ACPBench: Reasoning about Action, Change, and Planning

Harsha Kokel, Michael Katz, Kavitha Srinivas, Shirin Sohrabi

IBM Research

Abstract

There is an increasing body of work using Large Language Models (LLMs) as agents for orchestrating workflows and making decisions in domains that require planning and multi-step reasoning. As a result, it is imperative to evaluate LLMs on core skills required for planning. In this work, we present ACPBench, a benchmark for evaluating the reasoning tasks in the field of planning. The benchmark consists of 7 reasoning tasks over 13 planning domains. The collection is constructed from planning domains described in a formal language. This allows us to synthesize problems with provably correct solutions across many tasks and domains. Further, it allows us the luxury of scale without additional human effort, i.e., many additional problems can be created automatically. Our extensive evaluation of 22 LLMs and OpenAI o1 reasoning models highlight the significant gap in the reasoning capability of the LLMs. Our findings with OpenAI o1, a multi-turn reasoning model, reveal significant gains in performance on multiple-choice questions, yet surprisingly, no notable progress is made on boolean questions.

ACPBench collection is available at the following link: https://ibm.github.io/ACPBench

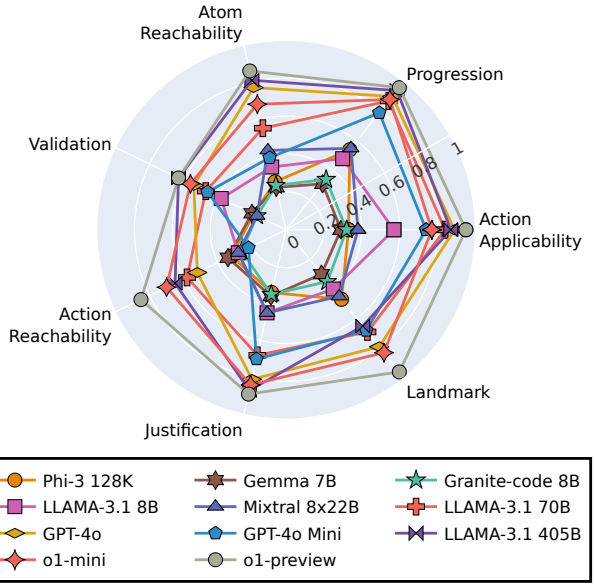


Figure 1: Performance of few state-of-the-art LLMs and OpenAI o1 reasoning models over different tasks in ACPBench. While the largest LLMs achieve more than 80% accuracy on few tasks, the variance in performance across tasks and across LLMs is still big. This signifies the long way to go before they can be reliably used in practical scenarios.

1 Introduction

Recent research has explored the potential of using Large Language Models (LLMs) as reasoners for solving multi-step reasoning problems (Chu et al. 2024). Building on their success in certain reasoning tasks and benchmarks, there is a growing interest in using LLMs as agents for orchestrating workflows and making decisions in domains that require planning (Huang et al. 2024; Wang et al. 2024a). This is a promising area of research, with potential applications in various fields. However, there is a lack of systematic evaluation of LLMs reasoning and planning capabilities.

This work aims at evaluating and improving language models' ability to plan. However, end-to-end evaluation of planning ability is challenging. One, if an agent reaches a goal it does not necessarily mean it can plan. Second, evaluating a plan might be difficult in a domain where there can be multiple plans to achieve the goal. So, instead of focusing on the entire end-to-end planning ability, we distill 7 atomic reasoning tasks that are critical for reliable planning and create datasets of such tasks. These tasks focus on reasoning about Actions, Change (transitions) and Planning; hence, we call our benchmark as ACPBench. The tasks include single step reasoning, like evaluating whether an action can be performed in the described state, as well as multi step reasoning, like whether a sequence of actions is a valid plan for the described state and the described goal.

For each task, ACPBench features both boolean (Bool) and multiple-choice (MCQ) style questions from 13 domains. All the datasets are generated from a formal representation of the domain in Planning Domain Definition Language (PDDL) (McDermott 2000). Twelve of these domains are well-established benchmarks in both planning and reinforcement learning communities, readily available in PDDL for-

mat. Inspired by the shuffle task in BigBenchHard Suite (Suzgun et al. 2023), we have created an additional domain from scratch. The benefit of constructing the dataset from PDDL descriptions is twofold. First, it allows us to use existing planning tools and second, and arguably more important, it allows obtaining *provably correct* information for all the tasks. Natural language templates for these domains were carefully crafted by 5 researchers. These templates and planning tools enable us to generate massive data for each task.

We evaluate performance of OpenAI o1 reasoning model and 22 state-of-the-art language models (including open-sourced `Phi-3 128K` (Abdin et al. 2024), `Mixtral 8x22B` (MistralAI 2024), `LLAMA-3 70B` (Dubey et al. 2024), and a closed source `GPT-4o` (OpenAI 2024a)) on the ACPBench. We found that, with Chain-of-Thought prompting (COT) (Wei et al. 2022) and 2-shot examples, `GPT-4o` was only able to achieve 78.40% accuracy on MCQ questions in the ACPBench; with lowest accuracy of 52.50% for the most difficult (validation) task. Similarly, OpenAI o1 preview achieves accuracy of 87.31% on average for the MCQ questions, with lowest accuracy of 63.08% for the most difficult task. Figure 1 shows the overall performance of few selected models on all 7 tasks of ACPBench. To understand whether the smaller language models can improve their performance on these tasks, we fine-tune a language model on these tasks. The fine-tuning resulted in substantial improvements in performance across tasks and even demonstrated the ability to generalize to previously unseen domains.

In summary, our contributions are as follows:

- We identify a collection of 7 reasoning tasks required for efficient planning and introduce the first of its kind large-scale benchmark—ACPBench.
- We evaluate OpenAI o1 reasoning models and 22 state-of-the-art language models of different sizes on ACPBench.
- We finetune a 8B parameter model and show that the finetuned model performs on par with the large models.
- We conduct following ablations. First to understand effects of in-context example and COT. Second to investigate if tasks in ACPBench capture the plan generation ability.

2 Related Work and Background

Recognizing the importance of evaluating reasoning and planning ability of LLMs, various benchmarks have been proposed (Liu et al. 2023; Ma et al. 2024). Most relevant to our work are the benchmarks that are generated from PDDL tasks. He et al. (2023) proposed a natural language based question answering style dataset to evaluate LLMs on 4 tasks of projection, execution, planning, and goal recognition. PlanBench (Valmeekam et al. 2023b) is a benchmark suite with 8 planning tasks including plan generation, reasoning about plan execution, and plan verification. Both these benchmarks focus on a limited number of planning domains (mainly the BlocksWorld domain), employing a template-based approach to generate natural language text. In contrast, AutoPlanBench (Stein et al. 2024) proposes to leverage LLMs to generate the natural language template. Specifically, they prompt an LLM for natural language template per predicate and per action. By reducing the human effort required for template generation, they were able to scale up the dataset to 12 domains. However, they limit their focus to a single task - plan generation.

```
Context: This is a swap domain where agents are swapping
 items or roles. Each agent is always assigned a single
item/role. The goal is to obtain desired items/roles
assigned. There are 8 agents: carol, michelle, xena, vic
, dave, zoe, heidi, and alice. There are 8 items/roles:
quadcopter, frisbee, necklace, whale, iceskates, guitar,
 zebra, and slinky. Currently, heidi is assigned
necklace, michelle is assigned quadcopter, dave is
assigned iceskates, vic is assigned whale, xena is
assigned slinky, carol is assigned frisbee, alice is
assigned zebra, and zoe is assigned guitar.
Bool: Is the following action applicable in this state:
trade guitar of zoe for iceskates of dave?
MCQ: Which of the following actions will be applicable
in this state?
A. exchange frisbee of carol with zebra of alice.
B. exchange guitar of zoe with necklace of vic.
C. exchange guitar of heidi with zebra of zoe.
D. exchange guitar of vic with zebra of zoe.
```

Figure 2: Example of boolean and multi-choice questions from the Applicablity task in ACPBench. The context contains the domain and the problem description. Query to LLM consists of context and a boolean or multi-choice question.

In parallel, Handa et al. (2024) proposed ActionReasoningBench, featuring six tasks: Fluent Tracking, State Tracking, Action Executability, Effects of Actions, Numerical RAC, and Composite Questions. Although there is some overlap between the tasks in ActionReasoningBench and ACPBench (for example, the Effects of Actions task overlaps with our Progression task), the majority of the tasks we propose are not covered by ActionReasoningBench: Reachability, Action Reachability, Validation, Justification, Landmarks. Similarly, the following ActionReasoningBench tasks are not covered in ACPBench: State Tracking, and Numerical RAC.

We now switch to providing the necessary background. The ACPBench questions collection is generated based on PDDL tasks. A PDDL task is defined over the first-order language; consisting of predicates, variables, and objects. A state s is defined as a conjunction of grounded (by objects) predicates, also called atoms. An action a is defined as a triple $\langle pre(a), add(a), del(a)\rangle$; consisting of preconditions, add effects and delete effects, each being a conjunction of atoms. An action a is applicable in a state s if the state satisfies the preconditions of the action, i.e $pre(a) \subseteq s$. On performing an action a in state s, the world transitions to the next state $t = s[a] = s \setminus del(a) \cup add(a)$. A goal g is also a conjunction of atoms, and a state s is a goal state if $g \subseteq s$. A sequence of actions $\pi_s = a_1 \dots a_n$ is applicable in the state s if the actions are applicable in a sequence to the resulting states. π_s is a plan for the state s if π_s is an applicable sequence of actions that results in a goal state.

3 ACPBench

3.1 Domains

ACPBench collection consists of 11 classical planning domains, Alfworld (Shridhar et al. 2021), and a novel swap domain. The 11 classical planning domain, which were also used by AutoPlanBench (Stein et al. 2024), have public problem instance generators (Seipp, Torralba, and Hoffmann 2022). Alfworld is a text-based reinforcement learning environment where an agent is given house hold tasks like 'put a pan on the table' as a goal. Alfworld uses goals from the Alfred dataset (Shridhar et al. 2020) and encodes the dynamics of the domain as PDDL. This PDDL domain is publicly available[1] and PDDL problem files are obtained from the MINT benchmark (Wang et al. 2024b). For the novel Swap domain, we created the PDDL domain and the problem instance generator. Figure 2 contains an example problem description in this domain. All the domains are summarized in Table 1.

We meticulously curated a set of templates to transform the PDDL task into a natural language description. Following AutoPlanBench, we explored using LLMs to automatically generate the templates, however, we found the templates were not reliable and needed significant modification. So, instead, 5 researchers crafted the translations, carefully selecting and refining the templates to ensure they accurately convey the desired information. Specifically, we have templates for *domain description*, *problem description* and *actions*, from which we can compose (partial) states – current state or a goal. These three templates, together with the PDDL files, are to be provided for every new domain, should we decide to extend the benchmark in the future.

3.2 ACPBench Tasks

We focus on 7 reasoning tasks within the realm of planning. For each task, we provide a description and explain how the data was collected.

1. Applicability (App) The first, basic requirement for efficient planning is to determine the valid, available actions in a given situation. Various existing work have discussed LLMs fall short of this basic ability. When using GPT-4 Turbo for travel planning, Xie et al. (2024) found that more than 30% of the failed plans had *invalid action dead loop*–that is even when the model was informed that the action is invalid, LLMs repeated these actions.

For an action to be valid, its preconditions must hold in the state. Given a state s and the set of actions O, the subset of applicable actions would be $O(s) = \{a \in O \mid pre(a) \subseteq s\}$, easily computable by iterating over the actions. We therefore can create a boolean question with a positive answer by sampling from $O(s)$ and with a negative answer by sampling from $O \setminus O(s)$. A multiple-choice question (MCQ) can be created by sampling the correct answer from $O(s)$ and wrong candidates from $O \setminus O(s)$. Figure 2 shows example of the domain description and problem description used in the context as well as one example each of Bool and MCQ question for applicability task.

[1]https://github.com/alfworld/alfworld/blob/master/alfworld/data/alfred.pddl

Domain	# Pred.	# Actions	Max char.
Blocksworld	5	4	1770
Logistics	9	6	1065
Grippers	4	3	1057
Grid	12	5	1235
Ferry	7	3	2132
FloorTile	10	7	3196
Rovers	25	9	3631
VisitAll	3	1	1347
Depot	6	5	1301
Goldminer	12	7	1140
Satellite	8	5	4302
Swap	1	1	849
Alfworld	34	19	4099

Table 1: Statistics of the 13 domains in ACPBench. The top 8 domains are used for finetuning as well as evaluation. The bottom 5 domains are exclusively used for evaluations. Second column indicates the number of predicates in the PDDL domain, third column presents the number of lifted actions in the domain, and the last column indicates the max character length of the NL problem description in the generated dataset.

2. Progression (Prog) The next task evaluates LLMs ability to understand the outcome of an action or change. This ability is important to track information across transitions. The subpar performance of LLMs on the *Tracking Shuffled Objects* task in the Big Bench Hard dataset suggests a significant limitation in their ability to reason about the consequences of actions or changes (Suzgun et al. 2023). Further, a few papers have proposed to use LLMs to execute a plan. For example, Wang et al. (2023) asks LLM to devise a plan and execute it step-by-step to reach the goal. To faithfully execute a plan, it is important for LLMs to demonstrate understanding of progression; how the world state is changed by the action.

When a valid action is performed, the state changes in the following manner: The delete effects of that action will no longer hold and the add effects will hold. Everything else remains unchanged. Given a state s and an action a, the next state is $t = s \setminus del(a) \cup add(a)$. We can now partition the facts in the problem into four sets: the facts that held before applying the action and still hold ($s \cap t$), the facts that held before but not anymore ($s \setminus t$), those that did not hold but now hold ($t \setminus s$), and those that did not hold before and still don't hold ($F \setminus (s \cup t)$). While the answer of whether the fact is true after applying the action depends only on whether it is in t, the chain of thoughts leading to the answer differs for the aforementioned four cases. We construct a boolean question by sampling from each of the four fact sets (if they are not empty), getting at most two positive and two negative examples per state. A single MCQ is constructed by sampling one possible answer from each of the four fact sets (non-empty ones), according to a uniform procedure described above.

3. Reachability (Reach) The reachability task evaluates if a specific sub-goal can eventually be reached from the given state by taking (possibly multiple) actions. This is a multi-

step reasoning task that can help avoid exploring unfeasible options. To maximize the efficiency of LLMs, it is crucial to detect unreachable (sub)goals early on. This can avoid unnecessary prompting and wasteful exploration, ensuring that the LLMs are utilized effectively, especially when used during search (Yao et al. 2023).

Reachability is PSPACE-hard to answer positively in general (Bylander 1994) for a specific fact, since that would require an evidence - a sequence of actions that achieves a state where the specified facts hold. However, generating positive examples is easy, based on any action sequence, taking the facts out of the end state. For negative examples, we explore multiple cases of unreachable facts and fact pairs. First, existing planning methods (under)approximate the reachability with poly-time computable delete-relaxed reachability (Hoffmann and Nebel 2001). Facts that are not delete-relaxed reachable are therefore guaranteed not to be reachable. Another possible reason for a pair of facts that are individually reachable not to be reachable in the same state is if they are mutually exclusive (Lin 2004; Fišer and Komenda 2018). A simple example of mutually exclusive facts in the ferry domain are *(empty-ferry)* and *(on ?c)*, meaning that the ferry cannot be empty and at the same time a car is on the ferry. Third, static facts that are not true in the initial state will never become true. For instance, c0 can never become a location, so *(location c0)* is unreachable (not captured by the methods in the first case, as they focus solely on non-static predicates). The chain of thoughts for a positive example is based on a sequence of actions that achieve the fact. For the negative examples, the chain of thoughts follows the argument laid out above for each of the cases. As in the previous case, the MCQ is captured by choosing from the lists of positive and negative options.

4. Action Reachability (AReach) In API-driven workflows, the objective is typically presented as an instruction to execute a specific function (Qin et al. 2024). In these scenarios, an LLM must identify the necessary prerequisites for execution and formulate a strategy to meet them. Therefore, it is essential for LLMs to assess whether a given instruction is executable from the provided starting point. We formulate this ability as action reachability task.

The action reachability task is closely related to the atom reachability. If an action model is available, then action reachability is equivalent to the atom reachability over the preconditions of the action. Therefore, this task requires an additional reasoning step about action preconditions. Similarly to the atom reachability task, the positive examples are generated from action rollouts, while the negative examples are generated by collecting actions with preconditions including unreachable atoms according to two of the three cases mentioned above delete-relaxed reachability and mutexes. The third case, unreachable static facts, was not used as often creates non-sensible actions *board car l0 at location c1*. Instead, we added incorrect action templates for each action, like *"board the car c1 at location l0 into the airplane"* or *"drive from location l0 to location l1"*. Here as well, the chain of thoughts are created in a similar manner, and the MCQ is captured based on the positive and negative options lists.

5. Validation (Val) A body of research has advocated the use of LLMs for validation and refinement (Shinn et al. 2023; Gou et al. 2024; Madaan et al. 2023). In line with this research, we propose a Validation task. Here, given an initial state and a goal condition, the objective is to assess whether the specified sequence of actions is valid, applicable, and successfully achieves the intended goal.

There are essentially only four options in this case: (a) the sequence is not valid, (b) the sequence is valid, but not applicable, (c) the sequence is valid, applicable, but does not achieve the goal, and (d) the sequence is a plan. These are the four options used for all MCQ for this task. Since the options do not change, we generate four questions per sample, for each of the options to be a correct answer. In the boolean case, we create six different questions, with positive and negative variants for the three cases of whether the sequence is valid, applicable, and a plan. We generate the data for these questions from plans as follows. For the case (c), starting from a plan, we replace a suffix with a random rollout, ensuring that the goal is not achieved at the end of the rollout, but the sequence remains applicable. For the case (b), we try to replace an action on the sequence with an inapplicable action (one whose precondition does not hold in the state), starting from the end of the sequence. Once successful, we return the sequence ending with the inapplicable action. For the case of (a), we simply randomly choose an action on the sequence to replace its template with an incorrect action template, as in the previous task.

6. Justification (Just) A major criteria for plans to be considered reasonable is whether they include unnecessary actions. In the realm of LLMs and API workflows, it is desirable to avoid calling unnecessary APIs as well as reduce wasteful explorations. Hence, it would be of immense value if LLMs are able to identify whether an action is necessary. This corresponds to the justification task in planning literature.

The justification task reasons whether every action is actually needed on the plan. The problem was studied in the literature (Fink and Yang 1992; Salerno, Fuentetaja, and Seipp 2023) and found to be NP-hard in general. However, optimal plans are known to have all their actions being justified and checking whether a single action or a pair of consequent actions can be removed can be done in polynomial time. We consider the following cases, for either a single action or a pair of consequent actions in a plan: 1) a single action can be removed from the plan and the remaining plan is still a valid plan for the same problem 2) an action cannot be removed from the plan 3) the consequent pairs of actions can be removed from the plan 4) the immediate pairs of action cannot be removed from the plan. Note that we truncate the considered plans and only consider two actions after the goal is reached except if the truncation leads to a non-plan. Given a large set of plans, we consider the above four cases, and generate positive and negative examples for both boolean and multiple choice questions.

7. Landmarks (Land) LLMs have shown to hallucinate or deviate from the task when the trajectory is long (Huang et al. 2024). To alleviate this problem, various work has proposed to use LLMs to decompose the goal into subgoals and achieve

each of these subgoals separately. To do this faithfully, it is crucial for LLMs to be able to identify subgoals that are necessary to achieve the goal. In planning literature such subgoals are often called landmarks (Porteous, Sebastia, and Hoffmann 2001). Landmarks are facts that must become true sometime along every plan. So, the last task in ACPBench evaluates LLMs ability to recognize landmarks.

While checking whether a fact is a landmark is PSPACE-hard (Porteous, Sebastia, and Hoffmann 2001), there are several methods that can find a subset of landmarks (Keyder, Richter, and Helmert 2010; Hoffmann, Porteous, and Sebastia 2004; Richter, Helmert, and Westphal 2008; Zhu and Givan 2003). We use the so-called RHW method (Richter, Helmert, and Westphal 2008). Further, negative evidence can be obtained from a collection of plans - a fact that does not appear on all of these plans is not a landmark. We sample from positive and negative examples obtained that way and construct two boolean questions and one MCQ. Here as well, the chain of thoughts generated capture the described logic.

3.3 Data Generation

We use 25 PDDL problem files of varying sizes per domain. The specific arguments used to generate these problem files can be found in the appendix. These 25 tasks are partitioned into a training and a test set. For each task, we use classical planners to generate a large collection of 1000 plans (Katz and Lee 2023; Katz and Sohrabi 2020). With these plans, we sample the state space as follows. First, given a set of plans, we gather the states along these plans. Then, in order to obtain a diverse sample, we run random rollouts from each of the states found. The number of plans and the sample size are configurable parameters. In the *landmarks* task described above, we also find plans for the sampled states. To do that, we replace the initial state with the sampled state in the planning problem instance and run a top-k planner (Katz and Lee 2023). For finding mutexes, we exploit lifted mutex groups implementations from Fišer (2020). In this manner, we can potentially generate as many examples as we want. But to keep the test set of reasonable size, we generate only 10 examples per domain, per task.[2]

4 Experiments

4.1 Evaluation of pre-trained language models

We first analyse how existing pre-trained language models perform on ACPBench. Table 2 presents the accuracy of all the language models on the 7 ACPBench tasks. These results are mean over 5 runs for all models; except GPT family models and LLAMA-3.1 405B (Dubey et al. 2024), which were run once due to resource constraints. All LLMs were either accessed using API or hosted locally using hugging face transformer library on machines with 2 $A100$ 80GB GPU. Note that accuracy of 50.00 on boolean questions indicates that the performance of the model is as good as a random guess. As all the MCQs in the datasets have 4 options, accuracy less than 25.00 indicates that the performance is worse than random guess. To investigate the out-of-the-box performance, we restrict the evaluation to single turn Chain-of-Thought (COT) prompting with two in-context examples. An example prompt for the Bool applicability question is shown in Figure 3.

```
**Question**: This is a ferry domain, where the task is
 to transport cars from their start to their goal
 locations, using a ferry. Each location is accessible by
  ferry from each other location. The cars can be
 debarked or boarded, and the ferry can carry only one
 car at a time.  There are 2 locations and 2 cars,
 numbered consecutively.  Currently, the ferry is at l0,
 with the car c1 on board. The cars are at locations as
 follows: c0 is at l0.  Is the following action
 applicable in this state:  travel by sea from location
 l1 to location l0?
**Thoughts**: Let's think step by step.
Step 1: In order to apply the action travel by sea from
 location l1 to location l0, the following fact(s) must
 hold in this state: The ferry is at l1 location.
Step 2: These facts do not hold in the mentioned state.
So, the action is not applicable.
**Final Answer**: No.
**Question**: ...
**Thoughts**: ...
**Final Answer**: Yes.
**Question**:  <context> + <question>
**Thoughts**: Let's think step by step.
```

Figure 3: Example of the COT prompt.

Notably, `LLAMA-3.1 405B` and `GPT-4o` consistently outperform other models on these tasks, although they do not **always** achieve the top performance. When it comes to smaller open-sourced models, `Codestral 22B` stands out for its exceptional performance on boolean questions, while `Mixtral 8x7B` excels in handling multi-choice questions. However, both of them lag significantly behind `GPT-4o`, which is the best performer in these tasks. Action Reachability and Validation are the most challenging tasks for LLMs. Surprisingly, the GPT family models are not even among top-3 for the action reachablity task. Across all the tasks, GPT-4o performs best for boolean questions and LLAMA-3.1 405B performs best for multi-choice questions.

Figure 4 displays a domain-wise analysis of the performance of LLMs on multi-choice questions. This analysis showcases the top 8 performing models[3]. The average performance of these top-8 models is shown in Figure 4 as the dotted line in black. This indicates that across models no specific domain seems too easy. However, Rovers, FloorTile, Blocksworld, Alfworld and Satellite domains pose the greatest challenges to LLMs, in that particular order.

4.2 Fine-tuning

Foundational models, and LLMs specifically, have shown to improve performance on specific tasks when they are fine-tuned for those tasks. So, next we investigate if finetuning

[2]This test set will be made publicly available upon acceptance.

[3]All supplementary information on the remaining models and boolean questions are relegated to the Appendix to maintain clarity.

Model	Applicability		Progression		Reachability		Validation		Action Reach.		Justification		Landmark		Mean	
	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ
Phi-3 128K	66.15	33.08	68.46	53.85	52.31	26.15	50.77	19.23	53.33	32.50	49.23	33.85	49.23	46.92	55.53	34.75
Gemma 7B	63.23	28.62	64.92	31.08	53.08	23.08	46.92	20.0	55.67	34.50	50.77	36.46	27.54	30.31	51.80	28.93
Granite 7B	56.92	29.54	55.23	35.38	50.77	34.62	32.15	26.15	48.33	28.33	40.77	25.38	47.69	32.15	48.20	29.67
Mistral 7B	61.54	32.31	73.08	38.46	53.08	28.46	47.85	17.69	*65.00	19.17	48.46	30.00	35.38	33.08	55.00	28.67
Mistral Inst. 7B	63.08	31.54	61.54	46.92	61.54	33.08	52.15	36.15	45.83	34.17	43.08	29.23	57.69	50.77	55.45	37.30
Granite-c 8B	59.23	32.31	70.00	34.31	52.31	24.31	44.15	17.08	57.50	25.83	46.92	34.62	37.23	35.38	53.09	29.21
Granite-c Inst. 8B	55.38	32.31	69.23	34.46	50.77	29.23	45.85	22.31	42.50	39.33	46.15	32.31	43.85	38.46	50.53	32.63
LLAMA-3 8B	72.92	49.23	73.08	56.00	55.23	41.08	51.54	*49.23	63.50	36.67	57.54	32.31	56.92	43.85	61.53	44.05
LLAMA-3.1 8B	65.38	56.92	63.85	47.69	53.08	33.85	60.00	37.69	42.50	28.33	46.92	45.38	33.85	40.00	51.46	41.52
Mixtral 8x7B	75.85	*57.69	74.00	*61.38	*76.00	40.00	65.69	34.77	52.83	*55.00	55.38	51.38	59.54	*60.00	65.53	*51.44
Granite 13B	42.00	29.23	52.46	20.77	47.69	28.46	51.54	34.62	45.17	26.33	45.38	27.69	50.31	19.23	47.79	26.66
Codestral 22B	*84.62	39.23	*83.85	51.54	54.62	28.46	*66.15	24.62	53.33	38.33	*67.69	*62.31	59.23	42.31	*67.4	40.97
Mixtral 8x22B	80.77	37.69	72.31	54.62	50.00	*42.62	37.69	16.92	58.50	27.83	43.08	44.62	44.77	45.23	55.63	39.25
Deepseek Inst. 33B	70.77	37.23	68.46	46.31	53.08	31.69	51.54	37.69	50.00	27.50	46.92	26.15	*62.31	39.23	57.58	35.11
LLAMA-c 34B	80.77	42.31	73.08	43.85	53.08	25.69	50.15	28.46	53.17	33.33	55.38	35.38	46.92	40.62	59.02	35.71
LLAMA-2 70B	78.46	24.62	71.54	36.77	53.08	26.92	51.38	16.15	60.83	22.00	49.23	55.54	24.46	26.00	55.72	29.71
LLAMA-c 70B	74.77	36.15	54.77	52.92	48.62	23.69	40.0	17.69	49.67	28.83	46.92	31.54	37.08	42.31	50.9	32.87
LLAMA-3 70B	90.77	82.31	93.08	86.15	**87.69**	**82.31**	**78.62**	56.62	60.50	63.00	62.31	85.38	78.15	64.77	78.71	74.30
LLAMA-3.1 70B	93.08	84.31	89.85	86.77	61.38	54.92	66.15	46.62	63.00	58.00	56.92	68.46	34.62	69.23	66.67	66.94
LLAMA-3.1 405B	95.38	86.92	93.08	**93.85**	59.23	80.77	77.23	**62.92**	**65.00**	**65.00**	**90.00**	**86.92**	83.08	65.38	80.49	**77.42**
GPT-4o Mini	90.77	73.85	**95.38**	79.23	80.77	39.23	67.69	46.15	54.17	21.67	77.69	70.00	76.92	67.69	77.74	56.50
GPT-4o	**96.92**	**89.23**	94.62	90.00	79.23	76.92	61.54	53.85	57.50	52.50	88.46	80.77	**95.38**	**79.23**	**81.84**	74.97

Table 2: Accuracy of 22 LLMs on 7 ACPBench tasks (boolean as well as multi-choice questions). The best results are **boldfaced**, second best are *underlined*, and the best among the small, open-sourced models are highlighted with $*$. All models were evaluated with two in-context examples and Chain-of-Thought prompt. The right-most column is mean across tasks.

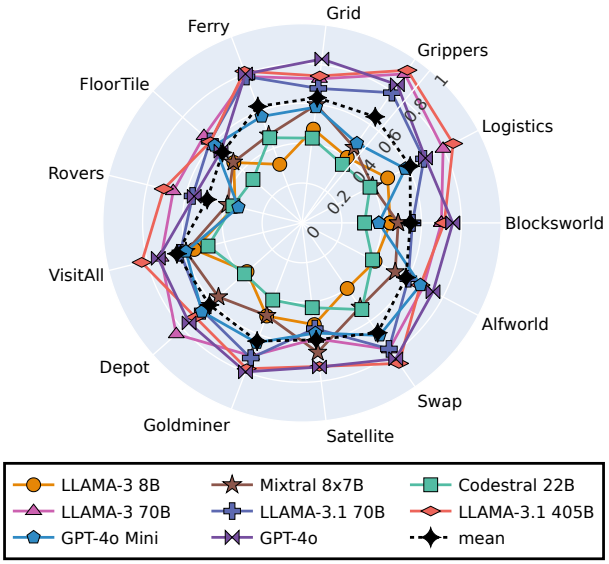


Figure 4: Comparison of 8 top performing LLMs on multi-choice questions in 13 domains of ACPBench. The mean of performance across the top-8 models is presented with dotted line in Black. The mean line indicates that none of the domains are exceptionally easy.

a language model provides any improvement. For this investigation, we keep aside the following 5 domains, Depot, Goldminer, Satellite, Swap, and Alfworld, and generate a training set for the remaining 8 domains. Then we pick one of the small models, `Granite-code 8B` (Mishra et al. 2024), and finetune it with QLoRA. The resulting performance improvement is shown in Table 3. As finetuned models have already seen examples during training, we use only IO prompts with the finetuned model. We finetuned Granite 8B available on HuggingFace on a machine with two A100 80GB GPUs.

Upon finetuning, the average accuracy of the model improves from 51.43% to 95.71% on boolean questions and from 19.18% to 94.29% on multi-choice questions. Further, Table 4 presents the performance on the remaining 5 unseen domains. It is remarkable to observe such a significant improvement even on unseen domains; sometimes surpassing the GPT-4o performance. This indicates that finetuning a model, even on a separate domain, improves performance on these tasks. The right-most column in Tables 3 and 4 presents the performance of the best on that task LLM with COT 2-shots prompting. As can be seen; `Granite Finetuned` model outperforms the best of all models for most of the tasks in the training domains. Even in testing domains, the accuracy difference is significantly reduced upon finetuning.

4.3 Ablations

Prompt Style From previous section, it is clear that COT 2-shot yields better results than IO prompts for ACPBench tasks. However, it is not clear whether COT or 2-shot examples provide the performance gain. To investigate this, we perform the following ablation study. We compare four prompt styles: (1) IO prompt, (2) Chain-of-Thought prompt without in-context examples (COT), (3) IO prompt with two in-context examples (IO 2-shots), and (4) Chain-of-Thought with two in-context examples (COT 2-shots).[4]

We include `Granite-code 8B base` model, `LLAMA-3 70B` (one of the top-performing open source model), and the `Granite-code 8B finetuned FT` model. To have a fair comparison, we use 2-shot examples

[4]Examples of prompts are included in the Appendix.

Task		Base IO	Base COT 2-shot	Finetuned IO	Best
App	B	53.75	62.5 (+8.75)	**98.75** (+45.0)	97.50
	MC	15.0	36.75 (+21.75)	**92.5** (+77.5)	90.00
Prog	B	52.5	76.25 (+23.75)	**97.5** (+45.0)	96.25
	MC	22.5	33.25 (+10.75)	**93.75**(+71.25)	**93.75**
Reach	B	47.5	52.5 (+5.0)	**97.5** (+50.0)	87.50
	MC	15.0	20.75 (+5.75)	**98.75**(+83.75)	82.5
Val	B	45.0	40.5 (−4.5)	100.0 (+55.0)	78.75
	MC	38.5	20.0 (−18.5)	**87.5** (+49.0)	57.75
AReach	B	45.0	56.25 (+11.25)	**97.5** (+52.5)	65.75
	MC	14.25	28.75 (+14.5)	**95.0** (+80.75)	78.75
Just	B	56.25	50.0 (−6.25)	**97.5** (+41.25)	90.0
	MC	16.25	35.0 (+18.75)	**96.25** (+80.0)	82.5
Land	B	60.0	41.25 (−18.75)	81.25(+21.25)	**97.50**
	MC	20.0	18.5 (−1.5)	**90.0** (+70.0)	71.25
Mean	B	51.43	54.18 (+2.75)	**95.71**(+44.28)	81.07
	MC	20.21	27.57 (+7.36)	**93.39**(+73.18)	77.68

Table 3: Comparison of the `Granite-code Base 8B` model and the finetuned model on 8 training domains of ACPBench. We present accuracy values for the Base model with Input-Output prompts (IO) as well as with Chain-of-Thought prompt with two in-context examples (COT 2-shot). The values enclosed in parentheses represent the improvement over the base model w/ IO prompts. The right-most column presents the performance of the best LLM with COT 2-shot on training domain. Best results are in bold.

Task		Base IO	Base COT 2-shot	Finetuned IO	Best
App	B	50.0	54.0 (+4.0)	74.0 (+24.0)	**96.00**
	MC	14.0	25.2 (+11.2)	62.0 (+48.0)	**88.00**
Prog	B	50.0	60.0 (+10.0)	80.0 (+30.0)	**94.00**
	MC	28.0	36.0 (+8.0)	82.0 (+54.0)	**96.0**
Reach	B	46.0	52.0 (+6.0)	82.0 (+36.0)	**88.00**
	MC	10.0	30.0 (+20.0)	56.0 (+46.0)	**82.00**
Val	B	46.0	50.0 (+4.0)	80.0 (+34.0)	**84.0**
	MC	26.0	12.4 (−13.6)	54.0 (+28.0)	**71.2**
AReach	B	35.0	60.0 (+25.0)	**82.5** (+47.5)	77.5
	MC	5.0	20.0 (+15.0)	**70.0** (+65.0)	57.50
Just	B	42.0	42.0 (+0.0)	**98.0** (+56.0)	96.0
	MC	16.0	34.0 (+18.0)	80.0 (+64.0)	**94.0**
Land	B	44.0	30.8 (−13.2)	72.0 (+28.0)	**92.0**
	MC	20.0	62.4 (+42.4)	92.0 (+72.0)	**94.0**
Mean	B	44.71	49.83 (+5.21)	81.21 (+36.5)	**82.79**
	MC	17.00	31.43 (+14.43)	70.86(+53.86)	**78.07**

Table 4: Comparison of `Granite-code Base 8B` and finetuned model on 5 ACPBench domains that are unseen during training. The values enclosed in parentheses represent the improvement over the base model w/ IO prompts. The right-most column presents performance of the best LLM with COT 2-shot on testing domain. Best results are in bold.

from the training domains and only compare performance on the testing domains for MCQ tasks. Figure 5 presents the results. For the two pretrained models, we see that while COT 2-shots prompting yields better result than IO, IO 2-shots prompting had the best performance. For finetuned model, we see that neither COT nor 2-shots provide any advantage; rather IO prompts yield the best results.

Generalization ACPBench consists of tasks that are crucial for effective, robust and reliable planning. Improving performance on ACPBench should improve LLM's ability to reason about these tasks, and hence should improve LLM's ability to generate plans. To verify this hypothesis, we compare the `Granite-code Base 8B` model and `Granite-code finetuned 8B` model on plan generation task (t1) in PlanBench (Valmeekam et al. 2023a). Table 5 presents the results. Granite finetuned model, which was QLoRA (Dettmers et al. 2023) trained on ACPBench tasks for 8 training domains, shows improvement on plan generation ability.

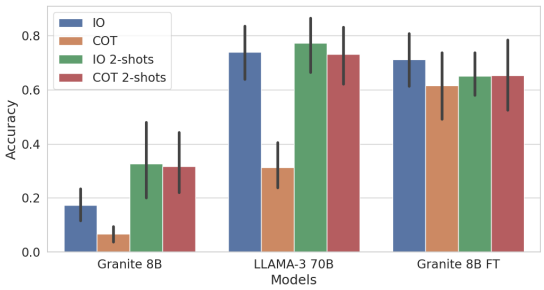


Figure 5: Comparison of different prompt styles on two pretrained models: Granite 8B and LLAMA-3 70B, and finetuned Granite 8B model for MCQ tasks in 5 testing domains.

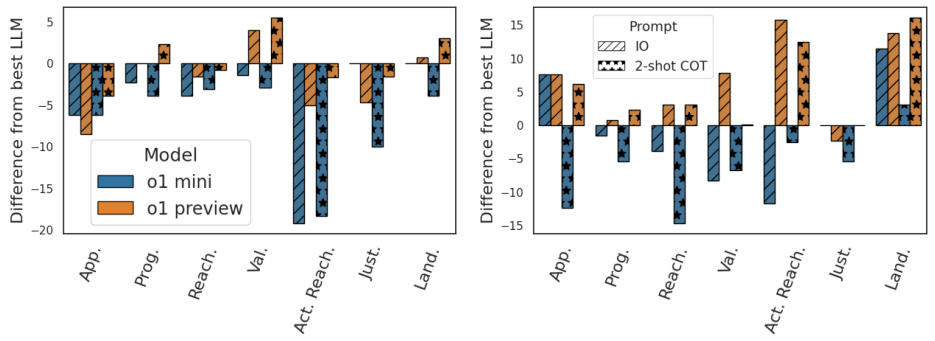
Domain	Base	Finetuned	LLAMA-3 70B
Blocksworld (600)	24	44	57
Logistics (285)	14	15	14

Table 5: Comparison of `Granite-code Base`, `finetuned`, and `LLAMA-3 70B` model on PlanBench Dataset.

4.4 Reasoning Model: OpenAI o1

Recently, OpenAI released a series of LLM-based reasoning models called OpenAI o1 (OpenAI 2024b), that show significant improvement over GPT-4o on benchmarks that require reasoning. Although OpenAI o1 preview and mini are made available via similar APIs as previous LLMs, they do not truly fit the LLM category; rather, they are a system (or an agent) that makes multiple calls to LLMs before providing an answer. Note that OpenAI o1were also referred to as Large Reasoning Models (Valmeekam, Stechly, and Kambhampati 2024). While we acknowledge the difference, it is interesting to compare the best performing LLMs to the OpenAI o1 models. The comparison is depicted in Table 6. Further, Figure 6 shows the performance difference of OpenAI o1 models (with zeroshot IO and 2-shot COT prompts) from the best performing LLMs. Our results indicate that OpenAI o1 models fail to yield performance gains for boolean questions, but demonstrate notable improvements on MCQs. Specifically, OpenAI o1 preview consistently performs better or equal to the best performing model for MCQs. The responses for MCQ tasks suggests that OpenAI o1 models consider each option individually, perform a case-by-case analysis, and only then select an option.

We would like to reiterate that while we present results of



(a) Bool (b) MCQ

Figure 6: Comparing OpenAI o1 models with the best LLM. Positive difference shows OpenAI o1 model performing better than the best of the LLMs. Negative difference is when OpenAI o1 model lags behind the best LLM.

Model	Applicability		Progression		Reachability		Validation		Action Reach.		Justification		Landmark		Mean	
	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ
2-shot Chain-of-Thought prompt																
LLAMA 405B	95.38	86.92	93.08	93.85	59.23	80.77	77.23	62.92	65.00	65.00	90.00	86.92	83.08	65.38	80.43	77.39
GPT-4o Mini	90.77	73.85	95.38	79.23	80.77	39.23	67.69	46.15	54.17	21.67	77.69	70.00	76.92	67.69	77.63	56.83
GPT-4o	96.92	89.23	94.62	90.00	79.23	76.92	61.54	53.85	57.50	52.50	88.46	80.77	95.38	79.23	81.95	74.64
o1-preview	93.08	95.38	97.69	96.15	86.92	86.15	90.00	63.08	72.50	85.00	88.46	89.23	98.46	96.15	89.59	87.31
o1-mini	90.77	76.92	91.54	88.46	84.62	68.46	81.54	56.15	55.83	70.00	80.00	83.85	91.54	83.08	82.26	75.27
zeroshot Input-Output prompt																
LLAMA 405B	88.46	83.08	90.77	90.77	85.38	83.08	84.46	50.00	74.17	72.50	77.69	89.23	83.08	69.23	83.43	76.84
GPT-4o Mini	70.77	66.92	68.46	80.77	80.00	58.46	54.62	21.54	57.50	55.83	56.92	44.62	64.62	66.15	64.7	56.33
GPT-4o	68.46	83.08	71.54	84.62	74.62	77.69	56.15	37.69	60.00	69.17	59.23	86.92	76.92	80.00	66.7	74.17
o1-preview	88.46	96.92	95.38	94.62	86.15	86.15	88.46	70.77	69.17	88.33	85.38	86.92	96.15	93.85	87.02	88.22
o1-mini	90.77	96.92	93.08	92.31	83.85	79.23	83.08	54.62	55.00	60.83	90.00	89.23	95.38	91.54	84.45	80.67

Table 6: Comparison of o1 Reasoning Model with the best performing LLMs on 7 ACPBench tasks (Bool as well as MCQ). The right-most column is mean across tasks.

OpenAI o1 side by side with LLAMA-3.1 405B and GPT-4o LLMs, the comparison is not even-handed due to below mentioned reasons:

- All our LLM experiments had a generated token limit of 1024; OpenAI o1 models did not have that limit. On average the number of tokens generated by OpenAI o1 preview for MCQ tasks, where we see the maximum improvement, was 5705 (this includes the completion token (3164) and the reasoning token (2542)).
- LLM evaluations are based on a single generation. We did not evaluate multi-turn prompts (such as self-consistency or self-reflection). OpenAI o1 models seem to internally make multiple calls to an LLM.

In terms of pure monetary cost, the OpenAI o1 evaluation is approximately 20 times more expensive than of GPT-4o. It remains to be seen if a multi-turn prompting of an open-sourced LLM like LLAMA-3.1 can achieve similar improvement with lower cost.

5 Discussion and Future Work

In this work, we introduce ACPBench—a collection of datasets to evaluate the ability of LLMs to reason about action, change and planning. By evaluating 22 state-of-the-art LLMs of varying size, we find these models underperform, even the largest ones, especially on tasks such as plan validation and action reachability. On the other hand, we show that finetuning a small language model, Granite 8B, can improve its reasoning ability to bring it on par with the best performing models. Further, we observe that the fine-tuned model exhibits remarkable generalization ability to unseen domains in ACPBench as well as to a different task in PlanBench. Further, our investigation with OpenAI o1 reasoning model indicates that OpenAI's multi-turn approach yields improvements for multi-choice questions but fails to make an impact on boolean questions in ACPBench.

Performance of LLMs is known to be sensitive to prompt text as well as prompt style. Hence, it is possible to elicit better performance from each of these models with prompt engineering. In our work we do not modify prompts across models – our objective in the evaluation is to set a baseline. We hope our benchmark serves as a useful resource for improving LLM abilities. We encourage creative solutions (not limited to prompt engineering) to improve LLM performance across various tasks of ACPBench.

References

Abdin, M. I.; Jacobs, S. A.; Awan, A. A.; et al. 2024. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. *CoRR*, abs/2404.14219.

Bylander, T. 1994. The Computational Complexity of Propositional STRIPS Planning. *AIJ*, 69(1–2): 165–204.

Chu, Z.; Chen, J.; Chen, Q.; Yu, W.; He, T.; Wang, H.; Peng, W.; Liu, M.; Qin, B.; and Liu, T. 2024. Navigate through Enigmatic Labyrinth A Survey of Chain of Thought Reasoning: Advances, Frontiers and Future. In *ACL*. Association for Computational Linguistics.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. In *NeurIPS*.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; et al. 2024. The Llama 3 Herd of Models. arXiv:2407.21783.

Fink, E.; and Yang, Q. 1992. Formalizing Plan Justifications. In *Proc. CSCSI 1992*.

Fišer, D. 2020. Lifted Fact-Alternating Mutex Groups and Pruned Grounding of Classical Planning Problems. In *Proc. AAAI 2020*, 9835–9842.

Fišer, D.; and Komenda, A. 2018. Fact-Alternating Mutex Groups for Classical Planning. *JAIR*, 61: 475–521.

Gou, Z.; Shao, Z.; Gong, Y.; Shen, Y.; Yang, Y.; Duan, N.; and Chen, W. 2024. CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing. In *ICLR*. OpenReview.net.

Handa, D.; Dolin, P.; Kumbhar, S.; Baral, C.; and Son, T. C. 2024. ActionReasoningBench: Reasoning about Actions with and without Ramification Constraints. *CoRR*, abs/2406.04046.

He, W.; Huang, C.; Xiao, Z.; and Liu, Y. 2023. Exploring the Capacity of Pretrained Language Models for Reasoning about Actions and Change. In *ACL*. Association for Computational Linguistics.

Hoffmann, J.; and Nebel, B. 2001. RIFO Revisited: Detecting Relaxed Irrelevance. In *Proc. ECP 2001*, 127–135.

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered Landmarks in Planning. *JAIR*, 22: 215–278.

Huang, X.; Liu, W.; Chen, X.; Wang, X.; Wang, H.; Lian, D.; Wang, Y.; Tang, R.; and Chen, E. 2024. Understanding the planning of LLM agents: A survey. *CoRR*, abs/2402.02716.

Katz, M.; and Lee, J. 2023. K* Search Over Orbit Space for Top-k Planning. In *Proc. IJCAI 2023*.

Katz, M.; and Sohrabi, S. 2020. Reshaping Diverse Planning. In *Proc. AAAI 2020*, 9892–9899.

Keyder, E.; Richter, S.; and Helmert, M. 2010. Sound and Complete Landmarks for And/Or Graphs. In *Proc. ECAI 2010*, 335–340.

Lin, F. 2004. Discovering State Invariants. In *Proc. KR 2004*, 536–544.

Liu, X.; Yu, H.; Zhang, H.; Xu, Y.; Lei, X.; Lai, H.; Gu, Y.; Ding, H.; Men, K.; Yang, K.; Zhang, S.; Deng, X.; Zeng, A.; Du, Z.; Zhang, C.; Shen, S.; Zhang, T.; Su, Y.; Sun, H.; Huang, M.; Dong, Y.; and Tang, J. 2023. AgentBench: Evaluating LLMs as Agents. *CoRR*, abs/2308.03688.

Ma, C.; Zhang, J.; Zhu, Z.; Yang, C.; Yang, Y.; Jin, Y.; Lan, Z.; Kong, L.; and He, J. 2024. AgentBoard: An Analytical Evaluation Board of Multi-turn LLM Agents. *CoRR*, abs/2401.13178.

Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; Alon, U.; Dziri, N.; Prabhumoye, S.; Yang, Y.; Gupta, S.; Majumder, B. P.; Hermann, K.; Welleck, S.; Yazdanbakhsh, A.; and Clark, P. 2023. Self-Refine: Iterative Refinement with Self-Feedback. In *NeurIPS*.

McDermott, D. 2000. The 1998 AI Planning Systems Competition. *AI Magazine*, 21(2): 35–55.

Mishra, M.; Stallone, M.; Zhang, G.; Shen, Y.; Prasad, A.; et al. 2024. Granite Code Models: A Family of Open Foundation Models for Code Intelligence. *CoRR*, abs/2405.04324.

MistralAI. 2024. Mixtral 8x22B. https://mistral.ai/news/mixtral-8x22b/.

OpenAI. 2024a. GPT 4o. https://openai.com/index/hello-gpt-4o/.

OpenAI. 2024b. Learning to Reason with LLMs. https://openai.com/index/learning-to-reason-with-llms/.

Porteous, J.; Sebastia, L.; and Hoffmann, J. 2001. On the Extraction, Ordering, and Usage of Landmarks in Planning. In *Proc. ECP 2001*, 174–182.

Qin, Y.; Liang, S.; Ye, Y.; Zhu, K.; Yan, L.; Lu, Y.; Lin, Y.; Cong, X.; Tang, X.; Qian, B.; Zhao, S.; Hong, L.; Tian, R.; Xie, R.; Zhou, J.; Gerstein, M.; Li, D.; Liu, Z.; and Sun, M. 2024. ToolLLM: Facilitating Large Language Models to Master 16000+ Real-world APIs. In *ICLR*. OpenReview.net.

Richter, S.; Helmert, M.; and Westphal, M. 2008. Landmarks Revisited. In *Proc. AAAI 2008*, 975–982.

Salerno, M.; Fuentetaja, R.; and Seipp, J. 2023. Eliminating Redundant Actions from Plans using Classical Planning. In *Proc. KR 2023*, 774–778.

Seipp, J.; Torralba, Á.; and Hoffmann, J. 2022. PDDL Generators. https://doi.org/10.5281/zenodo.6382173.

Shinn, N.; Cassano, F.; Gopinath, A.; Narasimhan, K.; and Yao, S. 2023. Reflexion: language agents with verbal reinforcement learning. In *Proc. NeurIPS 2023*.

Shridhar, M.; Thomason, J.; Gordon, D.; Bisk, Y.; Han, W.; Mottaghi, R.; Zettlemoyer, L.; and Fox, D. 2020. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *CVPR*, 10737–10746. Computer Vision Foundation / IEEE.

Shridhar, M.; Yuan, X.; Côté, M.-A.; Bisk, Y.; Trischler, A.; and Hausknecht, M. 2021. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Stein, K.; Fišer, D.; Hoffmann, J.; and Koller, A. 2024. AutoPlanBench: Automatically generating benchmarks for LLM planners from PDDL. arXiv:2311.09830 [cs.AI].

Suzgun, M.; Scales, N.; Schärli, N.; et al. 2023. Challenging BIG-Bench Tasks and Whether Chain-of-Thought Can Solve Them. In *ACL (Findings)*, 13003–13051. Association for Computational Linguistics.

Valmeekam, K.; Marquez, M.; Olmo, A.; Sreedharan, S.; and Kambhampati, S. 2023a. PlanBench: An Extensible Benchmark for Evaluating Large Language Models on Planning and Reasoning about Change. In *Proc. NeurIPS 2023*, 38975–38987.

Valmeekam, K.; Marquez, M.; Sreedharan, S.; and Kambhampati, S. 2023b. On the Planning Abilities of Large Language Models - A Critical Investigation. In *Proc. NeurIPS 2023*.

Valmeekam, K.; Stechly, K.; and Kambhampati, S. 2024. LLMs Still Can't Plan; Can LRMs? A Preliminary Evaluation of OpenAI's o1 on PlanBench. arXiv:2409.13373.

Wang, L.; Ma, C.; Feng, X.; Zhang, Z.; Yang, H.; Zhang, J.; Chen, Z.; Tang, J.; Chen, X.; Lin, Y.; Zhao, W. X.; Wei, Z.; and Wen, J. 2024a. A survey on large language model based autonomous agents. *Frontiers Comput. Sci.*, 18(6): 186345.

Wang, L.; Xu, W.; Lan, Y.; Hu, Z.; Lan, Y.; Lee, R. K.; and Lim, E. 2023. Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models. In *ACL*. Association for Computational Linguistics.

Wang, X.; Wang, Z.; Liu, J.; Chen, Y.; Yuan, L.; Peng, H.; and Ji, H. 2024b. MINT: Evaluating LLMs in Multi-turn Interaction with Tools and Language Feedback. In *ICLR*. OpenReview.net.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E. H.; Le, Q. V.; and Zhou, D. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proc. NeurIPS 2022*, 24824–24837.

Xie, J.; Zhang, K.; Chen, J.; Zhu, T.; Lou, R.; Tian, Y.; Xiao, Y.; and Su, Y. 2024. TravelPlanner: A Benchmark for Real-World Planning with Language Agents. In *ICML*.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T.; Cao, Y.; and Narasimhan, K. 2023. Tree of thoughts: Deliberate problem solving with large language models. In *Proc. NeurIPS 2023*.

Zhu, L.; and Givan, R. 2003. Landmark Extraction via Planning Graph Propagation. In *ICAPS 2003 Doctoral Consortium*, 156–160.

6 Reproducibility Checklist

This paper:

- Includes a conceptual outline and/or pseudocode description of AI methods introduced: **NA**
- Clearly delineates statements that are opinions, hypothesis, and speculation from objective facts and results: **Yes**
- Provides well marked pedagogical references for less-familiare readers to gain background necessary to replicate the paper **Yes**

Does this paper make theoretical contributions? **No**
Does this paper rely on one or more datasets? **yes**
If yes, please complete the list below.

- A motivation is given for why the experiments are conducted on the selected datasets **yes**
- All novel datasets introduced in this paper are included in a data appendix. **no**
- All novel datasets introduced in this paper will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. **yes**
- All datasets drawn from the existing literature (potentially including authors' own previously published work) are accompanied by appropriate citations. **yes**
- All datasets drawn from the existing literature (potentially including authors' own previously published work) are publicly available. **yes**
- All datasets that are not publicly available are described in detail, with explanation why publicly available alternatives are not scientifically satisficing. **yes**

Does this paper include computational experiments? **yes**
If yes, please complete the list below.

- Any code required for pre-processing data is included in the appendix. **no**
- All source code required for conducting and analyzing the experiments is included in a code appendix. **no**
- All source code required for conducting and analyzing the experiments will be made publicly available upon publication of the paper with a license that allows free usage for research purposes. **yes**
- All source code implementing new methods have comments detailing the implementation, with references to the paper where each step comes from **yes**
- If an algorithm depends on randomness, then the method used for setting seeds is described in a way sufficient to allow replication of results. **yes**
- This paper specifies the computing infrastructure used for running experiments (hardware and software), including GPU/CPU models; amount of memory; operating system; names and versions of relevant software libraries and frameworks. **yes**
- This paper formally describes evaluation metrics used and explains the motivation for choosing these metrics. **yes**
- This paper states the number of algorithm runs used to compute each reported result. **yes**
- Analysis of experiments goes beyond single-dimensional summaries of performance (e.g., average; median) to include measures of variation, confidence, or other distributional information. **yes**
- The significance of any improvement or decrease in performance is judged using appropriate statistical tests (e.g., Wilcoxon signed-rank). **NA**
- This paper lists all final (hyper-)parameters used for each model/algorithm in the paper's experiments. **NA**
- This paper states the number and range of values tried per (hyper-) parameter during development of the paper, along with the criterion used for selecting the final parameter setting. **NA**

A Appendix

A.1 ACPBench Task Examples

Table 7 exemplified the domain description, the problem description and the goal description for each domain. For each task we use all or subset of these descriptions as context. Table 8 indicates what is included as part of the context for each of the tasks and also provide one example of boolean and multi-choice questions each.

A.2 Pretrained LLMs

Our paper presents domain-wise performance of few selected models in Figure 4, where we only presented results for the MCQ due to space constraints. In this section, we present the domain-wise analysis for all the 22 pretrained LLMs for both boolean and multi-choice questions in Table 9.

In the main paper, Table 2 presents accuracy values with 2-shot COT prompting. We also attempted zeroshot Input Output (IO) prompt, the trends were similar. These zeroshot IO results aggregated over 5 runs are presented in Table 10.

A.3 Finetuned LLM

Tables 11 and Table 12 show per-domain comparision of the 7 tasks between the Granite (code 8B) Base model and the finetuned model, on multiple choice questions and boolean questions respectively. The "Diff" column shows the average gain in performance. Generally we may see a greater performance gain in the seen domains as they have been included in the training set as oppose to the unseen domains. Note, due to memory limitations we were not able to test the Alfworld domain on action reachability.

In Tables 3 and 4, we compare te finetuned model against the best performing model (last column). These values are obtained by looking at the aggregated performance on train and test domains respectively. We present performance of all the models on train and test domains in Tables 13 and 14.

A.4 Ablation: Prompt Style

In our paper, we present an ablation to compare prompting styles. For that analysis; we presented results on test domains for multi-choice questions in Figure 5. Here, in Figure 7, we present performance on boolean questions.

We also compared the prompt-style on LLAMA 3.1 405B model. Figure 8 presents aggregated results on 7 ACPBench tasks for all the domains. Although we were only run this experiment once due to resource constraint, we see that COT and IO 2-shot has significant different in performance. The difference is IO 2-shot vs COT 2-shot is significant.

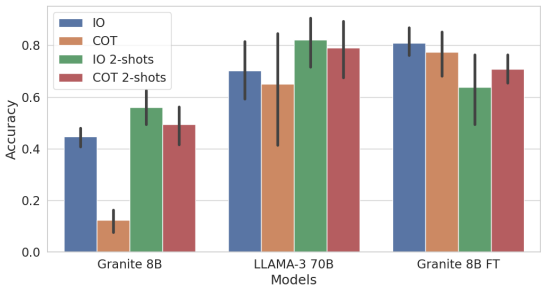


Figure 7: Comparison of different prompt styles on two pretrained models: Granite 8B and LLAMA-3 70B, and finetuned Granite 8B model for boolean tasks in 5 testing domains.

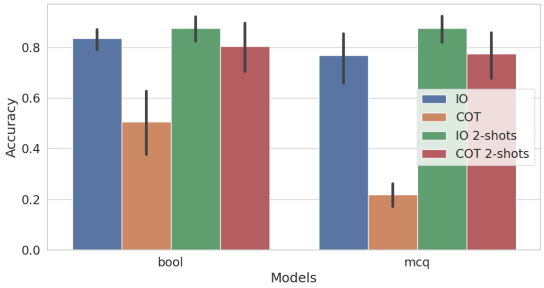


Figure 8: Comparison of different prompt styles on LLAMA 3.1 405B for Bool & MCQ tasks in all ACPBench domains.

Table 7: Example of domain description, problem description and goal description of 13 domains in ACPBench.

	Domain Desc.	Problem Desc.	Goal Desc.
Blocksworld	This is a blocksworld domain where blocks can be placed on top of each other or on the table. There is one robotic arm that can move the block.	There are 3 blocks. Currently, the robotic arm is empty. The following block(s) is on the table: block_1. The following block(s) are stacked on top of another block: block_3 is on block_2 and block_2 is on block_1.	The goal is to reach a state where the following facts hold: The block block_3 is currently situated above the block block_1.
Logistics	There are several cities, each containing several locations, some of which are airports. There are also trucks, which can drive within a single city, and airplanes, which can fly between airports. The goal is to get some packages from various locations to various new locations. There are 2 trucks and 1 airplane, as well as 4 packages. There are 6 locations across 2 cities.	The locations are in cities as follows: l0-1, l0-2, and l0-0 are in c0; l1-2, l1-1, and l1-0 are in c1. Currently, t0 and p0 are at l0-0, t1 is at l1-1, p3 and p1 are at l0-2, p2 and a0 are at l1-0.	The goal is to reach a state where the following facts hold: p2 is at l0-1, p0 is at l0-1, p1 is at l0-1, and p3 is at l0-1.
Grippers	This is a grippers domain, where there is a robot with two grippers. The robot can carry a ball in each. The goal is to take the balls from one room to another. There are 1 robot, 5 rooms, and 4 balls, numbered consecutively.	Currently, the robot robot1 is at room4 and both grippers are free. Additionally, ball3 is at room5, ball4 and ball2 are at room1, ball1 is at room2.	The goal is to reach a state where the following facts hold: Ball ball3 is at room4 location.
Grid	A robot is in a grid and can only move to places that are connected to its current position. The grid size is 5x5, and the locations are of the form fi-jf (e.g., f3-2f or f0-1f). The grid cells are connected to their neighbors (e.g., f1-2f is connected to the four neighbors f0-2f, f2-2f, f1-1f, and f1-3f). Some positions on the grid are locked and can be opened with a key of a matching shape. The robot has an arm that can pick up a key when the key is in same location as the robot and the arm is empty. There are 2 keys in 1 different shapes: Key key0-0 is of shape shape0, Key key0-1 is of shape shape0.	Currently, the robot is at position f0-2f and its arm is empty. All the positions are open except the following: f0-1f has shape0 shaped lock, f2-1f has shape0 shaped lock. Key key0-0 is at position f2-2f. Key key0-1 is at position f4-1f.	The goal is to reach a state where the following facts hold: Key key0-0 is at f3-1f location and Key key0-1 is at f4-1f location.
Ferry	This is a ferry domain, where the task is to transport cars from their start to their goal locations, using a ferry. Each location is accessible by ferry from each other location. The cars can be debarked or boarded, and the ferry can carry only one car at a time.	There are 5 locations and 3 cars, numbered consecutively. Currently, the ferry is at l0 location and it is empty. The cars are at locations as follows: c0 and c2 are at l4; c1 is at l0.	The goal is to reach a state where the following facts hold: Car c1 is at location l3, Car c0 is at location l3, and Car c2 is at location l3.

Continued on next page

Table 7: Example of domain description, problem description and goal description of 13 domains in ACPBench. (Continued)

FloorTile	A set of robots use different colors to paint patterns in floor tiles. The robots can move around the floor tiles in four directions (up, down, left and right). Robots paint with one color at a time, but can change their spray guns to any available color. However, robots can only paint the tile that is in front (up) and behind (down) them, and once a tile has been painted no robot can stand on it. Robots need to paint a grid with black and white, where the cell color is alternated always. There are 2 robots and 12 tiles. The tiles locations are: tile_6 is to the right of tile_5, tile_12 is to the right of tile_11, tile_8 is to the right of tile_7, tile_5 is to the right of tile_4, tile_11 is to the right of tile_10, tile_3 is to the right of tile_2, tile_9 is to the right of tile_8, and tile_2 is to the right of tile_1. Further, tile_4 is down from tile_7, tile_8 is down from tile_11, tile_1 is down from tile_4, tile_9 is down from tile_12, tile_5 is down from tile_8, tile_7 is down from tile_10, tile_6 is down from tile_9, tile_3 is down from tile_6, and tile_2 is down from tile_5.	Currently, robot robot1 is at tile_8 and holding color white and robot robot2 is at tile_7 and holding color black; tile_12, tile_4, tile_3, tile_1, tile_9, tile_2, tile_10, tile_11, tile_5, and tile_6 are clear.	The goal is to reach a state where the following facts hold: Tile tile_7 is painted in black color, Tile tile_12 is painted in white color, Tile tile_4 is painted in white color, Tile tile_8 is painted in white color, Tile tile_11 is painted in black color, Tile tile_6 is painted in white color, Tile tile_9 is painted in black color, Tile tile_10 is painted in white color, and Tile tile_5 is painted in black color.

Continued on next page

Table 7: Example of domain description, problem description and goal description of 13 domains in ACPBench. (Continued)

Rovers	This is a Rovers domain where rovers must navigate between waypoints gathering data and transmitting it back to a lander. Rovers cannot navigate to all waypoints and this makes particular routes impassable to some of the rovers. Data transmission is also constrained by the visibility of the lander from the waypoints. There are 2 rovers, 5 waypoints, 2 stores, 2 cameras, 2 objectives numbered consecutively. Further, there is 1 lander and 3 modes for the camera namely colour, high resolution, and low resolution.	Rover(s) rover0 and rover1 are equipped for soil analysis. Rover(s) rover1 is equipped for rock analysis. Rover(s) rover0 and rover1 are equipped for imaging. Rover rover0 has store store0. Rover rover1 has store store1. Rover rover0 has camera0 on board. Rover rover1 has camera1 on board. Camera camera1 can be calibrated on objective0. Camera camera0 can be calibrated on objective0. Camera camera1 supports colour and low_res. Camera camera0 supports colour and low_res. Rover rover0 can traverse from waypoint4 to waypoint1, waypoint0 to waypoint1, waypoint1 to waypoint0, waypoint1 to waypoint4. Rover rover1 can traverse from waypoint0 to waypoint2, waypoint1 to waypoint2, waypoint2 to waypoint1, waypoint2 to waypoint0. Waypoint(s) are visible from waypoint2: waypoint3, waypoint0, and waypoint1. Waypoint(s) are visible from waypoint1: waypoint4, waypoint2, and waypoint0. Waypoint(s) are visible from waypoint0: waypoint4, waypoint2, waypoint1, and waypoint3. Waypoint(s) are visible from waypoint3: waypoint0 and waypoint2. Waypoint(s) are visible from waypoint4: waypoint0 and waypoint1. Objective objective0 is visible from waypoint1 and waypoint2. Objective objective1 is visible from waypoint4. Lander general is at waypoint waypoint3. Currently, Rover rover0 is at waypoint0. Rover rover1 is at waypoint2. Rocks can be sampled at the following location(s): waypoint0 and waypoint1. Soil can be sampled at the following location(s): waypoint0. Rovers rover0 and rover1 are available. Store(s) store0 and store1 are empty.	The goal is to reach a state where the following facts hold: Image objective0 was communicated in mode colour, Image objective1 was communicated in mode low_res, Rock data was communicated from waypoint waypoint0; Rock data was communicated from waypoint waypoint1;, Soil data was communicated from waypoint waypoint0;, and Image objective0 was communicated in mode low_res.
Visitall	This is a visitall domain where a robot in a grid must visit all the cells or places in the grid. There are some unavailable places in the grid. The grid size is 4x5, and the location cell names are of the form loc-xi-yj (e.g., loc-x0-y2 or loc-x1-y1). The grid cells are connected to their available neighbors. The unavailable cells are loc-x2-y3, loc-x1-y2, and loc-x0-y4.	Currently, the robot is in place loc-x0-y2.Place loc-x0-y2 has been visited.	The goal is to reach a state where the following facts hold: Place loc-x2-y0 has been visited, Place loc-x3-y3 has been visited, Place loc-x3-y4 has been visited.

Continued on next page

Table 7: Example of domain description, problem description and goal description of 13 domains in ACPBench. (Continued)

Depot	This is a depot domain, a combination of blocks and logistics. In this domain, trucks can transport crates, the crates can be stacked onto pallets using hoists. There are 2 depots, 4 hoists, 4 pallets, 2 distributors, 2 crates, 2 trucks, numbered consecutively.	Currently, crate1, crate0, pallet1, and pallet2 are clear; hoist1, hoist3, hoist0, and hoist2 are available; pallet0 is at depot0, hoist3 is at distributor1, truck1 is at depot0, pallet2 is at distributor0, hoist2 is at distributor0, truck0 is at distributor0, hoist1 is at depot1, crate1 is at distributor1, crate0 is at depot0, pallet3 is at distributor1, pallet1 is at depot1, and hoist0 is at depot0; crate0 is on pallet0 and crate1 is on pallet3.	The goal is to reach a state where the following facts hold: crate1 is on pallet0 and crate0 is on crate1.
Goldminer	A robotic arm is in a grid and can only move to locations that are connected to its current location. The 3x4 grid locations may have gold, hard rocks, or soft rocks. Rocks cannot be moved. The robotic arm can pick up laser or bomb. Only one item can be picked at a time. There is one laser is the grid that can be used to clear rocks. Robotic arm can fire laser at a location from a connected location. The locations are of the form fi-jf (e.g., f3-2f or f0-1f). The grid cells are connected to their neighbors (e.g., f1-2f is connected to the four neighbors f0-2f, f2-2f, f1-1f, and f1-3f). If a bomb is picked, it cannot be placed back. It can only be detonated at connected location that have soft rock. Bomb supply is available at f0-0f location.	Currently, the robot is at position f2-0f and its arm is empty. The following locations have hard rock: f2-1f, f0-3f, and f2-2f. The following locations have soft rock: f0-2f, f1-2f, f2-3f, f1-3f, f1-1f, and f0-1f. The gold is at f1-3f location. The laser is at f0-0f location.	The goal is to reach a state where the following facts hold: The robot is holding gold.

Continued on next page

Table 7: Example of domain description, problem description and goal description of 13 domains in ACPBench. (Continued)

Satellite	This domain consists of satellite(s) equipped with various instruments that can be switched on when the power is available. Each instrument has a calibration target object and supports taking images of objects in particular modes. When the instrument power is switched on, it is not calibrated. To calibrate an instrument, the satellite should point to the calibration target object and the instrument should be powered on. To take an image of an object, the satellite must point to that object and the instrument must be calibrated. There are 10 satellite(s), numbered consecutively. There are 7 possible target object(s): groundstation1, groundstation0, star2, planet5, star4, planet6, groundstation3. There are 3 image mode(s): image1, thermograph0, infrared2. There are 16 instrument(s), numbered consecutively. Satellite satellite0 has following instruments onboard: instrument0. $\cdots$ Instrument instrument11 supports image of mode infrared2 and its calibration target is groundstation3. Instrument instrument5 supports image of mode thermograph0 and its calibration target is groundstation1. $\cdots$	Currently, Satellite satellite6 is pointing to groundstation1. Satellite satellite5 is pointing to groundstation3. Satellite satellite3 is pointing to groundstation1. Satellite satellite2 is pointing to planet6. Satellite satellite1 is pointing to groundstation0. Satellite satellite7 is pointing to star2. Satellite satellite0 is pointing to groundstation1. Satellite satellite9 is pointing to star4. Satellite satellite4 is pointing to planet6. Satellite satellite8 is pointing to star2. Power is available on the following satellite(s): satellite1, satellite2, satellite5, satellite0, satellite6, satellite7, satellite8, satellite9, satellite4, satellite3.	The goal is to reach a state where the following facts hold: A thermograph0 mode image of target planet5 is available, Satellite satellite6 is pointing to star4, A infrared2 mode image of target planet6 is available, and Satellite satellite8 is pointing to planet6.
Swap	This is a swap domain where agents are swapping items or roles. Each agent is always assigned a single item/role. The goal is to obtain desired items/roles assigned. There are 8 agents: vic, alice, zoe, dave, heidi, carol, michelle, and xena. There are 8 items/roles: necklace, whale, iceskates, frisbee, guitar, quadcopter, slinky, and zebra.	Currently, zoe is assigned frisbee, heidi is assigned necklace, carol is assigned guitar, michelle is assigned zebra, dave is assigned slinky, xena is assigned whale, alice is assigned iceskates, and vic is assigned quadcopter.	The goal is to reach a state where the following facts hold: heidi is assigned guitar, michelle is assigned quadcopter.

Continued on next page

Table 7: Example of domain description, problem description and goal description of 13 domains in ACPBench. (Continued)

Alfworld	This is an alfworld domain where an agent is asked to carry different tasks such as: picking up objects, opening or closing receptacles, warming up an object in a microwave, cleaning an object in a sink, or toggling an object. There are 21 object types: 3 alarmclocks, 1 baseballbat, 1 basketball, 2 blindss, 1 book, 3 bowls, 3 cds, 3 cellphones, 2 chairs, 1 creditcard, 1 desklamp, 2 keychains, 2 laptops, 1 laundryhamperlid, 1 lightswitch, 1 mirror, 2 mugs, 3 pencils, 1 pen, 2 pillows, 2 windows, 7 receptacle types: 1 bed, 2 desks, 6 drawers, 1 garbagecan, 1 laundryhamper, 1 safe, 6 shelves, and 27 locations all numbered consecutively. The receptacles are at locations as follows. laundryhamper1 is at location8. shelf1 is at location20. drawer1 is at location21. bed1 is at location13. shelf3 is at location11. shelf4 is at location23. desk2 is at location10. drawer5 and drawer4 are at location12. desk1 is at location3. drawer6 is at location1. safe1 is at location6. shelf2 is at location25. shelf6 is at location24. drawer3 is at location17. drawer2 is at location18. shelf5 is at location22. garbagecan1 is at location2.	Currently, the objects are at locations as follows. bowl1, alarmclock1, mug1, cd1, and pencil1 are at location3. window2 is at location4. basketball1 is at location7. pen1, mug2, pencil3, cellphone2, and cd3 are at location10. pillow1, laptop2, book1, cellphone1, laptop1, and pillow2 are at location13. chair1 is at location21. laundryhamperlid1 is at location8. baseballbat1 is at location9. pencil2 and creditcard1 are at location22. desklamp1, bowl2, and alarmclock3 are at location23. bowl3 is at location24. keychain2 and keychain1 are at location6. mirror1 is at location19. cd2 is at location2. lightswitch1 is at location14. cellphone3 is at location12. chair2 is at location26. blinds2 is at location15. blinds1 is at location16. alarmclock2 is at location11. window1 is at location5. agent agent1 is at location location27. The objects are in/on receptacle as follows. pen1, cellphone2, cd3, bowl2, mug2, desklamp1, pencil3, and alarmclock3 are on desk2. bowl1, mug1, alarmclock1, pencil1, and cd1 are on desk1. pencil2 and creditcard1 are on shelf5. keychain2 and keychain1 are in safe1. cd2 is in garbagecan1. laptop2, laptop1, book1, cellphone1, pillow2, and pillow1 are in bed1. cellphone3 is in drawer5. alarmclock3, bowl2, and desklamp1 are on shelf4. alarmclock2 is on shelf3. bowl3 is on shelf6. drawer3, drawer6, safe1, and drawer1 are closed. desklamp1 is off. Nothing has been validated. agent1's hands are empty.	The goal is to reach a state where the following facts hold: an object of type book is examined under an object of type desklamp.

Table 8: Example questions for 7 ACPBench tasks. For each question, an LLM was provided with context and then the question. The context contains natural language descriptions.

Task	Context	Bool Questions	MCQ Questions
Applicability	domain + problem	Is the following action applicable in this state: `debark the car c2 from the ferry to location l1?`	Which of the following actions will be applicable in this state? A. `travel by sea from location l2 to location l1.` B. `debark the car c2 to location l0 from the ferry.` C. `travel by sea from location l0 to location l1.` D. `board the car c5 at location l1 on to the ferry.`
Progression	domain + problem	Will the fact "`The ferry is empty`" hold after performing the action "`embark the car c0 at location l1 on to the ferry`" in the current state?	Which of the following facts hold after performing the action "`sail from location l1 to location l0`" in the current state? A. `The ferry is at l0 location and Car c3 is at location l0.` B. `The ferry is at l0 location and The ferry is at l1 location.` C. `The ferry is at l0 location and Car c4 is at location l0.` D. `Car c4 is at location l0 and Car c3 is at location l0.`
Reachability	domain + problem	Is it possible to transition to a state where the following holds: `The ferry is at l0 location and Car c0 is on board the ferry?`	Which of the following options can hold in a state that can potentially be reached? A. `The ferry is at c9 location and Car c6 is at location l2.` B. `The ferry is at l0 location and The ferry is at l2 location.` C. `Car l2 is on the ferry and Car c4 is at location l0.` D. `There are no cars on the ferry and Car c5 is at location l1.`

Validation	domain + problem + goal	Is the following sequence of actions a plan for the current state?
		`sail from location l2 to location l0,`
		`board car c6 at location l0,`
		`sail from location l0 to location l2, ...`
			Which of the following claims is true with regard to the following sequence of actions
			`board the car c12 at the location l1,`
			`travel by sea from location l1 to location l0,`
			`debark the car c12 from the ferry to location l0,`
			`board the car c33 at the location l0, ...`
			A. `The sequence is not valid.`
			B. `The sequence is applicable, but does not achieve the goal.`
			C. `The sequence is a plan.`
			D. `The sequence is not applicable.`

Continued on next page

Table 8: Example questions for 7 ACPBench tasks. For each question, an LLM was provided with context and then the question. The context contains natural language descriptions. (Continued)

Act. Reachability	domain + problem	Is it possible to transition to a state where the action "`embark the car l2 at location c8 on to the ferry`" can be applied?	Which of the following actions can eventually be applied? A. `board the car c20 at location l0.` B. `travel by sea from location c43 to location c4.` C. `debark the car c2 to location c8 from the ferry.` D. `board the car c26 at location c23.`
Justification	domain + problem + goal	Given the plan: "`board the car c13 at location l1 on to the ferry, sail from location l1 to location l3, debark car c13 to location l3 from the ferry, board the car c29 at location l3 on to the ferry,...`"; can the following action be removed from this plan and still have a valid plan: `sail from location l1 to location l3`?	Given the plan: "`travel by sea from location l2 to location l1, board the car c0 at the location l1, travel by sea from location l1 to location l0, debark the car c0 from the ferry to location l0,...`"; which of the following pairs of consecutive actions can be removed from this plan and still have a valid plan? A. `board the car c0 at the location l1 and travel by sea from location l1 to location l0.` B. `board the car c0 at the location l0 and travel by sea from location l0 to location l2.` C. `board the car c6 at the location l0 and travel by sea from location l0 to location l2.` D. `debark the car c0 from the ferry to location l0 and board the car c0 at the location l0?`
Landmark	domain + problem + goal	Is the following fact a landmark (must hold at some point along any plan) for the current state? `There are no cars on the ferry`	Which of the following facts is a landmark (must hold at some point along any plan) for the current state? A. `Car c7 is on board the ferry.` B. `Car c5 is at location l2.` C. `Car c7 is at location l2.` D. `Car c0 is at location l1.`

Model	Blocksworld		Logistics		Grippers		Grid		Ferry		FloorTile		Rovers		VisitAll		Depot		Goldminer		Satellite		Swap		Alfworld	
	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ
Phi-3 128K	40.0	27.14	61.43	25.71	60.0	28.57	50.0	45.71	55.71	32.86	45.71	31.43	48.57	27.14	71.43	45.71	67.14	44.29	58.57	32.86	50.0	42.86	55.71	31.43	51.43	35.71
Gemma 7B	44.29	27.43	50.0	24.86	51.71	22.29	50.86	33.71	54.29	28.29	54.0	33.71	57.14	26.57	62.57	30.57	40.57	25.43	42.29	34.86	50.86	31.14	53.14	28.0	52.86	27.14
Granite 7B	39.71	25.43	45.71	27.14	52.86	22.86	51.43	30.0	52.86	32.0	40.0	24.29	47.14	30.0	64.29	38.57	50.0	38.57	42.86	25.71	37.14	34.29	58.29	32.86	27.14	27.14
Mistral 7B	40.31	18.46	56.15	26.92	61.54	25.23	50.77	30.77	53.08	31.54	64.0	23.08	62.31	32.15	60.77	38.77	54.92	35.08	54.62	16.15	49.23	25.38	54.62	31.54	46.15	36.15
Mistral instruct 7B	47.14	36.86	53.71	36.29	64.29	54.29	57.71	38.0	65.71	32.57	54.29	30.0	43.71	28.57	59.71	35.71	63.14	38.57	62.86	41.43	43.14	32.86	52.86	44.86	40.0	31.43
Granite-code 8B	55.71	15.71	58.57	39.43	67.43	22.86	41.43	24.0	50.86	33.14	44.29	19.43	52.29	27.43	62.86	38.57	42.0	28.29	50.0	28.57	47.14	34.29	54.29	31.71	47.14	31.43
Granite8b code instruct	48.57	31.43	52.86	33.14	55.71	32.86	47.14	30.57	57.14	25.71	41.43	21.43	51.14	34.29	58.57	36.29	47.14	31.43	40.0	32.86	49.43	38.57	51.43	41.43	50.29	28.57
LLAMA-3 8B	54.29	44.29	61.43	48.57	67.14	40.0	58.29	47.43	72.86	31.43	47.14	45.71	52.86	34.57	71.43	55.71	60.86	36.86	58.86	50.0	65.71	51.43	64.29	40.0	55.71	41.43
LLAMA-3.1 8B	54.29	47.14	52.86	35.71	58.57	37.14	52.86	38.57	48.57	40.0	51.43	40.0	60.0	30.0	51.43	52.86	51.43	48.57	48.57	42.86	41.43	34.29	58.57	42.86	42.86	44.29
Mixtral 8x7B	65.43	48.29	60.29	40.0	71.43	46.0	65.71	59.71	80.0	47.14	53.71	46.0	53.43	38.57	81.71	60.0	71.14	56.0	65.14	49.71	64.0	65.43	63.71	51.43	49.71	52.86
Granite 13B	46.29	26.57	64.29	26.86	57.43	27.14	37.14	37.14	52.86	26.0	40.86	24.29	49.43	12.29	44.86	31.71	38.0	35.71	40.0	13.71	51.43	21.43	44.29	31.43	48.0	28.0
Codestral 22B	71.43	31.43	70.0	38.57	80.0	35.71	68.57	42.86	68.57	45.71	60.0	32.86	64.29	35.71	60.0	48.57	68.57	38.57	64.29	41.43	65.71	42.86	62.86	52.86	60.0	40.0
Mixtral 8x22B	46.86	44.29	65.14	36.29	66.57	38.57	50.57	45.14	67.71	34.86	39.71	22.0	45.71	32.0	60.57	42.57	69.43	42.86	50.0	43.43	61.71	44.86	66.57	44.86	20.0	24.86
Deepseek-33b instruct	51.43	31.14	64.29	42.86	62.86	20.86	56.0	45.71	61.43	30.0	48.57	32.86	60.0	35.71	70.0	44.29	60.0	35.71	57.14	36.29	54.29	38.57	46.86	22.86	48.57	35.71
CodeLLAMA 34B	53.14	42.0	58.57	34.29	58.57	27.14	52.86	37.14	61.43	31.43	58.57	35.71	60.0	21.43	62.86	54.29	52.86	36.86	57.14	37.14	61.14	28.57	61.43	42.86	60.0	30.0
LLAMA-2 7OB	51.43	28.57	59.71	24.29	53.43	30.0	50.0	25.71	58.57	14.57	57.14	15.71	53.14	32.86	61.43	42.86	48.57	36.86	57.14	28.57	57.43	31.43	58.57	36.0	47.14	35.71
CodeLLAMA 70B	51.14	27.43	52.86	30.0	60.0	24.57	45.43	42.86	59.71	36.57	46.86	30.29	48.0	32.86	63.71	35.71	46.86	30.0	45.43	40.86	58.86	30.86	45.71	32.57	21.71	34.29
LLAMA-3 70B	81.43	70.0	77.71	80.0	80.0	90.86	84.57	72.86	83.71	78.57	74.29	65.71	75.71	66.57	68.57	72.86	75.71	84.29	82.86	71.43	81.71	58.57	75.71	77.14	72.86	68.86
LLAMA-3.1 70B	74.29	55.43	73.71	69.14	73.14	79.43	60.29	68.0	61.14	78.57	60.29	61.71	67.14	56.57	59.71	62.86	63.14	66.57	70.86	72.57	62.0	53.43	64.86	76.86	64.0	60.29
LLAMA-3.1 405B	81.43	71.43	92.86	85.71	88.57	93.14	83.43	74.29	81.43	81.43	57.14	61.43	80.0	71.43	81.43	82.86	85.71	71.43	81.43	78.29	71.43	72.57	81.43	85.71	70.0	67.14
GPT-4o Mini	70.0	38.57	78.57	58.57	77.14	48.57	80.0	58.57	81.43	57.14	64.29	58.57	71.43	32.86	72.86	60.0	75.71	67.14	88.57	64.29	80.0	55.71	84.29	68.57	77.14	67.14
GPT-4o	87.14	75.71	80.0	70.0	80.0	84.29	88.57	82.86	90.0	80.0	61.43	54.29	72.86	55.71	87.14	74.29	77.14	75.71	90.0	80.0	75.71	72.86	94.29	82.86	72.86	74.29

Table 9: Accuracy values for 22 pretrained LLMs on Bool and MCQ tasks in ACPBench; segregated per domain.

Model	Applicability		Progression		Reachability		Validation		Action Reach.		Justification		Landmark		Mean	
	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ
Phi-3 128K	66.92	40.77	43.85	56.92	58.46	23.85	55.38	25.38	63.33	27.50	51.54	26.15	73.85	48.46	55.64	35.08
Gemma 7B	47.69	25.23	48.62	34.46	53.85	20.00	55.38	29.23	58.17	21.00	48.31	29.08	46.92	33.85	51.73	29.15
Granite 7B	49.23	23.85	52.62	35.38	57.38	26.92	55.38	30.00	51.67	20.00	48.00	26.92	52.31	32.31	47.41	30.22
Mistral 7B	54.62	28.46	53.08	20.00	49.23	26.15	46.92	22.62	45.00	16.67	49.23	22.31	47.69	29.23	54.91	28.45
Mistral instruct 7B	67.69	27.69	50.77	39.23	65.38	23.08	64.15	33.85	53.33	24.17	53.08	25.38	76.15	50.77	54.99	37.41
Granite-code 8B	52.31	14.62	51.54	24.62	46.92	13.08	45.38	33.69	41.67	11.17	50.77	16.15	53.85	20.00	52.48	29.12
Granite8b code instruct	53.08	13.85	55.69	25.38	49.23	23.08	49.23	34.92	53.33	12.50	50.77	18.46	59.23	28.46	50.53	32.63
LLAMA-3 8B	59.23	30.77	51.54	48.46	55.38	25.38	53.08	36.92	55.83	21.67	54.77	26.15	66.15	40.77	61.53	44.05
LLAMA-3.1 8B	53.85	36.92	48.46	45.38	52.31	33.85	56.15	28.46	58.33	33.33	49.23	22.31	47.69	45.38	52.23	41.41
Mixtral 8x7B	67.08	45.38	52.62	62.15	60.62	49.23	50.62	43.69	44.83	43.83	52.77	24.62	77.23	66.92	65.61	51.46
Granite 13B	57.69	36.92	50.00	38.46	58.46	29.23	70.00	38.46	61.67	43.33	68.31	26.92	62.31	60.00	47.79	26.62
Codestral 22B	72.31	30.77	56.92	53.85	65.38	24.62	48.46	26.15	54.17	23.33	51.54	23.08	71.54	34.62	67.07	40.97
Mixtral 8x22B	50.15	40.92	52.77	47.69	65.38	30.92	56.92	38.46	50.50	28.33	47.69	33.85	64.00	63.08	55.30	38.50
Deepseek-33b instruct	60.00	18.92	46.00	29.23	63.08	24.62	56.15	34.31	48.83	15.83	50.77	19.23	70.46	22.31	57.58	35.11
CodeLLAMA 34B	60.92	16.92	53.08	42.31	52.31	20.00	48.46	16.92	44.17	14.00	50.77	22.31	69.23	40.77	58.94	35.66
LLAMA-2 70B	53.38	27.69	49.38	47.08	38.46	25.38	44.00	36.31	43.17	18.83	51.08	25.85	53.85	40.77	55.57	29.71
CodeLLAMA 70B	57.54	32.92	51.08	38.46	46.92	35.23	44.62	34.62	42.50	31.00	50.00	24.62	64.62	46.92	50.26	33.30
LLAMA-3 70B	90.00	71.54	62.31	86.92	76.15	78.46	60.15	48.00	54.17	57.67	54.62	83.08	84.62	74.31	78.73	74.36
LLAMA-3.1 70B	40.00	72.00	40.31	85.38	65.85	71.69	48.00	40.77	57.00	58.83	35.54	70.92	17.38	76.92	66.43	66.90
LLAMA-3.1 405B	88.46	83.08	90.77	90.77	85.38	83.08	84.46	50.00	74.17	72.50	77.69	89.23	83.08	69.23	80.43	77.39
GPT-4o Mini	70.77	66.92	68.46	80.77	80.00	58.46	54.62	21.54	57.50	55.83	56.92	44.62	64.62	66.15	77.63	56.83
GPT-4o	68.46	83.08	71.54	84.62	74.62	77.69	56.15	37.69	60.00	69.17	59.23	86.92	76.92	80.00	81.95	74.64

Table 10: Accuracy of 22 LLMs on 7 ACPBench tasks (Bool as well as MCQ) with zeroshot IO prompt. The right-most column is mean across tasks.

	Applicability		Progression		Reachability		Validation		Action Reach.		Justification		Landmark		
Domain	Base	Finetuned	Base	Finetuned	Base	Finetuned	Base	Finetuned	Base	Finetuned	Base	Finetuned	Base	Finetuned	Diff
Ferry	60	100	30	100	80	100	70	100	50	100	40	100	50	80	42.86
Logistics	50	100	80	100	30	100	30	100	30	100	40	100	50	70	51.43
Blocksworld	70	100	80	90	50	100	30	100	70	100	80	100	60	90	34.29
Grid	20	100	70	100	80	100	40	100	60	100	60	100	60	60	38.57
Floortile	50	100	50	100	50	100	40	100	30	100	90	90	60	90	44.29
Grippers	60	100	30	100	30	100	60	100	40	100	50	100	70	90	50.00
Rovers	60	90	50	90	30	80	30	100	50	80	70	90	60	80	37.14
Visitall	60	100	30	100	30	100	60	100	30	100	20	100	70	90	55.71
Depot	50	90	30	70	30	70	30	90	40	80	20	100	30	60	47.14
Goldminer	70	60	40	90	80	90	30	80	30	50	40	100	60	90	30.00
Satellite	70	40	80	100	50	80	50	80	30	100	50	100	30	70	30.00
Swap	20	90	30	90	40	90	70	60	40	100	50	90	70	100	42.86
Alfworld	40	90	70	50	30	80	50	70	NA	NA	50	100	30	40	26.67
Mean	52	89	52	91	47	92	45	91	45	93	51	98	54	78	40.84

Table 11: Per-domain comparison of 7 tasks on the boolean questions between the Base model, Granite8b code base, and the Finetuned model. The first 8 domains are domains that are in the training set (seen domains), and the last 5 domains are unseen domains. "Diff" shows the average difference between the base and fine-tuned model.

	Applicability		Progression		Reachability		Validation		Action Reach.		Justification		Landmark		
Domain	Base	Finetuned	Base	Finetuned	Base	Finetuned	Base	Finetuned	Base	Finetuned	Base	Finetuned	Base	Finetuned	Diff
Ferry	30	90	40	100	0	100	20	80	10	100	0	90	20	100	77.14
Logistics	10	90	20	100	0	100	40	80	18	100	0	90	10	80	77.43
Blocksworld	10	70	20	50	40	90	30	70	10	90	40	100	50	100	52.86
Grid	0	100	10	100	20	100	20	100	10	100	40	100	20	100	82.86
Floortile	40	100	50	100	10	100	30	90	20	90	10	90	10	100	71.43
Grippers	20	100	10	100	30	100	70	100	26	100	10	100	10	100	74.86
Rovers	0	90	20	100	0	100	68	80	0	80	0	100	10	60	73.14
Visitall	10	100	10	100	30	100	30	100	20	100	30	100	30	80	75.71
Depot	0	60	0	90	0	60	40	50	0	30	20	100	10	60	54.29
Goldminer	0	100	20	100	20	100	20	60	0	90	50	90	10	100	74.29
Satellite	20	60	30	90	10	30	20	50	10	80	0	90	20	100	55.71
Swap	20	50	40	40	10	50	30	50	10	80	10	100	40	100	44.29
Alfworld	30	40	50	90	10	40	20	60	NA	NA	0	20	20	100	36.67
Mean	15	81	25	89	13	82	34	75	11	87	16	90	20	91	65.44

Table 12: Per-domain comparison of 7 tasks on the multiple choice questions between the Base model, Granite8b code base, and the Finetuned model. The first 8 domains are domains that are in the training set (seen domains), and the last 5 domain are unseen domains. "Diff" shows the average difference between the base and fine-tuned model.

Model	Applicability		Progression		Reachability		Validation		Action Reach.		Justification		Landmark		Mean	
	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ
Phi-3 128K	67.50	32.50	71.25	52.50	51.25	30.00	51.25	20.00	51.25	35.00	43.75	32.50	42.50	28.75	54.11	33.04
Gemma 7B	64.25	29.25	70.25	34.00	52.50	21.00	50.00	18.75	56.25	39.25	55.00	39.00	23.50	17.75	53.11	28.43
Granite 7B	57.50	25.50	63.50	36.25	52.50	32.50	30.00	28.75	51.25	30.00	42.50	25.00	47.50	23.50	49.25	28.79
Mistral 7B	55.00	36.25	76.25	37.50	52.50	32.50	49.00	17.50	58.75	21.25	52.50	35.00	42.50	17.50	55.21	28.21
Mistral instruct 7B	66.25	32.50	67.50	46.25	62.50	33.75	49.25	35.75	50.00	32.50	37.50	32.50	57.50	42.50	55.79	36.54
Granite-code 8B	62.50	36.75	76.25	33.25	52.50	20.75	40.50	20.00	56.25	28.75	50.00	35.00	41.25	18.50	54.18	27.57
Granite8b code instruct	53.75	37.50	72.50	34.75	50.00	28.75	49.25	21.25	46.25	44.00	48.75	30.00	40.50	18.75	51.57	30.71
LLAMA-3 8B	73.75	52.50	73.75	57.50	56.00	41.75	50.00	45.00	65.00	47.50	55.00	32.50	51.25	27.50	60.68	43.46
LLAMA-3.1 8B	61.25	63.75	71.25	52.50	52.50	37.50	62.50	30.00	45.00	31.25	52.50	38.75	31.25	27.50	53.75	40.18
Mixtral 8x7B	80.00	56.00	79.25	61.25	77.50	35.00	61.75	36.50	58.50	53.75	51.25	48.75	57.00	46.25	66.46	48.21
Granite 13B	42.75	32.50	53.25	22.25	48.75	33.75	50.00	35.00	50.00	23.25	46.25	28.75	53.00	10.00	49.14	26.50
Codestral 22B	88.75	41.25	87.50	50.00	55.00	30.00	70.00	27.50	55.00	36.25	67.50	62.50	51.25	25.00	67.86	38.93
Mixtral 8x22B	86.00	37.75	77.00	60.00	52.50	43.25	37.50	20.00	56.25	26.50	40.00	44.00	38.25	27.25	55.36	36.96
Deepseek-33b instruct	76.25	37.50	73.75	47.75	52.50	34.00	52.50	41.25	52.50	28.75	43.75	30.00	64.00	28.75	59.32	35.43
CodeLLAMA 34B	82.50	45.00	78.75	48.75	52.50	24.25	49.00	36.25	43.75	36.25	57.50	37.50	43.75	20.00	58.25	35.43
LLAMA-2 70B	78.75	23.75	76.25	40.00	52.50	26.25	48.25	16.25	56.25	20.00	53.75	49.00	23.50	12.50	55.61	26.82
CodeLLAMA 70B	77.75	41.50	61.25	55.25	52.50	28.50	40.00	20.00	55.75	27.50	50.00	28.75	37.00	26.25	53.46	32.54
LLAMA-3 70B	90.00	86.25	93.75	86.25	87.50	82.50	78.75	52.00	65.75	73.25	61.25	82.50	70.75	60.00	78.25	74.68
LLAMA-3.1 70B	95.50	84.25	90.25	90.25	57.75	54.50	64.50	45.50	67.50	62.00	58.75	69.25	29.25	59.50	66.21	66.46
LLAMA-3.1 405B	97.50	87.50	93.75	92.50	61.25	80.00	75.50	57.75	70.00	78.75	90.00	82.50	77.50	65.00	80.79	77.71
GPT-4o Mini	88.75	67.50	96.25	77.50	82.50	37.50	57.50	41.25	55.00	22.50	71.25	63.75	70.00	51.25	74.46	51.61
GPT-4o	97.50	90.00	96.25	93.75	78.75	75.00	50.00	45.00	62.50	57.50	83.75	72.50	97.50	71.25	80.89	72.14

Table 13: Accuracy of 22 LLMs on 8 seen domains of ACPBench. All models were evaluated with two in-context examples and Chain-of-Thought prompt. The right-most column is mean across tasks.

Model	Applicability		Progression		Reachability		Validation		Action Reach.		Justification		Landmark		Mean	
	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ	Bool	MCQ
Phi-3 128K	64.00	34.00	64.00	56.00	54.00	20.00	50.00	18.00	57.50	27.50	58.00	36.00	60.00	76.00	58.21	38.21
Gemma 7B	61.60	27.60	56.40	26.40	54.00	26.40	42.00	22.00	54.50	25.00	44.00	32.40	34.00	50.40	49.50	30.03
Granite 7B	56.00	36.00	42.00	34.00	48.00	38.00	35.60	22.00	42.50	25.00	38.00	26.00	48.00	46.00	44.30	32.43
Mistral 7B	72.00	26.00	68.00	40.00	54.00	22.00	46.00	18.00	77.50	15.00	42.00	22.00	24.00	58.00	54.79	28.71
Mistral instruct 7B	58.00	30.00	52.00	48.00	60.00	32.00	56.80	36.80	37.50	37.50	52.00	24.00	58.00	64.00	53.47	38.90
Granite-code 8B	54.00	25.20	60.00	36.00	52.00	30.00	50.00	12.40	60.00	20.00	42.00	34.00	30.80	62.40	49.83	31.43
Granite8b code instruct	58.00	24.00	64.00	34.00	52.00	30.00	40.40	24.00	35.00	30.00	42.00	36.00	49.20	70.00	48.66	35.43
LLAMA-3 8B	71.60	44.00	72.00	53.60	54.00	40.00	54.00	56.00	60.50	15.00	61.60	32.00	66.00	70.00	62.81	44.37
LLAMA-3.1 8B	72.00	46.00	52.00	40.00	54.00	28.00	56.00	50.00	37.50	22.50	38.00	56.00	38.00	60.00	49.64	43.21
Mixtral 8x7B	69.20	60.40	65.60	61.60	73.60	48.00	72.00	32.00	41.50	57.50	62.00	55.60	63.60	82.00	63.93	56.73
Granite 13B	40.80	24.00	51.20	18.40	46.00	20.00	54.00	34.00	35.50	32.50	44.00	26.00	46.00	34.00	45.36	26.99
Codestral 22B	78.00	36.00	78.00	54.00	54.00	26.00	60.00	20.00	50.00	42.50	68.00	62.00	72.00	70.00	65.71	44.36
Mixtral 8x22B	72.40	37.60	64.80	46.00	46.00	41.60	38.00	12.00	63.00	30.50	48.00	45.60	55.20	74.00	55.34	41.04
Deepseek-33b instruct	62.00	36.80	60.00	44.00	54.00	28.00	50.00	32.00	45.00	25.00	52.00	20.00	59.60	56.00	54.66	34.54
CodeLLAMA 34B	78.00	38.00	64.00	36.00	54.00	28.00	52.00	16.00	72.00	27.50	52.00	32.00	52.00	73.60	60.57	35.87
LLAMA-2 70B	78.00	26.00	64.00	31.60	54.00	28.00	56.40	16.00	70.00	26.00	42.00	66.00	26.00	47.60	55.77	34.46
CodeLLAMA 70B	70.00	27.60	44.40	49.20	42.40	16.00	40.00	14.00	37.50	31.50	42.00	36.00	37.20	68.00	44.79	34.61
LLAMA-3 70B	92.00	76.00	92.00	86.00	88.00	82.00	78.40	64.00	50.00	42.50	64.00	90.00	90.00	72.40	79.20	73.27
LLAMA-3.1 70B	89.20	84.40	89.20	81.20	67.20	55.60	68.80	48.40	54.00	50.00	54.00	67.20	43.20	84.80	66.51	67.37
LLAMA-3.1 405B	92.00	86.00	92.00	96.00	56.00	82.00	80.00	71.20	55.00	37.50	90.00	94.00	92.00	66.00	79.57	76.10
GPT-4o Mini	94.00	84.00	94.00	82.00	78.00	42.00	84.00	54.00	52.50	20.00	88.00	80.00	88.00	94.00	82.64	65.14
GPT-4o	96.00	88.00	92.00	84.00	80.00	80.00	80.00	68.00	47.50	42.50	96.00	94.00	92.00	92.00	83.36	78.36

Table 14: Accuracy of 22 LLMs on 5 unseen domains of ACPBench. All models were evaluated with two in-context examples and Chain-of-Thought prompt. The right-most column is mean across tasks.