REASONING WITH SEQUENCE-LEVEL POWER SAMPLING BY STOCHASTIC BEAM SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning is the standard route to higher reasoning accuracy in large language models, and several recent analyses attribute its gains to sharpening a distribution the base model already carries rather than to acquiring new capabilities. Obtaining that sharpening through RL carries three costs: it needs a verifier, it needs a curated training set, and it rewrites the weights, so the gain is confined to mechanically checkable domains and is paid for elsewhere. Sampling the sequence-level power distribution $\pi_\beta \propto p^{1/\beta}$ gives the same sharpening from the base model alone, with no verifier, no dataset and no weight update, but π_β is not autoregressive: Markov chain Monte Carlo samples it correctly, but costs much time per candidate comparing to sample from the base model directly. In this work, we show that stochastic beam search samples π_β exactly once every prefix is scored by its power-weighted subtree mass, give a one-token lookahead estimator of that quantity, and show that omitting it reduces the sampler to low-temperature decoding. The method returns k candidates without replacement from one batched decode, at approximately the base model's cost per candidate. Evaluating three models on MATH500, HumanEval and GPQA, our sampler is more accurate than MCMC power sampling and GRPO on single-shot accuracy, while returning more distinct answers per query than either. Furthermore, this method can be applied to any model exposing token log-probabilities.

1 INTRODUCTION

Much of the recent gain in language-model reasoning can be read as *distribution sharpening*: biasing generation toward high-likelihood trajectories the pretrained model already supports, rather than changing its weights. In the previous work's findings, a GRPO-trained model which can solve some problems with fewer samples can be beaten by simply drawing more samples from the model it started from (He et al., 2025); outcome-based RL induces a systematic of generation diversity, even on the training set (Song et al., 2025); and post-RL traces sit inside the base model's own high-likelihood region (Yue et al., 2025). All three are consistent with relocation within a distribution rather than extension of it. The natural formalization is the sequence-level power distribution

$$\pi_\alpha(y \mid x) \;\propto\; p_\theta(y \mid x)^\alpha, \qquad \alpha > 1, \tag{1}$$

which concentrates mass on whole sequences rather than adjusting a token-level temperature. Its appeal is that it is specified by the base model alone: no verifier, no preference data, no training run.

Reinforcement learning reaches a similar place by a more expensive route. GRPO (Shao et al., 2024) and its relatives require a ground-truth verifier, a curated training set, and write access to the weights, which confines them to domains where correctness is mechanically checkable; they trade multi-shot coverage for single-shot accuracy (Song et al., 2025); and because they move weights, sharpening one distribution can damage others. Sampling from Eq. (1) has none of these failure modes, because nothing is trained and nothing is overwritten.

The obstacle is that Eq. (1) is not autoregressive. Lowering the decoding temperature to $\tau = 1/\alpha$ appears to sample it. It does not: temperature weights a token by an exponential of a sum over futures where the power distribution weights it by a sum of exponentials, so temperature rewards a prefix for having *many* viable continuations. In contrast, Eq. (1) rewards it for having *one strong* continuation. The two disagree exactly where a sharpening operator is meant to be doing work. (Karan & Du,

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.

2025) therefore sample Eq. (1) with Metropolis–Hastings, recovering strong reasoning performance but costing an expensive computation: proposals regenerate whole suffixes, accept/reject decisions are sequential, and each sample is drawn independently, so obtaining k candidates costs k chains. Sequential Monte Carlo has been applied to language-model inference before (Zhao et al., 2024), and concurrent work attacks this latency problem with it directly (Azizi et al., 2026).

In this work, we present a new approach that, despite targeting the same power distribution π_α, is different from Karan & Du (2025). Our mechanism can be described as follow: given a partial sequence, we will build from it recursively by extending it one next token at a time, assigning scores for them with a theoretical guarantee in terms of following a power distribution, and propose to open searching concurrently over k branches for diversity purposes.

Contributions. Our contributions are as follows:

1. We propose a new sampler for the power distribution but reaches it by a different mechanism. We build a sequence recursively built on the result of Gumbel-softmax distribution theory and apply this theory efficiently in our implementation. We also argue that our k concurrent exploring paths idea also shares the same principle compared to the work in (Karan & Du, 2025) and therefore propose using voting strategies to investigate k them better within a one search. We call our method as **Po**wer Stochastic **B**eam **S**earch (PoBS)
2. Empirical finding on various models and benchmarks, our sampler beats MCMC power sampling on every model and every benchmark, and it matches or beats each model's own GRPO checkpoint on every one. Our method also returns more distinct answers per problem than MCMC, GRPO so its accuracy does not come from collapsing onto one answer. Furthermore, its cost, measured in Table 2, is much cheaper ($31\times$ faster) than MCMC, and only a little more than sampling the base model directly.

2 PRELIMINARIES

2.1 SETUP AND NOTATION

Let p be an autoregressive language model over a vocabulary $\mathcal{V}$, so that a complete sequence $\mathbf{x} = (x_1, \ldots, x_T)$ has probability $p(\mathbf{x}) = \prod_{t=1}^{T} p(x_t \mid x_{<t})$. We write $\mathbf{u} = \mathbf{x}^{(j)}$ for a *prefix* with length j, identified with a node of the generation tree, $v \in \mathcal{V}$ for a candidate next token; $\mathbf{u} \oplus v$ for the extended prefix, and $\mathcal{L}(\mathbf{u})$ for the set of complete sequences extending $\mathbf{u}$. A prefix is *terminal* when it ends in EOS.

We parameterize sharpening throughout by a *temperature* $\beta \in (0, 1]$, and take as the object of interest the *sequence-level power distribution*

$$\pi_\beta(\mathbf{x}) \;=\; \frac{p(\mathbf{x})^{1/\beta}}{Z_\beta}, \qquad Z_\beta = \sum_{\mathbf{x}} p(\mathbf{x})^{1/\beta}, \tag{2}$$

which concentrates p without altering its support, and which reduces to p itself at $\beta = 1$. Prior work often writes this with an exponent $\alpha = 1/\beta$; we use β exclusively, because it is the quantity that behaves like a temperature, i.e., it is the scale of the perturbation in Eq. (4), and the smoothing parameter of the value recursion Eq. (7).

2.2 THE PERTURBED SCORE PROCESS ON THE TREE

We begin with Lemma 1, which connects the Gumbel-Softmax distribution to categorical sampling.

Lemma 1 (Scaled Gumbel-max)**.** *Let $S_i = \mu_i + \beta G_i$ with $G_i \overset{\text{iid}}{\sim} \text{Gumbel}(0, 1)$ and $\beta > 0$. Then*

$$\max_i S_i \;\sim\; \text{Gumbel}\Big(\beta \log \sum_i e^{\mu_i/\beta},\; \beta\Big), \qquad \arg\max_i S_i \;\sim\; \text{softmax}\big(\mu/\beta\big), \tag{3}$$

and the two are independent.

With the support of Lemma 1, for each completion $\mathbf{x}$, by defining the Gumbel score $S(\mathbf{x})$ as

$$S(\mathbf{x}) \;=\; \log p(\mathbf{x}) \;+\; \beta\, G(\mathbf{x}), \qquad G(\mathbf{x}) \overset{\text{iid}}{\sim} \text{Gumbel}(0, 1), \tag{4}$$

we can sample $\mathbf{x} \sim \pi_\beta(\cdot)$ by computing $\arg\max_{\mathbf{x}} S(\mathbf{x})$. However, this approach is simply intractable due to the vast volume of the space of samples $\mathbf{x}$.

We aim to sample $\mathbf{x}$ recursively. For a prefix $\mathbf{u} = \mathbf{x}^{(j)} = x_{1:j}$, we write the *subtree maximum* $S(\mathbf{u}) = \max_{\mathbf{x}\in\mathcal{L}(\mathbf{u})} S(\mathbf{x})$. Applying Lemma 1 at scale β and factorising $p(\mathbf{x}) = p(\mathbf{u})\, p(\mathbf{x} \mid \mathbf{u})$, we reach the following property on $S(\mathbf{u})$.

Proposition 1 (Subtree maximum)**.** *For every prefix* $\mathbf{u}$*,*

$$S(\mathbf{u}) \;\sim\; \mathrm{Gumbel}\big(\varphi_\beta(\mathbf{u}),\, \beta\big), \qquad \varphi_\beta(\mathbf{u}) \;=\; \log p(\mathbf{u}) \;+\; \beta \log Z_\beta(\mathbf{u}), \tag{5}$$

where we have defined

$$Z_\beta(\mathbf{u}) \;=\; \sum_{\mathbf{x}\in\mathcal{L}(\mathbf{u})} p(\mathbf{x} \mid \mathbf{u})^{1/\beta}, \qquad Z_\beta(\mathbf{u}) = 1, \textit{ for terminal } \mathbf{u}, \tag{6}$$

Moreover $\arg\max_{\mathbf{x}\in\mathcal{L}(\mathbf{u})} S(\mathbf{x}) \sim \pi_\beta(\cdot \mid \mathbf{u})$.

The location $\varphi_\beta(\mathbf{u})$ summarises everything reachable below $\mathbf{u}$ and vanishes only at $\beta = 1$. Moreover, to find out the next token v to form $\mathbf{u} \oplus v$, we need to compute and rank according to $S(\mathbf{u} \oplus v)$.

Writing $\varphi_\beta(\mathbf{u}, v) = \log p(\mathbf{u}, v) + \beta \log Z_\beta(\mathbf{u}, v)$ for children, Eq. (6) gives the *recursion*

$$\varphi_\beta(\mathbf{u}) \;=\; \beta \operatorname*{logsumexp}_{v\in\mathcal{V}} \frac{\varphi_\beta(\mathbf{u}, v)}{\beta}, \tag{7}$$

exhibiting a node's location as the β*-smoothed maximum* of its children's - the soft Bellman backup of entropy-regularised control (Haarnoja et al., 2017), with $\log p$ in the role of a reward. Moreover, conditioning the children on their parent's realised score, $\max_v S(\mathbf{u} \oplus v) = S(\mathbf{u})$, induces the *monotone chain*

$$S(\varnothing) \;\geq\; S\left(\mathbf{x}^{(1)}\right) \;\geq\; S\left(\mathbf{x}^{(2)}\right) \;\geq\; \cdots \tag{8}$$

along every root-to-leaf path.

3 Method

Sampling π_β exactly would require a quantity which no autoregressive model exposes. This section will propose the core foundation for our method and the efficient approximation in practice.

3.1 Construct sequence through Gumbel-Softmax Distribution

Assume we are given a prefix $\mathbf{u}$ of length j , our aim is to extend it by $\ell - j$ next tokens, so that what we append is distributed as the sharpened target restricted to that horizon. Denote

$$Z_\beta^{j:\ell}(\mathbf{u}) \;=\; \sum_{\mathbf{x}^{(\ell)}\supset\mathbf{u}} p(\mathbf{x}^{(\ell)})^{1/\beta}, \qquad \pi_\beta^{j:\ell}(\mathbf{x}^{(\ell)} \mid \mathbf{u}) \;=\; \frac{p(\mathbf{x}^{(\ell)})^{1/\beta}}{Z_\beta^{j:\ell}(\mathbf{u})}, \tag{9}$$

as the partition function and the target over the $|\mathcal{V}|^{\ell-j}$ sequences $\mathbf{x}$ extending $\mathbf{u}$ that reach length ℓ; at the full sequence length these are the $Z_\beta(\mathbf{u})$ and $\pi_\beta(\cdot \mid \mathbf{u})$ of Eq. (6) and Eq. (2). The following theorem is the result the rest of this section is built on.

Theorem 1. *Define* $S^{j:\ell}(\mathbf{u}) = \max_{\mathbf{x}^{(\ell)}\supset\mathbf{u}} \big\{\log p(\mathbf{x}^{(\ell)}) + \beta G(\mathbf{x}^{(\ell)})\big\}$*, the maximum perturbed log-probability score over the finitely many length-*ℓ *sequences* $\mathbf{x}^{(\ell)}$ *extending* $\mathbf{u}$*, with* $G(\mathbf{x}^{(\ell)}) \overset{\text{iid}}{\sim} \mathrm{Gumbel}(0, 1)$*. Then* $S^{j:\ell}(\mathbf{u})$ *will follow a Gumbel distribution:*

$$S^{j:\ell}(\mathbf{u}) \;\sim\; \mathrm{Gumbel}\big(\beta \log \sum_{\mathbf{x}^{(\ell)}\supset\mathbf{u}} p(\mathbf{x}^{(\ell)})^{1/\beta},\; \beta\big) \;=\; \mathrm{Gumbel}\big(\beta \log Z_\beta^{j:\ell}(\mathbf{u}),\; \beta\big),$$

and $\arg\max_{\mathbf{x}^{(\ell)}\supset\mathbf{u}} \{\log p(\mathbf{x}^{(\ell)}) + \beta G(\mathbf{x}^{(\ell)})\} \sim \pi_\beta^{j:\ell}(\cdot \mid \mathbf{u})$ *of Eq. (9), independently of* $S^{j:\ell}(\mathbf{u})$.

Suppose we have already drawn a prefix $\mathbf{u}$ from $\pi_\beta^{j:\ell}$ and know the value of its score $S^{j:\ell}(\mathbf{u})$. To append the $(\ell{+}1)^{\text{th}}$ token, we have the following property for the sequence with one-token expansion from the given prefix $\mathbf{u}$:

$$S^{j:(\ell+1)}(\mathbf{u}) = \max_{v\in\mathcal{V}} \Big\{ S^{(j+1):(\ell+1)}(\mathbf{u}\oplus v) \Big\} \tag{10}$$

Based on the Theorem 1, the score $S^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)$ follows a Gumbel distribution $S^{(j+1):(\ell+1)}(\mathbf{u}\oplus v) \sim \text{Gumbel}\big(\beta \log \sum_{\mathbf{x}^{(\ell+1)}\supset \mathbf{u}\oplus v} p(\mathbf{x}^{(\ell+1)})^{1/\beta},\ \beta\big)$. Combining with the Eq. (10) gives us the new approximation for the Gumbel scores over one-token expansion from the prefix $\mathbf{u}$:

Lemma 2 (Max-pinned Gumbel score)**.** *Assume we draw $S^{(j+1):(\ell+1)}(\boldsymbol{u}\oplus v)$ from a Gumbel distribution: $S^{(j+1):(\ell+1)}(\boldsymbol{u}\oplus v) \sim \text{Gumbel}\big(\beta \log \sum_{\mathbf{x}^{(\ell+1)}\supset \mathbf{u}\oplus v} p(\mathbf{x}^{(\ell+1)})^{1/\beta},\ \beta\big)$ and let $Z = \max_{v\in\mathcal{V}} \big\{S^{(j+1):(\ell+1)}(\boldsymbol{u}\oplus v)\big\}$ then:*

$$\widetilde{S}^{(j+1):(\ell+1)}(\mathbf{u}\oplus v) \;=\; -\beta \log\Big(e^{-S^{j:\ell}(\mathbf{u})/\beta} - e^{-Z/\beta} + e^{-S^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)/\beta}\Big). \tag{11}$$

Here we use the given term $S^{j:\ell}(u)$ instead of $S^{j:(\ell+1)}(u)$ by observing an inequality $S^{j:(\ell+1)}(u) \le S^{j:\ell}(u)$ when $\beta < 1$

With that choice of $v^\star = \arg\max_{v\in\mathcal{V}} \widetilde{S}^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)$, it will lead to that property:

Theorem 2. *Define $\pi_\beta(\mathbf{x}^{(\ell)})$ be the power distribution over length-ℓ sequences, $\pi_\beta^{1:j}(\mathbf{x}^{(\ell)})$ be the marginal distribution of $\pi_\beta(\mathbf{x}^{(\ell)})$ at position $j < \ell$. Given the prefix $\mathbf{u} = \mathbf{x}^{(j)}, \mathbf{u} \sim \pi_\beta^{1:j}(\mathbf{x}^{(\ell)})$, choosing optimal $v^\star = \arg\max_{v\in\mathcal{V}} \widetilde{S}^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)$ will leads a distribution of one-token expasion from $\mathbf{u}$*

$$\mathbf{u}\oplus v \sim \pi_\beta^{1:(j+1)}(\mathbf{x}^{(\ell+1)}) \tag{12}$$

3.2 Algorithm

In practice, we choose $\ell = j+1$, from the Eq. (10), we have

$$S^{j:(j+2)}(\mathbf{u}) = \max_{v\in\mathcal{V}} \Big\{ S^{(j+1):(j+2)}(\mathbf{u}\oplus v) \Big\} \tag{13}$$

By Theorem 1, $S^{(j+1):(j+2)}(\mathbf{u}\oplus v)$ is the maximum over the length-$(j+2)$ sequences extending $\mathbf{u}\oplus v$, factorize them as $p(\mathbf{u}\oplus v\oplus w) = p(\mathbf{u}\oplus v)\,p(w \mid \mathbf{u}\oplus v)$ so that $S^{(j+1):(j+2)}(\mathbf{u}\oplus v) = \log p(\mathbf{u}\oplus v) + \max_{w\in\mathcal{V}} \{\log p(w \mid \mathbf{u}\oplus v) + \beta G(w)\}$. Lemma 1 replaces that inner maximum by its own location plus a fresh perturbation, and we write

$$\beta \log Z_\beta(\mathbf{u}, v) \;=\; \beta \log \sum_{w\in\mathcal{V}} p(w \mid \mathbf{u}\oplus v)^{1/\beta} \tag{14}$$

for that location, the exact one-token mass below the child $\mathbf{u}\oplus v$. Substituting into Eq. (13) gives

$$\begin{aligned} S^{j:(j+2)}(\mathbf{u}) &= \max_{v\in\mathcal{V}} \Big\{ \log p(\mathbf{u}\oplus v) + \max_{w\in\mathcal{V}} \{\log p(w \mid \mathbf{u}\oplus v) + \beta G(w)\} \Big\} \\ &= \max_{v\in\mathcal{V}} \{\log p(\mathbf{u}\oplus v) + \beta \log Z_\beta(\mathbf{u}, v) + \beta G(v)\} \end{aligned} \tag{15}$$

with $G(v) \sim \text{Gumbel}(0,1)$ drawn independently across v.

In our implementation, for the efficient computation purpose, we do approximation as follow: with $p(v_{(1)} \mid \mathbf{u}) \ge p(v_{(2)} \mid \mathbf{u}) \ge \cdots$,

$$\mathcal{C}(\mathbf{u}) = \{v_{(1)}, \ldots, v_{(m)}\}, \qquad m = \min\Big\{ m' \le E \;:\; \sum_{i\le m'} p(v_{(i)} \mid \mathbf{u}) \ge \rho \Big\}, \tag{16}$$

Algorithm 1 Power stochastic beam search for π_β with horizon-1 lookahead.

Require: : model p; temperature β; length n; beam size k; budget E; nucleus mass ρ
Ensure: : Top k sequences, without replacement, ranked by the horizon-1 surrogate scores.

$B_0 \leftarrow \{(\varnothing,\ S(\varnothing) = \beta G)\}, \quad G \sim \mathrm{Gumbel}(0,1)$
for $j = 0$ **to** $n-1$ **do**
  $\mathcal{A} \leftarrow \varnothing$
  for $\big(\mathbf{u},\ S^{j:(j+1)}(\mathbf{u})\big) \in B_j$ **do**
    evaluate $\log p(\cdot \mid \mathbf{u})$ ▷ one forward pass
    $\mathcal{C}(\mathbf{u}) \leftarrow \{v_{(1)}, \ldots, v_{(m)}\}$ ▷ Eq. (16)
    for $v \in \mathcal{C}(\mathbf{u})$ **do**
      evaluate $\log p(\cdot \mid \mathbf{u} \oplus v)$ ▷ one forward row
    end for
    $\beta \log \hat{Z}_\beta(\mathbf{u}, v) \leftarrow$ Eq. (17) $\quad \forall v \in \mathcal{V}$
    $S^{(j+1):(j+2)}(\mathbf{u} \oplus v) \leftarrow \log p(\mathbf{u} \oplus v) + \beta \log \hat{Z}_\beta(\mathbf{u}, v) + \beta G_v, \quad G_v \sim \mathrm{Gumbel}(0,1)$ ▷ Eq. (18)
    $Z \leftarrow \max_{v \in \mathcal{V}} S^{(j+1):(j+2)}(\mathbf{u} \oplus v)$
    $\widetilde{S}^{(j+1):(j+2)}(\mathbf{u} \oplus v) \leftarrow$ max-pin to $S^{j:(j+1)}(\mathbf{u})$ ▷ Lemma 2
    $\mathcal{A} \leftarrow \mathcal{A} \cup \big\{\big(\mathbf{u} \oplus v,\ \widetilde{S}^{(j+1):(j+2)}(\mathbf{u} \oplus v)\big)\big\}_v$
  end for
  $B_{j+1} \leftarrow$ top-k of $\mathcal{A}$ by $\widetilde{S}$ ▷ prune
end for
return B_n

$$\beta \log \hat{Z}_\beta(\mathbf{u}, v) = \begin{cases} 0, & v = \text{EOS}, \\ \beta \log \sum_{w \in \mathcal{V}} p(w \mid \mathbf{u} \oplus v)^{1/\beta}, & v \in \mathcal{C}(\mathbf{u}) \setminus \{\text{EOS}\}, \\ \min_{v' \in \mathcal{C}(\mathbf{u})} \beta \log \hat{Z}_\beta(\mathbf{u}, v'), & \text{otherwise}, \end{cases} \tag{17}$$

$$S^{(j+1):(j+2)}(\mathbf{u} \oplus v) \ = \ \log p(\mathbf{u} \oplus v) \ + \ \beta \log \hat{Z}_\beta(\mathbf{u}, v) \ + \ \beta G(v). \tag{18}$$

Why beam search with k candidates. When choosing $\ell = j+1$, Theorem 1 tells us that picking the value $v^\star$ followed the Lemma 2 will expand the given prefix towards a sequence that also follows the power distribution $\pi_\beta^{j:(j+1)}(\cdot \mid \mathbf{u})$. However, that only keep one child achieved $\arg\max$ and ignores other branches, so if once the kept child is wrong at one next step, the $\arg\max$ picking will go far from proper search and cannot recover the correct path. Therefore we maintain k over sampling process for the diverse exploration purpose. In the same point of view, MCMC (Karan & Du, 2025) also maintains multiple candidates together in only search time through Metropolis-Hasting procedure which compares all candidates and returns one the best one. By experiment comparing running inference time in Table 2, we also found that our proposed method achieved approximate $31\times$ faster than MCMC. On the other hand, Kool et al. (2019) also proposes one type of Stochastic Beam Search and maintains k-beam during the searching process. However this method only target the proper distribution from an auto-regressive model, but not the power distribution. We also conducted experiments to compare ours against this work in Table 3, demonstrating our better gains across benchmarks and models.

Voting Strategy. Following the idea of comparing and picking the best candidate within one search, we proposed 4 different voting strategies for our k completed sequences:

- **Native:** We take the first candidate the sampler returned.

$$\hat{a} = a_{i^\star}, \qquad i^\star = \arg\max_{i \in \mathcal{P}} \big\{\log p(\mathbf{x}_i) + \beta G(\mathbf{x}_i)\big\}.$$

- **Best-of-N:** We take the candidate with the largest confidence $s_i = \frac{1}{T_i} \sum_{t=1}^{T_i} \log p\big(x_{i,t} \mid x_{i,<t}\big), i = 1, \ldots, k.$

$$\hat{a} = a_{i^\star}, \qquad i^\star = \arg\max_{i \in \mathcal{P}} s_i.$$

- **Vote:** Group the candidates by their answer and take the answer that the most candidates share. Every candidate counts once,

$$\hat{a} = \arg\max_{a} \sum_{i \in \mathcal{P}} \mathbf{1}[a_i = a].$$

- **Weighted vote:** Group the same way, but candidate i contributes $e^{\lambda z_i}$ instead of 1, where $z_i = (s_i - \bar{s})/\sigma_s$ is:

$$\hat{a} = \arg\max_{a} \sum_{i \in \mathcal{P}} \mathbf{1}[a_i = a]\, e^{\lambda z_i}.$$

The ablation study among these strategies is conducted and reported at the Table 4 and we selected **Weighted Vote** as the main voting rule for the main experiments and summarize our algorithm in Algorithm 1. All proof of our theory can be found at Appendix A

4 EXPERIMENTS

4.1 SETUP

Models, Benchmarks and Baselines. We evaluate three models: Qwen2.5-Math-7B (Yang et al., 2024), Qwen2.5-7B (Qwen et al., 2025) and DeepSeek-Math-7B-Instruct (Shao et al., 2024) on the three benchmarks MATH500 ($n = 500$) (Lightman et al., 2024), HumanEval ($n = 164$) (Chen et al., 2021) and GPQA ($n = 198$) (Rein et al., 2024) benchmarks, with chain-of-thought prompting and a budget of $T_{\max} = 3072$ new tokens. Answers on MATH500 and GPQA Diamond are extracted by symbolic parsing and graded by equivalence rather than string match; HumanEval is graded by executing the benchmark's reference tests. All experiments are conducted on a single NVIDIA A100 (80 GB). For the baselines, we mainly compare our method against GRPO (Shao et al., 2024), MCMC (Karan & Du, 2025) and their base models.

Sampler configuration. We use beam size $k = 8$, the max-pin draw rule, and the horizon-1 surrogate Eq. (17) with candidate budget $E = 8$ and nucleus mass $\rho = 0.999$ for our method, the sum in Eq. (17) is evaluated exactly as a single `logsumexp` over the vocabulary row, which is already in memory. The default temperature is $\beta = 0.25$, matching prior work.

Selection configuration. A beam returns k candidates, so a single-shot number needs a rule for picking one. We use **weighted vote** with $\lambda = 0.5$ on every beam method unless a row says otherwise: we normalize the confidences $s_i = \log p(x_i)/T_i$ by its length within each problem, rank answers by $\sum_{i:a_i=a} e^{\lambda z_i}$, break ties by first occurrence, and submit the most confident member of the winning answer, excluding candidates whose answer fails to parse and ordering them last. Since those weights decide the answer, we recompute every $\log p(x_i)$ teacher-forced under the base model instead of reusing the perturbed score S the sampler ranks by, which keeps the Gumbel noise out of the rule. comparisons below are *paired*.

4.2 RESULTS

Main Results. Table 1 summarizes the single-shot accuracy over 3 benchmarks across three models. On the benchmark MATH500, our sampler raises Qwen2.5-Math-7B from 0.477 to 0.828, Qwen2.5-7B from 0.414 to 0.772 and DeepSeek-Math-7B-Instruct from 0.372 to 0.534, in each case above the corresponding GRPO by 5.0, 9.0 and 2.0 points. MCMC power sampling targets the same distribution and reaches 0.724, 0.692 and 0.460. The closest competitor is beam search on the base model, which reaches 0.800 and 0.752 on the two Qwen models and beats GRPO and our sampler stays out-performed it in every block. The results highlight a generalization capability of our method on verifiable domains.

We also evaluated on the out-of-domain scenarios including: GPQA and HumanEval followed by (Karan & Du, 2025). On HumanEval, GRPO falls to or below low-temperature decoding on every model, with about ten points lower on both Qwen checkpoints and matching it on DeepSeek-Math-7B-Instruct, while every sampling method improves on the base model it sharpens. Our sampler takes the best entry in each block, at 0.579, 0.591 and 0.610. On GPQA, our sampler is at or above

Table 1: Single-shot accuracy on MATH500, HumanEval and GPQA Diamond across three models.

	MATH500	HumanEval	GPQA
Qwen2.5-Math-7B			
Base, $\tau = 1.0$	0.477	0.311	0.313
Low-temperature, $\tau = 0.25$	0.684	0.518	0.340
MCMC power sampling	0.724	0.549	0.308
Base + beam search	0.800	0.452	**0.344**
GRPO	0.778	0.427	0.328
PoBS (ours)	**0.828**	**0.579**	0.338
Qwen2.5-7B			
Base, $\tau = 1.0$	0.414	0.341	0.303
Low-temperature, $\tau = 0.25$	0.614	0.530	0.333
MCMC power sampling	0.692	0.573	0.298
Base + beam search	0.752	0.565	0.288
GRPO	0.682	0.421	**0.343**
PoBS (ours)	**0.772**	**0.591**	**0.343**
DeepSeek-Math-7B-Instruct			
Base, $\tau = 1.0$	0.372	0.439	0.335
Low-temperature, $\tau = 0.25$	0.460	0.561	0.283
MCMC power sampling	0.460	0.585	0.318
Base + beam search	0.502	0.549	0.323
GRPO	0.514	0.561	0.298
PoBS (ours)	**0.534**	**0.610**	**0.348**

every other method on Qwen2.5-7B and DeepSeek-Math-7B-Instruct, at 0.343 and 0.348, and sits to 0.338 on Qwen2.5-Math-7B, just under low-temperature decoding.

Inference-Time Comparison. Table 2 reported wall-clock cost per problem running on a single A100. Our method costs 9.6 seconds per candidate against the base model's 7.5, $1.3\times$ higher, and that proximity is entirely the extra forward rows the horizon-1 lookahead evaluates at each step. Against MCMC power sampling in which a single MCMC draw costs 295 seconds, so the 8 candidates our beam returns in 77 seconds would cost 2357, our method is $31\times$ faster at a same budget. These numbers indicate that our method achieved not only better performance but also a approximated running time compared to base model's sampling.

Pass@k Results. Figure 1 reports accuracy against the number of candidates consulted, for the same five methods on MATH500 with Qwen2.5-Math-7B. At $k = 1$ our sampler leads at 0.828, above GRPO by 5.5 points and above the base model by 33.4. The ordering almost reverses by $k = 8$: the base model rises from last to second and GRPO falls from second to fourth, at 0.874 against the base model's 0.885, which is the behaviour reported as diversity collapse (Yue et al., 2025). On the other hand, sharpening at inference time does not decay this way: our sampler leads at every budget and ends first at 0.888. These results indicate that RL post-training pays for its single-shot gain with coverage, while sampling targeting π_β obtains the gain without that cost.

4.3 Ablation Study

Sensitivity on hyper-parameters. Figure 2 varies the two hyper-parameters the method exposes: Panel (a) varies the sharpening strength β. The four values lie between 0.812 at $\beta = 0.15$ and 0.828 at $\beta = 0.25$, so the sweep is flat and we use $\beta = 0.25$ for our main experiments. The panel separates the method from the baselines far more sharply than it separates one β from another: the weakest setting we experimented is still higher than MCMC power sampling by 0.098 and the base model by 0.342. Panel (b) varies the beam width k, each width a separate run of the sampler. Weighted vote rises from 0.818 at $k = 4$ to 0.867 at $k = 32$. We chose $k = 8$ in our main experiment to balancing between both performance and latency. These results indicate that our proposed method is robust to the choice of hyper-parameters.

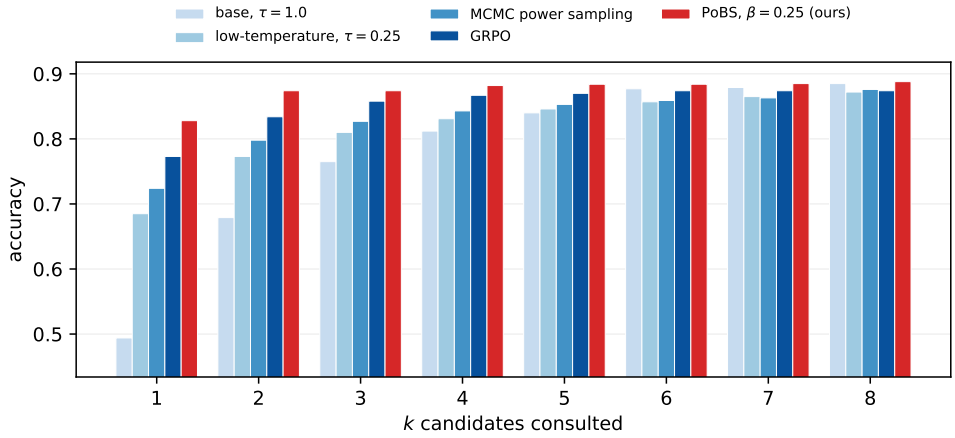

Figure 1: Accuracy against the number of candidates consulted, on MATH500 with Qwen2.5-Math-7B.

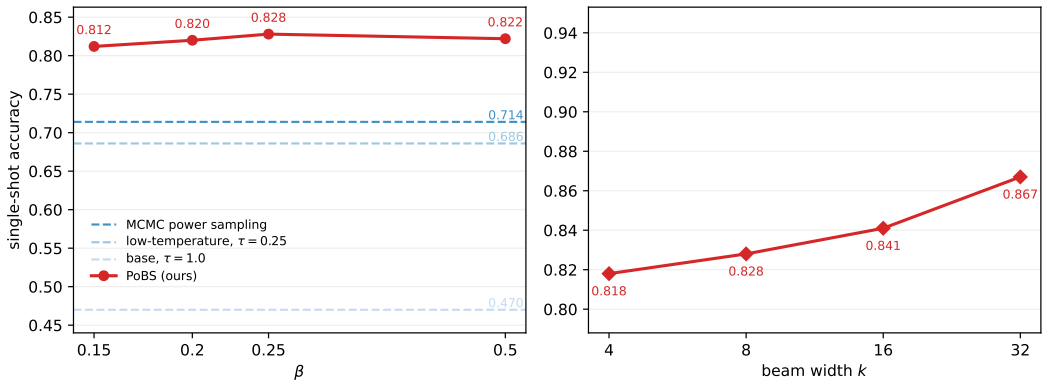

Figure 2: Single-shot accuracy against: (left) sharpening value β and (right) beam size k – on MATH500 with Qwen2.5-Math-7B.

5 Related Work

RL post-training and distribution sharpening. Optimizing a model with a verifier-based setting is a standard route to higher reasoning accuracy, through rejection-sampling fine-tuning (Xiong et al., 2025), preference optimization over sampled pairs (Liu et al., 2024), or GRPO and its variants (Shao et al., 2024; Yu et al., 2025); see Xu et al. (2025) and Zhang et al. (2025) for surveys. A growing evidence holds that the gains come from sharpening a distribution the base model already carries rather than from new capability: Yue et al. (2025) show that the advantage shrinks and reverses as the sampling budget grows, the diversity collapse Figure 1 reproduces on Qwen2.5-Math-7B. Work that treats the collapse as a defect to repair, by rewarding unlikely completions (He et al., 2025) or by exploring against outcome statistics (Song et al., 2025), keeps the training loop and its verifier.

Inference-time sampling. Low-temperature decoding (Wang et al., 2020) sharpens each token independently and so ignores the quality of the sequence it commits to. Repeated sampling (Brown et al., 2024), Best-of-N (Huang et al., 2025) and self-consistency (Wang et al., 2023) improve on a single draw by consuming several, but they are selection rules applied after the fact rather than changes to the distribution being sampled; we report both as rules over our own candidates in Table 4, where they recover the endpoints of the weighted vote we use. Search procedures such as Tree of Thoughts (Yao et al., 2023) and Monte Carlo tree search (Zhang et al., 2023; 2024) add state tracking and backtracking; our method remains a single left-to-right traversal with no backtracking.

Table 2: Wall-clock cost per problem on MATH500 with Qwen2.5-Math-7B.

Sampler	k	n	mean (s)	s / candidate
Low-temperature, $\tau = 0.25$	8	500	22	2.7
Base, $\tau = 1.0$	8	500	60	7.5
PoBS, $\beta = 0.25$ (ours)	8	500	77	9.6
MCMC power sampling	1	500	295	294.6

Diverse beam search (Vijayakumar et al., 2018) shares our goal of returning distinct candidates but enforces diversity through a penalty rather than obtaining it from the sampling scheme, and carries no distributional guarantee.

Sampling the power distribution. Karan & Du (2025) proposed the object we target, showing that samples from $\pi_\beta \propto p^{1/\beta}$ recover RL-level accuracy from the base model alone, and sampled it with Metropolis–Hastings. That implementation is correct but slow: one chain yields one draw, so k candidates cost k chains, and a finite chain gives no exactness guarantee, since Markov dynamics can mix exponentially slowly even in models whose stationary law is known exactly (Gheissari et al., 2017). We measure 295 seconds per candidate against 9.6 for our beam. Subsequent work accepts the target and focuses on addressing the latency, replacing the chain with sequential Monte Carlo (Azizi et al., 2026) or with autoregressive approximations whose per-token correction is estimated by Monte Carlo rollouts (Ji et al., 2026). We target the same distribution and differ in what replaces the intractable term: our estimate is the partition function of the child's own next-token distribution, a horizon-one surrogate available in closed form from one $\mathrm{logsumexp}$ over a single forward row. Related sequence-level methods outside the power-sampling line include twisted sequential Monte Carlo (Zhao et al., 2024), quality-aware Metropolis–Hastings for translation (Faria et al., 2024), and inference-time steering of a pretrained diffusion model toward a reward-tilted target (Karan et al., 2025).

Sampling without replacement. Stochastic beam search (Kool et al., 2019), which perturbs every complete sequence once with Gumbel noise and returns the top k under a max-pin that keeps the perturbed scores consistent along every path. Their construction is exact at $\beta = 1$, where the subtree partition function is identically one and the score reduces to $\log p$, and it returns k sequences drawn without replacement from p in a single traversal. Sharpening breaks that simplification: at $\beta < 1$ the same construction needs a per-node partition function no autoregressive model exposes, and dropping it silently retargets the sampler to the token-wise p_β rather than π_β. Our contribution is the estimator that supplies it: with the exact mass the without-replacement guarantee carries over unchanged, and the horizon-one estimate inherits it up to the estimator's own error. The single traversal is inherited outright, which is why k candidates cost what one beam costs rather than what k chains cost.

6 Conclusion

In this work, we proposed a new framework that targets power distribution without training but utilizes the power of Gumbel softmax distibution and investigate the exploring capability with k-beam search. The empirical results also demonstrate that our proposed method significantly outperforms other baseline while achieving almost similar inference cost compared to traditional sampling from a base model.

In the future work, we continue to apply this framework in various settings: sampling from a distribution that reflects a specific property, as long as we can access the probability of a given strong base model: one or a group of preferences (Rafailov et al., 2024) (Luo et al., 2026) (Zhou et al., 2024), without training data; GRPO (Shao et al., 2024) with diverse and high-quality rollouts for better generalization.

AI use statement

In this work, we used generative AI tools to assist with drafting, to obtain feedback on mathematical explanations and interpretation of results. The authors reviewed and edited the resulting material,

including its claims and conclusions. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

The complete sampler formulation and its score recursion are given in Section 3, with the max-pin draw rule, the horizon-one estimator and the candidate restriction stated in Algorithm 1, selection hyperparameters, together with the evaluation protocol, the answer graders and the baselines, are reported in Section 4, and Appendix B gives the candidate-budget sweep and the full selection-rule comparison.

References

Seyedarmin Azizi, Erfan Baghaei Potraghloo, Minoo Ahmadi, Souvik Kundu, and Massoud Pedram. Power-smc: Low-latency sequence-level power sampling for training-free llm reasoning, 2026. URL https://arxiv.org/abs/2602.10273.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling, 2024. URL https://arxiv.org/abs/2407.21787.

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

Gonçalo R. A. Faria, Sweta Agrawal, António Farinhas, Ricardo Rei, José G. C. de Souza, and André F.T. Martins. Quest: Quality-aware metropolis-hastings sampling for machine translation. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 89042–89068. Curran Associates, Inc., 2024. doi: 10.52202/079017-2825. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/a221d22ff6a33599142c8299c7ed06bb-Paper-Conference.pdf.

Reza Gheissari, Eyal Lubetzky, and Yuval Peres. Exponentially slow mixing in the mean-field swendsen-wang dynamics, 2017. URL https://arxiv.org/abs/1702.05797.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies, 2017. URL https://arxiv.org/abs/1702.08165.

Andre He, Daniel Fried, and Sean Welleck. Rewarding the unlikely: Lifting grpo beyond distribution sharpening, 2025. URL https://arxiv.org/abs/2506.02355.

Audrey Huang, Adam Block, Qinghua Liu, Nan Jiang, Akshay Krishnamurthy, and Dylan J. Foster. Is best-of-n the best of them? coverage, scaling, and optimality in inference-time alignment, 2025. URL https://arxiv.org/abs/2503.21878.

Xiaotong Ji, Rasul Tutunov, Matthieu Zimmer, and Haitham Bou Ammar. Scalable power sampling: Unlocking efficient, training-free reasoning for llms via distribution sharpening, 2026. URL https://arxiv.org/abs/2601.21590.

Aayush Karan and Yilun Du. Reasoning with sampling: Your base model is smarter than you think, 2025. URL https://arxiv.org/abs/2510.14901.

Aayush Karan, Kulin Shah, and Sitan Chen. Reguidance: A simple diffusion wrapper for boosting sample quality on hard inverse problems, 2025. URL https://arxiv.org/abs/2506.10955.

Wouter Kool, Herke Van Hoof, and Max Welling. Stochastic beams and where to find them: The Gumbel-top-k trick for sampling sequences without replacement. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3499–3508. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/kool19a.html.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/aca97732e30bcf1303bc22ac3924fd16-Paper-Conference.pdf.

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J. Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization, 2024. URL https://arxiv.org/abs/2309.06657.

Linhao Luo, Thuy-Trang Vu, Van-Anh Nguyen, Junae Kim, Gholamreza Haffari, and Dinh Phung. Mato: Multi-objective personalized alignment with test-time optimization for large language models, 2026. URL https://arxiv.org/abs/2605.25342.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL https://arxiv.org/abs/2305.18290.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Yuda Song, Julia Kempe, and Remi Munos. Outcome-based exploration for llm reasoning, 2025. URL https://arxiv.org/abs/2509.06941.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R. Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search: Decoding diverse solutions from neural sequence models, 2018. URL https://arxiv.org/abs/1610.02424.

Pei-Hsin Wang, Sheng-Iou Hsieh, Shih-Chieh Chang, Yu-Ting Chen, Jia-Yu Pan, Wei Wei, and Da-Chang Juan. Contextual temperature for language modeling, 2020. URL https://arxiv.org/abs/2012.13575.

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, 2023. URL https://arxiv.org/abs/2203.11171.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, and Hanze Dong. A minimalist approach to llm reasoning: from rejection sampling to reinforce, 2025. URL https://arxiv.org/abs/2504.11343.

Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, Chenyang Shao, Yuwei Yan, Qinglong Yang, Yiwen Song, Sijian Ren, Xinyuan Hu, Yu Li, Jie Feng, Chen Gao, and Yong Li. Towards large reasoning models: A survey of reinforced reasoning with large language models, 2025. URL https://arxiv.org/abs/2501.09686.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024. URL https://arxiv.org/abs/2409.12122.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023. URL https://arxiv.org/abs/2305.10601.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 57654–57689. Curran Associates, Inc., 2025. doi: 10.52202/085713-1933. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/537d5aa768c2d534016a4d06f87bc8fb-Paper-Conference.pdf.

Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang Li, and Wanli Ouyang. Accessing gpt-4 level mathematical olympiad solutions via monte carlo tree self-refine with llama-3 8b, 2024. URL https://arxiv.org/abs/2406.07394.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, Yu Fu, Xingtai Lv, Yuchen Zhang, Sihang Zeng, Shang Qu, Haozhan Li, Shijie Wang, Yuru Wang, Xinwei Long, Fangfu Liu, Xiang Xu, Jiaze Ma, Xuekai Zhu, Ermo Hua, Yihao Liu, Zonglin Li, Huayu Chen, Xiaoye Qu, Yafu Li, Weize Chen, Zhenzhao Yuan, Junqi Gao, Dong Li, Zhiyuan Ma, Ganqu Cui, Zhiyuan Liu, Biqing Qi, Ning Ding, and Bowen Zhou. A survey of reinforcement learning for large reasoning models, 2025. URL https://arxiv.org/abs/2509.08827.

Shun Zhang, Zhenfang Chen, Yikang Shen, Mingyu Ding, Joshua B. Tenenbaum, and Chuang Gan. Planning with large language models for code generation, 2023. URL https://arxiv.org/abs/2303.05510.

Stephen Zhao, Rob Brekelmans, Alireza Makhzani, and Roger Grosse. Probabilistic inference in language models via twisted sequential monte carlo, 2024. URL https://arxiv.org/abs/2404.17546.

Zhanhui Zhou, Jie Liu, Jing Shao, Xiangyu Yue, Chao Yang, Wanli Ouyang, and Yu Qiao. Beyond one-preference-fits-all alignment: Multi-objective direct preference optimization, 2024. URL https://arxiv.org/abs/2310.03708.

A PROOFS

Statements are reproduced for readability; numbering refers to the main text. The proofs are ordered by dependency rather than by order of appearance: Lemma 1 is proved first, Theorem 1 follows from it, and Proposition 1 is the special case of the theorem at full sequence length.

A.1 LEMMA 1 (SCALED GUMBEL-MAX)

Let $S_i = \mu_i + \beta G_i$ *with* $G_i \overset{\text{iid}}{\sim} \mathrm{Gumbel}(0,1)$ *and* $\beta > 0$. *Then* $\max_i S_i \sim \mathrm{Gumbel}\big(\beta \log \sum_i e^{\mu_i/\beta},\ \beta\big)$ *and* $\arg\max_i S_i \sim \mathrm{softmax}(\mu/\beta)$*, and the two are independent.*

Proof. Write $a_i = e^{\mu_i/\beta}$ and $A = \sum_i a_i$. Since G_i is standard Gumbel,

$$F_i(s) \;=\; \Pr[S_i \le s] \;=\; \exp\big(-e^{-(s-\mu_i)/\beta}\big) \;=\; \exp\big(-a_i e^{-s/\beta}\big),$$

and by independence $\Pr[\max_i S_i \le s] = \prod_i F_i(s) = \exp(-Ae^{-s/\beta})$. Since $Ae^{-s/\beta} = e^{-(s-\beta\log A)/\beta}$, this is the CDF of $\mathrm{Gumbel}(\beta\log A, \beta)$. For the index, let f_j be the density of S_j. Then

$$\Pr\Big[\arg\max_i S_i = j,\ \max_i S_i \le s\Big] = \int_{-\infty}^{s} f_j(t)\prod_{i\neq j} F_i(t)\,dt = \int_{-\infty}^{s} \tfrac{1}{\beta} a_j e^{-t/\beta}\exp\big(-Ae^{-t/\beta}\big)\,dt,$$

the exponential factors having combined into one. Substituting $w = Ae^{-t/\beta}$, so that $\frac{1}{\beta}e^{-t/\beta}\,dt = -dw/A$, the integral evaluates to $(a_j/A)\int_{Ae^{-s/\beta}}^{\infty} e^{-w}\,dw = (a_j/A)\Pr[\max_i S_i \le s]$. Letting $s \to \infty$ gives $\Pr[\arg\max_i S_i = j] = a_j/A = \mathrm{softmax}(\mu/\beta)_j$, and because the joint probability factorises into that marginal times the CDF for every s, the index and the value are independent. □

A.2 THEOREM 1

Proof. Write $\mathcal{X} = \{\mathbf{x}^{(\ell)} : \mathbf{x}^{(\ell)} \supset \mathbf{u}\}$ for the finite set of length-ℓ extensions of $\mathbf{u}$, and apply Lemma 1 to the family $\{\log p(\mathbf{x}^{(\ell)}) + \beta G(\mathbf{x}^{(\ell)})\}_{\mathbf{x}^{(\ell)}\in\mathcal{X}}$, that is with locations $\mu_{\mathbf{x}^{(\ell)}} = \log p(\mathbf{x}^{(\ell)})$ and noise scale β. The perturbations are independent across $\mathcal{X}$ and $\mathcal{X}$ is finite, so the lemma applies. Its first half places the maximum at

$$\beta\log\sum_{\mathbf{x}^{(\ell)}\in\mathcal{X}} e^{\mu_{\mathbf{x}^{(\ell)}}/\beta} \;=\; \beta\log\sum_{\mathbf{x}^{(\ell)}\supset\mathbf{u}} p(\mathbf{x}^{(\ell)})^{1/\beta} \;=\; \beta\log Z_\beta^{j:\ell}(\mathbf{u})$$

by Eq.(9), with scale β, which is the first claim. Its second half gives $\arg\max_{\mathbf{x}^{(\ell)}\in\mathcal{X}} S^{j:\ell} \sim \mathrm{softmax}(\mu/\beta)$, whose $\mathbf{x}^{(\ell)}$ coordinate is $p(\mathbf{x}^{(\ell)})^{1/\beta}/Z_\beta^{j:\ell}(\mathbf{u}) = \pi_\beta^{j:\ell}(\mathbf{x}^{(\ell)} \mid \mathbf{u})$, again by Eq.(9). The independence of the two is the independence asserted by the lemma. □

A.3 PROPOSITION 1 (SUBTREE MAXIMUM)

For every prefix $\mathbf{u}$, $S(\mathbf{u}) \sim \mathrm{Gumbel}\big(\varphi_\beta(\mathbf{u}),\ \beta\big)$ *with* $\varphi_\beta(\mathbf{u}) = \log p(\mathbf{u}) + \beta\log Z_\beta(\mathbf{u})$, *and* $\arg\max_{\mathbf{x}\in\mathcal{L}(\mathbf{u})} S(\mathbf{x}) \sim \pi_\beta(\cdot \mid \mathbf{u})$.

Proof. This is Theorem 1 with ℓ equal to the full sequence length, where $\mathcal{X}$ is $\mathcal{L}(\mathbf{u})$ and $S^{j:\ell}(\mathbf{u})$ is $S(\mathbf{u})$. It remains to rewrite the two conclusions in the conditional form of Eq.(6). Factorising $p(\mathbf{x}) = p(\mathbf{u})\,p(\mathbf{x} \mid \mathbf{u})$ pulls a common $p(\mathbf{u})^{1/\beta}$ out of the sum,

$$\beta\log\sum_{\mathbf{x}\in\mathcal{L}(\mathbf{u})} p(\mathbf{x})^{1/\beta} \;=\; \log p(\mathbf{u}) \;+\; \beta\log\sum_{\mathbf{x}\in\mathcal{L}(\mathbf{u})} p(\mathbf{x} \mid \mathbf{u})^{1/\beta} \;=\; \varphi_\beta(\mathbf{u}),$$

which is the location. The same factor appears in the numerator and the denominator of the argmax law and cancels, leaving $p(\mathbf{x} \mid \mathbf{u})^{1/\beta}/Z_\beta(\mathbf{u}) = \pi_\beta(\mathbf{x} \mid \mathbf{u})$. □

A.4 LEMMA 2 (MAX-PINNED GUMBEL SCORE)

Let the children's scores be independent across $v \in \mathcal{V}$, with

$$S^{(j+1):(\ell+1)}(\mathbf{u}\oplus v) \;\sim\; \mathrm{Gumbel}\big(\beta \log Z_\beta^{(j+1):(\ell+1)}(\mathbf{u}\oplus v),\, \beta\big), \qquad Z \;=\; \max_{v\in\mathcal{V}} S^{(j+1):(\ell+1)}(\mathbf{u}\oplus v).$$

Then the max-pinned scores $\widetilde{S}^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)$ of Lemma 2 satisfy $\max_v \widetilde{S}^{(j+1):(\ell+1)}(\mathbf{u}\oplus v) = S^{j:\ell}(\mathbf{u})$ identically, and are distributed as the children's scores conditioned on that maximum.

Proof. We follow the construction of Kool et al. (2019), carried to noise scale β.

Truncated Gumbels. If $S \sim \mathrm{Gumbel}(\mu, \beta)$ then for $s \le b$

$$\Pr[S \le s \mid S \le b] \;=\; \frac{\exp\big(-e^{\mu/\beta}e^{-s/\beta}\big)}{\exp\big(-e^{\mu/\beta}e^{-b/\beta}\big)} \;=\; \exp\Big(-e^{\mu/\beta}\big(e^{-s/\beta} - e^{-b/\beta}\big)\Big), \tag{19}$$

the Gumbel distribution truncated above at b. A child $\mathbf{u}\oplus v$ has location $\beta \log Z_\beta^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)$, so the factor $e^{\mu/\beta}$ in Eq. (19) is the partition function $Z_\beta^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)$ itself. We write $F^{(b)}(s \mid \mathbf{u}\oplus v)$ for Eq. (19) read with that value.

By definition,

$$\widetilde{S}^{(j+1):(\ell+1)}(\mathbf{u}\oplus v) \;=\; -\beta \log\Big(e^{-S^{j:\ell}(\mathbf{u})/\beta} \;-\; e^{-Z/\beta} \;+\; e^{-S^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)/\beta}\Big),$$

which rearranges to the identity

$$e^{-\widetilde{S}^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)/\beta} - e^{-S^{j:\ell}(\mathbf{u})/\beta} \;=\; e^{-S^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)/\beta} - e^{-Z/\beta}. \tag{20}$$

The bracket decreases in $S^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)$ and equals $e^{-S^{j:\ell}(\mathbf{u})/\beta} > 0$ at the child attaining Z, so it is positive on the whole range, and the map $s \mapsto -\beta\log\big(e^{-S^{j:\ell}(\mathbf{u})/\beta} - e^{-Z/\beta} + e^{-s/\beta}\big)$ has derivative $e^{-s/\beta}\big/\big(e^{-S^{j:\ell}(\mathbf{u})/\beta} - e^{-Z/\beta} + e^{-s/\beta}\big) > 0$. The pin is therefore a strictly increasing bijection from $(-\infty, Z]$ onto $(-\infty, S^{j:\ell}(\mathbf{u})]$ carrying Z to $S^{j:\ell}(\mathbf{u})$. In particular it preserves the ranking of a parent's children and their argmax, while different parents carry different $S^{j:\ell}(\mathbf{u})$ and Z and so apply different maps; that is what places children of different parents on a common scale.

Condition on Z and on the token $v^\star$ attaining it. Then $S^{(j+1):(\ell+1)}(\mathbf{u}\oplus v^\star) = Z$, and the remaining children are independent with law $F^{(Z)}(\,\cdot \mid \mathbf{u}\oplus v)$, since each is a Gumbel variable conditioned to lie below the realised maximum. The pin sends $\mathbf{u}\oplus v^\star$ to $S^{j:\ell}(\mathbf{u})$. For $v \neq v^\star$ and $\tilde{s} \le S^{j:\ell}(\mathbf{u})$, monotonicity lets us pull the event back through the map, and Eq. (20) cancels the two occurrences of Z:

$$\begin{aligned}
\Pr\big[\widetilde{S}^{(j+1):(\ell+1)}(\mathbf{u}\oplus v) \le \tilde{s}\big] \;&=\; F^{(Z)}(s \mid \mathbf{u}\oplus v) \;=\; \exp\Big(-Z_\beta^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)\big(e^{-s/\beta} - e^{-Z/\beta}\big)\Big) \\
&=\; \exp\Big(-Z_\beta^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)\big(e^{-\tilde{s}/\beta} - e^{-S^{j:\ell}(\mathbf{u})/\beta}\big)\Big) \\
&=\; F^{(S^{j:\ell}(\mathbf{u}))}(\tilde{s} \mid \mathbf{u}\oplus v),
\end{aligned}$$

where s is the preimage of $\tilde{s}$. The same deterministic map acts on each child separately, so the pinned children stay independent given $v^\star$.

□

A.5 THEOREM 2

Define $\pi_\beta(\mathbf{x}^{(\ell)})$ be the power distribution over length-ℓ sequences, $\pi_\beta^{1:j}(\mathbf{x}^{(\ell)})$ be the marginal distribution of $\pi_\beta(\mathbf{x}^{(\ell)})$ at position $j < \ell$. Given the prefix $\mathbf{u} = \mathbf{x}^{(j)}, \mathbf{u} \;\sim \pi_\beta^{1:j}(\mathbf{x}^{(\ell)})$, choosing optimal $v^\star = \arg\max_{v\in\mathcal{V}} \widetilde{S}^{(j+1):(\ell+1)}(\mathbf{u}\oplus v)$ will leads a distribution of one-token expasion from $\mathbf{u}$

$$\mathbf{u}\oplus v \sim \pi_\beta^{1:(j+1)}(\mathbf{x}^{(\ell+1)}) \tag{21}$$

Proof. With the choice of $v^\star = \arg\max_{v \in \mathcal{V}} \left\{ S^{(j+1):(\ell+1)}(\mathbf{u} \oplus v) \right\}$ s.t $S^{(j):(\ell+1)}(\mathbf{u} \oplus v) = \max S^{(j+1):(\ell+1)}(\mathbf{u})$, it becomes the best candidate to approach to a new length-$(\ell+1)$ sequence $\mathbf{x}^{(\ell+1)}$. We also point out that $\mathbf{x}^{(\ell+1)}$ follows a power distribution for over length-$(\ell+1)$ sequences. So marginalizing out of sequences passing through $\mathbf{u} \oplus v^\star$ will lead to the distribution $\mathbf{u} \oplus v \sim \pi_\beta^{1:(j+1)}(\mathbf{x}^{(\ell+1)})$ □

B Additional Empirical Analysis

B.1 Compare work in (Kool et al., 2019) against our method

Kool et al. (2019) maintain the same k-beam search that we do, but they target the base model p while we target the power distribution π_β. The two samplers therefore differ in one place only, the score by which they rank prefixes, and ours is the more accurate of the two on all three models

Table 3: One-shot accuracy comparison with Stochastic Beam Search (Kool et al., 2019) on MATH500.

Model	Stochastic Beam Search (Kool et al., 2019)	PoBS, $\beta = 0.25$ (ours)
Qwen2.5-Math-7B	0.752	**0.828**
Qwen2.5-7B	0.648	**0.772**
DeepSeek-Math-7B-Instruct	0.500	**0.534**

B.2 Applying Different Voting Strategies

Table 4 compares selection rules applied to the same set of $k = 8$ candidates generated by each method on MATH500 with Qwen2.5-Math-7B. Under native ordering, deterministic beam achieves the highest initial accuracy of 0.800, while PoBS attains 0.692. This disparity arises because deterministic beam ranks candidates according to model likelihood, whereas PoBS relies on perturbed scores introduced during search. However, deterministic beam results in the lowest candidate coverage, with a Pass@8 of only 0.828, so subsequent selection rules offer only marginal improvement, increasing accuracy to 0.806. PoBS delivers the most diverse candidate set, achieving the highest Pass@8 at 0.888. When post-hoc selection rules are applied, PoBS consistently surpasses all baselines, reaching 0.810 with Best-of-N, 0.824 with majority vote, and 0.828 with weighted vote. These findings suggest that PoBS generates a high-coverage set of candidates, enabling weighted vote to identify the most accurate final answers.

Table 4: Selection rules applied to the same $k = 8$ candidates, MATH500 with Qwen2.5-Math-7B.

Sampler	Native	Best-of-N	Vote	Weighted Vote	Pass@8
Base, $\tau = 1.0$	0.502	0.736	0.774	0.780	0.885
Low-temperature, $\tau = 0.25$	0.686	0.782	0.798	0.794	0.872
MCMC power sampling	0.714	0.770	0.768	0.790	0.876
Deterministic beam	**0.800**	0.804	0.806	0.806	0.828
GRPO	0.772	0.786	0.814	0.818	0.874
PoBS, $\beta = 0.25$ (ours)	0.692	**0.810**	**0.824**	**0.828**	**0.888**

B.3 Candidate Budget Scaling

To evaluate how candidate budget scales under wider budgets on a different model architecture, Table 5 presents performance for up to $k = 16$ candidates on MATH500 using DeepSeek-Math-7B-Instruct. In this analysis, the baseline MCMC power sampling computes Pass@k across 16 independent chains, while PoBS measures coverage over candidate trajectories ranked by weighted vote. As shown in Table 5, PoBS achieves higher accuracy at all tested budgets. At $k = 1$, single-shot accuracy is 0.548 for PoBS compared to 0.474 for MCMC, while coverage rises steadily to 0.718 at $k = 4$ and reaches 0.788 at $k = 16$ in contrast to 0.744. Across this budget range,

these results suggest that PoBS scales well with candidate budget, maintaining consistent coverage improvements at larger k.

Table 5: Comparison of candidate budget scaling between MCMC power sampling and PoBS on MATH500 with DeepSeek-Math-7B-Instruct across $k \in \{1, 2, 4, 8, 16\}$.

Pass@k	MCMC power sampling	PoBS (ours)
$k = 1$	0.474	**0.548**
$k = 2$	0.550	**0.646**
$k = 4$	0.616	**0.718**
$k = 8$	0.679	**0.774**
$k = 16$	0.744	**0.788**

B.4 Beyond Sharpening: Exploring Distribution Flattening

While our primary study focuses on distribution sharpening with $0 < \beta \leq 1$ on base models, the ideal sequence-level power target also permits $\beta > 1$ in our finite-horizon setting. In this regime, the power transformation reduces relative probability differences among sequences, yielding a flatter target distribution. We consider this setting as an exploratory extension beyond the sharpening regime studied in our main analysis. Shao et al. (2024) report that reinforcement learning does not improve Pass@K in their evaluation on MATH. Motivated by this observation, we evaluate PoBS with $\beta = 1.2$ on DeepSeek-Math-7B-RL as an exploratory post-RL flattening step. As shown in Table 6, PoBS with $\beta = 1.2$ on DeepSeek-Math-7B-RL attains 0.488 one-shot accuracy and 0.738 Pass@8 on MATH500. For reference, base sampling with DeepSeek-Math-7B-Instruct attains 0.374 and 0.678, respectively. These results provide preliminary evidence that PoBS can operate effectively in the $\beta > 1$ regime, suggesting that its practical scope may extend beyond distribution sharpening.

Table 6: Exploratory MATH500 results for PoBS with $\beta = 1.2$ on DeepSeek-Math-7B-RL. Base-sampling results on DeepSeek-Math-7B-Instruct are included for reference.

Model	Sampler	One-shot	Pass@8
DeepSeek-Math-7B-Instruct	Base sampler	0.374	0.678
DeepSeek-Math-7B-RL	PoBS, $\beta = 1.2$	0.488	0.738

C Qualitative Case Studies

Table 7: Example on MATH500 when our method can solve while MCMC cannot.

Problem. Below is the graph of $y = a\sin(bx + c) + d$ for some positive constants a, b, c, and d. Find the smallest possible value of c.

```
[asy]import TrigMacros;
size(400);
real f(real x) { return 2*sin(3*x + pi) + 1; }
draw(graph(f,-3*pi,3*pi,n=700,join=operator ..),red);
trig_axes(-3*pi,3*pi,-4,4,pi/2,1); layer(); rm_trig_labels(-5,5, 2);
label("$1$", (0,1), E); label("$2$", (0,2), E); label("$3$", (0,3), E);
label("$-1$", (0,-1), E); label("$-2$", (0,-2), E); label("$-3$", (0,-3), E);
[/asy]
```

Ground truth. $c = \pi$

MCMC (answer $\frac{\pi}{2}$, incorrect; 2588 characters).	**Ours** (answer π, correct; 1521 characters).
"... **Identify the period T and the frequency b:** The period T is the length of one complete cycle of the sine wave. From the graph, we can see that the period is 2π. The frequency b is related to the period by the formula $b = 2\pi/T$. Since $T = 2\pi$, we have $b = 1$. ... the sine wave starts at $x = -\pi/2$ instead of $x = 0$ ... so $-\pi/2 + c = k\pi$ ... The smallest positive value of c occurs when $k = 0$: $c = \pi/2$. ... Let's confirm this with Python and sympy: `[...]` The output $1.5707\ldots$ is the numerical value of $\pi/2$. Therefore, the smallest possible value of c is $\boxed{\pi/2}$."	"... **Identify the period $2\pi/b$:** The period of the sine function is the distance between two consecutive peaks or troughs. From the graph, the period appears to be $2\pi/3$, so $b = 3$. ... In our function $y = 2\sin(3x + c) + 1$, the first peak occurs at $x = -\pi/6$, since $3x + c = \pi/2$ when $x = -\pi/6$. Then $3(-\pi/6) + c = \pi/2$, so $-\pi/2 + c = \pi/2$ and $c = \pi$. Therefore, the smallest possible value of c is $\boxed{\pi}$."

Table 8: Example on MATH500 where both methods solve the problem and ours uses less text.

Problem. The volume of the cylinder shown is 45π cubic cm. What is the height in centimeters of the cylinder?

```
[asy] size(120);
draw(shift(2.2,0)*yscale(0.3)*Circle((0,0), 1.2));
draw((1,0)--(1,-2)); draw((3.4,0)--(3.4,-2));
draw((1,-2)..(2.2,-2.36)..(3.4,-2));
label("$h$",midpoint((3.4,0)--(3.4,-2)),E);
draw (((2.2,0)--(3.4,0)));
label("$r=3$",midpoint((2.2,0)--(3.4,0)),N); [/asy]
```

Ground truth. $h = 5$ centimeters

MCMC (answer 5, correct; 710 characters).	**Ours** (answer 5, correct; 463 characters).
"To solve this problem, we need to use the formula for the volume of a cylinder, which is given by $V = \pi r^2 h$, where V is the volume, r is the radius, and h is the height of the cylinder. Given: the volume $V = 45\pi$ cubic cm; the radius $r = 3$ cm. We need to find the height h. First, substitute the given values into the volume formula: $45\pi = \pi(3)^2 h$. Simplify the equation: $45\pi = \pi \cdot 9 \cdot h$, $45\pi = 9\pi h$. Next, divide both sides of the equation by 9π to isolate h: $h = 45\pi/(9\pi)$, $h = 45/9$, $h = 5$. Therefore, the height of the cylinder is $\boxed{5}$ centimeters."	"To find the height of the cylinder, we can use the formula for the volume of a cylinder, which is $V = \pi r^2 h$, where V is the volume, r is the radius, and h is the height. We are given that the volume of the cylinder is 45π cubic cm, and the radius is 3 cm. Plugging these values into the formula, we have $45\pi = \pi(3^2)h$. Simplifying, we have $45\pi = 9\pi h$. Dividing both sides by 9π, we have $h = 5$. Therefore, the height of the cylinder is 5 centimeters. The answer is: 5."