

REACTREE: HIERARCHICAL TASK PLANNING WITH DYNAMIC TREE EXPANSION USING LLM AGENT NODES

Anonymous authors

Paper under double-blind review

ABSTRACT

Recent advancements in task planning using large language models (LLMs) have made remarkable progress. However, most existing methods, such as ReAct, face limitations when handling complex, long-horizon tasks due to inefficiencies in processing entire tasks through a single sequential decision-making process. To address these challenges, we propose ReAcTree, a hierarchical task planning method that automatically decomposes complex tasks into manageable subgoals within a tree structure. This tree consists of control flow nodes, which manage the execution order of agent nodes, and agent nodes that reason, act, and expand nodes into subgoals to achieve their goals. To further enhance performance, we introduce memory systems: each agent node retrieves goal-specific, agent-level experiences from episodic memory to use as in-context examples, and all agent nodes share and recall information obtained during task execution via working memory. Experiments on the WAH-NL dataset demonstrate that ReAcTree consistently outperforms ReAct across various LLMs and model sizes. For example, when using Qwen2.5 72B, ReAcTree achieves a goal success rate of 63%, significantly surpassing ReAct’s 24%.

1 INTRODUCTION

In recent years, large language models (LLMs) have emerged as powerful tools for task planning, extending the capabilities of traditional approaches such as task and motion planning and hierarchical reinforcement learning. Early works, including Zero-Shot Planner (Huang et al., 2022a) and Say-Can (Ahn et al., 2022), demonstrated the potential of pre-trained LLMs to generate executable skill sequences from natural language instructions without the need for parameter updates. These methods leverage the general procedural knowledge encoded during pre-training, along with in-context examples, to generate contextually appropriate skill sequences. Further advancements introduced feedback mechanisms, wherein LLMs receive textual observations from the environment following skill execution, enabling more adaptive planning (Huang et al., 2022b). Building on these developments, ReAct (Yao et al., 2023) incorporated reasoning capabilities into LLM-based task planners, significantly improving their overall planning performance.

Despite recent advancements, most existing LLM-based methods, including ReAct, still struggle with complex, long-horizon tasks. One key inefficiency stems from attempting to solve entire tasks using a single sequential decision-making process. Zhou et al. (2023) has shown that LLMs perform more effectively when breaking down complex problems into a series of simpler subproblems. Several LLM-based task planning methods have adopted such decomposition strategies (Wang et al., 2023; Sun et al., 2023; Chen et al., 2024; Wang et al., 2024a; Wong et al., 2023). However, many of these approaches either limit the hierarchy to only two stages or are effective only within predefined domains. Another significant inefficiency arises from the accumulation of long text trajectories encompassing reasoning, actions, and observations. Although providing relevant in-context examples has been shown to significantly enhance performance (Wang et al., 2024b; Rubin et al., 2022), as text trajectories grow, it becomes increasingly difficult to include diverse and detailed examples.

In this paper, we propose ReAcTree, a hierarchical task planning method designed to automatically decompose complex tasks into manageable subgoals. ReAcTree constructs a tree structure with two

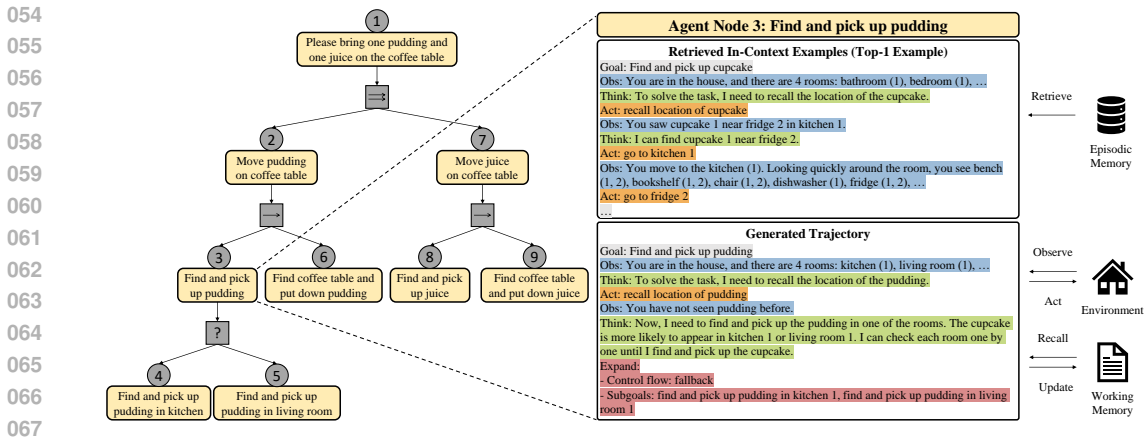


Figure 1: An illustrative example of how ReAcTree generates a tree structure for the natural language instruction: *Please bring one pudding and one juice to the coffee table*. The left side shows the tree structure with agent nodes represented as circles and control flow nodes as squares. Each agent node is annotated with its corresponding natural language goal and execution sequence. The right side presents the text trajectory generated by agent node 3, which includes reasoning, acting, expanding, and retrieval of in-context examples from episodic memory, as well as access to working memory.

primary components: control flow nodes and agent nodes. Control flow nodes, inspired by behavior trees (Colledanchise et al., 2018), are responsible for selecting which agent node to execute at each step. Agent nodes, each functioning as an LLM-based task planner, handle reasoning, acting, and expanding—where expanding involves decomposing goals into subgoals and extending the node into a subtree that includes one control flow node and new agent nodes, each assigned a specific subgoal. To further enhance planning capabilities of ReAcTree, we introduce a memory system. Each agent node retrieves relevant experiences from episodic memory, where past task planning experiences—segmented at the agent level—are stored. This enables the retrieval of experiences closely aligned with the agent node’s current goal. Additionally, all agent nodes utilize working memory to update and recall observations during task execution. By sharing observations across all agent nodes, ReAcTree enables more efficient task planning. Figure 1 illustrates an example outcome produced by ReAcTree.

To evaluate the effectiveness of ReAcTree, we conducted extensive experiments using the WAH-NL dataset (Choi et al., 2024; Puig et al., 2021). The WAH-NL dataset involves solving tasks via natural language commands in household environments, each consisting of multiple rooms and long-horizon tasks composed of several subgoals. We implemented a partially observable setting within the VirtualHome simulator (Puig et al., 2018), where agents operate with limited perceptual information, simulating real-world conditions. Our experimental results demonstrate that ReAcTree consistently outperforms the ReAct baseline across various LLMs. Specifically, ReAcTree attains a 53% goal success rate (GSR) with the LLaMA-3.1 70B model, significantly surpassing ReAct’s 20%. Furthermore, even with the smaller LLaMA-3.1 8B model, ReAcTree achieves a GSR of 30%, outperforming ReAct on the larger LLaMA-3.1 70B model. These results highlight ReAcTree’s superior ability to handle complex tasks through its hierarchical task decomposition. Additionally, our analysis confirms the effectiveness of ReAcTree’s memory systems, with both episodic memory and working memory contributing substantially to the observed performance improvements.

In summary, this paper presents the following contributions: (1) We propose ReAcTree, a novel hierarchical task planning algorithm that automatically decomposes complex tasks into manageable subgoals using a tree structure. Each subgoal is handled by an agent node, which performs LLM-based task planning through reasoning, acting, and expanding—decomposing tasks further into smaller subgoals. (2) We introduce memory systems, including episodic memory and working memory, to enhance the performance of ReAcTree by enabling the retrieval of relevant agent-level experiences and facilitating the sharing of information across agent nodes during task execution. (3) We conduct extensive experiments in a partially observable setting to demonstrate the superiority of ReAcTree over the baseline model, ReAct. Additionally, we will release our code to support future research (anonymous code for review is available at figshare.com/s/97dd86282bd050f66d11).

2 RELATED WORKS

LLM-based Task Planning. Since the inception of LLMs, researchers have continuously explored their emerging capabilities. In particular, their reasoning abilities have recently been applied to decision-making processes for embodied agents, such as robots. Huang et al. (2022a) were among the first to demonstrate that LLMs can infer task procedures for embodied agents without requiring additional training, while SayCan (Ahn et al., 2022) introduced more robust planning by integrating visual affordances from a robot’s perspective. Following these foundational studies, various approaches have evolved. For example, Inner Monologue (Huang et al., 2022b) enhanced procedural inference by allowing agents to revise plans based on textual feedback, such as the success of individual steps, object detection results, and Q&A with humans. Additionally, ReAct (Yao et al., 2023) adopted Chain-of-Thought prompting (Wei et al., 2022), enabling agents to explicitly reason through tasks, leading to improved planning performance.

Hierarchical Task Planning with LLMs. Recently, research has increasingly focused on hierarchical task planning with LLMs to address complex, long-horizon tasks. These approaches often utilize bi-level hierarchies, where an overall plan is refined through next-step decisions, as demonstrated in DEPS (Wang et al., 2023) and AdaPlanner (Sun et al., 2023). Classical planning integrated with LLMs, such as in Ada (Chen et al., 2024), offers flexibility but remains constrained by predefined environments. While these methods enable interactive and adaptive planning, they are limited in hierarchical depth, unlike ReAcTree, which employs a deeper structure that allows for dynamic sub-goal decomposition. MOSAIC (Wang et al., 2024a) proposed a task planner for robots collaborating with humans in cooking. This planner organizes reasoning into a tree structure with LLM agents, where nodes determine whether to define subtasks or seek clarification from the human. However, this study relies on a manually designed structure specific to cooking tasks and lacks generalizability to other domains. In contrast, our approach dynamically generates behavior trees for various tasks in household environments.

Tree Search-Based Planning with LLMs. Another line of research explores tree search-based methods. These approaches construct action/thought trees to look ahead and evaluate multiple future paths to select the best next step. Tree of Thoughts (Yao et al., 2024) and Graph of Thoughts (Besta et al., 2024) demonstrate how systematic tree search improves decision-making by exploring and evaluating reasoning paths. LLM-MCTS (Zhao et al., 2024) extends these ideas to task planning by constructing action trees and applying Monte Carlo Tree Search but relies on ground-truth transition functions, limiting its applicability to real-world scenarios. Tree-Planner (Hu et al., 2023) constructs action trees, executes actions, and observes outcomes, avoiding explicit state prediction. However, it assumes reversible actions (e.g., *pick-place*), which are not feasible for irreversible actions like slicing. While these methods focus on constructing action or thought trees to search for the best trajectory, ReAcTree takes a fundamentally different approach by building an LLM agent tree. It divides complex goals into manageable subgoals, dynamically assigns specialized agent nodes, and determines their execution order using control flow strategies such as sequence and fallback nodes.

3 PRELIMINARIES

Problem Formulation. We consider the task planning problem as a sequential decision-making problem aimed at achieving a goal g expressed in natural language. At each time step t , the agent has access the context $c_t = (o_1, a_1, o_2, a_2, \dots, a_{t-1}, o_t)$, where o_i and a_i represent the observation and action at each previous time step i , respectively. The objective of agent is to generate the next appropriate action a_t based on the context c_t , with the aim of eventually achieving the goal g .

ReAct (Yao et al., 2023). ReAct is a representative method that leverages a pre-trained LLM, p_{LLM} , to solve the task planning problem by interleaving reasoning and action execution. The LLM defines the action policy as follows: $a_t \sim p_{LLM}(\cdot|P, g, c_t)$, where $P = (P_{sys}, P_{ic})$ is the initial prompt, composed of a system prompt, P_{sys} , and in-context examples, P_{ic} . The key idea of ReAct is to use the augmented action space, $\hat{\mathcal{A}}_t = \mathcal{A}_t \cup \mathcal{L}$, where \mathcal{A}_t is the set of executable skills available at time t , and \mathcal{L} is the language space representing reasoning steps or thoughts. If $a_t \in \mathcal{A}_t$, the agent executes the action and obtains a text observation from the environment. If $a_t \in \mathcal{L}$, it is called a thought or reasoning trace, which aids in the logical inference of the LLM. In this case, the agent does not receive a new observation from the environment, i.e., $o_{t+1} = \phi$.

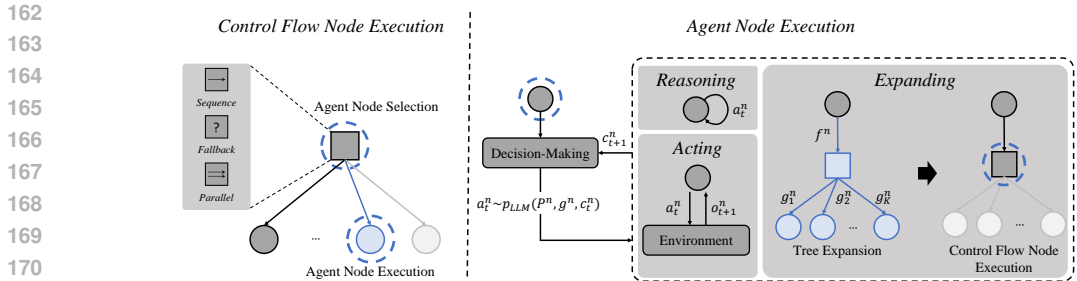


Figure 2: Illustration of control flow node execution and agent node execution in ReAcTree.

4 REACTREE

In this section, we introduce ReAcTree, a hierarchical task planning algorithm designed to efficiently manage complex tasks by decomposing them into subgoals using a tree structure. The tree is composed of two types of nodes: control flow nodes and agent nodes. Control flow nodes determine which agent node should be executed next. Each agent node operates as an LLM-based task planner, with its goal expressed in natural language. A key feature of ReAcTree is the augmentation of each agent node’s action space to include not only reasoning and acting but also expanding, which allows the decomposition of its goal into subgoals and the creation of a subtree. This subtree consists of a child control flow node and grandchild agent nodes, each assigned a specific subgoal. Figure 2 illustrates the node execution process for both control flow nodes and agent nodes. The details are described in Section 4.1.

We introduce a memory system designed to enhance the performance of ReAcTree. Episodic memory is employed to construct in-context examples for each agent node. Each node retrieves relevant, agent-level experiences from episodic memory. Additionally, working memory stores information gathered by agent nodes through their interactions with the environment during task execution. This working memory is shared among all agent nodes, enabling them to access and utilize the stored information as needed. Further details are provided in Section 4.2.

4.1 REACTREE ALGORITHM

ReAcTree is an LLM-based hierarchical task planning algorithm that generates a tree structure T , composed of control flow nodes and agent nodes, to achieve a task goal g expressed in natural language. In the following, we offer a detailed explanation of the execution process for both control flow nodes and agent nodes, as well as the overall ReAcTree algorithm.

Control Flow Nodes. Control flow nodes, inspired by behavior trees (Colledanchise et al., 2018), manage the selection and execution of agent nodes within the tree. Each control flow node has child nodes, which are agent nodes assigned specific natural language goals. Executing a control flow node involves executing its child nodes sequentially. Each child node reports its execution status (success or failure) to its parent, the control flow node. Based on the status of its child nodes, the control flow node decides whether to proceed with the next child node or to return its own status to its parent node. The specific behavior of the control flow node depends on its type.

ReAcTree employs three types of control flow nodes. The first is the *sequence node* (\rightarrow), which executes its child nodes in order. It returns success if all child nodes succeed; however, if any child node fails, the sequence node returns failure. The second type is the *fallback node* ($?$), which also executes its child nodes sequentially but returns success as soon as any child node succeeds. If none of the child nodes succeed, it returns failure. The third type is the *parallel node* (\Rightarrow), a variation of the traditional parallel node concept. While the traditional definition of a parallel node involves executing child nodes simultaneously, in ReAcTree, simultaneous execution is not possible. Instead, the parallel node executes its child nodes independently, regardless of their individual success or failure. After all nodes are executed, the outcomes are aggregated according to a predefined policy to determine the overall success or failure. This node is particularly useful in tasks such as household chores, where multiple subgoals, like picking and placing objects, need to be executed sequentially without interruption, yet their success or failure does not immediately halt the overall task.

Agent Nodes. Each agent node operates as an LLM-based task planner with a specific natural language goal, responsible for making sequential decisions to achieve that goal. These decisions include *acting*, *reasoning*, and *expanding*. Similar to the ReAct framework, *acting* refers to executing actions and receiving feedback in the form of textual observations, while *reasoning* enables logical inference. Additionally, *expanding* decomposes the agent node’s current goal into subgoals and combines them with a specific control flow strategy to further expand the tree structure.

An agent node n is first initialized with a goal g^n . Executing the agent node involves sequential decision-making, where the agent returns its status (whether the goal was achieved) after execution. More specifically, when executing an agent node, an initial prompt $P^n = (P_{sys}, P_{ic}^n)$ is constructed, where P_{sys} is the system prompt, and P_{ic}^n consists of in-context examples specific to agent node n . At each time step t , the agent node accesses its context $c_t^n = (o_1^n, a_1^n, o_2^n, a_2^n, \dots, a_{t-1}^n, o_t^n)$, where o_i^n and a_i^n represent the observation and action at each previous time step i . The action policy is then defined using the pre-trained LLM, P_{LLM} , as: $a_t \sim p_{LLM}(\cdot | P^n, g^n, c_t^n)$. A key feature of ReAcTree is its extended action space, $\hat{\mathcal{A}}_t^n = \mathcal{A}_t^n \cup \mathcal{L} \cup \mathcal{E}$, where \mathcal{A}_t^n represents the set of executable skills at time t (e.g., *move*, *pick*, *turn on*); \mathcal{L} is the language space, used for generating self-reasoning text and specifying subgoals in natural language; and $\mathcal{E} = \mathcal{F} \times \mathcal{L}$ is the expand space, where \mathcal{F} represents the set of control flow types.

If the action $a_t^n \in \mathcal{A}_t^n$ or $a_t^n \in \mathcal{L}$, the agent operates as in the ReAct framework, either performing actions or engaging in reasoning. However, if $a_t^n \in \mathcal{E}$, the agent expands the tree structure by adding a control flow node and handing over execution to it. In this case, the action is represented as $a_t^n = (f^n, [g_1^n, \dots, g_K^n])$, where f^n is the control flow type and g_i^n are the subgoals expressed in natural language. A control flow node n_f , with type f^n , is added as a child of node n , and agent nodes n_i are initialized with their corresponding subgoals g_i^n and added as children of n_f . The agent then waits for n_f to complete execution. The agent node terminates when one of the following occurs: generating the action *done* (resulting in success), generating action *failure* or reaching the maximum decision count (both resulting in failure), or completing the execution of the control flow node (returning success or failure based on its result).

Overall Algorithm. The overall process of ReAcTree is outlined in Algorithm 1. It begins by initializing the root agent node n with the goal g (line 1). The agent node is then executed until meets a termination condition TC (line 2). If TC is *done*, it indicates that the agent has successfully achieved the goal, and the agent node returns *Success* status (lines 3-4). If TC is *failure* or the maximum decision count is reached, it indicates failure, and the agent node returns *Failure* status (lines 5-6). If TC is *Expand*, the agent has decided to decompose the goal into subgoals. The expansion information $(f^n, [g_1^n, \dots, g_K^n])$ is retrieved (line 8), where f^n is the control flow type and g_i^n are the subgoals. A control flow node n_f is initialized (line 9) and added as a child of the current agent node n (line 10). For each subgoal, a new agent node n_i is initialized (line 12) and added as a child of the control flow node n_f (line 13). The control flow node n_f is then executed (line 15), where it selects one of its child agent nodes based on its control flow strategy. The selected agent node is then executed, and its status is returned to the control flow node. This process repeats until the control flow node completes execution, after which it returns its resulting status (success or failure).

Algorithm 1 ReAcTree Algorithm

Input: Natural language goal g

- 1: $n \leftarrow \text{INITAGENTNODE}(g)$
- 2: $TC \leftarrow \text{EXECAGENTNODE}(n)$
- 3: **if** TC is *done* **then**
- 4: **return** Success
- 5: **else if** TC is *failure* or max decision **then**
- 6: **return** Failure
- 7: **else if** TC is *Expand* **then**
- 8: $(f^n, [g_1^n, \dots, g_K^n]) \leftarrow \text{EXPANDINFO}$
- 9: $n_f \leftarrow \text{INITCTRLFLOWNODE}(f^n)$
- 10: $\text{ADDCHILD}(n, n_f)$
- 11: **for** $i = 1$ to K **do**
- 12: $n_i \leftarrow \text{INITAGENTNODE}(g_i^n)$
- 13: $\text{ADDCHILD}(n_f, n_i)$
- 14: **end for**
- 15: **return** $\text{EXECCTRLFLOWNODE}(n_f)$
- 16: **end if**

4.2 MEMORY SYSTEMS

To enhance the performance of ReAcTree, we introduce two complementary memory systems: *episodic memory* and *working memory*. *Episodic memory* is used to retrieve relevant past agent-level experiences and incorporate them as in-context examples before each agent node begins its

270 decision-making process. On the other hand, working memory is designed to share key observa-
 271 tions, such as the latest location of movable objects, across agent nodes during task execution. In
 272 the following, we provide a detailed explanation of how each memory is utilized and integrated with
 273 agent nodes of ReAcTree.

274 **Episodic Memory.** Episodic memory, M_{ep} , stores the agent-level experiences of all ReAcTree
 275 agent nodes involve in successfully completing tasks. The agent-level experience of an agent
 276 node e , with a goal sentence g^e and a final time step is T , is defined as (t^e, v^e, s^e) . Here,
 277 $t^e = (g^e, o_1^e, a_1^e, \dots, o_T^e, a_T^e)$ represents the full text trajectory, where o_t^e and a_t^e denote the ob-
 278 servation and action at each time step t . The vector $v^e = f_{sen}(g^e)$ is the goal embedding generated
 279 using the sentence embedding model f_{sen} . Finally, s^e indicates the termination state of the agent
 280 node, categorized as *success*, *failure*, or *expand*.

281 Before an agent node begins its decision-making process, it retrieves in-context examples by com-
 282 paring its goal g^n to the stored goals in episodic memory using cosine similarity. Specifically,
 283 the agent embeds its goal as $v^n = f_{sen}(g^n)$ and computes the similarity with stored embeddings:
 284 $sim(v^n, v^e) = v^n \cdot v^e / (|v^n| |v^e|), \forall v^e \in M_{ep}$. Based on the similarity scores, the system retrieves
 285 the top k examples until a predefined token limit is reached. To handle cases where multiple stored
 286 experiences yield identical similarity scores, the termination state s is used to break ties. When sim-
 287 ilarity scores are tied, examples are sampled uniformly across termination states, *success*, *failure*, or
 288 *expand*. This mechanism promotes diversity in the retrieved in-context examples.

289 Since ReAcTree is designed to decompose complex goals into manageable subgoals, it generates
 290 concise trajectories tailored to specific tasks, unlike the monolithic trajectories used by ReAct. For
 291 instance, while ReAct stores a single trajectory for *Bring one pudding and onne juice to the coffee*
 292 *table*, ReAcTree breaks this down into subgoals like *find and pick up pudding in kitchen*, producing
 293 shorter and more specific examples. This enables ReAcTree to retrieve highly relevant and focused
 294 trajectories from episodic memory, enhancing its task-planning effectiveness.

295 **Working Memory.** Working memory serves as a shared repository for storing and recalling key
 296 observations during task execution. In this paper, working memory focuses on tracking the latest
 297 locations of movable objects to minimize redundant interactions with the environment and mitigate
 298 potential hallucinations by providing accurate, environment-specific data.

299 Working memory is integrated into agent nodes of ReAcTree through two key mechanisms. First, the
 300 executable skill set \mathcal{A}_t^n is augmented with special actions like *recall location of <movable object>*,
 301 which are predefined for all movable objects in the environment, enabling agents to retrieve stored
 302 object locations directly from working memory instead of interacting with the environment. Second,
 303 working memory is automatically updated whenever an agent interacts with the environment and
 304 detects movable objects. For instance, if an agent opens a fridge and observes juice, working memory
 305 updates the location of juice as near fridge for future use. Such interactions can also be viewed as an
 306 extension of how tool usage is integrated into language models, as discussed in (Schick et al., 2024).

308 5 EXPERIMENTS

309 5.1 EXPERIMENTAL SETUP

312 **Datasets and Simulators.** We primarily evaluate LLM-based task planning using the WAH-NL
 313 dataset within the VirtualHome simulator, as introduced in LoTa-Bench (Choi et al., 2024). The
 314 WAH-NL dataset, derived from the Watch-and-Help dataset (Puig et al., 2021), was originally de-
 315 signed for human-agent collaboration. It has been adapted for autonomous agents, with natural lan-
 316 guage commands collected via crowdsourcing. Each data instance comprises a natural language
 317 instruction, an initial environment setup, and a goal condition. The dataset includes 250 tasks in the
 318 training set and 100 tasks in the test set, covering five task categories: *Setup a dinner table*, *Put*
 319 *groceries*, *Prepare a meal*, *Wash dishes*, and *Prepare snacks*. Each task involves multiple subgoals,
 320 making WAH-NL well-suited for evaluating task planning in complex, long-horizon scenarios.

321 All experiments are conducted in the VirtualHome simulator, which provides a simulated household
 322 environment where agents can perform various actions such as *pick up*, *open*, *close*, *turn on*, *put*
 323 *down*, and *go to*. Unlike the fully observable setting in LoTa-Bench, we implement a partially ob-
 servable setting in VirtualHome to simulate real-world conditions. In this configuration, the agent

has access to limited observations. Specifically, objects are identified by both their class name and instance (e.g., *cup 1*, *cup 2*). After executing an action, the agent receives a text-based observation reflecting its surroundings. For instance, after performing a *go to room* action, the agent can observe all receptacles (e.g., tables, shelves) within the room. If the action is *go to object*, the agent observes nearby visible objects, provided they are not inside closed receptacles. For actions like *pick up* or *open*, the agent receives feedback indicating whether the action was successful or not. A more detailed implementation of the partially observable setting is described in Appendix A, and we will release our code and modified simulator for reproducibility (anonymous code for review is available at figshare.com/s/97dd86282bd050f66d11).

Evaluations. To assess the effectiveness of ReAcTree, we compare it against ReAct, with both utilizing the same underlying LLMs. We evaluate performance using the goal success rate (GSR), defined as the percentage of tasks in which the agent successfully achieves the given task goal, and the subgoal success rate (SSR), which is the ratio of successfully completed subgoals to the total number of subgoals. **Both methods were evaluated under the same maximum decision count, set to a sufficiently large value of 199 for all experiments, ensuring fair comparisons.** All results are averaged across the entire test set.

Episodic Memory Construction. To construct the episodic memory, we first collect human-annotated text trajectories for WAH-NL in both ReAct and ReAcTree formats. Specifically, one random task from each task category is selected for trajectory collection, resulting in five tasks for WAH-NL. Subsequently, we run both ReAct and ReAcTree on the training set using the LLaMA-3 70B model (Dubey et al., 2024). Only the trajectories of tasks that are successfully completed are added to the episodic memory. To encode the agent’s goals, we use Sentence BERT (Reimers & Gurevych, 2019) for sentence embeddings.

5.2 MAIN RESULTS

Table 1 summarizes the comparison between ReAcTree and ReAct across various LLMs, including LLaMA 3, LLaMA 3.1 (Dubey et al., 2024), Qwen2, Qwen2.5 (Yang et al., 2024), Mistral (Jiang et al., 2023), Gemma (Team et al., 2024), and GPT-4o (see Appendix B for the complete list). Performance was evaluated using GSR and SSR metrics, with the primary results presented in Table 1 and additional results provided in Appendix D. Since direct access to log probabilities is not available for GPT-4o, its implementation was slightly modified. **These details are in Appendix C.**

Overall, ReAcTree consistently outperforms ReAct across all model types and sizes. For instance, with the Qwen2.5 72B model, ReAcTree achieves a GSR of 63.00%, compared to ReAct’s GSR of 24.00%. This consistent performance improvement highlights the effectiveness of ReAcTree’s task decomposition approach. The results clearly demonstrate the advantage of breaking tasks into manageable subgoals, enabling more efficient task completion, particularly in complex, long-horizon tasks. **In terms of computational cost with this model, ReAcTree incurs a higher average decision steps (75.00) compared to ReAct (58.08) for tasks where both methods succeed. However, its hierarchical design resets the input prompt for each agent node, mitigating computational overhead.**

Table 1: Performance comparison of ReAct and ReAcTree across different models and sizes. WM (✓) indicates the use of working memory for task planning, while (✗) denotes its absence.

Metric	Method	WM	LLaMA 3.1		Qwen2.5		Mistral	GPT-4o
			8B	70B	7B	72B	7B	
GSR (%)	ReAct	✗	10.00	23.00	6.00	20.00	3.00	-
		✓	15.00	20.00	13.00	24.00	6.00	33.00
	ReAcTree	✗	17.00	30.00	12.00	51.00	10.00	-
		✓	30.00	53.00	35.00	63.00	20.00	46.00
SSR (%)	ReAct	✗	29.45	49.35	24.55	45.13	21.72	-
		✓	42.27	52.87	35.77	48.43	31.03	57.30
	ReAcTree	✗	53.28	62.83	42.43	70.95	37.43	-
		✓	60.60	73.43	58.80	79.37	43.83	62.77

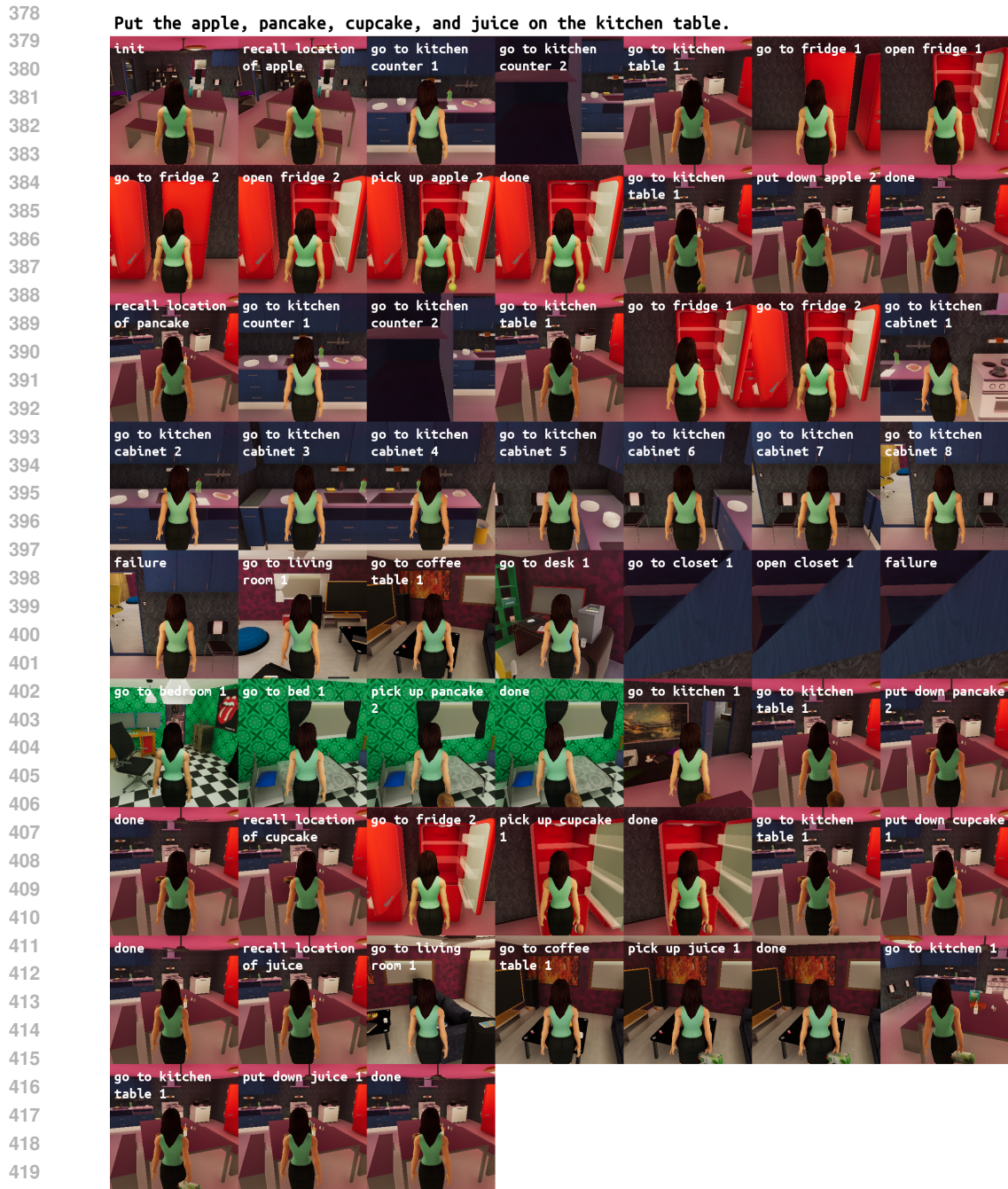


Figure 3: Success case of ReAcTree on the WAH-NL dataset using LLaMA 3 70B.

Moreover, the results suggest that ReAcTree with smaller models can perform even better than larger models running ReAct. For instance, ReAcTree using the LLaMA 3.1 8B model achieves a GSR of 30.00%, outperforming ReAct’s GSR of 20.00% when using the significantly larger LLaMA 3.1 70B model. This performance improvement can be attributed to ReAcTree’s ability to decompose tasks into simpler subproblems, allowing smaller models to efficiently manage less complex components while retrieving more relevant in-context examples. By focusing on these smaller, more manageable tasks, ReAcTree effectively compensates for the limitations of smaller models.

We also present qualitative results comparing failed trajectories in ReAct with successful ones in ReAcTree using the LLaMA 3 70B model. The natural language instruction for the task is, *Put the*

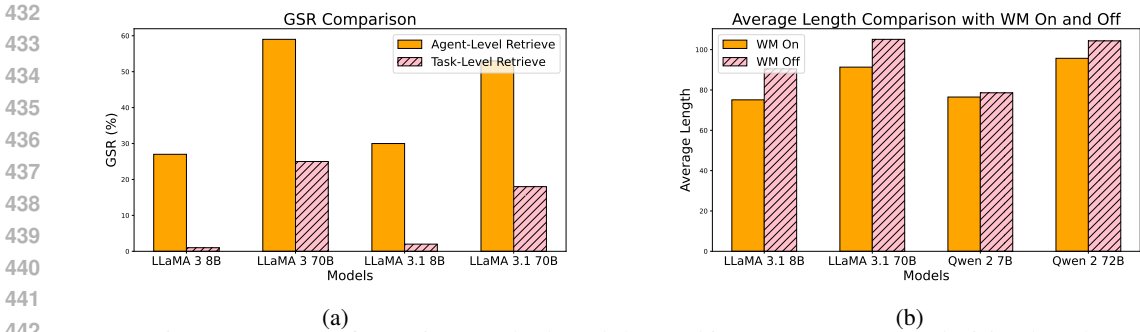


Figure 4: (a) GSR for retrieve methods and (b) Working memory average decision length.

apple, pancake, cupcake, and juice on the kitchen table, which requires locating and moving multiple objects. ReAcTree breaks down the instruction into specific subgoals, such as *move the apple to the kitchen table*, *move the pancake to the kitchen table*, *move the cupcake to the kitchen table*, and *move the juice to the kitchen table*, using a persistent sequence node. Notably, during the search for the pancake, ReAcTree utilizes a fallback node to explore different rooms. In contrast, ReAct struggles to locate the pancake, as it only searches the initial room, the kitchen. Furthermore, ReAct **misidentifies** a pudding as the pancake, **and redundantly moves an already relocated apple**. Figure 3 illustrates the successful trajectories from ReAcTree, while Figure 5 in Appendix G.1 shows the failed trajectories from ReAct. The full text trajectories for both ReAcTree and ReAct are provided in List 11 and List 12 in Appendix H.1, respectively. [For a more detailed analysis of ReAcTree’s failure cases, including examples and common error patterns, please refer to Appendix E.](#)

5.3 EPISODIC MEMORY

In this section, we investigate the impact of episodic memory on the performance of ReAcTree. ReAcTree utilizes episodic memory by retrieving similar past experiences at the agent level, providing in-context examples to each agent. This allows agents to reference relevant trajectories. To evaluate the effectiveness of this approach, we compare it with task-level example retrieval.

In the agent-level retrieval method, each agent independently retrieves the most contextually similar experiences from episodic memory, irrespective of the task. This enables each agent to draw from a diverse set of agent-specific experiences. In contrast, task-level retrieval selects the most similar past task and uses the entire sequence of trajectories from all agents involved in solving that task as the in-context example. This approach gives the agent access to a complete set of trajectories used to solve a previous task that closely resembles the current one.

The performance results are shown in Figure 4a. Our findings demonstrate that agent-level retrieval yields better performance, especially as model size decreases. Smaller models benefit more from decomposing complex tasks into simpler subgoals and retrieving contextually similar examples for each subgoal, resulting in more efficient problem-solving. By contrast, task-level retrieval, which provides a broader set of trajectories, may offer a wider contextual understanding but can be less effective for fine-tuned decision-making at the subgoal level.

5.4 WORKING MEMORY

In this section, we analyze the impact of the working memory component on the performance of both ReAcTree and ReAct models. As shown in Table 1, incorporating working memory consistently leads to performance improvements across various LLMs and model sizes. This enhancement is observed not only in ReAcTree but also in the ReAct baseline. In most cases, the performance improvement occurs because the agent retrieves stored object locations from working memory using the *recall location of* action, rather than generating locations through text. This finding aligns with previous research on tool usage in LLMs (Schick et al., 2024), which also demonstrated similar benefits.

In ReAcTree, working memory additionally contributes to efficiency improvements through information sharing between agent nodes. When one agent node identifies the location of an object or

486 gathers relevant information, it updates this data in the working memory. Other agent nodes can then
 487 access this information, eliminating redundant searches and actions. This inter-node communication
 488 streamlines decision-making and accelerates task completion.

489 To quantify the efficiency gains provided by working memory, Figure 4b illustrates the average
 490 number of decision-making steps required to complete tasks that are commonly solved both with
 491 and without working memory. The figure shows that across all models, using working memory
 492 significantly reduces the number of steps needed to achieve task success. This reduction indicates
 493 that agents can plan more efficiently and execute tasks with fewer unnecessary actions by leveraging
 494 the shared information stored in working memory.

496 5.5 EXPERIMENTS ON ALFRED

498 **Datasets and Simulators.** We further extended our experiments to include the ALFRED
 499 dataset (Shridhar et al., 2020), which, similar to WAH-NL, involves task planning based on LLM
 500 in a household environment setting. This dataset provides NL instructions, an initial state of the
 501 environment, and a goal condition related to various household tasks. The task types in this experi-
 502 ment are similar to those used in LoTa-Bench (Choi et al., 2024). We used the AI2-THOR simulator,
 503 which corresponds to the ALFRED dataset.

504 **Evaluations.** The ALFRED is consists of three sets: *train*, *valid-seen*, *valid-unseen*. The planning
 505 performance of both ReAct and ReAcTree models was evaluated using the *valid-seen* set. We evalu-
 506 ated using only 30% of the *valid-seen* set, which consists of 208 task trajectories. We measured plan-
 507 ning performance using the goal success rate (GSR) for ALFRED. GSR is determined by whether
 508 the goal condition of the dataset matches the final state of the simulator after completing the plan-
 509 ning.

510 **Experimental Results.** Table 2 summarizes the comparison between ReAct and ReAcTree
 511 across different LLM models. The proposed
 512 ReAcTree method demonstrates a 2.88% per-
 513 formance improvement over ReAct in both the
 514 LLaMA 3 70B and Qwen2 72B models. For
 515 smaller models, ReAcTree also outperforms
 516 ReAct by 2.93% in the LLaMA 3 8B model.
 517 However, in the Qwen2 7B model, ReAct sur-
 518 passes ReAcTree with a 3.4% advantage. Overall, except for the Qwen2 7B model, ReAcTree con-
 519 sistentlly improves task planning performance. Visualizations and full text trajectories for a specific
 520 ALFRED task where ReAcTree succeeds and ReAct fails are provided in Appendix G.2 and H.2,
 521 respectively. Please refer to Appendix F for details on the experiments with the ALFRED.

Table 2: Goal success rate comparison of ReAct and ReAcTree on ALFRED with working memory.

Method	LLaMA 3		Qwen2	
	8B	70B	7B	72B
ReAct	5.77	16.35	7.73	14.90
ReAcTree	8.70	19.23	4.33	19.71

523 6 CONCLUSION

525 In this paper, we presented ReAcTree, a hierarchical task planning method that automatically de-
 526 composes complex tasks into subgoals. Our approach uses a tree structure that integrates control
 527 flow nodes for managing tasks and agent nodes to perform sequential decision-making through an
 528 extended action space, which includes not only *reasoning* and *acting*, but also *expanding*. Addition-
 529 ally, we introduced memory systems, with episodic memory retrieving agent-level experiences and
 530 working memory sharing observations between nodes. Our experiments on the WAH-NL dataset
 531 show that ReAcTree consistently outperforms the ReAct baseline, achieving 35% and 63% success
 532 rates with Qwen2.5 7B and 72B, respectively, compared to 24% with ReAct using Qwen2.5 72B.
 533 We also demonstrated the effectiveness of our memory systems.

534 One limitation of our approach is that the observations obtained after executing actions rely on
 535 ground truth from a simulator, with pre-specified formats for certain action types. These observa-
 536 tions may not always be optimal for solving the current subgoal. Additionally, while ReAcTree
 537 shows significant improvement with smaller models, further enhancements are still needed. For fu-
 538 ture work, we will explore methods to gather more relevant observations from the environment after
 539 action execution. We will also focus on improving the performance of ReAcTree with smaller mod-
 els.

REFERENCES

- 540
541
542 Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea
543 Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say:
544 Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.
- 545
546 Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gian-
547 inazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. Graph of
548 thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI*
549 *Conference on Artificial Intelligence*, volume 38, pp. 17682–17690, 2024.
- 550
551 Xinglin Chen, Yishuai Cai, Yunxin Mao, Minglong Li, Wenjing Yang, Weixia Xu, and Ji Wang.
552 Integrating intent understanding and optimal behavior planning for behavior tree generation from
553 human instructions. *arXiv preprint arXiv:2405.07474*, 2024.
- 554
555 Jae-Woo Choi, Youngwoo Yoon, Hyobin Ong, Jaehong Kim, and Minsu Jang. Lota-bench: Bench-
556 marking language-oriented task planners for embodied agents. In *The Twelfth International Con-*
557 *ference on Learning Representations*, 2024.
- 558
559 Michele Colledanchise, Ramviyas Parasuraman, and Petter Ögren. Learning of behavior trees for
560 autonomous agents. *IEEE Transactions on Games*, 11(2):183–189, 2018.
- 561
562 Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha
563 Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models.
564 *arXiv preprint arXiv:2407.21783*, 2024.
- 565
566 Mengkang Hu, Yao Mu, Xinmiao Yu, Mingyu Ding, Shiguang Wu, Wenqi Shao, Qiguang Chen,
567 Bin Wang, Yu Qiao, and Ping Luo. Tree-planner: Efficient close-loop task planning with large
568 language models. *arXiv preprint arXiv:2310.08582*, 2023.
- 569
570 Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot
571 planners: Extracting actionable knowledge for embodied agents. In *International conference on*
572 *machine learning*, pp. 9118–9147. PMLR, 2022a.
- 573
574 Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan
575 Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through
576 planning with language models. *arXiv preprint arXiv:2207.05608*, 2022b.
- 577
578 Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot,
579 Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al.
580 Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- 581
582 Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Tor-
583 ralba. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE*
584 *Conference on Computer Vision and Pattern Recognition*, pp. 8494–8502, 2018.
- 585
586 Xavier Puig, Tianmin Shu, Shuang Li, Zilin Wang, Yuan-Hong Liao, Joshua B. Tenenbaum, Sanja
587 Fidler, and Antonio Torralba. Watch-and-help: A challenge for social perception and human-ai
588 collaboration. In *International Conference on Learning Representations*, 2021.
- 589
590 Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-
591 networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*
592 *Processing*, 11 2019.
- 593
594 Ohad Rubin, Jonathan Herzig, and Jonathan Berant. Learning to retrieve prompts for in-context
595 learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Associa-*
596 *tion for Computational Linguistics: Human Language Technologies*, pp. 2655–2671, 2022.
- 597
598 Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke
599 Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach
600 themselves to use tools. *Advances in Neural Information Processing Systems*, 36, 2024.

- 594 Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi,
595 Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions
596 for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern
597 recognition*, pp. 10740–10749, 2020.
- 598
599 Haotian Sun, Yuchen Zhuang, Ling kai Kong, Bo Dai, and Chao Zhang. Adaplaner: adaptive plan-
600 ning from feedback with language models. In *Proceedings of the 37th International Conference
601 on Neural Information Processing Systems*, pp. 58202–58245, 2023.
- 602 Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya
603 Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open
604 models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.
- 605 Huaxiaoyue Wang, Kushal Kedia, Juntao Ren, Rahma Abdullah, Atiksh Bhardwaj, Angela Chao,
606 Kelly Y Chen, Nathaniel Chin, Prithwish Dan, Xinyi Fan, Gonzalo Gonzalez-Pumariega, Aditya
607 Kompella, Maximus Adrian Pace, Yash Sharma, Xiangwan Sun, Neha Sunkara, and Sanjiban
608 Choudhury. Mosaic: A modular system for assistive and interactive cooking, 2024a.
- 609
610 Liang Wang, Nan Yang, and Furu Wei. Learning to retrieve in-context examples for large language
611 models. In *Proceedings of the 18th Conference of the European Chapter of the Association for
612 Computational Linguistics (Volume 1: Long Papers)*, pp. 1752–1767, 2024b.
- 613 Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, Yitao Liang, and Team Craft-
614 Jarvis. Describe, explain, plan and select: interactive planning with large language models enables
615 open-world multi-task agents. In *Proceedings of the 37th International Conference on Neural In-
616 formation Processing Systems*, pp. 34153–34189, 2023.
- 617
618 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny
619 Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in
620 neural information processing systems*, 35:24824–24837, 2022.
- 621 Lionel Wong, Jiayuan Mao, Pratyusha Sharma, Zachary S Siegel, Jiahai Feng, Noa Korneev,
622 Joshua B Tenenbaum, and Jacob Andreas. Learning adaptive planning representations with natu-
623 ral language guidance. *arXiv preprint arXiv:2312.08566*, 2023.
- 624
625 An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li,
626 Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang,
627 Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai,
628 Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng
629 Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai
630 Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan
631 Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang
632 Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2
633 technical report. *arXiv preprint arXiv:2407.10671*, 2024.
- 634
635 Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan
636 Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International
637 Conference on Learning Representations*, 2023.
- 638
639 Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik
640 Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances
641 in Neural Information Processing Systems*, 36, 2024.
- 642
643 Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for
644 large-scale task planning. *Advances in Neural Information Processing Systems*, 36, 2024.
- 645
646 Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuur-
647 mans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. Least-to-most prompting enables complex
reasoning in large language models. In *The Eleventh International Conference on Learning Rep-
resentations*, 2023.

A IMPLEMENTATION DETAILS FOR PARTIALLY OBSERVABLE SETTINGS

As outlined in Section 5, we adapt LoTa-bench (Choi et al., 2024) to support partial observability. Oracle-generated partial observation feedback signals are incorporated into the context information provided to the LLM. Tables 3 and 4 present detailed example trajectories for each action type under partial observation. In this setup, the agent is aware only of the receptacles in the current room and can only see the objects visible within its ego-centric view. VirtualHome is a multi-room apartment simulator, so it has a “*go to room*” action type. On the other hand, AI2-THOR is a single-room studio apartment simulator, so it does not use the “*go to room*” action type because there is no need to move between rooms. Table 3 illustrates an example of partial observation feedback from the VirtualHome simulator, while Table 4 provides a similar example from the AI2-THOR simulator.

Table 3: Action types and sample text observations in the VirtualHome simulator.

Action Type	Action	Text Observation
<i>initialization</i>	-	You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen (1), living room (1). You are in the middle of a bathroom (1). Looking quickly around the room, you see bathroom cabinet (1), bathroom counter (1), faucet (1), sink (1), toilet (1), towel rack (1), washing machine (1).
<i>go to room</i>	go to kitchen 1	You move to the kitchen (1). Looking quickly around the room, you see chair (1, 2, 3, 4), dishwasher (1), faucet (1), fridge (1), kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 3), kitchen table (1), microwave oven (1), nightstand (1, 2), sink (1), sofa (1), stove (1), toaster (1).
<i>go to receptacle</i>	go to kitchen table 1	You arrive at the kitchen table (1). You see apple (2), book (1), bowl (2, 3), cereal (1), chair (1, 2, 3, 4), chocolate syrup (1), condiment bottle (1, 2), condiment shaker (1, 2), cupcake (1, 2), cutlery fork (2, 3, 4), cutlery knife (3, 5, 6), juice (1), kitchen table (1), pancake (1), pear (1), plate (2, 3, 4, 5), pudding (2), water glass (1, 2, 4)
<i>pick up object</i>	pick up juice 1	You pick up juice. You hold juice (1).
<i>put down object</i>	put down apple 1	You put down apple on coffee table.
<i>open receptacle</i>	open fridge 1	You open fridge. You see bench (1), cupcake (1), fridge (1, 2), juice (1).
<i>close receptacle</i>	close dishwasher 1	You close dishwasher.
<i>turn on receptacle</i>	turn on dishwasher 1	You turn on dishwasher.

702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755

Table 4: Action types and sample text observations in the AI2-THOR simulator.

Action Type	Action	Text Observation
<i>initialization</i>	-	You are in the house, and you arrive at the bathroom, Looking quickly around the room, you see TowelHolder (1), Toilet (1), ToiletPaperHanger (1), Sink (1), SinkBasin (1), Shelf (1, 2, 3), HandTowelHolder (1), GarbageCan (1), BathtubBasin (1).
<i>go to receptacle</i>	go to BathtubBasin (1)	You arrive at BathtubBasin (1). You see SoapBar (2), Cloth (1), Bathtub (1), BathtubBasin (1).
<i>pick up object</i>	pick up SoapBar (2)	You pick up SoapBar (2). You see SoapBar (2), Cloth (1), Bathtub (1), BathtubBasin (1). I am holding SoapBar (2) now.
<i>put down object</i>	put down SoapBar (2)	You put SoapBar (2) on Shelf (1). You see SoapBar (1, 2, 3), Shelf (1), Candle (2).
<i>open receptacle</i>	open Microwave (1)	You open Microwave (1). You see StoveKnob (1, 2, 3, 4), StoveBurner (3), Potato (1), Pan (1, 2), Microwave (1), Bowl (3).
<i>drop object</i>	put down DishSponge (1)	You put down failed, drop it. You see SoapBar (3), Sink (1), SinkBasin (1), HandTowel (1), Faucet (2), DishSponge (1).
<i>close receptacle</i>	close Microwave (1)	You close Microwave (1). You see StoveKnob (1, 2, 3, 4), StoveBurner (3), Pan (1, 2), Microwave (1).
<i>turn on receptacle</i>	turn on Microwave (1)	You turn on Microwave (1). You see Microwave (1), Drawer (5, 6), Cabinet (7, 8, 10).
<i>turn off receptacle</i>	turn off Microwave (1)	You turn off Microwave (1). You see Microwave (1), Drawer (5, 6), Cabinet (7, 8, 10).

B LANGUAGE MODELS

Table 5 lists the exact language model names used in the experiments.

Table 5: List of language models used in the experiments. Model names are either from HuggingFace model hub or OpenAI API.

Class	Model name	Model size
LLaMA 3	meta-llama/Meta-Llama-3-8B	8B
	meta-llama/Meta-Llama-3-70B	70B
LLaMA 3.1	meta-llama/Llama-3.1-8B	8B
	meta-llama/Llama-3.1-70B	70B
Qwen2	Qwen/Qwen2-7B	7B
	Qwen/Qwen2-72B	72B
Qwen2.5	Qwen/Qwen2.5-7B	7B
	Qwen/Qwen2.5-72B	72B
Mistral	mistralai/Mistral-7B-v0.3	7B
Gemma	google/gemma-7b	7B
GPT-4o	gpt-4o-2024-05-13	-

C GPT-4O IMPLEMENTATION DETAILS

GPT-4o was implemented slightly differently due to the lack of access to log probabilities. At each time step t , the next action a_t was generated using the OpenAI API. If a_t was a *reasoning* action, it was handled in the same way as with other models. For *acting* actions, we first checked if a_t was a valid skill in \mathcal{A}_t . If it was valid, it was processed as usual. If not, the following corrective observation was provided to GPT-4o:

You should only output sentences that begin with Think, Act, or Expand. If you output Act, you should use one of actions of this list: [go to, pick up, put down, open, close, turn on, recall location of, done, failure].

Similarly, for *expanding* actions, if a_t was valid, it proceeded as normal. If not, the same corrective observation was added, prompting GPT-4o to refine its decision. This method ensured GPT-4o remained consistent with the task’s action space, despite the lack of log probability access.

D EXTENDED RESULTS

Table 6: Performance comparison of ReAct and ReAcTree on WaH dataset. GSR and SSR represent goal success rate and subgoal success rate, respectively. WM represents the use of the working memory.

Metric	Method	WM	LLaMA 3		Qwen2		Gemma
			8B	70B	7B	72B	7B
GSR (%)	ReAct	✗	3.00	17.00	9.00	20.00	2.00
		✓	8.00	23.00	13.13	23.00	8.00
	ReAcTree	✗	13.00	53.00	16.00	58.00	10.00
		✓	27.00	59.00	28.00	61.00	36.00
SSR (%)	ReAct	✗	26.82	45.83	26.60	48.02	22.20
		✓	34.22	51.25	32.98	47.58	28.48
	ReAcTree	✗	50.32	78.38	42.73	77.08	43.52
		✓	61.53	77.08	54.80	77.28	59.83

We compare ReAct and ReAcTree on WaH using various recent LLMs. Table 6 presents results for additional LLMs not included in Table 1. Across both large-scale and lighter models, ReAcTree consistently outperforms ReAct, with or without working memory. This indicates that the ReAcTree structure consistently enhances task planning performance by adaptively breaking down complex tasks into solvable agent-level tasks with control flow, allowing the same LLM to handle tasks more effectively and easily. We used Llama 3, Qwen 2, and Gemma 2 as off-the-shelf LLMs. The strong performance of the proposed ReAcTree without working memory further emphasizes its robustness, making it an effective approach for both lightweight and large-scale language model.

D.1 WITH WORKING MEMORY RESULTS

For the lightweight models, in the case of the LLaMA 3 8B model, ReAcTree shows significant improvements over ReAct, with a 19% increase in GSR and a 27.31% increase in SSR. Similarly, in the Qwen 2 7B model, ReAcTree achieves a 14.87% improvement in GSR and a 21.82% improvement in SSR compared to ReAct. In the Gemma 2 7B model, ReAcTree demonstrates the most impressive gains in lightweight model, with a 28% increase in GSR and a 31.35% increase in SSR.

For large-scale models such as LLaMA 3 70B and Qwen2 72B, ReAcTree also significantly outperforms ReAct. In the LLaMA 3 70B model, ReAcTree outperforms ReAct by 36% in GSR and by 25.83% in SSR. In the Qwen 2 72B model, ReAcTree shows a 38% improvement in GSR and a 29.06% improvement in SSR over ReAct.

D.2 WITHOUT WORKING MEMORY RESULTS

Even without working memory, ReAcTree surpasses ReAct by a significant margin. In lightweight models, ReAcTree shows notable improvements over ReAct, with a 10% increase in GSR and a 23.5% increase in SSR. Similar trends are observed in the Qwen 2 7B and Gemma 2 7B models, where ReAcTree consistently outperforms ReAct in both GSR and SSR.

For large-scale models, ReAcTree continues to outperform ReAct. In the LLaMA 3 70B model, ReAcTree achieves a 36% improvement in GSR and a 32.55% increase in SSR over ReAct. Similarly, in the Qwen 2 72B model, ReAcTree surpasses ReAct by 38% in GSR and 29.06% in SSR, even without working memory.

E FAILURE ANALYSIS

E.1 FAILURE TYPES

In this section, we categorize the failure cases of the proposed ReAcTree on the WAH-NL results using QWEN-2.5 72B model. The failure cases are classified into four types: *Expand-level*, *Agent-level*, *Constraints*, and *Instruction errors*. Table 7 summarizes these failure types.

First, *Expand-level* failures occur during the expand procedure of ReAcTree. These failures are categorized *Task decomposition*, *Control-flow selection*, and *Expand-level infinite loop* failures. Second, *Agent-level* failures arises during the planning process within individual agent nodes. These include *Navigation & search*, *Agent-level planning*, *Agent-level infinite loop*, and *Termination reasoning* failures. Third *Constraints* failures happen when tasks could be completed with more relaxed limits but fail due to the current thresholds. This category includes *Max step* failures, where the maximum number of decision steps is exceeded. Finally, *Instruction error* stems from issues in task instructions of WAH-NL test set. These are divided into *Incorrect instructions*, where the instruction differs from the goal, and *Ambiguous instructions*, where the instructions are unclear.

Table 7: Failure types observed in the WAH-NL results of the proposed ReAcTree using the QWEN-2.5 72B model.

Hierarchy	Failure Type	Details
Expand-level	Task decomposition	Failure to break down instructions into agent-level sub-goals
	Control-flow selection	Incorrect selection of control flow within expand-level decomposition
	Expand-level infinite loop	Infinite loop in expand-level decomposition
Agent-level	Navigation & search	Failure in object search & navigation under partial observability
	Agent-level planning	Failure in sub-goal planning
	Agent-level infinite loop	Infinite loop in agent-level decomposition
	Termination reasoning	Termination condition check failure in reasoning
Constraints	Max step	Exceeded the maximum allowable number of steps
Instruction error	Incorrect Instruction	When the goal stated in the instructions differs from the actual goal
	Ambiguous Instruction	When the goal state cannot be determined solely from the instructions

E.2 FAILURE ANALYSIS

In this section, we analyze the failure cases of ReAcTree using the QWEN-2.5 72B model, both with and without working memory. ReAcTree without working memory exhibited a total of 41 failure cases, categorized into *Expand-level* (9 cases), *Agent-level* (30 cases), *Constraints* (1 case), and *Instruction error* failures (9 cases). In contrast, ReAcTree with WM showed fewer failures, with a total of 37 cases distributed as *Expand-level* (6 cases), *Agent-level* (22 cases), *Constraints* (0 cases), and *Instruction error* failures (9 cases). In both settings, *Agent-level* failures were the most prevalent.

Table 8: Comparison of failure cases with and without the proposed working memory. We analyze the failure cases from a test set of 100 samples on the WAH-NL benchmark. A failure is defined as any instance where at least one sub-goal remains unsatisfied. The table also reports the average sub-goal success rate (SSR). In the table, "WM" denotes working memory, and "diff." represents the difference in SSR between cases with and without working memory.

Failure Type	Number of failures		Avg. SSR		diff.
	w.o. WM	w. WM	w.o. WM	w. WM	
Task decomposition	6	5 (-1)	58.33	66.67	+8.34
Control-flow selection	2	1 (-1)	16.67	66.67	+50.00
Expand-level infinite loop	1	0 (-1)	0	100	+100
Navigation & search	4	5 (+1)	72.08	70.42	-1.66
Agent-level planning	14	9 (-5)	57.70	73.73	+16.03
Agent-level infinite loop	3	4 (+1)	39.72	41.67	+1.95
Termination reasoning	9	4 (-5)	29.24	57.73	+28.49
Max step	1	0 (-1)	60	100	+40
Incorrect Instruction	4	4	37.50	37.50	0
Ambiguous Instruction	5	5	40.56	39.17	-1.39

To further understand the impact of working memory, we measured the average subgoal success rate (SSR) for each failure type across tasks that failed in either configuration. Table 8 summarizes these results, reporting the number of failures and the corresponding average SSR for each failure type. For example, in the case of *Task decomposition* failures, the total number of unique tasks used to compute SSR was 9. These tasks included task IDs 3, 10, 12, 23, 37, 38, and 62 from the without working memory configuration, and task IDs 12, 23, 37, 38, 47, and 77 from the with working memory setting. The union of these task sets is {3, 10, 12, 23, 37, 38, 47, 62, 77}. We calculated the average SSR for both configurations using this union set of failure cases.

The following sections provide a detailed analysis of the major failure types to gain deeper insights into ReAcTree’s performance.

Task decomposition is working well. In the expanding process of ReAcTree, the agent node selects a control flow and breaks down its goal sentence into multiple subgoals. *Task decomposition* failure occurs when this process generates incorrect subgoals. In our experiments on the WAH-NL test set, ReAcTree encountered these failures in 6 sample cases (6%) without utilizing working memory. However, with the integration of the proposed working memory, this number was reduced to 5 cases (5%), as shown in Table 8. Furthermore, the adoption of working memory significantly improved the average SSR (Avg. SSR in Table 8), with an increase of 8.34% point, from 58.33% to 66.67%. This demonstrates that the proposed working memory enhances subgoal success rates, ensuring partial success even when expand-level task decomposition is not entirely successful.

Working memory enhance Agent-level planning performance. The proposed working memory significantly reduce the number of *Agent-level planning* failures. As demonstrated in Table 8, incorporating the proposed working memory reduces the number of Agent-level planning failure cases from 14 to 9, while improving the average SSR by 16.03%. By recalling the locations from previous observations, the context length is shortened, and the number of nodes is reduced, enabling the LLM to generate more accurate plans. This improvement effectively minimizes Agent-level planning failures.

Working memory improves step efficiency in long-horizon task. The proposed working memory reduces the number of search steps required in partially observable environments. In max step failure cases, ReAcTree achieves an average SSR of 60.00% without working memory. However, with the adoption of working memory, these cases are resolved without reaching the maximum step limit.

Limitations. While the proposed ReAcTree and its memory mechanisms enhance embodied task planning capabilities, certain limitations persist. First, ReAcTree faces challenges in searching for target objects in partially observable environments. Unlike previous work Choi et al. (2024), which assumes a fully observable environment, ReAcTree operates in settings with partial observability, adding significant complexity. As a result, 5% of the evaluation samples fail due to navigation and search failures, with an average SSR of 70.42%.

Second, there is a risk of error propagation. While expand-level decomposition failures occur in only 5% of the evaluation samples, with an average SSR of 66.67%, such errors in expand-level planning can propagate and potentially disrupt the entire process. This limitation could be mitigated in future work by introducing an additional verification step for expand-level planning

Third, unpredictable errors, such as infinite loops in agent-level planning and termination reasoning failures, remain significant challenges. In our experiments, infinite loops in agent-level planning accounted for 4% of the evaluation set, with an average SSR of 41.67%, while termination reasoning failures also made up 4%, with an average SSR of 57.73%. Although this work focuses on designing an efficient architecture for agent nodes and implementing effective memory mechanisms, further improvements to the agent node itself are essential. To address these issues, a robust verification process is needed, where each node evaluates the current progress of the task. While this evaluation is weakly performed in the “Think stage” of ReAct, additional verification mechanisms could significantly mitigate these problems.

E.3 DETAILED DESCRIPTIONS OF THE FAILURE TYPES

E.3.1 TASK DECOMPOSITION FAILURE

Listing 1: The example of the task decomposition failure.

```

### Agent Node 1
Your task is to: Home service robot, input 1 cutlery fork, 1 wine glass and 1
    plate into the dishwasher. When complete then turn the dishwasher on.
...
Think: To solve the task, I need to accomplish 4 subgoals independently. Find and
    open the dishwasher, move the cutlery fork in the dishwasher, move the wine
    glass in the dishwasher, and close and turn on the dishwasher.
Expand: {'control_flow': 'parallel', 'conditions': 'find and open the dishwasher,
    move the cutlery fork in the dishwasher, move the wine glass in the dishwasher
    , close and turn on the dishwasher'}
```

Task decomposition failure occurs when the system initially generates an incorrect sequence of sub-goal instructions to address the given instruction. For instance, in Listing 1, the agent fails to decompose one of the sub-goals during the expand-level task decomposition process.

E.3.2 CONTROL-FLOW SELECTION FAILURE

Listing 2: The example of the control-flow selection failure.

```

### Agent Node 1
Your task is to: Please serve a water glass on the kitchen table
...
Think: To solve the task, I need to accomplish 2 subgoals sequentially. Find and
    pick up the water glass, and find the kitchen table and put down the water
    glass.
Expand: {'control_flow': 'sequence', 'conditions': 'find and pick up the water
    glass, find the kitchen table and put down the water glass'}
```

```

### Agent Node 2
Your primary goal is to: Please serve a water glass on the kitchen table
```

1026 To achieve this, you should perform your sibling tasks in sequence. At this level,
 1027 your sibling tasks are: find and pick up the water glass, and find the
 1028 kitchen table and put down the water glass.
 1029

1030 Your task is to: find and pick up the water glass
 1031 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 1032 (1), living room (1). You are in the middle of a bathroom (1). Looking quickly
 around the room, you see ...
 1033 Think: To solve the task, I need to recall the location of the water glass.
 1034 Act: recall location of water glass
 1035 You have not seen water glass before.
 1036 Think: Now, I need to find and pick up the water glass in one of the rooms. The
 1037 water glass is more likely to appear in kitchen 1, living room 1, or bedroom
 1. I can check each room one by one until I find and pick up the water glass.
 1038 Expand: {'control_flow': 'fallback', 'conditions': 'find and pick up the water
 1039 glass in kitchen 1, find and pick up the water glass in living room 1, find
 1040 and pick up the water glass in bedroom 1'}

1041 ### Agent Node 3
 1042 Your primary goal is to: find and pick up the water glass
 1043 To achieve this, you should perform your sibling tasks using a fallback strategy.
 1044 At this level, your sibling tasks are: find and pick up the water glass in
 1045 kitchen 1, find and pick up the water glass in living room 1, and find and
 1046 pick up the water glass in bedroom 1.
 1047 Your task is to: find and pick up the water glass in kitchen 1
 1048 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 1049 (1), living room (1). You are in the middle of a bathroom (1). Looking quickly
 around the room, you see ...
 1050 Think: I am in bathroom 1. Now, I need to find the water glass. The water glass is
 1051 more likely to appear on bathroom counter (1), or in bathroom cabinet (1). I
 1052 can check each place one by one, starting from bathroom counter 1, until I
 find the water glass.
 1053 Act: go to bathroom counter 1
 1054 ...
 1055 Think: I cannot find the water glass in bathroom 1.
 1056 Act: failure

1057 ### Agent Node 4
 1058 Your primary goal is to: find and pick up the water glass
 1059 To achieve this, you should perform your sibling tasks using a fallback strategy.
 1060 At this level, your sibling tasks are: find and pick up the water glass in
 1061 kitchen 1, find and pick up the water glass in living room 1, and find and
 1062 pick up the water glass in bedroom 1.
 1063 Your task is to: find and pick up the water glass in living room 1
 1064 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 1065 (1), living room (1). You are in the middle of a bathroom (1). Looking quickly
 around the room, you see ...
 1066 Think: I am in bathroom 1. Now, I need to go to living room 1.
 1067 Act: go to living room 1
 ...
 1068 Think: I cannot find the water glass in living room 1.
 1069 Act: failure

1070 ### Agent Node 5
 1071 Your primary goal is to: find and pick up the water glass
 1072 To achieve this, you should perform your sibling tasks using a fallback strategy.
 1073 At this level, your sibling tasks are: find and pick up the water glass in
 1074 kitchen 1, find and pick up the water glass in living room 1, and find and
 1075 pick up the water glass in bedroom 1.
 1076 Your task is to: find and pick up the water glass in bedroom 1
 1077 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 1078 (1), living room (1). You are in the middle of a living room (1). Looking
 quickly around the room, you see ...
 1079 Think: I am in living room 1. Now, I need to go to bedroom 1.
 Act: go to bedroom 1

```

1080 Think: I cannot find the water glass in bedroom 1.
1081 Act: failure
1082
1083 [The parent sequence node terminates because the fallback node returns a failure.]
1084

```

Control-flow selection failure arises from selecting an inappropriate control flow, often involving cases where a "fallback" node is used for object searching. If the parent node of the "fallback" node is not a "parallel" node but a "sequence" node, and all child nodes of the "fallback" fail, the "sequence" node does not fully execute and terminates prematurely. An example is summarized in Listing 2.

1090

1091 E.3.3 EXPAND-LEVEL INFINITE LOOP

1092

1093 Listing 3: The example of the expand-level infinite loop failure

1094

```

1096 Your task is to: move the wine in the fridge
1097 ...
1098 Expand: {'control_flow': 'sequence', 'conditions': 'find and pick up the wine,
1099         find the fridge and put down the wine'}
1100
1101 ### find and pick up the wine
1102
1103 Your task is to: find and pick up the wine
1104 ...
1105 Expand: {'control_flow': 'fallback', 'conditions': 'find and pick up the wine in
1106         kitchen 1, find and pick up the wine in living room 1, find and pick up the
1107         wine in bedroom 1'}
1108
1109 Your task is to: find and pick up the wine in kitchen 1
1110 ...
1111 Act: failure
1112
1113 Your task is to: find and pick up the wine in living room 1
1114 ...
1115 Act: failure
1116
1117 Your task is to: find and pick up the wine in bedroom 1
1118 ...
1119 Act: done
1120
1121 ### find the fridge and put down the wine
1122
1123 [The agent start the infinite loop]
1124 Your task is to: find the fridge and put down the wine
1125 ...
1126 Expand: {'control_flow': 'sequence', 'conditions': 'find and pick up the wine,
1127         find the fridge and put down the wine'}
1128
1129 ### find and pick up the wine
1130
1131 Your task is to: find and pick up the wine
1132 ...
1133 Act: failure
1134
1135 Your task is to: find and pick up the wine in living room 1

```

```

1134 ...
1135 Act: failure
1136
1137 Your task is to: find and pick up the wine in bedroom 1
1138 ...
1139 Act: done
1140
1141 ### find the fridge and put down the wine
1142
1143 Your task is to: find the fridge and put down the wine
1144 Expand: {'control_flow': 'sequence', 'conditions': 'find and pick up the wine,
1145         find the fridge and put down the wine'}
1146 [The agent try to find the wine again already found]
1147
1148 ### find and pick up the wine
1149
1150 Your task is to: find and pick up the wine
1151 Expand: {'control_flow': 'fallback', 'conditions': 'find and pick up the wine in
1152         kitchen 1, find and pick up the wine in living room 1, find and pick up the
1153         wine in bedroom 1'}
1154 ...

```

Lastly, an expand-level infinite loop occurs when the expand-level decomposition succeeds, but a repetitive sub-goal sequence is endlessly generated, causing the tree structure to expand infinitely. While this issue arises in only one sample with ReAcTree without working memory and is completely absent when working memory is employed, it represents a critical failure type. In such cases, the system is unable to execute actions correctly and continues to expand indefinitely, making the goal impossible to achieve.

1161 E.3.4 NAVIGATION & SEARCH FAILURE

1162 Listing 4: The example of the navigation & search failure.

```

1165 ...
1166 Your primary goal is to: move the glass of water in the dishwasher
1167 To achieve this, you should perform your sibling tasks in sequence. At this level,
1168 your sibling tasks are: find and pick up the glass of water, and find the
1169 dishwasher and put down the glass of water.
1170 Your task is to: find and pick up the glass of water
1171 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
1172 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
1173 around the room, you see chair (1, 2, 3, 4), dishwasher (1), faucet (1),
1174 fridge (1), kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2),
1175 kitchen table (1), microwave oven (1), nightstand (1, 2), sink (1), sofa (1),
1176 stove (1), toaster (1).
1177 Think: To solve the task, I need to recall the location of the glass of water.
1178 Act: recall location of glasses
1179 You have not seen glasses before.
1180 Think: Now, I need to find and pick up the glass of water in one of the rooms. The
1181 glass of water is more likely to appear in kitchen 1, living room 1, or
1182 bedroom 1. I can check each room one by one until I find and pick up the glass
1183 of water.
1184 Expand: {'control_flow': 'fallback', 'conditions': 'find and pick up the glass of
1185         water in kitchen 1, find and pick up the glass of water in living room 1, find
1186         and pick up the glass of water in bedroom 1'}
1187 [Starting search for the water glass.]
1188 Your primary goal is to: find and pick up the glass of water
1189 To achieve this, you should perform your sibling tasks using a fallback strategy.
1190 At this level, your sibling tasks are: find and pick up the glass of water in
1191 kitchen 1, find and pick up the glass of water in living room 1, and find and
1192 pick up the glass of water in bedroom 1.

```

1188 Your task is to: find and pick up the glass of water in kitchen 1
 1189 ...
 1190 Think: I cannot find the glass of water in kitchen 1.
 1191 Act: failure

1192 Your primary goal is to: find and pick up the glass of water
 1193 To achieve this, you should perform your sibling tasks using a fallback strategy.
 1194 At this level, your sibling tasks are: find and pick up the glass of water in
 1195 kitchen 1, find and pick up the glass of water in living room 1, and find and
 1196 pick up the glass of water in bedroom 1.
 1197 Your task is to: find and pick up the glass of water in living room 1
 1198 ...
 1199 Think: I cannot find the glass of water in living room 1.
 1200 Act: failure

1201 Your primary goal is to: find and pick up the glass of water
 1202 To achieve this, you should perform your sibling tasks using a fallback strategy.
 1203 At this level, your sibling tasks are: find and pick up the glass of water in
 1204 kitchen 1, find and pick up the glass of water in living room 1, and find and
 1205 pick up the glass of water in bedroom 1.
 1206 Your task is to: find and pick up the glass of water in bedroom 1
 1207 ...
 1208 Think: I cannot find the glass of water in bedroom 1.
 1209 Act: failure

1210 [Unable to find the water glass, proceeding to the next parallel node]
 1211 Your primary goal is to: Please put a cutlery fork and a glass of water in the
 1212 dishwasher, then turn the dishwasher on.
 1213 To achieve this, you should perform your sibling tasks in parallel. At this level,
 1214 your sibling tasks are: find and open the dishwasher, move the cutlery fork
 1215 in the dishwasher, move the glass of water in the dishwasher, and close and
 1216 turn on the dishwasher.
 1217 Your task is to: close and turn on the dishwasher
 1218 ...

1218 [Navigation & search failure occurs when the agent is unable to find the target object despite executing the entire generated search action sequence \(commonly associated with the "fallback" node in ReAcTree\). Object navigation is particularly challenging due to partial observability, which can prevent the task from being completed even if the expand-level plan is successfully generated.](#)

1224 E.3.5 AGENT-LEVEL PLANNING FAILURE

1226 **Listing 5: The example of the agent-level planning failure.**

1228 Your primary goal is to: find and pick up the cake
 1229 To achieve this, you should perform your sibling tasks using a fallback strategy.
 1230 At this level, your sibling tasks are: find and pick up the cake in kitchen 1,
 1231 find and pick up the cake in living room 1, and find and pick up the cake in
 1232 bedroom 1.
 1233 Your task is to: find and pick up the cake in bedroom 1
 1234 ...
 1235 Think: I am in living room 1. Now, I need to go to bedroom 1.
 1236 Act: go to bedroom 1
 1237 You move to the bedroom (1). Looking quickly around the room, you see ..
 1238 Think: Now, I need to find the cake. The cake is more likely to appear on coffee
 1239 table (2), desk (2, 3), or in cabinet (1). I can check each place one by one,
 1240 starting from coffee table 2, until I find the cake.
 1241 ...
 1242 You open cabinet. You see apple (2), cabinet (1), computer (2), desk (2), light
 1243 switch (4), pound cake (1), sundae (1)
 1244 Think: I find pound cake 1. Now, I need to pick it up.
 1245 Act: pick up pound cake 1

1242 You pick up pound cake. You hold pound cake (1).
 1243 Think: Now I accomplish the goal.
 1244 Act: done

1245
 1246 Agent-level planning failure occurs when the agent generates an incorrect or meaningless action
 1247 sequence within action-performing nodes. This can happen when the action sequence is misaligned
 1248 with the provided sub-goal instructions or when the actions executed do not align with the previous
 1249 reasoning outcomes. For example, in Listing 5, the sub-goal is *"Find and pick up the cake in bed-*
 1250 *room 1."* The agent reasons correctly, stating: *"Think: Now I need to find the cake. The cake is more*
 1251 *likely to be on coffee table (2), desk (2, 3), or in cabinet (1). I can check each place one by one,*
 1252 *starting from coffee table 2, until I find the cake."* However, instead of picking up the cake, the agent
 1253 mistakenly picks up a pound cake.

1254
 1255

1256 E.3.6 AGENT-LEVEL INFINITE LOOP

1257
 1258

Listing 6: The example of the agent-level infinite loop.

1259 Your task is to: go find a bottle of wine and a pudding there on the coffee table.
 1260
 1261 [Starting to search a wine and pudding]
 1262 Think: I need to find a bottle of wine and a pudding. The wine is more likely to
 1263 appear in kitchen 1, and the pudding is more likely to appear in living room
 1264 1.
 1265 ...
 1266 Think: I can see fridge 1. I need to open fridge 1 to find the wine.
 1267 Act: go to fridge 1
 1268 You arrive at the fridge (1). The fridge (1) is closed. You see fridge (1), mug
 1269 (3)
 1270 Think: I need to open fridge 1.
 1271 Act: open fridge 1
 1272 You open fridge. You see apple (1), fridge (1), mug (3), sofa (1)
 1273
 1274 [Picked up an apple instead of wine]
 1275 Think: I can see apple 1. I need to pick up apple 1.
 1276 Act: pick up apple 1
 1277 You pick up apple. You hold apple (1).
 1278
 1279 [Search the pudding]
 1280 Think: Now, I need to find the pudding. The pudding is more likely to appear in
 1281 living room 1.
 1282 ...
 1283 Think: I can see coffee table 1. I need to go to coffee table 1 to find the
 1284 pudding.
 1285 Act: go to coffee table 1
 1286 You arrive at the coffee table (1). You see apple (1), coffee table (1), sofa (2),
 1287 tv (1) You hold apple (1).
 1288
 1289 [Put down apple]
 1290 Think: I can see apple 1. I need to put down apple 1.
 1291 Act: put down apple 1
 1292 You put down apple on coffee table
 1293 Think: Now, I need to find the pudding. The pudding is more likely to appear in
 1294 living room 1.
 1295 Act: go to living room 1
 You move to the living room (1). Looking quickly around the room, you see ...
 Think: I can see coffee table 1. I need to go to coffee table 1 to find the
 pudding.
 Act: go to coffee table 1
 You arrive at the coffee table (1). You see apple (1), coffee table (1), sofa (2),
 tv (1)
 [Pick up apple again]

1296 Think: I can see apple 1. I need to pick up apple 1.
 1297 Act: pick up apple 1
 1298 You pick up apple. You hold apple (1).
 1299 Think: Now, I need to find the pudding. The pudding is more likely to appear in
 1300 living room 1.
 1301 Act: go to living room 1
 1302 You move to the living room (1). Looking quickly around the room, you see
 1303 bookshelf (2, 3), chair (6), closet (1), coffee table (1), computer (1), desk
 1304 (2), nightstand (4), sofa (2, 3), tv (1). You hold apple (1).
 1305 Think: I can see coffee table 1. I need to go to coffee table 1 to find the
 1306 pudding.
 1307 Act: go to coffee table 1
 1308 You arrive at the coffee table (1). You see apple (1), coffee table (1), sofa (2),
 1309 tv (1) You hold apple (1).
 1310 [Put down apple]
 1311 Think: I can see apple 1. I need to put down apple 1.
 1312 Act: put down apple 1
 1313 You put down apple on coffee table
 1314 [Search the pudding]
 1315 Think: Now, I need to find the pudding. The pudding is more likely to appear in
 1316 living room 1.
 1317 Act: go to living room 1
 1318 You move to the living room (1). Looking quickly around the room, you see
 1319 bookshelf (2, 3), chair (6), closet (1), coffee table (1), computer (1), desk
 1320 (2), nightstand (4), sofa (2, 3), tv (1).
 1321 Think: I can see coffee table 1. I need to go to coffee table 1 to find the
 1322 pudding.
 1323 Act: go to coffee table 1
 1324 You arrive at the coffee table (1). You see apple (1), coffee table (1), sofa (2),
 1325 tv (1)
 1326 Think: I can see apple 1. I need to pick up apple 1.
 1327 ...

1326 An agent-level infinite loop occurs when the agent gets trapped in a continuous cycle of actions
 1327 during its execution and cannot break free. Unlike the previous planning failure, where the agent
 1328 could transition to the next node, this failure is more severe as it causes the agent’s performance to
 1329 stall at a specific node. For example, as shown in Listing 6, the agent repeatedly picks up and puts
 1330 down an apple in a loop, an action unrelated to achieving the task’s sub-goal.

1333 E.3.7 TERMINATION REASONING FAILURE

1335 Listing 7: The example of the termination reasoning failure.

1337 Your primary goal is to: find and pick up the juice
 1338 To achieve this, you should perform your sibling tasks using a fallback strategy.
 1339 At this level, your sibling tasks are: find and pick up the juice in kitchen
 1340 1, find and pick up the juice in living room 1, and find and pick up the juice
 1341 in bedroom 1.
 1342 Your task is to: find and pick up the juice in kitchen 1
 1343 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 1344 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
 1345 around the room, you see chair (1, 2, 3, 4), dishwasher (1), faucet (1),
 1346 fridge (1), kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 3),
 1347 kitchen table (1), microwave oven (1), nightstand (1, 2), sink (1), sofa (1),
 1348 stove (1), toaster (1).
 1349 Think: I am in kitchen 1. Now, I need to find the juice. The juice is more likely
 to appear on kitchen counter (1, 3), kitchen table (1), or in fridge (1),

1350 kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8). I can check each place one by one,
1351 starting from kitchen counter 1, until I find the juice.
1352 Act: go to kitchen counter 1
1353 ...
1354 Think: I cannot find the juice. Now, I need to go to living room 1.
1355 [The agent fail to terminate the node]
1356 Act: go to living room 1
1357 You move to the living room (1). Looking quickly around the room, you see
1358 bookshelf (2, 3), chair (6), closet (1), coffee table (1), computer (1), desk
1359 (2), nightstand (4), sofa (2, 3), tv (1).
1360 Think: I am in living room 1. Now, I need to find the juice. The juice is more
1361 likely to appear on coffee table (1), desk (2), or in nightstand (4). I can
1362 check each place one by one, starting from coffee table 1, until I find the
1363 juice.
1364 Act: go to coffee table 1
1365 ...
1366 [The agent fail to terminate the node]
1367 Think: I cannot find the juice. Now, I need to go to kitchen 1.
1368 Act: go to kitchen 1
1369 ...
1370 [The agent fail to terminate the node]
1371 Think: I cannot find the juice. Now, I need to go to living room 1.
1372 Act: go to living room 1
1373 ...
1374 [The agent fail to terminate the node]
1375 Think: I cannot find the juice. Now, I need to go to bedroom 1.
1376 Act: go to bedroom 1
1377 ...
1378 [The agent fail to terminate the node]
1379 Think: I cannot find the juice. Now, I need to go to kitchen 1.
1380 Act: go to kitchen 1
1381 ...
1382 Max steps

1383 A termination reasoning failure occurs when the agent either fails to recognize that the necessary
1384 actions have been completed and cannot decide to terminate the node or prematurely terminates the
1385 node before fully executing the planned action sequence. Unlike the agent-level infinite loop, where
1386 the agent endlessly repeats actions that are unrelated or only partially related to task completion, a
1387 termination reasoning failure arises when the agent fails to identify task completion or failure and re-
1388 peats the process, even after executing the entire action sequence derived from sub-goal instructions
1389 and reasoning processes.
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403

1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457

E.3.8 MAX STEP

Listing 8: The example of the Max Step failure.

```

### Generated ReAcTree
Instruction : Put a wine a juice a pound cake
a pudding and an apple on the kitchen table.

[a1]rootnode (parallel)

[a2]move the wine on the kitchen table (sequence)
  [a3]find and pick up the wine (fallback)
    [a4]find and pick up the wine in kitchen 1 [S]
    find and pick up the wine in living room 1
    find and pick up the wine in bedroom 1

  [a5]find the kitchen table and put down the wine [S]

[a6]move the juice on the kitchen table (sequence)
  [a7]find and pick up the juice (fallback)
    [a8]find and pick up the juice in kitchen 1 [F]
    [a9]find and pick up the juice in living room 1 [S]
    find and pick up the juice in bedroom 1

  [a10]find the kitchen table and put down the juice [S]

[a11]move the pound cake on the kitchen table (sequence)
  [a12]find and pick up the pound cake (sequence)
    [a13]find and pick up the pound cake in kitchen 1 [S]
    find and pick up the pound cake in living room 1
    find and pick up the pound cake in bedroom 1

  [a14]find the kitchen table and put down the pound cake [S]

[a15]move the pudding on the kitchen table (sequence)
  [a16]find and pick up the pudding (fallback)
    [a17]find and pick up the pudding in kitchen 1 [F]
    [a18]find and pick up the pudding in living room 1 [S]
    find and pick up the pudding in bedroom 1

  [a19]find the kitchen table and put down the pudding [S]

[a20]move the apple on the kitchen table (sequence)
  [a21]find and pick up the apple (fallback1)
    [a22]find and pick up the apple in kitchen 1 [MSE]
    find and pick up the apple in living room 1 [MSE]
    find and pick up the apple in bedroom 1 [MSE]

  find the kitchen table and put down the apple [MSE]
[S] : Success, [F] : Failure, [MSE] : Max Step Error
[An] : n-th agent

### Agent 22
Your primary goal is to: find and pick up the apple
To achieve this, you should perform your sibling tasks using a fallback strategy.
At this level, your sibling tasks are: find and pick up the apple in kitchen
1, find and pick up the apple in living room 1, and find and pick up the apple
in bedroom 1.
Your task is to: find and pick up the apple in kitchen 1
You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
(1), living room (1). You are in the middle of a kitchen (1). Looking quickly
around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet

```

1458 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
1459 oven (1), sink (2), stove (1), toaster (1).

1460 Think: I am in kitchen 1. Now, I need to find the apple. The apple is more likely
1461 to appear on kitchen counter (1, 2), kitchen table (1), or in fridge (1, 2),
1462 kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8). I can check each place one by one,
1463 starting from kitchen counter 1, until I find the apple.

1464 Act: go to kitchen counter 1

1464 You arrive at the kitchen counter (1). You see bowl (1, 2, 3, 4, 5), chair (2),
1465 chicken (1), cutlets (1), dishwashing liquid (1), faucet (2), kitchen cabinet
1466 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1), plate (1, 2, 3, 4), sink (2),
1467 washing sponge (1), wine (2)

1468 Act: go to kitchen counter 2

1468 You arrive at the kitchen counter (2). You see chair (1, 2), kitchen cabinet (3,
1469 5, 6, 7, 8), kitchen counter (1, 2), light switch (2), plate (1, 2, 3, 4),
1470 wine (2)

1471 Act: go to kitchen table 1

1472 You arrive at the kitchen table (1). You see bench (2), book (1), bowl (6), coffee
1473 pot (1), condiment shaker (1, 2), juice (1), kitchen table (1), pound cake
1474 (1), pudding (2), wine (1)

1474 Act: go to fridge 1

1475 You arrive at the fridge (1). The fridge (1) is open. You see bench (1), fridge
1476 (1)

1477 Act: go to fridge 2

1478 You arrive at the fridge (2). The fridge (2) is open. You see bench (1), fridge
1479 (1, 2), pancake (1)

1479 Act: go to kitchen cabinet 1

1480 You arrive at the kitchen cabinet (1). The kitchen cabinet (1) is open. You see
1481 chicken (1), cooking pot (1), cutlery knife (1, 2), cutlets (1), dishwashing
1482 liquid (1), faucet (2), frying pan (1), garbage can (1), kitchen cabinet (1,
1483 4), kitchen counter (1), sink (2), stove (1)

1484 Act: go to kitchen cabinet 2

1484 You arrive at the kitchen cabinet (2). The kitchen cabinet (2) is open. You see
1485 bowl (1, 2, 3, 4, 5), chair (2), chicken (1), cutlery knife (1, 2), cutlets
1486 (1), dishwashing liquid (1), faucet (2), kitchen cabinet (1, 2, 3, 4, 5, 6, 7)
1487 , kitchen counter (1), plate (1, 2, 3, 4), sink (2), washing sponge (1)

1488 Act: go to kitchen cabinet 3

1488 You arrive at the kitchen cabinet (3). The kitchen cabinet (3) is open. You see
1489 bowl (1, 2, 3, 4, 5), chair (1, 2), dishwashing liquid (1), faucet (2),
1490 kitchen cabinet (2, 3, 4, 5, 6, 7), kitchen counter (1, 2), plate (1, 2, 3, 4)
1491 , sink (2), washing sponge (1), wine (2)

1492 Act: go to kitchen cabinet 4

1493 You arrive at the kitchen cabinet (4). The kitchen cabinet (4) is open. You see
1494 bowl (1, 2, 3, 4, 5), chicken (1), cooking pot (1), cutlery knife (1, 2),
1495 cutlets (1), dishwashing liquid (1), faucet (2), garbage can (1), kitchen
1496 cabinet (1, 2, 4), kitchen counter (1), sink (2), washing sponge (1)

1496 Act: go to kitchen cabinet 5

1497 You arrive at the kitchen cabinet (5). The kitchen cabinet (5) is open. You see
1498 bowl (1, 2, 3, 4, 5), chair (1, 2), cutlets (1), dishwashing liquid (1),
1499 faucet (2), kitchen cabinet (2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2),
1500 plate (1, 2, 3, 4), sink (2), washing sponge (1), wine (2)

1501 Max steps

1502
1503
1504
1505
1506
1507
1508
1509
1510
1511

A max step error occurs when the agent successfully generates a valid agent tree and executes the plans correctly but is forced to terminate upon reaching the maximum step limit. This type of failure is likely solvable if the agent is given a higher step limit. An example of this case is presented in Listing 8, with the generated agent tree structure. Here, five sub-goals are correctly managed within the expand-level and agent-level instruction tree and executed properly by the agent until the maximum step limit is reached. The SSR achieved in this case is 60.00% as described in Table 8. However, the proposed working memory significantly enhances step efficiency in such cases, enabling all sub-goals to be resolved within the given step limit.

1512 E.3.9 INCORRECT INSTRUCTION

1513
1514 Listing 9: The example of the incorrect instruction.

```
1515 ### Instruction
1516 Please, put 1 wine glass, 1 water glass, and 1 plate on the
1517 table
1518
1519 ### Goal condition
1520 "on_plate_kitchentable": 1,
1521 "on_waterglass_kitchentable": 1,
1522 "on_wineglass_kitchentable": 1,
1523 "on_cutleryfork_kitchentable": 1
```

1524 Incorrect instructions occur when the instructions fail to specify all the required sub-goals. In such
1525 cases, no task planning agent can complete the task due to insufficient information. For example,
1526 as shown in Listing 9, there are four sub-goals, but the instruction specifies only three, omitting the
1527 step of placing a cutlery fork on the kitchen table. Additionally, the instruction is ambiguous, as
1528 it does not clarify which type of table is the intended target receptacle. Despite this incorrectness,
1529 ReAcTree successfully completes the three specified sub-goals in Listing. 9, but achieving an SSR
1530 of 75.00%.

1531 1532 1533 E.3.10 AMBIGUOUS INSTRUCTION

1534
1535 Listing 10: The example of the ambiguous instruction.

```
1536 [Case 1]
1537 ### Instruction
1538 give me an apple, a pudding and serve a glass of wine
1539
1540 ### Goal condition
1541 "on_wine_coffeetable": 1,
1542 "on_pudding_coffeetable": 1,
1543 "on_apple_coffeetable": 1
1544
1545 [Case 2]
1546 ### Instruction
1547 Always get a glass of water first before anything else,
1548 before food on the table with a cutlery fork and knife
1549
1550 ### Goal condition
1551 "on_plate_kitchentable": 1,
1552 "on_waterglass_kitchentable": 1,
1553 "on_cutleryfork_kitchentable": 1
```

1554 The WAH-NL dataset also contains ambiguous instructions. In such cases, the instructions fail to
1555 explicitly describe the goal condition, use unclear object names, or refer to objects with high-level
1556 conceptual nouns. This ambiguity can confuse the LLM when reasoning about the goal state, leading
1557 to the generation of incorrect plans.

1558
1559
1560
1561
1562
1563
1564
1565

F EXPERIMENTS DETAILS OF ALFRED DATASET

F.1 AI2THOR SIMLUATOR AND ALFRED DATASET

AI2THOR simulator supports 9 interaction actions: “pick up,” “open,” “close,” “turn on,” “turn off,” “slice,” and “put down,” “drop,” and one navigation action “go to.” ALFRED dataset consists of 7 task types: *simple pick & place*, *pick & place with movable receptacle*, *cool & place*, *heat & place*, *pick two object & place*, *clean & place*, and *look object in light*. Following previous work (Choi et al., 2024), the *pick two object and place* is excluded. In LoTa-Bench, the AI2THOR simulation used the “find” for navigation action, but since it wasn’t a partially observable settings, once “find [object]” was selected as an action, the agent moved directly to the object without considering whether it was inside a receptacle or an unobserved object. However, in this experiment, we improved this limitation by applying a partially observable settings, changing the navigation action name from “find” to “go to” directing the agent to either the object or its receptacle.

F.2 MEMORY CONSTRUCTION

To construct the episodic memory, we began by randomly selecting three task trajectories for each task type from the ALFRED training set. Then, we collected human-annotated text trajectories with working memory applied for both the ReAct and ReAcTree models. Next, while running the LLaMa 3 70B model on the training set, we embedded the human-annotated text trajectories as in-context examples. Due to the large size of the ALFRED dataset, we sampled 5% of the training set. Only the trajectories from successfully completed tasks were added to the episodic memory, with no more than five per task type. Finally, we combined these human-annotated text trajectories with the successfully completed task trajectories to form the episodic memory for both ReAct and ReAcTree.

F.3 QUALATATIVE RESULTS

Figures 6 and 7 show the qualitative results comparing a failed trajectory in ReAct and a successful trajectory in ReAcTree for the *heat & place* task type using the LLaMA 3 70B model. The NL instruction for the task is “put the cooked tomato slice on the kitchen island”, which requires slicing and moving the object. there are a challenge with the ALFRED dataset, where ambiguous NL instructions complicate task resolution. For example, in this task, the instruction ambiguously states to place the item on the “kitchen island,” yet the agent placed it on the countertop since the simulator does not provide a receptacle explicitly labeled “kitchen island.” Such ambiguity can hinder accurate task interpretation and execution.

ReAcTree successfully expands and executes the instruction using sequence control flow with eight subtask squence, such as: “find and pick up the knife, find and slice and then pick up the tomato, find and open the microwave, find the microwave and put down the tomato, find and close and turn on the microwave, find and turn off and open microwave, find and pick up tomato in microwave, and find a countertop and put down the tomato on the countertop.” In contrast, ReAct failed by initially navigating to SinkBasin (1) and picking up the tomato before the knife, causing the task to fail as the knife should have been picked up and used to slice the tomato first. Meanwhile, ReAcTree correctly searched through CounterTop (1) and CounterTop (2) to locate the knife, leading to the successful completion of the task. Please refer to Appendix H for detailed text trajectories.

1620 G EXAMPLE OF PLANNING RESULTS

1621

1622 G.1 WAH-NL EXAMPLE CASES

1623

1624

1625 **Put the apple, pancake, cupcake, and juice on the kitchen table.**

1626

1626 init go to kitchen 1 go to kitchen counter 1 go to kitchen counter 2 go to kitchen table 1 go to fridge 1

1627

1627

1628

1628

1629

1629

1630

1630

1631

1631 open fridge 1 pick up cupcake 1 go to kitchen table 1 put down cupcake 1 recall location of pancake go to kitchen cabinet 1

1632

1632

1633

1633

1634

1634

1635

1635

1636

1636

1637

1637 open kitchen cabinet 1 go to kitchen cabinet 2 open kitchen cabinet 2 go to kitchen cabinet 3 open kitchen cabinet 3 go to kitchen cabinet 4

1638

1638

1639

1639

1640

1640 open kitchen cabinet 4 go to kitchen cabinet 5 open kitchen cabinet 5 pick up pudding 2 go to kitchen table 1 put down pudding 2

1641

1641

1642

1642

1643

1643

1644

1644 recall location of juice go to kitchen cabinet 6 open kitchen cabinet 6 go to kitchen cabinet 7 open kitchen cabinet 7 pick up apple 1

1645

1645

1646

1646

1647

1647

1648

1648 go to kitchen table 1 put down apple 1 recall location of pancake go to kitchen cabinet 7 pick up pancake 1 go to kitchen table 1

1649

1649

1650

1650

1651

1651

1652

1652

1653

1653 put down pancake 1 recall location of juice go to kitchen cabinet 8 open kitchen cabinet 8 go to fridge 1 go to fridge 2

1654

1654

1655

1655

1656

1656

1657

1657

1658

1658

1659

1659 open fridge 2 pick up apple 2 go to kitchen table 1 put down apple 2 recall location of juice

1660

1660

1661

1661

1662

1662

1663

1663

1664

1664

1665

1665

1666

1666

1667

1667

1668

1668

1669

1669

1670

1671

1672

1673

Figure 5: Failure case of ReAct on the WAH-NL dataset using LLaMA 3 70B.

1674
 1675
 1676
 1677
 1678
 1679
 1680
 1681
 1682
 1683
 1684
 1685
 1686
 1687
 1688
 1689
 1690
 1691
 1692
 1693
 1694
 1695
 1696
 1697
 1698
 1699
 1700
 1701
 1702
 1703
 1704
 1705
 1706
 1707
 1708
 1709
 1710
 1711
 1712
 1713
 1714
 1715
 1716
 1717
 1718
 1719
 1720
 1721
 1722
 1723
 1724
 1725
 1726
 1727

G.2 ALFRED EXAMPLE CASES

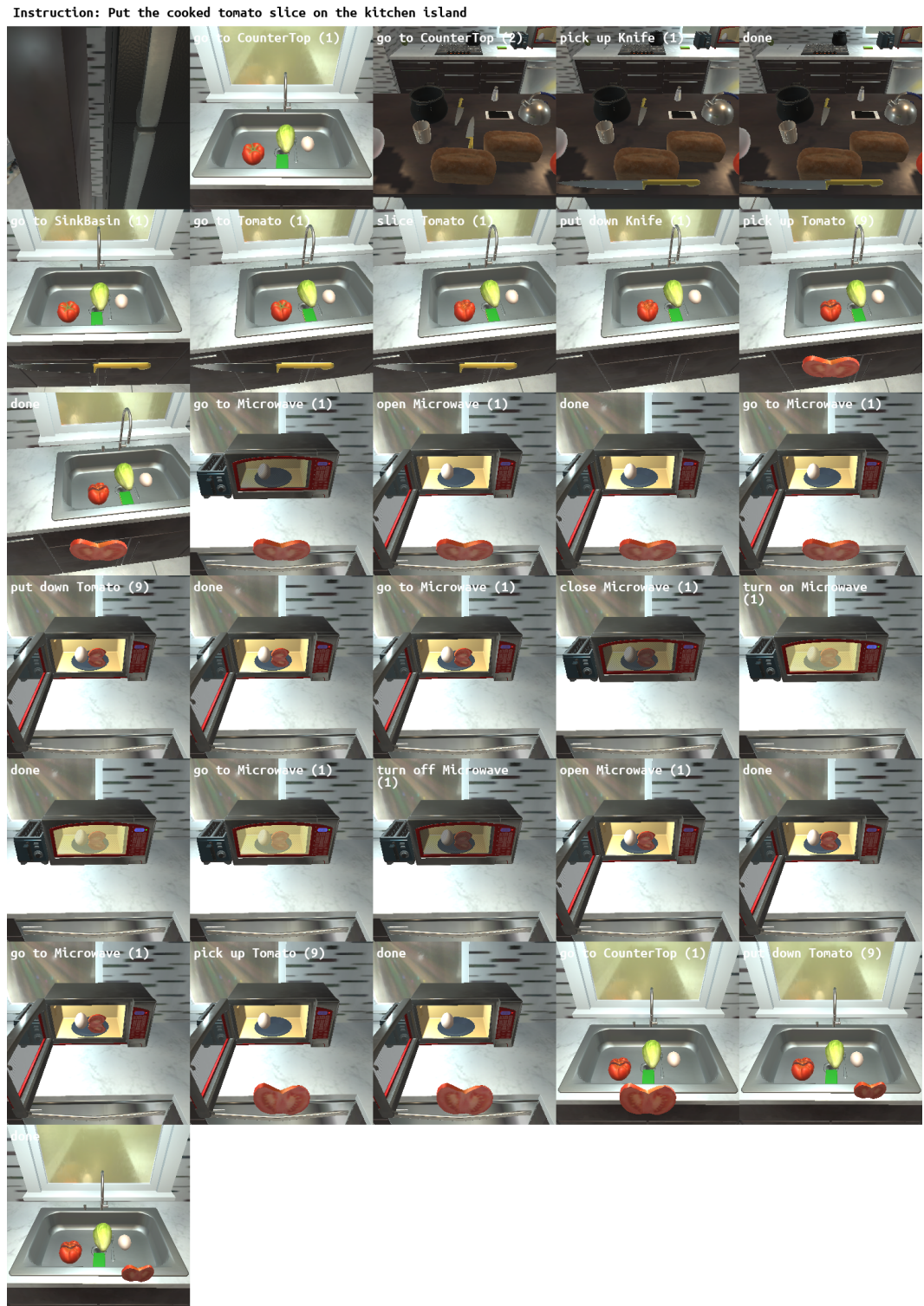


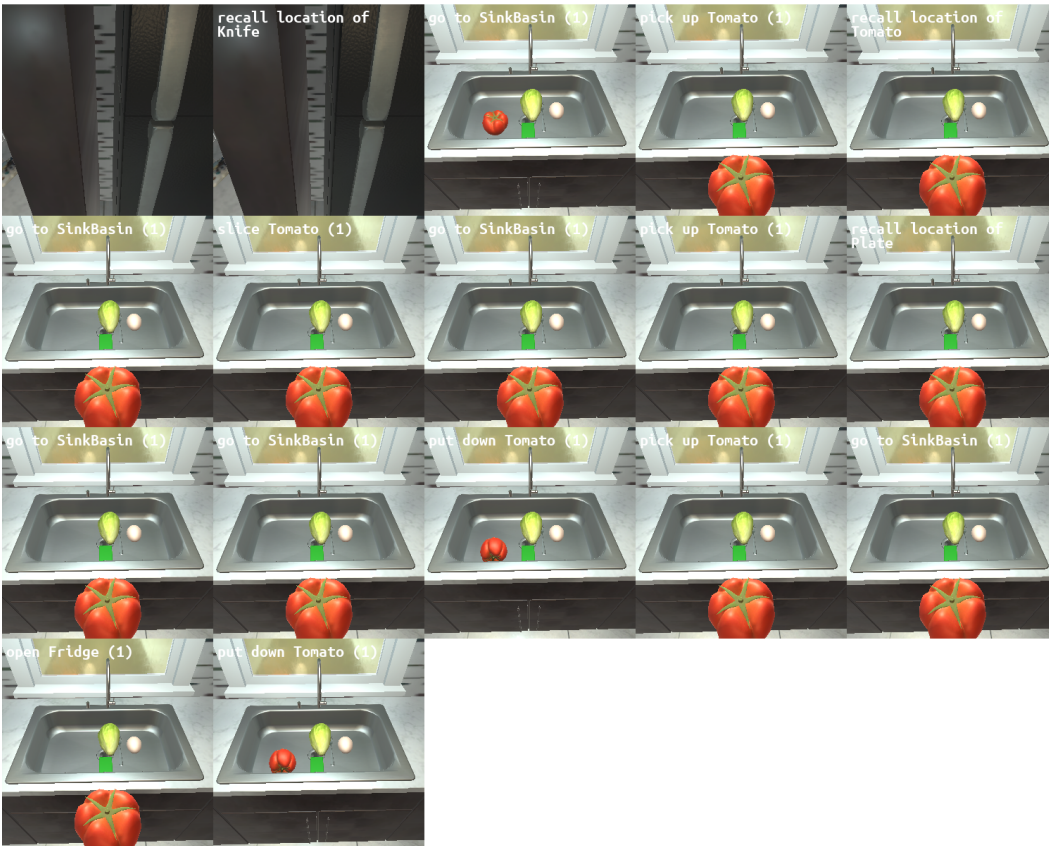
Figure 6: Success case of ReAcTree on the ALFRED dataset using LLaMA 3 70B.

1728

1729

Instruction: Put the cooked tomato slice on the kitchen island

1730



1731

1732

1733

1734

1735

1736

1737

1738

1739

1740

1741

1742

1743

1744

1745

1746

1747

1748

1749

1750

1751

1752

1753

1754

1755

1756

Figure 7: Failure case of ReAct on the ALFRED dataset using LLaMA 3 70B.

1757

1758

1759

H EXAMPLE OF TEXT TRAJECTORIES

1760

1761

H.1 WAH-NL EXAMPLE CASES

1762

1763

1764

Listing 11: Success trajectory of ReAcTree on the WAH-NL dataset using LLaMA 3 70B.

1765

```

### Agent Node 1
Your task is to: Put the apple, pancake, cupcake, and juice on the kitchen table.
You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
(1), living room (1). You are in the middle of a kitchen (1). Looking quickly
around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
(1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
oven (1), sink (2), stove (1), toaster (1).
Think: To solve the task, I need to accomplish 4 subgoals independently. Move the
apple on the kitchen table, move the pancake on the kitchen table, move the
cupcake on the kitchen table, and move the juice on the kitchen table.
Expand:
- control flow: parallel
- subgoals: move the apple on the kitchen table, move the pancake on the kitchen
table, move the cupcake on the kitchen table, move the juice on the kitchen
table

### Agent Node 2
Your primary goal is to: Put the apple, pancake, cupcake, and juice on the kitchen
table.

```

1766

1767

1768

1769

1770

1771

1772

1773

1774

1775

1776

1777

1778

1779

1780

1781

1782 To achieve this, you should perform your sibling tasks in parallel. At this level,
 1783 your sibling tasks are: move the apple on the kitchen table, move the pancake
 1784 on the kitchen table, move the cupcake on the kitchen table, and move the
 1785 juice on the kitchen table.

1786 Your task is to: move the apple on the kitchen table

1787 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 1788 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
 1789 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
 1790 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
 1791 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
 1792 oven (1), sink (2), stove (1), toaster (1).

1792 Think: To solve the task, I need to accomplish 2 subgoals sequentially. Find and
 1793 pick up the apple, and find the kitchen table and put down the apple.

1793 Expand

- 1794 - control flow: sequence
- 1795 - subgoals: find and pick up the apple, find the kitchen table and put down the
 1796 apple

1797

1798 ### Agent Node 3

1799 Your primary goal is to: move the apple on the kitchen table

1800 To achieve this, you should perform your sibling tasks in sequence. At this level,
 1801 your sibling tasks are: find and pick up the apple, and find the kitchen
 1802 table and put down the apple.

1803 Your task is to: find and pick up the apple

1804 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 1805 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
 1806 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
 1807 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
 1808 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
 1809 oven (1), sink (2), stove (1), toaster (1).

1808 Think: To solve the task, I need to recall the location of the apple.

1809 Act: recall location of apple

1810 You have not seen apple before.

1811 Think: Now, I need to find and pick up the apple in one of the rooms. The apple is
 1812 more likely to appear in kitchen 1, living room 1, or bedroom 1. I can check
 1813 each room one by one until I find and pick up the apple.

1813 Expand:

- 1814 - control flow: fallback
- 1815 - subgoals: find and pick up the apple in kitchen 1, find and pick up the apple
 1816 in living room 1, find and pick up the apple in bedroom 1

1817

1818 ### Agent Node 4

1819 Your primary goal is to: find and pick up the apple

1820 To achieve this, you should perform your sibling tasks using a fallback strategy.
 1821 At this level, your sibling tasks are: find and pick up the apple in kitchen
 1822 1, find and pick up the apple in living room 1, and find and pick up the apple
 1823 in bedroom 1.

1823 Your task is to: find and pick up the apple in kitchen 1

1824 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 1825 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
 1826 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
 1827 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
 1828 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
 1829 oven (1), sink (2), stove (1), toaster (1).

1829 Think: I am in kitchen 1. Now, I need to find the apple. The apple is more likely
 1830 to appear on kitchen counter (1, 2), kitchen table (1), or in fridge (1, 2),
 1831 kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8). I can check each place one by one,
 1832 starting from kitchen counter 1, until I find the apple.

1832 Act: go to kitchen counter 1

1833 You arrive at the kitchen counter (1). You see bowl (1, 2, 3, 4, 5), chair (2),
 1834 chicken (1), cutlets (1), dishwashing liquid (1), faucet (2), kitchen cabinet
 1835 (1, 2, 3, 4, 5, 6, 7), kitchen counter (1), plate (1, 2, 3, 4), sink (2),
 washing sponge (1)

1836 Act: go to kitchen counter 2
1837 You arrive at the kitchen counter (2). You see chair (1, 2), kitchen cabinet (5,
1838 6, 7, 8), kitchen counter (1, 2), light switch (2), plate (1, 2, 3, 4)
1839 Act: go to kitchen table 1
1840 You arrive at the kitchen table (1). You see bench (2), book (1), bowl (6),
1841 condiment shaker (1, 2), kitchen table (1)
1842 Act: go to fridge 1
1843 You arrive at the fridge (1). The fridge (1) is closed. You see bench (1), fridge
1844 (1)
1845 Act: open fridge 1
1846 You open fridge. You see bench (1), cupcake (1), fridge (1), pudding (3)
1847 Act: go to fridge 2
1848 You arrive at the fridge (2). The fridge (2) is closed. You see bench (1), cupcake
1849 (1), fridge (1, 2), pudding (3)
1850 Act: open fridge 2
1851 You open fridge. You see apple (2), bench (1), cupcake (1), fridge (1, 2), pudding
1852 (3)
1853 Think: I find apple 2. Now, I need to pick it up.
1854 Act: pick up apple 2
1855 You pick up apple. You hold apple (2).
1856 Think: Now I accomplish the goal.
1857 Act: done
1858
1859 ### Agent Node 5
1860 Your primary goal is to: move the apple on the kitchen table
1861 To achieve this, you should perform your sibling tasks in sequence. At this level,
1862 your sibling tasks are: find and pick up the apple, and find the kitchen
1863 table and put down the apple.
1864 Your task is to: find the kitchen table and put down the apple
1865 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
1866 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
1867 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
1868 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
1869 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
1870 oven (1), sink (2), stove (1), toaster (1). You hold apple (2).
1871 Think: I hold apple 2. Now, I need to find the kitchen table. I can see kitchen
1872 table 1.
1873 Act: go to kitchen table 1
1874 You arrive at the kitchen table (1). You see apple (2), bench (2), book (1), bowl
1875 (6), condiment shaker (1, 2), kitchen table (1) You hold apple (2).
1876 Think: Now, I need to put down apple 2.
1877 Act: put down apple 2
1878 You put down apple on kitchen table
1879 Think: Now I accomplish the goal.
1880 Act: done
1881
1882 ### Agent Node 6
1883 Your primary goal is to: Put the apple, pancake, cupcake, and juice on the kitchen
1884 table.
1885 To achieve this, you should perform your sibling tasks in parallel. At this level,
1886 your sibling tasks are: move the apple on the kitchen table, move the pancake
1887 on the kitchen table, move the cupcake on the kitchen table, and move the
1888 juice on the kitchen table.
1889 Your task is to: move the pancake on the kitchen table
1890 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
1891 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
1892 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
1893 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
1894 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
1895 oven (1), sink (2), stove (1), toaster (1).
1896 Think: To solve the task, I need to accomplish 2 subgoals sequentially. Find and
1897 pick up the pancake, and find the kitchen table and put down the pancake.
1898 Expand:

1890 - control flow: sequence
 1891 - subgoals: find and pick up the pancake, find the kitchen table and put down the
 1892 pancake
 1893
 1894
 1895 ### Agent Node 7
 1896 Your primary goal is to: move the pancake on the kitchen table
 1897 To achieve this, you should perform your sibling tasks in sequence. At this level,
 1898 your sibling tasks are: find and pick up the pancake, and find the kitchen
 1899 table and put down the pancake.
 1900 Your task is to: find and pick up the pancake
 1901 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 1902 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
 1903 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
 1904 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
 1905 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
 1906 oven (1), sink (2), stove (1), toaster (1).
 1907 Think: To solve the task, I need to recall the location of the pancake
 1908 Act: recall location of pancake
 1909 You have not seen pancake before.
 1910 Think: Now, I need to find and pick up the pancake in one of the rooms. The
 1911 pancake is more likely to appear in kitchen 1, living room 1, or bedroom 1. I
 1912 can check each room one by one until I find and pick up the pancake.
 1913 Expand
 1914 - control flow: fallback
 1915 - subgoals: find and pick up the pancake in kitchen 1, find and pick up the
 1916 pancake in living room 1, find and pick up the pancake in bedroom 1
 1917
 1918 ### Agent Node 8
 1919 Your primary goal is to: find and pick up the pancake
 1920 To achieve this, you should perform your sibling tasks using a fallback strategy.
 1921 At this level, your sibling tasks are: find and pick up the pancake in kitchen
 1922 1, find and pick up the pancake in living room 1, and find and pick up the
 1923 pancake in bedroom 1.
 1924 Your task is to: find and pick up the pancake in kitchen 1
 1925 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 1926 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
 1927 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
 1928 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
 1929 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
 1930 oven (1), sink (2), stove (1), toaster (1).
 1931 Think: I am in kitchen 1. Now, I need to find the pancake. The pancake is more
 1932 likely to appear on kitchen counter (1, 2), kitchen table (1), or in fridge
 1933 (1, 2), kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8). I can check each place one
 1934 by one, starting from kitchen counter 1, until I find the pancake.
 1935 Act: go to kitchen counter 1
 1936 You arrive at the kitchen counter (1). You see bowl (1, 2, 3, 4, 5), chair (2),
 1937 chicken (1), cutlets (1), dishwashing liquid (1), faucet (2), kitchen cabinet
 1938 (1, 2, 3, 4, 5, 6, 7), kitchen counter (1), plate (1, 2, 3, 4), sink (2),
 1939 washing sponge (1)
 1940 Act: go to kitchen counter 2
 1941 You arrive at the kitchen counter (2). You see chair (1, 2), kitchen cabinet (5,
 1942 6, 7, 8), kitchen counter (1, 2), light switch (2), plate (1, 2, 3, 4)
 1943 Act: go to kitchen table 1
 1944 You arrive at the kitchen table (1). You see apple (2), bench (2), book (1), bowl
 1945 (6), condiment shaker (1, 2), kitchen table (1)
 1946 Act: go to fridge 1
 1947 You arrive at the fridge (1). The fridge (1) is open. You see bench (1), cupcake
 1948 (1), fridge (1), pudding (3)
 1949 Act: go to fridge 2
 1950 You arrive at the fridge (2). The fridge (2) is open. You see bench (1), cupcake
 1951 (1), fridge (1, 2), pudding (3)
 1952 Act: go to kitchen cabinet 1

1944 You arrive at the kitchen cabinet (1). The kitchen cabinet (1) is closed. You see
1945 chicken (1), cooking pot (1), cutlery knife (1, 2), cutlets (1), dishwashing
1946 liquid (1), faucet (2), frying pan (1), garbage can (1), kitchen cabinet (1,
1947 4), kitchen counter (1), sink (2), stove (1)
1948 Act: go to kitchen cabinet 2
1949 You arrive at the kitchen cabinet (2). The kitchen cabinet (2) is closed. You see
1950 bowl (1, 2, 3, 4, 5), chair (2), chicken (1), cutlery knife (1, 2), cutlets
1951 (1), dishwashing liquid (1), faucet (2), kitchen cabinet (1, 2, 3, 4, 5, 6),
1952 kitchen counter (1), plate (1, 2, 3, 4), sink (2), washing sponge (1)
1953 Act: go to kitchen cabinet 3
1954 You arrive at the kitchen cabinet (3). The kitchen cabinet (3) is closed. You see
1955 bowl (1, 2, 3, 4, 5), chair (1, 2), dishwashing liquid (1), faucet (2),
1956 kitchen cabinet (2, 3, 4, 5, 6, 7), kitchen counter (1, 2), plate (1, 2, 3, 4)
1957 , sink (2), washing sponge (1)
1958 Act: go to kitchen cabinet 4
1959 You arrive at the kitchen cabinet (4). The kitchen cabinet (4) is closed. You see
1960 bowl (1, 2, 3, 4, 5), chicken (1), cooking pot (1), cutlery knife (1, 2),
1961 cutlets (1), dishwashing liquid (1), faucet (2), garbage can (1), kitchen
1962 cabinet (1, 2, 4), kitchen counter (1), sink (2), washing sponge (1)
1963 Act: go to kitchen cabinet 5
1964 You arrive at the kitchen cabinet (5). The kitchen cabinet (5) is closed. You see
1965 bowl (1, 2, 3, 4, 5), chair (1, 2), cutlets (1), dishwashing liquid (1),
1966 faucet (2), kitchen cabinet (2, 3, 4, 5, 6, 7), kitchen counter (1, 2), plate
1967 (1, 2, 3, 4), sink (2), washing sponge (1)
1968 Act: go to kitchen cabinet 6
1969 You arrive at the kitchen cabinet (6). The kitchen cabinet (6) is closed. You see
1970 bowl (1, 2, 3, 4, 5), chair (1, 2), dishwashing liquid (1), faucet (2),
1971 kitchen cabinet (2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), plate (1, 2, 3,
1972 4), sink (2), washing sponge (1)
1973 Act: go to kitchen cabinet 7
1974 You arrive at the kitchen cabinet (7). The kitchen cabinet (7) is closed. You see
1975 bowl (1, 2, 3, 4, 5), chair (1, 2), kitchen cabinet (2, 5, 6, 7, 8), kitchen
1976 counter (1), light switch (2), plate (1, 2, 3, 4), sink (2)
1977 Act: go to kitchen cabinet 8
1978 You arrive at the kitchen cabinet (8). The kitchen cabinet (8) is closed. You see
1979 bench (2), chair (1, 2), kitchen cabinet (5, 6, 7, 8), light switch (2)
1980 Think: I cannot find the pancake in kitchen 1.
1981 Act: failure
1982
1983 ### Agent Node 9
1984 Your primary goal is to: find and pick up the pancake
1985 To achieve this, you should perform your sibling tasks using a fallback strategy.
1986 At this level, your sibling tasks are: find and pick up the pancake in kitchen
1987 1, find and pick up the pancake in living room 1, and find and pick up the
1988 pancake in bedroom 1.
1989 Your task is to: find and pick up the pancake in living room 1
1990 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
1991 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
1992 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
1993 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
1994 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
1995 oven (1), sink (2), stove (1), toaster (1).
1996 Think: I am in kitchen 1. Now, I need to go to living room 1.
1997 Act: go to living room 1
1998 You move to the living room (1). Looking quickly around the room, you see
1999 bookshelf (3), chair (3), closet (1), coffee table (1), computer (1), desk (1)
2000 , garbage can (2), nightstand (1), sofa (1), tv (1).
2001 Think: Now, I need to find the pancake. The pancake is more likely to appear on
2002 coffee table (1), desk (1), or in closet (1). I can check each place one by
2003 one, starting from coffee table 1, until I find the pancake.
2004 Act: go to coffee table 1
2005 You arrive at the coffee table (1). You see alcohol (2), bananas (1), book (3, 6),
2006 bowl (7), cell phone (1, 2), coffee table (1), juice (1), lime (1), mug (1),
2007 plum (1, 2), sofa (1)

1998 Act: go to desk 1
1999 You arrive at the desk (1). You see book (7), chair (3), computer (1), desk (1),
2000 nightstand (1), water glass (2)
2001 Act: go to closet 1
2002 You arrive at the closet (1). The closet (1) is closed. You see closet (1), light
2003 switch (3)
2004 Act: open closet 1
2005 You open closet. You see closet (1), light switch (3)
2006 Think: I cannot find the pancake in living room 1.
2007 Act: failure
2008
2009 ### Agent Node 10
2010 Your primary goal is to: find and pick up the pancake
2011 To achieve this, you should perform your sibling tasks using a fallback strategy.
2012 At this level, your sibling tasks are: find and pick up the pancake in kitchen
2013 1, find and pick up the pancake in living room 1, and find and pick up the
2014 pancake in bedroom 1.
2015 Your task is to: find and pick up the pancake in bedroom 1
2016 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
2017 (1), living room (1). You are in the middle of a living room (1). Looking
2018 quickly around the room, you see bookshelf (3), chair (3), closet (1), coffee
2019 table (1), computer (1), desk (1), garbage can (2), nightstand (1), sofa (1),
2020 tv (1).
2021 Think: I am in living room 1. Now, I need to go to bedroom 1.
2022 Act: go to bedroom 1
2023 You move to the bedroom (1). Looking quickly around the room, you see bed (1),
2024 bookshelf (4), cabinet (1), chair (4), closet (2), coffee table (2), computer
2025 (2), desk (2, 3), garbage can (3).
2026 Think: Now, I need to find the pancake. The pancake is more likely to appear on
2027 bed (1), coffee table (2), desk (2, 3), or in cabinet (1). I can check each
2028 place one by one, starting from bed 1, until I find the pancake.
2029 Act: go to bed 1
2030 You arrive at the bed (1). You see bed (1), candy bar (1), coffee table (2),
2031 pancake (2)
2032 Think: I find pancake 2. Now, I need to pick it up.
2033 Act: pick up pancake 2
2034 You pick up pancake. You hold pancake (2).
2035 Think: Now I accomplish the goal.
2036 Act: done
2037
2038 ### Agent Node 11
2039 Your primary goal is to: move the pancake on the kitchen table
2040 To achieve this, you should perform your sibling tasks in sequence. At this level,
2041 your sibling tasks are: find and pick up the pancake, and find the kitchen
2042 table and put down the pancake.
2043 Your task is to: find the kitchen table and put down the pancake
2044 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
2045 (1), living room (1). You are in the middle of a bedroom (1). Looking quickly
2046 around the room, you see bed (1), bookshelf (4), cabinet (1), chair (4),
2047 closet (2), coffee table (2), computer (2), desk (2, 3), garbage can (3). You
2048 hold pancake (2).
2049 Think: I hold pancake 2. Now, I need to find the kitchen table. The kitchen table
2050 is more likely to appear in kitchen 1.
2051 Act: go to kitchen 1
You move to the kitchen (1). Looking quickly around the room, you see bench (1, 2)
, bookshelf (1, 2), chair (1, 2), dishwasher (1), faucet (2), fridge (1, 2),
garbage can (1), kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1,
2), kitchen table (1), microwave oven (1), sink (2), stove (1), toaster (1).
You hold pancake (2).
Think: I can see kitchen table 1.
Act: go to kitchen table 1

2052 You arrive at the kitchen table (1). You see apple (2), bench (2), book (1), bowl
2053 (6), condiment shaker (1, 2), kitchen table (1), pancake (2) You hold pancake
2054 (2).
2055 Think: Now, I need to put down pancake 2.
2056 Act: put down pancake 2
2057 You put down pancake on kitchen table
2058 Think: Now I accomplish the goal.
2059 Act: done
2060
2061 ### Agent Node 11
2062 Your primary goal is to: Put the apple, pancake, cupcake, and juice on the kitchen
2063 table.
2064 To achieve this, you should perform your sibling tasks in parallel. At this level,
2065 your sibling tasks are: move the apple on the kitchen table, move the pancake
2066 on the kitchen table, move the cupcake on the kitchen table, and move the
2067 juice on the kitchen table.
2068 Your task is to: move the cupcake on the kitchen table
2069 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
2070 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
2071 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
2072 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
2073 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
2074 oven (1), sink (2), stove (1), toaster (1).
2075 Think: To solve the task, I need to accomplish 2 subgoals sequentially. Find and
2076 pick up the cupcake, and find the kitchen table and put down the cupcake.
2077 Expand:
2078 - control flow: sequence
2079 - subgoals: find and pick up the cupcake, find the kitchen table and put down the
2080 cupcake
2081
2082 ### Agent Node 12
2083 Your primary goal is to: move the cupcake on the kitchen table
2084 To achieve this, you should perform your sibling tasks in sequence. At this level,
2085 your sibling tasks are: find and pick up the cupcake, and find the kitchen
2086 table and put down the cupcake.
2087 Your task is to: find and pick up the cupcake
2088 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
2089 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
2090 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
2091 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
2092 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
2093 oven (1), sink (2), stove (1), toaster (1).
2094 Think: To solve the task, I need to recall the location of the cupcake
2095 Act: recall location of cupcake
2096 You saw cupcake 1 near fridge 2 in kitchen 1.
2097 Think: Now, I need to go to fridge 2.
2098 Act: go to fridge 2
2099 You arrive at the fridge (2). The fridge (2) is open. You see bench (1), cupcake
2100 (1), fridge (1, 2), pudding (3)
2101 Think: I find cupcake 1. Now, I need to pick it up.
2102 Act: pick up cupcake 1
2103 You pick up cupcake. You hold cupcake (1).
2104 Think: Now I accomplish the goal.
2105 Act: done
2106
2107 ### Agent Node 13
2108 Your primary goal is to: move the cupcake on the kitchen table
2109 To achieve this, you should perform your sibling tasks in sequence. At this level,
2110 your sibling tasks are: find and pick up the cupcake, and find the kitchen
2111 table and put down the cupcake.
2112 Your task is to: find the kitchen table and put down the cupcake

2106 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
2107 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
2108 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
2109 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
2110 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
2111 oven (1), sink (2), stove (1), toaster (1). You hold cupcake (1).
2112 Think: I hold cupcake 1. Now, I need to find the kitchen table. I can see kitchen
2113 table 1.
2114 Act: go to kitchen table 1
2115 You arrive at the kitchen table (1). You see apple (2), bench (2), book (1), bowl
2116 (6), condiment shaker (1, 2), cupcake (1), kitchen table (1), pancake (2) You
2117 hold cupcake (1).
2118 Think: Now, I need to put down cupcake 1.
2119 Act: put down cupcake 1
2120 You put down cupcake on kitchen table
2121 Think: Now I accomplish the goal.
2122 Act: done

2123 ### Agent Node 14
2124 Your primary goal is to: Put the apple, pancake, cupcake, and juice on the kitchen
2125 table.
2126 To achieve this, you should perform your sibling tasks in parallel. At this level,
2127 your sibling tasks are: move the apple on the kitchen table, move the pancake
2128 on the kitchen table, move the cupcake on the kitchen table, and move the
2129 juice on the kitchen table.
2130 Your task is to: move the juice on the kitchen table
2131 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
2132 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
2133 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
2134 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
2135 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
2136 oven (1), sink (2), stove (1), toaster (1).
2137 Think: To solve the task, I need to accomplish 2 subgoals sequentially. Find and
2138 pick up the juice, and find the kitchen table and put down the juice.
2139 Expand:
2140 - control flow sequence
2141 - subgoals: find and pick up the juice, find the kitchen table and put down the
2142 juice

2143 ### Agent Node 15
2144 Your primary goal is to: move the juice on the kitchen table
2145 To achieve this, you should perform your sibling tasks in sequence. At this level,
2146 your sibling tasks are: find and pick up the juice, and find the kitchen
2147 table and put down the juice.
2148 Your task is to: find and pick up the juice
2149 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
2150 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
2151 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
2152 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
2153 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
2154 oven (1), sink (2), stove (1), toaster (1).
2155 Think: To solve the task, I need to recall the location of the juice.
2156 Act: recall location of juice
2157 You saw juice 1 near coffee table 1 in living room 1.
2158 Think: I can find juice 1 near coffee table 1.
2159 Act: go to living room 1
2160 You move to the living room (1). Looking quickly around the room, you see
2161 bookshelf (3), chair (3), closet (1), coffee table (1), computer (1), desk (1)
2162 , garbage can (2), nightstand (1), sofa (1), tv (1).
2163 Act: go to coffee table 1
2164 You arrive at the coffee table (1). You see alcohol (2), bananas (1), book (3, 6),
2165 bowl (7), cell phone (1, 2), coffee table (1), juice (1), lime (1), mug (1),
2166 plum (1, 2), sofa (1)

2160 Think: I find juice 1. Now I need to pick it up.
 2161 Act: pick up juice 1
 2162 You pick up juice. You hold juice (1).
 2163 Think: Now I accomplish the goal.
 2164 Act: done
 2165
 2166 ### Agent 16
 2167 Your primary goal is to: move the juice on the kitchen table
 2168 To achieve this, you should perform your sibling tasks in sequence. At this level,
 2169 your sibling tasks are: find and pick up the juice, and find the kitchen
 2170 table and put down the juice.
 2171 Your task is to: find the kitchen table and put down the juice
 2172 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 2173 (1), living room (1). You are in the middle of a living room (1). Looking
 2174 quickly around the room, you see bookshelf (3), chair (3), closet (1), coffee
 2175 table (1), computer (1), desk (1), garbage can (2), nightstand (1), sofa (1),
 2176 tv (1). You hold juice (1).
 2177 Think: I hold juice 1. Now, I need to find the kitchen table. The kitchen table is
 2178 more likely to appear in kitchen 1.
 2179 Act: go to kitchen 1
 2180 You move to the kitchen (1). Looking quickly around the room, you see bench (1, 2)
 2181 , bookshelf (1, 2), chair (1, 2), dishwasher (1), faucet (2), fridge (1, 2),
 2182 garbage can (1), kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1,
 2183 2), kitchen table (1), microwave oven (1), sink (2), stove (1), toaster (1).
 2184 You hold juice (1).
 2185 Think: I can see kitchen table 1.
 2186 Act: go to kitchen table 1
 2187 You arrive at the kitchen table (1). You see apple (2), bench (2), book (1), bowl
 2188 (6), condiment shaker (1, 2), cupcake (1), juice (1), kitchen table (1),
 2189 pancake (2) You hold juice (1).
 2190 Think: Now, I need to put down juice 1.
 2191 Act: put down juice 1
 2192 You put down juice on kitchen table
 2193 Think: Now I accomplish the goal.
 2194 Act: done

Listing 12: Failure trajectory of ReAct on the WAH-NL dataset using LLaMA 3 70B.

2193 Your task is to: Put the apple, pancake, cupcake, and juice on the kitchen table.
 2194 You are in the house, and there are 4 rooms: bathroom (1), bedroom (1), kitchen
 2195 (1), living room (1). You are in the middle of a kitchen (1). Looking quickly
 2196 around the room, you see bench (1, 2), bookshelf (1, 2), chair (1, 2),
 2197 dishwasher (1), faucet (2), fridge (1, 2), garbage can (1), kitchen cabinet
 2198 (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), kitchen table (1), microwave
 2199 oven (1), sink (2), stove (1), toaster (1).
 2200 Think: To solve the task, I need to find and pick up the apple, then put it on the
 2201 kitchen table. And I need to find and pick up the pancake, then put it on the
 2202 kitchen table. And I need to find and pick up the cupcake, then put it on the
 2203 kitchen table. And I need to find and pick up the juice, then put it on the
 2204 kitchen table.
 2205 Think: First, I need to find the apple. The apple is more likely to appear in
 2206 kitchen 1.
 2207 Act: go to kitchen 1
 2208 You move to the kitchen (1). Looking quickly around the room, you see bench (1, 2)
 2209 , bookshelf (1, 2), chair (1, 2), dishwasher (1), faucet (2), fridge (1, 2),
 2210 garbage can (1), kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8), kitchen counter (1,
 2211 2), kitchen table (1), microwave oven (1), sink (2), stove (1), toaster (1).
 2212 Think: Now, I need to find the apple. The apple is more likely to appear on
 2213 kitchen conter (1, 2), kitchen table (1), or in fridge (1, 2), kitchen cabinet
 2214 (1, 2, 3, 4, 5, 6, 7, 8). I can check each place one by one, starting from
 2215 kitchen counter 1, until I find the apple.
 2216 Act: go to kitchen counter 1
 2217 You arrive at the kitchen counter (1). You see bowl (1, 2, 3, 4, 5), chair (2),
 2218 chicken (1), cutlets (1), dishwashing liquid (1), faucet (2), kitchen cabinet

2214 (1, 2, 3, 4, 5, 6, 7), kitchen counter (1), plate (1, 2, 3, 4), sink (2),
 2215 washing sponge (1)

2216 Act: go to kitchen counter 2

2217 You arrive at the kitchen counter (2). You see chair (1, 2), kitchen cabinet (5,
 2218 6, 7, 8), kitchen counter (1, 2), light switch (2), plate (1, 2, 3, 4)

2219 Act: go to kitchen table 1

2220 You arrive at the kitchen table (1). You see bench (2), book (1), bowl (6),
 2221 condiment shaker (1, 2), kitchen table (1)

2222 Act: go to fridge 1

2223 You arrive at the fridge (1). The fridge (1) is closed. You see bench (1), fridge
 2224 (1)

2225 Act: open fridge 1

2226 You open fridge. You see bench (1), cupcake (1), fridge (1), pudding (3)
 2227 Think: I find cupcake 1. Now, I need to pick it up.

2228 Act: pick up cupcake 1

2229 You pick up cupcake. You hold cupcake (1).
 2230 Think: I hold cupcake 1. Now, I need to put it on the kitchen table.

2231 Act: go to kitchen table 1

2232 You arrive at the kitchen table (1). You see bench (2), book (1), bowl (6),
 2233 condiment shaker (1, 2), cupcake (1), kitchen table (1) You hold cupcake (1).
 2234 Think: Now, I need to put down cupcake 1.

2235 Act: put down cupcake 1

2236 You put down cupcake on kitchen table
 2237 Think: I put down cupcake 1 on kitchen table 1. Now, I need to recall the location
 2238 of the pancake.

2239 Act: recall location of pancake

2240 You have not seen pancake before.
 2241 Think: Now, I need to find the pancake. The pancake is more likely to appear in
 2242 kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8). I can check each place one by one,
 2243 starting from kitchen cabinet 1, until I find the pancake.

2244 Act: go to kitchen cabinet 1

2245 You arrive at the kitchen cabinet (1). The kitchen cabinet (1) is closed. You see
 2246 chicken (1), cooking pot (1), cutlery knife (1, 2), cutlets (1), dishwashing
 2247 liquid (1), faucet (2), frying pan (1), garbage can (1), kitchen cabinet (1,
 2248 4), kitchen counter (1), sink (2), stove (1)

2249 Act: open kitchen cabinet 1

2250 You open kitchen cabinet. You see chicken (1), cooking pot (1), cutlery knife (1,
 2251 2), cutlets (1), dishwashing liquid (1), faucet (2), frying pan (1), garbage
 2252 can (1), kitchen cabinet (1, 4), kitchen counter (1), sink (2), stove (1)

2253 Act: go to kitchen cabinet 2

2254 You arrive at the kitchen cabinet (2). The kitchen cabinet (2) is closed. You see
 2255 bowl (1, 2, 3, 4, 5), chair (2), chicken (1), cutlery knife (1, 2), cutlets
 2256 (1), dishwashing liquid (1), faucet (2), kitchen cabinet (1, 2, 3, 4, 5, 6),
 2257 kitchen counter (1), plate (1, 2, 3, 4), sink (2), washing sponge (1)

2258 Act: open kitchen cabinet 2

2259 You open kitchen cabinet. You see bowl (1, 2, 3, 4, 5), chair (2), chicken (1),
 2260 cutlery knife (1, 2), cutlets (1), dishwashing liquid (1), faucet (2), kitchen
 2261 cabinet (1, 2, 3, 4, 5, 6), kitchen counter (1), plate (1, 2, 3, 4), sink (2)
 2262 , washing sponge (1)

2263 Act: go to kitchen cabinet 3

2264 You arrive at the kitchen cabinet (3). The kitchen cabinet (3) is closed. You see
 2265 bowl (1, 2, 3, 4, 5), chair (1, 2), dishwashing liquid (1), faucet (2),
 2266 kitchen cabinet (2, 3, 4, 5, 6, 7), kitchen counter (1, 2), plate (1, 2, 3, 4)
 2267 , sink (2), washing sponge (1)

2268 Act: open kitchen cabinet 3

2269 You open kitchen cabinet. You see bowl (1, 2, 3, 4, 5), chair (1, 2), dishwashing
 2270 liquid (1), faucet (2), kitchen cabinet (2, 3, 4, 5, 6, 7), kitchen counter
 2271 (1, 2), plate (1, 2, 3, 4), sink (2), washing sponge (1)

2272 Act: go to kitchen cabinet 4

2273 You arrive at the kitchen cabinet (4). The kitchen cabinet (4) is closed. You see
 2274 bowl (1, 2, 3, 4, 5), chicken (1), cooking pot (1), cutlery knife (1, 2),
 2275 cutlets (1), dishwashing liquid (1), faucet (2), garbage can (1), kitchen
 2276 cabinet (1, 2, 4), kitchen counter (1), sink (2), washing sponge (1)

2277 Act: open kitchen cabinet 4

2268 You open kitchen cabinet. You see bowl (1, 2, 3, 4, 5), chicken (1), cooking pot
 2269 (1), cutlery knife (1, 2), cutlets (1), dishwashing liquid (1), faucet (2),
 2270 garbage can (1), kitchen cabinet (1, 2, 4), kitchen counter (1), sink (2),
 2271 washing sponge (1)
 2272 Act: go to kitchen cabinet 5
 2273 You arrive at the kitchen cabinet (5). The kitchen cabinet (5) is closed. You see
 2274 bowl (1, 2, 3, 4, 5), chair (1, 2), cutlets (1), dishwashing liquid (1),
 2275 faucet (2), kitchen cabinet (2, 3, 4, 5, 6, 7), kitchen counter (1, 2), plate
 (1, 2, 3, 4), sink (2), washing sponge (1)
 2276 Act: open kitchen cabinet 5
 2277 You open kitchen cabinet. You see bowl (1, 2, 3, 4, 5), chair (1, 2), cutlets (1),
 2278 dishwashing liquid (1), faucet (2), kitchen cabinet (2, 3, 4, 5, 6, 7),
 2279 kitchen counter (1, 2), plate (1, 2, 3, 4), pudding (2), sink (2), washing
 sponge (1)
 2280 Think: I find pudding 2. Now, I need to pick it up.
 2281 Act: pick up pudding 2
 2282 You pick up pudding. You hold pudding (2).
 2283 Think: I hold pudding 2. Now, I need to put it on the kitchen table.
 2284 Act: go to kitchen table 1
 2285 You arrive at the kitchen table (1). You see bench (2), book (1), bowl (6),
 2286 condiment shaker (1, 2), cupcake (1), kitchen table (1), pudding (2) You hold
 pudding (2).
 2287 Think: Now, I need to put down pudding 2.
 2288 Act: put down pudding 2
 2289 You put down pudding on kitchen table
 2290 Think: I put down pudding 2 on kitchen table 1. Now, I need to recall the location
 of the juice.
 2291 Act: recall location of juice
 2292 You have not seen juice before.
 2293 Think: Now, I need to find the juice. The juice is more likely to appear in
 2294 kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8). I can check each place one by one,
 starting from kitchen cabinet 1, until I find the juice.
 2295 Act: go to kitchen cabinet 6
 2296 You arrive at the kitchen cabinet (6). The kitchen cabinet (6) is closed. You see
 2297 bowl (1, 2, 3, 4, 5), chair (1, 2), dishwashing liquid (1), faucet (2),
 2298 kitchen cabinet (2, 3, 4, 5, 6, 7, 8), kitchen counter (1, 2), plate (1, 2, 3,
 4), sink (2), washing sponge (1)
 2299 Act: open kitchen cabinet 6
 2300 You open kitchen cabinet. You see bowl (1, 2, 3, 4, 5), chair (1, 2), dishwashing
 2301 liquid (1), faucet (2), kitchen cabinet (2, 3, 4, 5, 6, 7, 8), kitchen counter
 (1, 2), plate (1, 2, 3, 4), sink (2), washing sponge (1)
 2302 Act: go to kitchen cabinet 7
 2303 You arrive at the kitchen cabinet (7). The kitchen cabinet (7) is closed. You see
 2304 bowl (1, 2, 3, 4, 5), chair (1, 2), kitchen cabinet (2, 3, 4, 5, 6, 7, 8),
 2305 kitchen counter (1), light switch (2), plate (1, 2, 3, 4), sink (2)
 2306 Act: open kitchen cabinet 7
 2307 You open kitchen cabinet. You see apple (1), bowl (1, 2, 3, 4, 5), chair (1, 2),
 2308 kitchen cabinet (2, 3, 4, 5, 6, 7, 8), kitchen counter (1), light switch (2),
 pancake (1), plate (1, 2, 3, 4), sink (2)
 2309 Think: I find apple 1. Now, I need to pick it up.
 2310 Act: pick up apple 1
 2311 You pick up apple. You hold apple (1).
 2312 Think: I hold apple 1. Now, I need to put it on the kitchen table.
 2313 Act: go to kitchen table 1
 2314 You arrive at the kitchen table (1). You see apple (1), bench (2), book (1), bowl
 (6), condiment shaker (1, 2), cupcake (1), kitchen table (1), pudding (2) You
 2315 hold apple (1).
 2316 Think: Now, I need to put down apple 1.
 2317 Act: put down apple 1
 2318 You put down apple on kitchen table
 2319 Think: I put down apple 1 on kitchen table 1. Now, I need to recall the location
 of the pancake.
 2320 Act: recall location of pancake
 2321 You saw pancake 1 near kitchen cabinet 7 in kitchen 1.
 Act: go to kitchen cabinet 7

2322 You arrive at the kitchen cabinet (7). The kitchen cabinet (7) is open. You see
 2323 bowl (1, 2, 3, 4, 5), chair (1, 2), kitchen cabinet (2, 3, 4, 5, 6, 7, 8),
 2324 kitchen counter (1), light switch (2), pancake (1), plate (1, 2, 3, 4), sink
 2325 (2)
 2326 Think: I find pancake 1. Now, I need to pick it up.
 2327 Act: pick up pancake 1
 2328 You pick up pancake. You hold pancake (1).
 2329 Think: I hold pancake 1. Now, I need to put it on the kitchen table.
 2330 Act: go to kitchen table 1
 2331 You arrive at the kitchen table (1). You see apple (1), bench (2), book (1), bowl
 2332 (6), condiment shaker (1, 2), cupcake (1), kitchen table (1), pancake (1),
 2333 pudding (2) You hold pancake (1).
 2334 Think: Now, I need to put down pancake 1.
 2335 Act: put down pancake 1
 2336 You put down pancake on kitchen table
 2337 Think: I put down pancake 1 on kitchen table 1. Now, I need to recall the location
 2338 of the juice.
 2339 Act: recall location of juice
 2340 You have not seen juice before.
 2341 Think: Now, I need to find the juice. The juice is more likely to appear in
 2342 kitchen cabinet (1, 2, 3, 4, 5, 6, 7, 8). I can check each place one by one,
 2343 starting from kitchen cabinet 1, until I find the juice.
 2344 Act: go to kitchen cabinet 8
 2345 You arrive at the kitchen cabinet (8). The kitchen cabinet (8) is closed. You see
 2346 bench (2), chair (1, 2), kitchen cabinet (5, 6, 7, 8), light switch (2)
 2347 Act: open kitchen cabinet 8
 2348 You open kitchen cabinet. You see bench (2), chair (1, 2), kitchen cabinet (5, 6,
 2349 7, 8), light switch (2)
 2350 Think: I cannot find juice in kitchen 1. The juice is more likely to appear in
 2351 fridge 1, 2.
 2352 Act: go to fridge 1
 2353 You arrive at the fridge (1). The fridge (1) is open. You see bench (1), fridge
 2354 (1), pudding (3)
 2355 Act: go to fridge 2
 2356 You arrive at the fridge (2). The fridge (2) is closed. You see bench (1), fridge
 2357 (1, 2), pudding (3)
 2358 Act: open fridge 2
 2359 You open fridge. You see apple (2), bench (1), fridge (1, 2), pudding (3)
 2360 Think: I find apple 2. Now, I need to pick it up.
 2361 Act: pick up apple 2
 2362 You pick up apple. You hold apple (2).
 2363 Think: I hold apple 2. Now, I need to put it on the kitchen table.
 2364 Act: go to kitchen table 1
 2365 You arrive at the kitchen table (1). You see apple (1, 2), bench (2), book (1),
 2366 bowl (6), condiment shaker (1, 2), cupcake (1), kitchen table (1), pancake (1)
 2367 , pudding (2) You hold apple (2).
 2368 Think: Now, I need to put down apple 2.
 2369 Act: put down apple 2
 2370 You put down apple on kitchen table
 2371 Think: I put down apple 2 on kitchen table 1. Now, I need to recall the location
 2372 of the juice.
 2373 Act: recall location of juice
 2374 You have not seen juice before.
 2375 Failure: maximum context window size of 8192

2368

2369

2370 H.2 ALFRED EXAMPLE CASES

2371

2372 Listing 13: Success trajectory of ReAcTree on the ALFRED dataset using LLaMA 3 70B.

2373

```
2373 ### Agent Node 1
```

2374

```
2374 Your task is to: Put the cooked tomato slice on the kitchen island
```

2375

```
2375 You are in the house, and you arrive at the kitchen, Looking quickly around the
room, you see Toaster (1), StoveBurner (1, 2, 3, 4, 5, 6), SinkBasin (1),
Microwave (1), GarbageCan (1), Fridge (1), Drawer (1, 2, 3, 4, 5, 6, 7, 8, 9,
```

2376 10, 11, 12, 13), CounterTop (1, 2), CoffeeMachine (1), Cabinet (1, 2, 3, 4, 5,
2377 6, 7, 8, 9).

2378 Think: To solve the task, I need to accomplish 8 subgoals sequentially. find and
2379 pick up the knife, find and slice and then pick up the tomato, find and open
2380 the microwave, find the microwave and put down the tomato, find and close and
2381 turn on the microwave, find and turn off and open microwave, find and pick up
2382 tomato in microwave, and find a countertop and put down the tomato on the
2383 countertop.

2384 Expand:
2385 - control flow: sequence
2386 - subgoals: find and pick up the knife, find and slice and then pick up the tomato
2387 , find and open the microwave, find the microwave and put down the tomato,
2388 find and close and turn on the microwave, find and turn off and open microwave
2389 , find and pick up tomato in microwave, and find a countertop and put down the
2390 tomato on the countertop.

2391 ### Agent Node 2
2392 Your primary goal is to: Put the cooked tomato slice on the kitchen island
2393 To achieve this, you should perform your sibling tasks in sequence. At this level,
2394 your sibling tasks are: find and pick up the knife, find and slice and then
2395 pick up the tomato, find and open the microwave, find the microwave and put
2396 down the tomato, find and close and turn on the microwave, find and turn off
2397 and open microwave, find and pick up tomato in microwave, and and find a
2398 countertop and put down the tomato on the countertop.
2399 Your task is to: find and pick up the knife
2400 You are in the house, and you arrive at the kitchen, Looking quickly around the
2401 room, you see Toaster (1), StoveBurner (1, 2, 3, 4, 5, 6), SinkBasin (1),
2402 Microwave (1), GarbageCan (1), Fridge (1), Drawer (1, 2, 3, 4, 5, 6, 7, 8, 9,
2403 10, 11, 12, 13), CounterTop (1, 2), CoffeeMachine (1), Cabinet (1, 2, 3, 4, 5,
2404 6, 7, 8, 9).

2405 Think: To solve the task, I need to recall location of knife.
2406 Think: Now I need to find Knife. A knife is more likely to appear in CounterTop
2407 (1-2), SinkBasin (1), Cabinet (1-9). I can check one by one, starting with
2408 CounterTop (1).
2409 Act: go to CounterTop (1)
2410 You arrive at CounterTop (1). You see Window (2), Tomato (1), Sink (1), SinkBasin
2411 (1), Lettuce (2), Fork (1), Faucet (1), Egg (2), DishSponge (1), CounterTop
2412 (1), Cabinet (1, 8).
2413 Act: go to CounterTop (2)
2414 You arrive at CounterTop (2). You see Tomato (2), SoapBottle (2), Pot (1),
2415 PepperShaker (2), Mug (2), Knife (1, 2), Kettle (2), CounterTop (1, 2),
2416 CellPhone (2), Bread (1, 2).
2417 Think: Now I find Knife (1, 2). Next, I need to pick it up.
2418 Act: pick up Knife (1)
2419 You pick up Knife (1). You see Tomato (2), SoapBottle (2), Pot (1), PepperShaker
2420 (2), Mug (2), Knife (1, 2), Kettle (2), CounterTop (1, 2), CellPhone (2),
2421 Bread (1, 2). I am holding Knife (1) now.
2422 Think: Now I find and pick up Knife (1)
2423 Act: done

2424 ### Agent Node 3
2425 Your primary goal is to: Put the cooked tomato slice on the kitchen island
2426 To achieve this, you should perform your sibling tasks in sequence. At this level,
2427 your sibling tasks are: find and pick up the knife, find and slice and then
2428 pick up the tomato, find and open the microwave, find the microwave and put
2429 down the tomato, find and close and turn on the microwave, find and turn off
2430 and open microwave, find and pick up tomato in microwave, and and find a
2431 countertop and put down the tomato on the countertop.
2432 Your task is to: find and slice and then pick up the tomato
2433 You are in the house, and you arrive at the kitchen, Looking quickly around the
2434 room, you see Toaster (1), StoveBurner (1, 2, 3, 4, 5, 6), SinkBasin (1),
2435 Microwave (1), GarbageCan (1), Fridge (1), Drawer (1, 2, 3, 4, 5, 6, 7, 8, 9,

2430 10, 11, 12, 13), CounterTop (1, 2), CoffeeMachine (1), Cabinet (1, 2, 3, 4, 5,
2431 6, 7, 8, 9). I am holding Knife (1) now.

2432 Think: To solve the task, I need to recall location of tomato.

2433 Think: I saw Tomato (1) near SinkBasin (1). I need to go to SinkBasin (1), then
2434 slice the Tomato (1).

2435 Act: go to SinkBasin (1)

2436 You arrive at SinkBasin (1). You see Tomato (1), Sink (1), SinkBasin (1), Lettuce
2437 (2), Knife (1), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1),
2438 CounterTop (1), Cabinet (1, 8). I am holding Knife (1) now.

2438 Think: Now I find a part of the Tomato (1). I need to go in front of the Tomato
2439 (1) to see the whole thing.

2440 Act: go to Tomato (1)

2441 You see Tomato (1), Sink (1), SinkBasin (1), Lettuce (2), Knife (1), Fork (1),
2442 Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1), CounterTop (1), Cabinet
2443 (1, 8). I am holding Knife (1) now.

2443 Act: slice Tomato (1)

2444 You slice Tomato (1). You see Tomato (9, 10), Sink (1), SinkBasin (1), Lettuce (2)
2445 , Knife (1), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1),
2446 CounterTop (1), Cabinet (1, 8). I am holding Knife (1) now.

2447 Think: Now I slice the Tomato (1), and I do not need to hold Knife (1). I need to
2448 drop the Knife (1).

2448 Act: put down Knife (1)

2449 You put down failed, drop it. You see Tomato (9, 10), Sink (1), SinkBasin (1),
2450 Lettuce (2), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1),
2451 CounterTop (1), Cabinet (1, 8).

2451 Think: Now I put down Knife (1). Next, I need to pick up the Tomato (9).

2452 Act: pick up Tomato (9)

2453 You pick up Tomato (9). You see Tomato (7, 8, 9, 10), Sink (1), SinkBasin (1),
2454 Lettuce (2), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1),
2455 CounterTop (1), Cabinet (1, 8). I am holding Tomato (9) now.

2456 Think: Now I pick up Tomato (9).

2456 Act: done

2457

2458

2459 ### Agent Node 4

2460 Your primary goal is to: Put the cooked tomato slice on the kitchen island

2461 To achieve this, you should perform your sibling tasks in sequence. At this level,
2462 your sibling tasks are: find and pick up the knife, find and slice and then
2463 pick up the tomato, find and open the microwave, find the microwave and put
2464 down the tomato, find and close and turn on the microwave, find and turn off
2465 and open microwave, find and pick up tomato in microwave, and and find a
2466 countertop and put down the tomato on the countertop.

2466 Your task is to: find and open the microwave

2467 You are in the house, and you arrive at the kitchen, Looking quickly around the
2468 room, you see Toaster (1), StoveBurner (1, 2, 3, 4, 5, 6), SinkBasin (1),
2469 Microwave (1), GarbageCan (1), Fridge (1), Drawer (1, 2, 3, 4, 5, 6, 7, 8, 9,
2470 10, 11, 12, 13), CounterTop (1, 2), CoffeeMachine (1), Cabinet (1, 2, 3, 4, 5,
2471 6, 7, 8, 9). I am holding Tomato (9) now.

2471 Act: go to Microwave (1)

2472 You arrive at Microwave (1). You see Window (1), Tomato (9), Toaster (1),
2473 Microwave (1), Drawer (4), CounterTop (1). Microwave (1) is closed. I am
2474 holding Tomato (9) now.

2474 Act: open Microwave (1)

2475 You open Microwave (1). You see Window (1), Tomato (9), Toaster (1), Microwave (1)
2476 , Egg (1), Drawer (4), CounterTop (1). I am holding Tomato (9) now.

2476 Think: Now I find and open the Microwave (1).

2477 Act: done

2478

2479

2480 ### Agent Node 5

2481 Your primary goal is to: Put the cooked tomato slice on the kitchen island

2482 To achieve this, you should perform your sibling tasks in sequence. At this level,
2483 your sibling tasks are: find and pick up the knife, find and slice and then
2484 pick up the tomato, find and open the microwave, find the microwave and put
2485 down the tomato, find and close and turn on the microwave, find and turn off

2484 and open microwave, find and pick up tomato in microwave, and and find a
 2485 countertop and put down the tomato on the countertop.
 2486 Your task is to: find the microwave and put down the tomato
 2487 You are in the house, and you arrive at the kitchen, Looking quickly around the
 2488 room, you see Toaster (1), StoveBurner (1, 2, 3, 4, 5, 6), SinkBasin (1),
 2489 Microwave (1), GarbageCan (1), Fridge (1), Drawer (1, 2, 3, 4, 5, 6, 7, 8, 9,
 2490 10, 11, 12, 13), CounterTop (1, 2), CoffeeMachine (1), Cabinet (1, 2, 3, 4, 5,
 2491 6, 7, 8, 9). I am holding Tomato (9) now.
 2492 Think: To solve the task, I need to go to Microwave (1) and put down the Tomato
 2493 (9).
 2494 Act: go to Microwave (1)
 2495 You arrive at Microwave (1). You see Window (1), Tomato (9), Toaster (1),
 2496 Microwave (1), Egg (1), Drawer (4), CounterTop (1). Microwave (1) is open. I
 2497 am holding Tomato (9) now.
 2498 Think: Now I find a Microwave (1). The Microwave (1) is already open, so I need to
 2499 put the Tomato (9) down.
 2500 Act: put down Tomato (9)
 2501 You put Tomato (9) on Microwave (1). You see Window (1), Tomato (9), Toaster (1),
 2502 Microwave (1), Egg (1), Drawer (4), CounterTop (1).
 2503 Think: Now I put down Tomato (9) in the Microwave (1).
 2504 Act: done

2505 ### Agent Node 6
 2506 Your primary goal is to: Put the cooked tomato slice on the kitchen island
 2507 To achieve this, you should perform your sibling tasks in sequence. At this level,
 2508 your sibling tasks are: find and pick up the knife, find and slice and then
 2509 pick up the tomato, find and open the microwave, find the microwave and put
 2510 down the tomato, find and close and turn on the microwave, find and turn off
 2511 and open microwave, find and pick up tomato in microwave, and and find a
 2512 countertop and put down the tomato on the countertop.
 2513 Your task is to: find and close and turn on the microwave
 2514 You are in the house, and you arrive at the kitchen, Looking quickly around the
 2515 room, you see Toaster (1), StoveBurner (1, 2, 3, 4, 5, 6), SinkBasin (1),
 2516 Microwave (1), GarbageCan (1), Fridge (1), Drawer (1, 2, 3, 4, 5, 6, 7, 8, 9,
 2517 10, 11, 12, 13), CounterTop (1, 2), CoffeeMachine (1), Cabinet (1, 2, 3, 4, 5,
 2518 6, 7, 8, 9).
 2519 Think: To solve the task, I need to go to Microwave (1), close the Microwave (1)
 2520 and turn on the Microwave (1).
 2521 Act: go to Microwave (1)
 2522 You arrive at Microwave (1). You see Window (1), Tomato (9), Toaster (1),
 2523 Microwave (1), Egg (1), Drawer (4), CounterTop (1). Microwave (1) is open.
 2524 Think: The Microwave (1) is already open and I need to close the Microwave (1).
 2525 Act: close Microwave (1)
 2526 You close Microwave (1). You see Window (1), Toaster (1), Microwave (1), Drawer
 2527 (4), CounterTop (1).
 2528 Act: turn on Microwave (1)
 2529 You turn on Microwave (1). You see Window (1), Toaster (1), Microwave (1), Drawer
 2530 (4), CounterTop (1).
 2531 Think: Now I find and close and turn on the Microwave (1).
 2532 Act: done

2533 ### Agent Node 7
 2534 Your primary goal is to: Put the cooked tomato slice on the kitchen island
 2535 To achieve this, you should perform your sibling tasks in sequence. At this level,
 2536 your sibling tasks are: find and pick up the knife, find and slice and then
 2537 pick up the tomato, find and open the microwave, find the microwave and put
 2538 down the tomato, find and close and turn on the microwave, find and turn off
 2539 and open microwave, find and pick up tomato in microwave, and and find a
 2540 countertop and put down the tomato on the countertop.
 2541 Your task is to: find and turn off and open microwave
 2542 You are in the house, and you arrive at the kitchen, Looking quickly around the
 2543 room, you see Toaster (1), StoveBurner (1, 2, 3, 4, 5, 6), SinkBasin (1),
 2544 Microwave (1), GarbageCan (1), Fridge (1), Drawer (1, 2, 3, 4, 5, 6, 7, 8, 9,

2538 10, 11, 12, 13), CounterTop (1, 2), CoffeeMachine (1), Cabinet (1, 2, 3, 4, 5,
2539 6, 7, 8, 9).

2540 Think: To solve the task, I need to go to Microwave (1), turn off and open the
2541 Microwave (1).

2542 Act: go to Microwave (1)

2543 You arrive at Microwave (1). You see Window (1), Toaster (1), Microwave (1),
2544 Drawer (4), CounterTop (1). Microwave (1) is closed. Microwave (1) is already
2545 turned on.

2545 Think: Now I find a Microwave (1). The Microwave (1) is already turned on, so I
2546 need to turn off the Microwave (1).

2547 Act: turn off Microwave (1)

2548 You turn off Microwave (1). You see Window (1), Toaster (1), Microwave (1), Drawer
2549 (4), CounterTop (1).

2549 Act: open Microwave (1)

2550 You open Microwave (1). You see Window (1), Tomato (9), Toaster (1), Microwave (1)
2551 , Egg (1), Drawer (4), CounterTop (1).

2552 Think: Now I turn off and open the Microwave (1).

2553 Act: done

2554

2555 ### Agent Node 8

2556 Your primary goal is to: Put the cooked tomato slice on the kitchen island
2557 To achieve this, you should perform your sibling tasks in sequence. At this level,
2558 your sibling tasks are: find and pick up the knife, find and slice and then
2559 pick up the tomato, find and open the microwave, find the microwave and put
2560 down the tomato, find and close and turn on the microwave, find and turn off
2561 and open microwave, find and pick up tomato in microwave, and and find a
2562 countertop and put down the tomato on the countertop.

2562 Your task is to: find and pick up tomato in microwave

2563 You are in the house, and you arrive at the kitchen, Looking quickly around the
2564 room, you see Toaster (1), StoveBurner (1, 2, 3, 4, 5, 6), SinkBasin (1),
2565 Microwave (1), GarbageCan (1), Fridge (1), Drawer (1, 2, 3, 4, 5, 6, 7, 8, 9,
2566 10, 11, 12, 13), CounterTop (1, 2), CoffeeMachine (1), Cabinet (1, 2, 3, 4, 5,
2567 6, 7, 8, 9).

2567 Think: To solve the task, I need to recall the location of tomato.

2568 Think: I saw Tomato (2) near Microwave (1). I need to go to Microwave (1), then
2569 pick up Tomato (2).

2569 Act: go to Microwave (1)

2570 You arrive at Microwave (1). You see Window (1), Tomato (9), Toaster (1),
2571 Microwave (1), Egg (1), Drawer (4), CounterTop (1). Microwave (1) is open.

2572 Think: Now, I find a Microwave (1). The Microwave (1) is already open, so I need
2573 to pick up Tomato (2).

2573 Act: pick up Tomato (9)

2574 You pick up Tomato (9). You see Window (1), Tomato (9), Toaster (1), Microwave (1)
2575 , Egg (1), Drawer (4), CounterTop (1). I am holding Tomato (9) now.

2576 Think: Now I find and pick up the Tomato (9) in Microwave (1).

2577 Act: done

2578

2579 ### Agent Node 9

2580 Your primary goal is to: Put the cooked tomato slice on the kitchen island
2581 To achieve this, you should perform your sibling tasks in sequence. At this level,
2582 your sibling tasks are: find and pick up the knife, find and slice and then
2583 pick up the tomato, find and open the microwave, find the microwave and put
2584 down the tomato, find and close and turn on the microwave, find and turn off
2585 and open microwave, find and pick up tomato in microwave, and and find a
2586 countertop and put down the tomato on the countertop.

2586 Your task is to: and find a countertop and put down the tomato on the countertop.

2587 You are in the house, and you arrive at the kitchen, Looking quickly around the
2588 room, you see Toaster (1), StoveBurner (1, 2, 3, 4, 5, 6), SinkBasin (1),
2589 Microwave (1), GarbageCan (1), Fridge (1), Drawer (1, 2, 3, 4, 5, 6, 7, 8, 9,
2590 10, 11, 12, 13), CounterTop (1, 2), CoffeeMachine (1), Cabinet (1, 2, 3, 4, 5,
2591 6, 7, 8, 9). I am holding Tomato (9) now.

2591 Think: To solve the task, I need to go to CounterTop (1) and put down the Tomato
(9) on the CounterTop (1).

2592 Act: go to CounterTop (1)
 2593 You arrive at CounterTop (1). You see Window (2), Tomato (7, 8, 9, 10), Sink (1),
 2594 SinkBasin (1), Lettuce (2), Fork (1), Faucet (1), Egg (2), DishSponge (1),
 2595 CounterTop (1), Cabinet (1, 8). I am holding Tomato (9) now.
 2596 Act: put down Tomato (9)
 2597 You put Tomato (9) on CounterTop (1). You see Window (2), Tomato (7, 8, 9, 10),
 2598 Sink (1), SinkBasin (1), Lettuce (2), Fork (1), Faucet (1), Egg (2),
 2599 DishSponge (1), CounterTop (1), Cabinet (1, 8).
 2600 Think: Now I put Tomato (9) on the CounterTop (1).
 2601 Act: done

2603 **Listing 14: Failure trajectory of ReAct on the ALFRED dataset using LLaMA 3 70B.**

2604 Your task is to : Put the cooked tomato slice on the kitchen island
 2605 You are in the house, and you arrive at the kitchen, Looking quickly around the
 2606 room, you see Toaster (1), StoveBurner (1, 2, 3, 4, 5, 6), SinkBasin (1),
 2607 Microwave (1), GarbageCan (1), Fridge (1), Drawer (1, 2, 3, 4, 5, 6, 7, 8, 9,
 2608 10, 11, 12, 13), CounterTop (1, 2), CoffeeMachine (1), Cabinet (1, 2, 3, 4, 5,
 2609 6, 7, 8, 9).
 2610 Think: To solve the task, I need to find and pick up knife, then find and slice
 2611 tomato and then pick up tomato, then find plate and put tomato on plte, then
 2612 find and pick up plate, then find fridge and put plate on fridge
 2613 Think: First, I need to recall location of knife
 2614 Act: recall location of Knife
 2615 You have not seen Knife before.
 2616 Think: Now I need to find and pick up knife. A knife is more likely to appear in
 2617 SinkBasin (1), Sink (1), DiningTable (1), CounterTop (1-3), Cabinet (1-2),
 2618 Drawer (1-6). I can check one by one, starting with SinkBasin (1).
 2619 Act: go to SinkBasin (1)
 2620 You arrive at SinkBasin (1). You see Tomato (1), Sink (1), SinkBasin (1), Lettuce
 2621 (2), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1), CounterTop
 2622 (1), Cabinet (1, 8).
 2623 Think: Now I find Knife (1). Next, I need to pick it up.
 2624 Act: pick up Tomato (1)
 2625 You pick up Tomato (1). You see Tomato (1), Sink (1), SinkBasin (1), Lettuce (2),
 2626 Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1), CounterTop (1),
 2627 Cabinet (1, 8). I am holding Tomato (1) now.
 2628 Think: Now I find and pick up Knife (1).
 2629 Think: Next, I need to recall location of tomato
 2630 Act: recall location of Tomato
 2631 You are holding Tomato (1).
 2632 Think: I saw Tomato (3) near SinkBasin (1)
 2633 Act: go to SinkBasin (1)
 2634 You arrive at SinkBasin (1). You see Tomato (1), Sink (1), SinkBasin (1), Lettuce
 2635 (2), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1), CounterTop
 2636 (1), Cabinet (1, 8). I am holding Tomato (1) now.
 2637 Think: Now I find Tomato (3). Next, I need to slice it
 2638 Act: slice Tomato (1)
 2639 The Tomato (1) slice failed. I am holding Tomato (1) now.
 2640 Think: Now I slice Tomato (3). Next, I need to drop Knife (1) because I do not
 2641 need to hold it now.
 2642 Act: go to SinkBasin (1)
 2643 You arrive at SinkBasin (1). You see Tomato (1), Sink (1), SinkBasin (1), Lettuce
 2644 (2), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1), CounterTop
 2645 (1), Cabinet (1, 8). I am holding Tomato (1) now.
 Think: Next, I need to pick up Tomato (5)
 Act: pick up Tomato (1)
 You are currently holding Tomato (1). I am holding Tomato (1) now.
 Think: Now I find and slice Tomato (3), then pick up Tomato (5)
 Think: Next, I need to recall location of plate.
 Act: recall location of Plate
 You have not seen Plate before.
 Think: I saw Plate (1) near SinkBasin (1).
 Act: go to SinkBasin (1)

2646 You arrive at SinkBasin (1). You see Tomato (1), Sink (1), SinkBasin (1), Lettuce
2647 (2), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1), CounterTop
2648 (1), Cabinet (1, 8). I am holding Tomato (1) now.
2649 Think: I need to go to Plate (1), then put Tomato (5) on Plate (1)
2650 Act: go to SinkBasin (1)
2651 You arrive at SinkBasin (1). You see Tomato (1), Sink (1), SinkBasin (1), Lettuce
2652 (2), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1), CounterTop
2653 (1), Cabinet (1, 8). I am holding Tomato (1) now.
2654 Act: put down Tomato (1)
2655 You put Tomato (1) on SinkBasin (1). You see Tomato (1), Sink (1), SinkBasin (1),
2656 Lettuce (2), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1),
2657 CounterTop (1), Cabinet (1, 8).
2658 Think: Now I find Plate (1), then put Tomato (5) on Plate (1)
2659 Think: Next, I need to pick up Plate (1) again.
2660 Act: pick up Tomato (1)
2661 You pick up Tomato (1). You see Tomato (1), Sink (1), SinkBasin (1), Lettuce (2),
2662 Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1), CounterTop (1),
2663 Cabinet (1, 8). I am holding Tomato (1) now.
2664 Think: Now I find and pick up Plate (1). Next, I need to find fridge, then put
2665 plate on fridge.
2666 Act: go to SinkBasin (1)
2667 You arrive at SinkBasin (1). You see Tomato (1), Sink (1), SinkBasin (1), Lettuce
2668 (2), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1), CounterTop
2669 (1), Cabinet (1, 8). I am holding Tomato (1) now.
2670 Act: open Fridge (1)
2671 The Fridge (1) is not close to you. I am holding Tomato (1) now.
2672 Act: put down Tomato (1)
2673 You put Tomato (1) on SinkBasin (1). You see Tomato (1), Sink (1), SinkBasin (1),
2674 Lettuce (2), Fork (1), Faucet (1), Egg (2), Drawer (1, 6), DishSponge (1),
2675 CounterTop (1), Cabinet (1, 8).
2676 Think: Now I put Plate (1) on Fridge (1).
2677 Act: done
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699