

DO THINK TAGS REALLY HELP LLMs PLAN? A CRITICAL EVALUATION OF REACT-STYLE PROMPTING

Anonymous authors

Paper under double-blind review

ABSTRACT

The reasoning abilities of Large Language Models (LLMs) remain a topic of debate, which are critically tested in sequential decision-making problems. ReAct, a recently popular method has gained popularity for claiming to enhance LLM reasoning abilities while directly prompting them by “*interleaving reasoning trace with action execution*” in text-based planning domains such as AlfWorld and WebShop. However, given the different components of ReAct-style prompting, it remains unclear what the source of improvement in LLM performance is. In this paper, we critically examine the claims of ReAct-style prompting for sequential decision-making problems. By introducing systematic variations to the input prompt, we perform a sensitivity analysis along the original claims of ReAct. Contrary to these claims and common use-cases that utilize ReAct-style prompting, we find that the performance is minimally influenced by the interleaved reasoning trace or by the content of these generated reasoning traces. Instead, the performance of LLMs is primarily driven by the unreasonably high degree of similarity between input example tasks and queries, implicitly forcing the prompt designer to provide instance-specific examples which significantly increases the cognitive burden on the human. Our empirical results, on the same suite of domains as ReAct, show that the perceived reasoning abilities of LLMs stem from the exemplar-query similarity and approximate retrieval rather than any inherent reasoning abilities.

1 INTRODUCTION

Large Language Models (LLMs) have seen rapid advancements specifically in Natural Language Processing and Understanding (NLP & NLU). LLMs have unparalleled capabilities in text generation, summarization, translation, question answering to name a few. (Bubeck et al., 2023). Motivated by these capabilities of LLMs, there has also been a rush to look for other emergent abilities—especially for reasoning and planning. A popular way of enhancing LLM performance on reasoning/planning tasks has been in-context prompting or prompt-engineering (Sahoo et al., 2024) to include instructions (Giray, 2023), syntax structure (Marvin et al., 2023), criticism and plan guidance with verification (Kambhampati et al., 2024) etc. Among these approaches, ReAct (Yao et al., 2022b), presented at ICLR 2023, stands out which claims to improve LLM planning abilities through the use of reasoning traces interleaved with action execution given as plan guidance. While their original study includes multiple domains tasks such as question answering on HotPotQA and FEVER datasets (Yang et al., 2018; Thorne et al., 2018), in this paper we are particularly interested in planning tasks where ReAct claims superior performance.

In our initial experiments with ReAct for planning, we found that the system is overly dependent on a high degree of syntactic similarity between the example prompt and the query, and is extremely brittle to minor perturbations to the input prompt. For example, when provided with an explicit set of examples of *pick-and-place-object* task and asked to plan for a *pick-and-place-two-objects* task, it should be trivial to generalize the solution of the examples to the queried task. Unfortunately, even such a minor variation to the original ReAct agent setup disrupts its performance.

Given the seemingly widespread adoption of ReAct methodology (as of this writing, it has 1,408 citations), the brittleness we witnessed calls for a systematic study of the factors contributing to the performance of ReAct-based LLM Agents. Moreover, recent studies have highlighted similar case studies on the original Chain-of-Thought idea (Stechly et al., 2024a; Sprague et al., 2024). Based on

the claims of (Yao et al., 2022b), we isolate three possible reasons for the claimed performance of ReAct framework: 1) the utility of interleaving reasoning trace during action execution, 2) the utility of providing plan guidance, and, 3) the significance of example prompt provided to the the LLM.

The brittleness of ReAct becomes apparent when considering the variability in prompt designs. Depending on the domain and task, prompt designers may provide abstract guidance, task-specific instructions within the same domain, analogical examples, or global reasoning traces. ReAct’s inability to robustly adapt to such variations underscores its limitations in handling diverse input prompts, thereby necessitating a closer examination of its design and implementation. In this work, we systematically evaluate the brittleness of ReAct by studying which potential factors contribute to its performance. This analysis is conducted through variations in input prompts to understand how a ReAct LLM Agent responds to (1) where the guidance is provided, (2) the different types and structure of this guidance, and finally, (3) on varying the resemblance of example prompt to the queried problem. We investigate the research questions :

RQ1: Does the agent performance depend on interleaving reasoning trace with action execution?

RQ2: How does the nature of the reasoning trace or guidance information affect the performance of LLM Agents?

RQ3: How does the similarity between the example (problem, solution) and the query (problem, ?), which are present in the prompt, affect LLM Agent performance?

We conduct extensive experiments on the AlfWorld and WebShop domain using various LLM Models, including GPT-3.5-turbo, GPT-3.5-instruct, GPT-4-0314(Old Variant), GPT-4-0613 (Latest Variant), GPT-4o, Claude-Opus and Llama 3.1-8b. Through our comprehensive empirical study, we answer each of the research questions above. (RQ1) We find that LLM performance in-fact improves when the reasoning trace is **not** interleaved with action execution. (RQ2) Moreover, providing weaker guidance or placebo-guidance (where the text provides no information about the task) has comparable performance to strong reasoning trace-based guidance. Answers to RQ1 and RQ2 suggest that the source of ReAct LLM agent performance is **not** the interleaving aspect or the content of the reasoning trace. Finally, in (RQ3), we see that variations to the example prompt such that it belongs to different task within the same domain, or has a different goal or plan attributes than the queried problem; causes the performance of ReAct-LLM Agent to plummet.

Our findings highlight that the benefits of ReAct-style prompting are present when prompt engineers can curate highly instance-specific examples. This may not scale for domains with a large number of problem instance classes, and it places an undue burden on prompt engineers to provide instance specific examples. Finally, our experiments call into question claims of enhanced “emergent reasoning” of LLMs with prompt engineering efforts; corroborating contemporary research (Verma et al., 2024; Valmeekam et al., 2024; Stechly et al., 2024b; Ullman, 2023; Schaeffer et al., 2023; McCoy et al., 2023; Stechly et al., 2024a; Sprague et al., 2024) questioning reasoning abilities of LLMs.

For the rest of the paper, we begin with discuss the relevant literature discussing LLM prompting for planning problems in Section 2. Next, we provide the preliminaries in Section 3, and discuss our investigations of ReAct-style prompting in detail in Section 4. We present our empirical results with a detailed discussion drawing insights from our results in Section 5. Finally, we conclude the work in Section 6. An appendix with additional experiment details, results, and prompts, has also been attached, and the code will be released on acceptance.

2 RELATED WORK

Large Language Models have been shown to be successful in a plethora of natural language tasks (Kocón et al., 2023; Gilardi et al., 2023; Zhu et al., 2023; Bubeck et al., 2023; Bhattacharjee et al., 2024). However, there are two schools of thought when it comes to utilizing off-the-shelf LLMs for planning and reasoning tasks. Works such as Chain of Thought, ReAct, and others that followed (Wei et al., 2022; Yao et al., 2023; Long, 2023; Yao et al., 2024; Besta et al., 2024; Fu et al., 2024; Aksitov et al., 2023), have argued about the reasoning abilities of LLMs by proposing prompting methods. On the other hand, (Valmeekam et al., 2024; Stechly et al., 2024b) have refuted these claims by showing the inability of LLMs to solve deterministic planning and classical reasoning problems.

In particular, for investigating the use of LLMs in solving decision making problems, AlfWorld (Shridhar et al., 2020) is a popular domain that was originally proposed for training text-based

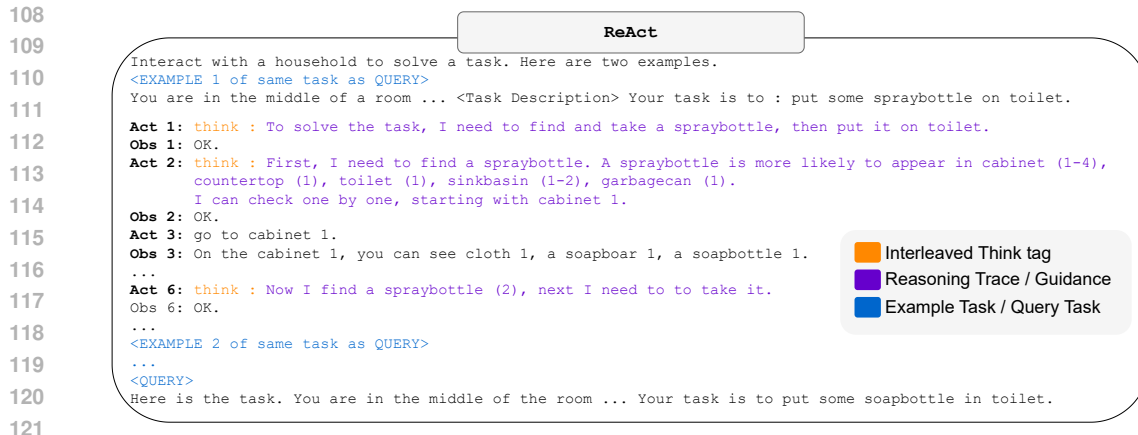


Figure 1: An example of ReAct in AlfWorld. We highlight the main components of ReAct, i.e., Interleaved reasoning and acting, the reasoning trace / plan guidance and the example and query task.

Reinforcement Learning agents. Lately, works such as ReAct, Reflexion, and their other variants (Yao et al., 2022b; Shinn et al., 2023) have argued on the prowess of LLMs’ reasoning abilities on AlfWorld. Furthermore, there have been several extensions to ReAct that boost their generalization abilities across more domains including multi-modal domains (Yang et al., 2023; Castrejon et al., 2024), autonomous vehicles (Cui et al., 2024), table question answering (Zhang et al., 2023), etc. While the effectiveness of ReAct is celebrated across different areas, these works only depend on anthropomorphization of LLMs (Min et al., 2022; Peng et al., 2024) for using ReAct based prompting with no justification on the source of improvement in performance. This motivates our work in investigating the components of ReAct with respect to sequential decision-making problems and analyzing the role each component plays.

3 PRELIMINARIES

3.1 DOMAINS

AlfWorld: (Shridhar et al., 2020) is a synthetic text-based game built on top of a STRIPS-style PDDL domain description (Fikes & Nilsson, 1971). ReAct (Yao et al., 2022b) defines six tasks (or problem classes) within this domain namely - Put, Clean, Heat, Cool, Examine, and PutTwo. Each problem class consists of several problem instances, such as *put a spraybottle on toilet* (see Fig. 1) is an example instance of Put class. Since AlfWorld is a partially observable environment, each of these problem instances can be solved by navigating and interacting with the environment simulator via text actions. For example, this task can be solved by the following actions- go to cabinet 2, take spraybottle 2 from cabinet 2, go to toilet 1, put spraybottle 2 in/on toilet 1.

WebShop: (Yao et al., 2022a) is an online shopping website environment with 1.18M real-world products and 12K human instructions. The agent is provided with an initial human instruction (for example, “I am looking for a nightstand with drawers. It should have a nickel finish, and priced lower than \$140”). The agent’s task is to crawl the shopping environment using actions such as search ‘nightstand drawers’, choose ‘white buttons’, back to search, etc. For this work, we randomly sample 500 test instructions from the environment and evaluate the success rate of the agent’s task completion.

3.2 REACT

ReAct (Yao et al., 2022b) claims to increase LLM’s performance on text-based planning tasks such as AlfWorld and WebShop primarily by augmenting the original action space of the agent with a *think* action. The *think* action tag provided by ReAct is claimed to comprise of **Reasoning + Action** trace that is provided in the solution for the example problems (exemplars) as part of the prompt. During

162 execution, the LLM is expected to generate a *think* action tag for the queried problem instance that is
 163 semantically similar to the one provided for the examples in the prompt.

164 **Location of *THINK* tag** In ReAct, the integration of the *think* tag within actions serves to expand
 165 the action space. This allows the language model (LLM) agent to execute a *think* action, prompting
 166 an ‘OK’ response. Through analysis of example prompts in ReAct experiments, we identify various
 167 instances of the *think* action. Typically, it appears after stating the problem instance, reiterating the
 168 task, and providing problem-specific guidance. However, the authors offer no structured guidelines
 169 for its implementation, placement, or guidance. This observation aligns with feedback from the
 170 paper’s reviewers (OpenReview, 2024) citing inconsistencies in the prompting format.

171 **Content of *THINK* tag** In ReAct, the *think* action consistently provides the decision-making
 172 agent with success-oriented guidance for task completion. For instance, upon encountering a
 173 spraybottle, the prompt might include: *think*: Now I find a spraybottle (2).
 174 Next, I need to take it. This guidance exposes forthcoming actions and sub-tasks for
 175 the agent.

176 **Few shot *EXAMPLES*** In the AlfWorld domain (which is a PDDL domain), ReAct authors (Yao
 177 et al., 2022b) classify six problem classes or tasks: *Put*, *Clean*, *Heat*, *Cool*, *Examine*,
 178 *PutTwo*. Despite representing different tasks, they share the same environment dynamics and action
 179 space, allowing for very similar execution trace. For instance, a *Heat* task might involve *Putting*
 180 an item into a microwave. In ReAct experiments, authors provide two example problem-solution
 181 pairs (referred to as exemplars in our work) before querying the LLM agent with a problem instance.
 182 Authors force ReAct agent to use examples and queries belonging to the same problem class without
 183 motivating this design decision. However, the queried problem may differ in objects or locations
 184 from the exemplars.

186 4 A CRITICAL EVALUATION OF REACT-STYLE PROMPTING

187 We examine the claims of ReAct to understand the performance of ReAct-based LLM agents. It is
 188 crucial to assess whether ReAct’s fundamental claims hold, particularly in planning. As outlined in
 189 Section 3, ReAct comprises three main components: interleaving the *think* tag with actions, plan
 190 guidance after the *think* tag, and the selection of exemplar problems for LLM prompts. We perform
 191 a sensitivity analysis by proposing alternatives along these three dimensions. The subsequent sections
 192 explore the design of exemplar prompt variations to investigate our research questions concerning
 193 the claims of ReAct. Each variation modifies the base ReAct prompt, and we use the AlfWorld
 194 domain as the running example for discussing these variations in the following sub-sections. We
 195 curate variations for WebShop domain in a similar manner. Readers can find the variations in the
 196 attached supplementary materials.

198 4.1 RQ1 : INTERLEAVING *thinking* WITH ACTING

199 *Does the agent performance depend on interleaving reasoning trace with action execution?*

200 To answer this research question, we propose collating the guidance information contained within the
 201 multiple *think* tags present in the examples of the input prompt into a single *think* tag appended after
 202 the example problem is specified. This approach can be interpreted as Chain-of-Thought (Kojima
 203 et al., 2022; Wei et al., 2022), where guidance information is generated before action execution.

204 **Variation 1: Exemplar-based CoT** AlfWorld is a partially observable environment where an agent
 205 can only observe objects after reaching that location. Hence, we remove specific location and
 206 object identifiers to modify the *think* actions that are originally interleaved with other actions in the
 207 environment (see RQ1 a in Figure 2). Finally, we append all the *think* actions together at the beginning
 208 of the example problem. **Intuition:** Problem-specific guidance for a sequential decision-making
 209 agent can be given step-by-step (as in ReAct) or all at once. **Note, that this variation is possible since**
 210 **AlfWorld is not a dynamically changing environment in which case providing information on the task**
 211 **will not be possible.**

212 **Variation 2: Anonymized Exemplar-CoT** We take one step further and modify the *think* tag to
 213 remove references to specific locations and objects, making it more general (see RQ1 b in Figure
 214 2). Similarly, in WebShop we anonymize specific item options as *desired options*. **Intuition:**
 215

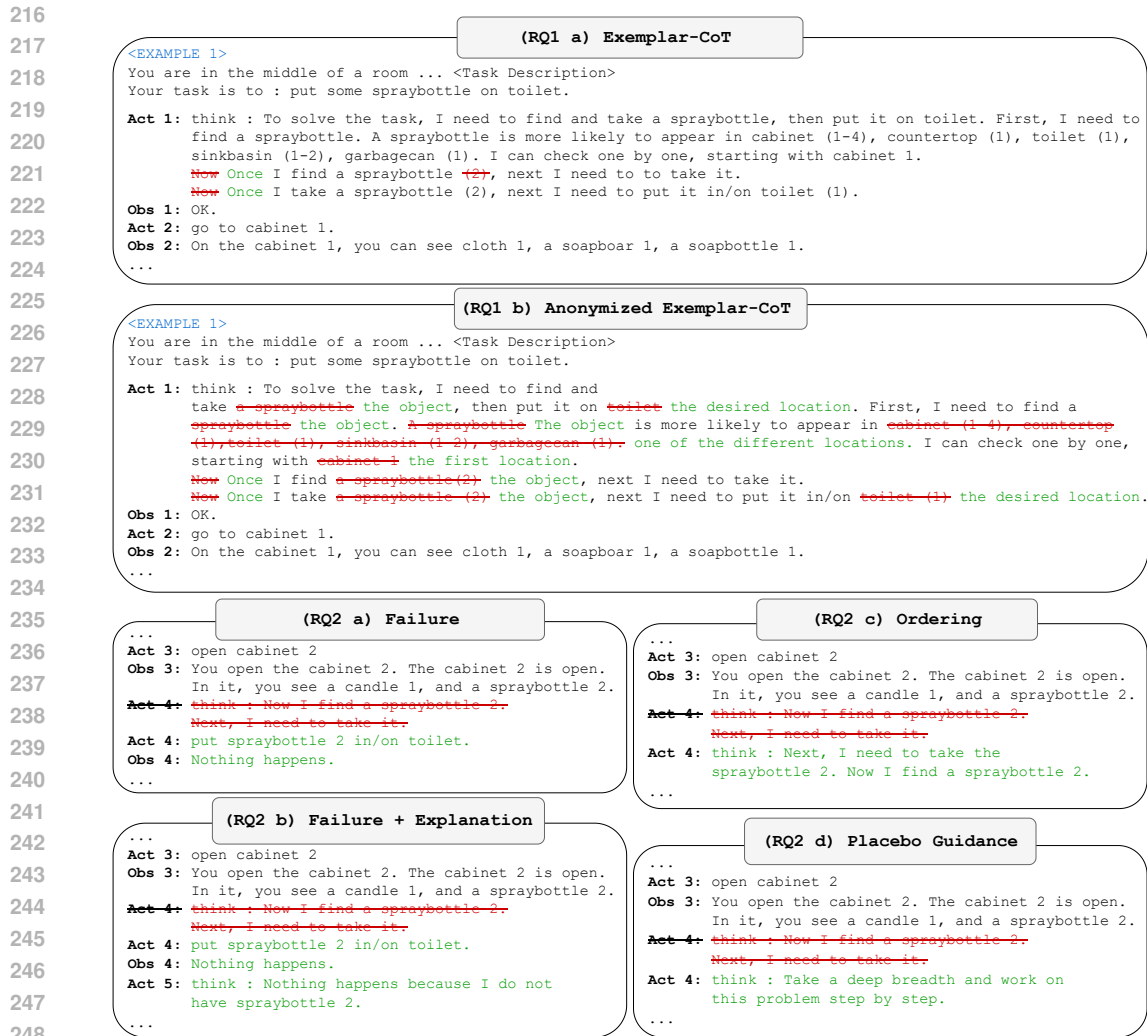


Figure 2: Example of prompt variations considered for RQ1 and RQ2.

Exemplars can be made more general by providing abstract guidance and exploiting LLMs ability to identify necessary semantic entity relations.

4.2 RQ2 : PLAN GUIDANCE FOLLOWING *think* TAG

How does the nature of the reasoning trace or guidance information affect the performance of LLM?

ReAct claims to use reasoning trace as the guidance information following the *think* tag. For instance, in ReAct () thoughts are to (1) decompose the goal (2) track subgoal completion (3) determine the next subgoal and (4) reason via common-sense where to find and object and what to do with it. It is, however, unclear what is the motivation to use these as the reasoning trace. The potential anthropomorphization of large language models (LLMs) may suggest that their thought processes are similar to the abstract plans humans make, and that they must be prompted in the same manner. However, it is unclear why this assumption should hold true. Alternatives can be, we can prompt the LLM to reflect on past failures and provide possible explanations (hindsight-guidance) or We can substitute task-relevant guidance with placebo-guidance by using "magic incantations".

Variation 1: Failure From the example prompts used in ReAct, we note that none of the examples for any task consist of invalid actions. We inject two invalid actions in the execution trace : the first that attempts to execute the action pertinent to the task (such as `put spraybottle 2 in/on`

toilet) when not possible and, second, executes some other invalid action. We include the expected simulator response, `Nothing happens.`, when invalid actions are taken. **Intuition:** Reasoning trace can be about *what to do* such as subgoals of the future, or *what not to do* such as mistakes in hindsight. This should be weaker guidance than in base ReAct as the exemplars do not point out what to do next.

Variation 2: Failure + Explanation We place *think* actions after invalid actions injected in Failure Variation which consist of explanations for the failure (see RQ2 b in Figure 2). **Intuition:** We can augment pointing out mistakes in hindsight with explanations to avoid similar failures. This is stronger guidance signal than Failure, however, the exemplars still not provide information on what to do next.

Variation 3: Guidance Ordering LLMs are known to be susceptible to minor syntactic perturbations to inputs. We test whether it is true for guidance information given as prompt as well (see RQ2 c in Figure 2). We identify chain of subtasks in a reasoning trace $S_1 \rightarrow S_2 \cdots S_n$ and reverse it to be $S_n \rightarrow S_{n-1} \cdots S_1$. **Intuition:** LLM agent should be invariant to the syntax of reasoning trace if the semantic information is preserved. This does not change the reasoning trace from the perspective of information content.

Variation 4: Placebo Guidance It is unclear to what extent LLM agent uses the supposed helpful thoughts for the decision making task. In this variation we replace *think* tag guidance with a placebo thought that does not contain any task relevant information (see RQ2 c in Figure 2), but has been widely used as prompt engineering trick (Kojima et al., 2022). **Intuition:** According to claims of ReAct, we expect the performance to get worse when the guidance does not have any information useful for task success.

4.3 RQ3 : SIMILARITY BETWEEN *EXAMPLES* AND *QUERY*

How does the similarity between the example (problem, solution) and the query (problem, ?), which are present in the prompt, affect LLM Agent performance?

RQ3 investigates the role of example similarity to the query in LLM agent’s performance. Establishing problem similarity can be challenging, especially where minor variations to the problem can have varied interpretations (such as an analogy to a different task altogether). Our work explores this challenge in a systematic way. During example prompt construction, prompt designers may use synonyms to refer to objects (*Domain*), come up with examples where the agent task is the same as query but the goals are different (*Instance*), or provide optimal solutions as the examples (*Optimal*) preventing LLM to obtain information regarding exploration strategy. Furthermore, given that the domain has the same underlying action dynamics and that the tasks reuse several actions, prompt designers may choose to provide query specific example prompts (as in base ReAct), provide one of a different task and one of the same task (*One*), provide both examples to be of a different task (*Both*), or take an exhaustive approach and provide one example of all tasks (*All*).

Variation 1: Synonyms - (Domain) For this variation, we replace the object and location names in the example prompts with their synonyms. For example, `spraybottle` \rightarrow `aerosolbottle`, `cabinet` \rightarrow `cupboard`, and, `microwave` \rightarrow `oven`. We make 36 such changes to object and location names across all the examples. Note that the object names / location are unchanged for the problem query and subsequent interaction with the simulator. **Intuition:** Exemplar guidance maybe specified with alternate synonymous object and location names. Reasoning agents should be invariant to variable name substitution for closed world dynamics such as PDDL based *AlfWorld*.

Variation 2: Problem Instance-level - Instance We inject instance-level changes to the examples provided in the prompts. Recall that we are updating the base ReAct’s prompts, where the exemplar tasks are same as the query. We change the goal location in exemplar problem to ensure that it does not match with any of the goal locations in query problem. Moreover, we add repetitive yet futile actions in the exemplar execution trace which does not effect the solution. **Intuition:** Ensuring a different goal location in exemplar from the queried problem is a natural usecase. Moreover, exemplars may contain arbitrary exploration strategies such as action repetition (Sharma et al., 2017). By ReAct’s claims, LLM agent performance should not be affected.

Variation 3: Problem Level - Both, One, All Recall that the environment dynamics for all the tasks are the same. In fact, several tasks subsume the use of our tasks such as `Heat` requires the

Table 1: Average Success % of LLM for RQ1 and RQ2 on six AlfWorld tasks.

Model / Prompt	Act	ReAct	RQ1		RQ2			
			CoT	Anon. CoT	Placebo	Order	Failure	Explanation
GPT-3.5-Turbo	34.3	27.6	46.6	41	30	28.3	43.3	41.6
GPT-3.5-Instruct	44	50.7	61.9	50.7	41	42.5	47	44.7
GPT-4-0314 (Old)	-	23.3	43.3	33.3	36.6	30	50	36.6
GPT-4-0613 (Latest)	70.0	26.7	40.0	26.6	36.6	30	60	36.6
Claude-Opus	43.3	56.6	50	46.6	30	50	53.3	30

Table 2: Average Success % of LLM for RQ1 and RQ2 on WebShop tasks.

Model / Prompt	Act	ReAct	RQ1		RQ2		
			CoT	Anon. CoT	Placebo	Failure	Explanation
GPT-3.5-Turbo	1.12	1.04	2.20	1.88	1.52	3.48	3.48
GPT-3.5-Instruct	7.24	7.16	7.52	6.12	7.40	7.20	7.24
GPT-4-0613 (Latest)	8	4	8	8	6	8	8
GPT-4o	4.64	2.24	4.68	4.52	4.08	4.68	4.68
Claude-Opus	4	4	4	2	4	2	4
LLAMA-3.1-8B	1.44	3.16	3.28	3.92	2.04	1.20	2.16

agent to Put an food in the microwave. In general, all the tasks share a large portion of actions (such as exploring cabinets and locations, picking objects etc.). Motivated by how tight relationship of these tasks we come up with three variations. First, `One`, uses one exemplar of an arbitrarily picked task and the other exemplar of the same task as the query. Second, `Both`, uses both exemplars from an arbitrarily picked task. Finally, `All`, uses a total of six exemplars (this is the only variation where we provide more than the standard two examples as in ReAct) corresponding to each task under consideration. Remember, this includes the query task which is always present at the end in the input prompt. **Intuition:** With a very similar action execution trace (such as exploration, picking and placing objects) across tasks, and shared dynamics, LLM agent should be minimally affected by the use of exemplars of a different task.

Variation 4: Exploration Strategy - Optimal As noted before, ReAct does not explain the choice of exemplars used. An important ingredient to the exemplars is the exploration strategy used. In this variation we provide exemplars which serendipitously take the optimal actions (as if the environment were fully observable) and therefore the example plan is the shortest possible. **Intuition:** Exploration strategy exposed in exemplars (that too for the same problem task) should not impact ReAct’s performance if the LLM agent is reasoning instead of retrieval (or pattern matching).

5 RESULTS

In the following sub-sections, we will answer our three RQs using the proposed prompt variations along three dimensions, i.e., the location of the *think* tag, the content of the *think* tag, and the similarity between exemplars and queried problems. All the variations modify the base ReAct prompts and we do not present a cross between variations unless otherwise noted. While the original ReAct experiments were carried out on PaLM (currently decommissioned), we reproduce their results with newer set of models. We use GPT-3.5-Turbo (context window size: 4096 tokens), GPT-3.5-Instruct (context window size: 16,384 tokens), GPT-4 (context window size: 8192 tokens), GPT-4o (context window size: 128,000 tokens), and Claude-Opus (context window size: 200,000 tokens), which are all newer models than those benchmarked in ReAct (Yao et al., 2022b). Note, that despite using newer models, our results shed doubts on the reproducibility and consistency across models of the original paper’s results. As noted, we use the setup in (Yao et al., 2022b) for all our experiments. In AlfWorld, GPT3.5(Turbo, Instruct) results are on 134 instances across six tasks, GPT-4/Claude-Opus on 60 instances (10 for each task) due to cost considerations. In WebShop, GPT3.5(Turbo, Instruct), GPT-4o, LLAMA-3.1-8B (context window size: 128,000 tokens) results are on 500 samples, GPT-4/Claude-Opus are on 50 instances due to cost considerations. In this work, we do not aim to benchmark or analyze any single LLM’s reasoning abilities on decision-making tasks, but rather intend to understand the robustness/brittleness of various LLMs with respect to different components in the ReAct-style prompting method for these tasks.

Table 3: Average Success % of LLM for RQ3 on six AlfWorld tasks. OC: Out of context limit

Model / Prompt	Act	ReAct	RQ3					
			Domain	Instance	Optimal	All	One	Both
GPT-3.5-Turbo	34.3	27.6	1.6	30	20.1	32	28.3	1.6
GPT-3.5-Instruct	44	50.7	47.6	42.5	39.5	OC	17.9	5.2
GPT-4-0314 (Old)	–	23.3	13.3	23.3	50	23.3	16.6	0
GPT-4-0613 (Latest)	70.0	26.7	10.0	20.0	53.3	23.3	20	3.3
Claude-Opus	43.3	56.6	50	46.6	43.3	50	60	6.6

5.1 UTILITY OF INTERLEAVING REASONING TRACE WITH ACTION EXECUTION

Given the claims of ReAct, a practitioner’s expectation is that all the rows would be red as the performance would drop by changing the location of the think tag. From Table 1 (RQ1) note that the exemplar CoT and the anonymized exemplar CoT performs significantly better than base ReAct for all GPT-X family of models. Moreover, the performance dips slightly for Claude-Opus along these variations. From Fig. 3 (larger area represents worse performance), we observe that base ReAct consistently performs worse in most of the tasks. This refutes ReAct’s first claim on the importance of interleaving reasoning trace generation with action execution. Even in the case of the Claude where there is a slight dip in performance, the models seems to be performing at reasonably high success rate which questions the importance of interleaved reasoning and action execution. We omit LLAMA-3.1-8B and GPT-4o (See B.4) for AlfWorld as they achieve zero performance over baselines and all the variations. From Table 2, we find a similar pattern: CoT and Anon. CoT variants perform closely or better than the baseline ReAct. A surprising result consistent in both the domains is the performance of Act baseline (where think tags are absent and actions are generated directly). Act baseline is weaker only for two models GPT-3.5-Instruct, Claude-Opus for both the domains, which further questions the utility of using ReAct style paradigm in the first place.

5.2 UTILITY OF GUIDANCE INFORMATION FOLLOWING think TAG

Recall that reasoning trace guidance pertains to the prospective actions or behaviors an agent should execute (foresight guidance). This type of guidance is more informative compared to other variations, such as hindsight guidance, which focuses on past errors without providing future solution steps, and placebo guidance, which is entirely unrelated to the task. ReAct claims that reasoning trace is crucial for LLM agent performance, which would predict a decline in performance with hindsight guidance and a collapse with placebo guidance. Therefore, a practitioner would expect all the rows to be a dark shade of red. In contrast, our findings in Table 1 indicate that hindsight guidance (Failure, Explanation) actually improves the performance of the GPT family of models. The Claude-Opus model’s performance remains stable with hindsight (Failure) guidance and declines with placebo guidance. Figure 4 illustrates these models’ performance across six AlfWorld tasks and variations, showing that the performance of LLMs either improved or remained consistent when provided with weaker or irrelevant guidance information. This refutes ReAct’s claim that task-specific reasoning trace is the source of LLM agent performance. Our argument that LLM agent’s performance is only slightly affected by the reasoning trace explains the indifference to ordering perturbation as well. If the LLM is not utilizing the reasoning trace for decision making, change in ordering would not affect the agent’s performance. Our arguments hold for the Webshop domain as well, where all of the variants perform closely or better than the baseline ReAct. Finally, contrary to the general perception that better GPT models would improve over reasoning, we find that GPT-4-(Old)’s performance is the worst among GPT-X family further highlighting the brittleness of claims of ReAct. GPT-4-(Latest) performs similarly to GPT-4-(Old), except for the Act baseline which again shows the futility of ReAct prompting. In all our experiment settings, we note that LLMs replicate the exact steps as shown for the examples in the prompts. Hence, they do not output what ReAct claims as think tags if those tags are not present in the original prompt.

5.3 UTILITY OF EXEMPLAR SIMILARITY TO QUERY TASK

Intuitively, the similarity of Domain examples is closest with base ReAct, followed by Instance and Optimal variations. Finally, All contains an overload of information followed by One and

Both which has the same action space but uses different tasks as exemplars. Recall that AlfWorld being a PDDL domain has a shared environment dynamics across all tasks with upto 80% of actions shared across execution traces. While ReAct does not investigate impact of varied exemplars, given the popular usage one expects LLMs to be robust to such changes especially in a common-sense household domain. Table 3 shows the severe brittleness of ReAct based LLM agent to even minor variations (such as Domain, Instance). Specifically, performance of GPT-3.5-Turbo and GPT-4 plummeted for Domain. Claude-Opus which was more robust in RQ1, RQ2, is also impacted severely by Domain, Instance variations. Furthermore, when we do not expose the exploration strategy and only provide Optimal exemplars, the performance of LLM agents further drops (except in GPT4).

Overloading the LLMs with more exemplars All does not impact its performance. We posit, this is because the query-task exemplar is still part of the large input prompt. Among the two exemplars, as provided in ReAct, when one of them is of a different task (One) then the performance significantly reduces for LLMs. When both of the exemplars are of a different task then the performance collapses to single digit success rates for all the models. This is a key result of this work highlighting the severe dependence of LLMs on the similarity of the exemplars to the query task. Through sensitivity analysis using our RQ3 variations we could find parts of the input (the task similarity of the exemplar with query) which is the source of ReAct performance. Essentially, the LLM is mimicking / performing approximate retrieval from the context presented to it. Moreover, our results corroborate the line of research that questions the inability of LLMs to reason or plan (Verma et al., 2024; Valmeekam et al., 2024; Stechly et al., 2024b; Ullman, 2023; Schaeffer et al., 2023; McCoy et al., 2023; Stechly et al., 2024a; Sprague et al., 2024).

The reported success-rate from the ReAct paper Yao et al. (2022b) on the WebShop domain is 40%. Due to the absence of the exact queries used in the paper, we randomly sampled queries from the WebShop dataset comprising 12K records. This approach possibly resulted in the decoupling of any relationship between the exemplars and the queries. Referring to Table 2, it is evident that the performance of the WebShop ReAct agent significantly declined, reaching single digit percentages (as well as other variants). This mirrors the trends observed in the Both variant of the Alfworld in Table 3, further supporting our arguments.

Unrolling and Subtask Similarity We perform additional experiments where the query task is to essentially repeat the task in the exemplar (Unrolling). For instance, in AlfWorld, the exemplar is Put and the query is PutTwo to put two objects at given location. In this case, the LLM has to unroll the given advice and repeat exemplar task execution to solve the query. The success rate of GPT-3.5-Instruct (the best performing GPT model in our experiments) drops down from 52% to 9%. Similarly, we experiment with a Subtask Similarity variation where the exemplar task subsumes execution of the query task. For instance, the Heat task requires the agent to pick and place object in the microwave (which is an instantiation of Put task). One would expect that Heat is a good exemplar for Put, however, the performance of GPT-3.5-instruct model goes from 18% to 0% in this case. These results further underscore the brittleness and the need for instance-specific exemplars in ReAct.

Thought operationalization ability of LLMs Given the free form nature of thought generation and arbitrary nature of thought (about subtask, common-sense next steps etc.), checking whether the generated thoughts are in-fact reasonable is a challenging problem. For completeness, we find that 40% of the times after generation of a *think* tag, subsequent environment action taken by the LLM was invalid (for GPT-3.5-instruct) in AlfWorld. It is much higher (80% for GPT-3.5-Turbo, 90% for Claude-Haiku) for weaker LLM models. This further highlights the inability of LLMs to operationalize its generated thought as also seen in (Roy et al., 2024). From manual inspection we find that the typical thoughts would enlist all possible locations as next locations to visit for most of the tasks. As demonstrated in Section 5.2, the performance of LLMs actually decreases when provided with foresight guidance, as seen with the base ReAct model. A detailed investigation into the validity of the generated reasoning traces is beyond the scope of this work and is suggested as future research.

5.4 DISCUSSION

In this sub-section, we aim to draw insights from our experiments across the three RQs which can be further extended to understanding the limitations of LLMs for planning problems. Specifically, we discuss a) the pitfalls of using ReAct-style prompting for planning domains which could further be exacerbated by approaches that build on top of ReAct framework, and b) scalability and generalizability issues as observed across multiple LLMs.

Pitfalls of ReAct-Style Prompting: Recall, that ReAct claims an improved performance for text-based planning domains, namely - AlfWorld and WebShop, where the presence of a *think* tag acts as guidance for the LLM to generate the next set of actions during the LLM-environment interaction. Through our sensitivity analysis, we dissect each component of ReAct-style prompting in a critical effort to understand the factor that leads to the observed success rates in these domains. With variations on the placement (RQ1) and content (RQ2) of the *think* tag, we eliminate it as the primary cause of any improvement. Furthermore, slight variations in exemplar tasks (RQ3) lead to a stark decline in success rate, clearly indicating the dependence of performance on the highly curated instance-specific examples by domain experts. While newer research in the art of prompting has pointed out the impact of well-curated examples, our work specifically highlights exemplar-query similarity as the cause of ReAct’s performance and rejects contemporary belief that the heavy-lifting of LLM reasoning & planning is done through the *think* tag.

Relevance of ReAct to newer LLMs: ReAct uses the `Act` baseline in their work to showcase improvements due to the presence of the proposed *think* tag. For AlfWorld, ReAct reports 45% success rate for `Act` baseline and 71% success rate for ReAct prompting using the PaLM model. For WebShop, ReAct reports 30.1% success rate for `Act` baseline and 40% success rate for ReAct prompting on PaLM. However, we note from our results on both domains that the `Act` baseline performs much better than ReAct for several LLMs, which questions on the compatibility of ReAct to newer-age LLMs. ReAct performs worse with newer models as compared to the results they report on PaLM, which is currently decommissioned. This observation also questions the contemporary belief that such prompting strategies are generalizable throughout different LLM families, including newer models.

We re-iterate our key result, given any LLM model, the success rates plummet with our RQ3 variations showing a consistent pattern of dependence on the provided examples irrespective of the LLM. Moreover, the performance of all the LLMs remain quite high (if not better) when we vary the location and content of the *think* tags. This highlights the need for higher rigor in agentic LLM experimentation and in-depth evaluation seeking source of improvements. Finally, we highlight our previous discussion on *unrolling*, *subtask-similarity* (discussing the brittleness of perceived reasoning abilities of LLMs) and the inability of LLMs to perform reliable *thought operationalization* as key limitations which exist despite ReAct style prompting.

6 CONCLUSION

ReAct based prompt engineering methods have been claimed to improve planning abilities of Large Language Models. In this study, we critically examine ReAct along three dimensions, informed by its claims and our hypotheses regarding its performance sources. Contrary to ReAct’s claims, our findings reveal that its performance is **neither** due to interleaving reasoning trace and guidance information generation with action execution, **nor** due to the specific nature of the guidance information. Instead, we identify that the true source of LLM performance in sequential decision-making tasks, such as AlfWorld, is the high degree of similarity between exemplar problems (few-shot) and the query task. We also showed that ReAct is susceptible to trivial variations in exemplar prompts (such as with the use of synonyms, or `Unrolling` and `Subtask Similarity` cases). Our findings caution against an uncritical adoption of ReAct-style frameworks for their putative abilities to enhance performance in domains requiring planning. **To conclude, we believe that it will be helpful for practitioners and future works to take these results into account, particularly when designing prompts for text-based decision-making problems, and benefit from avoiding putting any efforts into constructing reasoning traces but rather selecting the right examples for subsequent problems.**

REFERENCES

- 540
541
542 Renat Aksitov, Sobhan Miryoosefi, Zonglin Li, Daliang Li, Sheila Babayan, Kavya Kopparapu,
543 Zachary Fisher, Ruiqi Guo, Sushant Prakash, Pranesh Srinivasan, et al. Rest meets react: Self-
544 improvement for multi-step reasoning llm agent. [arXiv preprint arXiv:2312.10003](#), 2023.
- 545 Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi,
546 Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of thoughts:
547 Solving elaborate problems with large language models. In [Proceedings of the AAAI Conference](#)
548 [on Artificial Intelligence](#), volume 38, pp. 17682–17690, 2024.
- 549 Amrita Bhattacharjee, Raha Moraffah, Joshua Garland, and Huan Liu. Towards llm-guided causal
550 explainability for black-box text classifiers. 2024.
- 551 Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar,
552 Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence:
553 Early experiments with gpt-4. [arXiv preprint arXiv:2303.12712](#), 2023.
- 554 Lluís Castrejon, Thomas Mensink, Howard Zhou, Vittorio Ferrari, Andre Araujo, and Jasper Uijlings.
555 Hammr: Hierarchical multimodal react agents for generic vqa. [arXiv preprint arXiv:2404.05465](#),
556 2024.
- 557 Can Cui, Yunsheng Ma, Xu Cao, Wenqian Ye, and Ziran Wang. Receive, reason, and react: Drive
558 as you say, with large language models in autonomous vehicles. [IEEE Intelligent Transportation](#)
559 [Systems Magazine](#), 2024.
- 560 Richard E Fikes and Nils J Nilsson. Strips: A new approach to the application of theorem proving to
561 problem solving. [Artificial intelligence](#), 2(3-4):189–208, 1971.
- 562 Dayuan Fu, Jianzhao Huang, Siyuan Lu, Guanting Dong, Yejie Wang, Keqing He, and Weiran Xu. Pre-
563 act: Predicting future in react enhances agent’s planning ability. [arXiv preprint arXiv:2402.11534](#),
564 2024.
- 565 Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. Chatgpt outperforms crowd workers for
566 text-annotation tasks. [Proceedings of the National Academy of Sciences](#), 120(30):e2305016120,
567 2023.
- 568 Louie Giray. Prompt engineering with chatgpt: a guide for academic writers. [Annals of biomedical](#)
569 [engineering](#), 51(12):2629–2633, 2023.
- 570 Subbarao Kambhampati, Karthik Valmееkam, Lin Guan, Kaya Stechly, Mudit Verma, Siddhant
571 Bhambri, Lucas Saldyt, and Anil Murthy. Llms can’t plan, but can help planning in llm-modulo
572 frameworks. [arXiv preprint arXiv:2402.01817](#), 2024.
- 573 Jan Kocoń, Igor Cichecki, Oliwier Kaszyca, Mateusz Kochanek, Dominika Szydło, Joanna Baran,
574 Julita Bielanieicz, Marcin Gruza, Arkadiusz Janz, Kamil Kanclerz, et al. Chatgpt: Jack of all
575 trades, master of none. [Information Fusion](#), 99:101861, 2023.
- 576 Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large
577 language models are zero-shot reasoners. [Advances in neural information processing systems](#), 35:
578 22199–22213, 2022.
- 579 Jieyi Long. Large language model guided tree-of-thought. [arXiv preprint arXiv:2305.08291](#), 2023.
- 580 Ggaliwango Marvin, Nakayiza Hellen, Daudi Jjingo, and Joyce Nakatumba-Nabende. Prompt
581 engineering in large language models. In [International Conference on Data Intelligence and](#)
582 [Cognitive Informatics](#), pp. 387–402. Springer, 2023.
- 583 R Thomas McCoy, Shunyu Yao, Dan Friedman, Matthew Hardy, and Thomas L Griffiths. Embers
584 of autoregression: Understanding large language models through the problem they are trained to
585 solve. [arXiv preprint arXiv:2309.13638](#), 2023.
- 586 Sewon Min, Xinxu Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke
587 Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? [arXiv](#)
588 [preprint arXiv:2202.12837](#), 2022.

- 594 OpenReview. ReAct: Synergizing Reasoning and Acting in Language Models. [https://](https://openreview.net/forum?id=WE_vluYUL-X)
595 openreview.net/forum?id=WE_vluYUL-X, 2024.
- 596
- 597 Keqin Peng, Liang Ding, Yancheng Yuan, Xuebo Liu, Min Zhang, Yuanxin Ouyang, and Dacheng
598 Tao. Revisiting demonstration selection strategies in in-context learning. arXiv preprint
599 arXiv:2401.12087, 2024.
- 600 Shamik Roy, Sailik Sengupta, Daniele Bonadiman, Saab Mansour, and Arshit Gupta. Flap: Flow
601 adhering planning with constrained decoding in llms. arXiv preprint arXiv:2403.05766, 2024.
- 602
- 603 Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha.
604 A systematic survey of prompt engineering in large language models: Techniques and applications.
605 arXiv preprint arXiv:2402.07927, 2024.
- 606 Rylan Schaeffer, Kateryna Pistunova, Samar Khanna, Sarthak Consul, and Sanmi Koyejo. Invalid
607 logic, equivalent gains: The bizarreness of reasoning in language model prompting. arXiv preprint
608 arXiv:2307.10573, 2023.
- 609 Sahil Sharma, Aravind Srinivas, and Balaraman Ravindran. Learning to repeat: Fine grained action
610 repetition for deep reinforcement learning. arXiv preprint arXiv:1702.06054, 2017.
- 611
- 612 Noah Shinn, Beck Labash, and Ashwin Gopinath. Reflexion: an autonomous agent with dynamic
613 memory and self-reflection. arXiv preprint arXiv:2303.11366, 2023.
- 614 Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew
615 Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. arXiv
616 preprint arXiv:2010.03768, 2020.
- 617
- 618 Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann
619 Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. To cot or not to cot? chain-
620 of-thought helps mainly on math and symbolic reasoning. arXiv preprint arXiv:2409.12183,
621 2024.
- 622 Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. Chain of thoughtlessness: An
623 analysis of cot in planning. arXiv preprint arXiv:2405.04776, 2024a.
- 624 Kaya Stechly, Karthik Valmeekam, and Subbarao Kambhampati. On the self-verification limitations
625 of large language models on reasoning and planning tasks. arXiv preprint arXiv:2402.08115,
626 2024b.
- 627
- 628 James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale
629 dataset for fact extraction and verification. arXiv preprint arXiv:1803.05355, 2018.
- 630 Tomer Ullman. Large language models fail on trivial alterations to theory-of-mind tasks. arXiv
631 preprint arXiv:2302.08399, 2023.
- 632
- 633 Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the
634 planning abilities of large language models-a critical investigation. Advances in Neural Information
635 Processing Systems, 36, 2024.
- 636
- 637 Mudit Verma, Siddhant Bhambri, and Subbarao Kambhampati. Theory of mind abilities of large
638 language models in human-robot interaction: An illusion? In Companion of the 2024 ACM/IEEE
International Conference on Human-Robot Interaction, pp. 36–45, 2024.
- 639
- 640 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny
641 Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. Advances in
neural information processing systems, 35:24824–24837, 2022.
- 642
- 643 Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu,
644 Ce Liu, Michael Zeng, and Lijuan Wang. Mm-react: Prompting chatgpt for multimodal reasoning
645 and action. arXiv preprint arXiv:2303.11381, 2023.
- 646
- 647 Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov,
and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question
answering. arXiv preprint arXiv:1809.09600, 2018.

648 Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-
649 world web interaction with grounded language agents. Advances in Neural Information Processing
650 Systems, 35:20744–20757, 2022a.

651 Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan
652 Cao. React: Synergizing reasoning and acting in language models. In The Eleventh International
653 Conference on Learning Representations, 2022b.

654 Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan.
655 Tree of thoughts: Deliberate problem solving with large language models. Advances in Neural
656 Information Processing Systems, 36, 2024.

657 Yao Yao, Zuchao Li, and Hai Zhao. Beyond chain-of-thought, effective graph-of-thought reasoning
658 in large language models. arXiv preprint arXiv:2305.16582, 2023.

659 Yunjia Zhang, Jordan Henkel, Avriilia Floratou, Joyce Cahoon, Shaleen Deep, and Jignesh M Patel.
660 Reactable: Enhancing react for table question answering. arXiv preprint arXiv:2310.00815, 2023.

661 Yiming Zhu, Peixian Zhang, Ehsan-Ul Haq, Pan Hui, and Gareth Tyson. Can chatgpt reproduce
662 human-generated labels? a study of social computing tasks. arXiv preprint arXiv:2304.10145,
663 2023.

664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701