly positive softmax outputs approach it without reaching it. For interior steps $0 < a < b < 1$, $1 - C_{\tau,t} = a(1-b)/[b(1-a)] > 0$, so the change probability is bounded strictly below one, whatever the prediction.

Proof. In equation 16, the numerator decreases and the denominator does not decrease in f_u, so the change probability is strictly decreasing in f_u and its value at $f_u = 0$ is the first bound. □

Equivalently, $C_{\tau,t}(1-(1-b)/N) = 1 - \frac{1-b}{1-a}\big(\gamma + \frac{1-\gamma}{N}\big)$. The ceiling is vacuous at the boundaries: for $a = 0$, $\lambda_f = 1$ and the kernel redraws from ν. For $b = 1$, $C_{\tau,t} = 1$, and $f_u = 0$ changes the token with probability one. The kernel equation 15 is the one used by UDLM's ancestral sampler (Schiff et al., 2024) and the ceiling is our derivation from it.

D Full-Posterior Learning and Direct Prediction

Throughout, (c, x) is drawn from the data, $\alpha \sim U(0,1)$, and $s \sim q_\alpha(\cdot \mid x)$ from equation 2. We write $p_i^*(v \mid c, s, \alpha) = \Pr(x_i = v \mid c, s, \alpha)$ for the full posterior, e_i^* for the cavity conditional equation 6, and $\varepsilon_\alpha = (1-\alpha)/N$, so that $k_\alpha(u,v) = \alpha\mathbb{1}[u=v] + \varepsilon_\alpha$.

D.1 Population Optimum of the All-Position Cross-Entropy

Proposition 5. *For any predictor f with outputs $f_i(\cdot \mid c, s, \alpha)$ over the model vocabulary $\mathcal{V} \supseteq [N]$,*

$$\mathcal{L}(f) - \mathcal{L}(p^*) = \mathbb{E}_{c,s,\alpha}\Big[\frac{1}{L}\sum_{i=1}^{L} \mathrm{KL}\big(p_i^* \,\|\, f_i\big)\Big] \geq 0, \tag{17}$$

where $\mathcal{L}$ is the objective equation 3 and $\mathcal{L}(p^*) \leq \log N$. Equality holds if and only if $f_i = p_i^*$ at every position almost surely. In particular, the optimum assigns no probability outside $[N]$.

Proof. Given (c, s, α), the expected loss at position i is $-\sum_v p_i^*(v)\log f_i(v) = H(p_i^*) + \mathrm{KL}(p_i^* \| f_i)$, with $H(p_i^*) \leq \log N$. Averaging over positions and inputs gives equation 17, and each KL term vanishes only when the two distributions agree. □

The optimum is characterized through its marginals: no independence between clean tokens is assumed, and the objective does not identify a joint distribution over sequences. A trained network approximates p^* within its capacity, data, and optimization budget, whereas the proposition concerns the population optimum.

D.2 Preservation, Repair, and Synchronous Prediction

Because each position either keeps its clean token or is replaced uniformly, an observed token can match the clean token in two ways:

$$\Pr(s_i = x_i \mid x_i, \alpha) = \alpha + \varepsilon_\alpha, \qquad \Pr(\text{kept} \mid s_i = x_i, \alpha) = 1 - \frac{1}{\kappa_\alpha}. \tag{18}$$

The loss equation 3 supervises matching positions, teaching preservation, and mismatching positions, teaching repair; preservation is learned from this supervision rather than imposed by a copy rule. The map F_α equation 4 takes the positionwise modes of p_θ for one common input draft, synchronously at all positions, with ties broken toward the lowest token index. It is neither an autoregressive sweep nor a maximizer of the joint probability of the sequence, and one application need not lower the cavity energy. In our implementation the mode is taken over the full model vocabulary; a proposal containing a token outside the N-symbol canvas alphabet cannot be scored and is never retained (Appendix E.5). No such proposal occurred in any reported run.

D.3 Training conventions

Each training sequence draws one keep probability, shared by all its positions: $\alpha = 1 - U$ with $U \sim U[0, 1)$. The network receives the replacement rate $1 - \alpha$ through the features $\{\sin(\pi k(1 - \alpha)), \cos(\pi k(1 - \alpha))\}_{k=1}^{5}$, mapped by a two-layer MLP and added to every canvas embedding. Condition tokens are never corrupted. The loss covers all L canvas positions, including padding and end-of-sequence positions in Countdown, with the softmax over the full model vocabulary $\mathcal{V}$. Corruption and scoring use the N-symbol canvas alphabet. Architecture, optimizer, and budgets are given in Table 7.

The endpoint $\alpha = 0$. The first decoding round applies F_0 to a uniformly random canvas (Algorithm 1 with $\rho = 0$). Training never samples $\alpha = 0$ exactly, so F_0 relies on continuity at the endpoint. Under the exact posterior a canvas at $\alpha = 0$ carries no information about x, and the first pass would ignore it. The learned F_0 does not: on Countdown-4, the Pass@1 of the first pass differs across decode seeds (69.4%, 69.9%, 70.2%). We make no claim that uniform initialization alone produces diverse drafts.

E Cavity energy derivations and implementation

E.1 Proof of the Cavity Decomposition

Lemma 2. *Under the joint law $p(c, x)\, q_\alpha(s \mid x)$, the observation s_i is conditionally independent of (c, x_{-i}, s_{-i}) given x_i and α, and $\Pr(s_i = u \mid x_i = v, \alpha) = k_\alpha(u, v)$.*

Proof. The joint density $p(c, x) \prod_j k_\alpha(s_j, x_j)$ depends on s_i only through the factor $k_\alpha(s_i, x_i)$, which sums to one over s_i. □

Proof of Proposition 1. By Bayes' rule and Lemma 2,

$$\Pr(x_i = v \mid c, s, \alpha) = \frac{\Pr(s_i \mid x_i = v, \alpha)\ \Pr(x_i = v \mid c, s_{-i}, \alpha)}{\sum_u \Pr(s_i \mid x_i = u, \alpha)\ \Pr(x_i = u \mid c, s_{-i}, \alpha)} = \frac{k_\alpha(s_i, v)\, e_i^*(v)}{\sum_u k_\alpha(s_i, u)\, e_i^*(u)}.$$

For $\alpha < 1$ the denominator is at least $\varepsilon_\alpha > 0$. □

What is excluded is the observation s_i: the cavity conditional still conditions on c and on every other noisy token s_{-i}, and x_i remains the variable being predicted.

E.2 Likelihood Inversion and Exact Recovery

For $\alpha < 1$, $k_\alpha(u, v) \geq \varepsilon_\alpha > 0$, so dividing equation 7 by $k_\alpha(s_i, \cdot)$ and renormalizing inverts it.

Corollary 1 (Exact recovery). *If $p_\theta = p^*$, the estimate equation 8 equals $e_i^*(\cdot \mid c, s_{-i}, \alpha)$ and therefore does not depend on s_i. At the draft's own token, with $d_i = p_\theta(x_i = s_i \mid c, s, \alpha)$,*

$$\widehat{e}_{\theta,i}(s_i \mid c, s, \alpha) = \frac{d_i}{d_i + \kappa_\alpha(1 - d_i)}, \tag{19}$$

which gives the second line of equation 9.

Proof. The first claim inverts Proposition 1. For equation 19, $k_\alpha(s_i, \cdot)$ equals $\alpha + \varepsilon_\alpha$ at s_i and ε_α elsewhere, so the normalizer of equation 8 is $d_i/(\alpha + \varepsilon_\alpha) + (1 - d_i)/\varepsilon_\alpha$. Dividing by it and using $\kappa_\alpha = (\alpha + \varepsilon_\alpha)/\varepsilon_\alpha$ gives the result. □

Corollary 2 (Odds form). *For $v \neq u = s_i$ with $e_i^*(v) > 0$,*

$$\frac{p_i^*(u)}{p_i^*(v)} = \kappa_\alpha \frac{e_i^*(u)}{e_i^*(v)}, \qquad \kappa_\alpha = 1 + \frac{N\alpha}{1 - \alpha},$$

where $\kappa_0 = 1$, κ_α is strictly increasing, and $\kappa_\alpha \to \infty$ as $\alpha \uparrow 1$.

The self-likelihood ratio multiplies the observed token's odds against every alternative. It is local to the position, whereas e_i^* depends on α through the noisy context. For an approximate predictor, equation 8 defines an estimate: the correction removes the channel's known self-likelihood, not arbitrary model error, and $\widehat{e}_{\theta,i}$ may still depend on s_i.

E.3 Scoring Completed Candidates

Each candidate is scored by a fresh read: the completed candidate itself is the input, at the common read level α_s. The last proposal pass cannot serve as this read, because its predictions were conditioned on the draft that preceded the candidate rather than on the candidate. The read samples no corruption, replaces no token, and does not apply F_{α_s}. A single forward pass returns the logits from which all L cavity estimates are computed. The self-likelihood correction is applied only here: proposals use the full-posterior predictions.

E.4 Clean-Data Pseudo-Likelihood Limit

For the Bayes-optimal predictor and a fixed candidate s, the cavity conditional is a ratio of finite sums over clean sequences:

$$e_i^*(v \mid c, s_{-i}, \alpha) = \frac{\sum_{x : x_i = v} p(x \mid c) \prod_{j \neq i} k_\alpha(s_j, x_j)}{\sum_x p(x \mid c) \prod_{j \neq i} k_\alpha(s_j, x_j)}. \tag{20}$$

Proposition 6 (Pseudo-likelihood limit). *Assume $p(x_{-i} = s_{-i} \mid c) > 0$ for every i. As $\alpha \uparrow 1$,*

$$e_i^*(v \mid c, s_{-i}, \alpha) \to p(x_i = v \mid c, x_{-i} = s_{-i}),$$

$$E_\alpha^*(s) = -\sum_{i=1}^{L} \log e_i^*(s_i \mid c, s_{-i}, \alpha) \to -\log \mathrm{PL}(s \mid c) \in [0, \infty],$$

where $\mathrm{PL}(s \mid c) = \prod_i p(x_i = s_i \mid c, x_{-i} = s_{-i})$ is the pseudo-likelihood (Besag, 1975). The limit is finite if and only if every conditional is positive. At $\alpha = 0$, $e_i^*(v) = p(x_i = v \mid c)$.

Proof. As $\alpha \uparrow 1$, $k_\alpha(u, v) \to \mathbb{1}[u = v]$, so each finite sum in equation 20 converges to its terms with $x_{-i} = s_{-i}$: the numerator to $p(x_i = v, x_{-i} = s_{-i} \mid c)$ and the denominator to $p(x_{-i} = s_{-i} \mid c) > 0$. The energy converges in $[0, \infty]$ by continuity of $-\log$ on $[0, 1]$. At $\alpha = 0$, the constant kernel $k_0 \equiv 1/N$ cancels. □

If a context has zero probability under the data, the argument does not define the clean conditional. The approach to the limit can be slow when $p(x_{-i} = s_{-i} \mid c)$ is small, because sequences that differ from s_{-i} in one position enter equation 20 with relative weight $O(1 - \alpha)$. The limit concerns the exact posterior, and a learned predictor inherits it only insofar as p_θ approximates p^* near $\alpha = 1$. At the finite read level used in practice, the other draft tokens act as noisy observations, which is why we describe E_θ as a noise-smoothed pseudo-likelihood.

E.5 Stable Logit-Space Implementation

Let $h_{i,v}$ be the read's logits restricted to the canvas alphabet. Since $(1 - d_i)/d_i = \exp(\mathrm{logsumexp}_{v \neq s_i} h_{i,v} - h_{i,s_i})$, the energy equation 9 equals

$$E_\theta(s) = \sum_{i=1}^{L} \mathrm{softplus}\Big(\log \kappa_{\alpha_s} + \operatorname*{logsumexp}_{v \neq s_i} h_{i,v} - h_{i,s_i} \Big) = \sum_{i=1}^{L} \Big[\operatorname*{logsumexp}_{v} \widetilde{h}_{i,v} - \widetilde{h}_{i,s_i} \Big], \quad (21)$$

with corrected logits $\widetilde{h}_{i,v} = h_{i,v} - \log \kappa_{\alpha_s} \mathbb{1}[v = s_i]$ and $\log \kappa_{\alpha_s} = \mathrm{log1p}((N-1)\alpha_s) - \mathrm{log1p}(-\alpha_s)$. Neither form evaluates $1 - d_i$, which is lost to rounding when d_i is close to one. Our implementation computes the energy of a batch of candidates from one forward pass:

```
logits = net(cond, cand, level=1 - alpha_s)        # bf16 forward pass, [B, L, |V|]
lp     = log_softmax(logits[..., canvas].float())  # fp32, over the N canvas symbols
own    = one_hot(canvas_index(cand), N)            # [B, L, N]
field  = lp + log_kappa * (1 - own)                # log kappa added off the own token
E      = (logsumexp(field, -1) - (lp * own).sum(-1)).sum(-1)
E[candidates holding a non-canvas token] = +inf
```

Adding $\log \kappa_{\alpha_s}$ to every entry except the own token is equivalent to subtracting it from the own token, because each term of equation 21 is invariant to a common shift of the logits. We apply no clipping and no tolerance: a candidate is retained only if its energy is strictly lower, and the incumbent's energy is cached. With $\alpha_s = 7/8$, $\kappa_{\alpha_s} = 1+7N$, that is, 127 for Countdown ($N = 18$), 64 for Sudoku-Extreme ($N = 9$), and 36 for Maze-Hard ($N = 5$). We checked this implementation against a float64 evaluation of equation 8, of the d_i-form, and of the softplus form on random logits for $\alpha_s \in \{0.5, 0.875, 0.98\}$. The largest discrepancy, 7×10^{-5} on energies summed over 64 positions, is at float32 rounding level.

E.6 Boundary Cases and Interpretation

At $\alpha_s = 0$ the correction vanishes ($\kappa_0 = 1$) and, for the exact posterior, each term reduces to the condition-only marginal $p(x_i = s_i \mid c)$, which ignores the rest of the draft. As $\alpha_s \uparrow 1$ the ideal score approaches the clean pseudo-likelihood (Proposition 6), but $\log \kappa_{\alpha_s}$ diverges and amplifies errors in the predicted $1 - d_i$. At $\alpha_s = 1$ the kernel is the identity and cannot be inverted. We use the interior value $\alpha_s = 7/8$ (Appendix H.2 reports the sensitivity to it). The energy sums positionwise log-conditionals, so it is not a normalized sequence likelihood, and a lower energy does not certify correctness. All comparisons fix the condition and the sequence length.

F Refinement Algorithm and Reproducibility

F.1 Decoding Specification

Algorithm 2 expands Algorithm 1 for a batch of conditions, with separate incumbent and candidate buffers. Each round uses the inner keep-probability schedule

$$\beta_j = \rho + \frac{j-1}{J-1}(\alpha_s - \rho), \qquad j = 1, \ldots, J, \quad (22)$$

applying $F_{\beta_1}, \ldots, F_{\beta_{J-1}}$ and reading at $\beta_J = \alpha_s$ without a token update. For $J = 2$, a round is one proposal pass at $\beta_1 = \rho$ followed by the read. In the first round $\rho = 0$, so the re-noised canvas is uniform on $[N]^L$ and independent of the placeholder (our implementation starts from an all-padding placeholder, which is never scored). Later rounds use $\rho = \rho_e$ with fresh, independent noise. Re-noising replaces a token with probability $1 - \rho$, and the uniform draw may return the same token, so the expected fraction of changed positions is

$$\Pr(s'_i \neq s_i) = (1 - \rho)\Big(1 - \frac{1}{N}\Big), \quad (23)$$

not $1 - \rho$. Our code parameterizes noise by the replacement rate $1 - \beta$. The schedule is the same linear spacing expressed in that variable.

Algorithm 2 Energy-controlled refinement, batched form of Algorithm 1

Require: conditions $c^{(1)}, \ldots, c^{(B)}$; rounds $R \geq 1$; passes per round $J \geq 2$; $0 < \rho_e < \alpha_s < 1$

$s^{(b)} \leftarrow$ placeholder, $E^{(b)} \leftarrow +\infty$ for all b ▷ incumbent and cached energy
for $r = 1, \ldots, R$ **do**
  $\rho \leftarrow 0$ if $r = 1$, else ρ_e; compute $\beta_1, \ldots, \beta_J$ by equation 22
  $s'^{(b)} \sim q_\rho(\cdot \mid s^{(b)})$ for all b, with fresh noise ▷ candidate buffer
  for $j = 1, \ldots, J-1$ **do**
    $s'^{(b)} \leftarrow F_{\beta_j}(s'^{(b)})$ for all b ▷ one batched forward pass
  end for
  $E'^{(b)} \leftarrow E_\theta(s'^{(b)})$ at α_s for all b ▷ read; no token update
  for each b with $E'^{(b)} < E^{(b)}$ **do**
    $(s^{(b)}, E^{(b)}) \leftarrow (s'^{(b)}, E'^{(b)})$ ▷ retain; other rows unchanged
  end for
end for
return $s^{(1)}, \ldots, s^{(B)}$

F.2 Best-So-Far Invariant

Proposition 7. *Fix the predictor, condition, and read level. Let E'_r be the energy of the candidate read in round r, and E_r the cached incumbent energy after round r, with $E_0 = +\infty$. If $E'_1 < \infty$, then for every $r \geq 1$,*

$$E_r = \min_{1 \leq k \leq r} E'_k,$$

E_r is non-increasing in r, and the incumbent after round r is the earliest candidate attaining the minimum. Because a read is a deterministic function of its candidate, E_r is the incumbent's energy, which therefore need not be re-read.

Proof. By induction. Round one replaces the sentinel, since $E'_1 < \infty = E_0$. If $E_r = \min_{k \leq r} E'_k$, round $r+1$ either accepts, when $E'_{r+1} < E_r$, so that $E_{r+1} = E'_{r+1} = \min_{k \leq r+1} E'_k$, or keeps $E_{r+1} = E_r \leq E'_{r+1}$. Ties are rejected, so the earliest minimizer is kept. □

Equation equation 10 is this proposition. It concerns completed, scored candidates only: re-noised canvases and intermediate proposal drafts are not scored, and the energy need not decrease across proposal passes. It does not assert that the incumbent minimizes the energy globally, that it is correct, or that decoding may stop once the energy stops decreasing, since the decoder runs for the fixed budget R. A candidate holding a token outside the canvas alphabet receives $E' = +\infty$ and is never retained.

F.3 Model Evaluations and Cached State

Each round performs $J-1$ proposal passes and one read, so R rounds use $R(J-1)+R = RJ$ model evaluations per condition. The placeholder is never scored, and the incumbent is never re-read. After every completed round, the incumbent is a complete output. Between rounds, the decoder carries only each condition's incumbent tokens and cached energy, and within a batch each condition accepts or rejects its candidate independently. Curves that score intermediate states, such as accuracy after each pass, are analyses rather than part of the decoder.

F.4 Decoding Hyperparameters

Appendix H.2 reports the sensitivity to each.

Table 5: Decoding configuration at the default budget of $RJ = 128$ model evaluations per problem.

	Countdown (CD-3/4/5)	Sudoku-Extreme	Maze-Hard
Canvas alphabet N / canvas length L	18 / 64	9 / 81	5 / 900
Rounds $\times$ passes per round, $R \times J$	32×4	32×4	16×8
Read level α_s	$7/8$	$7/8$	$7/8$
Re-noising keep probability ρ_e	0.1	0.1	0.7
Positions changed by re-noising equation 23	85.0%	80.0%	24.0%
Inner schedule	equation 22, with $\rho = 0$ in round 1 and $\rho = \rho_e$ afterwards		
Acceptance	strictly lower cavity energy, no tolerance		
Decode seeds	3 (random generator seeds 1001–1003)		
Precision	bf16 forward passes; fp32 log-softmax and energies		

Table 6: Tasks. Lengths are in tokens; N is the canvas alphabet used for corruption and scoring.

	Countdown (CD-3/4/5)	Sudoku-Extreme	Maze-Hard
Training examples	500k per setting	3,831,994	1,000 ($\times$8 symmetries)
Test examples	1,000 per setting	422,786	1,000
Condition / canvas length	20 / 64	81 / 81	900 / 900
Model vocabulary / N	31 / 18	11 / 9	6 / 5

G Implementation and Evaluation Details

G.1 Tasks and Data

Countdown. We use the Countdown splits released with MGDM (Ye et al., 2024), whose generator follows Gandhi et al. (2024). Inputs are $k \in \{3, 4, 5\}$ source numbers in 1–99 and a target in 10–100 and outputs are up to $k - 1$ equations. The test set is target out-of-distribution: nine target values never occur in training. A solution is correct if every equation is arithmetically valid with exact division, each operand is an unused input number or an earlier result, and the final result equals the target.

Sudoku-Extreme. We use the released split of Wang et al. (2025a): 3,831,994 training puzzles, without augmentation, and 422,786 test puzzles. A prediction is correct if all 81 cells match the unique solution.

Maze-Hard. We use the 1,000 training and 1,000 test 30×30 mazes of Wang et al. (2025a), augmenting training with the eight symmetries of the square. The canvas copies the maze and marks path cells. A prediction is correct if it preserves all walls and endpoints and its marked cells, together with the endpoints, form a single simple 4-connected path of shortest length, verified by breadth-first search, so any shortest path is considered correct.

G.2 Model and Training

G.3 Baselines

Baselines trained here use our data splits, correctness criteria, and checkpoint rule, applied to their own decoders.

Diffusion baselines. UDLM (Schiff et al., 2024) uses our backbone (Table 7) with the same data, optimizer, and budget as DRLM. It differs only in its objective, the continuous-time uniform-noise NELBO, and its decoder, the ancestral sampler with $T = 128$ denoising steps. SUNDAE (Savinov et al., 2021) also uses our backbone with the same optimizer and training budget, and follows the original objective and sampler. It is trained by unrolled denoising with no corruption level conditioning: the model denoises all positions once, a second pass denoises a sample of its own first-pass prediction, and the loss averages the cross-entropy of both passes over all positions. MGDM (Ye et al., 2024) is trained with its official code and 6M recipe (a 3-layer GPT-2 of width 384) with identical training budget as DRLM, and decoded with its top-k decoder for $T = 20$ steps. In Countdown,

Table 7: DRLM training configuration, shared across tasks unless noted.

Architecture	bidirectional pre-LN Transformer: width 192, 14 layers, 4 heads, MLP width 768, GELU no biases in attention and MLP projections; untied output head
Embeddings	shared token embedding; learned position embeddings for condition and canvas; level features (Appendix D.3) added to canvas embeddings
Parameters	6.27M (Countdown), 6.28M (Sudoku-Extreme), 6.59M (Maze-Hard)
Objective	all-position cross-entropy equation 3, one $\alpha \sim U(0,1)$ per sequence
Optimizer	AdamW, learning rate 10^{-3}, default betas, weight decay 0.01, gradient-norm clipping 1.0
Learning-rate schedule	200-step linear warm-up, cosine decay to $0.1\times$ the peak
Batch	256 sequences (Maze-Hard: 64 with 4 accumulation steps)
Budget	300M sequences, 1,171,875 steps (Maze-Hard: 50M, 195,312 steps)
Precision	bf16 autocast, TF32 matrix multiplies; dropout 0
Initialization	$\mathcal{N}(0, 0.02^2)$; residual output projections $\mathcal{N}(0, 0.02^2/28)$
Hardware	$4\times$ NVIDIA RTX A6000

numbers differ from Ye et al. (2024) due to the differences in evaluation. We consider solutions invalid if the numbers aren't provided in the input or derived by an earlier result, following Jo et al. (2026). LDDM-G (Jo et al., 2026) adds loopholing to the MGDM recipe (self-conditioning rate 0.9) and decodes for $T = 20$ steps, keeping unmasked tokens fixed.

Autoregressive baselines. Countdown results for GPT-2 trained from scratch and for LLaMA-7B/13B are taken from Ye et al. (2024). On Sudoku-Extreme and Maze-Hard, we fine-tune LLaMA-13B with the LoRA (Hu et al., 2022) recipe in the official code of Ye et al. (2024) (rank 16, learning rate 10^{-4}, batch 128) for its number of optimizer steps.

Recurrent models. HRM(Wang et al., 2025a) and TRM(Jolicoeur-Martineau, 2025) reported results are from Jolicoeur-Martineau (2025). Reported numbers on Sudoku-Extreme are trained and evaluated on a small training set (1K) with $1,000$ shuffling augmentation per data example. Their Maze-Hard metric requires an exact match to a single reference path, whereas we accept any shortest path, verified by breadth-first search, since every test maze admits multiple shortest paths.

H Additional Experiments

All diagnostics use the default configuration of Table 5 and change one component at a time. Entries are Pass@1 (%) at 128 model evaluations, averaged over three decode seeds ($\pm$: standard deviation over seeds). The tables are generated from the run logs.

H.1 Effect of the Self-Likelihood Correction

To isolate the self-likelihood correction, we replace the cavity energy with the raw self-token energy $E_\theta^{\text{raw}}(s) = -\sum_i \log d_i$, computed from the same read, and keep everything else fixed: the model, the read level, the proposal schedule, and the budget. At a fixed read level, both scores sum one term per position, and both terms decrease in the read probability d_i: the raw term is $-\log d_i$, and by equation 19 the cavity term is $\log\big(1+\kappa_{\alpha_s}(1-d_i)/d_i\big)$. Since κ_{α_s} grows with the read level, the decomposition predicts that the correction matters more as α_s increases.

We test this prediction at five read levels. At each level we run the full decoder once with each score, with the proposal schedule ending at that level as in Algorithm 2, and pair the two runs over problems.

Table 8 compares decoding accuracy on Sudoku-Extreme under cavity and raw energy scoring. The cavity energy leads at every read level, and the lead widens with κ_{α_s}: $+5.2$ points at the default $\alpha_s = 7/8$ ($\kappa_{\alpha_s} = 64$), $+5.4$ at $15/16$, and $+5.8$ at both 0.98 and 0.995 ($\kappa_{\alpha_s} \geq 442$), where the paired intervals exclude zero. The two decoders drift apart as the read level rises: the cavity decoder improves from 93.9% to 94.3%, while the raw decoder slips from 88.8% to 88.5%. These results show that correcting for self-token evidence improves draft refinement over raw self-token scoring

Table 8: Self-likelihood correction on Sudoku-Extreme. Pass@1 at 128 evaluations when the decoder accepts candidates by the cavity or by the raw energy, both read at level α_s.

α_s	κ_{α_s}	Cavity	Raw	Difference (Cavity − Raw)
3/4	28	94.0	88.8	+5.2
7/8	64	93.9	88.7	+5.2
15/16	136	94.0	88.6	+5.4
0.98	442	94.3	88.5	+5.8
0.995	1,792	94.3	88.5	+5.8

at the same inference budget. The larger gains at higher read levels are consistent with the increasing self-likelihood advantage supplied by the corruption channel.

H.2 Sensitivity at a Fixed Budget

Table 9 varies one hyperparameter at a time with $RJ = 128$ fixed, so $R = 128/J$. Bold entries are the defaults.

Table 9: Pass@1 at $RJ = 128$ when one hyperparameter is changed.

Setting	CD-4	CD-5	Maze-Hard
$J = 2$	82.7	26.3	67.5
$J = 4$	87.6	44.4	72.3
$J = 8$	86.9	41.7	73.0
$J = 16$	86.2	36.7	72.4
$\alpha_s = 3/4$	87.7	44.9	71.5
$\alpha_s = 7/8$	87.6	44.4	73.0
$\alpha_s = 15/16$	87.7	44.7	73.4
$\rho_e = 0$ (restart)	85.3	30.6	66.2
$\rho_e = 0.1$	87.6	44.4	–
$\rho_e = 0.3$	87.0	35.5	–
$\rho_e = 0.5$	–	–	72.2
$\rho_e = 0.7$	–	–	73.0

One proposal pass per round ($J = 2$) is the worst setting on every task, and the defaults are the best tested values of J. Between $\alpha_s = 3/4$ and $15/16$ the read level changes Countdown accuracy by at most 0.5 points and Maze-Hard accuracy by 1.9 points. Restarting from noise ($\rho_e = 0$) is the worst re-noising setting on every task, and on CD-5 keeping more of the incumbent ($\rho_e = 0.3$) is also clearly worse than the default.

For Sudoku-Extreme we selected the configuration on the 2,000-puzzle subsample among four candidates (Table 10) and fixed it before evaluating the remaining 420,786 puzzles. Restarting from noise ($\rho_e = 0$, $J = 4$), which was not a candidate, reaches 77.0 ± 0.2 on the subsample: above keeping most of the incumbent ($\rho_e = 0.7$), but far below the selected $\rho_e = 0.1$.

Table 10: Sudoku-Extreme: configuration selection on the 2,000-puzzle subsample, and the selected configuration on the held-out complement and the full test set.

Configuration	Pass@1
Subsample: $J = 4$, $\rho_e = 0.1$	93.9 ± 0.2
Subsample: $J = 4$, $\rho_e = 0.7$	74.5 ± 0.1
Subsample: $J = 8$, $\rho_e = 0.1$	91.3 ± 0.4
Subsample: $J = 8$, $\rho_e = 0.7$	75.1 ± 0.3
Selected, held-out complement (420,786 puzzles, 3 seeds)	93.9 ± 0.0
Selected, full test set (422,786 puzzles)	93.9 ± 0.0

H.3 Budget, Acceptance, and Coverage

Table 11 reports Pass@1 as the budget grows and the fraction of problems whose incumbent is replaced in a given round. *Coverage* is the fraction of problems for which at least one completed candidate of the run was correct which upper-bounds what any rule selecting among these candidates could achieve.

Table 11: Pass@1 by model evaluations, coverage at 128 evaluations, and the fraction of problems whose incumbent is replaced in the last round.

	Pass@1 after this many model evaluations							
Task	4	8	16	32	64	128	Coverage	Accepted, last round (%)
CD-4	82.8	83.8	84.8	85.9	86.6	87.6	88.6	0.10
CD-5	24.6	26.5	29.3	33.5	38.2	44.4	44.8	1.93
Maze-Hard	–	62.9	69.3	70.7	72.2	73.0	73.9	0.47

Pass@1 stays within one point of coverage on every task, so the energy rarely discards a correct candidate that the decoder has proposed. The remaining failures are mostly problems for which no correct candidate appears within the budget. The examples in Appendix I show both kinds of failure. On Maze-Hard, most repairs by a retained candidate remove extra path cells, while a minority reroute part of the path or fill a gap. Across the rounds of decode seed 1 that turned an invalid draft into a solved one, the number of wrong path cells never increased from one proposal pass to the next.

H.4 Objective and Decoder

Table 3 crosses the training objective with the decoder on CD-5 at 128 model evaluations. Table 12 gives the cells with standard deviations over three decode seeds and the paired effects with 95% bootstrap intervals over problems. The UDLM model has the backbone of Table 7 and is trained with UDLM's NELBO on the same data and budget. Its predictions target the leave-one-out posterior, so under our decoder they are converted to full posteriors by adding $\log \kappa_\alpha$ to the observed token's logit over the canvas alphabet (Proposition 1), for proposals and reads alike. Under the ancestral sampler ($T = 128$, the kernel of Proposition 3), DRLM's full posteriors are converted to leave-one-out predictions by subtracting $\log \kappa_{\alpha_t}$ at every step. The conversions change the cross cells by at most 1.3 points (rows "without conversion").

Table 12: Objective × decoder on CD-5 at 128 model evaluations: Pass@1 per cell (±: standard deviation over decode seeds) and paired effects (points) with 95% bootstrap intervals over problems.

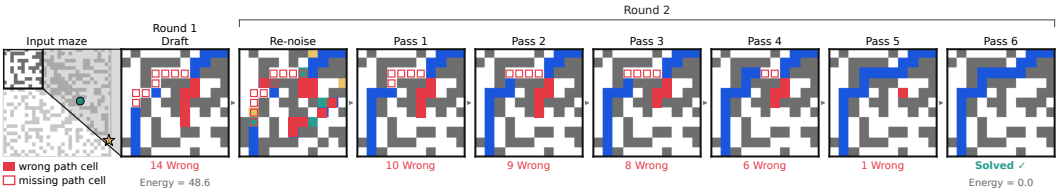
Cell or effect	Pass@1 or difference
DRLM (CE) × refinement decoder	44.4 ± 1.0
UDLM (NELBO) × refinement decoder	36.6 ± 0.6
DRLM (CE) × ancestral sampler	22.5 ± 0.3
UDLM (NELBO) × ancestral sampler	6.9 ± 0.5
UDLM × refinement decoder, without conversion	37.3 ± 0.8
DRLM × ancestral sampler, without conversion	21.2 ± 0.3
Objective, under the refinement decoder	$+7.8$ $[+5.7, +9.9]$
Objective, under the ancestral sampler	$+15.6$ $[+13.6, +17.7]$
Decoder, DRLM model	$+21.9$ $[+19.8, +24.1]$
Decoder, UDLM model	$+29.7$ $[+27.5, +32.1]$
Interaction	-7.8 $[-10.5, -5.2]$

Both the objective and the decoder improve accuracy under either choice of the other. The effects are not additive: the refinement decoder helps the UDLM model more than the DRLM model, and the objective matters more under the ancestral sampler.

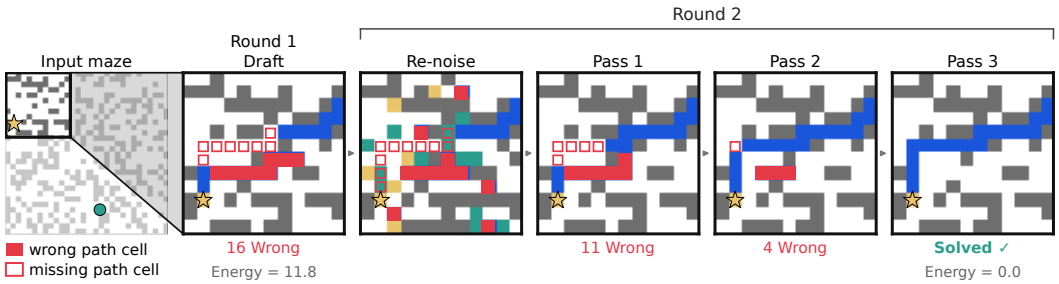
I QUALITATIVE EXAMPLES

All examples come from the reported runs (default configuration). Figure 4 shows Maze-Hard rounds in the format of Figure 2, Table 13 shows Countdown-4 rounds as text, and Figure 5 shows Sudoku-Extreme puzzles from the 2,000-puzzle subsample.

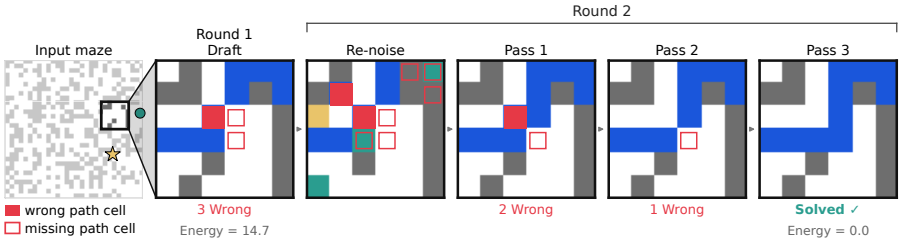
On every task, a round re-noises the retained draft, rebuilds a complete candidate over its passes, and keeps the candidate only if its energy is lower. The tasks differ in how much of the draft re-noising keeps, and hence in how much a successful revision changes. On Maze-Hard ($\rho_e = 0.7$), a revision that turns an invalid draft into a valid one changes a median of 2.5 of the 900 cells. On Countdown and Sudoku-Extreme ($\rho_e = 0.1$), it rewrites a large part of the answer: a median of 13 of 26 characters on CD-4, 14 of 33 on CD-5, and 28 of 81 cells on Sudoku-Extreme.).



(a) The first draft follows a wrong corridor; one round refines the path onto the correct one over six passes.



(b) Near the goal, the first draft uses the row below the shortest path; one round refines it.



(c) The first draft is broken; one round connects the two segments.

Figure 4: Maze-Hard repairs by a single retained proposal. Panels after the input show the boxed region. Red fill marks path cells not in the final path, and red outlines mark final-path cells not yet drawn.

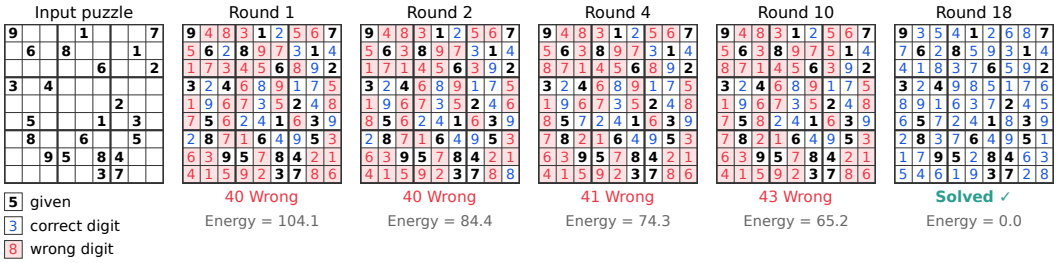
Single-token repairs such as the second example are the exception on Countdown: only 17% of the revisions that fix an invalid CD-4 draft change at most three characters (4% on CD-5). The failure in Table 13 illustrates the limit of a learned score: the energy measures the model's estimated contextual support for each token, not arithmetic validity, and here it prefers an invalid draft. Figure 6 shows the same failure on Maze-Hard, where a one-cell spur on a solved path lowers the energy and is retained. Such ranking errors are rare in aggregate (Appendix H.3).

Sudoku-Extreme. Figure 5 shows the same round structure on Sudoku. As on Maze-Hard, the round that fixes a draft removes errors progressively over its passes: for all 527 subsample puzzles that are wrong after round 1 and solved within 128 evaluations (decode seed 1), the number of wrong cells never increases from one pass of the solving round to the next. A solving revision changes more cells than on Maze-Hard because the errors of a retained draft are coupled. A median of only 20% of its wrong cells conflict with another cell of their row, column, or box, and a median of 70% of

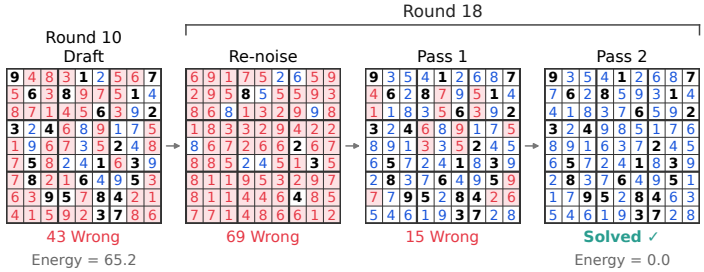
Table 13: Countdown-4 rounds. Each candidate is read at $\alpha_s = 7/8$ and kept only if its energy is lower than the incumbent's. Rounds whose candidate repeats the incumbent are omitted.

Round	Candidate	Valid	Energy	Kept
Numbers 9, 7, 13, 21*; target* 89*: the plan is replaced*				
1	`21-7=12,9*13=102,102-13=89`	no	15.73	yes
2	`9*13=117,7+21=28,117-28=89`	yes	0.06	yes
Numbers 81, 4, 1, 64*; target* 96*: one token is repaired, and a worse candidate is rejected*				
1	`81-1=80,64/4=14,80+16=96`	no	18.72	yes
2	`81-1=80,64/4=16,80+16=96`	yes	0.00	yes
3	`81+1=80,64/4=14,80+14=96`	no	15.83	no
Numbers 42, 47, 9, 14*; target* 81*: failure, a lower-energy invalid candidate replaces a valid draft*				
1	`47-42=5,14-5=9,9*9=81`	yes	2.40	yes
9	`47-42=5,14+5=9,9*9=81`	no	2.00	yes

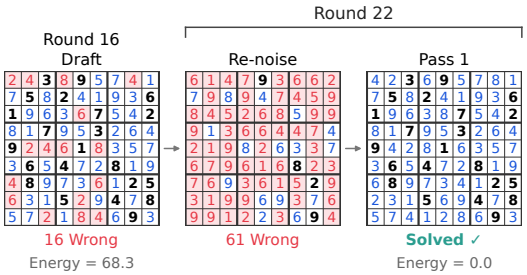
the rows, columns, and boxes that contain an error still hold all nine digits. The wrong cells of such a unit permute its correct digits, so they can only be corrected together. Before the solving round, retained drafts replace one another with decreasing energy while staying about as far from the solution (Figure 5a).



(a) The drafts retained for one puzzle, each with lower energy than the one it replaces, until round 18 retains the solution.



(b) Round 18 of the same puzzle: re-noising replaces most cells of the round-10 draft, and two passes reach the solution.



(c) Another puzzle: one pass solves it from the re-noised round-16 draft.

Figure 5: Sudoku-Extreme refinement on two puzzles of the 2,000-puzzle subsample. Bold digits are givens, blue digits match the unique solution, and red cells differ from it. Re-noising acts on all 81 cells, givens included. Each round has three proposal passes before its read; passes after the first solved one are omitted.

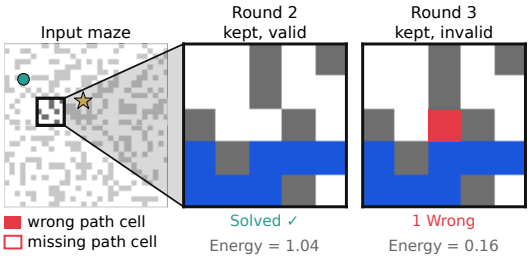


Figure 6: A Maze-Hard failure case. Round 2 retains a valid shortest path while round 3 proposes the same path with one extra cell, whose lower energy leads the decoder to replace the valid draft.