REVEAL POLICIES IN DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Masked diffusion language models (DLMs) generate text by repeatedly predicting masked tokens and choosing which position to reveal next. We show that this *reveal policy* is part of the decoder specification: with the same conditional predictions, different causal policies induce different distributions over final answers. We call each such distribution a reachable answer law and their set the reachable answer-law polytope, characterize that polytope recursively, and determine what repeated samples under one or several policies identify. For a finite acyclic decoder with known transition kernels, Bellman and occupancy-measure programs exactly optimize terminal-answer utility under hard per-generation or expected model-evaluation budgets. We prove a sharp value-of-information result: response information beyond current per-position marginals can carry nearly half the utility range, establishing revealed history and lookahead as valuable signals for optimal reveal control. On trained models, with one token revealed after each of 256 evaluations, confidence ordering beats random ordering across four DLMs and matches or beats left-to-right decoding at the same budget. A four-path reveal-policy portfolio improves over confidence batching by 6.7 points for Dream-7B and 5.6 for LLaDA-1.5 on the GSM8K grade-school math benchmark. This demonstrates a matched-compute gain from combining reveal policies. On the Massive Multitask Language Understanding (MMLU) benchmark, 32 randomized-order continuations from a threshold-selected partial state end in its leading option in 95.6–99.2% of samples across four models. Thus threshold-selected partial states induce highly concentrated answer laws even under randomized future reveal order.

1 INTRODUCTION

Masked diffusion language model decoders combine conditional token predictions with a causal reveal policy. At each step, the model predicts the masked tokens and the decoder chooses which position to reveal next (Campbell et al., 2022, Sahoo et al., 2024, Nie et al., 2025). That choice changes the partial sequence on which every later prediction is conditioned. An autoregressive model fixes one composition order, whereas a diffusion language model admits multiple causal compositions. When predictions from different partial sequences are incompatible with a single joint distribution over completed outputs, changing the next revealed position can change the distribution over final answers. The model and reveal policy therefore jointly determine the decoder's answer distribution. Confidence-based reveal ordering is already a useful decoding rule (Ghazvininejad et al., 2019, Nie et al., 2025), and confidence gaps have also been used to reveal all remaining tokens at once (Li et al., 2025). Prior theory shows that deterministic unmasking rules can induce different sequence distributions when model predictions depend on reveal order (Turok et al., 2026, Thm. 4.2). We develop an answer-level characterization that also covers randomized policies whose choices use only information available so far. It states what samples under one or multiple policies identify and supports optimization under model-evaluation constraints.

For a partial state s, let $\mathcal{R}(s)$ be the set of final-answer distributions reachable by admissible reveal policies, where an admissible policy never observes a masked token before choosing the next position. Choosing a policy selects one point in $\mathcal{R}(s)$. We prove an exact recursion for this set: choose the next position, average over the token drawn there, recurse from each resulting state, and allow probabilistic mixtures over actions. The resulting set is a nonempty convex polytope. It collapses to one point when the model's conditional predictions all agree with one joint output distribution. This view separates policy effects, sampling variation, and correctness. Different points in $\mathcal{R}(s)$ are answer

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.

distributions induced by different policies. Repeated generations under one policy estimate its answer law. Comparing a growing family of policies identifies its largest pairwise difference and gives an increasingly informative certificate of reachable-set width. Comparison with an external reference answer evaluates correctness separately. These distinctions determine what experiments that branch repeated continuations from the same saved state can establish.

The reachable set also defines a control problem. Assigning a score to each final answer and maximizing its expectation over $\mathcal{R}(s)$ gives a standard Bellman recursion. For a finite acyclic decoder with known transition kernels, augmenting the state with remaining budget finds the optimal policy under a cap that every generation must respect. An occupancy-measure linear program gives the exact utility frontier when only expected evaluation cost is constrained. Restricting planning to the next one or two reveal decisions, followed by a fixed baseline policy, gives limited-depth planning that becomes exact when it covers all remaining decisions. One consequence identifies the value of response-aware control. Confidence ordering uses only current token probabilities. Our two-field construction holds those marginals fixed while reversing the optimal first action, so second-step responses provide nearly half a utility range of additional value to a randomized controller. This identifies revealed history, lookahead, and learned structure as valuable signals for reveal control. In a finite-state benchmark with the full graph known, the preregistered primary arm (minimum-width Bellman) improves normalized five-budget utility area over confidence by 0.074182, with 95% interval $[0.070811, 0.077597]$, and unrestricted Bellman control improves it by 0.0805, with 95% interval $[0.0771, 0.0839]$ (Appendix C). The primary arm captures 92% of full control's gain while evaluating $8.9\times$ fewer planning transitions. The trained-model evidence covers four consequences of reveal control: policy sensitivity, portfolio value, answer-law concentration, and early stopping. First, at matched GSM8K evaluation budgets (Cobbe et al., 2021), confidence ordering improves accuracy over random reveal by 6.0–15.7 percentage points across four DLMs and matches or beats left-to-right decoding, demonstrating exploitable policy freedom (Section 3.1). Second, at that same budget, a fixed portfolio of confidence, margin, entropy, and left-to-right improves over four-path confidence batching by 6.7 points on Dream-7B and 5.6 in a preregistered LLaDA-1.5 replication. Third, repeated continuations that differ only in future reveal order, branched from the same threshold-selected partial MMLU state (Hendrycks et al., 2021), end in the leading option in 95.6%–99.2% of samples across four models. Fourth, a prospective two-model test uses the threshold as a stopping rule with full-decoding fallback and cuts model evaluations by 53–54% while staying within a preregistered one-point accuracy tolerance (Appendix C).

2 Reachable Answer Laws Under Reveal Control

A masked decoder supplies a token distribution at every masked position, while the reveal policy chooses which position to reveal next. We characterize the answer distributions reachable by changing that policy, determine what finite policy experiments identify, and derive the corresponding finite-state optimization under evaluation constraints. We validate exact and limited-depth planning on finite decoders.

2.1 Reachable Laws and Identification

Let $\mathcal{I} = \{1, \dots, L\}$ index the L output positions. Position i takes a token in a finite set $\mathcal{X}_i$, and a completed sequence $x \in \prod_i \mathcal{X}_i$ is mapped to an answer $A(x)$ in the finite answer set $\mathcal{A}$. The decoder state s contains the revealed tokens and any internal variables that affect later predictions. It is a Markov state, meaning that once s is known, earlier reveal choices and outcomes provide no additional information about the next transition.

Let $\mathcal{I}(s)$ be the positions still eligible to be revealed. If position $i \in \mathcal{I}(s)$ is chosen as the next reveal action, then $K_i(y \mid s)$ is the probability of drawing token y, and $s_+(i, y)$ is the resulting state. We write $\mathcal{X}_i^+(s) = \{y : K_i(y \mid s) > 0\}$ for the tokens that can be drawn. A state is terminal when every position has been revealed and nonterminal otherwise. Every terminal state is *viable*. Recursively, a nonterminal state is viable when some reveal action has only viable positive-probability successors. Call such an action viable at s, and write $\mathcal{I}_{\mathrm{v}}(s)$ for the set of viable reveal actions. Viability therefore means that at least one admissible sequence of actions completes the output. Assume that the resulting state graph is finite.

A causal reveal policy chooses the next eligible position using the current state and private randomness instead of the values of masked tokens. Its randomness is independent of the model's token draws. Every position is revealed at most once, and the policy eventually completes the output. We call such policies admissible. Let X be the random completed sequence generated from state s. For policy π, define its answer distribution by

$$r_{s,\pi}(a) = \Pr_{\pi}(A(X) = a \mid s), \qquad a \in \mathcal{A}.$$

We call $r_{s,\pi}$ the *answer law* of policy π at state s. The *reachable answer-law set* is

$$\mathcal{R}(s) = \{r_{s,\pi} : \pi \text{ is admissible from } s\} \subseteq \Delta(\mathcal{A}),$$

where $\Delta(\mathcal{A})$ is the set of probability distributions over $\mathcal{A}$.

The answer distribution averages two sources of randomness: the tokens drawn from the model and any random choices made by the policy. Fixing a policy selects one distribution in $\mathcal{R}(s)$. Changing the policy can select a different distribution. Repeated generations under one fixed policy estimate its selected answer law, and comparison with an external reference answer evaluates correctness on the same samples.

To describe this geometry, let $\mathbf{e}_a$ be the distribution that assigns probability one to answer a. Let conv denote the convex hull, the set of all finite probabilistic mixtures of its argument. A polytope is the convex hull of finitely many points.

Theorem 1 (Reachable answer-law recursion)**.** *For every terminal state identified with its completed sequence x, and every nonterminal viable state s, the reachable sets satisfy*

$$\mathcal{R}(x) = \{\mathbf{e}_{A(x)}\}, \tag{1}$$

$$\mathcal{R}(s) = \mathrm{conv} \bigcup_{i \in \mathcal{I}_v(s)} \left\{ \sum_{y \in \mathcal{X}_i^+(s)} K_i(y \mid s) r_y : r_y \in \mathcal{R}\big(s_+(i,y)\big) \right\}. \tag{2}$$

For every viable state s, $\mathcal{R}(s)$ is a nonempty convex polytope.

The recursion follows one reveal at a time. The union considers each viable next position, the weighted sum averages over its sampled token, and the separate r_y choices let later actions depend on that token. The convex hull represents randomization over the next action. Backward induction proves the result in Appendix B.

This set records more than a best policy. It contains every obtainable answer distribution, although different completed-sequence distributions can map to the same point through A. Small graphs permit explicit construction. More generally, the set clarifies when policy matters and what finite comparisons can identify. Optimizing linear scores over the set later yields control recursions without explicitly constructing the polytope. Randomization fills the polytope, but every linear terminal utility has a deterministic-policy optimizer.

The recursion also characterizes policy invariance. A useful sufficient condition is that all token kernels come from one joint distribution Q over completed outputs. Suppose every state under consideration represents conditioning Q on its revealed tokens and, for every eligible i and $y \in \mathcal{X}_i^+(s)$,

$$K_i(y \mid s) = Q(X_i = y \mid s), \qquad Q(\cdot \mid s_+(i,y)) = Q(\cdot \mid s, X_i = y).$$

Then every admissible policy produces the same completed-output and answer distribution, so $\mathcal{R}(s)$ is a singleton. Answer projection through A can additionally make distinct completion laws share one answer law.

To connect the full reachable set to finite policy experiments, consider two quantities with different meanings. First,

$$D_{\mathrm{TV}}(s) = \max_{r,r' \in \mathcal{R}(s)} \tfrac{1}{2} \|r - r'\|_1 \tag{3}$$

is the largest total-variation distance between two reachable answer distributions, where $\|r - r'\|_1 = \sum_{a \in \mathcal{A}} |r(a) - r'(a)|$. It is zero exactly when every admissible policy gives the same answer distribution, independently of whether that answer is correct. Second, for a particular answer a,

$$\delta_a(s) = \max_{r \in \mathcal{R}(s)} \{1 - r(a)\} = 1 - \min_{r \in \mathcal{R}(s)} r(a) \tag{4}$$

is the largest probability, over all admissible policies, of returning an answer other than a. Therefore, for $0 \le \epsilon \le 1$, $\delta_a(s) \le \epsilon$ means every admissible policy returns a with probability at least $1 - \epsilon$. Correctness of a is determined separately by an external reference.

Now fix a nonempty finite family Π_0 of tested policies. The quantities defined by their exact answer distributions are

$$D_{\mathrm{TV}}^{\Pi_0}(s) = \max_{\pi,\pi' \in \Pi_0} \tfrac{1}{2}\|r_{s,\pi} - r_{s,\pi'}\|_1, \qquad \delta_a^{\Pi_0}(s) = 1 - \min_{\pi \in \Pi_0} r_{s,\pi}(a).$$

Restricting the maximizations gives

$$D_{\mathrm{TV}}^{\Pi_0}(s) \le D_{\mathrm{TV}}(s), \qquad \delta_a^{\Pi_0}(s) \le \delta_a(s).$$

Independent completed generations from the same state estimate each $r_{s,\pi}$. For a fixed finite family, the empirical frequencies converge to the displayed population quantities as the number of generations per policy grows. At the population level, a nonzero distance between two policies proves a policy effect. Finite-sample uncertainty intervals certify such differences, and enlarging the tested family can only sharpen its population lower bound. On the exactly measured three-slot graphs, the eight tested policies realize 96.0% and 96.8% of the full reachable diameter on average for Dream-7B and LLaDA-1.5, demonstrating that a compact family can capture most reachable width (Appendix C.2).

2.2 Optimizing Over Reachable Answer Laws

Identification asks which reachable answer laws differ. Control asks which reachable law maximizes a specified answer score. Assign each answer a a numerical score v_a. For a distribution r, the expected score is $\langle v, r\rangle = \sum_{a \in \mathcal{A}} v_a r(a)$. The support function

$$h_s(v) = \max_{r \in \mathcal{R}(s)} \langle v, r\rangle$$

is therefore the largest expected score attainable by any admissible policy. Because $\mathcal{R}(s)$ is defined before v, the same set supports any bounded terminal-answer utility. For $C \subseteq \mathcal{A}$, let $\mathbf{1}_C$ be one on answers in C and zero elsewhere. The geometry above can be recovered from separate optimizations of these one-number scores:

$$D_{\mathrm{TV}}(s) = \max_{C \subseteq \mathcal{A}} \{h_s(\mathbf{1}_C) + h_s(-\mathbf{1}_C)\},$$

$$\max_{r \in \mathcal{R}(s)} r(a) = h_s(\mathbf{1}_{\{a\}}), \qquad \min_{r \in \mathcal{R}(s)} r(a) = -h_s(-\mathbf{1}_{\{a\}}),$$

$$\delta_a(s) = 1 + h_s(-\mathbf{1}_{\{a\}}).$$

These identities reduce questions about the whole polytope to optimization of an expected terminal score. In particular, if a completed output x receives score $U_v(x) = v_{A(x)}$, then $h_s(v) = \max_\pi \mathbb{E}_\pi[U_v(X) \mid s]$.

To extend this optimization to model-evaluation costs, parallel reveals, and termination actions, let $\mathcal{D}$ and $\mathcal{Z}$ be finite sets of decision and terminal states in an acyclic graph. At $s \in \mathcal{D}$, a policy chooses from the finite action set $\mathcal{U}(s)$. Action u has known successor distribution $T_u(\cdot \mid s)$ on $\mathcal{D} \cup \mathcal{Z}$ and nonnegative integer cost $c(s,u)$. It may reveal one or several positions, or terminate. Every action moves to a later decision state or a terminal state, every decision state has an action, and terminal state z has bounded utility $U(z)$.

For a nonnegative integer remaining budget b, let $\mathcal{U}_b(s)$ contain exactly the actions with $c(s,u) \le b$ for which, after every positive-probability successor s', some policy completes with realized future cost at most $b - c(s,u)$. Assume this set is nonempty at every state and budget considered.

Theorem 2 (Exact optimization with known transitions)**.** *Let $V_U^*(s)$ be the maximum expected terminal utility from s. Without a compute constraint, it satisfies*

$$V_U^*(z) = U(z), \qquad V_U^*(s) = \max_{u \in \mathcal{U}(s)} \sum_{s'} T_u(s' \mid s) V_U^*(s'). \tag{5}$$

Let $W_U^*(s,b)$ be the corresponding optimum when every realized cost is at most b. It satisfies

$$W_U^*(z,b) = U(z), \qquad W_U^*(s,b) = \max_{u \in \mathcal{U}_b(s)} \sum_{s'} T_u(s' \mid s) W_U^*(s', b - c(s,u)). \tag{6}$$

Both optima are attained by a deterministic policy that uses only the current state, and also the remaining budget in the second case.

For an initial state s_0, let C_π be a policy's total cost and let Z be its terminal state. For any expected-cost budget B that at least one policy can satisfy,

$$\begin{aligned} F_U(s_0, B) &:= \max_{\pi:\, \mathbb{E}_\pi[C_\pi] \le B} \mathbb{E}_\pi[U(Z)] \\ &= \min_{\lambda \ge 0} \{\lambda B + V_{\lambda,U}(s_0)\}, \\ V_{\lambda,U}(z) &= U(z), \\ V_{\lambda,U}(s) &= \max_{u \in \mathcal{U}(s)} \left\{ -\lambda c(s,u) + \sum_{s'} T_u(s' \mid s) V_{\lambda,U}(s') \right\}. \end{aligned} \tag{7}$$

An expected-cost optimum can be obtained by choosing, before decoding, between at most two deterministic state-based policies.

Equation (5) works backward from terminal utilities and selects the action with the best expected continuation value. Equation (6) repeats each state for every possible remaining budget. If the original graph has N states and E nonzero state–action–successor transitions, the two computations require $O(E)$ time and $O(N)$ memory without a budget, and $O((\bar{B}+1)E)$ time and $O((\bar{B}+1)N)$ memory for all integer budgets through a maximum $\bar{B}$.

For expected cost, an occupancy measure $x_{s,u}$ records the probability of reaching state s and taking action u. Its constraints equate the probability entering and leaving each state. They give the exact finite linear program, an optimization with a linear objective and linear constraints, in Appendix B. Equation (7) is an equivalent optimization in which λ is the price assigned to one unit of expected cost. The hard-budget policy guarantees the cap on every generation. The expected-budget solution guarantees average cost and can mix at most two deterministic policies whose expected costs bracket B.

Limited-depth planning extends this control framework to large state graphs. Let π_0 be a fixed policy that depends on state and remaining budget and completes within every budget considered. Let $W_0(s,b)$ be its exact expected utility. For $h \ge 1$, define

$$W_h(s,b) = \max_{u \in \mathcal{U}_b(s)} \sum_{s'} T_u(s' \mid s) W_{h-1}(s', b - c(s,u)), \qquad W_h(z,b) = U(z).$$

The policy associated with W_h plans the next h decisions and then follows π_0. Backward induction gives, pointwise for every (s,b),

$$W_0 \le W_1 \le \cdots \le W_U^*,$$

with equality once h is at least the maximum number of decisions remaining. We call this h-step lookahead. Its remaining planning depth decreases by one after each action. If the budget- and depth-expanded search graph contains N_h states and E_h nonzero transitions, storing and reusing each computed value evaluates every transition once. This uses $O(E_h)$ time and $O(N_h)$ memory, excluding computation of W_0.

The next result isolates the information available to the marginal-only policy class. Confidence ordering belongs to this class because its next action depends on the current per-position token probabilities. Richer controllers can additionally use how predictions respond to earlier reveals.

Corollary 1 (Value of response information for reveal control)**.** *For every $0 < \epsilon < 1/2$, two strictly positive binary conditional fields can have the same labeled current marginals and terminal utility but opposite unique optimal first actions. A response-aware optimal policy then has a worst-field expected-utility advantage of at least $(1-2\epsilon)/2$ over any rule whose first-action distribution uses only those marginals.*

The construction scores whether two binary outputs agree, so terminal utility lies in $\{0,1\}$. The two fields expose the same labeled current marginals, 0.8 and 0.7, yet have opposite optimal first reveals. They differ in each position's second-step response, a quantity absent from the current marginals. Appendix B gives the construction and proof. Consequently, response-aware control gains $1-2\epsilon$ over deterministic confidence in one field and at least half that value over the best randomized

marginal-only guarantee. The result pinpoints the value of response information: lookahead and revealed history enlarge the information available to a reveal controller. Section 3.2 complements this information result by evaluating a strong marginal-only portfolio on trained models.

Finite-state validation. A preregistered benchmark validates exact control and limited-depth approximations when the complete transition graph and terminal utility are known. With d tokens per position and a partial assignment in state, $N \leq (1+d)^L$, with equality when every partial assignment can occur. The implementation therefore provides an exact solver benchmark and a controlled test of limited-depth planning. The primary solver shares confidence's feasible width sequence, isolating reveal order, and its transition count measures planning rather than generation cost. Appendix C gives the construction, results, and planner costs. We additionally measure transition kernels from Dream-7B and LLaDA-1.5 on three-slot graphs and enumerate all 48 deterministic reveal policies. This yields the exact reachable set $\mathcal{R}(s)$, its diameter, and its answer-wise disagreement on $1{,}140$ validation graphs per model. On these three-slot graphs, the 48 deterministic history-dependent policies exhaust the finite policy class, so their convex hull is the full $\mathcal{R}(s)$, not a restricted-family lower bound. Fixing the first revealed slot still preserves 64.6%–71.5% of the diameter, so most width comes from later history-dependent choices (Appendix C.2).

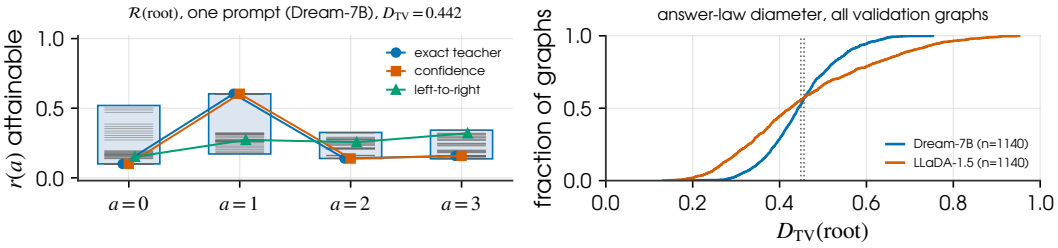


Figure 1: The measured reachable answer-law set on those graphs. *Left:* one validation prompt's $\mathcal{R}(\text{root})$, each box spanning that answer's attainable range over all 48 deterministic reveal policies. *Right:* $D_{\text{TV}}(\text{root})$ over all $1{,}140$ validation graphs of each model, means marked. Appendix C.2 gives the construction, exhaustive policy enumeration, reading rules, and example-selection rule. These exact three-slot measurements connect the reachable-law geometry to kernels measured from trained models.

These measured polytopes establish policy-dependent answer-law width in trained-model kernels. Section 3 next asks whether practical policies convert that freedom into task utility at full output length.

3 Empirical Consequences of Policy-Selected Answer Laws

Section 2 shows that reveal policy can change answer utility and that repeated completions under a fixed future policy estimate that policy's answer distribution. We evaluate these claims separately. We then test whether an answer selected before the surrounding text is complete can be returned at lower cost than full decoding. Appendix C gives the complete protocols and diagnostics.

3.1 Reveal Policy Changes Task Accuracy

We begin with GSM8K (Cobbe et al., 2021) and match the model-evaluation budget across reveal policies. A response is decoded in blocks of B output positions, where B is the number of positions eligible for the current reveal decision. At each model evaluation, the model predicts tokens for every still-masked position in the current block. The decoder reveals exactly one predicted token before evaluating the model again. Every 256-token response therefore uses exactly 256.0 model evaluations for every value of B.

At $B = 1$, only the next left-to-right position is eligible. For larger blocks, the confidence policy reveals the position whose most likely token has the highest predicted probability. The random policy instead samples one of the same masked positions uniformly. Both policies write the most

likely token at the chosen position. We compare LLaDA-8B-Instruct (Nie et al., 2025), LLaDA-1.5 (Zhu et al., 2025a), MMaDA-8B (Yang et al., 2025), and LLaDA-MoE-7B-A1B (Zhu et al., 2025b). Within each model, every policy is scored on the same prompts, and accuracy uses the all-prompt denominator with zero credit when no answer can be extracted.

At the common $B = 32$ setting, confidence improves accuracy over random reveal by 6.0–15.7 percentage points across all four models. The $B = 1$ control confirms that the two policies produce identical outputs when the decoder has no reveal-order choice. At $B = 128$, confidence retains a 4.8–19.6-point accuracy advantage over random across all four models.

The block axis therefore reveals both substantial policy sensitivity and useful control: confidence converts the same model predictions and evaluation budget into consistently stronger accuracy across reveal widths. Because $B = 1$ fixes left-to-right order, the final column directly compares confidence at $B = 32$ with left-to-right decoding. Confidence matches or exceeds it in all four models and gains 3.4 and 5.4 percentage points in LLaDA-1.5 and MMaDA-8B, with intervals excluding zero. The resulting block curves show that block width and reveal policy are complementary decoding choices, with $B = 32$ providing a strong common operating point.

Table 1: Matched-budget reveal-policy results on GSM8K. Accuracy columns list B=1/32/128. In both difference columns a positive entry favors the first-named setting. The policy gain is confidence minus random at $B = 32$. The final column is confidence at $B = 32$ minus confidence at $B = 1$, so a positive entry means the larger block is the more accurate one. Brackets are 95% confidence intervals. For every model, the random B=32 entry and policy-gain interval average ten independent random-order draws and resample both prompts and draws. Daggers mark final-column intervals that exclude zero.

model	n	random B=1/32/128	confidence B=1/32/128	policy gain at $B = 32$	confidence $B = 32 - 1$
LLaDA-8B	1000	.775/.713/.612	.775/.773/.758	+.0604 [+.0399, +.0811]	−.002 [−.024, +.020]
LLaDA-1.5	500	.776/.727/.618	.776/.810/.798	+.0830 [+.0538, +.1122]	+.034† [+.002, +.066]
MMaDA-8B	500	.484/.381/.300	.484/.538/.348	+.1572 [+.1216, +.1932]	+.054† [+.012, +.096]
LLaDA-MoE	500	.778/.706/.568	.778/.780/.764	+.0740 [+.0464, +.1018]	+.002 [−.028, +.032]

Across all four models, resampling prompts and ten independent reveal-order draws gives a confidence-minus-random interval entirely above zero. The gains range from +0.0604 on LLaDA-8B to +0.1572 on MMaDA-8B and are positive under every model's draw distribution (Appendix C).

3.2 A Fixed Portfolio Improves on Confidence Batching

Section 3.1 shows that confidence is a strong trained-model baseline. We test whether complementary reveal scores can improve that baseline without additional model evaluations, using a portfolio that allots confidence, margin, entropy, and a fixed left-to-right order across parallel paths. These scores can select different positions from the same partial state, so the portfolio diversifies causal reveal trajectories rather than resampling one policy.

We evaluate Dream-7B (Ye et al., 2025) on all 1319 GSM8K test prompts. Every arm spends exactly 256 sequence model evaluations per prompt and replicate, and results average over three frozen replicate groups. One arm runs one deep confidence path of 256 actions. The other three run four paths of 64 actions in one batch and use confidence throughout, independent keyed random orders, or a portfolio that gives each of the four paths a different policy: confidence, margin, entropy, and left-to-right (Appendix C.5). All scores share the same forward pass. Four-path arms vote by numeric equality, distinct sentinels prevent parser failures from agreeing spuriously, and tied winners share credit. The portfolio is a matched-compute ensemble decoder: each path follows an admissible reveal policy, and all four-path arms use the same plurality rule. Section 2's identification objects therefore apply to the individual paths, while the arm-level comparison evaluates the resulting voted answers.

Table 2: Dream-7B GSM8K accuracy at 256 sequence model evaluations per prompt and replicate, averaged over three replicate groups, all 1319 prompts. Parse failure is the per-path rate at which no answer can be extracted.

arm	reveal policies	paths × actions	accuracy	parse fail
deep confidence	confidence	1×256	.5037	.1145
confidence batch	confidence at four paths	4×64	.5811	.0063
random batch	random at four paths	4×64	.4335	.0006
portfolio	confidence, margin, entropy, left-to-right	4×64	.6476	.0150

Two contrasts were fixed in advance, with a Holm correction over the pair and paired 20,000-replicate prompt bootstrap intervals. The portfolio exceeds four-path confidence by .0665, interval $(.0517, .0815)$, and the one deep confidence path by .1439, interval $(.1201, .1683)$. Both adjusted p-values are .0002, with positive effects in all three replicate groups. Appendix C.5 reports the corresponding parseability and random-diversity analyses. The portfolio-over-confidence result is stable under the registered parseability sensitivity.

A preregistered LLaDA-1.5 replication gives portfolio accuracy .5776 against .5215 for confidence. The gain is .0561, interval $(.0418, .0711)$, $p = .00010$, and is positive in all three replicates. The parseable-only gain is .0517. A separately preregistered re-analysis that rejects punctuation-only marker matches while retaining the last-number fallback also gives $+.0750$, interval $[+.0598, +.0900]$, $p < 10^{-4}$ (Appendix Table 8). Together, the Dream-7B result and preregistered LLaDA-1.5 replication show that combining complementary reveal policies can improve over confidence-only batching at the same sequence-evaluation budget. Appendix C.5 reports the broader arm comparisons and protocol-specific sensitivities.

3.3 Threshold-Selected States Induce Concentrated Answer Laws

MMLU (Hendrycks et al., 2021) has four answer options, so the answer distribution from a partial generation can be enumerated directly. At each model evaluation we normalize the four option-letter logits at the designated answer position and use this score to select a branching state. For each prompt we save the first partial state where one option receives more than 65% of this mass while the answer position is still masked. We include a model when its highest-scoring option after full generation agrees with the extracted answer on at least 90% of parseable prompts.

The analysis starts from 1000 prompts per model, is conditional on those reaching this event, and reports their number as n in Table 10. From each saved state we generate 32 continuations under greedy token prediction and independently randomized future reveal order, plus a held-out 33rd. Here $\widehat{q}$ is the fraction of the 32 that end in the option leading at the saved state.

Across four models $\widehat{q}$ ranges from 0.956 to 0.992, all 32 continuations return the same answer in 91.6%–98.0% of saved states, and the separate 33rd continuation confirms the option that led at the saved state in 95.58%–99.44% of cases. Agreement also exceeds the benchmark implied by the saved-state option mass. At the saved state the leading option holds a mean of 0.839–0.881 of the four-option mass. Using the saved-state leading mass as the calibrated-posterior benchmark, $\widehat{q}$ exceeds it by 0.093–0.119 in every model, with positive paired-bootstrap intervals and Holm-adjusted $p < 10^{-40}$ (Appendix C.6). This experiment characterizes the answer law induced by randomized future reveal order from threshold-selected states. The event occurs on 537–631 of 1000 prompts per model, and the high continuation agreement is consistent across all four admitted models. The exact three-slot measurements in Appendix C.2 provide a complementary full reachable-set calculation. The two measurements describe different control regimes: the three-slot result is the full diameter over all policies from an all-masked root, whereas this continuation law is measured after selecting a deep-decoding state with a dominant option. Wide initial policy freedom and concentrated threshold-selected laws can therefore coexist. Together, these measurements show that wide root-level policy freedom can coexist with highly stable selected-state answer laws later in decoding.

Returning the partial answer. A prospective MMLU test on 4250 disjoint prompts per model turns the same threshold into a stopping rule with full-decoding fallback. On LLaDA-8B and MMaDA-8B, it reduces mean model evaluations by 53.1% and 53.9% and synchronized batch-eight makespan by 47.1% and 48.6%, while both models remain within the preregistered one-point accuracy tolerance

relative to full decoding (Appendix C). This converts selected-state concentration into an inference-time efficiency gain without changing model weights. On ARC-Challenge (Clark et al., 2018), a retrospective LLaDA-8B control saves 57% of evaluations at unchanged accuracy (Appendix C).

Together, the studies establish four empirical consequences of reveal control: policy changes task utility at matched compute, complementary policies improve confidence-only batching, threshold-selected partial states induce highly concentrated answer laws across randomized future reveal orders, and those states can support substantial stopping savings.

4 Related Work

Masked diffusion language models combine absorbing discrete diffusion (Austin et al., 2021, Sahoo et al., 2024) with blockwise autoregression (Arriola et al., 2025). Most theory studies approximation, convergence, noise, or reveal schedules (Campbell et al., 2022, Lou et al., 2024, Zhang et al., 2025, Chen & Ying, 2024, Liang et al., 2025, Li & Cai, 2025, Chen et al., 2025). Our starting point is conditional compatibility: incompatible conditionals make their causal composition policy-dependent (Arnold & Press, 1989, Heckerman et al., 2000, Wang & Cho, 2019). Generation-order work averages permutations (Hoogeboom et al., 2022), compares insertion orders (Chan et al., 2020), or establishes order-dependent sequence laws (Turok et al., 2026). Those works analyze a chosen order, an average over orders, or the sequence law produced by one rule. Our contribution characterizes the complete answer-level set reachable under causal randomized policies, which separates policy effects from sampling variation and correctness and states what one or several tested policies can identify. The answer map also treats textually different completions as equivalent when the task extractor assigns the same answer.

Practical decoders reveal confident positions or remask uncertain ones (Ghazvininejad et al., 2019, Nie et al., 2025). Related methods rank margins or learned scores (Zheng et al., 2024, Kim et al., 2025, Asano et al., 2026), analyze generation-order effects (Zhang et al., 2026), optimize reward-directed transitions (Bu et al., 2026), or optimize policies and schedules (Hong et al., 2026, Chen et al., 2025, Wainwright, 2026). Early-answer and parallel-decoding methods instead reduce cost by changing when or how many tokens are committed (Li et al., 2025, Wu et al., 2025, Israel et al., 2025). Our prospective stopping test complements this line by combining threshold selection with full-decoding fallback and reporting paired accuracy and synchronized latency. The main reveal-policy comparison instead holds the number of evaluations fixed to isolate where the next reveal occurs. The two protocols therefore separate where-to-reveal effects from when-to-stop savings. We use that set's support functions for exact utility and budgeted control. This set-level view complements single-policy optimization by characterizing what a finite tested family identifies. The control program uses standard constrained Markov decision process (MDP) machinery (Altman, 1999, Kaelbling et al., 1998). We pair its exact finite-graph guarantees with matched-compute policy tests at trained-model scale.

5 Conclusion

Reveal policy is a defining component of a diffusion-language-model decoder. We characterize the complete polytope of answer laws reachable under causal randomized policies and derive exact support-function, Bellman, and occupancy programs for utility and budgeted control on finite transition graphs. Our value-of-response-information result identifies revealed history and lookahead as valuable additional control signals, and our trained-kernel graphs connect this geometry to measured model transitions. The trained-model experiments show the practical consequences at matched compute. Confidence ordering improves over random reveal across four models, a complementary four-policy portfolio improves over confidence batching on Dream-7B and LLaDA-1.5, and threshold-selected MMLU states produce highly concentrated answer laws under randomized future reveal order. A prospective two-model test converts that concentration into roughly half as many model evaluations while preserving the preregistered accuracy tolerance. These results make reveal policy, batching, aggregation, and evaluation budget explicit parts of decoder design. The reachable-law framework provides the corresponding language for comparing those choices and optimizing them toward task utility.

A AI ASSISTANCE

In this work, we used generative AI tools for polishing the writing and implementing methods. We have not used generative AI tools for generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, propose or refine hypotheses, designing or providing feedback on research methodology or experiments, assisting with translation, cleaning and reformatting dataset, supporting qualitative and thematic data analysis, interpret results. We have reviewed all AI-assisted work. For example we read everything written and checked the code written by AI. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Eitan Altman. *Constrained Markov Decision Processes*. Stochastic Modeling Series. Chapman & Hall/CRC, 1st edition, 1999.

Barry C. Arnold and S. James Press. Compatible conditional distributions. *Journal of the American Statistical Association*, 84(405):152–156, 1989.

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

Hikaru Asano, Tadashi Kozuno, Kuniaki Saito, and Yukino Baba. Where-to-unmask: Ground-truth-guided unmasking order learning for masked diffusion language models. *arXiv preprint arXiv:2602.09501*, 2026.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In *Advances in Neural Information Processing Systems*, pp. 17981–17993, 2021.

Heli Ben-Hamu, Itai Gat, Daniel Severo, Niklas Nolte, and Brian Karrer. Accelerated sampling from masked diffusion models via entropy bounded unmasking. *arXiv preprint arXiv:2505.24857*, 2025.

Dake Bu, Wei Huang, Andi Han, Si Wu, Hau-San Wong, Qingfu Zhang, Taiji Suzuki, and Atsushi Nitanda. DPRM: A plug-in Doob h transform-induced token-ordering module for discrete diffusion models. *arXiv preprint arXiv:2604.24357*, 2026.

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

William Chan, Mitchell Stern, Jamie Kiros, and Jakob Uszkoreit. An empirical study of generation order for machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5764–5773, Online, November 2020. Association for Computational Linguistics.

Hongrui Chen and Lexing Ying. Convergence analysis of discrete diffusion model: exact implementation through uniformization. *arXiv preprint arXiv:2402.08095*, 2024.

Sitan Chen, Kevin Cong, and Jerry Li. Optimal inference schedules for masked diffusion models. *arXiv preprint arXiv:2511.04647*, 2025.

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, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. Mask-predict: Parallel decoding of conditional masked language models. In *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.

Satoshi Hayakawa, Yuhta Takida, Masaaki Imaizumi, Hiromi Wakaki, and Yuki Mitsufuji. Demystifying MaskGIT sampler and beyond: Adaptive order selection in masked diffusion. *Transactions on Machine Learning Research*, 2026.

David Heckerman, David Maxwell Chickering, Christopher Meek, Robert Rounthwaite, and Carl Kadie. Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, 1(Oct):49–75, 2000.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021.

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

Emiel Hoogeboom, Alexey A. Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022.

Daniel Israel, Guy Van den Broeck, and Aditya Grover. Accelerating diffusion LLMs via adaptive parallel decoding. *arXiv preprint arXiv:2506.00413*, 2025.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998.

Jaeyeon Kim, Kulin Shah, Vasilis Kontonis, Sham M. Kakade, and Sitan Chen. Train for the worst, plan for the best: Understanding token ordering in masked diffusions. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 30749–30768. PMLR, 2025.

Gen Li and Changxiao Cai. Breaking AR's sampling bottleneck: Provable acceleration via diffusion language models. In *Advances in Neural Information Processing Systems*, pp. 11700–11725, 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.

Yuchen Liang, Renxiang Huang, Lifeng Lai, Ness Shroff, and Yingbin Liang. Absorb and converge: provable convergence guarantee for absorbing discrete diffusion models. *arXiv preprint arXiv:2506.02318*, 2025.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32819–32848. PMLR, 2024.

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

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.

Gilad Turok, Chris De Sa, and Volodymyr Kuleshov. DUEL: Exact likelihood for masked diffusion via deterministic unmasking. *arXiv preprint arXiv:2603.01367*, 2026.

Martin J. Wainwright. The data geometry of masking diffusion: Certified-optimal schedules via unmasking growth complexity. *arXiv preprint arXiv:2608.13520*, 2026.

Alex Wang and Kyunghyun Cho. BERT has a mouth, and it must speak: BERT as a Markov random field language model. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pp. 30–36, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

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

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dLLM: Training-free acceleration of diffusion LLM by enabling KV cache and parallel decoding. *arXiv preprint arXiv:2505.22618*, 2025.

Ling Yang, Ye Tian, Bowen Li, Xinchen Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. MMaDA: Multimodal large diffusion language models. In *Advances in Neural Information Processing Systems*, pp. 138867–138907, 2025.

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

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

Zikun Zhang, Zixiang Chen, and Quanquan Gu. Convergence of score-based discrete diffusion models: a discrete-time analysis. In *International Conference on Learning Representations (ICLR)*, 2025.

Lin Zheng, Jianbo Yuan, Lei Yu, and Lingpeng Kong. A reparameterized discrete diffusion model for text generation. In *Conference on Language Modeling (COLM)*, 2024.

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

Fengqi Zhu, Zebin You, Yipeng Xing, Zenan Huang, Lin Liu, Yihong Zhuang, Guoshan Lu, Kangyu Wang, Xudong Wang, Lanning Wei, Hongrui Guo, Jiaqi Hu, Wentao Ye, Tieyuan Chen, Chenchen Li, Chengfu Tang, Haibo Feng, Jun Hu, Jun Zhou, Xiaolu Zhang, Zhenzhong Lan, Junbo Zhao, Da Zheng, Chongxuan Li, Jianguo Li, and Ji-Rong Wen. LLaDA-MoE: A sparse MoE diffusion language model. *arXiv preprint arXiv:2509.24389*, 2025b.

B Proofs and Extensions for Reachable Answer Laws

Reachable-law proof. We first prove Theorem 1 and its stated consequences. The proof records exactly where random policy choices and outcome-dependent continuation policies generate the convex recursion.

Proof of Theorem 1. Induct on the number $m(s)$ of unrevealed positions. If $m(s) = 0$, the completed sequence is x, no decision remains, and its answer law is the point mass $\mathbf{e}_{A(x)}$. This proves (1).

Suppose the claim holds below horizon m, and let $m(s) = m > 0$. Fix an admissible policy π. At s, its independent random draw selects a first position $i \in \mathcal{I}_{\mathrm{v}}(s)$ with probabilities α_i. Conditional on selecting i and observing $y \in \mathcal{X}_i^+(s)$, the policy subtree is an admissible reveal policy from $s_+(i, y)$. Write its answer law as $r_{i,y} \in \mathcal{R}(s_+(i, y))$. The law of total probability gives

$$\mathrm{Law}_\pi(A(X) \mid s) = \sum_i \alpha_i \sum_{y \in \mathcal{X}_i^+(s)} K_i(y \mid s) r_{i,y}.$$

It therefore belongs to the right-hand side of (2).

Conversely, take any point on that right-hand side. By the definition of a convex hull, it is a finite mixture of vectors $q_\ell = \sum_{y \in \mathcal{X}_{i_\ell}^+(s)} K_{i_\ell}(y \mid s) r_{\ell,y}$, where $r_{\ell,y} \in \mathcal{R}(s_+(i_\ell, y))$. Choose a reveal policy attaining each $r_{\ell,y}$, randomly choose the mixture component ℓ, reveal i_ℓ, and after observing y continue with the chosen branch policy. Closure under random policy choices and outcome-dependent continuation policies makes this policy admissible, and its terminal answer law is the desired mixture. This proves the reverse inclusion and hence (2).

At a viable state, at least one retained action has only viable positive-probability successors, so induction also gives nonemptiness. For the polytope claim, the terminal set is a singleton polytope. Inductively, for fixed i, the set inside braces in (2) is a weighted Minkowski sum of finitely many positive-support child polytopes and is therefore a polytope. The convex hull of their finite union is again a convex polytope. □

Support functions and disagreement. For a terminal state, $h_x(v) = \langle v, \mathbf{e}_{A(x)} \rangle = v_{A(x)}$. A linear functional has the same maximum over a set and its convex hull. Applying this fact to (2), then optimizing the branch choices r_y independently, gives

$$\begin{aligned} h_s(v) &= \max_{i \in \mathcal{I}_{\mathrm{v}}(s)} \max_{\{r_y\}} \sum_{y \in \mathcal{X}_i^+(s)} K_i(y \mid s) \langle v, r_y \rangle \\ &= \max_{i \in \mathcal{I}_{\mathrm{v}}(s)} \sum_{y \in \mathcal{X}_i^+(s)} K_i(y \mid s) h_{s_+(i,y)}(v), \end{aligned}$$

This is the single-position specialization of (5). The general finite-action form replaces i by u, y by s', and K_i by T_u. Maxima exist because the reachable sets are compact polytopes.

Unconstrained optimal control. The same induction applies to a general terminal utility U. Consider any observed history ending at state s, and let $q(i)$ be the policy's conditional distribution over eligible next positions. By the induction hypothesis, its conditional value is at most

$$\sum_i q(i) \sum_y K_i(y \mid s) V_U^*(s_+(i, y)) = \sum_i q(i) Q_U^*(s, i) \le \max_i Q_U^*(s, i) = V_U^*(s).$$

A state-action value here is $Q_U^*(s, i) = \sum_y K_i(y \mid s) V_U^*(s_+(i, y))$. A fixed maximizing action followed by maximizing child policies attains equality. Thus previous history and independent randomization only form convex mixtures of the same state-action values. They cannot exceed the maximum. Finite action sets make the argmax nonempty, and any fixed tie rule gives one deterministic state-Markov optimizer. On the explicit reachable-state graph, each transition is evaluated once and each state stores one value and action, giving the stated $O(E)$ time and $O(N)$ memory bounds.

Cost-constrained extension. For Theorem 2, enlarge the model to finite decision states $\mathcal{D}$ and disjoint terminal states $\mathcal{Z}$ on a finite directed acyclic graph (DAG). Each terminal state z has answer $A(z)$ and bounded utility $U(z)$. Each action $u \in \mathcal{U}(s)$ has a known kernel $T_u(s' \mid s)$ and integer model-evaluation cost $c(s,u) \geq 0$. A multi-position action uses its specified joint transition kernel directly. For hard-budget control, $\mathcal{U}_b(s)$ contains the actions that cost at most b and, after every positive-probability successor s', leave some policy that completes with realized future cost at most $b - c(s,u)$. The theorem considers augmented states (s,b) for which this set is nonempty. Let $N = |\mathcal{D} \cup \mathcal{Z}|$, and let E count the nonzero state–action–successor arcs.

The answer-law–cost set has the recursion

$$\mathcal{P}(z) = \{(\mathbf{e}_{A(z)}, 0)\}, \tag{8}$$

$$\mathcal{P}(s) = \operatorname{conv} \bigcup_{u \in \mathcal{U}(s)} \left\{ \left(\sum_{s'} T_u(s' \mid s) r_{s'}, c(s,u) + \sum_{s'} T_u(s' \mid s) C_{s'} \right) : (r_{s'}, C_{s'}) \in \mathcal{P}(s') \right\}. \tag{9}$$

The same backward induction as Theorem 1 proves this identity. It starts from one terminal pair, forms finite weighted Minkowski sums at each action, and takes their finite convex hull. Thus $\mathcal{P}(s) \subseteq \Delta(\mathcal{A}) \times \mathbb{R}_+$ is a compact convex polytope.

For the pathwise claim, induct in reverse DAG order and over remaining budget. Any feasible first action $u \in \mathcal{U}_b(s)$ has $c(s,u) \leq b$, and after its specified transition the child policy has budget $b - c(s,u)$. The induction hypothesis gives Equation (6). A maximizing action followed by maximizing child policies attains its value. Feasibility makes the maximization nonempty at every augmented decision state under consideration. The augmented graph has at most $(\bar{B}+1)N$ states and $(\bar{B}+1)E$ transitions. The corresponding bounds are $O((\bar{B}+1)E)$ time and $O((\bar{B}+1)N)$ memory. They are pseudo-polynomial when $\bar{B}$ is binary encoded.

For a feasible expected-cost budget, let $x_{s,u}$ be the expected number of visits to (s,u) and let q_z be the terminal mass at z. The exact occupancy program is

$$\max_{x,q \geq 0} \quad \sum_{z \in \mathcal{Z}} q_z U(z) \tag{10}$$

$$\text{subject to} \quad \sum_{u \in \mathcal{U}(s)} x_{s,u} - \sum_{w \in \mathcal{D}} \sum_{u \in \mathcal{U}(w)} x_{w,u} T_u(s \mid w) = \mathbf{1}\{s = s_0\}, \qquad s \in \mathcal{D}, \tag{11}$$

$$q_z = \sum_{w \in \mathcal{D}} \sum_{u \in \mathcal{U}(w)} x_{w,u} T_u(z \mid w), \tag{12}$$

$$\sum_{s \in \mathcal{D}} \sum_{u \in \mathcal{U}(s)} c(s,u) x_{s,u} \leq B. \tag{13}$$

Every causal policy induces a feasible occupancy. Conversely, normalizing the outgoing occupancy at each visited state defines a randomized Markov policy with the same flow. Induction through the DAG proves equality of its occupancy.

Strong finite-dimensional linear-program duality gives Equation (7). For fixed λ, reverse topological order gives the least feasible dual value through the displayed Bellman recursion. The flow polytope defined by the conservation constraints in Eqs. (11)–(12) alone, before imposing the cost cap in Eq. (13), is the convex hull of the finitely many deterministic Markov policy occupancies (Altman, 1999). The cost-capped polytope itself need not have deterministic vertices, which is exactly why the two-policy mixture below can be required. To prove the vertex characterization, let x be feasible for the conservation constraints with $x_{s,u_1} > 0$ and $x_{s,u_2} > 0$ for two distinct actions at some state s. If a successor s' of s had zero occupancy, then $T_{u_1}(s' \mid s) = T_{u_2}(s' \mid s) = 0$, because otherwise s' would receive positive inflow from x_{s,u_1} or x_{s,u_2}; so every successor whose inflow the following perturbation changes already has positive occupancy. Shift $\pm\tau$ of mass between u_1 and u_2 and propagate the resulting inflow change downstream in DAG order, splitting the change at each affected state in proportion to that state's existing positive outgoing shares, and into q_z at affected terminal states. Each perturbed coordinate is a fixed multiple of a coordinate that is already positive, so for all small $\tau > 0$ both perturbations remain nonnegative and satisfy the same conservation constraints, and x is their midpoint. Hence x is not a vertex, so every vertex uses one action at each visited state,

that is, it is a deterministic Markov occupancy. The flow polytope is bounded on a finite DAG, so it equals the convex hull of these finitely many vertices. Reducing to their cost–utility pairs leaves one normalization equality and one cost constraint. An optimal basic feasible solution therefore uses one policy when the cost constraint is slack and at most two when it binds. Both components maximize the same cost-penalized objective at an optimal multiplier.

For countdown lookahead, fix a state-budget Markov baseline π_0 that is defined and feasible from every reachable (s, b). Its exact budget-indexed value table W_0 is assumed available or precomputed. Let $\mathcal{T}_B$ denote the maximization on the right-hand side of Equation (6), acting on budget-indexed values, and define $W_h = \mathcal{T}_B W_{h-1}$. Induction on h proves that W_h is the optimum over policies that control the next h decisions and then follow π_0. At every (s, b), the baseline action distribution is supported on feasible actions. The maximization therefore dominates its baseline convex combination, so $\mathcal{T}_B W_0 \geq W_0$. Monotonicity of $\mathcal{T}_B$ yields $W_{h+1} \geq W_h$. The recursion equals exact budgeted Bellman once h reaches the remaining DAG depth.

Let N_h and E_h count the nodes and nonzero arcs in the budget-augmented depth-h lookahead graph. With constant-time lookup of W_0, memoization uses $O(E_h)$ time and $O(N_h)$ memory. These bounds exclude construction and storage of W_0. Let $\widetilde{W}_0$ be a supplied leaf approximation with $\|\widetilde{W}_0 - W_0\|_\infty \leq \eta$, and let $\widetilde{\pi}_h$ be the depth-h policy obtained by backing up $\widetilde{W}_0$ and taking the resulting maximizing actions. Bellman nonexpansiveness gives $\|\mathcal{T}_B^h \widetilde{W}_0 - W_h\|_\infty \leq \eta$. For any fixed depth-$h$ policy, replacing its exact leaf values by $\widetilde{W}_0$ changes its root value by at most η. Comparing the approximate values of $\widetilde{\pi}_h$ and the exact restricted optimizer therefore yields

$$0 \leq W_h(s, b) - J_h(\widetilde{\pi}_h; s, b, W_0) \leq 2\eta,$$

where $J_h(\pi; s, b, W_0)$ is the value of controlling the next h decisions with π and then following π_0. Thus the *depth-h restricted-class regret* of the induced countdown policy is at most 2η; the bound does not compare against policies that control more than h decisions. The leaf-value tolerance η is a separate quantity from the level ϵ used in Section 2 and in the corollary proof below.

We now prove Corollary 1, stated in Section 2.

Proof of Corollary 1. Use two positions, $U(x_1, x_2) = \mathbf{1}\{x_1 = x_2\}$, and in both fields set $K_1(1 \mid \varnothing) = 0.8$ and $K_2(1 \mid \varnothing) = 0.3$. In field A, after position 1 is revealed first, position 2 copies it with probability $1 - \epsilon$. After position 2 is first, position 1 anti-copies it with that probability. Field B swaps these two continuation rules. Thus A's first-action values are $(1 - \epsilon, \epsilon)$, B's are $(\epsilon, 1 - \epsilon)$, and their unique maximizers are opposite. Current confidence is 0.8 at position 1 and 0.7 at position 2, so deterministic confidence strictly chooses position 1 and incurs regret $1 - 2\epsilon$ in field B. A marginal-only rule must choose position 1 with the same probability ρ in both fields. Its regrets are $(1 - 2\epsilon)(1 - \rho)$ and $(1 - 2\epsilon)\rho$. Their maximum is at least $(1 - 2\epsilon)/2$. □

For distributions on a finite set,

$$\tfrac{1}{2}\|r - r'\|_1 = \max_{C \subseteq \mathcal{A}} \{r(C) - r'(C)\}.$$

The collection of subsets is finite, so maximizing over r, r' and C may be reordered. The two distributional optimizations separate:

$$\begin{aligned} D_{\mathrm{TV}}(s) &= \max_{C \subseteq \mathcal{A}} \left\{ \max_{r \in \mathcal{R}(s)} \langle \mathbf{1}_C, r \rangle + \max_{r' \in \mathcal{R}(s)} \langle -\mathbf{1}_C, r' \rangle \right\} \\ &= \max_{C \subseteq \mathcal{A}} \{h_s(\mathbf{1}_C) + h_s(-\mathbf{1}_C)\}, \end{aligned}$$

proving the support-function representation of D_{TV} displayed in Section 2. Likewise,

$$\sup_{r \in \mathcal{R}(s)} (1 - r(a)) = 1 - \inf_{r \in \mathcal{R}(s)} r(a) = 1 + h_s(-\mathbf{1}_{\{a\}}),$$

which proves (4). Finally, all admissible reveal policies induce the same answer law exactly when $\mathcal{R}(s)$ is a singleton. A nonempty set has total-variation diameter zero exactly when all of its elements coincide. This establishes the equivalence with answer-law policy invariance, not with compatibility of the reveal kernels. No step requires the common law to be a point mass. A singleton $\mathcal{R}(s) = \{q\}$ may have $q_a < 1$ for every answer a.

Table 3: What can be learned from repeated generations at a fixed partial state. Every statement is scoped to the specified token kernels and policy family.

Observed data	Identified quantity or bound
Independent completed generations under one fixed policy π	Estimate $r_{s,\pi}$, including its probability for each answer.
Two or finitely many sampled policies	The population pairwise TV distances, and their maximum, lower-bound $D_{\mathrm{TV}}(s)$. Empirical distances converge to these quantities, while finite samples require uncertainty intervals.
All admissible policies or exact optimization with known kernels	Determine $\delta_a(s)$, the largest probability any policy assigns away from a.
External reference answers	Evaluate correctness for the sampled laws. Claims about every policy still depend on policy coverage.

Common-joint sufficient condition. Suppose there is a joint distribution Q on completed sequences and a positive-probability start state. Call a state reachable when it has positive probability under some admissible policy from that start. Suppose that for every such s, every eligible i, and every $y \in \mathcal{X}_i^+(s)$,

$$K_i(y \mid s) = Q(X_i = y \mid s), \qquad Q(\cdot \mid s_+(i,y)) = Q(\cdot \mid s, X_i = y).$$

For any completion distribution P, write $A_{\#}P$ for the answer distribution obtained by applying the extractor A. Define $q_s = A_{\#}Q(\cdot \mid s)$. At a terminal completion x, $q_x = \mathbf{e}_{A(x)}$. If each positive-support child satisfies $\mathcal{R}(s_+(i,y)) = \{q_{s_+(i,y)}\}$, then for every first action i,

$$\sum_{y \in \mathcal{X}_i^+(s)} K_i(y \mid s) q_{s_+(i,y)} = \sum_{y \in \mathcal{X}_i^+(s)} Q(X_i = y \mid s) A_{\#}Q(\cdot \mid s, X_i = y) = A_{\#}Q(\cdot \mid s) = q_s.$$

Backward induction in (2) therefore yields $\mathcal{R}(s) = \{A_{\#}Q(\cdot \mid s)\}$ at every reachable state. This is the usual deferred-decisions implication from a common joint law; in masked diffusion the same implication appears as Hayakawa et al. (2026, Prop. 3) for exact conditionals with single-position, temperature-free reveals, and in the zero-model-error exactness statement for parallel reveals of Ben-Hamu et al. (2025).

The implication is only sufficient after applying A. Distinct policies may induce different completion laws that have the same pushforward through A. In the extreme, a constant extractor makes $\mathcal{R}(s)$ a singleton for arbitrary, potentially incompatible reveal kernels. Answer-law policy invariance therefore has no converse to kernel compatibility without further assumptions on A.

C Experimental Protocols and Additional Results

This appendix supplies the details needed to reproduce the finite-state illustration and the three trained-model studies in the main paper. The finite-state benchmark validates optimization when the transition graph and terminal utility are known. One trained-model study below runs that planner on a deliberately small graph whose kernels are measured from a trained model; the remaining trained-model studies do not use it. They test policy-dependent task utility, estimate one policy-selected answer law, and compare returning a partial answer with full decoding.

C.1 Finite-State Optimization Benchmark

The benchmark contains five independently generated groups of 128 finite decoders. Each decoder has six binary output positions. The probability of the token at one position may depend on every token already revealed, so different reveal orders can induce different completion and answer distributions. An action costs one evaluation and reveals one, two, or three positions. Tokens revealed by the same action are sampled independently from their pre-action distributions. Every policy must complete the output under a hard budget in $\{2, 3, 4, 5, 6\}$.

For decoder d in group g, the generator draws $b_i \sim \mathcal{N}(0, 0.8^2)$ and $w_{ij} \sim \mathcal{N}(0, 1)$, with $w_{ii} = 0$, from NumPy `SeedSequence([165100 + 100g, d])`. At partial assignment s, the probability

of token one at masked position i is

$$\mathrm{clip}_{[.02,.98]}\,\sigma\left(b_i + \sum_{j \text{ revealed}} w_{ij}(2s_j - 1)\right).$$

The five groups use $g = 0, \ldots, 4$ and each uses $d = 0, \ldots, 127$.

Each of the 64 completed sequences has a known utility. For each position, we reveal the other five tokens and record the log probability assigned to the completed token. Utility is the mean over positions, rescaled within each decoder so that its smallest and largest completion utilities are zero and one. The confidence baseline reveals the most certain positions subject to leaving enough future actions to finish. One- and two-decision lookahead optimize the next one or two actions before returning to confidence, conditional on the exact confidence value table W_0. Full control uses Theorem 2.

Table 4: Finite-state hard-budget benchmark. Gains are paired differences from confidence in normalized area under the five-budget utility frontier. Brackets are hierarchical 95% bootstrap intervals. Planning transitions count state–action–outcome branches evaluated while solving the control problem, not model evaluations during generation. Lookahead counts exclude construction and storage of W_0.

policy	gain over confidence	planning transitions
one-decision lookahead	$0.058285\ [0.055089, 0.061475]$	2,169
two-decision lookahead	$0.072829\ [0.069477, 0.076223]$	10,426
minimum-width Bellman	$0.074182\ [0.070811, 0.077597]$	2,110
full Bellman control	$0.080498\ [0.077132, 0.083900]$	18,722

Across budgets $B = (2, 3, 4, 5, 6)$, mean normalized utilities are $(.733, .765, .779, .805, .818)$ for confidence, $(.792, .831, .846, .858, .864)$ for one-decision lookahead, $(.792, .855, .867, .873, .876)$ for two-decision lookahead, $(.792, .845, .862, .882, .889)$ for minimum-width Bellman, and $(.792, .855, .877, .887, .890)$ for full control.

The minimum-width row is the preregistered primary arm. It is an exact dynamic program restricted to the same minimum-width action class that the confidence and random baselines use, the class in which the width sequence is pinned by the feasibility rule, so its gain isolates reveal order, and it equals full control at the smallest budget where the two classes coincide. It reaches 92% of full control's gain with $8.9\times$ fewer planning transitions, and its point estimate exceeds two-decision lookahead at one fifth of the transitions; unlike the countdown ladder, however, it has a ceiling, because restricting the action class caps the attainable value, so it cannot converge to the exact value.

Exhaustive enumeration on 16 independent length-three decoders agrees with the dynamic program to numerical precision. Compatible-joint controls have a maximum action-value spread of 4.44×10^{-16}, confirming that the implementation does not create policy value when every reveal order samples one common joint distribution. In the benchmark itself, the state graph, transition kernels, and terminal utilities are all available. The trained-model experiments below do not satisfy those conditions at full output length; the next subsection builds a deliberately small graph on which a trained model does supply them, and the remaining trained-model studies do not execute this planner.

C.2 Exact Reveal Control on Trained-Model Graphs

This study is the trained-model counterpart of the benchmark above. It instantiates Theorem 2 on graphs whose transition kernels are measured from a trained diffusion language model rather than drawn from a generator.

For each MMLU prompt (Hendrycks et al., 2021) we build a three-slot four-option constrained decoder. The three slots are semantic roles, indexed 0 direct, 1 check, and 2 final, and each slot is either masked or holds one of the four option indices, so the graph has $5^3 = 125$ states, 64 terminal and 61 nonterminal. Every nonterminal state costs exactly one model forward. Its aligned logits are softmaxed over only the four fixed option token identifiers, and the resulting four-vector is that state's

exact successor kernel. The all-masked root forward freezes one score vector $v_r \in [0,1]^4$ per role, and the terminal utility is the equal-weight mean

$$U(z) = \big(v_{\text{direct}}[z_{\text{direct}}] + v_{\text{check}}[z_{\text{check}}] + v_{\text{final}}[z_{\text{final}}]\big)/3.$$

The kernels are therefore measured from the trained model's own conditionals rather than assumed, and the exact backward induction of Theorem 2 then runs on the finished graph with no further model evaluations. We use 40 prompts from each of the 57 MMLU test subjects, giving $2{,}280$ graphs per model, split evenly into $1{,}140$ training graphs, which select the best fixed reveal order, and $1{,}140$ validation graphs, which supply every value reported here. A reveal order is written as its slot indices in the order they are revealed, so 012 is left-to-right. No label file was opened at any point: the utility above is label-free multi-role self-consistency, not accuracy.

The construction is non-degenerate on both models: root actions vary across validation prompts, root-action entropy is positive, and no state has a top-two action-value tie within 10^{-9}. The policy values below are reproduced by an independent terminal-law calculation and exhaustive enumeration of all 48 deterministic history-dependent reveal policies.

Table 5: Exact validation-mean utility of every reveal policy on the same graphs, at no model forwards beyond building them. Fixed orders are slot indices in the order revealed, with 0 direct, 1 check, 2 final, so 012 is left-to-right. The best fixed order is selected on the training half and differs between the two models.

policy	reveal rule	Dream-7B	LLaDA-1.5
exact teacher	state-dependent	0.375330	0.415924
best fixed order	201 / 120	0.364911	0.391315
confidence	state-dependent	0.364520	0.383019
random	uniform	0.344259	0.373601
fixed	012 (direct, check, final)	0.310372	0.342581
fixed	021 (direct, final, check)	0.327186	0.358755
fixed	102 (check, direct, final)	0.345190	0.377409
fixed	120 (check, final, direct)	0.356905	0.391315
fixed	201 (final, direct, check)	0.364911	0.381195
fixed	210 (final, check, direct)	0.360992	0.390353

Table 5 reads the same way in both models on two points and differently on two others. Left-to-right is the worst of the six fixed orders in both, and reveal order alone moves the utility by 0.054538 on Dream-7B and 0.048734 on LLaDA-1.5, which is 5.23 and 1.98 times what exact state-dependent control adds on top of the best fixed order. That ordering is, however, largely a property of this utility rather than of decoding quality: grouping the six orders by which slot they reveal first reproduces the ranking on both models, as the decomposition below predicts. What does not carry across is which fixed order is best, 201 on Dream-7B against 120 on LLaDA-1.5, and how close confidence comes to it: confidence is essentially at the best fixed order on Dream-7B, at -0.000391, and clearly below it on LLaDA-1.5, at -0.008296. Two identities implied by the construction hold on the reported values themselves: the exact teacher dominates every baseline, and the exact random policy equals the mean of the six fixed orders to floating point. An independent implementation through a separate evaluation path, an explicit terminal-law expectation in place of the backward recursion, reproduces both on rebuilt validation graphs and matches exhaustive enumeration over all 48 deterministic history-dependent reveal policies. The two identities hold to 1.7×10^{-16} and 4.4×10^{-16} on Dream-7B and to 3.3×10^{-16} and 7.8×10^{-16} on LLaDA-1.5, over the $1{,}140$ validation graphs. Teacher dominance is an inequality rather than an identity, and its margin is nonnegative on every graph, with a minimum of exactly zero and a maximum of 0.037123 on Dream-7B and 0.241073 on LLaDA-1.5.

The reachable answer-law set, measured. These same graphs also make Theorem 1's own objects exactly computable, which is the one setting in this paper where that is true, so we report them. Read the option index in the final slot as the answer, giving the answer alphabet $\mathcal{A} = \{0,1,2,3\}$. Because the decoder observes its full reveal history on an acyclic graph, every randomized admissible policy induces the terminal law of some mixture of deterministic ones, so $\mathcal{R}(s)$ is the convex hull of the 48 deterministic policies' answer laws. Total variation is convex in each argument, so $D_{\mathrm{TV}}(s)$ is attained at a pair of those 48 vertices and needs no search over the hull; $\delta_a(s) = 1 - \min_i r_i(a)$ likewise. We compute every quantity twice by disjoint routes: once by enumerating the vertices, and once from the support-function representation of Section 2, evaluating $h_s(\mathbf{1}_C)$ for all 16 subsets

$C \subseteq \mathcal{A}$ by the backward induction of Theorem 2 on the terminal score $\mathbf{1}_C(A(z))$ and reading $D_{\mathrm{TV}}(s) = \max_C\{h_s(\mathbf{1}_C) + h_s(\mathbf{1}_{\mathcal{A}\setminus C})\} - 1$. The two routes never call each other's machinery and agree to 4.4×10^{-16} on Dream-7B and 5.6×10^{-16} on LLaDA-1.5 at all 61 nonterminal states of all 2,280 graphs per model. This costs no model evaluations beyond building the graphs.

Table 6: Measured reachable answer-law geometry at the all-masked root, over the 1,140 validation graphs per model, reading the final slot as the answer. Distributions are across graphs, and every share is a mean or minimum of per-graph ratios, never a ratio of the reported means. The last row instead takes the answer map to be the identity on completed assignments, so it is the completed-output diameter at the same root state, not a quantity at a terminal state. Nothing here is a gated clause: a set width is descriptive, so no threshold is declared and none is met or missed.

quantity at the root	Dream-7B	LLaDA-1.5
D_{TV}, mean	0.449040	0.456301
D_{TV}, median	0.442466	0.425046
D_{TV}, minimum over graphs	0.186611	0.132889
D_{TV}, maximum over graphs	0.753356	0.951594
graphs with $D_{\mathrm{TV}} = 0$	0	0
δ_a at the teacher's own answer, mean	0.802091	0.806377
$\min_a \delta_a$, mean	0.786598	0.785622
$\max_a \max_r r(a)$, mean	0.623474	0.605025
$D_{\mathrm{TV}}^{\Pi_0}$ over the eight tested policies, mean	0.431765	0.443756
its share of D_{TV}, mean of per-graph ratios	0.959967	0.968343
its share of D_{TV}, smallest per-graph ratio	0.631437	0.572206
D_{TV} within one first-revealed slot, mean	0.282975	0.308618
its share of D_{TV}, mean of per-graph ratios	0.646433	0.714555
mean D_{TV} over the 12 depth-1 states	0.175233	0.174177
maximum D_{TV} over the 48 depth-2 states	0	0
root D_{TV} with $A(z) = z$, all 64 assignments	0.591226	0.682049

Figure 1 draws this measurement, and because it now sits in the body its reading rules are recorded here. In the left panel each box spans $[\min_r r(a), \max_r r(a)]$ over all 48 deterministic reveal policies, so its height is that answer's attainable range and its distance below 1 is $\delta_a(\mathrm{root})$; the grey ticks are the 48 policies themselves, one tick each. The exact teacher, confidence, and left-to-right are drawn as the single laws they are, with a small cosmetic horizontal offset so coinciding policies stay visible. Teacher and confidence nearly coincide, which is the structural property this study discloses below. The prompt is not chosen by eye: it is the validation graph whose diameter is closest to the validation median, ties to the smallest stable identifier. The right panel plots the empirical distribution of $D_{\mathrm{TV}}(\mathrm{root})$ over all 1,140 validation graphs of each model, one step per graph, with each model's mean marked; the two means nearly coincide while the shapes differ.

Table 6 and Figure 1 say that on these graphs reveal policy is not a tie-breaker. The reachable answer-law set is wide on both models, with mean diameter 0.449040 on Dream-7B and 0.456301 on LLaDA-1.5, and not one of the 2,280 validation graphs is degenerate, so Theorem 1's one-point case never occurs here. The disagreement quantity $\delta_a(\mathrm{root})$ of Equation (4) is correspondingly large and uniform. At the exact teacher's own answer it is 0.802091 and 0.806377, and even at the answer that is hardest to move, $\arg\min_a \delta_a$, it is 0.786598 and 0.785622: some admissible policy drives that answer below probability 0.22 on average, so no answer is guaranteed on these graphs. Three further readings matter more than the headline. First, the finite family bound of Section 2 is nearly tight in practice: the eight policies this paper actually evaluates realize 0.959967 and 0.968343 of the full diameter on average, so a tested-family measurement loses little here, although its minimum over graphs falls to 0.631437 and 0.572206, so the inequality is not tight uniformly. Second, the width is not the first-slot artifact that deflates this study's utility ranking. Partitioning the 48 policies by which slot they reveal first and recomputing the diameter inside each group leaves 0.282975 and 0.308618, which is 0.646433 and 0.714555 of the total, so most of the width comes from history-dependent choices made after the first reveal rather than from the $\|v_r\|^2/3$ decomposition discussed below. Third, the geometry closes as the graph fills: the mean diameter falls from 0.449040 at the root to 0.175233 one reveal later, and is exactly 0 two reveals later, where a single policy remains. The two models differ by 2.4 times in adaptive utility headroom but by under two percent in mean answer-law diameter, so the width is the more stable of the two measurements across checkpoints.

The exact root-level geometry and the full-length continuation measurement cover complementary control regimes. The former measures the complete reachable set from an all-masked root on a three-slot graph; the latter measures randomized continuations from deep states selected for a dominant option in a 256-position decode. We therefore report exact diameter for the measured three-slot roots and selected-state concentration for the full-length continuation protocol.

Scope. The three-slot, four-option construction makes exhaustive policy enumeration possible and uses label-free multi-role self-consistency as its utility. It is an exact finite-graph measurement rather than a deployed full-length planner. The full-length experiments separately evaluate task accuracy, policy portfolios, continuation concentration, and stopping efficiency.

C.3 Protocol Coverage

The matched-budget study covers LLaDA-8B, LLaDA-1.5, MMaDA-8B, and LLaDA-MoE. The portfolio study uses independently specified Dream-7B and LLaDA-1.5 runs, and the continuation and stopping protocols use the checkpoints satisfying their stated task-specific inclusion criteria. Each comparison is interpreted within its own prompt set, model revision, answer extractor, and compute budget.

C.4 Matched-Compute Reveal Policies

We evaluate LLaDA-8B-Instruct (Nie et al., 2025), LLaDA-1.5 (Zhu et al., 2025a), MMaDA-8B (Yang et al., 2025), and LLaDA-MoE-7B-A1B (Zhu et al., 2025b) on GSM8K (Cobbe et al., 2021). The numbers of prompts are 1000, 500, 500, and 500. Every response contains 256 generated positions. The block lengths used for the common four-model comparison are $\{1, 32, 128\}$. A separate 100-prompt LLaDA-8B sweep uses $\{1, 2, 4, 8, 16, 32, 64, 128, 256\}$. At every model evaluation, exactly one position is revealed, so all policies and all block lengths use exactly 256 evaluations.

The confidence policy reveals the position whose most probable token has the largest probability. The random policy samples uniformly from the same currently masked positions. Both write the most probable token at the chosen position. At block length one, there is only one eligible position and the policies coincide exactly. Each policy is evaluated on the same prompts. Outputs from which no answer can be extracted count as incorrect.

For LLaDA-8B, ten independent random-order draws are evaluated on the same 1000 prompts. For LLaDA-1.5, MMaDA-8B, and LLaDA-MoE, the ten-draw robustness panel evaluates ten fresh random-order draws on the same 500 prompts under one pinned runtime. The confidence-minus-random effect at block length 32 is positive for every model: $+0.0604$ on LLaDA-8B, $+0.0830$ on LLaDA-1.5, $+0.1572$ on MMaDA-8B, and $+0.0740$ on LLaDA-MoE. Their 95% intervals are $[+0.0399, +0.0811]$, $[+0.0538, +0.1122]$, $[+0.1216, +0.1932]$, and $[+0.0464, +0.1018]$, respectively. Each interval uses a crossed bootstrap that resamples prompts and random-order draws. For the three newly replicated models, the one-sided tests remain significant after Holm correction across models (adjusted $p = 3.0 \times 10^{-5}$). The $B = 1$ and $B = 128$ endpoints use one descriptive draw in that panel; at $B = 1$, reveal order is fixed, so the confidence and random policies coincide exactly.

For completeness, the separate 100-prompt LLaDA-8B sweep gives the full block-length curve below. It is not used for the four-model confidence-minus- random intervals in Table 1. All accuracies use the same all-prompt denominator as the main table.

Table 7: Complete 100-prompt LLaDA-8B block-length sweep at 256 evaluations per response.

B	1	2	4	8	16	32	64	128	256
confidence	.78	.79	.81	.79	.79	.77	.78	.80	.61
random	.78	.83	.82	.78	.72	.74	.66	.65	.52

C.5 Reveal-Policy Portfolios at Matched Compute

The comparison above varies one reveal policy at a time. A separate preregistered experiment instead runs several reveal policies concurrently and aggregates their answers. We evaluate Dream-7B (Ye

et al., 2025) on all 1319 GSM8K test prompts. Every response contains 256 generated positions, and tokens are sampled at temperature 0.2 with nucleus threshold 0.95. Each arm consumes exactly 256 sequence model evaluations per prompt and replicate, so the arms are matched on compute. Three frozen replicate groups of sampling seeds are run, and every reported accuracy averages over them.

One arm reveals a single position after each of 256 evaluations under confidence ordering. The other three arms run four trajectories of 64 actions in one batch and reveal four positions per action. Their policies are confidence at every path, an independent keyed random order at every path, and a portfolio of confidence, largest-minus-second-largest margin, smallest predictive entropy, and left-to-right. Portfolio policies rotate across sampling streams by prompt index, so no policy is permanently paired with one stream. Margin, entropy, and left-to-right scores are read from the same forward pass that supplies confidence, so they add no model evaluations. Answers within a four-trajectory arm vote by numeric equality, each unparseable output receives its own sentinel so that unrelated failures cannot agree, and accuracy averages analytically over tied winners. Outputs from which no answer can be extracted count as incorrect. Aggregating several sampled answers by majority vote is the established self-consistency scheme for language models (Wang et al., 2023). These arms differ from it in what generates the diversity: the trajectories share one model, one prompt, and one evaluation budget, and differ only in the reveal policy that orders their positions.

Section 3.2 reports the arms and the two preregistered headline contrasts. The gains remain positive across parseability checks: restricting the portfolio-minus-deep-confidence comparison to the 1167 prompts where the deep path parses in every group gives $+.1085$, an extreme assignment crediting every deep-path parse failure as correct retains $+.0294$, and the portfolio-minus-four-confidence sensitivity is $+.0685$. The two headline contrasts were fixed in advance, with a Holm correction over the pair and paired 20,000-replicate prompt bootstrap intervals whose resampling unit is the prompt. The replicate seed groups are $(18001, 18002, 18003, 18004)$, $(18011, 18012, 18013, 18014)$, and $(18021, 18022, 18023, 18024)$, and the single-trajectory arm uses the first seed of each group. Per prompt the target compute is exactly $3 \times 4 \times 256 = 3{,}072$ sequence model evaluations and 1,344 physical model invocations, so across all 1319 prompts it is 4,051,968 and 1,772,736. Within each prompt and replicate, arm execution order is a deterministic keyed permutation.

The LLaDA-1.5 replication uses LLaDA-1.5 revision and the same GSM8K revision, prompt suffix, parser, generation length, temperature, nucleus threshold, and matched 256-sequence-evaluation contrast. LLaDA uses unshifted logits at each masked position. The three seed groups are $(18101, 18102, 18103, 18104)$, $(18111, 18112, 18113, 18114)$, and $(18121, 18122, 18123, 18124)$. The all-prompt confirmatory arms are the four-policy portfolio and four confidence paths. Their exact target cost is 4,790,784 sequence model evaluations and 1,428,096 physical invocations after including the subset ablation below. A separately implemented numerical audit reproduced every statistic and gate to 10^{-12}.

LLaDA-1.5 extractor sensitivity. A separately preregistered re-analysis rejects punctuation-only marker matches while retaining the last-number fallback. Its primary all-prompt comparison and stricter no-fallback sensitivity both retain a positive portfolio-over-confidence effect. A positive control reproduced the committed analysis exactly. Table 8 reports both extractors. Entries give portfolio minus confidence, followed by retained prompt-replicates/effective prompts.

Table 8: LLaDA-1.5 extractor sensitivity. The single-unambiguous-winner (agreement-matched) subset requires both arms to return one unambiguous real winner. Intervals are paired prompt-bootstrap intervals.

instrument	denominator	portfolio minus confidence
primary	all prompts	$+.074994$ $[+.059831, +.090030]$ 3,957/1,319
primary	agreement-matched	$+.087071$ $[+.069352, +.105205]$ 3,132/1,204
strict sensitivity	all prompts	$+.021418$ $[+.007897, +.035317]$ 3,957/1,319
strict sensitivity	agreement-matched	$+.047250$ $[+.030089, +.064880]$ 1,782/709

The all-prompt comparison is primary. Agreement-matched rows are sensitivity checks because membership depends on each arm's output. Both the primary and strict extractors retain a positive portfolio-over-confidence effect.

Dream-7B extractor sensitivity. The corresponding Dream-7B re-analysis rejects punctuation-only marker matches while retaining the last-number fallback. Its primary all-prompt comparison and stricter no-fallback sensitivity both retain positive portfolio effects. Table 9 reports both extractors. Entries give portfolio minus comparator, followed by retained prompt-replicates/effective prompts.

Table 9: Dream-7B extractor sensitivity. The single-unambiguous-winner (agreement-matched) subset requires both arms to return one unambiguous real winner. Intervals are paired prompt-bootstrap intervals.

instrument	denominator	portfolio minus confidence	portfolio minus serial
primary	all prompts	$+.067033$ $[+.051870, +.082007]$ $3{,}957/1{,}319$	$+.146260$ $[+.122441, +.170647]$ $3{,}957/1{,}319$
primary	agreement-matched	$+.081874$ $[+.063803, +.100639]$ $3{,}009/1{,}199$	$+.146004$ $[+.118961, +.173503]$ $2{,}870/1{,}097$
strict sensitivity	all prompts	$+.071456$ $[+.056103, +.087062]$ $3{,}957/1{,}319$	$+.131855$ $[+.108289, +.155928]$ $3{,}957/1{,}319$
strict sensitivity	agreement-matched	$+.086323$ $[+.068387, +.104841]$ $2{,}834/1{,}143$	$+.146223$ $[+.118758, +.174001]$ $2{,}747/1{,}068$

The all-prompt comparisons are primary. Agreement-matched rows are sensitivity checks because membership depends on each arm's output. Both extractors retain positive portfolio effects against confidence and the serial path.

C.6 Repeated Continuations From One Saved State

Table 10: Repeated completions from the same saved MMLU state. Each state uses 32 continuations to estimate the answer distribution induced by greedy token prediction and randomized future reveal order. The final column reports whether a separate 33rd continuation differs from the leading option at the saved state. n counts the states reaching the 65% event out of 1000 prompts. State-level uncertainty intervals are in Table 11. The last two columns carry the saved-state-score benchmark: $\overline{b}_\tau$ is the mean four-option mass the leading option already holds at the saved state, which is the agreement that score would predict were it a posterior over the eventual answer, and the excess is $\widehat{q} - \overline{b}_\tau$.

model	n	$\widehat{q}$	all 32 agree	33rd differs	$\overline{b}_\tau$	excess
LLaDA-8B	537	.9920	.9795	.0056	.8735	$+.1185$
LLaDA-1.5	572	.9891	.9738	.0087	.8810	$+.1081$
MMaDA-8B	565	.9557	.9327	.0442	.8387	$+.1170$
LLaDA-MoE	631	.9635	.9160	.0380	.8702	$+.0933$

For each MMLU prompt (Hendrycks et al., 2021), the model generates an eight-position answer region by revealing one token after each evaluation. At each partial state, we normalize the logits for the four option letters at the designated answer position. The resulting four-option softmax is a current token score. It is not interpreted as a posterior over the answer after future decoding.

We include a model when the highest-scoring option after full generation agrees with the answer extracted from the completed output on at least 90% of parseable prompts. Four checkpoints satisfy this criterion and form the reported continuation panel.

From every saved state, 32 continuations use greedy token predictions and independently randomized future reveal order. A separate continuation is held out until after the 32-sample distribution is computed. The quantity $\widehat{q}$ is the mean fraction of the 32 continuations that return the option leading at the saved state. Its interval resamples saved states with $100{,}000$ bootstrap replicates. The binary all-agree and held-out-change columns use Wilson 95% intervals.

Concentration beyond the saved-state score. Selecting states at a 65% threshold guarantees a leading option, so the key comparison is with $\bar{b}_\tau$, the mean four-option mass already held by that option at the saved state. If the score were a calibrated posterior over the eventual answer, expected agreement would equal $\bar{b}_\tau$. Across the four reported models, observed agreement exceeds this benchmark by 0.0933 to 0.1185; paired prompt-bootstrap intervals exclude zero and Holm-adjusted one-sided tests give $p < 10^{-40}$. These estimates characterize the selected stratum, while the option score remains a selection statistic rather than an assumed calibrated posterior.

Table 11: Uncertainty for repeated continuations. Each row is conditional on reaching the score threshold while the answer position remains masked. Brackets are 95% intervals.

model	n	$\widehat{q}$	excess	all 32 agree	33rd differs
LLaDA-8B	537	.9920 [.9856,.9970]	+.1185 [.1073,.1294]	.9795 [.9637,.9885]	.0056 [.0019,.0163]
LLaDA-1.5	572	.9891 [.9818,.9951]	+.1081 [.0970,.1191]	.9738 [.9572,.9840]	.0087 [.0037,.0203]
MMaDA-8B	565	.9557 [.9403,.9698]	+.1170 [.1012,.1318]	.9327 [.9090,.9506]	.0442 [.0301,.0645]
LLaDA-MoE	631	.9635 [.9514,.9747]	+.0933 [.0805,.1059]	.9160 [.8918,.9352]	.0380 [.0257,.0560]

These intervals quantify concentration under the randomized continuation procedure specified above. Appendix C.2 complements this selected-state estimate with an exact reachable-set diameter on a smaller graph. The separate held-out continuation checks the empirical leader on a new draw after the 32-sample distribution is computed. Separately, on the original deterministic full-decoding trajectories, the final answer differs from the option leading at the same 65% event in only 0.008–0.041 of cases across six models. That trajectory analysis measures preservation under each model's own decoding policy rather than the conditional distribution under randomized order.

C.7 Returning the Partial Answer

The retrospective comparison uses the same 1000 prompts per model and retains every prompt in the denominator. At the first state where the leading option exceeds the threshold, the stopping rule returns that option. A prompt that never reaches the threshold is decoded fully. We compare this answer with the reference label and pair it with the fully decoded answer on the same prompt. Cost is the number of evaluations through the stopping state, or all eight evaluations for a noncrosser. The ARC-Challenge control (Clark et al., 2018) uses its prespecified 90% threshold. The intervals in Table 12 are paired prompt-bootstrap intervals. Across the seven model–task rows, stopping saves 52%–63% of evaluations and no interval indicates a detectable accuracy decrease. The mutually parseable SDAR-8B subset is near zero, with interval $[-0.012, +0.010]$.

Table 12: Retrospective stopping. The six MMLU rows use a 65% score threshold and ARC-Challenge uses 90%. Prompts that never reach the threshold decode fully. The 95% intervals pair stopping and full decoding by prompt. Evaluations are mean stored-trajectory counts out of eight.

model	task	full acc.	stop acc.	paired difference [95% CI]	evaluations	saved
LLaDA-8B	MMLU	0.694	0.692	-0.002 $[-0.007, +0.003]$	3.83/8	52%
LLaDA-1.5	MMLU	0.687	0.686	-0.001 $[-0.006, +0.004]$	3.52/8	56%
MMaDA-8B	MMLU	0.433	0.428	-0.005 $[-0.013, +0.003]$	3.54/8	56%
Dream-7B	MMLU	0.717	0.721	$+0.004$ $[-0.002, +0.010]$	2.95/8	63%
LLaDA-MoE	MMLU	0.656	0.652	-0.004 $[-0.014, +0.006]$	3.31/8	59%
SDAR-8B	MMLU	0.723	0.740	$+0.017$ $[+0.004, +0.031]$	3.30/8	59%
LLaDA-8B	ARC-Ch.	0.878	0.878	0.000 $[0.000, 0.000]$	3.46/8	57%

The prospective benchmark runs full decoding, threshold stopping with full fallback, and stopping after exactly four evaluations. It uses all 4250 MMLU prompts for LLaDA-8B and MMaDA-8B. Accuracy uses a subject-stratified paired bootstrap with $100{,}000$ replicates. Timing uses one A100 GPU, 128 fixed requests, three sessions, and six paired alternating repetitions per session. Synchronization occurs at both timing boundaries.

The 65% threshold, fixed-four comparator, accuracy margins, prompt set, packing seeds, and timing schedule were fixed before prospective execution. The prospective set is disjoint from the earlier 1000-prompt analyses. It was selected from $13{,}016$ eligible unique MMLU test IDs after excluding the 999 unique legacy IDs. The model identities are the LLaDA-8B revision and MMaDA snapshot SHA-256 reported above.

The execution environment used Python 3.12.6, PyTorch `2.10.0+cu128`, CUDA 12.8, cuDNN 9.1.0.2, Transformers 4.46.3, and Datasets 4.6.1. The three timing sessions use packing seeds $16401, 16402, 16403$. Each session performs 20 warm-up forwards before measurement, then alternates the arm order across six paired repetitions. CUDA is synchronized before and after each measured trace.

Table 13: Prospective accuracy and evaluation cost. The three accuracy and evaluation columns report full decoding, threshold stopping, and fixed-four stopping. Difference columns show the point estimate and preregistered one-sided 95% lower bound.

model	accuracy	mean evaluations	threshold − full	threshold − fixed four
LLaDA-8B	.6285 / .6247 / .6162	8.000 / 3.753 / 4.000	$-.0038$ [$-.0061$]	$+.0085$ [$+.0033$]
MMaDA-8B	.4181 / .4184 / .4186	8.000 / 3.687 / 4.000	$+.0002$ [$-.0028$]	$-.0002$ [$-.0054$]

Table 14: Prospective cost reduction of threshold stopping relative to full decoding. Each entry is the point estimate with its preregistered one-sided 95% lower bound. Batch eight is fixed-load makespan without replacement, not single-request latency.

model	evaluations	batch-1 latency	batch-8 makespan
LLaDA-8B	.5308 [.5212]	.5025 [.5018]	.4713 [.4664]
MMaDA-8B	.5391 [.5293]	.5282 [.5264]	.4858 [.4823]

Both models meet the prespecified one-percentage-point accuracy tolerance relative to full decoding. The threshold rule also improves over fixed-four stopping on LLaDA-8B and remains nearly matched on MMaDA-8B. The reported prospective efficiency result is the paired comparison with full decoding on these two models, MMLU, generation length eight, and one A100 system.