DSG-MCTS: A Dynamic Strategy-Guided Monte Carlo Tree Search for Diversified Reasoning in Large Language Models

Rui Ha[1], Chaozhuo Li[1], Rui Pu[1], Litian Zhang[2], Xi Zhang[1], Sen Su[1†]

[1]Beijing University of Posts and Telecommunications, China
[2]Beihang University, China
{harry, lichaozhuo, puruirui, zhangx, susen}@bupt.edu.cn
litianzhang@buaa.edu.cn

Abstract

Large language models (LLMs) have shown strong potential in complex reasoning tasks. However, as task complexity increases, their performance often degrades, resulting in hallucinations, errors, and logical inconsistencies. To enhance reasoning capabilities, Monte Carlo Tree Search (MCTS) has been introduced to guide the exploration of reasoning paths in a structured manner. Despite its advantages, traditional MCTS relies on fixed reasoning strategies, limiting the diversity of reasoning paths and the coverage of the solution space. To address these limitations, we propose Dynamic Strategy-Guided MCTS (DSG-MCTS), a novel framework that dynamically integrates multiple reasoning strategies, such as abductive and analogical reasoning, to expand the reasoning space. At the same time, DSG-MCTS enhances reasoning efficiency through a dynamic strategy selection mechanism that adapts to the task context. Experimental results on challenging reasoning benchmarks demonstrate that DSG-MCTS achieves improved accuracy and efficiency, outperforming existing state-of-the-art methods.

1 Introduction

Large language models (LLMs) have achieved impressive results in tasks such as mathematical reasoning, code generation, and complex planning (Anil et al., 2023; Zhao et al., 2024; Parmar et al., 2024; Ahn et al., 2024; Li et al., 2025; Wang et al., 2025), but their auto-regressive nature still leads to error accumulation and consistency issues in multi-step reasoning (Sprague et al., 2024; Li et al., 2023). To mitigate this, researchers have introduced heuristic search mechanisms such as Monte Carlo Tree Search (MCTS) to enhance reasoning through planning and path exploration (Hao et al., 2023a; Sun et al., 2024).

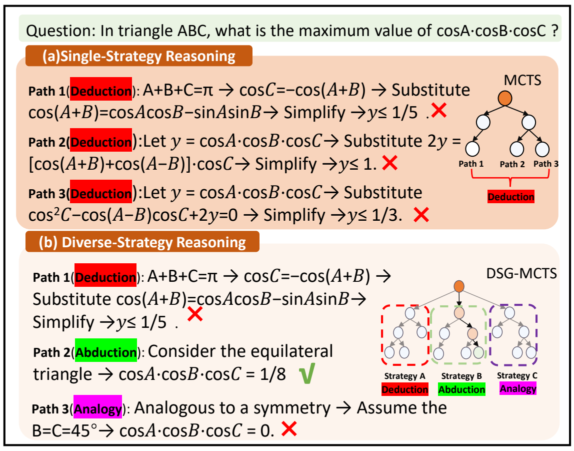

Question: In triangle ABC, what is the maximum value of cosA·cosB·cosC ?
(a)Single-Strategy Reasoning
Path 1(Deduction): A+B+C=π → cosC=−cos(A+B) → Substitute cos(A+B)=cosAcosB−sinAsinB→ Simplify →y≤ 1/5 . ✗
Path 2(Deduction):Let y = cosA·cosB·cosC→ Substitute 2y = [cos(A+B)+cos(A−B)]·cosC→ Simplify →y≤ 1. ✗
Path 3(Deduction):Let y = cosA·cosB·cosC→ Substitute cos²C−cos(A−B)cosC+2y=0 → Simplify →y≤ 1/3. ✗
MCTS
Path 1
Path 2
Path 3
Deduction
(b) Diverse-Strategy Reasoning
Path 1(Deduction): A+B+C=π → cosC=−cos(A+B) → Substitute cos(A+B)=cosAcosB−sinAsinB→ Simplify →y≤ 1/5 . ✗
Path 2(Abduction): Consider the equilateral triangle → cosA·cosB·cosC = 1/8 ✓
Path 3(Analogy): Analogous to a symmetry → Assume the B=C=45°→ cosA·cosB·cosC = 0. ✗
DSG-MCTS
Strategy A
Deduction
Strategy B
Abduction
Strategy C
Analogy

Figure 1: The comparison of reasoning answers between traditional MCTS and our proposed DSG-MCTS.

The core objective of MCTS-based methods is to conduct a thoughtful search for a broader range of solutions. The lack of diversity in the exploration limits their potential for further development. As shown in Figure 1(a), when solving the extreme value problem in trigonometric functions, traditional MCTS constructs a search tree with three solution paths. However, all three paths are based on the same deductive reasoning strategy. Specifically, the paths all start from the premises, apply the properties of triangles, and progressively reason towards the final solution. The deductive reasoning strategy leads the model into redundant trigonometric calculation traps, with the cumbersome reasoning steps often resulting in computational errors. Building on previous studies (de Freitas, 2022), extreme value problems are often solved through abduction, which uses prior knowledge and case-specific analysis to simplify complexity. In comparison, deductive reasoning relies on a rigid logical framework, limiting flexibility and narrowing the search space, which reduces its effectiveness for complex problems (Wang et al., 2024; Li et al., 2024).

The root cause of this limitation lies in the re-

[†]The corresponding author.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 10530–10544
November 4-9, 2025 ©2025 Association for Computational Linguistics

liance of existing MCTS methods on fixed action spaces to expand reasoning paths. For example, AlphaMath (Chen et al., 2024) and MindStar (Kang et al., 2024) both use predefined rules, such as generating the next reasoning step, to expand the search tree. This fixed strategy results in paths that are limited to traditional deductive reasoning during different sampling iterations. While this fixed strategy works well for simpler tasks with solutions derived through traditional deductive reasoning, it proves inadequate for more complex problems.

This limitation can be addressed by encouraging LLMs to utilize a broader range of reasoning strategies, which can enhance the diversity of their thinking. Humans employ different reasoning strategies depending on the nature of the problem at hand (Bronkhorst et al., 2020; Li et al., 2022). In addition to deductive reasoning, strategies such as induction (Flach and Kakas, 2000), abduction (Balepur et al., 2024), and analogical reasoning (Veloso, 2000) can significantly broaden the scope of reasoning. For instance, when presented with a multiple-choice mathematical problem, humans often prefer to verify the correctness of the options rather than derive a solution solely from the given conditions. This approach can greatly simplify the reasoning process, reducing the computational burden and increasing efficiency.

To enhance the diversity of paths in MCTS, we introduce the concept of Diversified Thought, which incorporates a variety of reasoning strategies to improve path diversity. In contrast to the more rigid and limited Static Thought approaches, which rely on a fixed set of reasoning strategies, Diversified Thought allows for the dynamic integration of diverse reasoning strategies as structural foundations for tree construction. As shown in Figure 1(b), this method generates three distinct solution paths guided by different reasoning strategies, including an optimal solution identified through a abduction strategy, which traditional methods fail to uncover. Our dynamic strategy enhances MCTS by increasing solution space diversity and coverage, improving performance on complex tasks.

While our dynamic strategy approach alleviates the lack of diversity in reasoning paths, it also introduces new challenges. One key issue is the potential efficiency loss due to managing and selecting among multiple diverse strategies. The system must dynamically evaluate task requirements and effectively identify the most suitable strategy to avoid excessive computation overhead. Another critical challenge is ensuring that the MCTS tree expansion process consistently adheres to the chosen strategy, maintaining alignment with the intended reasoning objectives throughout the search.

To address these challenges, we propose a novel approach, Dynamic Strategy-Guided Monte Carlo Tree Search (DSG-MCTS), aimed at enhancing path diversity and problem-solving capabilities. First, we propose a Markov Decision Process (MDP)-based strategy selection mechanism that evaluates the potential benefits of various strategies before tree expansion. By prioritizing high-potential strategies and pruning low-potential branches, this mechanism reduces unnecessary computation and exploration costs. Second, we develop a strategy-driven path generation module that ensures the expanded MCTS tree adheres to the objectives of the chosen strategy. This module constrains each step of the path expansion process to align with the requirements of the current strategy. Experiments conducted on complex reasoning benchmarks demonstrate that DSG-MCTS outperforms state-of-the-art (SOTA) methods in both reasoning accuracy and inference overhead. This paper makes the following contributions:

- We propose the DSG-MCTS framework that integrates diverse reasoning strategies to enhance path diversity and expand solution space coverage.
- We design efficient dynamic strategy selection and strategy-guided path generation modules to address challenges in reasoning efficiency and diversity.
- We validate DSG-MCTS on multiple reasoning benchmarks, demonstrating its superior performance over baselines in diversity, reasoning efficiency, and accuracy.

2 Problem Definition

For a LLM π_θ parameterized by θ, solving complex problems can be formalized as a multi-step reasoning process. Specifically, given an input problem q and a predefined prompt ψ , the model generates a reasoning path $\{s_0, s_1, \ldots, s_T\}$, where $s_0 = q$ is the initial state and s_T is the final answer $a \sim \pi_\theta(\psi(q))$. Each state s_t represents an intermediate result or step within the reasoning process.

At each reasoning step t, the current state s_{t-1} comprises the original input q and all previously generated reasoning steps $\{s_1, \ldots, s_{t-1}\}$. Based

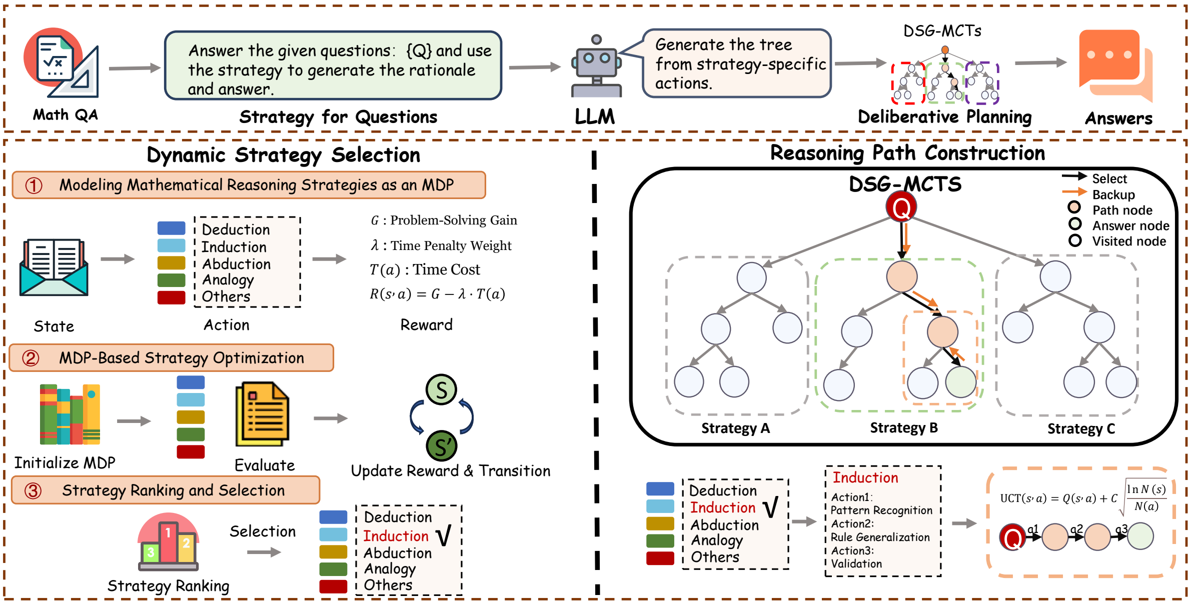

Math QA
Answer the given questions: {Q} and use the strategy to generate the rationale and answer.
Strategy for Questions
LLM
Generate the tree from strategy-specific actions.
DSG-MCTs
Deliberative Planning
Answers
Dynamic Strategy Selection
① Modeling Mathematical Reasoning Strategies as an MDP
Deduction
Induction
Abduction
Analogy
Others
G : Problem-Solving Gain
λ : Time Penalty Weight
$T(a)$: Time Cost
$R(s, a) = G - \lambda \cdot T(a)$
State
Action
Reward
② MDP-Based Strategy Optimization
Initialize MDP
Evaluate
S
S'
Update Reward & Transition
③ Strategy Ranking and Selection
1
2
3
Strategy Ranking
Selection
Deduction
Induction √
Abduction
Analogy
Others
Reasoning Path Construction
DSG-MCTS
Q
Select
Backup
Path node
Answer node
Visited node
Strategy A
Strategy B
Strategy C
Deduction
Induction √
Abduction
Analogy
Others
Induction
Action1:
Pattern Recognition
Action2:
Rule Generalization
Action3:
Validation
$\mathrm{UCT}(s, a) = Q(s, a) + C\sqrt{\frac{\ln N(s)}{N(a)}}$
Q
a1
a2
a3

Figure 2: The overview of our framework. It comprises two key steps: Dynamic Strategy Selection and Reasoning Path Construction.

on this context, the model produces an action $\alpha_t = \pi_\theta(\psi(s_{t-1}))$, which determines how the reasoning will proceed to transition from state s_{t-1} to a new state s_t. This process is repeated iteratively as the model incrementally constructs the reasoning path, terminating when a complete reasoning sequence is formed, culminating in the final answer s_T.

3 Methodology

Figure 2 illustrates the framework of the proposed Dynamic Strategy-Guided Monte Carlo Tree Search (DSG-MCTS) paradigm for solving multi-step reasoning tasks. Given the input reasoning task, the framework first employs an MDP-based strategy selection to evaluate and choose the most suitable reasoning approach. Each selected strategy guides the reasoning process, determining the steps taken to solve the problem. Subsequently, the DSG-MCTS framework integrates the selected strategies into the tree construction process. The tree expands dynamically, incorporating both the selected reasoning strategies and their associated paths. Finally, the reasoning paths are evaluated and refined, ensuring the optimal solution is achieved.

3.1 Dynamic Strategy Selection

In this subsection, we describe the process of selecting reasoning strategies for complex tasks using a Markov Decision Process (MDP). The detailed process is described as follows.

3.1.1 Modeling Reasoning Strategies

To select the optimal global reasoning strategy, we model the strategy selection process as a MDP. The MDP is defined as a five-tuple $\langle S, A, P, R, \mu \rangle$, where each component is carefully designed to capture the characteristics of reasoning tasks.

The **state space** (S) defines the context of the reasoning task, with each state $s \in S$ representing relevant task features, reasoning history, and structural information, such as variables and problem complexity in mathematical reasoning.

The **action space** (A) consists of various reasoning strategies, where each action $a \in A$ corresponds to a specific strategy. Examples include deductive reasoning, induction, abduction, analogical reasoning, which guide the reasoning process by facilitating high-level decisions.

The **state transition function** ($P(s'|s, a)$) models the probability of transitioning from state s to state s' after action a, estimated using historical data reflecting strategy impacts on state evolution.

The **reward function**, $R(s, a)$, measures the performance or effectiveness of a strategy a when executed in a specific state s. It is defined as follows:

$$R(s, a) = G - \lambda \cdot T(a), \tag{1}$$

where G is the gain, $T(a)$ is the time cost, and λ is the gain-time trade-off factor.

The **initial state distribution** (μ) represents the distribution of starting states, reflecting both problem complexity and the diversity of potential reasoning paths.

3.1.2 MDP-Based Strategy Optimization

Once the MDP is constructed, the next step is to optimize the selection of strategies to maximize cumulative rewards.In our framework, following (Liu et al., 2024), we employ a policy gradient-based reinforcement learning approach to train a policy network $\pi_\theta(a|s)$, which predicts the probability distribution over strategies a given a state s. The detailed process can be described as follows:

State Encoding: The current state s is represented as a high-dimensional feature vector, incorporating task-specific features and the historical success of strategies.

Action Selection: The policy network computes a probability distribution over strategies, normalizing scores with the softmax function:

$$\pi_\theta(a|s) = \frac{\exp(f_\theta(s, a))}{\sum_{a' \in A} \exp(f_\theta(s, a'))}, \tag{2}$$

where $f_\theta(s, a)$ is the score of action a for state s, produced by the policy network.

Reward Optimization and Policy Update: After executing a strategy, the reward $R(s, a)$ is computed based on its performance. The goal of reinforcement learning is to maximize the expected cumulative reward $J(\pi_\theta)$(Prajapat et al., 2024).

$$J(\pi_\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=1}^{T} R(s_t, a_t) \right]. \tag{3}$$

The policy is updated using the policy gradient method:

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_\tau \left[\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot R(s_t, a_t) \right]. \tag{4}$$

To encourage exploration, an entropy regularization term is added to the objective:

$$J'(\pi_\theta) = J(\pi_\theta) + \beta H(\pi_\theta), \tag{5}$$

where $H(\pi_\theta) = -\sum_{a \in A} \pi_\theta(a|s) \log \pi_\theta(a|s)$.

3.1.3 Strategy Ranking and Selection

After training the policy network, strategies are ranked and selected based on their predicted performance, with the most effective ones being chosen.

Strategy Scoring: The policy network assigns a score to each strategy based on its predicted effectiveness for a given state.

Prioritization: Strategies are ranked by their scores, and the top-ranked strategy is selected as the primary option. If the primary strategy fails, a fallback strategy is selected from the remaining candidates.

Validation: The chosen strategy is executed, and if its performance is suboptimal, the system dynamically switches to the next candidate strategy.

This MDP-based strategy selection process ensures the selection of the most optimal strategy for each reasoning task, thereby enhancing both the accuracy and efficiency of the problem-solving process.

3.2 Reasoning Path Construction

In our proposed Dynamic Strategy-Guided MCTS (DSG-MCTS) framework, MCTS is enhanced by integrating globally selected reasoning strategies to dynamically construct reasoning paths. This section provides a detailed explanation of the four core phases of MCTS: Selection, Expansion, Simulation, and Backpropagation, and explores how each phase integrates reasoning strategies and diversity constraints to optimize the construction of solution paths.

Selection Phase: Balancing Exploration and Exploitation The selection phase begins at the root node of the search tree and recursively selects child nodes until a leaf node is reached. Candidate actions at each node are evaluated using the UCT formula (Kocsis and Szepesvári, 2006).

$$\text{UCT}(s, a) = Q(s, a) + C\sqrt{\frac{\ln N(s)}{N(a)}}, \tag{6}$$

where $Q(s, a)$ represents the cumulative reward for taking action a in state s, $N(a)$ denotes the number of times action a has been selected, $N(s)$ refers to the total number of visits to state s, and C is the exploration constant.

To further integrate the influence of global reasoning strategies, DSG-MCTS introduces a *Strategy Alignment Bonus (SAB)*, defined as:

$$S_{\text{align}}(s, a) = \gamma \cdot \mathbb{I}(a \in \text{Active Strategy}), \tag{7}$$

where γ is a weight parameter, and $\mathbb{I}$ is an indicator function that equals 1 if the action a is aligned with the currently active global reasoning strategy, and 0 otherwise. The overall score for each action a is then updated as:

$$\text{Score}(s, a) = \text{UCT}(s, a) + S_{\text{align}}(s, a). \tag{8}$$

This integration ensures that the selection process prioritizes high-potential nodes aligned with the global strategy while maintaining sufficient exploration of alternative paths.

Expansion Phase: Strategy-Guided Action Selection. Upon reaching a leaf node, the expansion phase generates new child nodes based on the current state and the active global reasoning strategy. The core objective of this phase is to dynamically select actions that align with the strategy while maintaining computational efficiency.

To optimize the expansion process, a utility score $U(a)$ is assigned to each candidate action a:

$$U(a) = R(a) - C(a). \tag{9}$$

Here, $R(a)$ represents the predicted reward of the action, $C(a)$ denotes its computational cost.

Simulation Phase: Strategy-Aware Path Generation. Once new nodes are expanded, the simulation phase proceeds to complete the reasoning path from the current node to a terminal state. The goal of this phase is to evaluate the quality of the terminal state and the associated path. In DSG-MCTS, simulations are guided by the active global strategy.

The reward for path p is computed as:

$$R_{\text{sim}}(p) = \mathbb{I}(\text{Correct}) - \text{Cost}(p) + \text{Diversity}(p, \mathcal{P}), \tag{10}$$

where $\mathbb{I}(\text{Correct})$ is a binary indicator of whether the terminal state is correct, $\text{Cost}(p)$ measures the total computational cost of the path, $\text{Diversity}(p, \mathcal{P})$ quantifies the dissimilarity between path p and existing paths in the set $\mathcal{P}$.

Backpropagation Phase: Reward Optimization. The backpropagation phase updates the cumulative rewards and visit counts of all nodes along the path from the terminal node back to the root. The update formula for the cumulative reward $Q(s, a)$ at a node s for action a can be defined as follows:

$$Q_{\text{new}}(s, a) = (1 - \eta) \cdot Q_{\text{old}}(s, a) + \eta \cdot R_{\text{sim}}(p). \tag{11}$$

Here, η represents the learning rate, which determines the influence of the newly simulated reward. This process strengthens the nodes associated with successful paths, effectively steering the search toward promising solution regions.

4 Experiment

4.1 Experimental Settings

Datasets. We evaluate our framework on four reasoning datasets that encompass diverse reasoning tasks. These include GSM8K (Cobbe et al., 2021) and SVAMP (Patel et al., 2021a) for arithmetic reasoning, MATH (Patel et al., 2021b) for complex mathematical reasoning, AMC 2023 (AI-MO, 2024) and AIME 2024 (MAA Committees) are also included for more challenging mathematical reasoning, BBH (Suzgun et al., 2022) and MMLU (Hendrycks et al., 2021) for multitask reasoning, and StrategyQA (Geva et al., 2021) for multi-hop commonsense reasoning. These datasets span a broad range of reasoning problems, including multi-step arithmetic operations, complex algebraic calculations, and implicit multi-hop tasks.

Baselines. We compare our proposed DSG-MCTS framework with several advanced reasoning methods and closed-source large language models (LLMs). Specifically, the baselines include prompting-based methods such as CoT+SC (Wang et al., 2023) and MCTS-based methods including RAP (Hao et al., 2023b), BEATS (Sun et al., 2024), AlphaMath (Chen et al., 2024), and MindStar (Kang et al., 2024). For closed-source models, we evaluate Claude-3.5-Sonnet (Anthropic, 2024), GPT-4 (OpenAI et al., 2024), GPT-4o, and GPT-4o mini (OpenAI, 2024) to ensure a robust performance comparison against state-of-the-art reasoning systems.

Models. To evaluate the performance of DSG-MCTS, we leverage four open-source instruction-tuned models: Qwen2-7B-Instruct (Yang et al., 2024), Qwen2.5-7B-Instruct (Qwen Team, 2024), Llama-3-8B-Instruct (Grattafiori et al., 2024), Llama-3.1-8B-Instruct (Meta AI, 2024) and Deepseek-R1-7B (DeepSeek-AI et al., 2025). With parameter sizes ranging from 7 billion to 8 billion, these models are widely recognized for their effectiveness in instruction-following and reasoning tasks, ensuring that our experiments are both representative and credible.

Evaluation Metrics. We evaluate the proposed method using two key metrics: Accuracy(ACC) and Generation Length(LEN). ACC is calculated as $\text{ACC} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\{\mathcal{M}(\mathcal{LLM}(x_i)) = y_i\}$, where x_i is the input question, y_i is the ground-truth answer, $\mathcal{LLM}(\cdot)$ denotes the model's output, $\mathcal{M}(\cdot)$ extracts the predicted answer according to a predefined format (e.g., starting with "The answer is…"). LEN measures the average number of generated words, computed as $\text{LEN} = \frac{1}{N}\sum_{i=1}^{N} |\mathcal{LLM}(x_i)|$, where $|\cdot|$ counts the generated words. We use Levenshtein Distance and n-gram overlaps to evaluate diversity. A larger Levenshtein distance and smaller overlap indicate a more diverse path.

MODEL	SETTING	ARITHMETIC		MATH	MATH-HARD		COMMON	MULTITASK		AVERAGE
		GSM8K	SVAMP	MATH	AMC2023	AIME2024	StrategyQA	BBH	MMLU	
Qwen2-7B	CoT+SC	87.2$_{\pm 0.2}$	90.7$_{\pm 1.1}$	55.1$_{\pm 0.3}$	32.3$_{\pm 0.4}$	12.5$_{\pm 1.1}$	65.9$_{\pm 2.5}$	48.6$_{\pm 1.2}$	60.8$_{\pm 1.3}$	56.6$_{\pm 1.2}$
	AlphaMath	71.8$_{\pm 1.3}$	76.9$_{\pm 0.4}$	35.9$_{\pm 1.3}$	45.7$_{\pm 1.5}$	14.2$_{\pm 0.1}$	61.7$_{\pm 1.6}$	35.3$_{\pm 2.3}$	39.8$_{\pm 0.4}$	47.7$_{\pm 1.3}$
	MindStar	73.9$_{\pm 1.4}$	77.6$_{\pm 1.5}$	38.1$_{\pm 1.4}$	48.3$_{\pm 1.6}$	15.6$_{\pm 1.2}$	61.5$_{\pm 1.5}$	35.9$_{\pm 1.4}$	42.7$_{\pm 1.5}$	49.2$_{\pm 1.4}$
	RAP	75.3$_{\pm 0.5}$	81.1$_{\pm 1.6}$	41.2$_{\pm 1.3}$	43.5$_{\pm 0.4}$	13.1$_{\pm 1.2}$	69.2$_{\pm 2.6}$	43.1$_{\pm 1.5}$	39.9$_{\pm 1.3}$	50.8$_{\pm 1.3}$
	BEATS	81.2$_{\pm 1.6}$	88.6$_{\pm 0.7}$	58.9$_{\pm 1.5}$	53.6$_{\pm 1.6}$	16.3$_{\pm 1.3}$	67.7$_{\pm 1.3}$	43.8$_{\pm 1.5}$	65.6$_{\pm 0.6}$	59.5$_{\pm 1.5}$
	DSG-MCTS(Ours)	**91.6**$_{\pm 1.0}$	**91.9**$_{\pm 1.1}$	**63.7**$_{\pm 0.3}$	**57.2**$_{\pm 1.4}$	**23.8**$_{\pm 1.2}$	**71.9**$_{\pm 1.5}$	**49.8**$_{\pm 2.4}$	**70.3**$_{\pm 1.5}$	**65.0**$_{\pm 1.3}$
Qwen2.5-7B	CoT+SC	90.9$_{\pm 0.1}$	91.8$_{\pm 1.2}$	70.7$_{\pm 1.3}$	32.4$_{\pm 1.4}$	12.6$_{\pm 1.3}$	71.6$_{\pm 2.5}$	49.7$_{\pm 1.4}$	72.2$_{\pm 1.5}$	61.5$_{\pm 1.3}$
	AlphaMath	75.9$_{\pm 1.4}$	78.7$_{\pm 1.5}$	53.4$_{\pm 1.6}$	56.3$_{\pm 1.4}$	19.8$_{\pm 0.7}$	65.1$_{\pm 1.5}$	41.7$_{\pm 1.3}$	53.1$_{\pm 1.6}$	55.5$_{\pm 1.4}$
	MindStar	77.3$_{\pm 2.5}$	82.6$_{\pm 1.6}$	54.8$_{\pm 1.4}$	58.1$_{\pm 1.5}$	21.4$_{\pm 1.3}$	67.1$_{\pm 1.7}$	45.1$_{\pm 0.4}$	55.7$_{\pm 1.6}$	57.8$_{\pm 1.5}$
	RAP	79.6$_{\pm 0.6}$	83.8$_{\pm 0.7}$	55.1$_{\pm 1.9}$	45.9$_{\pm 1.5}$	16.8$_{\pm 1.9}$	68.9$_{\pm 1.6}$	41.9$_{\pm 1.4}$	39.8$_{\pm 1.3}$	54.0$_{\pm 1.5}$
	BEATS	84.7$_{\pm 1.7}$	90.3$_{\pm 1.8}$	68.5$_{\pm 1.9}$	52.1$_{\pm 0.6}$	20.5$_{\pm 1.1}$	69.9$_{\pm 1.7}$	45.9$_{\pm 1.6}$	44.6$_{\pm 1.5}$	59.6$_{\pm 1.7}$
	DSG-MCTS(Ours)	**93.1**$_{\pm 1.0}$	**92.2**$_{\pm 1.1}$	**78.7**$_{\pm 1.3}$	**65.8**$_{\pm 1.4}$	**31.4**$_{\pm 0.2}$	**73.8**$_{\pm 0.5}$	**51.9**$_{\pm 1.4}$	**74.1**$_{\pm 1.6}$	**70.1**$_{\pm 1.4}$
Llama-3-8B	CoT+SC	80.1$_{\pm 1.7}$	88.5$_{\pm 1.8}$	29.4$_{\pm 0.5}$	25.6$_{\pm 1.6}$	7.8$_{\pm 1.3}$	67.3$_{\pm 1.7}$	44.2$_{\pm 1.6}$	46.5$_{\pm 0.8}$	48.7$_{\pm 1.7}$
	AlphaMath	67.1$_{\pm 1.5}$	71.7$_{\pm 1.6}$	35.6$_{\pm 2.8}$	30.9$_{\pm 1.5}$	9.2$_{\pm 0.2}$	63.0$_{\pm 1.7}$	34.6$_{\pm 1.5}$	38.5$_{\pm 0.6}$	43.8$_{\pm 1.6}$
	MindStar	67.8$_{\pm 1.5}$	75.4$_{\pm 0.6}$	32.8$_{\pm 1.4}$	32.5$_{\pm 1.5}$	10.1$_{\pm 0.3}$	63.7$_{\pm 1.6}$	38.1$_{\pm 1.5}$	41.3$_{\pm 1.4}$	45.2$_{\pm 1.5}$
	RAP	81.0$_{\pm 1.7}$	84.3$_{\pm 0.8}$	19.4$_{\pm 1.7}$	28.7$_{\pm 1.6}$	8.3$_{\pm 1.3}$	69.2$_{\pm 0.7}$	44.5$_{\pm 2.6}$	45.6$_{\pm 1.7}$	47.6$_{\pm 1.7}$
	BEATS	86.3$_{\pm 1.6}$	88.9$_{\pm 0.7}$	39.1$_{\pm 1.5}$	36.8$_{\pm 1.6}$	14.7$_{\pm 1.3}$	71.7$_{\pm 1.6}$	38.0$_{\pm 1.5}$	43.7$_{\pm 1.2}$	52.4$_{\pm 1.3}$
	DSG-MCTS(Ours)	**89.5**$_{\pm 1.0}$	**92.7**$_{\pm 1.1}$	**45.3**$_{\pm 1.2}$	**39.5**$_{\pm 1.3}$	**17.2**$_{\pm 1.1}$	**74.1**$_{\pm 0.5}$	**45.5**$_{\pm 1.4}$	**48.7**$_{\pm 1.5}$	**56.6**$_{\pm 0.7}$
Llama-3.1-8B	CoT+SC	81.1$_{\pm 2.7}$	84.2$_{\pm 1.6}$	43.7$_{\pm 1.1}$	35.2$_{\pm 1.6}$	12.1$_{\pm 1.4}$	70.4$_{\pm 1.4}$	41.6$_{\pm 1.5}$	60.5$_{\pm 1.8}$	53.6$_{\pm 1.6}$
	AlphaMath	69.7$_{\pm 1.5}$	75.6$_{\pm 2.6}$	37.9$_{\pm 2.9}$	38.6$_{\pm 1.5}$	13.7$_{\pm 1.3}$	64.0$_{\pm 1.7}$	37.8$_{\pm 1.4}$	55.3$_{\pm 1.6}$	49.1$_{\pm 0.5}$
	MindStar	71.9$_{\pm 1.9}$	79.0$_{\pm 0.7}$	43.1$_{\pm 1.1}$	40.2$_{\pm 2.6}$	15.3$_{\pm 1.4}$	66.3$_{\pm 1.3}$	40.6$_{\pm 1.9}$	58.9$_{\pm 1.8}$	51.9$_{\pm 0.7}$
	RAP	82.9$_{\pm 1.5}$	89.0$_{\pm 1.6}$	22.1$_{\pm 0.9}$	31.5$_{\pm 1.5}$	9.8$_{\pm 2.3}$	73.1$_{\pm 1.7}$	39.7$_{\pm 0.5}$	48.6$_{\pm 1.6}$	49.6$_{\pm 1.4}$
	BEATS	88.1$_{\pm 1.6}$	84.9$_{\pm 1.0}$	47.5$_{\pm 1.6}$	42.9$_{\pm 1.5}$	14.2$_{\pm 1.3}$	70.7$_{\pm 1.6}$	41.3$_{\pm 1.6}$	60.3$_{\pm 1.5}$	56.2$_{\pm 0.6}$
	DSG-MCTS(Ours)	**90.2**$_{\pm 1.2}$	**94.3**$_{\pm 0.3}$	**52.7**$_{\pm 0.5}$	**48.2**$_{\pm 1.2}$	**21.7**$_{\pm 0.4}$	**73.5**$_{\pm 1.1}$	**47.7**$_{\pm 1.5}$	**65.4**$_{\pm 1.6}$	**61.7**$_{\pm 1.4}$

Table 1: Performance evaluation of DSG-MCTS on eight reasoning benchmarks. The best results in each category are highlighted in **bold**. The improvement over the best-performing baseline methods is statistically significant (significant test, $p < 0.05$). Results are reported as the mean and standard deviation over five sampling runs.

MODEL	SETTING	MATH	AIME2024
Claude-3.5	CoT	71.1	26.7
GPT-4	CoT	64.5	9.3
GPT-4o	CoT	76.6	13.4
GPT-4o mini	CoT	70.2	10.2
Qwen2.5-7B	Ours	**78.7**	**31.4**
Qwen2-7B	Ours	63.7	23.8
Llama-3-8B	Ours	45.3	17.2
Llama-3.1-8B	Ours	52.7	21.7

Table 2: Comparison with leading closed-source LLMs. The best results in each category are highlighted in bold.

Implementation Details. All experiments are conducted using the vLLM framework with a temperature of 0.9 and top-p of 0.9. The reasoning depth d is set to 5 for all tasks except for the MATH, AMC 2023, and AIME 2024 datasets, where d is increased to 8 to handle the higher complexity. To ensure reliability and statistical validity, each model and configuration is evaluated across five sampling runs, with reported performance metrics corresponding to the mean and standard deviation over these runs. To further assess the statistical significance of the performance differences between our proposed method and the baseline models, we conduct independent two-sample t-tests on the distributions of experimental results obtained from different random seeds. By applying a significance level of 0.05, we consider differences with p-values below this threshold to be statistically significant.

4.2 Main Results

Evaluation on Reasoning Benchmarks. As shown in Table 1, we evaluate DSG-MCTS on eight mainstream reasoning datasets and compare its performance with existing MCTS-based methods. DSG-MCTS consistently achieves state-of-the-art results across diverse reasoning tasks. For example, with the Qwen2.5-7B model, DSG-MCTS achieves 78.7% on MATH, 93.1% on GSM8K, and 92.2% on SVAMP, showcasing its strong problem-solving capabilities in structured mathematical reasoning. Furthermore, on the multi-task benchmark MMLU, DSG-MCTS achieves average accuracies of 74.1% using Qwen2.5-7B and 65.4% using Llama-3.1-8B, underscoring its robust adaptability across various reasoning domains. These gains can be attributed to its dynamic search-guided strategy, which enhances both reasoning efficiency and cross-domain generalization, enabling DSG-MCTS to better handle tasks of increasing complexity.

Comparison with Closed-Source Models. As shown in Table 2, we applied our DSG-MCTS method to relatively smaller open-source models and compared the results with those of closed-source models. For example, the Qwen2.5-

MODEL	SETTING	GSM8K		MATH		AMC 2023		AIME 2024		Overall	
		ACC ↑	LEN ↓	ACC ↑	LEN ↓	ACC ↑	LEN ↓	ACC ↑	LEN ↓	ACC ↑	LEN ↓
Llama3.1-8B	CoT+SC	81.1	2953	43.7	3895	35.2	6130	12.1	7155	43.0	5033
	MindStar	71.9	3257	43.1	4170	40.2	5893	15.3	8952	42.6	5568
	BEATS	88.1	3050	47.5	4855	42.9	6341	14.2	8347	49.2	5648
	DSG-MCTS(Ours)	**90.2**	**2375**	**52.7**	**3159**	**48.2**	**4622**	**21.7**	**5780**	**53.2**	**3984**
Qwen2.5-7B	CoT+SC	90.9	2373	70.7	3674	32.4	5226	12.6	6018	51.7	4323
	MindStar	77.3	3128	54.8	4319	58.1	5091	21.4	7205	52.9	4936
	BEATS	84.7	2760	68.5	3415	52.1	4738	20.5	6839	58.5	4438
	DSG-MCTS(Ours)	**93.1**	**1880**	**78.7**	**2869**	**65.8**	**3871**	**31.4**	**4696**	**67.3**	**3329**

Table 3: Comparison of reasoning accuracy and efficiency between DSG-MCTS (Ours) and baseline methods.

TASK	SETTING	Lev Distance↑	n-gram↓
GSM8K	CoT+SC	0.4914	0.3558
	MindStar	0.6211	0.2981
	BEATS	0.6974	0.2147
	Ours	**0.8097**	**0.1673**
	Ours(w/o DSG)	0.7128	0.2487
MATH	CoT+SC	0.5526	0.3123
	MindStar	0.6591	0.2616
	BEATS	0.5526	0.1948
	Ours	**0.7892**	**0.1285**
	Ours(w/o DSG)	0.5974	0.2321
AIME 2024	CoT+SC	0.3160	0.3891
	MindStar	0.4129	0.2840
	BEATS	0.4991	0.2125
	Ours	**0.6854**	**0.1139**
	Ours(w/o DSG)	0.4312	0.2748

Table 4: Diversity comparison based on Levenshtein distance and n-gram overlaps between baseline and proposed methods.

7B-Instruct model, enhanced with DSG-MCTS, achieves 78.7% accuracy on the challenging MATH benchmark, surpassing GPT-4o's 76.6%. Furthermore, on the challenging AIME 2024 dataset, it attains 31.4% accuracy, demonstrating remarkable reasoning capabilities. These results indicate that our approach effectively enhances the reasoning and problem-solving capacities of open-source models. Notably, DSG-MCTS enables the 7B-parameter Qwen2.5 model to reach performance levels comparable to powerful closed-source models like GPT-4o and Claude-3.5.

4.3 Inference Overhead

As shown in Table 3, the results demonstrate that our method consistently reduces reasoning length while improving accuracy across different models and four reasoning benchmarks with varying difficulty levels. For instance, on Llama3.1, our method decreases the generation length by an average of 29.5% compared to the second-best baseline across the benchmarks, while achieving approximately an 8% improvement in accuracy. Notably, our method exhibits more pronounced improvements on the more challenging AIME 2024 dataset. For example, with the Qwen2.5 model, our approach outperforms the second-best baseline by approximately 50%. The performance gain over the traditional CoT+SC method on this difficult dataset is even more significant. These results indicate that handling harder problems requires more diverse exploration strategies than simpler ones.

4.4 Diversity Improvement Analysis

Table 4 evaluates the diversity of generated answers by comparing our method with CoT+SC and several MCTS-based approaches. The experiments use Qwen2.5-7B as the backbone model, and diversity is measured by assessing the Levenshtein Distance and n-gram overlaps between generated solutions, averaged across the test set. As shown in the results, our method achieves significantly higher diversity compared to CoT+SC and other tree search methods. By incorporating diverse reasoning strategies, our approach enhances semantic variation and effectively reduces redundant patterns in the generated outputs, demonstrating its ability to explore a wider solution space and generate more varied reasoning paths. We also conducted an ablation study on the diversity of the Dynamic Strategy-Guided Path Construction (DSG) method. The ablation results verify that the diversity encouragement significantly improves reasoning diversity.

4.5 Ablation Study

Figure 3 shows a radar chart comparing model performance with and without the DSG module across multiple benchmark datasets on Qwen2.5-7B. The radar visualization illustrates the multi-dimensional accuracy improvements brought by the DSG module. Experimental results confirm that integrating the DSG module consistently enhances the model's

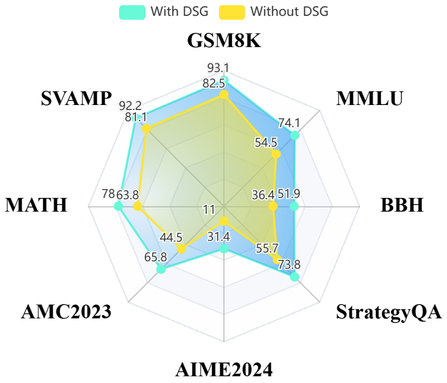


Figure 3: Ablation study on the influence of DSG on multiple datasets.

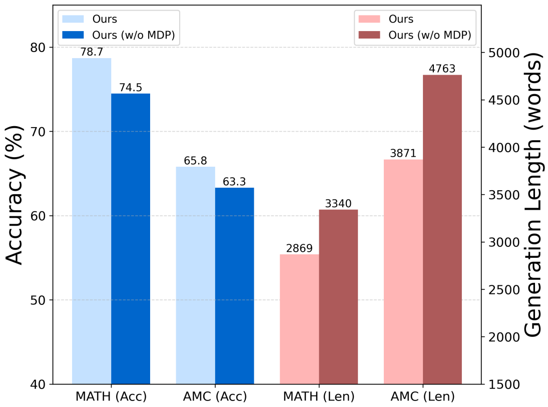


Figure 4: Ablation study results on the influence of MDP-based strategy selection module.

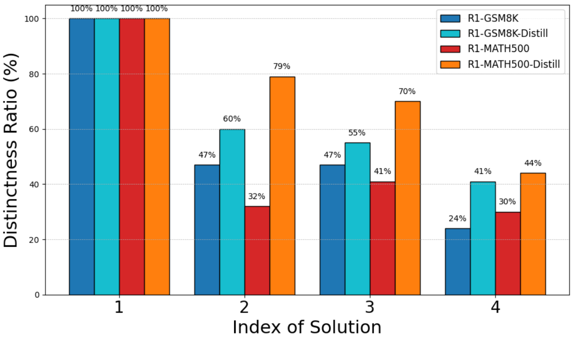


Figure 5: Ratio of whether a solution provides a new reasoning strategy for each index after fine-tuning.

reasoning capability across different tasks, expanding the coverage of the model's problem-solving space.

Figure 4 shows a bar chart comparing models equipped with and without the MDP-based Dynamic Strategy Selection module on the AMC 2023 and MATH datasets, evaluated on Qwen2.5-7B. The evaluation focuses on both accuracy and inference efficiency, measured by the generated output length (words). The results show that the MDP module enables the model to select more appropriate reasoning paths, reducing redundant steps and improving inference efficiency. Notably, we observe that the MDP module not only enhances efficiency but also brings a slight improvement in accuracy, indicating that strategy selection tailored to problem diversity contributes to improved problem-solving accuracy.

4.6 Enhancing Diversity in o1-like LLMs

We conducted a case study to evaluate the effectiveness of DSG-MCTS in enhancing the reasoning diversity of large models similar to o1. Since o1 is closed-source, we used the open-source Deepseek-R1-7b (R1) as a substitute, which exhibits slow-thinking capabilities. Experimental results showed that R1 tends to repeat reasoning paths in multi-step generation. To address this issue, we applied DSG-MCTS to generate structurally diverse slow-thinking data on the GSM8K and MATH datasets and fine-tuned R1 accordingly.

We adopted the previously proposed distinctness ratio to measure the diversity of reasoning strategies. As shown in Figure 5, after fine-tuning, the proportion of novel strategies in solutions 2 to 4 increases significantly, particularly on the MATH dataset. This demonstrates that DSG-MCTS effectively expands the reasoning space of models that have internalized slow thinking.

5 Related Work

Current research on improving diversity still faces significant limitations. While diversity-promoting prompts and increased temperature can easily generate superficially diverse outputs (Naik et al., 2024; Brown et al., 2024), they struggle to produce reasoning paths with substantial differences and high quality. Example-based methods (Yu et al., 2025) rely on a limited number of examples for guidance, and their diversity is constrained to the sampling level, lacking systematic optimization of reasoning path structures. This limitation restricts their potential in complex multi-step reasoning tasks. In contrast, we propose a diversity-driven reasoning framework based on Monte Carlo Tree Search (MCTS), which dynamically structures the search to generate multiple high-quality and substantially varied reasoning paths, effectively overcoming the limitations of superficial diversity and lack of structural optimization.

To address the above limitations, we propose a novel perspective: the empowerment of LLMs to enhance their problem-solving capabilities through the exploration and utilization of diverse reasoning strategies. Instead of confining the model to a single predefined reasoning path, this approach advocates for a dynamic and adaptable strategy.

6 Conclusion

We propose DSG-MCTS, a framework that enhances the reasoning capabilities of large language models by integrating dynamic and diverse reasoning strategies. Through dynamic strategy selection and MCTS-guided path generation, DSG-MCTS addresses the limitations of traditional MCTS methods, improving both path diversity and solution quality. Experiments on reasoning benchmarks demonstrate its state-of-the-art performance in accuracy and efficiency, highlighting its potential for tackling complex reasoning tasks.

Limitations

The DSG-MCTS method enhances reasoning capabilities through a diversity of strategies; however, there remains room for improvement in the interpretability of the reasoning process, particularly in applications where a clear understanding of model decisions is crucial. Therefore, enhancing the interpretability of the model to make reasoning paths and strategy choices more transparent will be an important area for future improvement. Moreover, while the method enhances the diversity of reasoning, the incorporation of higher-dimensional innovative thinking and adaptive strategy generation mechanisms to further augment flexibility and creativity represents a significant direction for future research.

Acknowledgements

This work is supported by the National Natural ScienceFoundation of China (62072052) and the National Key Research and Development Program of China (2024YFF0907401).

References

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. *Preprint*, arXiv:2402.00157.

AI-MO. 2024. Amc 2023.

Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, and et al. Dmitry Lepikhin. 2023. Palm 2 technical report. *Preprint*, arXiv:2305.10403.

Anthropic. 2024. Introducing claude 3.5 sonnet. Accessed: 2025-05-20.

Nishant Balepur, Abhilasha Ravichander, and Rachel Rudinger. 2024. Artifacts or abduction: How do llms answer multiple-choice questions without the question? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 10308–10330. Association for Computational Linguistics.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, et al. 2023. Graph of thoughts: Solving elaborate problems with large language models. *arXiv preprint arXiv:2308.09687*.

Hugo Bronkhorst, Gerrit Roorda, Cor Suhre, and Martin Goedhart. 2020. Logical reasoning in formal and everyday reasoning tasks. *International Journal of Science and Mathematics Education*, 18(8):1673–1694.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. 2024. Large language monkeys: Scaling inference compute with repeated sampling. *Preprint*, arXiv:2407.21787.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024. Alphamath almost zero: Process supervision without process. *Preprint*, arXiv:2405.03553.

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.

Elizabeth de Freitas. 2022. *The Role of Abduction in Mathematics: Creativity, Contingency, and Constraint*, pages 1–24. Springer International Publishing, Cham.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, and et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Shizhe Diao, Pengcheng Wang, Yong Lin, Rui Pan, Xiang Liu, and Tong Zhang. 2024. Active prompting with chain-of-thought for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1330–1350, Bangkok, Thailand. Association for Computational Linguistics.

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2022. Compositional semantic parsing with large language models. *arXiv preprint arXiv:2209.15003*.

Peter A. Flach and Antonis C. Kakas. 2000. *Abductive and Inductive Reasoning: Background and Issues*, pages 1–27. Springer Netherlands, Dordrecht.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Preprint*, arXiv:2101.02235.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, and et al. Abhishek Kadian. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. 2023a. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173, Singapore. Association for Computational Linguistics.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023b. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Jikun Kang, Xin Zhe Li, Xi Chen, Amirreza Kazemi, Qianyi Sun, Boxing Chen, Dong Li, Xu He, Quan He, Feng Wen, Jianye Hao, and Jun Yao. 2024. Mindstar: Enhancing math reasoning in pre-trained llms at inference time. *Preprint*, arXiv:2405.16265.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2022. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations*.

L. Kocsis and C. Szepesvári. 2006. Bandit based monte-carlo planning. In *Machine Learning: ECML 2006*, pages 282–293, Berlin, Heidelberg. Springer Berlin Heidelberg.

Chaozhuo Li, Pengbo Wang, Chenxu Wang, Litian Zhang, Zheng Liu, Qiwei Ye, Yuanbo Xu, Feiran Huang, Xi Zhang, and Philip S. Yu. 2025. Loki's dance of illusions: A comprehensive survey of hallucination in large language models. *CoRR*, abs/2507.02870.

Rui Li, Xu Chen, Chaozhuo Li, Yanming Shen, Jianan Zhao, Yujing Wang, Weihao Han, Hao Sun, Weiwei Deng, Qi Zhang, and Xing Xie. 2023. To copy rather than memorize: A vertical learning paradigm for knowledge graph completion. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 6335–6347. Association for Computational Linguistics.

Rui Li, Chaozhuo Li, Yanming Shen, Zeyu Zhang, and Xu Chen. 2024. Generalizing knowledge graph embedding with universal orthogonal parameterization. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Rui Li, Jianan Zhao, Chaozhuo Li, Di He, Yiqi Wang, Yuming Liu, Hao Sun, Senzhang Wang, Weiwei Deng, Yanming Shen, Xing Xie, and Qi Zhang. 2022. House: Knowledge graph embedding with householder parameterization. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 13209–13224. PMLR.

Rongxing Liu, Kumar Shridhar, Manish Prajapat, Patrick Xia, and Mrinmaya Sachan. 2024. Smart: Self-learning meta-strategy agent for reasoning tasks. *Preprint*, arXiv:2410.16128.

MAA Committees. Aime problems and solutions. `https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions`.

et al. Meta AI. 2024. Introducing llama 3.1.

Ranjita Naik, Varun Chandrasekaran, Mert Yuksekgonul, Hamid Palangi, and Besmira Nushi. 2024. Diversity of thought improves reasoning abilities of llms. *Preprint*, arXiv:2310.07088.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, and et al. Lama Ahmad. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

et al. OpenAI. 2024. Hello gpt-4o.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. 2024. Logicbench: Towards systematic evaluation of logical reasoning ability of large language models. *Preprint*, arXiv:2404.15522.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021a. Are nlp models really able to solve simple math word problems? *Preprint*, arXiv:2103.07191.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021b. Are nlp models really able to solve simple math word problems? *Preprint*, arXiv:2103.07191.

Manish Prajapat, Mojmir Mutny, Melanie Zeilinger, and Andreas Krause. 2024. Submodular reinforcement learning. In *The Twelfth International Conference on Learning Representations*.

et al. Qwen Team. 2024. Qwen2.5: A party of foundation models.

Swarnadeep Saha, Omer Levy, Asli Celikyilmaz, Mohit Bansal, Jason Weston, and Xian Li. 2023. Branch-solve-merge improves large language model evaluation and generation. *arXiv preprint arXiv:2310.15123*.

Kashun Shum, Shizhe Diao, and Tong Zhang. 2023. Automatic prompt augmentation and selection with chain-of-thought from labeled data. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12113–12139, Singapore. Association for Computational Linguistics.

Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. 2024. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. *Preprint*, arXiv:2409.12183.

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. *Preprint*, arXiv:2409.17972.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *Preprint*, arXiv:2210.09261.

Manuela Veloso. 2000. prodigy/analogy: Analogical reasoning in general problem solving. *Topics in Case-based Reasoning*.

Danqing Wang, Jianxin Ma, Fei Fang, and Lei Li. 2024. Typedthinker: Typed thinking improves large language model reasoning. *Preprint*, arXiv:2410.01952.

Pengbo Wang, Chaozhuo Li, Chenxu Wang, Liwen Zheng, Litian Zhang, and Xi Zhang. 2025. Two birds with one stone: Improving factuality and faithfulness of llms via dynamic interactive subspace editing. *CoRR*, abs/2506.11088.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, and et al. Chengpeng Li. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023a. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Yao Yao, Zuchao Li, and Hai Zhao. 2023b. Beyond chain-of-thought, effective graph-of-thought reasoning in large language models. *arXiv preprint arXiv:2305.16582*.

Fangxu Yu, Lai Jiang, Haoqiang Kang, Shibo Hao, and Lianhui Qin. 2025. Flow of reasoning:training llms for divergent problem solving with minimal examples. *Preprint*, arXiv:2406.05673.

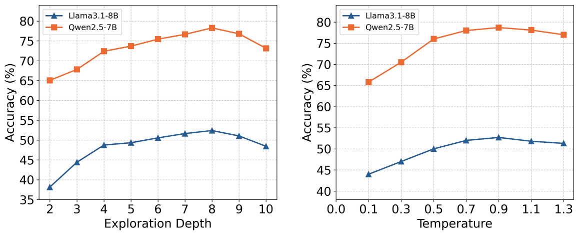
Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2023. Automatic chain of thought prompting in large language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2024. A survey of large language models. *Preprint*, arXiv:2303.18223.

A Related Work

CoT prompting methods have recently advanced the multi-step reasoning capabilities of LLMs to a remarkable degree. This can be achieved by prompting the models to generate intermediate reasoning steps before arriving at the final answer (Wei et al., 2022). Consequently, this method leads to more accurate solutions through the methodical structuring of thought patterns. Following the above initial CoT prompting work, lots of works spring up aim to improve different parts of original reasoning processing, including auto-cot (Zhang et al., 2023), self-consistency (Wang et al., 2023), active prompt (Diao et al., 2024) and automate-cot(Shum et al., 2023). These methods are effective but limited to certain tasks, as they depend on specific examples for the model to imitate, and such examples often vary significantly.

Then Least-to-Most prompting (Drozdov et al., 2022) and Decomposed prompting (Khot et al., 2022) are proposed. These methods usually emulate basic human reasoning patterns by deconstructing complex tasks into simpler, more manageable



(a) The trend of accuracy with the increasing value of exploration depth.

(b) The trend of accuracy with the increasing value of temperature.

Figure 6: Hyperparameter sensitivity analysis.

steps. However, since they fail to simulate the dynamic nature of human thought processes when confronted with varied and novel challenges, these approaches are typically rigid and lack flexibility.

At the same time, Tree-of-Thought (ToT) (Yao et al., 2023a), Graph-of-Thought (Besta et al., 2023; Yao et al., 2023b), and other related techniques like Branch-solve-merge (Saha et al., 2023) and RAP (Hao et al., 2023b) are also proposed. They are used to explore a broader range of reasoning pathways by simulating interconnected and branching thought processes. However, they are frequently complex and require manual intervention to tailor prompts for specific tasks.

B Algorithm

Algorithm 1 outlines the proposed DSG-MCTS framework for solving multi-step reasoning tasks. The algorithm is divided into three main components. First, an MDP-based strategy selection mechanism models reasoning as a Markov Decision Process, where states represent task contexts and actions correspond to reasoning strategies. A policy network is trained using reinforcement learning to select the optimal strategy based on the task at hand. Second, the selected strategy is integrated into MCTS, guiding the four phases of the search process: Selection, Expansion, Simulation, and Backpropagation. This ensures that the tree expansion aligns with the chosen strategy, incorporates diverse paths, and optimizes solution quality. Finally, the algorithm iteratively refines the selected reasoning strategy and reasoning paths until satisfactory termination criteria are achieved, ensuring a robust and efficient solution to complex reasoning tasks.

Algorithm 1 DSG-MCTS

```
Require: Initial state $s_0$, reasoning strategies $\mathcal{S}$, threshold $\epsilon$
Ensure: Optimized reasoning path and solution
  Step 1: Initialization
  Initialize search tree with root node $s_0$
  Define active reasoning strategies $\mathcal{S}_{\text{active}} \subseteq \mathcal{S}$
  Step 2: Tree Search Phases
  while termination criteria not met do
      Selection: Select a node using UCT
      Expansion: Expand the selected node based on $\mathcal{S}_{\text{active}}$
      Simulation: Simulate reasoning paths
      Backpropagation: Update node values with rewards
  end while
  Step 3: Strategy Optimization
  Use MDP to select strategies
  Reevaluate strategies if performance gap $> \epsilon$
  Output: Reasoning path and solution
```

C Hyper-parameter Analysis

DSG-MCTS relies on two critical hyperparameters: the exploration depth (d_s) and the temperature. Their influence on performance is illustrated in Figure 6.

Exploration Depth. Figure 6 (a) shows the accuracy on the MATH dataset as the exploration depth d_s varies from two to ten for both Llama3.1-8B and Qwen2.5-7B. For Llama3.1-8B, accuracy rises from approximately 38% at $d_s = 2$ to 52.7% at $d_s = 8$, then decreases to 51.0% at $d_s = 9$ and 48.4% at $d_s = 10$. For Qwen2.5-7B, accuracy increases from about 65.0% at $d_s = 2$ to 78.2% at $d_s = 8$, then falls to 76.8% at $d_s = 9$ and 73.2% at $d_s = 10$. The steady improvement up to $d_s = 8$ suggests that deeper exploration uncovers additional correct solution paths. Beyond this point, further increases in d_s lead to exploration of lower-utility branches, offering no benefit and slightly reducing performance. Therefore, setting $d_s = 8$ provides the best trade-off between accuracy and computational efficiency.

Temperature. Figure 6 (b) reports accuracy on the MATH dataset as temperature varies from 0.1 to 1.3 for both Llama3.1-8B and Qwen2.5-7B. For Llama3.1-8B, accuracy rises from 44.0% at temperature 0.1 to 52.7% at 0.9, then decreases to 51.8% at 1.1 and 51.3% at 1.3. For Qwen2.5-7B, accuracy increases from 65.8% at 0.1 to 78.7% at 0.9, then falls to 78.1% at 1.1 and 77.0% at 1.3. The steady gain up to temperature 0.9 indicates that moderate sampling randomness enhances exploration of alternative solution paths. Beyond 0.9, increased randomness injects noise without further benefits, causing performance to decline. Therefore, setting the temperature to 0.9 achieves the highest accuracy for both models while avoiding unnecessary sampling noise.

D Diversity Strategy Definitions and Action Construction in DSG-MCTS

C.1 Diversity Strategy Definitions

In DSG-MCTS, reasoning strategies (*Strategies*) serve as high-level guiding frameworks for problem-solving, determining the global generation of reasoning paths. To address various types of complex reasoning tasks, DSG-MCTS designs four core strategies, as outlined below:

- **Deduction Strategy**: This strategy derives conclusions from general principles or premises. In mathematical reasoning, for instance, it applies known axioms or theorems to deduce specific facts or solutions.
- **Induction Strategy**: This strategy generates general principles from specific instances. For example, in statistical reasoning, it identifies patterns or trends from observed data to make broader generalizations.
- **Abduction Strategy**: This strategy infers the best possible explanation for a set of observations, often under uncertainty. In diagnostic tasks, for instance, it proposes hypotheses that best explain observed symptoms or phenomena.
- **Analogy Strategy**: This strategy generates new reasoning paths by leveraging solutions to similar problems. For example, in geometry, it utilizes properties of analogous shapes to deduce solutions for the current problem.

Each strategy provides global guidance for reasoning path generation, enabling the construction of diverse reasoning paths by combining strategic goals with sub-actions.

C.2 Action Construction Based on Strategies

In DSG-MCTS, *Actions* are specific operations that execute the objectives of a reasoning strategy, dynamically expanding reasoning paths. These actions are essential for constructing and navigating the reasoning process in an efficient and organized manner. Each reasoning strategy corresponds to a set of sub-actions, which, under the guidance of the strategy, are structured and optimized as tree paths to lead toward the final conclusion or solution.

(1) Deduction Strategy Actions Deduction is a reasoning process that involves deriving specific conclusions from general principles or premises. It is a top-down approach where conclusions follow logically from established facts or laws. The sub-actions associated with Deduction ensure that the reasoning process is systematic and rooted in previously known facts. These sub-actions include:

- **Premise Identification**: In this step, we identify the foundational principles, rules, or axioms that will guide the deduction. This could involve selecting mathematical theorems, known facts, or logical principles relevant to the problem at hand. For example, in geometry, we might start by identifying the properties of triangles, such as the Pythagorean theorem.
- **Logical Application**: This sub-action involves applying the identified premises to generate new insights or conclusions. This is typically done using logical rules such as modus ponens or syllogisms. The reasoning process might involve deriving intermediate steps, each of which is valid under the rules of logic. For example, if “All mammals have hearts” and “A dog is a mammal”, we apply this logical structure to conclude that “A dog has a heart”.
- **Conclusion Validation**: After deriving the conclusions, it is necessary to validate them against the original problem or real-world context. This step ensures that the conclusions are consistent with the premises and that no logical errors have occurred. The validation could involve testing the conclusions with empirical data or checking for contradictions. For instance, in mathematical proofs, we might verify that a derived formula holds true for all edge cases.

(2) Induction Strategy Actions Induction is a reasoning process that moves from specific observations to broader generalizations or theories. It is

Type	Definition	Example
Deduction	*Derive conclusions based on established principles or premises.*	Given that "all even numbers are divisible by 2" and "12 is an even number", we can deduce that "12 is divisible by 2".
Induction	*Make generalizations based on a set of observed specific instances.*	Observing that 2, 4, 6, and 8 are all divisible by 2, we can infer that all even numbers are divisible by 2.
Abduction	*Generate a hypothesis based on available evidence and test its validity.*	If a number leaves a remainder of 1 when divided by 5, we might hypothesize that the number is of the form $5n + 1$. For example, 6, 11, 16, etc., fit this pattern.
Analogy	*Solve a problem by applying reasoning from a similar case.*	If a student can solve linear equations, they can likely also solve quadratic equations using similar methods of isolating variables. This analogy draws from the similarity between the two types of equations.

Table 5: Description of different reasoning types with mathematical examples.

typically a bottom-up approach where patterns or trends are observed from individual cases, which are then generalized to make broader conclusions. The sub-actions involved in Induction allow us to build and test hypotheses based on observed evidence. These sub-actions include:

- **Pattern Recognition**: In this phase, we observe specific instances or data points and look for recurring patterns or trends. This could involve analyzing numerical sequences, scientific data, or observations of natural phenomena. For example, if we observe that the temperatures in a city rise every summer, we might recognize a seasonal pattern.
- **Generalization**: Once patterns are identified, we move to generalizing them. This involves formulating broader theories or hypotheses that explain the observed patterns. For example, after noticing that all observed ravens are black, we might generalize that "all ravens are black", even though we haven't observed all ravens.
- **Validation**: The generalized conclusions derived through Induction must be tested for their validity. This involves applying the generalized theory to new, unobserved cases to determine whether it holds. This could involve experiments, surveys, or collecting additional data to confirm or refute the initial hypothesis. For instance, after generalizing that all ravens are black, we could attempt to find a non-black raven to test the validity of the generalization.

(3) Abduction Strategy Actions Abduction is a reasoning process that involves generating hypotheses to explain observed phenomena or events. It is often used in situations where we need to determine the most plausible explanation for incomplete or ambiguous data. Abductive reasoning is particularly useful when there are multiple possible explanations, and we must identify the one that best fits the evidence. The sub-actions involved in Abduction allow for hypothesis generation and testing. These sub-actions include:

- **Observation Analysis**: In this phase, we analyze the available data or observations to understand the underlying phenomena. This could involve identifying symptoms or evidence that point to a possible cause.
- **Hypothesis Generation**: After analyzing the observations, we generate one or more hypotheses that could explain the phenomenon. The goal here is to propose plausible explanations that fit the available data.
- **Hypothesis Testing**: The hypotheses generated need to be tested against further observations or experiments. In this phase, we try to validate the hypotheses by checking if the observed evidence is consistent with the proposed explanations.

(4) Analogy Strategy Actions Analogy is a reasoning process that constructs new reasoning paths based on the similarity between a current problem and a previous case or situation. It is widely used when direct knowledge of a problem is unavailable, and similarities to other known problems can guide the solution process. The sub-actions involved in Analogy allow for drawing parallels and making inferences based on past experiences. These sub-actions include:

- **Analog Retrieval**: In this phase, we retrieve similar problems or solutions from a knowledge base, database, or historical context. This could involve searching for previously solved problems that share similarities with the current problem. For example, if solving a new type of equation, we might retrieve solutions to similar types of equations from earlier work.
- **Solution Mapping**: After retrieving analogous problems and solutions, the next step is to map the known solutions onto the current problem. This involves identifying the relevant features of both the old and new problems and determining how the solutions can be applied to the new context. For instance, if solving a linear equation, we might map the method of isolating variables from a previous problem to the current one.
- **Analog Validation**: The final step is to test the validity of the analogy. We verify that the reasoning path derived from the analogy is applicable and correct in the new context. This involves checking whether the mapped solution correctly applies to the problem and does not overlook important differences. For example, after mapping a solution from a previous problem, we may need to verify that the same operations apply to the new problem's variables.