THOUGHTSCULPT: Reasoning with Intermediate Revision and Search

Yizhou Chi

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2024-55
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2024/EECS-2024-55.html

May 7, 2024

THOUGHTSCULPT: Reasoning with Intermediate Revision and Search

by Yizhou Chi

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Dan Klein

Professor Dan Klein
Research Advisor

5/3/2023

(Date)

* * * * * * *

Alane Suhr

Professor Alane Suhr
Second Reader

5/1/24

(Date)

THOUGHTSCULPT: Reasoning with Intermediate Revision and Search

Yizhou Chi, Kevin Yang & Dan Klein
UC Berkeley
Berkeley, CA 94720, USA
{yizhouchi,yangk, klein}@berkeley.edu

Abstract

We present THOUGHTSCULPT, a general reasoning and search method for tasks with outputs that can be decomposed into components. THOUGHTSCULPT explores a search tree of potential solutions using Monte Carlo Tree Search (MCTS), building solutions one action at a time and evaluating according to any domain-specific heuristic, which in practice is often simply an LLM evaluator. Critically, our action space includes revision actions: THOUGHTSCULPT may choose to revise part of its previous output rather than continuing to build the rest of its output. Empirically, THOUGHTSCULPT outperforms state-of-the-art reasoning methods across three challenging tasks: Story Outline Improvement (up to +30% interestingness), Mini-Crosswords Solving (up to +16% word success rate), and Constrained Generation (up to +10% concept coverage).

1 Introduction

Whilst Large Language Models (LLMs) such as GPT (Brown et al., 2020; OpenAI, 2024), LLaMA (Touvron et al., 2023a;b), and Claude (Anthropic, 2024) are developed to be increasingly capable in performing a variety of reasoning tasks, recent studies have revealed that the utilization of distinct prompting strategies and instructional guidance can have a notable influence on the performance of LLMs when tackling identical tasks.

Chain-of-Thought (CoT) is a prompting strategy detailed in (Wei et al., 2023) that directs LLMs to produce the final task output through intermediate steps of reasoning, referred to as "intermediate thoughts." Notably, CoT has demonstrated a substantial enhancement in the problem-solving proficiency of LLMs without necessitating any model updates. Self-consistency with CoT (CoT-SC) (Wang et al., 2023a) is proposed to improve output consistency by generating multiple CoTs and selecting the best outcome. Recently, in extension to CoT and CoT-SC, Tree-of-Thoughts (Yao et al., 2023a) and Graph-of-Thoughts (Besta et al., 2024) are proposed to shape the reasoning process of LLMs as a tree or an arbitrary graph structure. These approaches enable LLMs to explore different paths of thought and find better outputs by utilizing backtracking and graph-search algorithms. However, these approaches' reasoning capabilities are often limited by the set of candidates they generate at earlier steps. They cannot revise and edit their original answers continuously in later steps. As a result, these methods may not be effective in addressing problems that require frequent revision and modifications.

We propose THOUGHTSCULPT, a novel graph-based framework that allows LLMs to build an interwoven network of thoughts, emulating the human reasoning process. A key feature of our approach is a self-revision mechanism that enables LLMs to iteratively refine and improve upon their previous outputs while generating new thought nodes.

Additionally, we address the challenge of the vast search space in text generation, where exhaustively exploring all possible paths is computationally intractable for graph search algorithms like Depth First Search (DFS), Breadth First Search (BFS), and A* Search. To tackle these issues, we incorporate Monte Carlo Tree Search (MCTS), a powerful heuristic technique that can efficiently navigate the search space and provide high-quality solutions, albeit not necessarily the globally optimal one.

Our proposed method comprises three core modules: thought evaluator, thought generator, and decision simulator. The thought evaluator provides textual feedback on candidate partial outputs, as well as a numerical feedback score that serves as a heuristic for search algorithms. The thought generator produces potential solutions based on the initial instructions and self-evaluative feedback. Finally, the decision simulator acts as a part of the MCTS process, simulating consecutive lines of thought to evaluate the potential value of a particular path.

We conduct experiments with THOUGHTSCULPT across three distinct tasks that are challenging to state-of-the-art language models, including GPT-3.5 and GPT-4 (OpenAI, 2024): Story Outline Improvement, Mini-Crosswords Solving, and Constrained Generation. These tasks demand mature reasoning abilities, varying levels of exploration, and self-revision to derive optimal solutions. Compared to state-of-the-art reasoning strategies as baselines, THOUGHTSCULPT exhibits an up to 30% interestingness increase in Story Outline Improvement; up to 16% word success rate increase in Mini-Crossword Solving; and up to 10% concept coverage improvement in Constrained Generation. These findings underscore the efficacy of THOUGHTSCULPT across diverse tasks. All code will be open-sourced upon publication.

2 Related Works

Feedback Guided Generation. Human feedback has been shown to be effective in improving LLMs' generation (Tandon et al., 2022; Elgohary et al., 2021; Bai et al., 2022). However, human feedback is often costly and unable to be incorporated into an automated generation process. As a result, some works adopt a heuristic function to serve as an alternative to human feedback (Liu et al., 2022; Lu et al., 2022; Le et al., 2022; Welleck et al., 2022).

Madaan et al. (2023); Shinn et al. (2023); Paul et al. (2024) introduce a mechanism for LLMs to produce self-reflective feedback to improve their outputs. Along with the model-generated feedback, (Chen et al., 2023) uses execution results to help improve code generation. Likewise, (Kim et al., 2023) introduces a critic step to improve the model's performance in computer tasks. These approaches follow left-to-right linear processes, potentially overlooking alternative directions. In our work, each thought node having multiple children nodes allows for broader exploration, enhancing decision-making comprehensiveness.

Graph Reasoning. To enable LMs to make broader exploration during problem-solving, (Yao et al., 2023a) and (Xie et al., 2023) adopt a tree-search procedure to consider multiple decisions at different levels. These approaches represent each node as a partial solution to the instruction such that a complete solution is required to concatenate a set of nodes. It rigidly prohibits any editing or modification within the intermediate nodes, thereby forcing the final output to be even dependent on the candidates generated in the initial step. (Besta et al., 2024) has proposed a graph-based paradigm that models LLM reasoning as an arbitrary graph. Whilst keeping each node as a partial solution, this approach allows the combination of a set of connecting nodes. In our approach, we allow for review and modification at intermediate nodes– in fact, an intermediate node could even be an already-complete solution which is then revised or modified further. This flexibility gives more expressivity and enables language models to make mistakes initially and then correct them later on.

LM Planning. Long-form generation and complex problem-solving often require high-level planning or outlining. Natural language outliners and structured schemas play integral roles in generating long-form content (Tian & Peng, 2022; Mirowski et al., 2022; Yang et al., 2022; 2023). There are also works that utilize LLMs to tackle complex tasks such as video games, fact-checking, house keeping, and code optimization with planning using natural languages (Yao et al., 2023b; Huang et al., 2022a; Wang et al., 2023b; Huang et al., 2022b).

Our work could also be seen as a generic task planner using LLMs that leverages Monte Carlo Tree Search to facilitate various tasks in diverse domains.

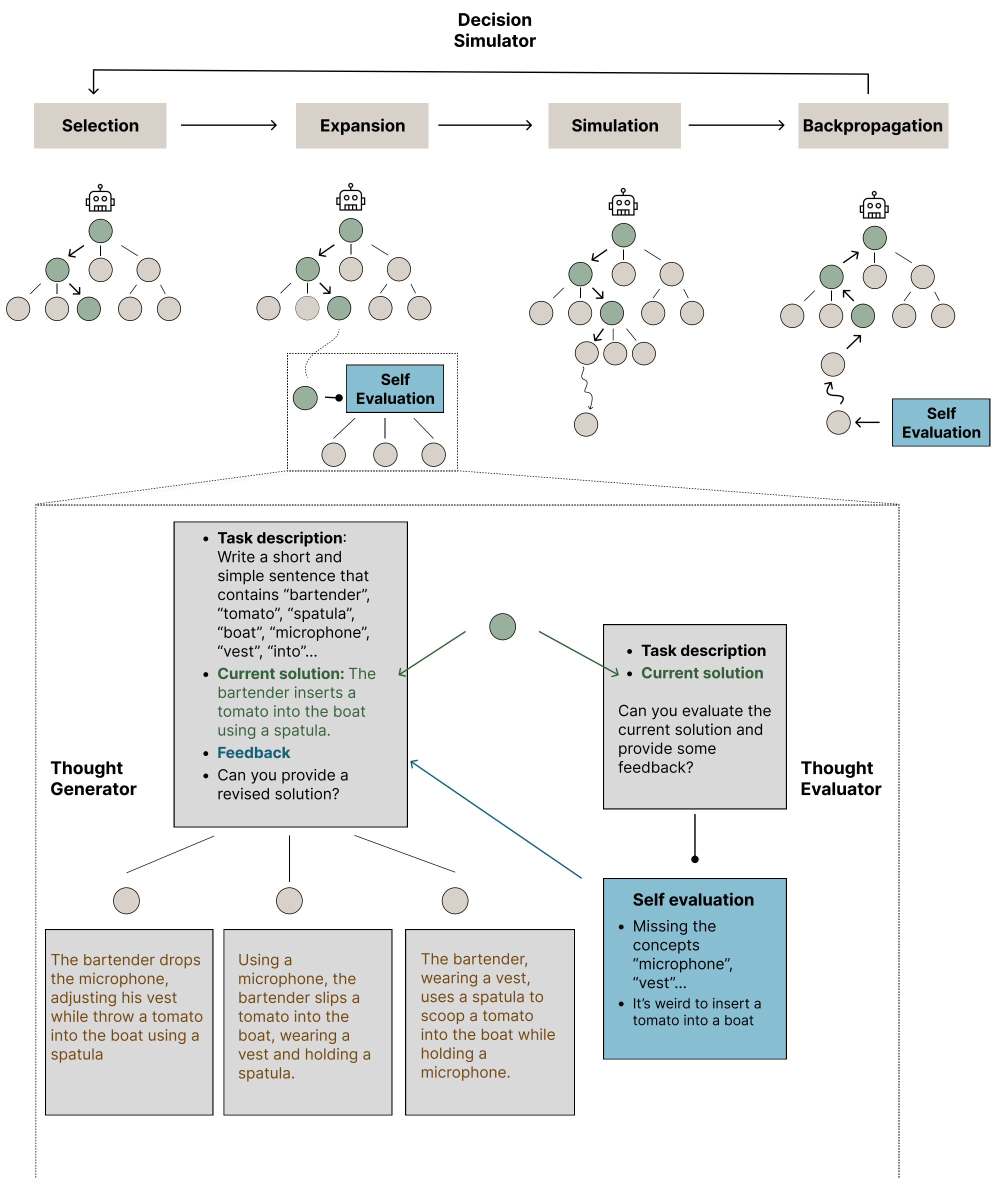


Figure 1: Illustration of THOUGHTSCULPT using Monte Carlo Tree Search. Each circle in the diagram represents a thought node generated by LLMs. *Selection:* choose a thought node x based on a selection algorithm such as UCB1 Eqn 4. *Expansion:* A new set of child nodes X is generated using the initial instruction, the current node, and self-evaluated textual feedback. The zoom-in of the expansion phase demonstrates the use of the **Thought Evaluator** and the **Thought Generator**, which entails assessing and refining the current solution for the task 4.3. *Simulation:* a single node x' is randomly chosen from the set X. This selected node x' generates further nodes in a sequence for several steps, corresponding to our **Decision Simulator**. *Backpropagation:* the numerical feedback evaluated at the last node is propagated back to the root node.

3 Method

We treat each formal output of LMs as a thought node $x \in \{x^0, x^1, ...x^i\}$, where x^0 is the root node and the initial output provided by LMs given the task instruction I. For instance, a thought node can be a few lines of items (Story Outline Improvement), a couple of words (Mini-Crosswords), or a sentence (Constrained Generation). To process the thought node and look for a better output, our method consists of three main modules: thought evaluator, thought generator, and decision simulator.

3.1 Thought Evaluator

The thought evaluator evaluates the status of each thought node and provides feedback for potential improvement. It not only works as a heuristic for the search algorithm but also gives potential directions and guidance to generate new candidates.

A feedback $f(x^i)$ of a node x^i consists of a numerical feedback $f_{numeric}(x^i)$ and a textual feedback $f_{text}(x^i)$. The numerical feedback will be used as the evaluation score $v(x^i)$ for the current node, and the textual feedback will be used as context to generate child nodes.

$$f(x^i) = < f_{text}(x^i), f_{numeric}(x^i) > \quad (1)$$

$$f_{numeric}(x^i) = v(x^i) \quad (2)$$

We present two types of textual feedback, each beneficial for various task scenarios. Furthermore, these strategies are flexible, allowing for independent or combined utilization.

1. *Holistic Evaluation:* Analyze the entirety of the thought node and offer comprehensive feedback addressing the node as a cohesive unit. This approach can be applied in most scenarios, allowing for a holistic assessment that captures the essence and coherence of the node.
2. *Itemized Evaluation:* Assess each sub-unit within the thought node individually, offering targeted feedback tailored to each component. As a result, the evaluation will be a list of feedback addressing each sub-unit of the node. This approach is useful when the thought node can be dissected into distinct items, enabling localized evaluation. For example, in the case of a story outline, breaking it down into various outline items allows for individual assessment and refinement.

The right part of the zoomed-in expansion phase depicted in Figure 1 shows how holistic evaluation feedback is generated by the thought evaluator based on the task description and the current solution.

3.2 Thought Generator

Once we have evaluation feedback of the current node, we can form subsequent thought nodes that aim to improve the current output. Based on the task description I, the current solution x_{parent}, and the textual feedback f_{text} provided by the self-evaluator, each thought node generates k candidate thought nodes using a pre-trained LM with a parameter θ.

A child node x_{child} will be generated as follows:

$$x_{child} \sim p_\theta(x | I, x_{parent}, f_{text}(x_{parent})) \quad (3)$$

The left part of the zoomed-in expansion phase depicted in Figure 1 illustrates how THOUGHTSCULPT leverages the task description, current solution, and evaluation feedback to produce a set of candidate nodes.

3.3 Decision Simulator

THOUGHTSCULPT is equipped with a decision simulator that enables it to simulate decisions at deeper layers and then backpropagate to update the score of the current decision. In other words, we are doing a rollout to get a better estimate of the reward for the node we are at. The behavior of the decision simulator is analogous to the processes in Monte Carlo Tree Search (MCTS; see Algorithm 1). It is possible to replace the decision simulator with other search algorithms such as DFS, BFS, or A* search (and we in fact run DFS as well in our experiments in Section 4), but MCTS offers a computational advantage by efficiently exploring complex search spaces, balancing exploration and exploitation to converge to optimal solutions with fewer evaluations, and scaling well to large problem instances due to its incremental and iterative nature.

MCTS explores potential moves and stores the outcomes in a search tree. With each search iteration, the tree expands, accumulating more information. As shown in Figure 1, MCTS can be divided into four phases: selection, expansion, simulation, and backpropagation.

In the selection phase, a leaf node will be selected based on Upper Confidence Bound 1 (UCB1) Eqn 4 which prioritizes nodes that have not been explored extensively but show promise. Therefore, the UCB1 value of node x takes into account not only the heuristic score $v(x)$ but also the total number of visits to the node itself, $n(x)$, as well as its parent node, $n(x_{parent})$.

$$UCB1(x) = v(x) + c\sqrt{\frac{\ln n(x_{parent})}{n(x)}} \tag{4}$$

In the expansion phase, the thought generator will expand the selected leaf node by generating a set of children nodes based on the feedback provided by the thought evaluator.

In the simulation phase, a child node is picked from the newly generated set using a uniform distribution. In the subsequent iterations, however, we generate only a single node iteratively until the maximum simulation depth $d_{simulation}$ is reached.

Finally, in the backpropagation phase, we update the reward of the latest node back to the root node or the current node, and iterate this process for $d_{rollout}$ steps. The node with the highest average reward will be chosen as the final output.

4 Experiments

We evaluate our method on three diverse tasks: Story Outline Improvement, Mini-Crossword Solving, and Constrained Generation.

We evaluate the tasks with Chain-of-Thought (CoT) (Wei et al., 2023), Self-Refine (Madaan et al., 2023), and Tree-of-Thoughts (ToT) with DFS (Yao et al., 2023a) as baselines. We use GPT-3.5 (`gpt-3.5-turbo-0125`) and GPT-4 (`gpt-4-0125-preview`) (OpenAI, 2024) as strong base LMs for the reasoning algorithms across all tasks. Both base LMs use a temperature of 0.7. To further evaluate the efficacy of our proposed approach, we conduct an ablation study by investigating the performance of our method when employing Depth-First Search (DFS) Algorithm 2 as an alternative search algorithm to the MCTS algorithm. In addition, running THOUGHTSCULPT with DFS facilitates closer comparison with ToT, which also uses DFS. While THOUGHTSCULPT with MCTS typically performs better, we observe in our experiments below that THOUGHTSCULPT's performance with DFS still outperforms our other baselines, demonstrating THOUGHTSCULPT's ability to generalize to other search algorithms.

4.1 Story Outline Improvement

One approach to generating long-form stories via LLMs is to adopt a high-level writing process that first designs an outline of the story and fills up the details based on the outline (Yang et al., 2022; 2023). An unengaging or uncompelling outline is unlikely to yield a captivating final draft, regardless of the subsequent detailing efforts. To address this challenge, we propose a task focused specifically on enhancing the interestingness of story outlines generated by LLMs.

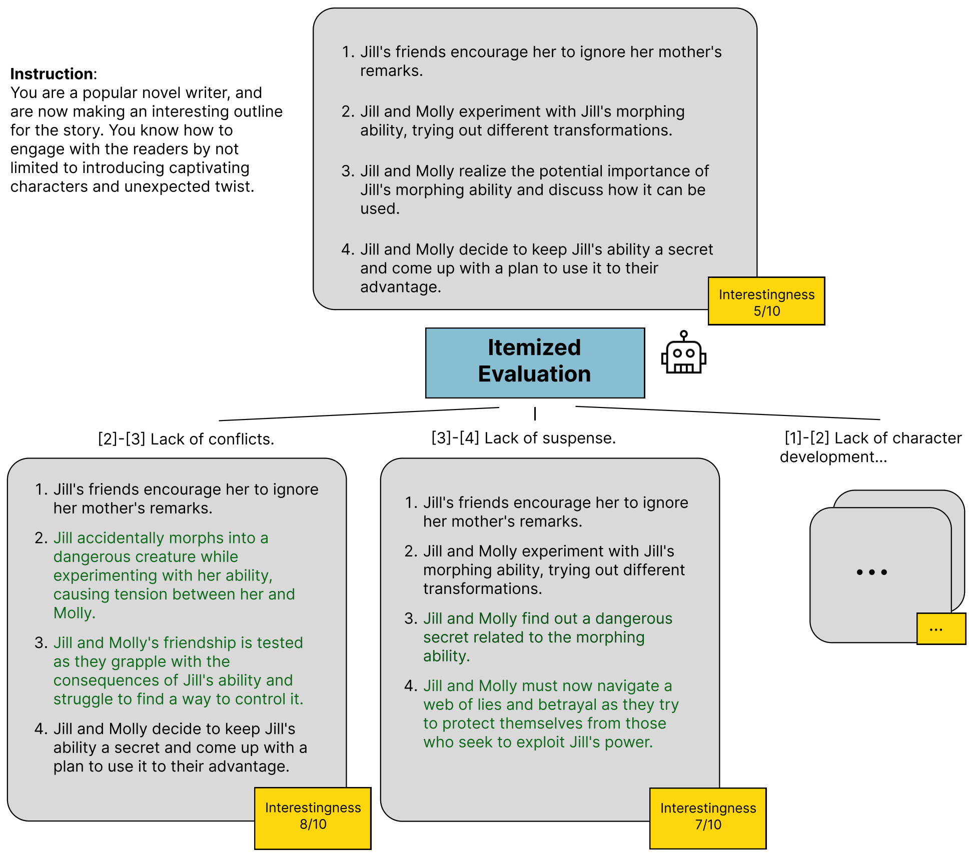


Figure 2: Illustration of our Story Outline Improvement task. A step involves employing the thought evaluator to conduct itemized evaluations of the story outline and utilizing the thought generator to generate a candidate set of improved story outlines for task 4.1.

Task Setup We sample 500 book descriptions from the WhatsThatBook dataset (Lin et al., 2023) as premises and generate the story outlines using DOC (Yang et al., 2023) with GPT-3.5 as the base LM. We use 400 book descriptions for training, 50 descriptions for validation, and the other 50 descriptions for testing. For each premise, we generate three outlines: one specifically prompted to make the outline interesting (interesting outline), one specifically prompted to make the outline boring (non-interesting outline), and the last one without extra instruction to make the outline either interesting or not (default outline).

Since there is no ground truth regarding the interestingness of the outline, we need an outline content evaluator to evaluate the final interestingness of the generated or revised outlines. Neither THOUGHTSCULPT nor baselines have access to the evaluator when generating the outlines. We fine-tune the pre-trained Flan-T5 model (Chung et al., 2022) on the training set to serve as our content evaluator. The outline content evaluator is trained to rate an interesting outline as 1 and a boring outline as 0. The output provided by the trained evaluator will serve as the score metric for this task. For evaluation, we instruct LMs to revise and improve the interestingness of the generated default outlines in the test set. As a result, there are 400 interesting and 400 non-interesting outlines for fine-tuning the content evaluator, 50 interesting and 50 non-interesting outlines for validating the evaluator, and 50 default outlines for experimenting with reasoning algorithms.

Method Setup Each method is allowed to search or iterate through a maximum depth of 3. The thought evaluator will perform an itemized evaluation on the current outline and provide an interesting score from 1 to 10 as the numerical feedback. Based on each itemized feedback, a child node will be proposed to modify the current outline in order to improve its interestingness. For THOUGHTSCULPT and ToT, each node will generate a maximum of 3 candidate child outlines. In this and all the experiments below, THOUGHTSCULPT with MCTS will have a maximum $d_{simulation}$ of 1.

Results As illustrated in Table 1, all methods unsurprisingly improve the level of interestingness relative to the initial outline (sampled from default outlines with no prompting for either interesting or boring). However, overall, THOUGHTSCULPT outperforms ToT even with DFS, while THOUGHTSCULPT with MCTS demonstrates the highest average interestingness percentage across both GPT-3.5 and GPT-4 with 89.9 and 65.0 respectively.[1]

	Base LLM	
Methods	**GPT3.5**	**GPT4**
Initial Outline	12.0	12.0
CoT	50.1	28.8
Self-refine	65.5	27.9
ToT	72.1	49.9
THOUGHTSCULPT (DFS)	79.3	53.7
THOUGHTSCULPT (MCTS)	**89.9**	**65.0**

Table 1: Average outline interestingness. Initial Outline is the starting point before rewriting with any reasoning method. THOUGHTSCULPT's outputs are judged to be interesting at a higher percentage compared to baselines.

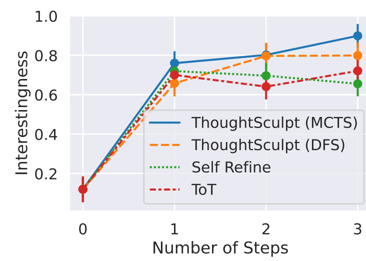


Figure 3: Average outline interestingness at each step. THOUGHTSCULPT's interestingness increases more with steps compared to baselines.

Continuous Improvement Figure 3 illustrates the progression of story outline interestingness at various steps, employing GPT-3.5 as the base LM. Among the tested methods, only THOUGHTSCULPT with MCTS has exhibited a consistent pattern of improvement over time. In contrast, both ToT and Self-refine exhibit a lack of continuous improvement. We suppose that Self-refine's limited search space and ToT's absence of a revision process contribute to this phenomenon.

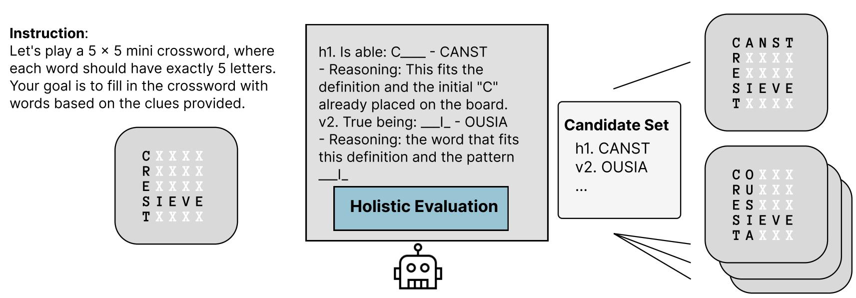


Figure 4: Illustration of a step in the deliberation process in the Mini-Crosswords task, where the current crossword board is assessed using the thought evaluator and a candidate set of words is proposed for task 4.2. One step is equal to one $d_{rollout}$

4.2 Mini crosswords

We also explore our method on 5x5 mini crosswords following the setup of (Yao et al., 2023a). For every puzzle, there are five horizontal (h1 to h5) and five vertical (v1 to v5) words to be filled. The task is to solve a five-by-five crossword puzzle in several steps (either filling

[1]One possible explanation for why GPT-4, serving as the base LM, exhibits lower overall interestingness could be attributed to the fact that the outline content evaluator was trained on outlines generated using GPT-3.5.

or editing a word counts as one step). For evaluation, we check the proportion of letters, words, and games correctly filled by each reasoning method.

Method Setup Each thought node represents a (possibly partial) solution to the crossword puzzle. To evaluate each thought node, the LM is prompted to evaluate each clue against the filled-in letters and suggest whether it is reasonable. For example, if the first row is filled with "AMIGO" and nothing else is filled, then the first column will be shown as "A____". Thus, in the prompt, there will be one line "v1. A Mennonite sect, named for Jacob Ammann: A____" that asks the LM to determine whether there are potential answers. The node evaluation's prompt setup is similar to (Yao et al., 2023a)'s except that we use the evaluation feedback to generate new candidates instead of pruning branches. Based on the evaluation feedback, every candidate for a node will be generated to either suggest a new word to fill a blank space or propose a modification to a word already filled in. For each node, THOUGHTSCULPT and ToT generate a maximum of 3 candidates. In contrast to the setup in (Yao et al., 2023a), where maximum search steps is set to 100, we impose a constraint on all methods to utilize only 20 search steps. This constraint aims to prevent attempts to artificially boost performance by exhaustively trying numerous word possibilities. With this restriction, each row or column of the crossword puzzle allows, on average, only two word attempts to be made within the allocated search budget.

Results As shown in Table 2, THOUGHTSCULPT with MCTS attains the highest letter success rate using GPT-3.5 and the highest word and game success rate using GPT-4; it is also always at least comparable to the best in all cases. With limited search steps, it is surprising that ToT using GPT-4 performs worse than even Self-refine; it turns out that a self-revision mechanism is very important in this task. In fact, THOUGHTSCULPT with MCTS achieves comparable performance to that reported by ToT (Yao et al., 2023a) using 100 search steps, despite employing just 20 search steps in our experiment.

	GPT3.5			GPT4		
Methods	% word	% letter	% game	% word	% letter	% game
CoT	10.5	34.6	0.0	15.6	40.6	5.0
Self-refine	13.5	27.4	**5.0**	46.5	**74.8**	5.0
ToT	**19.5**	36.6	0.0	39.5	64.8	5.0
THOUGHTSCULPT (DFS)	14.0	33.2	0.0	46.5	68.2	20.0
THOUGHTSCULPT (MCTS)	19.0	**41.6**	0.0	**54.0**	74.0	**25.0**

Table 2: Mini-crossword results of 20 puzzles for THOUGHTSCULPT and baselines (success % of letters, words, and games). THOUGHTSCULPT with MCTS is either best or closely comparable to best across the board.

4.3 Constrained Generation

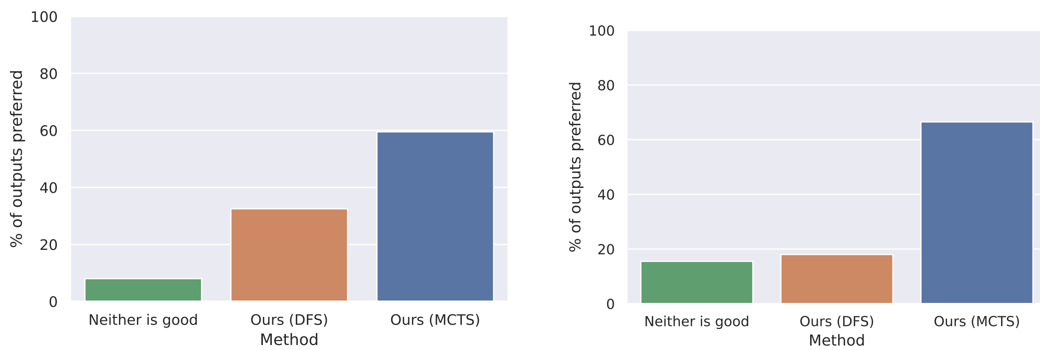
CommonGen is a benchmark dataset and a constrained text generation task designed to evaluate LMs' abilities in generative commonsense reasoning (Lin et al., 2020). An example instruction for the task is shown in Appendix A.3. However, currently, the coverage test of CommonGen can be completed with 90% or higher accuracy by many LLMs with one-shot prompting. Therefore, we instead test on CommonGen-Hard as introduced by (Madaan et al., 2023). Rather than just four concepts, CommonGen-Hard requires models to generate a sentence with 20-30 concepts.

Method Setup In this task, we first provide the set of concepts required and the task description for the LM to generate an initial thought node. During the thought evaluation, the LM will be prompted to give feedback about the quality of the concepts used and whether there are any missing concepts. A child node will be generated using the feedback along with the current solution. We set a maximum depth of 3 for this task. For each node, both THOUGHTSCULPT and ToT will generate a maximum of 3 child candidates.

Results Table 3 shows that THOUGHTSCULPT outperforms all other baselines when using either GPT-3.5 or GPT-4 as the base LM. While THOUGHTSCULPT with DFS achieves the highest coverage of 79.6% (GPT-3.5) and 99.1% (GPT-4), THOUGHTSCULPT with MCTS also demonstrates comparable concept coverage of 77.9% using GPT-3.5 and 99.0% using GPT-4. While MCTS exhibits notable exploration capabilities, it fails to surpass DFS due to the task's nature, where effective solutions are abundant as long as generated sentences correctly integrate assigned concepts. DFS, employing a greedy approach prioritizing nodes with the highest concept coverage, outperforms MCTS in this context. However, solely relying on concept coverage does not ensure appropriate concept utilization. Hence, to determine the preferred output based on both concept coverage and appropriateness, we conduct a comprehensive evaluation using GPT-4. Figure 5 illustrates that outputs from THOUGHTSCULPT with MCTS are significantly favored over THOUGHTSCULPT with DFS and a third baseline (intuitively, signifying that neither THOUGHTSCULPT version's output is good).

Methods	GPT3.5	GPT4
CoT	44.1	96.1
Self-refine	70.0	98.5
ToT	54.8	98.8
THOUGHTSCULPT (DFS)	**79.6**	99.1
THOUGHTSCULPT (MCTS)	77.9	99.0

Table 3: Constrained Generation Results (% Coverage of Concepts). THOUGHTSCULPT outperforms all baselines on both base LMs.



(a) Base LLM GPT-3.5

(b) Base LLM GPT-4

Figure 5: GPT-4's comprehensive preference based on concept coverage and appropriateness over the final outputs for Constrained Generation. THOUGHTSCULPT with MCTS is preferred by a wide margin.

5 Discussion

We introduce THOUGHTSCULPT, a framework designed to empower LLMs to handle complex tasks requiring continuous refinement and deep reasoning capabilities, all without necessitating any modifications or updates to the underlying model architecture.

By harnessing the Monte Carlo Tree Search (MCTS) algorithm, THOUGHTSCULPT enables LLMs to effectively explore vast search spaces while managing computational resource costs efficiently. Moreover, THOUGHTSCULPT facilitates a seamless self-revision process, allowing LLMs to iteratively refine and improve their outputs without the need for extensive prompt engineering. Through our experiments, we illustrate THOUGHTSCULPT's potential across diverse tasks, highlighting its versatility and broad applicability. The results underscore THOUGHTSCULPT's capacity to enhance LLM performance in challenges requiring continuous thought iteration, such as open-ended generation, multi-step reasoning, and creative ideation.

6 Ethics Statement

We affirm that all datasets utilized in our experiments have been appropriately sourced and cited, adhering to principles of academic integrity and proper attribution.

Our experiments primarily leverage GPT-3.5 and GPT-4 as the base LLMs. These models possess remarkable capabilities in generating human-like text based on prompts. However, we acknowledge the ethical concerns surrounding their potential misuse for spreading misinformation, generating harmful content, or impersonating individuals. We recognize the imperative for ethical considerations to include robust mechanisms aimed at preventing misuse and fostering responsible use of these models.

The purpose of THOUGHTSCULPT is to enhance the reasoning and complex problem-solving capabilities of Language Models (LMs). However, it is essential to acknowledge that THOUGHTSCULPT does not inherently include mechanisms to prevent LMs from generating harmful content. Therefore, we strongly advise anyone utilizing our model to exercise caution and be mindful of the potential for misuse. Users must take proactive measures to mitigate the risk of harmful content generation by implementing effective safeguards and appropriate controls.

7 Reproducibility

In our experiments, we aim for transparency and reproducibility by utilizing publicly accessible datasets. Furthermore, for the content evaluator utilized in the story outline improvement task, we employed Flan-T5, an open-source model. To facilitate reproducibility, our codebase will also be made available for reference and validation upon publication. However, as we access GPT-3.5 and GPT-4 through the OpenAI API, we acknowledge that reproducibility may be affected subject to OpenAI changing their API.

References

Anthropic, 2024. URL `https://www.anthropic.com/news/claude-3-family`.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. Graph of thoughts: Solving elaborate problems with large language models, 2024.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug, 2023.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

Ahmed Elgohary, Christopher Meek, Matthew Richardson, Adam Fourney, Gonzalo Ramos, and Ahmed Hassan Awadallah. Nl-edit: Correcting semantic parse errors through natural language interaction, 2021.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents, 2022a.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models, 2022b.

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks, 2023.

Hung Le, Yue Wang, Akhilesh Deepak Gotmare, Silvio Savarese, and Steven C. H. Hoi. Coderl: Mastering code generation through pretrained models and deep reinforcement learning, 2022.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In Trevor Cohn, Yulan He, and Yang Liu (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1823–1840, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.165. URL https://aclanthology.org/2020.findings-emnlp.165.

Kevin Lin, Kyle Lo, Joseph E. Gonzalez, and Dan Klein. Decomposing complex queries for tip-of-the-tongue retrieval, 2023.

Jiacheng Liu, Skyler Hallinan, Ximing Lu, Pengfei He, Sean Welleck, Hannaneh Hajishirzi, and Yejin Choi. Rainier: Reinforced knowledge introspector for commonsense question answering, 2022.

Ximing Lu, Sean Welleck, Jack Hessel, Liwei Jiang, Lianhui Qin, Peter West, Prithviraj Ammanabrolu, and Yejin Choi. Quark: Controllable text generation with reinforced unlearning, 2022.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative Refinement with Self-Feedback, May 2023. URL http://arxiv.org/abs/2303.17651. arXiv:2303.17651 [cs].

Piotr Mirowski, Kory W. Mathewson, Jaylen Pittman, and Richard Evans. Co-writing screenplays and theatre scripts with language models: An evaluation by industry professionals, 2022.

OpenAI. Gpt-4 technical report, 2024.

Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. REFINER: Reasoning Feedback on Intermediate Representations, February 2024. URL http://arxiv.org/abs/2304.01904. arXiv:2304.01904 [cs].

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning, October 2023. URL http://arxiv.org/abs/2303.11366. arXiv:2303.11366 [cs].

Niket Tandon, Aman Madaan, Peter Clark, and Yiming Yang. Learning to repair: Repairing model output errors after deployment using a dynamic memory of feedback, 2022.

Yufei Tian and Nanyun Peng. Zero-shot sonnet generation with discourse-level planning and aesthetics features. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3587–3597, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.262. URL https://aclanthology.org/2022.naacl-main.262.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023b.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023a.

Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents, 2023b.

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, 2023.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. Generating sequences by learning to self-correct, 2022.

Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, Xu Zhao, Min-Yen Kan, Junxian He, and Qizhe Xie. Self-evaluation guided beam search for reasoning, 2023.

Kevin Yang, Yuandong Tian, Nanyun Peng, and Dan Klein. Re3: Generating Longer Stories With Recursive Reprompting and Revision, October 2022. URL http://arxiv.org/abs/2210.06774. arXiv:2210.06774 [cs].

Kevin Yang, Dan Klein, Nanyun Peng, and Yuandong Tian. DOC: Improving Long Story Coherence With Detailed Outline Control, June 2023. URL http://arxiv.org/abs/2212.10077. arXiv:2212.10077 [cs].

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of Thoughts: Deliberate Problem Solving with Large Language Models, May 2023a. URL http://arxiv.org/abs/2305.10601. arXiv:2305.10601 [cs].

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023b.

A Prompts

Generally, THOUGHTSCULPT requires only three prompts: TASK_DESCRIPTION, NEW_CANDIDATE, and EVALUATE_CURRENT.

1. TASK_DESCRIPTION is the general instruction for the specific task. It will be placed in front of rest of the prompts.
2. NEW_CANDIDATE is the prompt to generate new candidates based on the evaluation feedback and the current solution.
3. EVALUATE_CURRENT instructs the language model to evaluate the current solution. The prompt can be tailored to ask for itemized evaluation, holistic evaluation, or both.

A.1 Task 1 Story Outline Improvement

```
TASK_DESCRIPTION = """\
# Task Description
You are a popular novel writer. You are now making an interesting outline
                                     for the story. You know how to
                                    engage with the readers by not
                                    limited to introducing interesting
                                    characters and unexpected twist.
You also know how to make the story outline coherent and consistent.
"""

NEW_CANDIDATE = TASK_DESCRIPTION + """\
# Original Outline
{outline}

# Feedback
{feedback}

Based on the feedback and the task description, can you make a better
                                    story outline by replacing the items
                                     suggested by the feedback?

Write the outline in this format just like the original outline from [1]
                                    to [{num}]:
[1] ...
[2] ...
...

# Your response:
"""

EVALUATE_CURRENT = TASK_DESCRIPTION + """\
# Original Outline
{outline}

Do you think that this outline is good enough?
Write a score from 1 to 100 where 100 means the outline is perfect based
                                    on the task description, and provide
                                     an explanation on strengths and
                                    weaknesses. Please be specific.
# Write in this format:
[score: 1-100] [reason] xxx (50 words max)

# Example:
[score: 50] [reason] the current outline is too predictable

# Your response:
"""
```

A.2 Task 2 Mini-Crossword Solving

```
TASK_DESCRIPTION = """\
Task Description:
Let's play a 5 x 5 mini crossword, where each word should have exactly 5
                                    letters. Your goal is to fill in the
                                     crossword with words based on the
                                    hints provided.
"""

NEW_CANDIDATE = TASK_DESCRIPTION + """\
#Current board:
{obs}

#Strategy:
{feedback}

Given the current status of the board and the strategy, list all possible
                                     answers for unfilled or changed
                                    words, and your confidence levels (
                                    certain/high/medium/low), using the
                                    format like this:
Use "certain" cautiously and only when you are 100% sure this is the
                                    correct word. You can list more then
                                     one possible answer for each word.

h1. [hint: _____] xxxxx (medium)
h2. [hint: _____] xxxxx (certain)
...
v1. [hint: _____] xxxxx (high)
...

Write your response in the format:
h1. [A financial loss; a negative profit; to remove bits from: D_B__]
                                    DEBTS (low)
h2. [Fatuous; empty headed: _____] INANE (high)
...
v1. [A dice player; something that cuts into small cubes: _____] DICER (
                                    high)
v5. [An Indian tent: _____] TEPEE (medium)

Each line can only have one candidate answer.
#Your response:
"""

EVALUATE_CURRENT = TASK_DESCRIPTION + """\
# Current board:
{obs}
Evaluate the current board and provide a strategy on how to continue to
                                    fill in the blank or correct
                                    potential mistakes.
Write your response in the format:
v1. [reasoning and potential answers]
v2. [reasoning and potential answers]
...
h1. [reasoning and potential answers]
...
# Example:
v2. [Current answer: tough; since  the filled in h1. is debit; e is
                                    conflicted with t, we could consider
                                     other options such as ENURE]
v3. [Current answer: ??? CUTUP could be a potential answer]
# Your response:
"""
```

A.3 Task 3 Constrained Generation

```
TASK_DESCRIPTION = """\
# Instruction Given several concepts (i.e., nouns or verbs), write a
                                    short and simple sentence that
                                    contains *all* the required words.
                                    The sentence should describe a
                                    common scene in daily life, and the
                                    concepts should be used in a natural
                                     way.
# # Examples
# ## Example 1 - Concepts: "dog, frisbee, catch, throw" - Sentence: The
                                    dog catches the frisbee when the boy
                                     throws it into the air.
# ## Example 2 - Concepts: "apple, place, tree, pick" - Sentence: A girl
                                    picks some apples from a tree and
                                    places them into her basket.
"""
INSTRUCTION = """\
Your Task - Concepts: {concepts}
"""
NEW_CANDIDATE = TASK_DESCRIPTION + """\
Instruction:
{instruct}

Here is a proposed sentence.
{solution}

Here is the feedback of outline item.
{feedback}

Based on the feedback, can you make a revised solution?
# Sentence:
"""
EVALUATE_CURRENT = TASK_DESCRIPTION + """\
Instruction:
{instruct}

Here is a proposed sentence.
{solution}

Do you think that the proposed sentence is good enough? Write "no need to
                                     improve" if you think 1) the
                                    sentence covers all the concepts
                                    listed in the instruction; and 2)
                                    the sentence describes a common
                                    scene in daily life.

Otherwise, write "still need to improve" and provide a reason.

# Write in this format:
[No need to improve/still need to improve] [reason] xxx (50 words max)

# Example 1:
[still need to improve] the sentence misses the concept "dog", "ladder",
                                    and "drum".
# Example 2:
[still need to improve] the cat does not fly.

# Your response:
"""
```

B Computation Efficiency

Table 4, Table 5, and Table 6 show the estimated number of input/output tokens usage and the cost of completing one case. THOUGHTSCULPT with DFS has a comparable cost to ToT while THOUGHTSCULPT with MCTS requires a greater computation since it has an additional decision simulation process.

	Input/Output Tokens	Cost per case
ToT	10.1k/4.9k	$0.248
THOUGHTSCULPT with DFS	11.3k/4.6k	$0.251
THOUGHTSCULPT with MCTS	25.0k/9.9k	$0.547

Table 4: Token use and estimated cost for Story Outline Improvement (Base LLM: gpt-4-0125-preview)

	Input/Output Tokens	Cost per case
ToT	64.5k/8.9k	$0.912
THOUGHTSCULPT with DFS	41.6k/7.1k	$0.629
THOUGHTSCULPT with MCTS	100.2k/16.3k	$1.491

Table 5: Token use and estimated cost for Mini-Crossword (Base LLM: gpt-4-0125-preview)

	Input/Output Tokens	Cost per case
ToT	7.1k/1.1k	$0.104
THOUGHTSCULPT with DFS	7.0k/0.7k	$0.091
THOUGHTSCULPT with MCTS	15.7k/2.0k	$0.217

Table 6: Token use and estimated cost for Constrained Generation (Base LLM: gpt-4-0125-preview)

C Alternative Search Algorithm

Algorithm 1 THOUGHTSCULPT with MCTS

Input: Initial node x_0
Output: Output node x^*
Initialize empty search tree T
for $j \leftarrow 1$ to $d_{rollout}$ **do**
  Select a leaf node x using the tree policy UCB1 Eqn 4
  Expand node x by generating a set of children nodes X_{child}
  node $x \leftarrow$ uniformly_sampled(X_{child})
  for $k \leftarrow 1$ to $d_{simulation}$ **do**
    node $x \leftarrow$ generate_single_child(x)
  end for
  Evaluate reward $v(x)$
  Propagate the reward v and number of explorations n back to x_0
end for
Choose the best node x^* with the highest reward v
return x^*

Algorithm 2 THOUGHTSCULPT with DFS

Input: Initial node x, Depth d
Output: Goal node x^*
$x \leftarrow x_0$
if $d = 0$ **then**
  return x
end if
Expand node x by generating a set of children nodes X_{child}
for $k \leftarrow 1$ to $max\_candidates$ **do**
  Evaluate reward $v(X_{child}[k])$
end for
Choose the node x^* with the highest reward v in X_{child}
$DFS(x^*, d-1)$

D Output Examples

D.1 Story Outline Improvement

The examples below show the final outputs of each reasoning algorithm in the story outline improvement task. In general, THOUGHTSCULPT with MCTS displays a more complex story with a greater tendency to introduce twists and characters. The final output of THOUGHTSCULPT with MCTS may be greatly modified. Self Refine is less prone to change from the initial outline. ToT and THOUGHTSCULPT with DFS are in the middle and would like to add interesting details to the initial outlines.

D.1.1 Example 1

Initial Outline:

1. Jack arrives in Maine and starts attending the same high school as Leo.
2. Jack begins observing Leo's daily routine and interests to find common ground.
3. Jack strategically plans his first encounter with Leo to establish a connection.
4. Jack successfully befriends Leo and starts spending more time together.
5. Jack and Leo spend more time together, deepening their bond.
6. Jack and Leo's attraction becomes undeniable, leading to their first kiss.
7. Jack and Leo navigate the challenges of their secret relationship.
8. Jack and Leo's love for each other grows stronger, despite the risks involved.
9. The Center orders Jack to cut ties with Leo and leave Maine immediately.
10. Jack struggles with the decision to leave Leo and return to the Center.
11. Leo's father becomes a target of the Center's operation, putting Leo in danger.
12. Jack risks his own safety to protect Leo and his father, defying the Center's orders.

Self Refine:

1. Jack arrives in Maine and starts attending the same high school as Leo.
2. Jack begins observing Leo's daily routine and interests to find common ground.
3. Jack strategically plans his first encounter with Leo to establish a connection.
4. Jack successfully befriends Leo and starts spending more time together.
5. Jack and Leo spend more time together, deepening their bond.
6. Jack and Leo's attraction becomes undeniable, leading to their first kiss.
7. Jack and Leo navigate the challenges of their secret relationship.
8. Jack and Leo's love for each other grows stronger, despite the risks involved.
9. The Center orders Jack to cut ties with Leo and leave Maine immediately.
10. Jack struggles with the decision to leave Leo and return to the Center.
11. A mysterious organization threatens to expose Jack and Leo's relationship, putting their lives in danger.
12. Jack must make a decision to protect Leo and himself, risking everything to defy both the Center and the new threat.

ToT:

1. Jack arrives in Maine and starts attending the same high school as Leo.
2. Jack is immediately drawn to Leo's mysterious and aloof demeanor, sparking his curiosity.

3. Jack discovers Leo's passion for art and secretly starts leaving anonymous gifts for him.
4. Leo becomes intrigued by the gifts and begins to search for the mysterious admirer.
5. Jack and Leo finally meet at a school art exhibition, where Leo realizes Jack's connection to the gifts.
6. Their initial awkward encounter slowly blossoms into a genuine friendship as they bond over their shared love for art.
7. Jack and Leo's friendship is tested when a scandal rocks their school, threatening to tear them apart.
8. Despite the challenges, Jack and Leo's bond deepens as they confide in each other and support one another.
9. The Center, a secretive organization Jack is a part of, reveals their true intentions and orders Jack to complete a dangerous mission that puts Leo at risk.
10. Jack is torn between his loyalty to the Center and his love for Leo, struggling to find a way to protect both.
11. Leo's life is put in grave danger when he uncovers a dark secret that implicates the Center in a series of crimes.
12. Jack makes the ultimate sacrifice to save Leo and expose the Center's corrupt actions, putting his own life on the line for the person he loves.

THOUGHTSCULPT with DFS

1. Jack arrives in Maine and starts attending the same high school as Leo.
2. Jack is immediately drawn to Leo, but soon realizes that Leo is hiding a dark secret.
3. Instead of confronting Leo right away, Jack decides to secretly follow him and discovers that Leo is actually working undercover to bring down the dangerous organization.
4. Jack is conflicted between his growing feelings for Leo and the danger of being involved in Leo's mission.
5. Jack offers to help Leo in his mission, leading to a deepening of their bond as they face increasing risks together.
6. Their forbidden romance blooms amidst the chaos, complicating their mission and putting them both in jeopardy.
7. The organization catches wind of Jack and Leo's alliance, putting both of their lives at risk.
8. Jack and Leo must make difficult choices to protect each other and navigate the treacherous waters of their relationship.
9. The Center threatens to harm Leo's family if they don't stop their investigation.
10. Jack and Leo are forced to go underground and make a dangerous deal to protect their loved ones.
11. The stakes are raised when Leo's father is kidnapped, pushing Jack and Leo to their limits in a race against time to save him.
12. Jack and Leo face off against the leaders of the organization in a thrilling final showdown, risking everything to bring them down and secure a future together.

THOUGHTSCULPT with MCTS:

1. Jack arrives in Maine and starts attending the same high school as Leo.
2. Jack notices a mysterious new student, Alex, who seems to be watching Leo closely.
3. Jack becomes curious about Alex's intentions and starts investigating their background.

4. Jack discovers that Alex is actually a spy from a rival organization called the Center targeting Leo.
5. Jack confronts Alex and warns Leo about the danger, leading to a confrontation between the three.
6. Jack, Leo, and Alex form an unlikely alliance to uncover the truth behind the Center's motives.
7. Jack and Leo's bond is tested as they navigate the complexities of trust and betrayal within their group.
8. The rival organization escalates their actions, putting Jack, Leo, and Alex in grave danger.
9. Leo sacrifices himself to protect Jack and defeat the Center, shocking everyone.
10. The truth behind the Center's motives is revealed, leading to a deeper understanding of the sacrifices made.
11. Jack and Alex must work together to honor Leo's sacrifice and take down the Center.
12. In a shocking revelation, it is uncovered that Leo's twin brother, who was thought to be dead, is actually the mastermind behind the Center, creating a complex dynamic between the characters.

D.1.2 Example 2

Initial outline:

1. The Health Department wants to close the Boys Club due to infestation of rats and cockroaches.
2. Closing the Boys Club means the kids will be left without a place to go.
3. The kids are worried about being out on the street again.
4. Buddy suggests organizing a bake sale to raise funds.
5. Johnny proposes hosting a talent show as another fundraising idea.
6. Rich suggests reaching out to local businesses for donations or sponsorships.
7. Cool suggests organizing a community car wash to raise money.
8. Buddy Williams suggests asking the local pet store owner if he knows anything about the rich cat.
9. The kids visit the pet store and ask the owner about the rich cat.
10. The pet store owner tells the kids about a rumor he heard of a rich cat living in a mansion nearby.
11. The kids decide to investigate the mansion and see if they can find the rich cat.
12. The townspeople share their findings and clues with the kids, who compile the information and continue the search.
13. The kids and townspeople follow leads and investigate potential sightings of the rich cat.
14. The search intensifies as more people join in, creating a sense of community and determination to find the rich cat.
15. After an extensive search, the kids and townspeople finally locate the rich cat, who generously donates the money needed to save the Boys Club.

Self Refine:

1. The Health Department wants to close the Boys Club due to infestation of rats and cockroaches.
2. Closing the Boys Club means the kids will be left without a place to go.
3. The kids are worried about being out on the street again.

4. Buddy suggests organizing a bake sale to raise funds.
5. Johnny proposes hosting a talent show as another fundraising idea.
6. Rich suggests reaching out to local businesses for donations or sponsorships.
7. Cool suggests organizing a community car wash to raise money.
8. Buddy Williams suggests asking the local pet store owner if he knows anything about the rich cat.
9. The kids visit the pet store and ask the owner about the rich cat.
10. The pet store owner tells the kids about a rumor he heard of a rich cat living in a mansion nearby.
11. The kids decide to investigate the mansion and see if they can find the rich cat.
12. While searching the mansion, the kids stumble upon a hidden room filled with mysterious artifacts and clues.
13. The artifacts lead them to a map that hints at a hidden treasure in the town.
14. The kids and townspeople embark on a thrilling treasure hunt, following the map's clues and facing challenges along the way.
15. After overcoming obstacles and working together, the kids and townspeople unearth the hidden treasure, which turns out to be the rich cat's fortune that saves the Boys Club.

ToT:

1. The Health Department wants to close the Boys Club due to infestation of rats and cockroaches.
2. Closing the Boys Club means the kids will be left without a place to go.
3. The kids are worried about being out on the street again.
4. Buddy suggests organizing a bake sale to raise funds.
5. Johnny proposes hosting a talent show as another fundraising idea.
6. Rich suggests reaching out to local businesses for donations or sponsorships.
7. Cool suggests organizing a community car wash to raise money.
8. Buddy Williams suggests asking the local pet store owner if he knows anything about the rich cat.
9. The kids visit the pet store and ask the owner about the rich cat.
10. The pet store owner tells the kids about a rumor he heard of a rich cat living in a mansion nearby.
11. The kids decide to investigate the mansion and see if they can find the rich cat.
12. While investigating the mansion, the kids stumble upon a hidden room with a mysterious map that leads to a hidden treasure.
13. The townspeople share their findings and clues with the kids, who now have a new mission to find the hidden treasure.
14. The search for the hidden treasure leads the kids and townspeople to uncover secrets about the town's history and a long-lost treasure that could save the Boys Club.
15. After a thrilling adventure and overcoming obstacles, the kids and townspeople finally find the hidden treasure, which not only saves the Boys Club but also brings the community closer together.

ThoughtSculpt with DFS

1. The Health Department discovers that the Boys Club is in danger of losing its funding due to budget cuts.

2. The kids are devastated as they learn that the Boys Club may have to shut down if they can't come up with a solution.
3. Buddy suggests organizing a scavenger hunt around town to raise funds and awareness for the Boys Club.
4. Johnny proposes creating a social media campaign to rally support and donations for the Boys Club.
5. Rich suggests hosting a charity concert with local musicians and performers to raise money.
6. Cool suggests organizing a charity auction with items donated by local businesses and community members.
7. The kids brainstorm and come up with a plan to combine all the fundraising ideas into one big event to save the Boys Club.
8. Buddy Williams suggests asking the local historian if he knows of any hidden treasures or artifacts that could help raise funds.
9. The kids visit the local historian and learn about a legendary treasure rumored to be hidden somewhere in town.
10. The historian gives the kids a map and clues to help them on their quest to find the treasure.
11. The kids embark on an exciting adventure to follow the map and solve the clues to locate the hidden treasure.
12. Along the way, they face challenges and obstacles that test their friendship and determination.
13. The townspeople join in the treasure hunt, adding to the excitement and suspense of the search.
14. After overcoming obstacles and following the clues, the kids and townspeople finally discover the hidden treasure, which turns out to be the key to saving the Boys Club.
15. The treasure not only provides the funds needed to keep the Boys Club open but also brings the community together in a heartwarming and unforgettable way.

THOUGHTSCULPT with MCTS

1. The Health Department wants to close the Boys Club due to infestation of rats and cockroaches.
2. Closing the Boys Club means the kids will be left without a place to go.
3. The kids are worried about being out on the street again.
4. Buddy suggests organizing a scavenger hunt around town to raise funds.
5. Johnny proposes creating a community mural project as another fundraising idea.
6. Luna, a mysterious new girl in town, suggests hosting a talent show to raise money, revealing her own hidden talents in the process.
7. Rich suggests organizing a neighborhood cleanup event to raise money.
8. Cool suggests hosting a movie night in the park to raise funds.
9. Buddy Williams suggests asking the local historian if he knows anything about a hidden treasure in the town.
10. The kids visit the local historian and ask about the hidden treasure.
11. The historian tells the kids about a legend of a lost pirate treasure buried somewhere in the town.
12. Luna shares a personal connection to the treasure hunt, sparking curiosity and excitement among the group.
13. The kids face challenges and puzzles during their treasure hunt that test their problem-solving skills and teamwork.

14. In a surprising twist, the treasure turns out to be a time capsule filled with memories and mementos from the town's past.
15. The kids realize that the true treasure was the journey itself and the bonds of friendship they formed along the way, teaching them the value of community and unity. Luna's involvement adds depth to the story and enhances the theme of unity and friendship.

D.2 Constrained Generation

The examples below show the final outputs of different reasoning strategies in Constrained Generation.

D.2.1 Example 1

Concepts: drum, clip, window, cover, graffiti, wave, vomit, hurdle, dye, sidewalk, animal, slice, swing, pottery, pineapple, fix, decorate, tire, tag, bicycle, watermelon, sign, mask, plant

Refine Children decorated pottery with vibrant dye near the graffiti-covered wall, while a masked animal rode a bicycle past a sign, waving as they passed by a tire swing hanging from a tree and a clip holding a sign on the sidewalk, avoiding a hurdle, slicing a pineapple and a watermelon to plant in the garden.

ToT A child on a swing by the sidewalk waved at a bicyclist with a tire fixed and decorated with graffiti.

THOUGHTSCULPT with DFS The children decorated the pottery with vibrant dye, while the adults fixed the tire on the bicycle near the sidewalk, as a group of friends played the drum and waved at passersby from the swing, with a clip holding a sign on the window, a slice of watermelon in hand, and a pineapple plant in the background.

THOUGHTSCULPT with MCTS A person walks down the sidewalk past a graffiti-covered window, a pottery clip lies on the ground, a bicycle rider swerves to avoid a tire, and a sign is being fixed and decorated with dye next to a pineapple stand, as a drum beats in the background, a wave crashes on the shore, an animal scurries across the path, a slice of watermelon is enjoyed under the shade of a swing, a mask-wearing individual tags a plant with a new growth, and a hurdle is cleared by a runner as they vomit from exhaustion.

D.2.2 Example 2

Concepts: plant, word, remove, toddler, hamburger, potato, set, line, place, control, eat, banner, singe, hurdle, bed, bar, alley, stove, top, air, cowboy, sharpen, pair, beat, arena, wave, singer, jacket

Refine The toddler removes the banner from the line and places it on the top of the bar to singe while the cowboy sharpens a pair of knives to beat the hurdle in the arena.

ToT The toddler eats a hamburger and potato at the place where the plant is growing, while the singer on stage waves under the banner to the cheering crowd, and the cowboy sharpens his pair of beat-up boots backstage.

THOUGHTSCULPT with DFS The toddler eagerly eats a hamburger while the cowboy sharpens his pair of scissors at the plant nursery.

THOUGHTSCULPT with MCTS The toddler removes a plant from the line, places it in the bed, eats a hamburger with a side of potato, then sets the table for a cowboy sharpening his pair of knives at the top of the bar, while a singer controls the banner in the arena, singing as the wave of music beats through the alley where a jacketed cowboy hurdles over a singe stove.