Informed Tree of Thought: Cost-efficient Problem Solving with Large Language Models

Sajad Mousavi, Desik Rengarajan, Ashwin Ramesh Babu, Sahand Ghorbanpour, Vineet Gundecha, Avisek Naug, Soumyendu Sarkar

Hewlett Packard Enterprise

Introduction

Large Language Models (LLMs) are powerful for problem-solving but often inefficient in reasoning over complex tasks and prompts.

Challenges

- High **costs** and **failures** when integrating external tools for reasoning.
- Inefficient decision-making processes in existing approaches like Chain of Thought (CoT) and Tree of Thought (ToT).

iToT introduces a novel framework that:

- Optimizes reasoning with informed search algorithms (iToT-A* and iToT-D* Lite).
- Minimizes tool usage costs and adapts to tool failures through dynamic re-planning.

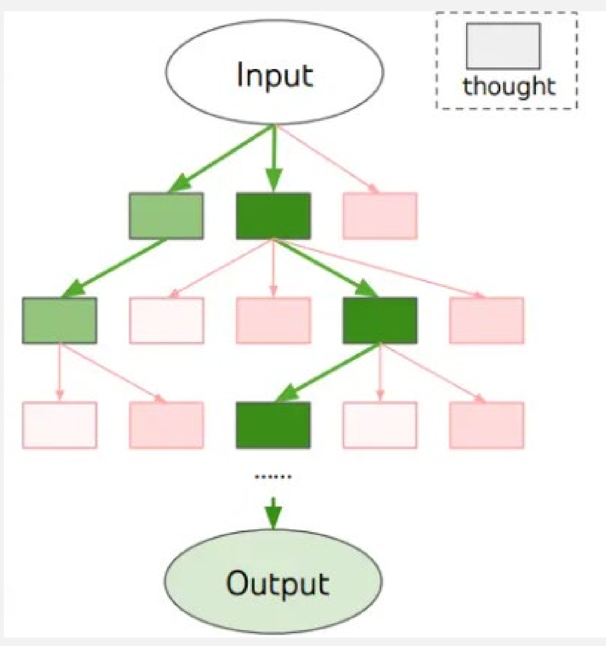


Fig. 1: Tree of Thoughts

Problem Statement

These following gaps necessitate a cost-efficient, failure-aware framework:

- **Inefficient Search**: Tree-based methods like **Tree of Thought** explore reasoning paths but use inefficient algorithms that do not consider the costs or relevance of reasoning steps.
- **Failure Handling**: Current frameworks such as CoT and ToT do not account for tool failures, reducing reliability in real-world scenarios.
- **Cost Awareness**:
  - Language Agent Tree Search (LATS) combines external tools with tree search to enhance reasoning.
  - However they ignore the financial, computational costs and potential failures associated with tool usage, leading to inefficient problem-solving.

Proposed Approach: iToT

iToT Framework: Enhances ToT by integrating tools, cost-awareness, and adaptive re-planning.

iToT Variants:

1. **iToT-A***: Based on the A* search algorithm, it prioritizes states with the lowest combined cost and highest potential for solution progress.
   - Selects the state with the lowest final value. $V_f(s') =$ Cumulative tool cost $V_g(s') -$ Heuristic representing the potential to solve the task $V(s')$.
2. **iToT-D Lite*:** An extension of D* Lite search, it handles tool failures by assigning additional costs to failed states.
   - Failure costs are propagated back to preceding states, updating the cumulative tool cost V_g and re-prioritizing the search.

Experiment Setup

Dataset: HotPotQA for multi-hop question answering.
Baselines: Compare iToT with CoT, ToT (DFS, Best-First Search), and direct prompting [1, 2, 3].

Tools:

- **Search**[entity]**:** searches and retrieve a passage for the requested entity's Wikipedia page.
- **Lookup**[string]: looks up the string and returns the next sentence from retrieved passage
- **Finish**[answer]: ends the task with the answer

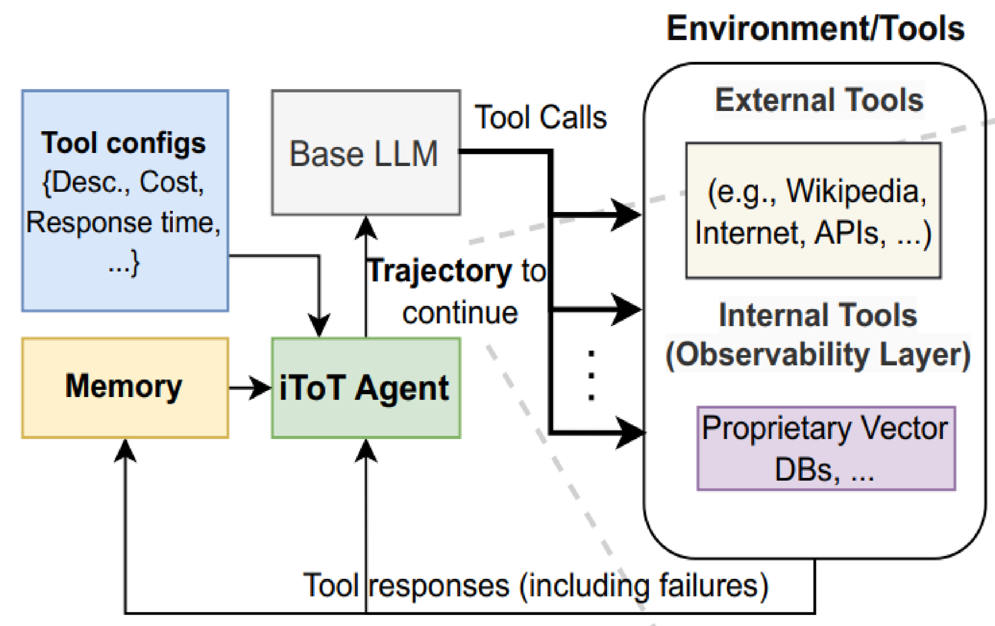
Failure integration

- E.g., if the search tool's API returns "could not find" for the requested item, we treat this as a tool failure.

Evaluation Metrics:

1. Exact Match (EM) score for correctness.
2. Dynamic re-planning efficiency in high failure scenarios.

iToT (informed Tree of Thought)



Memory is used to store the tree of expanded nodes (includes the sequence of thoughts, actions and observation) that is used as context in subsequent decision-making.

Trajectory refers to the sequence of actions and observations generated by the language model during its problem-solving process.

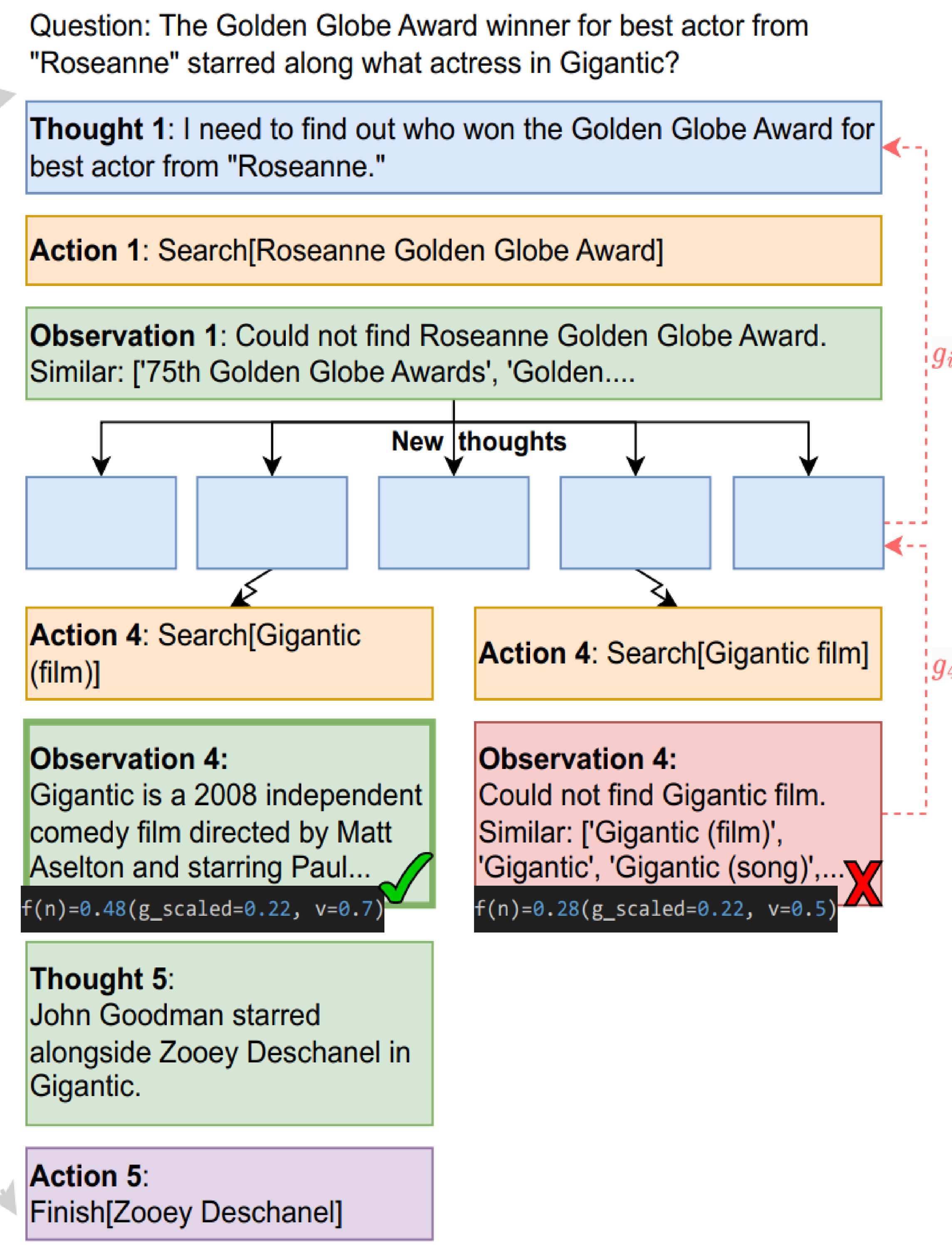


Fig. 2: **Left**: iToT (informed Tree of Thought) methods. **Right**: An example trajectory from the iToT-D* Lite method on HotPotQA. Starting from node value (v), cumulative tool cost (v_g), with failure propagation mechanisms. g_i refers to the propagation of the tool costs to previous states. $f(n)$ refers to the final value v_f of the state n.

iToT- A* and iToT- D* Lite

Algorithm 1 iToT-A*

- **Input:** LLM input x, depth limit L, round limit R, tool cost function V_T, Value function V
- **Initialize** $\mathcal{O} \leftarrow \{x\}$, Cumulative cost and final value vectors $V_g(s) = V_f(s) = \inf \quad \forall s$
- **for** $r = 1 \dots R$ **do**
  - $s \leftarrow \arg\min_{\tilde{s} \in \mathcal{O}} V_f(\tilde{s})$
  - **if** s is terminal **then**
    - **return** s
  - **end if**
  - Let i be the length of state $s = [x, z_1, \dots, z_i]$
  - **if** $i > L$ **then**
    - **continue**
  - **end if**
  - Generate $\mathcal{C}(s) = \{s' = s \cup z_{i+1} | z_{i+1} \sim \pi(\cdot|s)\}$
  - **for** $s' \in \mathcal{C}(s)$ **do**
    - $V_g(s') = V_g(s) + V_T(s'|s)$
    - $V_f(s') = V_g(s') - V(s')$
    - $\mathcal{O} \leftarrow \mathcal{O} \cup s'$
  - **end for**
- **end for**
- **return** Null

Left: iToT-A*: Utilizes A* search to balance tool cost and state value for optimal decision-making.

Right: **iToT-D Lite***: Handles tool failures by propagating failure costs and dynamically updating plans.

Algorithm 2 iToT-D* Lite

- **Input:** LLM input x, depth limit L, round limit R, tool cost function V_T, Value function V, Tool failure cost δ_T
- **Initialize** $\mathcal{O} \leftarrow \{x\}$, Cumulative cost and final value vectors $V_g(s) = V_f(s) = \inf \quad \forall s$
- **for** $r = 1 \dots R$ **do**
  - $s \leftarrow \arg\min_{\tilde{s} \in \mathcal{O}} V_f(\tilde{s})$
  - **if** s is terminal **then**
    - **return** s
  - **end if**
  - Let i be the length of state $s = [x, z_1, \dots, z_i]$
  - **if** $i > L$ **then**
    - **continue**
  - **end if**
  - Generate $\mathcal{C}(s) = \{s' = s \cup z_{i+1} | z_{i+1} \sim \pi(\cdot|s)\}$
  - **for** $s' \in \mathcal{C}(s)$ **do**
    - **if** s' has a tool failure **then**
      - $V_g(s') = V_g(s) + \delta_T + V_T(s'|s)$
      - PROPAGATE_FROM(s') ▷ See Appendix A
    - **else**
      - $V_g(s') = V_g(s) + V_T(s'|s)$
    - **end if**
    - $V_f(s') = V_g(s') - V(s')$
    - $\mathcal{O} \leftarrow \mathcal{O} \cup s'$
  - **end for**
- **end for**
- **return** Null

Results

Method	HotpotQA (EM)↑
Base LM	0.42
Base LM (Few shot)	0.44
Base LM (CoT) (Wei et al. [2022])	0.47
ToT DFS (CoT + ReAct) (Yao et al. [2023])	0.46
ToT Best First Search (CoT + ReAct)	0.59
iToT A* Search (CoT + ReAct)	**0.62**
iToT-D* Lite Search (CoT + ReAct)	**0.62**

Table 1: Exact Match (EM) scores on the HotpotQA dataset for various prompt methods.

- iToT-A* and iToT-D* Lite achieve the highest EM (0.62), outperforming all baselines.
- Robust performance under tool failures with significant cost reductions.
- D* Lite excels in dynamic environments with frequent tool failures, and its equivalent performance is due to the limited tool failures in experiments.
- Attempted to run LATS (CoT + ReAct), but it was too slow to Complete.

Conclusion and Future work

- iToT offers a novel, cost-efficient solution for LLM reasoning with external tools.
- Achieves state-of-the-art performance on HotPotQA by balancing exploration, exploitation, and tool usage.

Future work:

- Extend iToT to dynamic real-world scenarios with higher tool failure rates and information updates.

Broader Impacts

- Reduces computational costs and enhances tool integration for scalable reasoning.
- Promotes robust LLM reasoning by dynamically adapting to failures.
- Sets a benchmark for future cost-aware, failure-resilient problem-solving frameworks.

References

[1] Wei, Jason, et al. "Chain-of-thought prompting elicits reasoning in large language models." Advances in neural information processing systems 35 (2022): 24824-24837..
[2] Yao, Shunyu, et al. "Tree of thoughts: Deliberate problem solving with large language models, 2023." URL https://arxiv. org/pdf/2305.10601. pdf (2023).
[3] Zhou, Andy, et al. "Language agent tree search unifies reasoning acting and planning in language models." arXiv preprint arXiv:2310.04406 (2023).