Summary of Tree of Thoughts: Deliberate Problem Solving with Large Language Models and ReAct: Synergizing Reasoning and Acting in Language Models

Rashi Tyagi (rtyagi4), Ziyang Zheng (ziyangz5), Haoran Wu (haoran29)

February 13, 2025

1 Problem and Motivation

Although large language models have demonstrated impressive language abilities, their problem-solving capabilities remain limited by their sequential, token-by-token generation paradigm. While techniques such as Chain of Thought (CoT) prompting have improved reasoning by introducing intermediate steps, these approaches still struggle with complex planning, decision-making, and interaction with external environments. To address these challenges, Yao et al. proposed two novel frameworks, Tree of Thoughts (ToT) [6] and ReAct [7], which introduce more deliberate reasoning and action-oriented capabilities into LLMs.

To enhance structured reasoning, Yao et al. introduced ToT, which extends CoT by exploring multiple reasoning paths in a tree structure. Unlike CoT's linear progression, ToT enables lookahead, backtracking, and self-evaluation, improving LLMs' ability to solve strategic problems. It significantly outperforms CoT in tasks like mathematical reasoning (4% vs. 74% success in Game of 24), creative writing, and crosswords, highlighting the power of structured search in problem-solving.

However, both CoT and ToT operate in isolation, relying only on internal model knowledge. To address this, Yao et al. proposed ReAct, which interleaves reasoning with real-world actions, allowing LLMs to retrieve information, update beliefs, and dynamically adjust strategies. By mitigating hallucinations and error propagation, ReAct improved both accuracy and interpretability, outperforming reinforcement and imitation learning baselines (e.g., 34% gain on ALFWorld). These findings underscore the importance of integrating reasoning with interactive decision-making.

While ToT and ReAct tackle different challenges, they share a common goal: enhancing structured reasoning and adaptability. ToT strengthens deliberate problem-solving through tree-based exploration, while ReAct enables real-world interaction and decision-making. Together, they pave the way for more robust, generalizable, and interpretable AI systems capable of planning, adapting, and engaging with external environments.

In the rest of this summary, we review previous research, detail the methods used, discuss limitations, and explore future directions for this topic.

2 Related Works

2.1 The related work before these two papers

- **Chain-of-Thought Prompting Elicits Reasoning in Large Language Models** [5]. This paper introduces Chain-of-Thought (CoT) Prompting, a technique that improves reasoning in LLMs by incorporating intermediate reasoning steps into few-shot learning. To explore whether complex reasoning can emerge through in-context learning, the paper evaluates CoT prompting across arithmetic, commonsense, and symbolic reasoning tasks. Experiments demonstrate that CoT prompting significantly outperforms standard prompting but only becomes effective in sufficiently large models (100B+ parameters). Notably, CoT-enabled PaLM 540B achieves state-of-the-art performance on math word problems (GSM8K), surpassing even fine-tuned GPT-3. The results suggest that CoT prompting expands the capabilities of LLMs without requiring additional fine-tuning, improving both performance and interpretability in reasoning-intensive tasks.

- **Self-Consistency Improves Chain of Thought Reasoning in Language Models** [4]. This paper introduces Self-Consistency, a novel decoding strategy that enhances Chain-of-Thought (CoT) prompting by replacing greedy decoding with a sampling-and-aggregation approach. To explore whether reasoning accuracy can be improved by leveraging diverse thought processes, the paper proposes generating multiple reasoning paths and selecting the most consistent final answer. Evaluated on various large-scale models (PaLM-540B, GPT-3-175B, UL2-20B, LaMDA-137B), Self-Consistency significantly boosts performance across arithmetic and commonsense reasoning tasks, achieving state-of-the-art results on benchmarks like GSM8K (+17.9%) and AQuA (+12.2%). The findings suggest that Self-Consistency enhances LLMs' robustness, interpretability, and reasoning capabilities without requiring additional fine-tuning.

- **WebGPT: Browser-assisted question-answering with human feedback** [2]. This paper introduces WebGPT, a fine-tuned GPT-3 model designed for long-form question-answering (LFQA) using a text-based web-browsing environment. To explore whether retrieval-augmented synthesis can improve factual accuracy, the paper integrates real-time search via Bing, human demonstrations, and reinforcement learning with rejection sampling. Evaluated on the ELI5 benchmark, WebGPT's answers are preferred over human-written responses 56% of the time and outperform top Reddit answers 69% of the time. The findings suggest that combining live web search with AI reasoning enhances truthfulness, coherence, and reliability, setting a new standard for evidence-based AI-generated responses.

2.2 The related work after these two papers

- **Graph of Thoughts: Solving Elaborate Problems with Large Language Models** [1]. This paper introduces Graph of Thoughts (GoT), a framework that extends prompting techniques beyond Chain-of-Thought (CoT) and Tree-of-Thoughts (ToT) by modeling reasoning as an arbitrary graph. To explore whether LLM reasoning can be improved through networked thought structures, the paper represents intermediate thoughts as graph nodes and their dependencies as edges. This enables merging multiple reasoning paths, iterative refinement, and feedback loops, making reasoning more dynamic and synergistic. Experimental results

show that GoT significantly outperforms ToT, improving sorting accuracy by 62% while reducing costs by 31%. The findings suggest that GoT generalizes existing prompting methods, enhancing problem-solving efficiency and adaptability without requiring model fine-tuning.

- **Reflexion: Language Agents with Verbal Reinforcement Learning** [3]. This paper introduces Reflexion, a reinforcement learning framework that enables language agents to self-improve using verbal feedback instead of model fine-tuning. To explore whether text-based memory can guide iterative learning, the paper proposes storing self-generated reflections on past mistakes and incorporating them into future decision-making. Reflexion achieves state-of-the-art performance across decision-making (AlfWorld: +22%), reasoning (HotPotQA: +20%), and programming (HumanEval: 91% Pass@1, surpassing GPT-4's 80%). The findings suggest that verbal reinforcement allows LLMs to iteratively refine their reasoning, planning, and coding abilities, making AI agents more adaptive, interpretable, and efficient without additional fine-tuning.

3 Solution Overview

3.1 Tree of Thoughts: Deliberate Problem Solving with Large Language Models

The ToT framework structures problem-solving as a tree of thoughts (as seen in Figure 1), where each thought represents a meaningful step toward a solution. It allows the LM to self-evaluate its reasoning process, guiding decision-making through deliberate reflection and search heuristics. By leveraging BFS and DFS, ToT systematically explores different solution paths, using lookahead and backtracking to refine its approach.

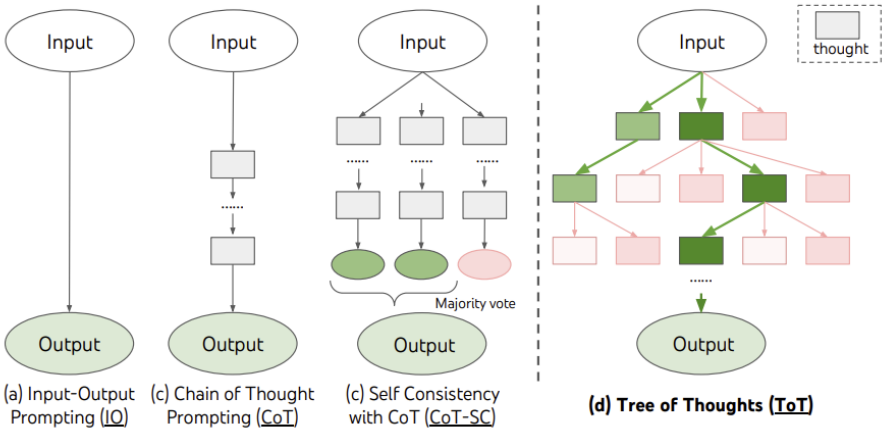


Figure 1: Schematic illustrating various approaches to problem solving with LLMs. Each rectangle box represents a thought, which is a coherent language sequence that serves as an intermediate step toward problem solving.

The Core Components of ToT can be broken down into 4 steps: **(1) Thought decomposition:** where a thought is broken down into manageable steps. A thought ought to be small enough for diversity however big enough for meaningful evaluation. **(2) Thought Generation** $G(p_\theta, s, k)$**:** where multiple thoughts are explored. Two strategies are discussed in the paper- Independent Sampling and Sequential Proposing. **(3) Thought Evaluation** $V(p_\theta, S)$**:** where the LM acts as a heuristic evaluator to rank and filter possible solutions. Two evaluation strategies have been discussed - Independent Valuation and Voting Mechanism. **(4) Search Algorithm:** which determines how the LM navigates the tree of thoughts. Breadth-First Search (BFS) expands multiple paths in parallel, useful for shallow trees. Whereas, Depth-First Search (DFS) approach explores deeply first, backtracking when needed.

- **Experiment:** Evalutation of the ToT framework was performed against three tasks. These tasks (Game of 24, Creative Writing, and Mini Crosswords) require mathematical reasoning, creative writing, and structured problem-solving, demonstrating how ToT improves systematic search and decision-making.

	Game of 24	**Creative Writing**	**5x5 Crosswords**
Input	4 numbers (4 9 10 13)	4 random sentences	10 clues (h1. presented;..)
Output	An equation to reach 24 (13-9)*(10-4)=24	A passage of 4 paragraphs ending in the 4 sentences	5x5 letters: SHOWN; WIRRA; AVAIL; ...
Thoughts	3 intermediate equations (13-9=4 (left 4,4,10); 10-4=6 (left 4,6); 4*6=24)	A short writing plan (1. Introduce a book that connects...)	Words to fill in for clues: (h1. shown; v5. naled; ...)
#ToT steps	3	1	5-10 (variable)

Table 1: Task overview. Input, output, thought examples are in blue.

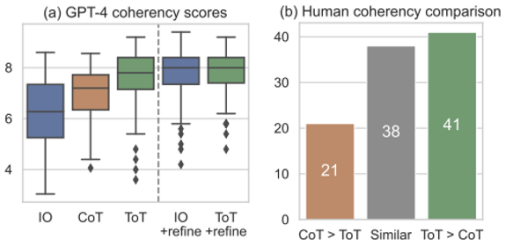
The fundamental setup for the experiments is shown in Table 1. We will further discuss the ToT setup in each challenge. Breadth-first search (BFS) strategy with a heuristic evaluation mechanism rating the steps as "sure/maybe/impossible" was used in the ToT structure for the **Game of 24** challenge. In the **Creative Writing** challenge, ToT structured the process into two steps—first generating multiple plans, then selecting the best one via voting before writing the passage. For **Mini Crossword** challenge, ToT leveraged depth-first search (DFS) with heuristic pruning, generating word candidates while checking if constraints made a solution infeasible.

- **Results:**The Tree of Thoughts (ToT) framework significantly outperformed standard prompting methods across all three tasks.

  In **Game of 24** [Figure 2 (a)], ToT with a breadth of b = 1 achieved 45% success, while increasing breadth to b = 5 raised success to 74%, far surpassing even the best of 100 CoT samples, which only reached 49%. For **Creative Writing** [Figure 2 (b)], ToT-generated passages were rated more coherent than IO (6.19) and CoT (6.93), with an average GPT-4 score of 7.56. Additionally, iterative refinement improved ToT's coherence even further, raising its score from 7.56 to 7.91. In the **Mini Crosswords** [Figure 2 (c)], ToT leveraged depth-first

Method	Success
IO prompt	7.3%
CoT prompt	4.0%
CoT-SC (k=100)	9.0%
ToT (ours) (b=1)	45%
ToT (ours) (b=5)	**74%**
IO + Refine (k=10)	27%
IO (best of 100)	33%
CoT (best of 100)	49%

(a) Game of 24 results



(b) Creative Writing results

Method	Success Rate (%)		
	Letter	Word	Game
IO	38.7	14	0
CoT	40.6	15.6	1
ToT (ours)	**78**	**60**	**20**
+best state	82.4	67.5	35
-prune	65.4	41.5	5
-backtrack	54.6	20	5

(c) Mini Crossword results

Figure 2: Results for Tree of Thoughts (ToT)Experiments.

search (DFS), improving accuracy to 60% and successfully solved 4 out of 20 puzzles. Ablation studies confirmed that pruning heuristics and backtracking were critical.
Across different problem types, ToT significantly outperforms traditional prompting techniques.

3.2 ReAct: Synergizing Reading and Acting in Language Models

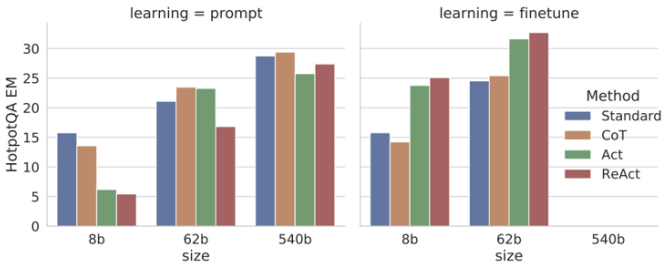
The main idea of ReAct is to combine Reasoning (e.g. Chain-of-thought prompting) and Acting (e.g. action plan generation) in an interleaved manner using LLMs. This paper explores PaLM 540B LLM to generate actions and thoughts. The ReAct framework does three things in every iteration: (1) Thought or reasoning trace (2) Action and, (3) Observation based on the action.

- **Experiment:** The ReAct framework was tested on knowledge-intensive reasoning tasks using two datasets: HotPotQA (multi-hop question answering) and FEVER (fact verification). These tasks required models to retrieve relevant knowledge using Wikipedia API interactions and reason about the information. The API allowed three actions: searching entities, looking up specific words, and submitting final answers. The models only received the question or claim as input and had to retrieve supporting evidence autonomously. Several baselines were compared, including Standard prompting (direct answer generation), Chain-of-Thought (CoT) prompting (reasoning without retrieval), and Act-only prompting (retrieval without structured reasoning). Additionally, a hybrid approach combining ReAct and CoT-SC (self-consistency in reasoning) was introduced to balance factual knowledge retrieval and structured reasoning. Fine-tuning was also explored using 3,000 generated trajectories to improve model performance on these tasks.

- **Results:** ReAct consistently outperformed Act-only prompting, demonstrating that integrating reasoning with retrieval significantly improves answer synthesis. On FEVER, ReAct outperformed CoT (60.9% vs. 56.3%), indicating that fact verification benefits from retrieving up-to-date information rather than relying solely on internal knowledge. However, for HotPotQA, CoT slightly outperformed ReAct (29.4% vs. 27.4%), suggesting that CoT provides better structured reasoning, though it suffers from hallucinations (false information) in 56% of cases, compared to only 6% for ReAct. A key issue for ReAct was reasoning errors and

Prompt Method[a]	HotpotQA (EM)	Fever (Acc)
Standard	28.7	57.1
CoT (Wei et al., 2022)	29.4	56.3
CoT-SC (Wang et al., 2022a)	33.4	60.4
Act	25.7	58.9
ReAct	27.4	60.9
CoT-SC → ReAct	34.2	**64.6**
ReAct → CoT-SC	**35.1**	62.0
Supervised SoTA[b]	67.5	89.5

(a) Table 2: PaLM-540B prompting results on HotpotQA and Fever.



(b) Scaling results for prompting and finetuning on HotPotQA with ReAct (ours) and baselines.

	Type	Definition	ReAct	CoT
Success	True positive	Correct reasoning trace and facts	94%	86%
	False positive	Hallucinated reasoning trace or facts	6%	14%
Failure	Reasoning error	Wrong reasoning trace (including failing to recover from repetitive steps)	47%	16%
	Search result error	Search return empty or does not contain useful information	23%	-
	Hallucination	Hallucinated reasoning trace or facts	0%	56%
	Label ambiguity	Right prediction but did not match the label precisely	29%	28%

(c) Table 3: Types of success and failure modes of ReAct and CoT on HotpotQA, as well as their percentages in randomly selected examples studied by human.

Figure 3: Result for ReAct Experiment

repetitive actions, which limited flexibility. The best approach was a hybrid model (ReAct + CoT-SC), which achieved the highest accuracy by dynamically switching between methods based on task difficulty. Fine-tuning significantly improved performance—ReAct fine-tuned on 3,000 examples outperformed all prompting methods, with smaller models (PaLM-8B) surpassing larger models (PaLM-62B) using standard prompting. These results highlight that combining reasoning and retrieval enhances factual accuracy, while fine-tuning further refines model performance for complex reasoning tasks.

4 Limitations

Tree of Thoughts (ToT):

1. The method has only been tested on three relatively simple tasks, and it is unclear how well it performs on more complex real-world problems.
2. Running multiple reasoning paths at the same time largely increases the computational cost, which makes it difficult to use for large-scale tasks.
3. It relies on an off-the-shelf LM rather than fine-tuning ones, which may limit its problem-solving capabilities to adapt to new tasks over time.

ReAct Framework:

1. The few-shot prompting setup may not work well for more complex tasks where training on larger datasets is needed.

2. Its structured approach improves reliability but can lead to repetitive reasoning and actions, which may cause the model to get stuck in loops and make reasoning errors.

3. It struggles with complex tasks that have large action spaces since they require more demonstrations, which can exceed the input length limit of in-context learning.

4. ReAct relies on successful knowledge retrieval, but 23 percent of errors come from noninformative searches, which disrupt reasoning and make it difficult for the model to recover.

5 Future Research Directions

Tree of Thoughts (ToT):

1. We can explore ways to make the search process more efficient to reduce computational costs.

2. We can combine these methods with other approaches, such as reinforcement learning, to improve its performance.

3. We can introduce fine-tuning methods to complement few-shot learning and improve performance on more challenging tasks.

ReAct Framework:

1. We can improve its performance by fine-tuning on more high-quality human-annotated data to enhance its reasoning and decision-making abilities.

2. We can scale it up with multi-task training and reinforcement learning, which can create more robust models for a wider range of applications.

6 Summary of Class Discussion

- Q1: [**Limitation**] Did the authors provide Chain of Thought (CoT) with the same number of prompts as Tree of Thoughts (ToT) to ensure a fair comparison in the experiment?
  A1: No, the Tree of Thoughts (ToT) paper does not provide an equal number of prompts to Chain of Thought (CoT), leading to potentially unfair comparisons in the experiments. A potential reason could be since, ToT generates, evaluates, and prunes multiple thoughts, it effectively gets more computational power per query.

- Q2: [**Comparison**] ReAct focuses on combining chain of thoughts with action plan generation, and stated that their method could allow the LLM to query wikipedia and improve performance, but this can also be addressed by RAG, what would be the differences and similarities between ReAct and RAG?
  A2: In RAG, the user needs to provide relevant and specific external resources, but in ReAct, the LLM will search for general information from web search engines or APIs based on the input.

- Q3: [**Limitation**] Although ToT outperforms CoT or standard IO in a single run, it requires a lot more resources. What if we run CoT multiple times till we get the right answer, will that cost more than running a single ToT?

A3: In some experiments, the authors constrain ToT to function similarly to CoT by generating only a single thought branch and pruning all others. In this setup, ToT essentially behaves like CoT. The authors demonstrate that selecting a high-value thought from ToT requires more computational resources (computationally more expensive and valuable) compared to generating a response from CoT. Therefore, they provide a form of comparison between the two approaches, showing that ToT inherently demands more computing power, even when structured to mimic CoT.

- Q4: [**Robustness**] React usually relies on external resources, Wikipedia can know many but not everything, when ReAct cannot find the correct answer, what will ReAct do?
  A4: It depends on the nature of the question being asked. If the question is highly specific, ReAct would need access to a more specialized source, such as a company document or a domain-specific database. Since ReAct operates by leveraging external tools, if the available tools do not provide the correct information, it must evaluate their reliability and, if necessary, replace them with more suitable ones. Additionally, new tools can be integrated anytime. For furture works maybe more sources like web pages, articles or journals could be added in the searches.

- Q5: [**Feedback**] ReAct Uses external environment feedback meaning it will get feedback from API calls, web results, database queries, etc. Do you think this kind of feedback is enough for ReAct? Can you think of more ways to improve upon this feedback system? Thoughts on Human-in-the-loop feedback approach?
  A5: ReAct follows a Thought-Action-Observation loop, adjusting its approach based on external feedback. To enhance this, an additional feedback layer could refine observations by correcting or augmenting the observation prompt. However, since ReAct relies heavily on LLMs, there will eventually be a need for human-in-the-loop feedback to validate conclusions and ensure accuracy after multiple iterations.

7 Appendix: Additional Class Questions

- Q6: [**Method**] How can the Thought-Action-Observation loop be formally modeled mathematically in ReAct?
  A6: To provide a general syntax, the agent (LLM) iteratively updates its state based on past reasoning (Thought), executes an interaction (Action), and receives new information (Observation) from the environment. It delivers feedback in a way that prioritizes high-reward outcomes. From the model's perceptive generating more thoughts is just some additional tokens to be generated.

- Q7: [**Additional**] Are there any ToT in deep seek?
  A7: The data used for training is still not published, based on observation it is using tree search to generate training data.

- Q8: [**Additional**] If two models have the same number of parameters and computational cost, which would be more cost-efficient: running DeepSeek with Tree of Thoughts (ToT) or using a dedicated reasoning model?
  A8: DeepSeek has recently increased its pricing, making the use of DeepSeek with Tree of Thoughts (ToT) comparatively more expensive. From a distillation perspective, a tree search

approach is generally more efficient because a linear reasoning structure (such as standard CoT) attempts to mimic a tree structure but with certain compromises in exploration.

If ToT's reasoning process could be optimized—potentially through reward-based training—we might see improvements over current DFS/BFS search techniques. Looking ahead, one hypothesis is that reinforcement learning could make ToT more efficient in decision-making, though it may also increase computational costs compared to existing models.

Another important consideration is why OpenAI has not adopted ToT widely. One likely reason is the high token generation cost associated with ToT, as it requires multiple forward passes and extensive computational power for each prompt. This makes ToT expensive and potentially impractical for large-scale deployment.

References

[1] Besta, M., Blach, N., Kubicek, A., Gerstenberger, R., Podstawski, M., Gianinazzi, L., Gajda, J., Lehmann, T., Niewiadomski, H., Nyczyk, P., and Hoefler, T. Graph of thoughts: Solving elaborate problems with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence 38*, 16 (Mar. 2024), 17682–17690.

[2] Nakano, R., Hilton, J., Balaji, S., Wu, J., Ouyang, L., Kim, C., Hesse, C., Jain, S., Kosaraju, V., Saunders, W., Jiang, X., Cobbe, K., Eloundou, T., Krueger, G., Button, K., Knight, M., Chess, B., and Schulman, J. Webgpt: Browser-assisted question-answering with human feedback, 2022.

[3] Shinn, N., Cassano, F., Berman, E., Gopinath, A., Narasimhan, K., and Yao, S. Reflexion: Language agents with verbal reinforcement learning, 2023.

[4] Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models, 2023.

[5] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models, 2023.

[6] Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., and Narasimhan, K. Tree of thoughts: Deliberate problem solving with large language models, 2023.

[7] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. React: Synergizing reasoning and acting in language models, 2023.