Compiling Consciousness: The RR-Tree and the Syntax of Machine Reasoning

YC Huang
Independent Researcher
joffrey.oh@gmail.com

July 17, 2025

Abstract

The pursuit of autonomous agents has been constrained by the limitations of linear (Chain-of-Thought) and unstructured exploratory (Tree-of-Thought) reasoning frameworks. This paper argues for a new paradigm: treating reasoning as a formal, self-modifying program. We introduce a cognitive architecture built upon three core ideas.

First, we define the **Recursive Reasoning Tree (RR-Tree)**, a stateful S-expression tree that represents thought as a computable data structure, moving beyond unstructured text.

Second, to navigate this tree, we propose the **Thought-Quality-Resonance (TQR) model**, a multi-dimensional evaluation function that assesses the logical alignment, novelty, and complexity of each reasoning step, replacing simple heuristics with rigorous, quantitative guidance.

Third, we define a set of **explicit transformation operators** (`CHOOSE`, `EXPAND`, `REWRITE`, `DEEP_DIVE`) that serve as the primitives for meta-reasoning, allowing an agent to systematically and audibly manipulate its own cognitive process. Together, these components create a framework that enhances robustness, provides radical explainability, and enables the recursive decomposition of complex problems, laying the foundation for a new class of auditable and dynamic AI agents.

1 Introduction

The remarkable progress in Large Language Models (LLMs) has brought the prospect of Artificial General Intelligence (AGI) into sharper focus. A critical component of such intelligence is the ability to reason—to systematically process information, explore hypotheses, and construct coherent plans to solve complex, multi-step problems. While early models relied on implicit, black-box reasoning, the development of Chain-of-Thought (CoT) prompting [9] demonstrated that making the reasoning process explicit and sequential significantly improves performance on a wide range of tasks.

However, this very linearity is also a fundamental weakness. A single logical error or suboptimal choice early in a CoT can derail the entire process, leading to a cascade of failures from which the model cannot easily recover. The reasoning process is brittle, lacking the mechanisms for self-correction, parallel exploration, or nuanced evaluation of different possibilities. The real world is rarely linear; it is a complex, branching tree of possibilities.

To address this, researchers have proposed more exploratory frameworks like Tree-of-Thought (ToT) [10], which allows a model to consider multiple reasoning paths simultaneously. While a significant conceptual advance, current ToT implementations often rely on informal generation and simple evaluation heuristics (e.g., voting), failing to provide a sufficiently rigorous and auditable framework for navigating complex problem spaces. The process remains opaque, and the agent's ability to introspect and deliberately modify its own reasoning is limited.

This paper argues for a paradigm shift: from treating reasoning as an unstructured stream of text to treating it as a **formal, computable program**. We introduce the **Recursive Reasoning Tree (RR-Tree)**, a cognitive architecture designed to instantiate this paradigm. The RR-Tree operationalizes an agent's thought process as a dynamic S-expression tree, where every cognitive step is a structured, stateful, and inspectable node.

The evolution of this reasoning tree is not random; it is guided by the **Thought-Quality-Resonance (TQR) model**, a quantitative evaluation function we propose. The TQR model moves beyond simple heuristics to provide a nuanced, multi-dimensional assessment of each potential thought, balancing its logical coherence, its novelty and insight, and its cognitive simplicity. This allows the agent to make transparent, justifiable decisions about where to focus its mental energy.

Furthermore, when faced with problems that are not merely complex but are rife with uncertainty and potential deception (e.g., deducing interstellar strategy in a game-theoretic "dark forest" scenario), existing methods fall short. Such challenges demand more than heuristic search; they require a framework capable of rigorous, auditable deduction from first principles. The RR-Tree is engineered to meet this demand.

Our contributions are threefold:

1. **A Formal Cognitive Architecture (RR-Tree):** We propose a stateful S-expression tree structure for representing thought, complete with metadata for auditable tracking of the reasoning process.

2. **A Sophisticated Evaluation Model (TQR):** We introduce a multi-dimensional, non-linear scoring function that enables agents to perform nuanced evaluations of their own reasoning paths.

3. **An Explicit Meta-Reasoning Toolkit:** We define a set of discrete, symbolic operators (`CHOOSE`, `EXPAND`, `REWRITE`, `MERGE`, `DEEP_DIVE`) that

allow an agent to manipulate its reasoning tree, enabling true meta-reasoning.

Together, these components create a system that is not only more powerful but also radically transparent. The RR-Tree transforms reasoning from a black box into an inspectable, auditable, and self-optimizing program. This paper details the structure and dynamics of this system, compares it to existing methods, and explores its implications for building the next generation of autonomous agents.

2 Related Work

The RR-Tree builds upon several key threads of research in AI and cognitive science, aiming to synthesize their strengths into a cohesive framework.

2.1 From Chains to Trees

The most direct predecessors to our work are Chain-of-Thought (CoT) and Tree-of-Thought (ToT).

Chain-of-Thought (CoT) Prompting [9] was a pivotal discovery, showing that prompting LLMs to produce a series of intermediate reasoning steps before giving a final answer improves performance on arithmetic, commonsense, and symbolic reasoning tasks. However, CoT is inherently sequential and greedy. Once a step is generated, it is fixed, making the process vulnerable to error propagation.

Self-Consistency [8] was an early attempt to mitigate this by sampling multiple CoT paths and taking a majority vote on the final answer. While effective, this is a brute-force approach that is computationally expensive and does not involve any explicit evaluation of the reasoning paths themselves.

Tree-of-Thought (ToT) [5, 10] generalized CoT by allowing the exploration of a tree of reasoning paths. This allows for backtracking and the consideration of multiple alternatives. ToT represents a crucial conceptual leap. However, its implementations often lack a formal structure for the thoughts themselves and rely on simple heuristics (e.g., LLM-based "votes" or simple scores) to prune the tree. The RR-Tree can be seen as a formalization and extension of the ToT concept, replacing informal text generation with a structured S-expression program and replacing simple heuristics with the more sophisticated TQR evaluation model and explicit transformation operators.

2.2 Reasoning as a Program

The idea of treating reasoning programmatically has appeared in other forms.

Program-Aided Language Models (PALs) [2] and **Tool-Augmented Language Models (TALMs)** [7] augment LLMs by allowing them to generate and execute code (e.g., Python) to solve problems. This offloads complex calculations or structured data manipulation to a reliable external environment. The RR-Tree shares the "reasoning as a program" ethos but applies it to the *internal* cognitive process itself, rather than just external tool use. Our transformation operators are the "keywords" of an internal programming language for thought.

2.3 Cognitive Architectures

Finally, our work is inspired by classic cognitive architectures like **SOAR** [4] and **ACT-R** [1], and draws its core data structure concept from the foundational work on LISP and symbolic expressions by McCarthy [6], as well as its modern interpretation by Graham [3]. These systems, originating from cognitive science and computer science, have long sought to create unified models of cognition. They operate on structured representations and follow a "recognize-act" cycle that is analogous to our "evaluate-transform" loop. The RR-Tree can be viewed as a modern, LLM-native cognitive architecture that replaces hand-crafted production rules with a flexible, learnable evaluation function (the TQR model) and a more general set of meta-reasoning primitives, making it more adaptable to the vast and varied domains that LLMs can handle.

3 The Recursive Reasoning Tree (RR-Tree)

At the heart of our proposed cognitive architecture lies the Recursive Reasoning Tree (RR-Tree). It is a formal structure designed to move beyond the limitations of linear or unstructured thought processes. The RR-Tree operationalizes an agent's reasoning, transforming it from an ephemeral internal state into a tangible, inspectable, and manipulable data structure. This section details the core principles, structure, and operational dynamics of the RR-Tree.

3.1 Core Idea: Reasoning as a Program

The fundamental premise of the RR-Tree is to treat the entire process of reasoning as a program. Specifically, we represent the agent's thought process as a symbolic expression (S-expression) tree, an idea whose intellectual roots trace back to the original LISP paper by McCarthy [6] and its influential explanation by Graham [3]. This choice is deliberate. S-expressions, foundational to Lisp and Scheme, provide a simple, uniform syntax for data and code, making them exceptionally well-suited for metaprogramming—in this case, an agent's meta-reasoning about its own thought process.

By structuring thought as a tree, we explicitly enable the exploration of multiple, branching lines of reasoning, a critical feature for complex problem-solving that is absent in linear CoT. Each node in the tree represents a discrete cognitive step—a hypothesis, a piece of evidence, a sub-goal, or a final conclusion. The hierarchical nature of the tree allows for the recursive decomposition of complex problems into manageable sub-problems, a process we will later show is managed by the `DEEP_DIVE` operator.

3.2 Structure: The S-Expression Tree

An RR-Tree is an S-expression tree where each node is a list containing a symbolic name and its children. Crucially, every node must contain a `(meta ...)` block as its first child, which stores essential metadata for tracking and manipulating the node's state.

A typical node structure is as follows:

```
(node_name
  (meta (id "...") (version "...") (status "...") (tqr\_score "..."))
  (child_1 ...)
  (child_2 ...)
  ...
)
```

- `id`: A unique identifier for the node, allowing for precise targeting by transformation operations.
- `version`: Tracks the number of times a node has been refined via the `REWRITE` operator.
- `status`: A state machine for the node's lifecycle, typically including states like `hypothesized`, `evaluating`, `resolved`, `deprecated`, or `merged`. This allows the agent to manage the focus of its reasoning process.
- `tqr_score`: Stores the result of the TQR evaluation, providing a quantitative basis for decision-making.

This structure ensures that the entire reasoning process is not only hierarchical but also fully stateful and auditable.

3.3 Dynamics: The Tree Transformation Instruction Set

The RR-Tree is not a static data structure; it is a dynamic program that evolves. The agent interacts with and refines its own reasoning tree through a well-defined set of transformation instructions. These instructions are the primitives of the agent's meta-reasoning capability.

- `CHOOSE(parent_node_id, chosen_node_id)`: This is the primary decision-making operator. When a node contains several `alternatives`, the agent applies the TQR model to each, then uses `CHOOSE` to select the most promising path, thereby resolving uncertainty. For example, after evaluating three potential moves in a chess game, `CHOOSE` selects the one with the highest TQR score.

- `EXPAND(parent_node_id, new_child_node)`: This operator is used for elaborative reasoning. Once a hypothesis is chosen, `EXPAND` is used to add supporting arguments, break down a plan into steps, or provide further details. For instance, after choosing a diagnosis, `EXPAND` would add nodes for `(treatment_plan)`, `(prognosis)`, and `(required_tests)`.

- `REWRITE(node_id, new_content)`: This is the refinement operator. It allows the agent to improve upon its own thoughts, clarifying a statement, correcting a logical flaw, or incorporating new information. For example, an initial thought `(conclusion "earth is big")` could be rewritten to `(conclusion "earth has a mass of 5.972e24 kg")`.

- `MERGE(node_ids_array, new_parent_node)`: This operator facilitates synthesis. It enables the agent to identify multiple, related ideas and abstract them into a single, higher-level concept. For example, the nodes `(finding "planet has water")` and `(finding "planet has oxygen")` could be merged into a new parent node `(synthesis "planet may be habitable")`.

- `DEEP_DIVE(node_id)`: This is a recursive operator that embodies the fractal nature of complex problem-solving. When a sub-problem is sufficiently complex (e.g., "design a novel fusion reactor"), the agent can invoke `DEEP_DIVE` on that node, treating it as a new, self-contained `root` for a sub-tree. A new "explore-evaluate-converge" loop is initiated for this sub-problem, and only upon its convergence does the main reasoning loop resume.

Together, these components transform reasoning from a black box into a transparent, iterative, and self-optimizing process. The RR-Tree provides the structure, while the TQR-driven transformation instructions provide the dynamics for intelligent and auditable exploration of complex problem spaces.

4 The TQR Evaluation Model

The ability to generate a tree of potential reasoning paths is only useful if the agent possesses a mechanism to navigate it effectively. Simply exploring all

branches exhaustively is computationally intractable and strategically naive. A robust autonomous agent requires a sophisticated evaluation function to guide its exploration, prune unpromising avenues, and focus its cognitive resources.

Consider an AI agent tasked with medical diagnosis. Faced with a set of symptoms, it might generate several hypotheses. Is a persistent cough a sign of a common cold, a rare fungal infection, or an environmental allergy? To decide which path to explore first, the agent must ask itself:

1. How well does this hypothesis align with the known facts? (Is it a logical continuation of my reasoning?)
2. Does this hypothesis represent a potentially crucial, non-obvious insight? (Is it a rare but critical diagnosis I shouldn't miss?)
3. Is this hypothesis the simplest explanation that fits the facts? (Am I overcomplicating things?)

To formalize this internal deliberation, we introduce the **Thought-Quality-Resonance (TQR) model**, a quantitative framework designed to serve as the agent's internal compass for navigating its own RR-Tree.

4.1 Motivation: Guiding the Evolution of Thought

The primary motivation behind the TQR model is to move beyond simple heuristics (e.g., "vote" or "majority rule" in some ToT implementations) and provide a more nuanced, multi-dimensional, and domain-adaptable evaluation metric. The model is designed to answer a critical question at each step of the reasoning process: "Of all the things I *could* think about next, what *should* I think about?"

The TQR model achieves this by decomposing the "goodness" of a thought-node into three distinct, yet complementary, dimensions. This multi-faceted approach allows the agent to balance logical consistency, creative insight, and cognitive efficiency, leading to a more holistic and robust decision-making process.

4.2 The TQR Metrics

The TQR score of any given node in the RR-Tree is a composite of three core metrics:

- α**: Thought Alignment (T)**: This metric assesses the logical coherence and relevance of a node in relation to its parent and the overall goal. It answers the question: "Is this thought a direct and logical continuation of my current line of reasoning?" A high Alpha score

indicates that a thought is on-topic and directly contributes to solving the immediate sub-problem. It penalizes non-sequiturs and logical leaps, ensuring the reasoning process remains grounded and coherent.

- β**: Quality & Insight (Q)**: This is the measure of a thought's intrinsic value and novelty. It answers the question: "Is this thought interesting, insightful, or potentially groundbreaking?" A high Beta score is assigned to nodes that introduce novel perspectives, synthesize disparate information into a new whole, or represent a significant step towards a creative or non-obvious solution. This metric encourages the agent to move beyond trivial or well-trodden paths and to favor genuine insight.

- **C: Cognitive Complexity (R for Resonance/Simplicity)**: This metric evaluates the simplicity and elegance of a thought-node. It answers the question: "Is this the most efficient and parsimonious way to express this idea or solve this sub-problem?" A low Complexity score is awarded to nodes that are clear, concise, and avoid unnecessary steps or entities. This metric acts as a form of Occam's Razor, pushing the agent towards solutions that are not only correct but also elegant. The "Resonance" aspect implies that simpler, more fundamental ideas often have a wider-reaching impact and "resonate" better within the overall logical structure.

4.3 Scoring Function and Decision Mechanism

The individual metrics are combined into a single TQR score using a weighted, non-linear function designed to balance their competing influences. The formula is:

$$unreleased \tag{1}$$

This specific formulation has several desirable properties:

- The **OP1** relationship between Alpha and Beta ensures that a thought must be both relevant *and* insightful to score highly.

- The **OP2** gives a disproportionate reward to truly insightful ideas ($\beta > 1$), reflecting the high value of creative leaps in problem-solving.

- The **OP3** in the denominator provides a significant penalty for overly complex thoughts, enforcing cognitive efficiency.

The agent uses this score to drive its decision-making. When faced with a set of `alternatives`, it computes the TQR score for each, and the `CHOOSE` operator selects the branch with the highest score. This TQR-driven evolution allows the RR-Tree to dynamically and autonomously prune its own

search space, focusing its efforts on the most promising paths and converging efficiently on a high-quality solution.

5 Comparative Analysis

The proposal of the RR-Tree and its accompanying TQR model is situated within a rapidly evolving landscape of research aimed at improving the reasoning capabilities of large language models. To clarify its unique contribution, this section provides a comparative analysis against two prominent paradigms: Chain-of-Thought (CoT) and Tree-of-Thought (ToT).

5.1 Beyond Linearity: vs. Chain-of-Thought (CoT)

Chain-of-Thought prompting has been a significant step forward, demonstrating that prompting models to "think step-by-step" elicits more robust reasoning. However, its fundamental limitation is its **linearity**. A CoT is a single, sequential path from problem to solution. This structure suffers from several key drawbacks:

1. **Error Propagation**: An error or suboptimal choice made early in the chain is difficult to correct. The model tends to commit to its initial path, and subsequent reasoning is built upon a potentially flawed foundation.

2. **Lack of Exploration**: CoT does not inherently support the exploration of multiple hypotheses or solution paths in parallel. It follows a single trajectory, which may not be the optimal one.

3. **Implicit Decision-Making**: The choices made at each step of a CoT are implicit and opaque. It is difficult to determine *why* the model chose one logical step over another.

The RR-Tree directly addresses these limitations. Its tree structure is inherently non-linear, allowing for the explicit representation of `alternatives` at any reasoning step. The `CHOOSE` operator, guided by the TQR model, makes the decision-making process transparent and quantitative. By systematically exploring and evaluating different branches, the RR-Tree can prune suboptimal paths and is far more resilient to early-stage errors.

5.2 Formalizing Exploration: vs. Tree-of-Thought (ToT)

The Tree-of-Thought paradigm is a conceptual ancestor to the RR-Tree, recognizing the need for non-linear exploration. ToT proposes generating multiple thought-paths and using heuristics (like voting or simple scoring) to select the most promising one. While ToT established the value of tree-based

exploration, the RR-Tree advances this concept by introducing a higher degree of formalism, structure, and meta-reasoning.

The key distinctions are:

1. **Formal Structure vs. Informal Generation**: ToT is often implemented by generating multiple, unstructured text completions that form a conceptual tree. The RR-Tree, by contrast, is a **formal data structure**—an S-expression tree with mandatory metadata. This structure is not just a record of thought; it is a computable program that the agent can inspect and manipulate.

2. **Sophisticated Evaluation vs. Simple Heuristics**: ToT relies on relatively simple heuristics for evaluation. The RR-Tree introduces the **TQR model**, a multi-dimensional, quantitative evaluation function that balances logical coherence (α), novelty (β), and simplicity (C). This provides a much richer and more nuanced basis for decision-making.

3. **Explicit Transformation Operators**: The RR-Tree is defined by a set of explicit, discrete transformation instructions (`EXPAND`, `REWRITE`, `MERGE`, `DEEP_DIVE`). These operators represent the fundamental primitives of meta-reasoning. They allow the agent to perform targeted, auditable modifications to its own thought process, moving beyond simple generation and selection to include refinement, synthesis, and recursive problem decomposition.

In essence, while ToT introduced the "what" (explore a tree), the RR-Tree specifies the "how": a formal system for representing, evaluating, and transforming the reasoning process itself.

5.3 Reasoning as a Program

The most fundamental distinction is that the RR-Tree treats **reasoning as a program**. A CoT is a static text string. A ToT is a collection of such strings. An RR-Tree is a dynamic, executable program that writes, evaluates, and rewrites itself to arrive at a solution. This programmatic approach provides a clear path towards more autonomous, auditable, and robust reasoning systems.

To summarize the comparison, we offer the following table:

6 Implications and Future Work

The conceptual framework of the RR-Tree and the TQR model opens several promising avenues for both practical application and future research. By structuring reasoning as a formal, self-modifying program, this architecture has significant implications for the future of autonomous agents.

enewcommand11.5

Feature	Chain-of-Thought (CoT)	Tree-of-Thought (ToT)	Recursive Reasoning Tree (RR-Tree)
Structure	Linear Sequence	Tree	Formal S-Expression Tree w/ Metadata
Exploration	Single Path	Multi-Path	Multi-Path with Recursive Decomposition
Evaluation Mechanism	Implicit (Greedy)	Heuristic (e.g., Vote)	Quantitative (TQR Model)
Meta-Reasoning	None	Rudimentary (Pruning)	Explicit Operators (Rewrite, Merge, etc.)
Auditability	Low	Medium	High (Full, stateful log)

Table 1: A comparison of reasoning frameworks.

6.1 Implications

- **Enhanced Robustness**: The ability to explore, evaluate, and prune multiple reasoning paths makes agents less susceptible to the kind of single-path, cascading failures common in linear models. This leads to more reliable and robust problem-solving.
- **Radical Explainability**: The RR-Tree is inherently auditable. The final reasoning path is accompanied by a complete, explicit history of the alternatives considered and the quantitative justification (TQR scores) for the decisions made. This transforms the "black box" of AI reasoning into a transparent, inspectable process.
- **Complex Problem Decomposition**: The `DEEP_DIVE` operator provides a formal mechanism for recursive problem decomposition. This allows an agent to tackle highly complex, nested problems by systematically breaking them down into manageable sub-problems, solving them in a focused manner, and then integrating the solutions back into the main reasoning thread.
- **Autonomous Strategy Adaptation**: The TQR model can be adapted to different domains or tasks by adjusting the weights of its components. An agent could learn to prioritize novelty (high β) for creative

tasks or logical coherence (high α) for formal verification, enabling more flexible and context-aware reasoning strategies.

6.2 Future Work

This paper introduces a conceptual framework that invites extensive empirical validation and further development. Key directions for future work include:

- **Empirical Validation**: Implementing the RR-Tree architecture on a variety of benchmark tasks (e.g., advanced mathematics, strategic games, scientific hypothesis generation) to quantitatively assess its performance against state-of-the-art models.
- **Reinforcement Learning from TQR**: The TQR score provides a dense, high-quality reward signal at each step of the reasoning process. This opens a significant research avenue: framing the generation of the next reasoning step as a reinforcement learning (RL) problem. A policy network could be trained, using the TQR score as its reward function, to learn how to autonomously generate high-quality reasoning paths. This would move the TQR model from a purely evaluative role to a generative one, enabling the agent to learn and improve its own reasoning strategies over time.
- **Collaborative Reasoning**: The explicit and structured nature of the RR-Tree makes it an ideal substrate for human-AI collaboration. A human expert could inspect an agent's reasoning tree, prune unpromising branches, or suggest new avenues for exploration, which the agent could then formally integrate into its tree.

7 Conclusion

This paper has argued that the next frontier in autonomous agent intelligence requires a move from linear or unstructured thought processes to a more formal, explicit, and dynamic cognitive architecture. We have introduced the **Recursive Reasoning Tree (RR-Tree)**, a system that treats reasoning as a self-modifying program represented by a stateful S-expression tree. The evolution of this tree is guided by the **Thought-Quality-Resonance (TQR) model**, a quantitative evaluation function that enables the agent to make transparent, multi-dimensional decisions about its own cognitive path.

By defining a set of explicit tree transformation operators—`CHOOSE`, `EXPAND`, `REWRITE`, `MERGE`, and `DEEP_DIVE`—we have provided the building blocks for a system that can explore, evaluate, refine, synthesize, and recursively decompose complex problems. The comparative analysis shows that

the RR-Tree architecture addresses the fundamental limitations of Chain-of-Thought and provides a more formalized and powerful implementation of the concepts underlying Tree-of-Thought.

The ultimate vision of the RR-Tree is to transform the ephemeral act of reasoning into a tangible, auditable, and optimizable engineering discipline. This approach not only promises to create more powerful and robust AI agents but also to make their inner workings radically transparent, paving the way for a future of more explainable and collaborative intelligence.

How to Cite

If you find this work useful in your research, please consider citing it as follows:

```
@misc{huang2025compiling,
    title   = {Compiling Consciousness: The RR-Tree and the Syntax of Machine Reason
    author  = {YC Huang},
    year    = {2025},
    howpublished = {https://github.com/DJoffreyH/RRTree}

    archivePrefix = {arXiv},
    primaryClass = {cs.AI}
}
```

References

[1] John R Anderson. *How can the human mind occur in the physical universe?* Oxford University Press, 2009.

[2] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. Pal: Program-aided language models. In *International Conference on Machine Learning*, pages 10761–10785. PMLR, 2023.

[3] Paul Graham. The roots of lisp. Retrieved from http://www.paulgraham.com/rootsoflisp.html, 2002.

[4] John E Laird. *The Soar cognitive architecture*. MIT press, 2012.

[5] Jieyi Long. Large language model guided tree search. *arXiv preprint arXiv:2305.08291*, 2023.

[6] John McCarthy. Recursive functions of symbolic expressions and their computation by machine, part i. *Communications of the ACM*, 3(4): 184–195, 1960.

[7] Aaron Parisi, Evan Schwartz, and Misha Dror. Talm: Tool-augmented language models. *arXiv preprint arXiv:2205.12255*, 2022.

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

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

[10] Shunyu Yao, Dian Yu, Jeffrey Zhao, Dan Sha, an Butler, and Zhen (Carl) Liu. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.