Breaking the Self-Evaluation Barrier: Reinforced Neuro-Symbolic Planning with Large Language Models

Jie-Jing Shao[1*], **Hong-Jie You**[1,2*], **Guohao Cai**[3], **Quanyu Dai**[3], **Zhenhua Dong**[3], **Lan-Zhe Guo**[1,4†]
[1]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
[2]School of Artificial Intelligence, Nanjing University, Nanjing, China
[3]Huawei Noah's ArkLab, Shenzhen, China
[4]School of Intelligence Science and Technology, Nanjing University, Nanjing, China
{shaojj, youhj, guolz}@lamda.nju.edu.cn, {caiguohao1, daiquanyu, dongzhenhua}@huawei.com

Abstract

Large Language Models (LLMs) have demonstrated remarkable capabilities in language understanding and commonsense reasoning, yet they often struggle with constraint satisfaction in planning problems. Previous studies relying on test-time improvement with self-evaluation fail to address this limitation effectively. In this work, we identify this critical gap and propose a novel neuro-symbolic framework, Reinforced Neuro-Symbolic Planning (RNSP), that enhances LLM-powered planning by incorporating a symbolic verifier. The verifier provides explicit feedback on constraint satisfaction, enabling iterative refinement of the state evaluation. Specifically, we utilize the outcome feedback from each logical goal to update the process value along planning paths through a reinforcement value function maximization objective. We further employ T-norms to aggregate the satisfaction levels of multiple constraints, which provided more effective guidance for the test-time search. Our framework bridges the strengths of neural and symbolic methods, leveraging the generative power of LLMs while ensuring rigorous adherence to constraints through symbolic verification. Extensive experiments demonstrate that our approach significantly improves planning accuracy and constraint satisfaction across various domains, outperforming traditional self-evaluation methods. It highlights the potential of hybrid neuro-symbolic systems to address complex constrained planning tasks.

1 Introduction

Large Language Models (LLMs) have made a profound impact on the AI community [OpenAI, 2022; Bubeck *et al.*, 2023; Chang *et al.*, 2024]. Through pre-training on web-scale corpora, they have demonstrated impressive natural language understanding and reasoning capabilities, driving advancements across various domains, including machine translation [Vilar *et al.*, 2023; Min *et al.*, 2024], code generation [Liu *et al.*, 2024b; Liang *et al.*, 2024], and content generation [Li *et al.*, 2024; Keskar *et al.*, 2019], among others. Despite their impressive performance, these models often struggle with planning tasks that require specific goal understanding and process constraints [Xie *et al.*, 2024; Kambhampati *et al.*, 2024]. To address this issue, test-time search methods have been proposed and attracted much attention [Yao *et al.*, 2023a; Chen *et al.*, 2024; Snell *et al.*, 2024].

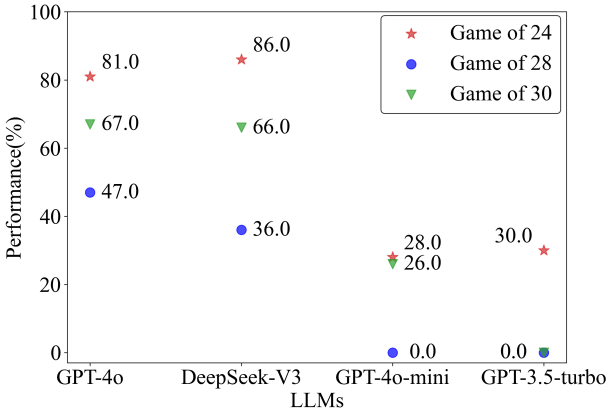

Figure 1: Performance of LLM planning with self-evaluation in the Game of 24, Game of 28, and Game of 30. These results underscore the fragility of LLM self-evaluation, as a simple change in the target (goal: 24 → 28, 30) leads to a significant decline in performance.

Previous test-time search methods guide the problem-solving process of planning through a search mechanism. They typically utilize the self-evaluation mechanisms to prioritize partial solutions during the search process, aiming to enhance search efficiency. For example, Xie *et al.* [2023] integrate self-evaluation guidance with stochastic beam search to enhance the reasoning capabilities on arithmetic reasoning and commonsense reasoning. Hao *et al.* [2023] repurpose the LLM as a world model, leveraging self-evaluation to evaluate the quality of each Monte Carlo Tree Search reasoning step. It allows the model to iteratively refine its reasoning paths and ultimately find high-reward solutions efficiently. Yao *et al.* [2023a] present the Tree of Thought (ToT), which reformulates the general problem-solving as the search process

and utilizes self-evaluation to guide the large language models exploring multiple reasoning paths. Shao *et al.* [2024b] present a domain-specific language to define travel requirement constraints, using it to control backtracking in the search process for travel plans.

Despite the positive results demonstrated by these studies in mathematical reasoning and commonsense reasoning, there are some research [Kambhampati *et al.*, 2024; Huang *et al.*, 2024] have indicated that self-evaluation is unreliable for planning tasks that require domain-specific knowledge. A classic example used to discuss the self-evaluation capability of large language models in long-term foresight assessment for planning tasks is the Game of 24 [Yao *et al.*, 2023a]. We extend this task to verify the self-evaluation abilities of LLMs in both the Game of 28 and the Game of 30. The results are shown in Figure 1. The results have shown that for the Game of 24, both GPT-4o and Deepseek-v3 achieved over an 80% success rate with ToT self-evaluation [Yao *et al.*, 2023a], indicating their capability to find computational paths to derive the number 24 from the given numbers. This success aligns with previous self-evaluation efforts on this task, demonstrating the effectiveness of self-evaluation in computation-based planning tasks. However, when we introduced a simple variation in the task objective, requiring the goal of the number 28 or 30, the planning performance of LLMs significantly deteriorated. Both GPT-4o and Deepseek-v3 dropped to around 60% success rate, while GPT-4o-mini and GPT-3.5-turbo performed even a 0% success rate. This stark decline clearly highlights the fragility of LLM's self-evaluation abilities. One plausible reason for this phenomenon is that the abundant internet data regarding the Game of 24 has enabled LLMs to develop self-evaluation capabilities specifically tailored to this task, rather than providing assessments through foresight in planning. Consequently, this limits the LLM's self-evaluation abilities, leading to a notable decline in performance on the Game of 28/30.

To address these limitations, we propose a novel neuro-symbolic framework, Reinforced Neuro-Symbolic Planning (RNSP), that integrates the generative strengths of LLMs with a symbolic verifier. This verifier provides explicit feedback on constraint satisfaction, enabling iterative refinement of the state evaluation. Specifically, we utilize the outcome feedback from each logical goal to update the process value along planning paths through a reinforcement value function maximization objective. We further employ T-norms to aggregate the satisfaction levels of multiple constraints, which provided more effective guidance for the test-time search. Our approach bridges the gap between neural flexibility and symbolic rigor, ensuring robust adherence to logical and procedural constraints. Extensive experiments on various domains demonstrate the effectiveness of our method, significantly improving planning accuracy and constraint satisfaction compared to traditional self-evaluation techniques. Overall, our framework highlights the potential of hybrid neuro-symbolic systems for complex planning tasks. By leveraging symbolic verification to enhance the foresight planning capabilities of LLMs, our approach not only achieves state-of-the-art results but also establishes a promising paradigm for addressing constraint satisfaction in LLM planning.

2 Related Work

LLM Planning. Large Language Models (LLMs) have recently shown remarkable capabilities in natural language understanding and reasoning [Brown *et al.*, 2020; Wei *et al.*, 2022; Yao *et al.*, 2023b]. Based on these advancements, exploring LLMs for planning tasks has gained widespread attention. For example, Huang *et al.* [2022] investigate the potential of large language models to generate executable action plans for embodied agents without any additional training. Ichter *et al.* [2022] ground language models in robotic affordances by combining the semantic commonsense from pre-trained models with value functions that assess the feasibility of actions in the real world, enabling robots to execute complex tasks more effectively. Compared to the direct planning generation, Wei *et al.* [2022] find encouraging LLMs to perform step-by-step reasoning could improve the planning capabilities, and present the chain-of-thought (CoT) prompting appends reasoning steps before the answer for complex problems. Zhao *et al.* [2023] evaluate the role of LLMs as a commonsense world model in the Monte Carlo Tree Search (MCTS) process for large-scale task planning. By integrating an LLM's world model, which provides a commonsense prior belief, MCTS can perform more effective reasoning. This integration allows the system to navigate complex decision spaces more efficiently, as the LLM's heuristic policy guides the search process, significantly improving search efficiency. Yao *et al.* [2023a] presents a novel framework called Tree of Thoughts (ToT) that enables large language models (LLMs) to perform deliberate problem-solving by exploring multiple reasoning paths. ToT introduces pre-defined symbolic workflows and guides the search process with the LLM self-evaluation.

Neuro-Symbolic Planning. Pure-learning-based methods, like those using deep neural networks, often excel at recognizing complex patterns from data but struggle with satisfying strict logical or procedural constraints in planning tasks. In contrast, symbolic planners are highly effective at long-horizon planning, adeptly managing sequences of actions to meet specific goals while strictly adhering to logical and procedural constraints. Neuro-symbolic planning offers a compelling integration of these approaches, combining the pattern recognition and adaptability of data-driven learning with the rigorous constraint satisfaction of symbolic reasoning [Yang *et al.*, 2024; Jia *et al.*, 2025] to facilitate effective long-horizon planning [Xu *et al.*, 2019; Silver *et al.*, 2022; Shao *et al.*, 2024a]. In the era of LLMs, Pan *et al.* [2023] presents the LogicLM integrates LLMs with separate symbolic solvers for various logical reasoning tasks. They first utilize LLMs to translate a natural language problem into a symbolic formulation. Afterward, a deterministic symbolic solver performs inference on the formulated problem to ensure the correctness of the results. Wang *et al.* [2023] utilizes LLMs to decompose the desired goal and generate symbolic workflow to help imitation learning for robotic manipulation.

LLM Self-Evaluation. Recent studies have explored the potential and limitations of LLM self-evaluation, aiming to enhance the models' ability to assess and refine their own outputs. For instance, Xie *et al.* [2023] integrate self-evaluation

Notation	Meaning
$\mathcal{S}$	State space
$\mathcal{A}$	Action space
$\mathcal{T}: \mathcal{S} \times \mathcal{A} \to \mathcal{S}$	Transition function
$\mathcal{S}_0$	The distribution of initial states
$\mathcal{P}$	Task-related predicates
$C = \{p_c\} \sim \mathcal{P}$	Progress constraints of planning
$G = \{p_g\} \sim \mathcal{P}$	Final goals of planning
$g(a\|s, \phi)$	Action generator with LLM ϕ
$f_s(\cdot\|p)$	Symbolic verifier function
$v(s\|p)$	Value estimation of predicate p
$f_v(s\|\mathcal{G})$	Aggregated value estimation for goal $\mathcal{G}$

Table 1: Summary of Notations

guidance with stochastic beam search to enhance the reasoning capabilities on arithmetic reasoning and commonsense reasoning. Hao *et al.* [2023] repurpose the LLM as a world model, leveraging self-evaluation to evaluate the quality of each Monte Carlo Tree Search reasoning step. It allows the model to iteratively refine its reasoning paths and ultimately find high-reward solutions efficiently. Huang *et al.* [2023] show that self-evaluation can improve selective generation in LLMs by reformulating open-ended tasks into token-level prediction tasks, thereby enhancing accuracy and content quality Yao *et al.* [2023a] reformulate the general problem-solving as the search process and utilizes self-evaluation to guide the large language models exploring multiple reasoning paths. Huang *et al.* [2024] demonstrate that LLMs often struggle to self-correct their responses without external feedback, and in some cases, their performance may even degrade after attempting self-correction. Kambhampati *et al.*[2024] identify that the inadequate self-evaluation capability of LLMs for domain-specific tasks is a crucial factor limiting their effectiveness in planning tasks.

3 The Proposed RNSP Method

3.1 Problem Formulation

In this paper, we focus on the constrained planning, which could be formally defined as $< \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{S}^0, \mathcal{P}, \mathcal{C}, \mathcal{G} >$. Here, $\mathcal{S}$ represents the state space, and $\mathcal{A}$ represents the action space. The transition between states and actions is governed by a deterministic workflow function $\mathcal{T}: \mathcal{S} \times \mathcal{A} \to \mathcal{S}$. $\mathcal{S}_0$ denotes the distribution of initial states. Additionally, $\mathcal{P}$ is a finite set of task-related predicate symbols. A ground atom p is a predicate given the state information $s \sim \mathcal{S}$. If a state s satisfies $s \models p$, it indicates that s semantically entails the interpretation of p. The $\mathcal{C} = \{p_c\}$ represents the constraints in the planning process. The set $G = \{p_g\}$ consists of ground atoms representing the final task's target. The task is to find an action sequence that generates a planning trajectory $\{s_0, a_0, s_1, a_1, \ldots, a_T, s_T\}$ satisfying $(s_t, a_t) \models \bar{p}, \forall t \in [T], \bar{p} \in C$ and final state s_T satisfying $s_T \models \bar{p}, \forall \bar{p} \in G \sim \mathcal{G}$.

Consider the game of 24, where the objective is to manipulate a given set of numbers to achieve the target number 24 using basic arithmetic operations (addition, subtraction, multiplication, or division). At each step, the current state consists of the remaining numbers available for manipulation. An action represents a transformation, which involves selecting two numbers and applying one of the four arithmetic operators to produce a new number. The goal is to explore a sequence of actions that transforms the initial set of numbers to derive the number 24. The progress constraints $\mathcal{C}$ stipulate that the numbers selected for each action must exist within the current state and conform to the established arithmetic syntax. This means that only valid numeric combinations and operator applications are permissible, ensuring that all transformations are mathematically sound. For example, given the initial state $s_0 = (4, 9, 10, 13)$, LLMs provide a transformation action $a_0 : 13 - 9 = 4$, which consists of three numbers $c_0^1 = 13, c_0^2 = 9, c_0^3 = 4, op_0 = -$. The process constraints require that c_0^1 and c_0^2 must be selected from the current state s_0 and that op_0 belongs to the predefined set of arithmetic operators. Furthermore, the resulting calculation $(c_0^1, op_0, c_0^2, c_0^3)$ must adhere to mathematical principles, ensuring that the operation is valid. The transition $T(s_0, a_0)$ modifies the state to $s_1 = (4, 4, 10)$ where the returned state s_1 includes the remaining $4, 10$ and the newly derived 4.

3.2 Planning Workflow with Symbolic Verifier

The search paradigm addresses limitations in current large language models by enabling the exploration of multiple planning paths. Nevertheless, they typically rely on the pre-trained process reward models [Lightman *et al.*, 2024; Snell *et al.*, 2024] or self-evaluation [Yao *et al.*, 2023a]. In this work, we introduce the symbolic verifiers to provide the process supervision and guide the search with constraints. Figure 2 provides an illustration of overall framework.

Beam Search. Beam search serves as a foundational mechanism for navigating the planning space. By utilizing a predefined LLM workflow, i.e., scheduling the LLMs to generate action with the transition function, we guide the exploration of multiple paths concurrently, maintaining a balance between exploration and exploitation. This approach allows the process to consider a breadth of states, retaining only the most promising candidates for further planning. We set a beam width B to control the complexity of the search. This structured search ensures that computational resources are efficiently used to explore feasible solutions.

Action Proposal. Given a state s_t in the current candidates, the LLMs ϕ are employed to generate K action candidates $[a_t^1, a_t^2, \cdots, a_t^K] \sim g(a_t|s_t, \phi)$. This set serves to expand the state nodes within a tree-searching process. Each action candidate a_t represents a potential planning path forward $(s_t, a_t, T(s_{t+1}|s_t, a_t))$, branching the search tree and allowing for a diverse exploration of possible solutions. The action proposal stage leverages the generative capabilities of LLMs to produce a wide array of viable next steps, which are essential for navigating complex problem spaces.

Progress Symbolic Verification Subsequently, a symbolic verifier $f_v(s, a, \mathcal{P})$ is applied to the state-action pairs to ensure adherence to process constraints $\mathcal{C}$. We extract concepts from the generated text using regular expression matching to compute the value of the predicates p_c within the constraints $\mathcal{C}$, thereby facilitating constraint verification. This

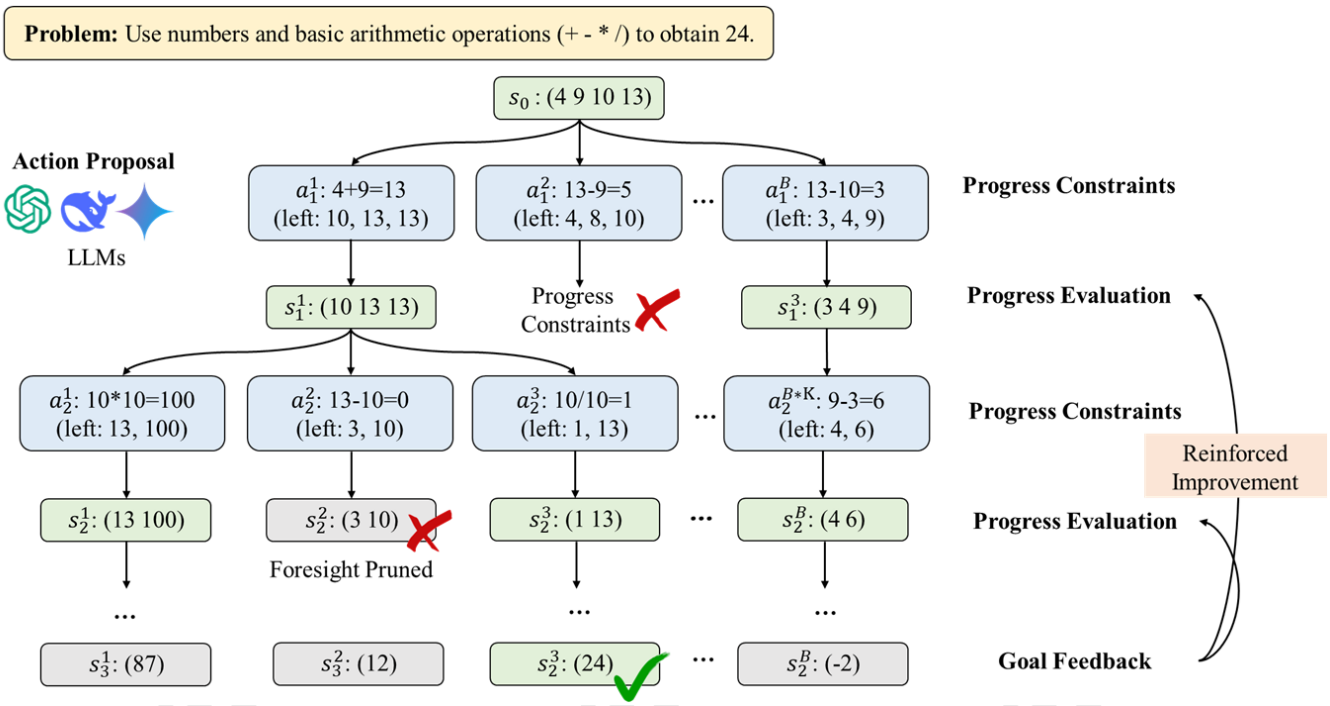

Figure 2: The illustration of proposed RNSP framework.

verifier acts as a gatekeeper, filtering out actions that do not conform to the defined logical or procedural requirements, thereby guaranteeing the legality of the resultant solutions. The integration of symbolic verification adds a layer of rigor to the planning process, enhancing reliability by ensuring that only valid paths are explored further. This synergy between the generative strengths of LLMs and the stringent checks of symbolic verification significantly bolsters the overall robustness and effectiveness of the planning workflow.

3.3 Reinforced Evaluation Improvement

As we discussed above, LLMs are weak at handling the domain-specific knowledge required for planning tasks. A promising solution is learning from the goal feedback of the planning task. In this section, we incorporate reinforced evaluation mechanisms that utilize principles of reinforcement learning to update process-wise evaluations along the search paths. Specifically, for each feedback predictive signal p in the terminal states, we employ a value function $v(s, a|p)$ to update the process value, ensuring that the evaluation adapts and improves over time. Feedback is quantified using fuzzy logic, which allows for a nuanced assessment of the partial solutions. We first define the path $(s'_0, s'_1, s'_2, \ldots, s'_L)$ as the sequence from the root node to the terminal node s'_L in the search tree. For the terminal state s'_L, we perform a symbolic evaluation of the target predicates in $\mathcal{G}$ to obtain the signal feedback for each $p_g \in \mathcal{G}$. This feedback is then propagated back to the preceding parent nodes $s'_{L-1}, s'_{L-2}, \ldots$ in the planning trajectory:

$$v(s'_l|p) = \max(v(s'_l|p), f_s(s'_L, p)), \forall l \in [0, L] \quad (1)$$

Algorithm 1 The proposed RNSP

Require: The LLM ϕ, n initial states S_0, constraints verifier $V_C(\cdot|\mathcal{P})$, progress evaluator $f_v(\cdot|\mathcal{G})$, beam width B, proposal width K, planning length L

for $i = 1, 2, \ldots, n$ **do**
  Sample problem $s_0 \sim S_0$
  for $l = 1, 2, \ldots, L$ **do**
    $C = \{s_0\}$.
    for each s in S_{l-1} **do**
      for $k = 1, 2, \ldots, K$ **do**
        $a_t \sim g(\cdot|s_t, \phi)$
        if $(s_l, a_l) \models \bar{p}, \forall \bar{p} \in \mathcal{C}$ **then**
          $C = C \cup T(\cdot|s_l, a_l)$.
        end if
      end for
    end for
    $S_{l+1} \leftarrow$ prune(C,B) with estimation Eq. 2.
  end for
  Update value estimation $v(s|p) \forall p \in \mathcal{G}$ via Eq. 1.
end for
return S_L

This is equivalent to the value update in reinforcement learning, where dynamic programming is used to transition state values under different actions.

$$V(s) = \max_{a \in \mathcal{A}, s' \sim T(s,a)} V(s')$$

To aggregate the multiple constraint assessments for $\mathcal{G}$, we incorporate the T-norms to the process value estimation. By

	Game of 24/28/30	Constrained Knapsack	Travel Planning
Input	4 numbers (4 9 10 13)	Blocks: (pink, 14),(blue, 5)... Required colors: pink, red	Query: Please help me plan a trip from St. Petersburg to Rockford spanning 3 days with a budget of $1700. Information: Flight from ..., Restaurants ...
State	13 - 10 = 3 (left: 3 4 9) 9 - 3 = 6 (left: 4 6)	Selected blocks'id: 30, 35, 22, 9, 16, 18, 51	Day: 1, Current city: from St. Petersburg to Rockford, Transportation: Flight Number: F3573659, Breakfast: Dunkin' Donuts, Rockford, Attraction: Anderson Japanese Gardens, Rockford,
Action	4 * 6 = 24 (left: 24)	Select block with id 37	Lunch: Moets Arabica, Rockford.
Steps	3	10	21-49

Table 2: Task overview.

applying the T-norm operations, we evaluate the conjunction of constraints as follows:

$$f_v(s|\mathcal{G}) = \min_{p \in \mathcal{G}} v(s|p) \tag{2}$$

This formula calculates the minimum satisfaction level across all constraints, ensuring that the lowest level of satisfaction dictates the overall assessment of a state-action pair's viability. This approach aligns with the physical interpretation of value estimation, which reflects foresight regarding the probability of successfully achieving the goal $\mathcal{G}$. It helps maintain a conservative and realistic appraisal of the planning paths.

Integrating these reinforcement objectives enhances the search by ensuring that the consideration of actions not only respects constraints but also continuously improves through learning from past evaluations. This makes the planning process more robust and adaptable, capable of responding to dynamic constraints and feedback efficiently. The overall method is summarized in the Algorithm 1.

4 Empirical Study

4.1 Experimental Setup

We evaluate our proposal on diverse tasks, including Game of 24, Game of 28, Game of 30, Constrained Knapsack, and Travel Planning. We benchmark a series of well-validated or cutting-edge methods as baselines. (1) Standard Input-Output prompting (**IO**): it generates a direct answer without additional reasoning steps, (2) Chain-of-Thought prompting (**CoT**) [Wei *et al.*, 2022]: it encourages the model to break down complex problems into simpler, step-by-step reasoning processes. (3) Tree-of-Thoughts (**ToT**) [Yao *et al.*, 2023a]: it explores multiple reasoning paths with symbolic workflow and performs self-evaluation to find optimal solutions. We compare these methods with our proposed approach, based on advanced LLMs, including DeepSeek-V3 [Liu *et al.*, 2024a], GPT-4o [OpenAI, 2024b], GPT-4o-mini [OpenAI, 2024a], and GPT-3.5-turbo [OpenAI, 2022].

4.2 Game of 24, 28 and 30

The Game of 24 is a well-known benchmark that tests the mathematical reasoning of large language models through the use of basic arithmetic operations, includes addition, subtraction, multiplication, and division. The premise of the Game of 24 is straightforward: given four numbers and these fundamental operations, the objective is to derive the target number 24. This game serves as an exemplary scenario for assessing the self-evaluation capabilities of LLMs, as it requires not only computational competence but also logical reasoning to identify valid sequences of operations.

To extend this concept, we generalize the framework to include different target numbers, resulting in variations such as the Game of 28, and the Game of 30. Each of these games presents unique challenges while maintaining the fundamental mechanics of the original Game of 24. By varying the target numbers, we can observe how effectively LLMs adapt their strategies and reasoning processes, providing insights into their planning and problem-solving abilities.

For the game of 24, we follow the [Yao *et al.*, 2023a] and collect the data from 4nums.com, a website hosting mathematical games, specifically selecting 1,362 games sorted by human solving time from easy to hard. The samples indexed 800-900 are utilized to train, and the samples indexed 901-1000 are utilized to test. For the Game of 28 and Game of 30, we randomly generate problems and use a ground-truth symbolic solver to identify and retain only those problems that have solutions. The scale of the training and testing data is the same as that for the Game of 24, with each consisting of 100 problems. We evaluate the success rates using the top 1 candidate and the top 5 candidates, denoted as Success@1 and Success@5, respectively.

The overall results are provided in the Table 3. From the results, we could find that the direct planning methods, i.e., IO and CoT methods generally demonstrate lower success rates compared to our proposed method and other comparable methods like ToT. Their significantly lower Success@1 rates across all games reflect LLMs' struggle to identify valid action sequences for the desired goals. It also underscores the value of search strategies guided by pre-defined symbolic workflows in addressing such complex tasks, where structured symbolic guidance helps navigate through the challenges of constraint satisfaction and goal attainment. By leveraging these more structured approaches, notably our hybrid model, improvements are evident across various game scenarios, showcasing its ability to enhance planning accuracy and success rates effectively. This emphasizes the potential for integration of symbolic reasoning to comple-

Method	LLM	Game of 24		Game of 28		Game of 30	
		Success@1	Success@5	Success@1	Success@5	Success@1	Success@5
IO	GPT-3.5-turbo	14.54	-	6.990	-	5.340	-
	GPT-4o	12.47	-	4.260	-	5.840	-
	GPT-4o-mini	13.78	-	7.010	-	5.210	-
	DeepSeek-v3	15.90	-	6.050	-	9.800	-
CoT	GPT-3.5-turbo	5.420	-	8.430	-	5.300	-
	GPT-4o	9.440	-	4.260	-	13.39	-
	GPT-4o-mini	6.060	-	7.010	-	4.570	-
	DeepSeek-v3	12.34	-	6.050	-	12.89	-
ToT	GPT-3.5-turbo	30.00	46.00	0.000	0.000	0.000	0.000
	GPT-4o	81.00	85.00	47.00	51.00	67.00	67.00
	GPT-4o-mini	28.00	42.00	0.000	0.000	26.00	36.00
	DeepSeek-v3	86.00	87.00	36.00	39.00	66.00	68.00
RNSP	GPT-3.5-turbo	57.00	57.00	44.00	44.00	50.00	50.00
	GPT-4o	77.00	77.00	43.00	43.00	56.00	56.00
	GPT-4o-mini	56.00	56.00	49.00	49.00	55.00	55.00
	DeepSeek-v3	**93.00**	**93.00**	**56.00**	**56.00**	**70.00**	**70.00**

Table 3: Experimental results on Game of 24, Game of 28, and Game of 30.

ment LLMs, bridging gaps in tasks requiring intricate planning and reasoning capabilities. In contrast, our proposed method RNSP demonstrates robust performance across all Games of 16, 24, and 30. For instance, RNSP with Deepseek-v3 achieves a success rate of 93% for the Game of 24, which is substantially higher than the ToT's 86% and CoT's 12%. Moreover, the performance gap exhibited by ToT in the Games of 28 and 30 compared to the Game of 24 highlights the fragility of self-evaluation. In contrast, our approach demonstrates greater stability, achieving success rates of 93%, 56%, and 70% for the Games of 24, 28, and 30, respectively. This indicates that relying solely on pre-trained LLMs for general planning is unreliable. It is necessary to incorporate symbolic verifiers to aid LLMs in understanding the requirements of downstream planning tasks.

4.3 Constrained Knapsack

This is a classic knapsack problem aimed at maximizing the value of selected blocks under specific constraints. Each query presents blocks with different colors and scores, and the planning objective is to select a specified number of unique blocks with particular required colors while maximizing the total score. In this scenario, we consider five available block colors: red, yellow, black, pink, and blue. Each block is assigned a random score ranging from 1 to 20, along with a random color from the available options. The total number of blocks is randomly determined, ranging from 50 to 200. For the task query, we randomly select 1 to 3 colors as the required colors and specify that 10 blocks must be chosen. In this task, the method must consider the number of remaining blocks available for selection at each step, as well as the impact of the required colors on the final scores. This involves evaluating the current state to effectively guide the search.

We consider two metrics for evaluation: the success rate of samples achieving the maximum possible score, abbreviated as SR, and the relative percentage of the score obtained by the method compared to the optimal score, abbreviated as RP. It is important to note that any solution failing to meet the color requirements will have its score disregarded and replaced with zero. This process prevents the model from converging on a greedy selection of high scores without adhering to the constraints, thereby ensuring a fair evaluation. The results of this experiment are provided in Table 4.

Method	LLM	SR@1 (%)	RP (%)
IO	GPT-3.5-turbo	0.000	0.829
	GPT-4o-mini	0.000	0.818
	DeepSeek-v3	0.560	2.572
CoT	GPT-3.5-turbo	0.000	0.663
	GPT-4o-mini	0.000	0.853
	DeepSeek-v3	0.000	0.000
ToT	GPT-3.5-turbo	0.000	7.351
	GPT-4o-mini	0.000	14.15
	DeepSeek-v3	8.000	70.10
RNSP	GPT-3.5-turbo	16.00	90.16
	GPT-4o-mini	28.00	94.41
	DeepSeek-v3	72.00	95.65

Table 4: Experimental Results on Constrained Knapsack problem.

From the results, we could find that the IO method exhibits weak performance across both metrics. The SR@1 of GPT-3.5-turbo and GPT-4o-mini is consistently zero, indicating it fails to achieve the maximum possible score on any query sample. Additionally, the relative percentage (RP) is notably low, with values like 0.829% and 0.818%, demonstrating its general ineffectiveness in this constrained planning context. The Chain-of-Thought (CoT) method, typically regarded as

effective for long-chain reasoning tasks, also shows poor performance in this scenario. It results in zero success rates across all tested LLMs. Notably, CoT even leads to a decrease in the RP score. This also suggests that solely relying on a training-free LLM for constraint-based planning tasks is highly limited. The ToT method has achieved notable gains in RP, particularly when combined with DeepSeek-v3, where it reached an RP of 70.10%. However, it remains weak in terms of SR, with a success rate of only 8% for DeepSeek-v3 and 0% for the other two LLMs. This indicates that while ToT improves the value estimation aspect of planning, it struggles to satisfy constraint requirements effectively during its self-evaluation process. Our method, RNSP, not only achieved over 90% RP across all three LLMs but also demonstrated significant improvements in SR. Notably, with DeepSeek-v3, it reached an SR of 72%. This performance highlights how reinforced optimization, guided by symbolic feedback, enhances the value estimation. As a result, planning becomes more effective in taking the goal constraints into consideration and overcoming the limitations of self-evaluation.

4.4 Travel Planning

We further conduct the experiments on a real-world planning benchmark TravelPlanner [Xie *et al.*, 2024]. TravelPlanner is a novel benchmark dataset designed to evaluate the planning capabilities of language agents in real-world scenarios, specifically focusing on travel planning. This dataset provides a comprehensive testbed to assess how well language agents can make a travel plan and satisfy multiple constraints while generating feasible plans. Each query involves planning a travel itinerary, including transportation, meals, attractions, and accommodations while adhering to various constraints. As pointed out by [Shao *et al.*, 2024b], in travel planning, unrealistic or incorrect information might lead to shortcutting logical constraints, such as misreporting costs to fit budget requirements. To evaluate different methods, we utilize the Conditional Logical Pass Rate (C-LPR) and Final Pass Rate (FPR). The C-LPR represents the pass rate of meeting hard logical constraints under the conditions of environmental constraint validity. Meanwhile, the FPR indicates the proportion of itineraries that satisfy both environmental constraints and user-specific logical requirements.

$$\textit{C-LPR} = \frac{\sum_{q \in Q} \mathbb{I}[R_q \models C_q] \cdot \sum_{p \in G_q} \mathbb{I}[R_q \models p)]}{\sum_{q \in Q} |G_q|}$$

Q is the set of querying samples, G_q is the set of goal constraints for query q, C_q is the progress constraints, and R_p is the resulted plan for query q.

The results are provided in the Table 5. From the results, we could find that the IO method demonstrates relatively weak performance. Specifically, GPT-4o-mini yields a C-LPR of 3.809% and an FPR of 0.000%, while DeepSeek-v3 fares slightly better with a C-LPR of 10.47% and an FPR of 4.444%. These low values indicate that the IO method struggles to effectively meet both environmental and user-specific constraints in travel planning tasks. The Chain-of-Thought (CoT) method also underperforms in this scenario. These results suggest that CoT, while useful for long-chain reasoning in other contexts, is less capable of handling complex logical constraints in travel planning, which needs more than 20 reasoning steps. Thanks to the symbolic workflow, the ToT method more easily satisfies the requirements of process constraints, showing better results compared to the IO and CoT methods. Despite this improvement, the ToT method struggles with long-chain goal constraints in travel planning, such as considering the total itinerary cost and the variety of cuisine types. The self-evaluation capability of the LLMs underlying the ToT method is insufficient for these complex tasks. Our method, RNSP, addresses this limitation by utilizing reinforced value estimation. This approach feeds the impact of long-chain constraints back into intermediate nodes within the search tree, thereby providing a foresightful state evaluation function during subsequent searches. As a result, it enhances the planning capacity by improving the ability to anticipate and meet the intricate requirements of long-chain constraints.

Method	LLM	C-LPR	FPR
IO	GPT-4o-mini	3.809	0.000
	DeepSeek-v3	10.47	4.444
CoT	GPT-4o-mini	2.857	2.222
	DeepSeek-v3	8.571	2.222
ToT	GPT-4o-mini	18.09	11.11
	DeepSeek-v3	19.04	11.11
Ours	GPT-4o-mini	28.57	22.22
	DeepSeek-v3	25.71	15.55

Table 5: Experimental Results on Travel Planning.

5 Conclusion and Discussion

In this work, we addressed the limitations of Large Language Models (LLMs) in constrained planning tasks by proposing Reinforced Neuro-Symbolic Planning (RNSP), a framework that combines the generative power of LLMs with symbolic verification. The symbolic verifier ensures strict adherence to constraints, while reinforcement learning principles enable iterative refinement of state evaluations. By using T-norms to aggregate constraint satisfaction levels, our framework provides effective guidance for test-time search, bridging the gap between neural flexibility and symbolic rigor.

While RNSP has significantly improved constraint satisfaction, its computational efficiency can be enhanced for real-time applications. The current method uses the concise beam search to verify the effectiveness of the RNSP idea. Future work could explore advanced exploitation-exploration techniques to explore more informative states in the value optimization phase and reduce the overhead during the test-time search, which may make the framework more practical for deployment in time-sensitive environments. Another interesting direction is to convert the learned value estimation into a token-level reward model, which could be used to assist in the post-training of the LLMs. It may provide a novel learning paradigm for enhancing the LLM's intrinsic ability to perform constrained planning.

Acknowledgements

This research was supported by the Key Program of Jiangsu Science Foundation (BK20243012), Leading-edge Technology Program of Jiangsu Science Foundation (BK20232003), the Fundamental Research Funds for the Central Universities (022114380023) and the Postgraduate Research & Practice Innovation Program of Jiangsu Province (KYCX24_0233).

Contribution Statement

Jie-Jing Shao and Hong-Jie You contributed equally to this work. Lan-Zhe Guo is the corresponding author.

References

[Brown *et al.*, 2020] 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. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020*, Virtual Event, 2020.

[Bubeck *et al.*, 2023] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Túlio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with GPT-4. *CoRR*, abs/2303.12712, 2023.

[Chang *et al.*, 2024] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):39:1–39:45, 2024.

[Chen *et al.*, 2024] Ziru Chen, Michael White, Raymond J. Mooney, Ali Payani, Yu Su, and Huan Sun. When is tree search useful for LLM planning? it depends on the discriminator. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 13659–13678, Bangkok, Thailand, 2024.

[Hao *et al.*, 2023] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8154–8173, Singapore, 2023.

[Huang *et al.*, 2022] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 9118–9147, Baltimore, MD, 2022.

[Huang *et al.*, 2023] Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore, 2023.

[Huang *et al.*, 2024] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The 12th International Conference on Learning Representations*, Vienna, Austria, 2024.

[Ichter *et al.*, 2022] Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu, Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Nikhil J. Joshi, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. Do as I can, not as I say: Grounding language in robotic affordances. In *Conference on Robot Learning*, volume 205, pages 287–318, Auckland, New Zealand, 2022.

[Jia *et al.*, 2025] Lin-Han Jia, Wen-Chao Hu, Jie-Jing Shao, Lan-Zhe Guo, and Yu-Feng Li. Verification learning: Make unsupervised neuro-symbolic system feasible. *arXiv preprint arXiv:2503.12917*, 2025.

[Kambhampati *et al.*, 2024] Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Saldyt, and Anil Murthy. Position: Llms can't plan, but can help planning in llm-modulo frameworks. In *Proceedings of the 41st International Conference on Machine Learning*, Vienna, Austria, 2024.

[Keskar *et al.*, 2019] Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858, 2019.

[Li *et al.*, 2024] Junyi Li, Tianyi Tang, Wayne Xin Zhao, Jian-Yun Nie, and Ji-Rong Wen. Pre-trained language models for text generation: A survey. *ACM Comput. Surv.*, 56(9):230:1–230:39, 2024.

[Liang *et al.*, 2024] Jenny T. Liang, Chenyang Yang, and Brad A. Myers. A large-scale survey on the usability of AI programming assistants: Successes and challenges. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering, ICSE 2024, Lisbon, Portugal, April 14-20, 2024*, pages 52:1–52:13, 2024.

[Lightman *et al.*, 2024] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy

Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The 12th International Conference on Learning Representations*, Vienna, Austria, 2024.

[Liu *et al.*, 2024a] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[Liu *et al.*, 2024b] Fang Liu, Yang Liu, Lin Shi, Houkun Huang, Ruifeng Wang, Zhen Yang, and Li Zhang. Exploring and evaluating hallucinations in llm-powered code generation. *CoRR*, abs/2404.00971, 2024.

[Min *et al.*, 2024] Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2):30:1–30:40, 2024.

[OpenAI, 2022] OpenAI. Introducing chatgpt. https://openai.com/index/chatgpt/, 2022. Accessed: 2022-11-30.

[OpenAI, 2024a] OpenAI. Gpt-4o mini: advancing cost-efficient intelligence. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/, 2024. Accessed: 2024-07-18.

[OpenAI, 2024b] OpenAI. Hello gpt-4o. https://openai.com/index/hello-gpt-4o/, 2024. Accessed: 2024-05-13.

[Pan *et al.*, 2023] Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3806–3824, 2023.

[Shao *et al.*, 2024a] Jie-Jing Shao, Hao-Ran Hao, Xiao-Wen Yang, and Yu-Feng Li. Learning for long-horizon planning via neuro-symbolic abductive imitation, 2024.

[Shao *et al.*, 2024b] Jie-Jing Shao, Xiao-Wen Yang, Bo-Wen Zhang, Baizhi Chen, Wen-Da Wei, Guohao Cai, Zhenhua Dong, Lan-Zhe Guo, and Yu feng Li. Chinatravel: A real-world benchmark for language agents in chinese travel planning, 2024.

[Silver *et al.*, 2022] Tom Silver, Ashay Athalye, Joshua B. Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Learning neuro-symbolic skills for bilevel planning. In *Conference on Robot Learning*, volume 205, pages 701–714, Auckland, New Zealand, 2022.

[Snell *et al.*, 2024] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *CoRR*, abs/2408.03314, 2024.

[Vilar *et al.*, 2023] David Vilar, Markus Freitag, Colin Cherry, Jiaming Luo, Viresh Ratnakar, and George F. Foster. Prompting palm for translation: Assessing strategies and performance. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, pages 15406–15427, Toronto, Canada, 2023.

[Wang *et al.*, 2023] Renhao Wang, Jiayuan Mao, Joy Hsu, Hang Zhao, Jiajun Wu, and Yang Gao. Programmatically grounded, compositionally generalizable robotic manipulation. In *The 11th International Conference on Learning Representations*, Kigali, Rwanda, 2023.

[Wei *et al.*, 2022] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022*, New Orleans, LA, 2022.

[Xie *et al.*, 2023] Yuxi Xie, Kenji Kawaguchi, Yiran Zhao, James Xu Zhao, Min-Yen Kan, Junxian He, and Michael Qizhe Xie. Self-evaluation guided beam search for reasoning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023*, New Orleans, LA, 2023.

[Xie *et al.*, 2024] Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. Travelplanner: A benchmark for real-world planning with language agents. In *Proceedings of the 41st International Conference on Machine Learning*, Vienna, Austria, 2024.

[Xu *et al.*, 2019] Danfei Xu, Roberto Martín-Martín, De-An Huang, Yuke Zhu, Silvio Savarese, and Li Fei-Fei. Regression planning networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019*, pages 1317–1327, Vancouver, Canada, 2019.

[Yang *et al.*, 2024] Xiaowen Yang, Wenda Wei, Jie-Jing Shao, Yufeng Li, and Zhi-Hua Zhou. Analysis for abductive learning and neural-symbolic reasoning shortcuts. In *Proceedings of the 41st International Conference on Machine Learning*, Vienna, Austria, 2024.

[Yao *et al.*, 2023a] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023*, New Orleans, LA, 2023.

[Yao *et al.*, 2023b] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The 11th International Conference on Learning Representations*, Kigali, Rwanda, 2023.

[Zhao *et al.*, 2023] Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for large-scale task planning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023*, New Orleans, LA, 2023.