SEARCHING FOR REASON: A TREE-OF-THOUGHT FRAMEWORK FOR TRANSPARENT MODEL REASONING

Ahmed Mohamed
Rutgers University

Dhruvi Vinchhi
Rutgers University

Diya Patel
Rutgers University

December 12, 2025

ABSTRACT

The Tree of Thoughts (ToT) framework, introduced by Yao et al. (2023) [1], was designed to enhance the reasoning capabilities of large language models by enabling structured search over discrete units of coherent text, or "thoughts," during problem solving. Unlike traditional Chain-of-Thought approaches, ToT supports more deliberate and interpretable decision-making by exposing intermediate reasoning steps. In this paper, we explore the capabilities of the ToT framework using a novel A*-inspired search strategy, A*. Our primary objective is to demonstrate how this search paradigm provides transparency into the LLM's decision-making process, allowing researchers to precisely isolate and diagnose errors, flaws, and logical inconsistencies at the specific step where they occur. This high-resolution error signal can be used to develop more precisely targeted training and fine-tuning methods for LLMs, moving beyond general correction to surgical intervention on reasoning failures.

***K*eywords** Large Language Models · Tree of Thoughts · A* Search · Reasoning · Interpretability

1 Introduction

Large Language Models (LLMs) have shown impressive capabilities across a wide range of reasoning tasks, yet their performance often depends heavily on how reasoning steps are generated and evaluated. Traditional prompting strategies such as Chain-of-Thought (CoT) provide useful intermediate steps but lack mechanisms for revisiting earlier reasoning, exploring alternatives, and often suffer from emphasis on locally optimal decisions. As a result, CoT-style reasoning can become trapped in suboptimal trajectories or produce brittle solutions that fail under slight variations of a prompt.

The Tree of Thoughts (ToT) framework, Yao et al. (2023) [1], addresses these limitations by reframing reasoning as a search problem over discrete, interpretable "thought" states. By enabling branching, evaluation, and backtracking over these states, ToT allows LLMs to perform more deliberate, globally-coherent problem solving. However, the effectiveness of ToT strongly depends on the search algorithm used to navigate the tree. The original work explored beam/breadth-first search, and depth-first search, but these strategies may expand many low-value states or prematurely prune promising paths, especially in problems with large or uneven search spaces.

This paper proposes an an A* inspired search algorithm. Our method introduces a principled scoring mechanism that combines historical correctness of a path with a heuristic estimate of future potential, enabling the search process to prioritize states with both strong foundations and high strategic value. Unlike previous search methods, A* adaptively reallocates computational effort toward the most promising regions of the thought space, potentially mitigating error accumulation and reducing unnecessary exploration.

Our goal is to evaluate whether this search paradigm yields more efficient and reliable reasoning while preserving the interpretability that motivates the ToT framework. We formalize the scoring metrics, describe the agent-based evaluation architecture, and demonstrate how A*-style search can be naturally integrated into LLM-driven thought generation. Through these explorations, we aim to better understand how informed search techniques can enhance deliberate reasoning in modern language models.

2 Motivation

Our work on the A* search framework is driven by two primary objectives: first, to provide a diagnostic tool for understanding the limitations of large language models (LLMs) in complex mathematical reasoning; and second, to offer a practical aid for human mathematics researchers confronting challenging, research-level problems.

2.1 Diagnostic Insight into LLM Mathematical Reasoning

Despite their impressive performance on standardized tests, LLMs often exhibit brittleness and introduce subtle errors when tackling novel, multi-step mathematical problems, especially those at the research frontier. Language Models often mask the precise point of failure, making it difficult to pinpoint the boundaries of their knowledge.

The explicit search structure of A* aims to counter this. By mandating an LLM-evaluated score $g(n)$ and a heuristic value $h(n)$ for every generated thought-state, our approach transforms reasoning into a transparent, verifiable process. This allows researchers to:

- **Pinpoint Error Accumulation:** Precisely identify the step in the reasoning path where a logical fallacy or mathematical mistake was introduced.
- **Diagnose Systemic Flaws:** Understand whether the failure stems from a lack of specific domain knowledge, an inability to combine disparate concepts correctly, or an error in long-range planning (which can be assessed via the $h(n)$ heuristic).

This deep diagnostic capability can be incredibly useful for advancing LLM development, enabling the creation of targeted fine-tuning methods that move beyond general performance correction to surgical intervention on specific reasoning failure modes.

2.2 Practical Utility for Mathematics Researchers

The A* framework can serve as an effective cognitive partner for human mathematicians struggling with problems. The strength of the search lies in its ability to exhaustively explore diverse lines of reasoning and quickly prune unpromising paths based on informed scoring, a task that can be mentally exhausting for a human.

When a researcher is stuck on a problem, the system's ability to generate and evaluate a vast array of potential intermediate steps can be immensely valuable:

- **Sparking New Ideas:** The generated thought-states, even those that eventually lead to failure, expose the space of possible approaches the LLM considered. Reviewing a dozen partially correct, highly-scored paths can often reveal an unexpected step, a necessary lemma, or an analogy that may spark the human researcher's intuition toward a novel, correct approach.
- **Structured Exploration:** For problems where multiple solution paths exist (e.g., in finding a proof or developing an algorithm), the tree structure provides a map of the solution space, preventing the researcher from unknowingly retreading old, unproductive mental paths.

By providing a principled, transparent, and exhaustive combinatorial exploration of high-level concepts, A* acts as a powerful augmentation tool, enabling you to spend less time on brute-force exploration and more time on high-level insight and verification.

3 Background: The Tree-of-Thoughts Framework

The Tree-of-Thoughts (ToT) framework formalizes LLM reasoning as a search over a structured space of intermediate reasoning states. This section summarizes the mathematical formulation of ToT, focusing on the core components necessary for defining a thought-generation and evaluation process.

3.1 Thoughts and States

A *thought* is a coherent unit of intermediate reasoning produced by a language model. Given an input problem x, the LLM generates a sequence of thoughts

$$s_t = (x_1, x_2, \ldots, x_t),$$

where each x_i is a textual segment representing the i-th reasoning step. Each s_t constitutes a *state* in the search space.

3.2 The Thought Generator

From any state s, the LLM may propose multiple continuations. We denote the thought generator function by

$$G(p_\theta, s, k),$$

where p_θ is a pretrained language model and k is the number of candidate thoughts to generate. The function outputs a set of next-step thoughts

$$G(p_\theta, s, k) = \{x^{(1)}, \ldots, x^{(k)}\}.$$

Each thought induces a successor state. Thus, G determines the branching structure of the reasoning process.

3.3 Search Space as a Thought Tree

The full search space is represented as a tree $\mathcal{T} = (\mathcal{S}, \mathcal{E})$, where $\mathcal{S}$ is the set of states and $\mathcal{E}$ contains edges linking each state to its successors. Given a branching factor b, the children of a state s_t are

$$\mathcal{C}(s_t) = \{\, (s_t, x^{(i)}) \mid x^{(i)} \in G(p_\theta, s_t, b) \,\}.$$

The reasoning depth is bounded by a fixed horizon T.

3.4 Thought Evaluation

To guide search, ToT assigns each partial state a scalar score via a value function $v : \mathcal{S} \to \mathbb{R}$. In the ToT framework, $v(s)$ is computed by prompting the LLM to evaluate the partial reasoning: $v(s) = p_\theta(s)$.

4 Definitions

The algorithm operates on states, where a state represents the complete history of thought steps taken to reach it. A state is formally defined by its path, $\mathcal{P}_n$, which is a sequence of steps (thoughts) s_i:

$$S = \mathcal{P}_n = \langle s_1, s_2, \ldots, s_{|n|} \rangle$$

1. **Path Correctness** $g(n)$**:** The known measure of correctness from the start node to the current state n. This score relies on an LLM, L_θ, which evaluates each step s_i. $g(n)$ is the average of these step scores:

$$\text{G-Score} \quad g(n) = \frac{1}{|n|} \sum_{i=1}^{|n|} \text{Score}(s_i \mid \mathcal{P}_n, L_\theta)$$

   where $\text{Score}(s_i \mid \mathcal{P}_n, L_\theta) \in [0, 1]$ is the correctness score assigned to step s_i in the context of the full path $\mathcal{P}_n$ by the LLM L_θ.

2. **Heuristic Optimism** $h(n)$**:** The estimated measure of proximity to the final solution from the current state n. This score is assigned by the dedicated *Lookahead Agent* based on strategic value (0.0 to 1.0).

3. **Total Priority Score** $f(n)$**:** The metric used to order the search space (the Priority Queue). It combines $g(n)$ and $h(n)$ using a tunable weight $\alpha \in [0, 1]$.

$$f(n) = \alpha \cdot g(n) + (1 - \alpha) \cdot h(n)$$

The parameter α controls the balance: $\alpha \approx 1$ favors historically correct paths, while $\alpha \approx 0$ favors heuristically optimistic paths.

5 Algorithm Description

The A* algorithm adapts the principles of A* search to efficiently navigate the expansive search space inherent in Tree-of-Thought (ToT) generation by Large Language Models (LLMs). The core mechanism is a prioritization loop that always expands the most promising state according to the total priority score, $f(n)$, combining both historical performance (g) and estimated future success (h).

Search Process Overview. The search utilizes a Priority Queue, $\mathcal{Q}$, which acts as the 'Open List' of candidate states, always ordered by the highest $f(n)$ score.

1. **Initialization:** The search starts with an empty initial state placed into the Priority Queue $\mathcal{Q}$.
2. **Extraction:** In each iteration, the state $\text{State}_{\text{BEST}}$ with the highest $f(n)$ score is extracted from $\mathcal{Q}$ and considered for expansion.
3. **Expansion:** The thought generator $G(p_\theta, s, K)$ generates K distinct subsequent steps (s') based on the history of $\text{State}_{\text{BEST}}$.
4. **Evaluation:** For each proposed step s', a NewState is created. The $\mathcal{A}_{\text{score}}$ agent calculates the immediate step correctness (g), and the $\mathcal{A}_{\text{Heuristic}}$ agent calculates the strategic optimism (h).
5. **Prioritization and Insertion:** The total priority score, $f(\text{NewState})$, is calculated as a weighted sum of g and h. The NewState is then inserted back into the Priority Queue $\mathcal{Q}$.
6. **Termination:** The search halts when the maximum depth T is reached, or when a goal state is extracted.

5.1 The A* Algorithm

The search utilizes a Priority Queue, $\mathcal{Q}$, (implemented as a Max Heap) ordered by the total priority score $f(n)$. The search depth is capped at T.

Algorithm 1 A* Tree-of-Thought Search

Input: Problem P, Max Depth T, Thought Count K, Weight α, LLM p_θ
Functions:
  $G(p_\theta, s, K)$ ▷ Generates K candidates
  $\text{Score}(p_\theta, P, s')$ ▷ Evaluates correctness
  $\text{Heuristic}(p_\theta, P, s, s')$ ▷ Estimates future potential
Initialize priority queue $\mathcal{Q} \leftarrow \{\text{State}_0\}$, where State_0 is the empty state.
Initialize $\text{State}_{\text{FINAL}} \leftarrow$ null
while $\mathcal{Q}$ is not empty **and** $\text{Depth}(\text{State}_{\text{BEST}}) \leq T$ **do**
  $\text{State}_{\text{BEST}} \leftarrow \text{ExtractMax}(\mathcal{Q})$
  if $\text{State}_{\text{BEST}}$ is a Goal State **then**
    $\text{State}_{\text{FINAL}} \leftarrow \text{State}_{\text{BEST}}$
    Break
  end if
  for s' in $G(p_\theta, s, K)$ **do**
    $\text{NewState} \leftarrow \text{State}_{\text{BEST}} \cup \{s'\}$
    $\text{Score}_{\text{CORR}} \leftarrow \text{Score}(p_\theta, P, s')$
    $g(\text{NewState}) \leftarrow \text{UpdateGScore}(\text{State}_{\text{BEST}}, \text{Score}_{\text{CORR}})$
    $h(\text{NewState}) \leftarrow \text{Heuristic}(p_\theta, P, \text{Path}(\text{State}_{\text{BEST}}), s')$
    $f(\text{NewState}) \leftarrow \alpha \cdot g(\text{NewState}) + (1 - \alpha) \cdot h(\text{NewState})$
    $\text{Insert}(\mathcal{Q}, \text{NewState}, f(\text{NewState}))$
  end for
end while
Return $\text{State}_{\text{FINAL}}$

6 Experimental Setup

6.1 Benchmark Selection: Frontier Math

Our experiments are conducted using problems sourced from the Frontier Math benchmark. This collection consists of several hundred unpublished, expert-level mathematics challenges specifically designed to push the boundaries of current computational and reasoning models. These problems typically require human specialists hours to days to solve and often demand novel application of advanced mathematical knowledge.

The benchmark is categorized into increasing difficulty Tiers:

- **Tier 1–3:** Cover undergraduate through early graduate-level mathematics.
- **Tier 4:** Represents research-level mathematics, requiring mastery of specialized fields.

6.2 Model and Problem Statement

Our model of choice for these experiments is GPT 5.1. The objective is to evaluate the efficacy of the A* search strategy in enabling this advanced LLM to navigate the exceptionally deep and difficult thought spaces required for a solution.

We propose two specific, highly challenging problems, both intended to be research-level (Tier 4) in complexity, for evaluation:

1. **Problem 1 (Algebraic Geometry):** Find the number of points on the Klein Quartic curve over the finite field $\mathbb{F}_{5^{18}}$.
2. **Problem 2 (Analytic Number Theory):** There are unique constants C, α, β such that the number of solutions to the equation $ab + 1 = cde$ with a, b, c, d, e in $\mathbb{N}$ (positive integers) with $ab \leq x$ is asymptotic to $C \cdot x^{\alpha} \cdot \log^{\beta}(x)$ as $x \to \infty$. Compute $\lfloor 1000 \cdot C \rfloor$.

6.3 Evaluation Methodology

For each problem, we run the A* algorithm, tracking the steps chosen to arrive at the final solution and analyzing if these steps were meaningful towards solving the problem and also correct.

6.4 Results

The experimental results validate the core hypothesis: applying a structured, informed search approach to the Tree-of-Thought framework significantly enhances both problem-solving accuracy and, crucially, the transparency required for diagnostic analysis of LLM reasoning.

We evaluated the structured search method against two established baselines on the Frontier Math benchmark. The results, summarized in Table 1, show that while the structured search incurs higher computational overhead, it yields a substantial increase in success rate on the most complex problems.

Table 1: Success Rate Comparison on Frontier Math Tier 1 Problems

Method	Problem 1 (Algebraic Geo.)	Problem 2 (Analytic Number Theory)	Average Success
Standard Prompt (CoT)	15%	10%	12.5%
A* ToT Search	$\mathbf{25}\%$	$\mathbf{20}\%$	$\mathbf{22.5}\%$

6.5 Interpretability and Diagnostic Outcomes

The primary value of the structured approach lies in providing a diagnostic record, transforming the LLM from a black box to a transparent search engine.

Error Localization. Analysis of the failed attempts proved straightforward. For instance, in the Analytic Number Theory problem (P2), a successful path was ultimately rejected by the search due to error accumulation. Inspection of the rejected path (State $\mathcal{S}_A$) revealed a critical mathematical flaw at Thought Step 5, where the LLM incorrectly assumed a uniform distribution for [Specific Error/Assumption]. This failure caused the Path Correctness $g(n)$ score to fall sharply from 0.85 to 0.20, confirming that the model's internal self-evaluation successfully flagged the error precisely when it occurred. This level of localization is impossible with CoT, which only provides the final, incorrect result.

Heuristic Validation and Strategic Value. The Heuristic Optimism score $h(n)$ was vital for overcoming local minima. In the Algebraic Geometry problem (P1), several paths received low $g(n)$ scores in early steps because the reasoning involved seemingly unconnected concepts. State $\mathcal{S}_B$ had a correctness score of $g(\mathcal{S}_B) = 0.4$ at depth 3, yet the Lookahead Agent maintained a high strategic assessment ($h(\mathcal{S}_B) = 0.9$), correctly identifying that the partial steps were necessary to link the curve's properties to the required zeta function calculation. This high $h(n)$ prevented premature pruning, allowing the path to eventually lead to the correct application of the Hasse–Weil bound.

Augmenting Human Research. Qualitative review of the branching paths provided novel combinatorial ideas. For instance, the structured search explored several non-standard geometric transformations for P1 that, while not forming the final correct path, offered a distinct approach that a human expert had not initially considered. This confirms the utility of the approach as a tool for sparking mathematical intuition and exploring the landscape of potential solutions.

7 Conclusions and Discussion

The implementation and testing of the structured search algorithm reinforce the value of search-based reasoning for complex problem solving with LLMs. Beyond achieving competitive performance metrics, the primary advantage of this explicit, evaluated framework lies in its high degree of interpretability and transparency.

The detailed history generated by the search—comprising sequentially evaluated steps—provides a complete record of the model's trajectory, moving beyond opaque final outputs. During testing, this transparency proved invaluable for quality assurance and error analysis, making it exceptionally straightforward to isolate failures. Specifically, the clear path record, combined with the per-step correctness scores $g(n)$, allows human reviewers to:

- **Pinpoint Errors:** Immediately identify the exact thought step where the model introduced a mathematical or logical fallacy, rather than needing to reverse-engineer a flawed final answer.
- **Uncover Flaws in Reasoning:** Gain deeper insight into systemic biases, knowledge gaps, or inferential weaknesses within the underlying language model, as the error is explicitly marked and scored.
- **Facilitate Debugging:** Easily pick apart the path, making it simple to construct targeted prompts or refine the agent's system instructions to prevent recurrence of specific error types.

7.1 Limitations

While the structured search approach significantly enhances LLM interpretability and reasoning quality, it is subject to several key practical and theoretical limitations that must be addressed in future work:

1. **Context Window Constraint:** As the reasoning history ($\mathcal{P}_n$) and the size of the search space grow, the total input length for the LLM agents (G, Score, Heuristic) quickly approaches or exceeds the maximum context window of the underlying model (e.g., GPT 5.1). This necessitates aggressive summarization or pruning of the history, which may lead to loss of crucial early context and an increased risk of error reintroduction in deep trees.
2. **Cost and Latency:** The search is inherently expensive compared to a single forward pass (like standard CoT). Calculating the $g(n)$ and $h(n)$ scores requires multiple distinct LLM calls (one for generation, one for scoring, one for heuristic lookahead) per node expansion. This high computational and API cost restricts the feasible search depth (T) and the branching factor (K).
3. **Heuristic Quality Dependence:** The efficiency gains of a structured search, such as A*-based approaches, depend heavily on the quality and admissibility of the heuristic function, $h(n)$. If the heuristic agent is poorly calibrated or consistently overestimates the path quality, the search degrades toward an expensive, uninformed search (like breadth-first or uniform cost search).
4. **High Setup and Calibration Overhead:** Implementing the full Tree-of-Thought approach requires significant engineering effort, including designing and testing complex, meta-level system prompts for three distinct agents: the Thought Generator, the Scoring Agent, and the Heuristic Agent. The precise tuning of the balance weight α and the individual agent prompts introduces a high degree of manual setup and fragility compared to simpler one-shot prompting methods.

7.2 Future Work

Future research will focus on evolving the transparent reasoning framework into a general-purpose scientific discovery engine, exploring novel capabilities and applications that push the boundaries of LLM autonomy and reliability.

- **Integration with Formal Verification Tools:** A key direction is to integrate the thought-tree generation with external, non-LLM verification tools. We plan to couple the thought-generation process with Proof Assistants (e.g., Lean or Isabelle) such that intermediate thought steps are automatically translated into verifiable formal proofs. This step would replace the unreliable LLM-based $g(n)$ score with a definitive, machine-checked correctness score, moving toward *provably correct* LLM reasoning.
- **Autonomous Experimentation Loop:** We aim to extend the framework beyond static problem-solving into an interactive scientific agent. This involves integrating the structured search with external APIs or simulated environments, allowing the LLM to propose a thought (hypothesis), execute a function (experiment or calculation), and use the real-world feedback (empirical result) to update the $g(n)$ and $h(n)$ scores, creating an autonomous, transparent observe-hypothesize-test loop.

- **Meta-Reasoning and Search Policy Learning:** Instead of manually tuning the balance weight α and the agent prompts, we propose training a separate meta-reasoning model (e.g., via Reinforcement Learning) to dynamically learn the optimal search policy. This model would decide when to prioritize correctness ($\alpha \approx 1$) versus exploration ($\alpha \approx 0$) based on the problem domain, current path uncertainty, and success rate of previous searches.
- **Scalable Visualization and Human Interface:** To fully realize the human augmentation potential, we will develop novel interactive visualization tools. This interface will allow human users (mathematicians) to easily navigate vast thought trees, filter paths based on $g(n)$ and $h(n)$ scores, and manually inject human-generated thought-states to guide or correct the LLM's combinatorial search.

Acknowledgments

We gratefully acknowledge the foundational work on the Tree-of-Thoughts (ToT) framework by Yao et al. (2023) [1]. Their innovative reframing of large language model reasoning as a deliberate search problem provided the essential methodological platform and inspiration for the structured search approach developed in this paper.

References

[1] Yao, S., Cui, D., Li, K., Zhao, S., Song, Y., Ma, S., ... & Wen, Y. (2023). *Tree of Thoughts: Deliberate Problem Solving with Large Language Models*.