Guideline Forest: Retrieval-Augmented Reasoning with Branching Experience-Induced Guidelines

Anonymous ACL submission

Abstract

Retrieval-augmented generation (RAG) has been widely adopted to ground large language models (LLMs) in external knowledge, yet it remains largely underexplored for improving reasoning. Existing methods either rely on online exploration during inference or heuristic supervision over reasoning trajectories, but they fail to effectively accumulate and reuse past reasoning experience. We propose Guideline Forest, a retrieval-augmented reasoning framework that explicitly leverages experience to guide multi-step reasoning. The framework stores high-quality, label-consistent reasoning traces as reusable memory, retrieves relevant experiences for new problems, and induces them into structured guidelines that steer reasoning and enable controlled branching and aggregation. Experiments on mathematical (GSM8K, MATH-500) and programming (MBPP, HumanEval) benchmarks demonstrate consistent improvements over strong reasoning baselines, including CoT, ReAct, ToT, FoT, and AFlow. Further analyses show that experience retrieval, guideline-induced diversity, and stepwise aggregation are key to the framework's effectiveness. Beyond single-model reasoning, Guideline Forest generalizes to enhance diverse reasoning paradigms and supports multi-model collaboration, highlighting its flexibility and scalability.

1 Introduction

A key characteristic of human reasoning is its flexibility and reliance on accumulated experience. When solving complex problems, humans rarely commit to a single fixed line of thought. Instead, they recall relevant examples, reuse effective strategies, and explore multiple reasoning paths. These paths are iteratively compared, revised, and integrated through reflection and self-correction, before converging on a coherent solution. Such non-linear reasoning enables new problems to be grounded in prior experience, yielding robust and adaptable problem solving.

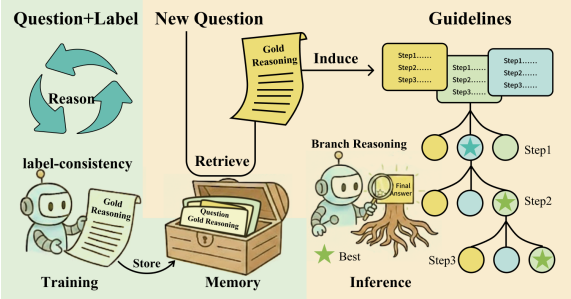

Figure 1: A conceptual illustration of our approach. The agent is trained with question–label pairs to produce high-quality, label-consistent **gold reasoning** stored in memory. When facing a new question, it retrieves relevant gold reasoning, induces **guidelines**, and solves the problem through stepwise branch reasoning.

In contrast, large language models (LLMs) (Achiam et al., 2023; Meta AI, 2024; Liu et al., 2024; Yang et al., 2024) typically generate reasoning as a single linear sequence, without explicit mechanisms to store, retrieve, or reuse prior reasoning experience. Existing approaches attempt to mitigate this limitation in two main ways. Exploration-based methods (e.g., CoT (Wei et al., 2022; Wang et al., 2022), ToT (Yao et al., 2023a), FoT (Bi et al., 2024)) generate multiple reasoning paths during inference, but rely on unguided search without leveraging accumulated experience. Workflow-oriented methods (e.g., AFlow (Zhang et al., 2024)) learn preferred reasoning procedures under supervision, yet often lack adaptability when transferred to new problems. As a result, prior reasoning is rarely reused in a systematic and structured manner.

Motivated by this gap, we propose **Guideline Forest**, a retrieval-augmented reasoning framework that grounds inference in accumulated reasoning experience. Unlike conventional RAG (Edge et al.,

*Equal contribution.

2024; Guo et al., 2024; Zhang et al., 2025b), which retrieves external knowledge, Guideline Forest retrieves and reuses *reasoning experience*. During training, we construct a memory of *label-consistent gold reasoning trajectories*. Specifically, reasoning is first generated using standard chain-of-thought prompting; when the resulting answer is incorrect, we apply *label-guided generation* to regenerate reasoning that is explicitly constrained to produce the correct label. This process yields verified, coherent reasoning traces that are stored as reusable experience.

At inference time, given a new problem, the framework retrieves relevant gold reasoning trajectories and abstracts them into structured *guidelines*. These guidelines drive *branch reasoning*, where multiple reasoning variants are expanded and refined in parallel. A stepwise aggregation mechanism compares candidate branches at each stage and selects the most promising continuation, leading to more stable and consistent multi-step reasoning. The same aggregation principle naturally extends to multi-model settings, where outputs from different models are jointly evaluated to mitigate individual biases and improve robustness.

We evaluate **Guideline Forest** on four benchmarks covering math and code reasoning (GSM8K, MATH-500, MBPP, HumanEval). Across all settings, our framework consistently outperforms strong baselines including CoT, ReAct, ToT, FoT, and AFlow, while achieving a favorable accuracy–cost trade-off through experience-induced branching and step-wise aggregation. Moreover, the induced guidelines generalize beyond our framework, improving existing reasoning methods and enabling robust multi-model collaboration. Ablation studies further validate the roles of retrieval scale, path diversity, stepwise refinement, and aggregation.

Our key contributions are as follows:

- We propose **Guideline Forest**, a retrieval-augmented reasoning framework that treats verified reasoning trajectories as reusable experience, extending RAG from knowledge retrieval to *reasoning retrieval*.

- We introduce **label-guided generation** to construct label-consistent gold reasoning, together with a structured **branch reasoning and aggregation mechanism** for robust and coherent multi-step inference.

- Through extensive experiments on math and code benchmarks, we show that **Guideline Forest** consistently outperforms strong baselines, achieves a favorable **accuracy–cost trade-off**, and naturally supports effective multi-model collaboration; ablation studies further validate the individual contributions of key components within the framework.

2 Related Work

2.1 Exploratory Reasoning

Exploratory prompting methods improve LLM reasoning by sampling multiple reasoning paths at inference time (Wei et al., 2022; Besta et al., 2024; Yao et al., 2023a; Bi et al., 2024; Yao et al., 2023b; Wang et al., 2023). CoT (Wei et al., 2022) generates explicit intermediate steps and has been widely applied to mathematical and programming tasks (Imani et al., 2023; Ke et al., 2025; Nam et al., 2024; Yang et al., 2025). ToT (Yao et al., 2023a) and FoT (Bi et al., 2024) extend CoT with tree- or forest-structured exploration, while ReAct (Yao et al., 2023b) interleaves reasoning with external actions. However, these approaches construct reasoning trajectories from scratch for each instance and rely entirely on test-time exploration. They do not reuse prior reasoning experience, leading to inefficiency and limited generalization. Our work instead grounds reasoning in reusable trajectories accumulated during training.

2.2 Search-based Reasoning

Search-based methods cast reasoning as a structured search or workflow optimization problem (Sun et al., 2024; Wu et al., 2024; Zhang et al., 2025a; Hu et al., 2024; Zhang et al., 2024). AFlow (Zhang et al., 2024) and HaIR-ICL (Wu et al., 2024) apply MCTS (Chaslot et al., 2008; Browne et al., 2012) to explore reasoning operators or strategy spaces, while WorkflowLLM (Fan et al., 2024) focuses on workflow orchestration. BoostStep (Zhang et al., 2025a) retrieves reasoning steps from a case library, and AdaS (Hu et al., 2024) allocates computation adaptively. Although effective, these methods often incur high computational cost and depend on heuristic search or handcrafted operators. In contrast, our approach learns experience-grounded reasoning guidelines from verified examples and aggregates them efficiently during inference.

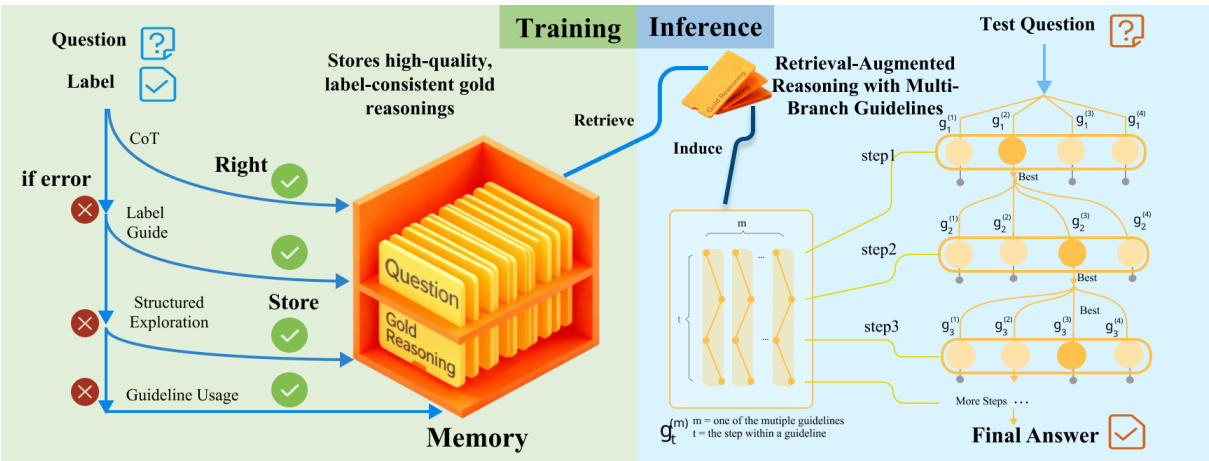

Figure 2: **Overall framework of Guideline Forest.** (**Left**) During training, the model learns to produce high-quality, label-consistent *gold reasonings* through iterative correction— first via chain-of-thought (CoT), then with label guidance, structured exploration (ToT), and memory-based guideline usage when necessary. Verified trajectories are stored in a memory repository, forming a growing collection of reusable reasoning experience. (**Right**) During inference, the system retrieves relevant gold reasonings, induces *multi-branch guidelines*, and executes them in parallel, selecting the best at each step to refine reasoning and yield a higher-quality answer.

2.3 Collaborative Inference

Collaborative inference is closely related to ensemble learning, as both improve robustness by aggregating multiple predictors or experts at inference time (Quinlan, 2014; Cortes and Vapnik, 1995; Breiman et al., 2017; Breiman, 2001; Freund et al., 1996; Breiman, 1996; Jacobs et al., 1991; Dai et al., 2024). In reasoning, this idea appears in methods such as Self-Consistency (Wang et al., 2022), Tree-of-Thought (Yao et al., 2023a), and Forest-of-Thought (Bi et al., 2024), which aggregate multiple reasoning outputs during inference, as well as in multi-LLM systems like Avengers (Zhang et al., 2025c) and SMACS (Tang et al., 2025) that leverage complementary model strengths. Different from these approaches, our method performs collaborative inference at the level of reasoning paths, guided by reusable reasoning experience rather than pure test-time sampling, enabling structured branching and aggregation across paths and models. This perspective frames our approach as a form of **branch-enhanced learning** driven by accumulated reasoning trajectories.

3 Method

3.1 Preliminaries

Multi-Step Reasoning Formulation. We study multi-step reasoning in large language models (LLMs) using latent reasoning trajectories. Given an input problem x and final answer y, the reasoning process is modeled as a sequence of intermediate steps $\mathbf{z} = (z_1, \ldots, z_T)$. The prediction is formulated as a marginalization over latent trajectories:

$$P_\theta(y \mid x) = \sum_{\mathbf{z}} P_\theta(y \mid \mathbf{z}, x)\, P_\theta(\mathbf{z} \mid x), \quad (1)$$

where $P_\theta(\mathbf{z} \mid x)$ represents trajectory generation and $P_\theta(y \mid \mathbf{z}, x)$ predicts the answer conditioned on the full reasoning. Existing methods such as CoT and ToT instantiate this formulation by sampling or searching trajectories at test time.

Trajectories vs. Guidelines. We distinguish between *reasoning trajectories* and *guidelines*. A reasoning trajectory $\mathbf{z}$ is a concrete, instance-specific sequence of inference steps. In contrast, a guideline $\mathbf{g} = (g_1, \ldots, g_T)$ is an abstract, reusable plan induced from verified trajectories. Guidelines capture general reasoning structure rather than exact solutions, and act as instructional scaffolds that guide step-wise generation.

3.2 Training: Constructing Gold Reasoning

Our training goal is to build a repository of verified *gold reasoning* trajectories that can later support guideline induction. For each supervised pair $(x^{(i)}, y^{*(i)})$, we generate candidate trajectories $\mathbf{z} \sim P_\theta(\mathbf{z} \mid x^{(i)})$ and retain only those producing the correct answer.

Label-Guided Generation. If an initial trajectory yields an incorrect prediction, we regenerate a

label-consistent explanation by conditioning on the ground-truth answer, $\mathbf{z} \sim P_\theta(\mathbf{z} \mid x^{(i)}, y^{*(i)})$, ensuring logical coherence aligned with supervision.

Structured Exploration and Memory Augmentation. To increase diversity, we optionally apply structured exploration (e.g., Tree-of-Thought) to generate multiple candidates. When verified trajectories are insufficient, we retrieve similar past cases from a memory buffer $\mathcal{M}_{\text{mem}}$, induce provisional guidelines, and generate refined trajectories conditioned on them. Only trajectories yielding the correct answer are retained.

Closed-Loop Growth. All verified trajectories are appended to memory,

$$\mathcal{M}_{\text{mem}} \leftarrow \mathcal{M}_{\text{mem}} \cup \{(x^{(i)}, \mathbf{z}^{(i)*})\}, \qquad (2)$$

forming a self-improving loop in which accumulated experience continuously expands the reasoning repository.

3.3 Inference: Guideline Induction and Multi-Branch Reasoning

At inference time, given a new input x, the model retrieves K semantically similar examples from the memory repository using embedding similarity. From their verified reasoning trajectories, it constructs a set of M abstract guidelines $\mathcal{G}_x = \{\mathbf{g}^{(1)}, \ldots, \mathbf{g}^{(M)}\}$, where each guideline represents a reusable, high-level reasoning plan induced from past successful solutions and captures shared reasoning structure rather than instance-specific details.

Guided by $\mathcal{G}_x$, reasoning proceeds in a step-wise, multi-branch manner. At each step t, each guideline independently proposes a candidate reasoning step $z_t^{(m)} \sim P_\theta(z_t \mid \hat{\mathbf{z}}_{<t}, g_t^{(m)}, x)$, which may via lightweight self-correction to obtain refined step result $\tilde{z}_t^{(m)}$. The refined candidates are then aggregated into a single unified reasoning step,

$$\hat{z}_t \sim P_\theta\big(z_t \mid \{\tilde{z}_t^{(1)}, \ldots, \tilde{z}_t^{(M)}\}, x\big), \qquad (3)$$

using confidence-based scoring or voting. The aggregated step $\hat{z}_t$ is appended to the partial trajectory and serves as shared context for the next reasoning step, ensuring that branching improves robustness without fragmenting the reasoning process.

Multi-Inference Collaboration. As an optional extension, different guideline branches can be executed by different LLMs to exploit complementary reasoning behaviors. Given the guideline set $\mathcal{G}_x$, each model ℓ follows one guideline during inference, producing a step-level candidate $\tilde{z}_t^{(\ell)} \sim P_{\theta_\ell}(z_t \mid \hat{z}_{<t}, \mathbf{g}^{(\ell)}, x)$. These candidates are aggregated by weighted voting,

$$\hat{z}_t = \arg\max_z \sum_{\ell=1}^{L} w_\ell \cdot s_\ell(z), \qquad (4)$$

where $s_\ell(z)$ denotes the confidence score of model ℓ and w_ℓ its reliability weight. This collaborative aggregation mitigates individual model bias while preserving the same step-wise inference structure.

4 Experiments

4.1 Experiment Setup

Dataset We evaluate our approach on four benchmarks: GSM8K (Cobbe et al., 2021) and MATH-500 (Lightman et al., 2023) for mathematical reasoning, and MBPP (Austin et al., 2021) and HumanEval (Chen et al., 2021) for code generation. For GSM8K, we use 500 samples for training and 1,319 for testing. For MATH-500, covering algebra, precalculus, prealgebra, counting and probability, geometry, intermediate algebra, and number theory, we randomly sample 100 training instances from each category's training set and use the full benchmark for testing. For MBPP and HumanEval, following AFlow (Zhang et al., 2024), we sample 100 and 30 training instances, with 341 and 130 reserved for testing, respectively. Valid guidelines extracted from the training samples are employed to guide model inference.

Baselines We compare against representative manually designed reasoning methods in three categories. **Direct inference** uses input–output prompting (IO) without intermediate steps. **Exploratory reasoning** includes CoT (Wei et al., 2022), CoT-SC (Wang et al., 2022), and ReAct (Yao et al., 2023b), as well as tree-structured variants ToT (Yao et al., 2023a) and FoT (Bi et al., 2024). **Search-based methods** include AFlow (Zhang et al., 2024), which leverages MCTS to discover reusable reasoning workflows.

Implementation Details We use GPT-4o-mini (OpenAI, 2024a) for reasoning and GPT-4o (OpenAI, 2024b) for verification (Figure 2). **Training:** we construct gold reasoning trajectories via CoT prompting, label guidance, and ToT-based exploration; verified correct trajectories (problem, steps, embeddings) are stored, while failed cases

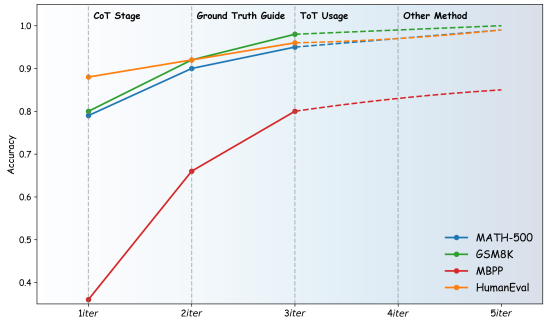

Figure 3: Illustration of the relationship between training accuracy and different stage of iterations.

are discarded. Incorrect cases are refined with label guidance using a best-of-n strategy (up to three rounds), and ToT (Yao et al., 2023a) is applied for harder instances; the resulting improvements are shown in Figure 3. We compute embeddings with text-embedding-3-small. **Inference:** we retrieve the most relevant positive sample by cosine similarity, extract high-level strategies with a Guideline Generation Prompt, rewrite them into three diverse branches, execute them step by step with per-step aggregation and refinement, and finally produce the answer with a Conclusion Prompt. We set maximum input lengths to 400 tokens for MATH-500 (Lightman et al., 2023) and GSM8K (Cobbe et al., 2021), and 600 tokens for MBPP (Austin et al., 2021) and HumanEval (Chen et al., 2021). For multi-model collaboration, we use uniform weights $w_\ell = 1$. All prompts are provided in the Appendix.

Metrics For math reasoning benchmarks (MATH-500(Lightman et al., 2023) and GSM8K(Cobbe et al., 2021)), we employ Solve Rate (%), defined as the proportion of problems for which the model produces a correct final answer, as the primary evaluation metric. For code generation benchmarks (HumanEval(Chen et al., 2021) and MBPP(Austin et al., 2021)), we adopt pass@1, denoting the fraction of problems for which a single generated solution successfully passes all test cases, to quantify model performance in code generation.

4.2 Comparison to State-of-the-art Methods

Main Results We evaluate **Guideline Forest** on two categories of multi-step reasoning tasks: *mathematical problem solving* (GSM8K, MATH-500) and *code generation* (MBPP, HumanEval). As shown in Table 1, our method achieves strong performance across all benchmarks using GPT-4o-mini, with **93.5%** on GSM8K and **69.2%** on MATH-500. On HumanEval, it attains the highest accuracy of **95.4%**. For MBPP, it achieves **81.6%**, slightly below AFlow's **83.4%**, yet outperforming other approaches by a notable margin. We attribute this to AFlow's task-specific search optimization, whereas Guideline Forest maintains higher modularity and generality across task types. These results reflect the benefit of combining two key principles: *multi-path execution* and *experience reuse*. Compared to methods that either sample multiple outputs or rely on fixed workflows, Guideline Forest induces reusable guidelines from verified trajectories and executes them in parallel with step-wise aggregation. This design supports stable, interpretable, and adaptable reasoning without requiring costly search. Across both symbolic and generative tasks, it offers a strong balance between accuracy, efficiency, and structural generalization.

Comparison with Different Base Models Table 2 reports results across multiple LLM backbones on MATH-500 and HumanEval. Performance consistently improves with model scale. Under our method, GPT-4o achieves the best results on both benchmarks (71.4% on MATH-500 and 96.2% on HumanEval), outperforming GPT-4o-mini and Qwen2.5-32B-Instruct. Other prompting methods such as CoT and FoT exhibit similar scaling trends.

Overall, Guideline Forest scales favorably with model capacity and consistently outperforms baseline reasoning methods. Notably, its advantage grows with stronger models—especially on challenging tasks like MATH-500—indicating that experience-induced guidelines and step-wise aggregation better leverage the reasoning capacity of advanced LLMs.

Computational Cost Comparison. For convenience, we report inference cost on the **Num. Th.** subset of MATH-500, as summarized in Table 3. Guideline Forest uses 12.6k tokens per problem, compared to 7.2k for ToT, 5.4k for ReAct, and 21.8k for FoT. Although GF requires more decoding than ReAct and ToT, it achieves substantially higher accuracy (85.48% vs. 75.81% and 67.74%) and uses about 40% fewer tokens than FoT while outperforming it.

To summarize inference efficiency, we adopt a simple accuracy–cost utility defined as accuracy minus a linear token penalty. Under this metric, Guideline Forest achieves the most favorable trade-off among all methods. Despite executing multiple

Table 1: Comparison with current state-of-the-art methods on mathematics and code benchmarks, where ✓ denotes support and ✓✓ indicates strong support.

Method	Multi-Path	Experience Reuse	GSM8K	MATH-500	MBPP	HumanEval	Avg.
IO	–	–	85.7	48.6	63.6	87.3	71.3
CoT	–	–	87.2	57.8	66.3	88.6	75.0
CoT-SC	✓	–	89.0	60.4	71.8	91.4	78.2
ReAct	–	–	78.2	58.0	60.0	92.0	72.1
ToT	✓	–	90.1	65.2	67.3	90.0	78.2
FoT	✓✓	✓(retrieval enhance)	90.4	65.8	69.1	93.1	79.6
AFlow	–	✓✓(heuristic search)	93.1	65.6	**83.4**	94.7	84.1
Ours	✓✓	✓✓(experience-induced)	**93.5**	**69.2**	81.6	**95.4**	**84.9**

Table 2: Comparison with state-of-the-art methods on MATH-500 and HumanEval using different LLM backbones.

Dataset	Model	IO	CoT	CoT-SC	ReAct	ToT	FoT	AFlow	Ours
MATH-500	GPT-4o-mini	48.6	57.8	60.4	58.0	65.2	65.8	65.6	**69.2**
	GPT-4o	51.2	60.4	63.2	54.8	67.6	68.4	66.6	**71.4**
	Qwen2.5-32B-Instruct	50.0	59.0	61.4	56.4	66.2	66.6	64.4	**67.6**
HumanEval	GPT-4o-mini	87.0	88.6	91.6	92.0	90.0	93.1	94.7	**95.4**
	GPT-4o	93.9	93.1	94.7	95.3	92.5	94.4	96.2	**98.0**
	Qwen2.5-32B-Instruct	89.0	89.5	90.3	93.5	91.4	93.7	94.5	**95.1**

Table 3: Inference cost and accuracy–cost utility comparison on **Num. Th.** from MATH-500. Acc–1kTok is defined as Acc $-$ Tok$/1000$ with a fixed token penalty. T denotes the number of reasoning steps, M the number of parallel candidates per step, and K the number of parallel trees used in FoT.

Method	Tokens	Acc. (%)	Acc–1kTok	Comp.
ReAct	**5.4k**	67.74	62.34	$\mathcal{O}(T)$
ToT	7.2k	75.81	68.61	$\mathcal{O}(MT)$
FoT	21.8k	83.87	62.07	$\mathcal{O}(KMT)$
Ours	12.6k	**85.48**	**72.88**	$\mathcal{O}(MT)$

branches, inference remains lightweight, as only a small number ($\sim$3) of experience-guided paths are evaluated and aggregated at each step. As a result, inference cost scales linearly with the number of branches and reasoning steps, i.e., $\mathcal{O}(MT)$.

4.3 Ablation Studies on Mechanism

Retrieval Scale To construct a guideline for each test sample, we retrieve top-k similar instances from a gold reasoning set based on cosine similarity. The solution steps of these retrieved samples are aggregated to form the guideline. Varying the value of k directly impacts the quality and specificity of the resulting guideline. As shown in As shown in Figure 4(a), aggregating three positive samples yields the best average performance. We find that too few samples produce overly specific reasoning paths that may misalign with the test instance, while too many introduce generalization noise that undermines inference precision. In contrast, three positive samples strike a favorable balance between preserving domain-specific structure and avoiding noise. It shows that fewer but more focused examples, the guideline retains more domain-specific structure, leading to stronger inductive signals.

Path Diversity We compare single-path reasoning with a multi-path setting, where a guideline is rewritten into multiple variants that are executed independently and aggregated. As shown in Figure 4(b) and Table 4, multi-path reasoning consistently improves accuracy across math domains (e.g., from 41.66% to 53.66% on Geometry). This indicates that moderate path diversity mitigates local reasoning errors while maintaining overall consistency. Although derived from the same guideline, diverse rewrites introduce complementary perspectives, and aggregation yields an ensemble-like robustness effect.

Stepwise Refinement We introduce a step-level refinement mechanism to revise intermediate outputs before proceeding. As shown in Figure 4(c)

Table 4: Ablation study on reasoning configurations across various mathematical subdomains.

Method	Algebra	Precal.	Prealg.	Count/Prob.	Geom.	Int. Alg.	Num. Th.	Acc
w/o multi-guideline	84.68	35.71	73.17	**76.32**	41.66	39.18	79.03	61.39
w/o aggregation	86.29	32.13	70.73	**76.32**	41.34	41.42	85.48	62.19
w/o refinement	86.29	35.71	76.83	73.68	43.90	46.39	85.48	64.63
Full (Ours)	**87.10**	**42.68**	**80.20**	73.68	**53.66**	**50.52**	**85.48**	**69.20**

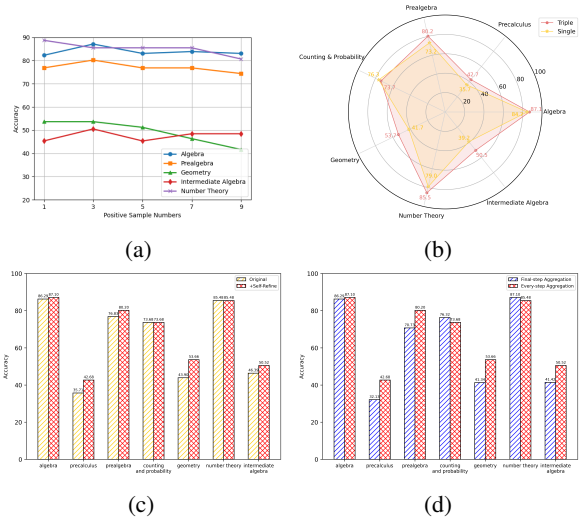

Figure 4: Ablation studies on MATH-500 dataset evaluating the effects of (a) positive sample count, (b) reasoning path count, (c) self-refinement operation, and (d) aggregation strategies.

Table 5: Guideline-enhanced performance across different reasoning methods

Method	w/o Guideline	w/ Guideline
CoT	57.8	59.8
ReAct	58.0	66.0
ToT	65.2	66.0
Ours	–	**69.8**

and Table 4, this improves average accuracy by about 4% (e.g., Prealgebra from 76.83% (w/o multi-refinement) to 80.20% (Full)). The improvement is because refinement reduces error accumulation and acts as a local optimization loop, enhancing both the fluency and factual precision of multi-step reasoning especially in longer multi-step tasks.

Aggregation Strategy We compare final-step aggregation, which merges outputs only after full reasoning, with stepwise aggregation that integrates results at each step. As shown in Figure 4(d) and Table 4, stepwise aggregation consistently performs better, yielding about 7% absolute gains. While Count/Prob. tasks remain sensitive to intermediate

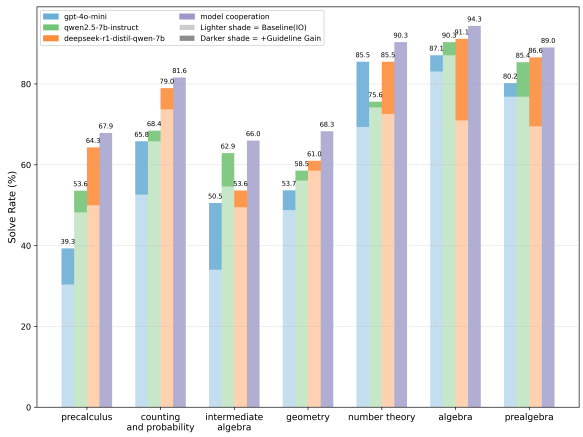

Figure 5: Comparative experiments are conducted to highlight the impact of guideline on different models and the additional benefit from model collaboration.

ambiguities, stepwise aggregation still improves overall robustness. These results show that stepwise aggregation corrects errors early, leading to more stable and accurate reasoning.

4.4 Extending the Role of Guidelines

We further examine how guidelines extend beyond individual reasoning processes—enhancing diverse reasoning paradigms and facilitating collaboration among multiple models.

Enhancing Different Reasoning Methods. To evaluate the generality of the collected guidelines, we incorporate them into three representative reasoning paradigms—CoT, ReAct, and ToT—under identical experimental settings. As shown in Table 4, adding guidelines consistently improves accuracy across all methods, with the largest gains observed for ReAct and ToT. These results indicate that the guidelines distilled from verified reasoning trajectories are not bound to any single paradigm but capture transferable heuristics that strengthen structured and goal-directed inference. Such evidence highlights their utility as reusable reasoning priors that enhance diverse inference strategies.

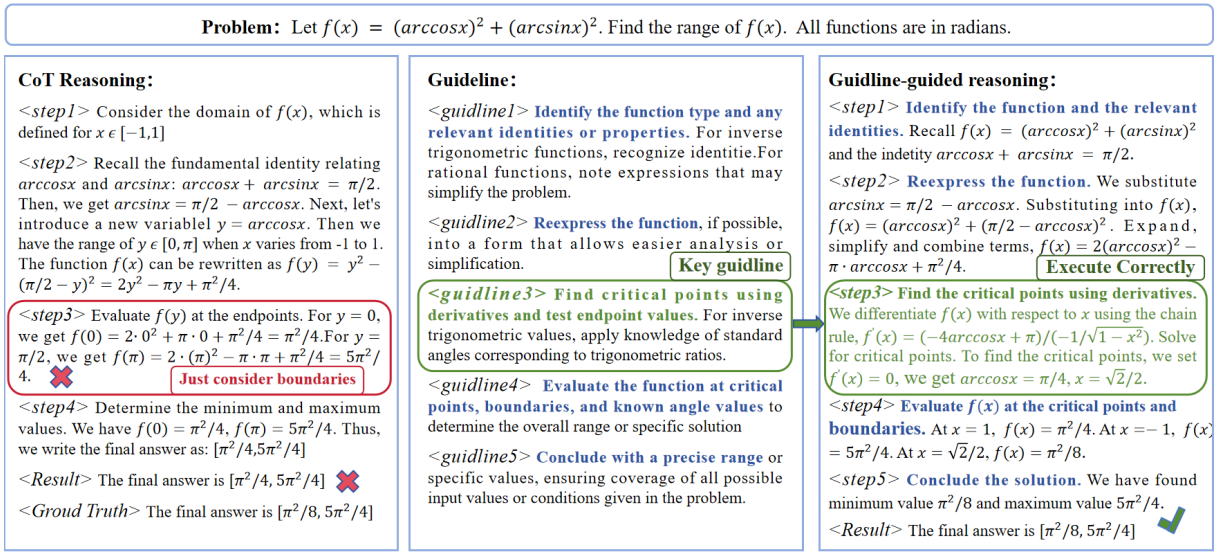

Figure 6: Qualitative experiments are conducted in mathematical scenarios and compared with the CoT method to emphasize the significant role of guideline in the reasoning process.

Multi-model Collaboration We further evaluated the effect of the collected guidelines on different models as well as the impact of model collaboration. As shown in Figure 5, across multiple mathematical domains, all three models (GPT-4o-mini, Qwen2.5-7B-Instruct, and DeepSeek-R1-Distil-Qwen-7B) achieved consistent performance improvements when applying the guidelines. Building on this, we define model cooperation as assigning the three guidelines to the three models during the reasoning phase, where each model independently generates intermediate reasoning steps following its assigned guideline. These step-level results are then aggregated to form a more robust and coherent reasoning trajectory. Experimental results show that model cooperation outperforms single-model settings across all domains and achieves the best performance of 90.32% on number theory tasks, demonstrating that guidelines not only enhance the reasoning ability of individual models but can also be further amplified through cooperation, thereby strengthening consistency and complementarity among models.

4.5 Qualitative Experimental Analysis

To illustrate how our approach influences the reasoning process, we analyze representative mathematical cases. As shown in Figure 6, Chain-of-Thought (CoT) reasoning often relies on boundary values when determining ranges, overlooking critical points such as local extrema and leading to incomplete or incorrect conclusions. In contrast, our approach retrieves structurally similar examples from a verified corpus and derives an explicit reasoning strategy that emphasizes identifying critical points via derivatives. This structured guidance clarifies the intended reasoning path, helps the model avoid common pitfalls, and results in a more principled and interpretable inference process that yields the correct solution.

Beyond this example, additional case studies in the Appendix further demonstrate that our approach consistently steers reasoning toward the correct solution space and supports iterative refinement at the step level. Across diverse cases, the model exhibits improved robustness and accuracy by progressively optimizing intermediate reasoning steps, highlighting the effectiveness of structured experience in shaping reliable reasoning behavior.

5 Conclusion

We propose **Guideline Forest**, a reasoning framework that reuses verified reasoning trajectories as structured guidelines with multi-path execution and step-wise aggregation, achieving a favorable accuracy–cost trade-off with linear inference complexity. Experiments on four math and code benchmarks show consistent gains over strong baselines, with ablations validating the roles of selective retrieval, path diversity, and aggregation. Moreover, the framework supports collaborative inference across multiple models, enabling effective integration of complementary strengths. Overall, our results show that grounding inference in structured reasoning experience leads to more robust and interpretable LLM reasoning.

Limitations

Although we explore inter-model collaboration during refinement, the design space of combining models at different scales remains underexplored. Future work may investigate more diverse and adaptive model configurations to further enhance efficiency, robustness, and scalability in practical deployments.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and 1 others. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and 1 others. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690.

Zhenni Bi, Kai Han, Chuanjian Liu, Yehui Tang, and Yunhe Wang. 2024. Forest-of-thought: Scaling test-time compute for enhancing llm reasoning. *arXiv preprint arXiv:2412.09078*.

Leo Breiman. 1996. Bagging predictors. *Machine learning*, 24:123–140.

Leo Breiman. 2001. Random forests. *Machine learning*, 45:5–32.

Leo Breiman, Jerome Friedman, Richard A Olshen, and Charles J Stone. 2017. *Classification and regression trees*. Routledge.

Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. 2012. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43.

Guillaume Chaslot, Sander Bakkes, Istvan Szita, and Pieter Spronck. 2008. Monte-carlo tree search: A new framework for game ai. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 4, pages 216–217.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, and 1 others. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20:273–297.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, and 1 others. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Shengda Fan, Xin Cong, Yuepeng Fu, Zhong Zhang, Shuyan Zhang, Yuanwei Liu, Yesai Wu, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. Workflowllm: Enhancing workflow orchestration capability of large language models. *arXiv preprint arXiv:2411.05451*.

Yoav Freund, Robert E Schapire, and 1 others. 1996. Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. Lightrag: Simple and fast retrieval-augmented generation. *arXiv preprint arXiv:2410.05779*.

Shengran Hu, Cong Lu, and Jeff Clune. 2024. Automated design of agentic systems. *arXiv preprint arXiv:2408.08435*.

Shima Imani, Liang Du, and Harsh Shrivastava. 2023. Mathprompter: Mathematical reasoning using large language models. *arXiv preprint arXiv:2303.05398*.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Zixuan Ke, Fangkai Jiao, Yifei Ming, Xuan-Phi Nguyen, Austin Xu, Do Xuan Long, Minzhi Li, Chengwei Qin, Peifeng Wang, Silvio Savarese, and 1 others. 2025. A survey of frontiers in llm reasoning: Inference scaling, learning to reason, and agentic systems. *arXiv preprint arXiv:2504.09037*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Meta AI. 2024. Introducing llama 3.1. https://ai.meta.com/blog/meta-llama-3-1/.

Daye Nam, Andrew Macvean, Vincent Hellendoorn, Bogdan Vasilescu, and Brad Myers. 2024. Using an llm to help with code understanding. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pages 1–13.

OpenAI. 2024a. Gpt-4o mini: Advancing cost-efficient intelligence. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/. Accessed: 2025-05-15.

OpenAI. 2024b. Hello gpt-4o. https://openai.com/index/hello-gpt-4o/.

J Ross Quinlan. 2014. *C4. 5: programs for machine learning*. Elsevier.

Linzhuang Sun, Hao Liang, Jingxuan Wei, Bihui Yu, Conghui He, Zenan Zhou, and Wentao Zhang. 2024. Beats: Optimizing llm mathematical capabilities with backverify and adaptive disambiguate based efficient tree search. *arXiv preprint arXiv:2409.17972*.

Shengji Tang, Jianjian Cao, Weihao Lin, Jiale Hong, Bo Zhang, Shuyue Hu, Lei Bai, Tao Chen, Wanli Ouyang, and Peng Ye. 2025. Open-source llms collaboration beats closed-source llms: A scalable multi-agent system. *arXiv preprint arXiv:2507.14200*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Zhenhailong Wang, Shaoguang Mao, Wenshan Wu, Tao Ge, Furu Wei, and Heng Ji. 2023. Unleashing the emergent cognitive synergy in large language models: A task-solving agent through multi-persona self-collaboration. *arXiv preprint arXiv:2307.05300*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Jinyang Wu, Mingkuan Feng, Shuai Zhang, Feihu Che, Zengqi Wen, and Jianhua Tao. 2024. Beyond examples: High-level automated reasoning paradigm in in-context learning via mcts. *arXiv preprint arXiv:2411.18478*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Dayu Yang, Tianyang Liu, Daoan Zhang, Antoine Simoulin, Xiaoyi Liu, Yuwei Cao, Zhaopu Teng, Xin Qian, Grey Yang, Jiebo Luo, and 1 others. 2025. Code to think, think to code: A survey on code-enhanced reasoning and reasoning-driven code intelligence in llms. *arXiv preprint arXiv:2502.19411*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023b. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Beichen Zhang, Yuhong Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Haodong Duan, Yuhang Cao, Dahua Lin, and Jiaqi Wang. 2025a. Booststep: Boosting mathematical capability of large language models via improved single-step reasoning. *arXiv preprint arXiv:2501.03226*.

Jiale Zhang, Jiaxiang Chen, Zhucong Li, Jie Ding, Kui Zhao, Zenglin Xu, Xin Pang, and Yinghui Xu. 2025b. Slimrag: Retrieval without graphs via entity-aware context selection. *arXiv preprint arXiv:2506.17288*.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, and 1 others. 2024. Aflow: Automating agentic workflow generation. *arXiv preprint arXiv:2410.10762*.

Yiqun Zhang, Hao Li, Chenxu Wang, Linyao Chen, Qiaosheng Zhang, Peng Ye, Shi Feng, Daling Wang, Zhen Wang, Xinrun Wang, and 1 others. 2025c. The avengers: A simple recipe for uniting smaller language models to challenge proprietary giants. *arXiv preprint arXiv:2505.19797*.

A Appendix

Prompt Template: CoT Reasoning

Task Description: You are a math expert skilled in step-by-step reasoning. Solve the problem methodically, clearly showing each step of your reasoning. Wrap the final answer in the format: <answer>Your Answer</answer>. Ensure your response is concise and well-structured.

Question: `<Your Question>`

Solution: `<Output Solution>`

Prompt Template: Rectify Reasoning

Task Description: You are a math expert skilled in step-by-step reasoning. Solve the problem methodically, clearly showing each step of your reasoning. Wrap the final answer in the format: <answer>Your Answer</answer>. Ensure your response is concise and well-structured.

The question which you are facing is <Your Question>
The correct answer is <Ground Truth>

Please reference the correct answer, give the correct reasoning results.

Solution: <Output Solution>

Prompt Template: Guideline Generation Prompt

Task Description: You're an expert at math reasoning. You need to extract the core of a problem and chain of thought from examples as concise as possible. You need to complete the below tasks:
1. Observe the Positive Samples: Carefully analyze the problem descriptions and solutions of each positive sample.
2. Summarize the Guideline: Based on these samples, summarize a general guideline as concise as possible for solving this type of problem.

Output Format
Problem: <core problem>
Chain of Thought:
<step 1>: [First step of the guideline]
<step 2>: [Second step of the guideline]
<step 3>: [Third step of the guideline]

Please format your summary using the above step-by-step structure. Ensure that each step is as specific and concise as possible and directly addresses the problem.

Prompt Template: Guideline Rewriting Prompt

Task Description: You are a math expert and kill at reasoning. You need to complete the below tasks: 1.Rewrite the Guideline from Different Perspectives: Provide multiple versions of the guideline, each emphasizing a different aspect or approach.
2. Maintain the Same Structure: Ensure each version has the same number of steps and maintains the logical order of the original guideline.
3. Ensure Clarity and Conciseness: Make sure each version is clear and to the point.

Original Problem: <Your Question>

Original Guideline: <Your Chain of Thought>

Output Format:
New Guideline:
<step 1>: [First step of the guideline]
<step 2>: [Second step of the guideline]
<step 3>: [Third step of the guideline]

Prompt Template: Guideline Execution Prompt

Task Information: You are a math expert. Your task is to solve a given problem by executing a guideline provided for each step to reach the correct answer.

Question: <Your Question>

Previous analysis: <Previous Steps>

Execute the guideline: <Guideline of Current Step>. Please provide a concise result.

Prompt Template: Aggregation Mechanism Prompt

Task Description: You are an expert in reasoning and will choose the best result from the given candidates.

Question: <Your Question>

Answers List: <Answer List of Current Step>

Select the most logically sound and informative answer. Finally, directly return <answer is the best>, in which is 1, 2, or 3.

Prompt Template: Refine Mechanism Prompt

Task Description: You are an expert in this task and will refine the response with precise corrections. Analyze the result carefully and identify any logical errors, inconsistencies, or missing details. If needed, provide a refined and corrected version of the result.

Question: <Your Question>

Current Step: <Current Step>

Result: <Your Result>

Prompt Template: Conclusion Prompt

Task Information: You are an expert in this task and please provide a structured final conclusion based ob previous analysis.

Previous Analysis
You have completed an analysis for the question: <Your Question>

Here are the analysis steps performed: <Previous Steps>

Based on the above analysis, summarize the final causal conclusion in a short, clear and structured manner.

Output format
<The final answer is: answer>

Case Study: A Case on MATH-500 Dataset

Question: Convert the point $(0, 3)$ in rectangular coordinates to polar coordinates. Enter your answer in the form (r, θ), where $r > 0$ and $0 \leq \theta < 2\pi$.

Similarity Score: [0.7248 0.7160 0.6174]

Positive Samples:

Positive Sample 1:
Question: Convert the point $(-2, -2\sqrt{3}, -1)$ in rectangular coordinates to cylindrical coordinates. Enter your answer in the form (r, θ, z), where $r > 0$ and $0 \leq \theta < 2\pi$.
Solution:
<step> To convert from rectangular coordinates (x, y, z) to cylindrical coordinates (r, θ, z), we use the following formulas:

- $r = \sqrt{x^2 + y^2}$
- $\theta = \tan^{-1}\left(\frac{y}{x}\right)$
- z remains the same as in rectangular coordinates.

Case Study: A Case on MATH-500 Dataset

<step> Given the point $(-2, -2\sqrt{3}, -1)$, we identify values for x, y, and z:

- $x = -2$
- $y = -2\sqrt{3}$
- $z = -1$

<step> First, we calculate r:

$$r = \sqrt{x^2 + y^2} = \sqrt{(-2)^2 + (-2\sqrt{3})^2} = \sqrt{4 + 12} = \sqrt{16} = 4.$$

<step> Next, we find θ:

$$\theta = \tan^{-1}\left(\frac{y}{x}\right) = \tan^{-1}\left(\frac{-2\sqrt{3}}{-2}\right) = \tan^{-1}(\sqrt{3}).$$

Since $\tan^{-1}(\sqrt{3})$ corresponds to an angle of $\frac{\pi}{3}$, we need to determine the correct quadrant.
Given both x and y are negative, the point is in the third quadrant. Therefore, we add π to $\frac{\pi}{3}$:

$$\theta = \frac{\pi}{3} + \pi = \frac{\pi}{3} + \frac{3\pi}{3} = \frac{4\pi}{3}.$$

<step> Lastly, z stays the same:

$$z = -1.$$

<step> Thus, the cylindrical coordinates are:

$$(r, \theta, z) = (4, \frac{4\pi}{3}, -1).$$

<The final answer is: $(4, \frac{4\pi}{3}, -1)$**>**
Positive Sample 2: ...
Positive Sample 3: ...
Guideline:

Core Problem: The core problem is converting a point from one coordinate system to another (rectangular to cylindrical, rectangular to spherical, or adjusting spherical coordinates for standard representation)

Reasoning Steps:
<step 1> Identify the given point's current coordinate system and values.
<step 2> Apply the appropriate formulas for conversion:

- For cylindrical: $r = \sqrt{x^2 + y^2}$, $\theta = \tan^{-1}(y/x)$, and z remains unchanged.
- For spherical: $\rho = \sqrt{x^2 + y^2 + z^2}$, $\theta = \tan^{-1}(y/x)$, $\phi = \cos^{-1}(z/\rho)$.

<step 3> Ensure all angles are in their respective domains:

- $0 \leq \theta < 2\pi$
- For ϕ in spherical coordinates: $0 \leq \phi \leq \pi$.

<step 4> Adjust angles if necessary:

- Correct θ based on quadrant, or adjust ϕ if beyond standard range.

Rewritten Guideline:

Core Problem: The core problem is converting a point from one coordinate system to another (rectangular to cylindrical, rectangular to spherical, or adjusting spherical coordinates for standard representation)

Reasoning Steps:
<Step 1> Determine the coordinate system of the provided point and note down the exact values.
<Step 2> Execute the necessary mathematical transformations:

- Cylindrical conversion: Compute $r = \sqrt{x^2 + y^2}$, ascertain $\theta = \tan^{-1}(y/x)$, and retain z.
- Spherical conversion: Calculate $\rho = \sqrt{x^2 + y^2 + z^2}$, ascertain $\theta = \tan^{-1}(y/x)$, and determine $\phi = \cos^{-1}(z/\rho)$.

<Step 3> Confirm that computed angles fall within standard mathematical intervals:

- Ensure $0 \leq \theta < 2\pi$.
- For ϕ in spherical coordinates: validate $0 \leq \phi \leq \pi$.

<Step 4> Modify angles when necessary by assessing quadrant positioning or correcting ϕ to maintain standard practice.

Case Study: A Case on MATH-500 Dataset

Execution Result:

Step1 Result: The current coordinate system is the rectangular (Cartesian) coordinate system, and the provided values are the coordinates of the point: $(0, 3)$.

Refined Result: The given result correctly identifies the rectangular coordinates as $(x, y) = (0, 3)$. This statement accurately represents the initial step in the problem-solving process, where the point's coordinates are extracted for conversion. There are no logical errors or inconsistencies in this initial step.
Step2 Result: To convert the point $(0, 3)$ from rectangular coordinates to polar coordinates, we'll use the formulas for polar coordinate conversion.

- Calculate the radial distance r:
$$r = \sqrt{x^2 + y^2} = \sqrt{0^2 + 3^2} = \sqrt{9} = 3$$
- Calculate the azimuthal angle θ: Since $x = 0$, to find θ, we observe the position. The point is along the positive y-axis.
$$\theta = \frac{\pi}{2}$$

Thus, the polar coordinates of the point $(0, 3)$ are $(3, \frac{\pi}{2})$.
Refined Result: The original result is correct, and there are no logical errors or inconsistencies in the steps or the solution. The provided conclusion accurately represents the polar coordinates of the given rectangular coordinates.

Step3 Result: To convert the point $(0, 3)$ from rectangular to polar coordinates, we follow these steps:

- **Calculate r:**
$$r = \sqrt{x^2 + y^2} = \sqrt{0^2 + 3^2} = \sqrt{9} = 3$$
- **Calculate θ:** Since the point $(0, 3)$ lies on the positive y-axis, θ is $\frac{\pi}{2}$.
  Adjust θ to ensure $0 \leq \theta < 2\pi$. In this case, $\theta = \frac{\pi}{2}$ is already within the required range.

Therefore, the polar coordinates are $(r, \theta) = (3, \frac{\pi}{2})$.

Refined Result: The given analysis is mostly correct, but let's ensure each step is clearly correct and complete.(Verification process omitted due to space constraints) . Thus, the polar coordinates of the point $(0, 3)$ are $(r, \theta) = (3, \frac{\pi}{2})$. The original solution correctly calculates the polar coordinates without errors or missing details.

Step4 Result: The point $(0, 3)$ in rectangular coordinates can be converted to polar coordinates by calculating:

- **r (radius):**
$$r = \sqrt{x^2 + y^2} = \sqrt{0^2 + 3^2} = \sqrt{9} = 3$$
- θ **(angle):** Since the point lies on the positive y-axis, θ is $\frac{\pi}{2}$.

Therefore, the polar coordinates are $(3, \frac{\pi}{2})$.

Refined Result: The conversion from rectangular coordinates $(0, 3)$ to polar coordinates is mostly correct, but a deeper analysis of the approach should be conducted to ensure accuracy for edge cases like this one.

Final Result: The final answer is: The polar coordinates of the point $(0, 3)$ are $(3, \frac{\pi}{2})$.