Dynamic planning with an LLM

Citation for published version:
Dagan, G, Keller, F & Lascarides Keller, A 2024, Dynamic planning with an LLM. in *Proceedings of the Language Gamification Workshop 2024 at NeurIPS.* Neural Information Processing Systems Foundation (NeurIPS), pp. 1-14, Language Gamification Workshop 2024 at NeurIPS, Vancouver, British Columbia, Canada, 14/12/24. https://doi.org/10.48550/arXiv.2308.06391

Digital Object Identifier (DOI):
10.48550/arXiv.2308.06391

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Proceedings of the Language Gamification Workshop 2024 at NeurIPS

Dynamic Planning with a LLM

Gautier Dagan
University of Edinburgh
gautier.dagan@ed.ac.uk

Frank Keller
University of Edinburgh
frank@ed.ac.uk

Alex Lascarides Keller
University of Edinburgh
alex@ed.ac.uk

Abstract

While Large Language Models (LLMs) can solve many NLP tasks in zero-shot settings, applications involving embodied agents remain problematic. In particular, plans that require multi-step reasoning become difficult and too costly as the context window grows. Planning requires understanding the likely effects of actions and identifying whether the current environment satisfies the goal. While symbolic planners can often find optimal solutions quickly, their capacity to handle noisy observations and uncertainty is relatively rudimentary, severely limiting their practical use. In contrast, Large Language Models (LLMs) cope with noisy observations and high levels of uncertainty. This paper presents LLM Dynamic Planner (LLM-DP): a neuro-symbolic framework where an LLM works hand-in-hand with a traditional planner to solve an embodied task. Given action-descriptions, LLM-DP solves Alfworld more successfully and efficiently than a LLM-only ReAct baseline.

1 Introduction

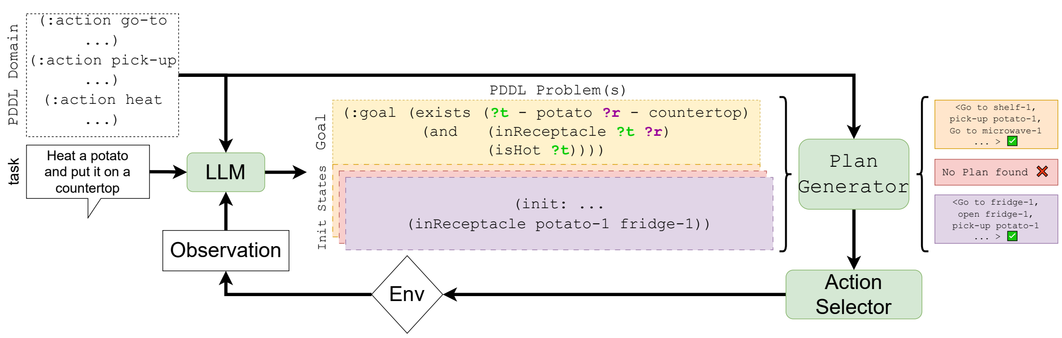

Figure 1: LLM Dynamic Planner (LLM-DP). The LLM grounds observations and processes natural language instructions into PDDL to use with a symbolic planner. LLM-DP can solve plans involving previously unknown objects because the LLM generates plausible predicates for them through semantic and pragmatic inference. Through sampling, multiple plans can be found, and an Action Selector decides whether to act, to review its understanding of the problem, or to ask for clarification.

Incorporating LLMs into embodied agents that interact with the environment presents substantial challenges. As well as hallucinating, LLMs are brittle to the phrasing of prompts (Ji et al., 2022) and are ill-equipped for naive long-term planning—managing an extensive context over multiple steps is complex and resource-consuming (Silver et al., 2022; Liu et al., 2023). Various approaches aim to improve LLM performance, for instance by augmenting the context with a reasoning trace (Wei et al., 2022; Wang et al., 2023b; Yao et al., 2023). But they frequently involve high computational costs and still face challenges dealing with the limits of the context window and hallucinations, compromising the quality of the plans. Conversely, symbolic planners find optimal plans efficiently (Hoffmann and

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

Nebel, 2001; Lipovetzky et al., 2014). But they have high information demands that cannot always be met in real-world scenarios (McDermott, 2000): for instance, they require knowing a complete and accurate description of the goal, but that may be impossible before exploring the environment through actions.

In this work, **we introduce the LLM Dynamic Planner (LLM-DP), a neuro-symbolic framework that integrates an LLM with a symbolic planner to solve embodied tasks**. LLM-DP capitalises on the LLM's ability to understand actions and their impact on their environment and combines it with the planner's efficiency in finding solutions. Using domain knowledge, LLM-DP solves the Alfworld test set faster (in number of steps) and more efficiently (in number of tokens used) than a LLM-only (ReAct) approach. The remainder of this paper explores the architecture of LLM-DP, discusses how to combine the strengths of LLMs and symbolic planning and presents potential research avenues for future work in LLM-driven agents.

2 Related Work

Symbolic Planners operate over symbolic representations of the world to find a sequence of actions that transition from the current state to a goal state (Fikes and Nilsson, 1971). Since the introduction of `PDDL` (McDermott, 2000), an array of efficient planning algorithms have been developed, via heuristics that decompose the goal or search over relaxed versions of the problem (Hoffmann and Nebel, 2001; Lipovetzky et al., 2014). These planners find high-quality or optimal solutions quickly in well-defined domains, but their up-front requirement for comprehensive problem and domain descriptions limits their practical use in complex real-world settings.

In contrast to symbolic planners, **LLMs** have shown promise in adapting to noisy planning and reasoning tasks through various methods. For instance, Chain-of-Thought (Wei et al., 2022), Self-Consistency (Wang et al., 2023b), and Reasoning via Planning (Hao et al., 2023) augment the context with a reasoning trace that the LLM generates to improve its final prediction. Alternatively, giving the LLM access to tools/APIs (Schick et al., 2023; Patil et al., 2023), external knowledge bases (Peng et al., 2023; Hu et al., 2023), code (Surís et al., 2023), or symbolic reasoners (Yang et al., 2023) can enrich the LLM's context and ability to reason so as to improve its performance in planning: the LLM can learn when and how to do this enrichment via fine-tuning or prompting. In a parallel direction, works such as ReAct (Yao et al., 2023), Reflexion (Shinn et al., 2023), AutoGPT (Significant-Gravitas, 2023), and Voyager (Wang et al., 2023a) take an agent-based approach, augmenting reasoning by iteratively feeding environment observations back to the LLM. ReAct (Yao et al., 2023) allows the LLM agent to take either an action or a 'thinking' step—effectively an agent-driven Chain-of-Thought prompting. Voyager (Wang et al., 2023a) incrementally builds an agent's capabilities from its interactions with the environment and an accessible memory component (skill library). While many of these works show promising results (Wang et al., 2023a), they still require many expensive calls to the LLMs, are limited by the LLM's context window, and do not guarantee optimal plans.

3 Alfworld

Alfworld (Shridhar et al., 2020) is a text-only home environment where an agent is tasked with seven possible tasks, such as interacting with one or more objects and placing them in a specific receptacle. At the start of each episode, the goal is given in natural language, and the initial observation does not include the location of any objects. The agent must navigate the environment to search for the relevant objects and perform the correct actions. The possible locations are known, and the agent can navigate to any receptacle by using a 'go to' action. However, since none of the objects' locations are initially observed, the agent must be able to plan around uncertainty, estimate where objects are likely to be observed and adjust accordingly.

4 LLM-DP

To tackle an embodied environment like Alfworld, we introduce the Large Language Model Dynamic Planner (LLM-DP), which operates as a closed-loop agent. LLM-DP uses a combination of language understanding and symbolic reasoning to plan and solve tasks in the simulated environment. The model tracks a World State $\mathcal{W}$ and beliefs $\mathcal{B}$ about predicates in the environment, uses an LLM to

	Average Accuracy (%)							
Model	clean	cool	examine	heat	put	puttwo	overall ± *std* (↑)	LLM Tokens ± *std* (↓)
`LLM-DP`	1.00	1.00	0.80	0.99	1.00	1.00	$\mathbf{0.97} \pm 0.00$	$702k \pm 16k$
`LLM-DP-random`	0.99	0.98	0.83	1.00	1.00	1.00	$\mathbf{0.97} \pm 0.01$	$\mathbf{67}k \pm 0$
`ReAct` (Yao et al., 2023)	0.61	0.81	0.89	0.30	0.79	0.47	0.64	—*
`ReAct` (ours)	0.61	0.76	0.14	0.64	0.95	0.73	0.65 ± 0.02	$9.48M \pm 231k$

(a) The average accuracy and number of LLM Tokens processed (context + generation) for each model. $*$Not reported.

	Average Episode Length						
Model	clean	cool	examine	heat	put	puttwo	overall ± *std* (↓)
`LLM-DP`	13.85	13.25	9.46	12.18	10.43	15.91	12.53 ± 7.11
`LLM-DP-random`	15.25	13.97	9.77	14.19	13.27	19.75	14.35 ± 7.71
`ReAct` (ours)	21.09	14.89	31.23	18.75	15.88	22.08	20.27 ± 12.32

(b) The average episode length for each model, where the length of an episode denotes how many actions the agent has taken or attempted to take to complete a task. We do not count the 'thinking' action of `ReAct` as an action in this metric.

Table 1: Summary of model performance on the Alfword test set. `LLM-DP` and `LLM-DP-random` have different sampling strategies: `LLM-DP` uses an LLM to generate $n = 3$ plausible world states, while `LLM-DP-random` randomly samples $n = 3$ plausible world states. We evaluate each setup with five seeds and report the average for all results.

translate the task description into an executable goal state and samples its beliefs to generate plausible world states. We describe the working of the LLM-DP agent as pseudo-code in Appendix A.

We make several **simplifying assumptions** when applying LLM-DP to Alfworld:

1. **Known action-descriptions and predicates**: Input to the planner and the LLM requires the `PDDL` domain file: i.e., all predicates and action schemata (with preconditions and effects).
2. **Perfect observations**: The Alfworld environment provides a perfect textual description of the current location, including intrinsic attributes of observed objects and receptacles, such as whether or not a given receptacle can be opened.
3. **Causal Environment**: changes in the environment are entirely caused by the agent.
4. ***Valid* actions always succeed**

Generating a goal state. LLM-DP uses an LLM to generate a `PDDL` goal, given the natural language instruction ($task$) and the valid predicates defined by the `PDDL` domain file. Figure 1 shows an example task converted to a valid `PDDL` goal. For each episode, we use a set of three in-context examples that are fixed for the entire evaluation duration. We use the OpenAI gpt-4o-mini-2024-07-18 LLM model with a temperature of 0.6 in all our LLM-DP experiments.

Sampling beliefs. We parse the scene description into a structured representation $\mathcal{W}$ and a set of beliefs $\mathcal{B}$. The world $\mathcal{W}$ contains all *known* information, such as receptacles and their attributes (e.g., `isFridge`). In contrast, $\mathcal{B}$ consists of predicates that may be true or false. Since object locations are unknown in Alfworld, the possible predicates for each object include all potential locations. LLM-DP uses observations ($\mathcal{W}$), beliefs ($\mathcal{B}$), and an LLM to generate planning problem files in `PDDL` . These files define objects (`:objects`), the world state (`:init`), and goals (`:goal`).

The LLM derives the goal, while $\mathcal{W}$ and $\mathcal{B}$ provide object attributes and beliefs. Because $\mathcal{B}$ contains unknowns, we sample from $\mathcal{B}$ using the LLM to obtain w_{belief}. For instance, `(inReceptacle tomato ?x)` may suggest several locations for `?x`. Sampling selects a value for `?x` by passing $\mathcal{W}$ and the predicate to the LLM. We compare LLM sampling with random sampling (`llmdp-random`). The likely world state is the union of sampled beliefs and known states, $w_{belief} \bigcup \mathcal{W}$. By sampling N belief sets, we obtain N likely world states, which are converted to predicates for the `PDDL` planner.

Plan Generator. Upon constructing the different `PDDL` problems, the agent uses a Plan Generator (PG) to solve each problem and obtain a plan. We use the BFS(f) solver (Lipovetzky et al., 2014) implemented as an executable by LAPKT (Ramirez et al., 2015). A generated plan is a sequence of actions, each represented in a symbolic form, which, if executed in the intial state, yields a goal state.

Action Selector. The Action Selector (AS) module decides the agent's immediate next action. It takes the planner's output, a set of plans, and selects an action from them. In our Alfworld experiments, the Action Selector simply selects the shortest plan returned. If no valid plans are returned, then all sampled states satisfy goal states, or there is a mistake with the constructed domain/problem files, or the planner has failed to find a path to the goal. In the first case, we re-sample random world states and re-run the planners once. We also propose exploring different strategies when valid plans cannot be found. For instance, similarly to self-reflection (Shinn et al., 2023), the Action Selector could prompt an update in the agent's belief about the world state if none of generated problem descriptions are solvable. The Action Selector could also interact with a human teacher or oracle to adjust its understanding of the environment (problem) or its logic (domain).

Observation Processing. LLM-DP uses the result of each action to update $\mathcal{W}$ and $\mathcal{B}$. It uses the symbolic effects of the action to infer changes in the state of the objects and receptacles. Then it integrates the information from the new observation, which might reveal additional details not directly inferred from the action itself: for instance, opening an unseen drawer might reveal new objects inside. If an object is observed at a location, it cannot be elsewhere; if it's not, then it cannot be there. These observations trigger updates to $\mathcal{W}$ and $\mathcal{B}$. If the agent detects new information from the scene, such as discovering new objects, it triggers a re-planning process. The agent then generates a new set of possible `PDDL` problems using the updated state representation and corresponding plans using the Plan Generator. This approach is similar to some Task and Motion Planning (TAMP) methods (Garrett et al., 2018; Chen et al., 2023), enabling the agent to adapt to environmental changes and unexpected outcomes of actions.

5 Results

We contrast the LLM-DP approach with ReAct (LLM-only baseline) from the original implementation by Yao et al. (2023). Since LLM-DP uses a chat model rather than the original text-davinci-002, we also reproduce ReAct's results using gpt-4o-mini and adapt its prompts to a chat format. The ReAct baseline makes different assumptions about the problem: it doesn't require a domain file containing the action-descriptions and predicates, but instead uses two separate human-annotated episodes per example to bootstrap its in-context logic. In ReAct, we select the two few-shot examples based on the type of task being solved.

As shown in Table 1a, LLM-DP solves Alfworld almost perfectly (97%), in contrast to the baselines. Errors occur when sampling, for instance, picks states where the goal is already satisfied. Our reproduction of ReAct obtains similar results to the original, doing worse on some tasks (e.g. examine) and better on others (e.g. puttwo). We also measure the length of each successful episode (Table 1b) and find that LLM-DP reaches the goal state faster on average than ReAct and a random search strategy.

6 Conclusion

The LLM-DP agent integrates language understanding, symbolic planning and state tracking. It offers a trade-off between a wholly symbolic solution and an LLM-only model: the LLM's semantic knowledge is leveraged to translate the natural language problem into `PDDL` and to support belief sampling. Our experiments show that LLM-DP can handle complex tasks in Alfworld, making it a promising approach for embodied tasks that involve language understanding, reasoning and decision-making. It was not only cheaper, on a per-token comparison, but also faster and more successful at long-term planning than an LLM-only baseline.

These initial results, while promising, raise numerous topics that remain open. Key among these is devising strategies to encode the world model and belief, currently handled symbolically, and managing uncertain observations—particularly from an image model—along with propagating any uncertainty to the planner and Action Selector. Future work may also explore more sophisticated Action Selector strategies that encourage self-reflection: for instance, if all plans prove invalid, it might indicate an incorrect domain definition. Such instances may necessitate interactions with an instructor, who provides insights about the domain. Indeed, such interactions could lead to changes in the domain file, making the agent truly adaptable to new environments.

References

Yongchao Chen, Jacob Arkin, Yang Zhang, Nicholas A. Roy, and Chuchu Fan. 2023. Autotamp: Autoregressive task and motion planning with llms as translators and checkers. *ArXiv*, abs/2306.06531.

Richard E. Fikes and Nils J. Nilsson. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3):189–208.

Caelan Reed Garrett, Tomas Lozano-Perez, and Leslie Pack Kaelbling. 2018. Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *International Conference on Automated Planning and Scheduling*.

Shibo Hao, Yilan Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. *ArXiv*, abs/2305.14992.

Jörg Hoffmann and Bernhard Nebel. 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302.

Chenxu Hu, Jie Fu, Chenzhuang Du, Simian Luo, Junbo Jake Zhao, and Hang Zhao. 2023. Chatdb: Augmenting llms with databases as their symbolic memory. *ArXiv*, abs/2306.03901.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Wenliang Dai, Andrea Madotto, and Pascale Fung. 2022. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55:1 – 38.

Nir Lipovetzky, Miquel Ramirez, Christian Muise, and Hector Geffner. 2014. Width and inference based planners: Siw, bfs (f), and probe. *Proceedings of the 8th International Planning Competition (IPC-2014)*, page 43.

B. Liu, Yuqian Jiang, Xiaohan Zhang, Qian Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. 2023. Llm+p: Empowering large language models with optimal planning proficiency. *ArXiv*, abs/2304.11477.

Drew McDermott. 2000. The 1998 ai planning systems competition. *AI Magazine*, 21(2):35–55.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*. Computation and Language (cs.CL); Artificial Intelligence (cs.AI).

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. 2023. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*.

Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Lidén, Zhou Yu, Weizhu Chen, and Jianfeng Gao. 2023. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *ArXiv*, abs/2302.12813.

Miquel Ramirez, Nir Lipovetzky, and Christian Muise. 2015. Lightweight Automated Planning ToolKiT. `http://lapkt.org/`. Accessed: 2020.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *ArXiv*, abs/2302.04761.

Noah Shinn, Beck Labash, and Ashwin Gopinath. 2023. Reflexion: An autonomous agent with dynamic memory and self-reflection. *arXiv preprint arXiv:2303.11366*.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. 2020. Alfworld: Aligning text and embodied environments for interactive learning. *CoRR*, abs/2010.03768.

Significant-Gravitas. 2023. An experimental open-source attempt to make gpt-4 fully autonomous. `https://github.com/significant-gravitas/auto-gpt`. Accessed: 2023-06-09.

Tom Silver, Varun Hariprasad, Reece S Shuttleworth, Nishanth Kumar, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. 2022. Pddl planning with pretrained large language models. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Dídac Surís, Sachit Menon, and Carl Vondrick. 2023. Vipergpt: Visual inference via python execution for reasoning. *ArXiv*, abs/2303.08128.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi (Jim) Fan, and Anima Anandkumar. 2023a. Voyager: An open-ended embodied agent with large language models. *ArXiv*, abs/2305.16291.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Huai hsin Chi, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*.

Zhun Yang, Adam Ishay, and Joohyung Lee. 2023. Coupling large language models with logic programming for robust and general reasoning from text. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 5186–5219. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

A Pseudo-code

We describe the LLM-DP algorithm in Algorithm 1.

Algorithm 1 LLM-DP Pseudo-code

Require: LLM, PG, AS, Domain, $task$, obs_0
  $goal \leftarrow$ LLM(Domain, $task$)
  $\mathcal{W}, \mathcal{B} \leftarrow$observe($goal, obs_0$)
  while $goal$ not reached **do**
    $plans \leftarrow \emptyset$
    for i in N **do**
      $w_{belief} \leftarrow$LLM($\mathcal{B}, \mathcal{W}$)
      $plans \leftarrow$PG($w_{belief} \bigcup \mathcal{W}$)
    end for
    $action \leftarrow$AS($plans$)
    $obs \leftarrow$Env($action$)
    $\mathcal{W}, \mathcal{B} \leftarrow$observe($action, obs$)
  end while

B Prompts and Few-shot details

See Table 2 and Table 3 for LLM-DP prompts used.

C ReAct

C.1 Reproduction with Chat Model

We slightly modify the ‘system’ prompt of the original ReAct (see Table 4) to guide the model away from its conversational tendencies. gpt-4o-mini apologises significantly more than the text-davinci-002 model, and we found that it would often get stuck in loops of apologising. We also modify the code so that we replace all generated instances of ‘in’ and ‘on’ with ‘in/on’ if the model did not generate it correctly, since Alfworld expects ‘in/on’ but gpt-4o-mini tends to generate only the correct preposition. Without these changes, ReAct would be significantly worse than our reported metric.

```
(define (domain alfred)
(:predicates
    (isReceptacle ?o - object) ; true if the object is a receptacle
    (atReceptacleLocation ?r - object) ; true if the robot is at the receptacle location
    (inReceptacle ?o - object ?r - object) ; true if object ?o is in receptacle ?r
    (openable ?r - object) ; true if a receptacle is openable
    (opened ?r - object) ; true if a receptacle is opened
    (isLight ?o - object) ; true if an object is light source
    (examined ?o - object ?l - object) ; whether the object has been looked at with light
    (holds ?o - object) ; object ?o is held by robot
    (isClean ?o - object) ; true if the object has been cleaned in sink
    (isHot ?o - object) ; true if the object has been heated up
    (isCool ?o - object) ; true if the object has been cooled
    (isSink ?o - object) ; true if the object is a sink
    (isMicrowave ?o - object) ; true if the object is a microwave
    (isFridge ?o - object) ; true if the object is a fridge
))
```

Table 2: System Prompt used by gpt-4o-mini for generating the `:goal` in LLM-DP

```
Your task is to: put a clean plate in microwave.
(:goal
(exists (?t - plate ?r - microwave)
(and (inReceptacle ?t ?r)
(isClean ?t)
)))

Your task is to: examine an alarmclock with the desklamp",
(:goal
(exists (?t - alarmclock ?l - desklamp)
(and (examined ?t ?l) (holds ?t)
)))

Your task is to: put two cellphone in bed
(:goal
(exists (?t1 - cellphone ?t2 - cellphone ?r - bed)
(and (inReceptacle ?t1 ?r)
(inReceptacle ?t2 ?r)
(not (= ?t1 ?t2))
)))
```

Table 3: Fixed Few-shot examples used by gpt-4o-mini for generating the `:goal` in LLM-DP

D LLM-DP

D.1 Generated Goal Examples

See Table 5 for examples of generated goals, both valid and invalid.

D.2 Varying the number of plans n

See Table 6 for results when different varying n and fallback. Fallback is when no plans are sampled successfully through the LLM, LLM-DP re-samples n plans randomly.

```
Interact with a household to solve a task.
Only reply with > followed by the action to take or 'think'.
Do not apologize.
Follow the format of the two examples below.
```

Table 4: System Prompt used by gpt-4o-mini in our reproduction of ReAct

task: put some peppershaker on drawer.
Generated:
(:goal
    (exists (?t - peppershaker ?r - drawer)
        (inReceptacle ?t ?r)
))
 	
VALID ✓
	task: put a clean mug in coffeemachine.
	Generated:
	(:goal
	    (exists (?t - mug ?r - coffeemachine)
	        (and (inReceptacle ?t ?r)
	            (isClean ?t)
	)))
	
	VALID ✓

task: put two cd in safe.
Generated:
(:goal
    (exists (?t1 - cd ?t2 - cd ?r - safe)
        (and (inReceptacle ?t1 ?r)
            (inReceptacle ?t2 ?r)
            (not (= ?t1 ?t2))
)))
VALID ✓
	task: heat some mug and put it in coffeemachine.
	Generated:
	(:goal
	    (exists (?m - mug ?c - coffeemachine)
	        (and (isReceptacle ?m)
	            (isHot ?m)
	            (inReceptacle ?m ?c)
	)))
	INVALID ✗

Table 5: Sample of generated PDDL goals from LLM-DP. The generation gets confused by the semantics of 'receptacle' and identifies a mug as a receptacle. While it is true that a mug is a receptacle, in our defined logic, receptacles are fixed, immovable objects which can contain other objects and therefore, a mug is not a Receptacle which leads the planning to fail subsequently.

	SR	EL
LLM-DP (n=3)	0.97	12.53
LLM-DP (n=3) - fallback	0.82	11.34
LLM-DP (n=5)	0.97	12.49
LLM-DP (n=5) - fallback	0.84	11.08

Table 6: We compare the average Success Rate (SR) and average Episode Length (EL) for different sampling sizes n and with or without a fallback to random sampling. The random sampling fallback affects the success rate as the LLM sampler can more often sample n world states which are already satisfied. However, as n increases, it becomes more likely for the sampling procedure to at find at least one plan, and therefore the SR increases when no fallback (- fallback) is used. We also note, that while the success rate without fallback is lower, the paths found to the goal tend to be shorter.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, all the claims in the abstract and introduction are supported by our results.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: While we touch upon some of the limitations of in the Conclusion, we do not include a Limitations section due to lack of space. The main limitation of LLM-DP is that it still requires a formal definition of actions and a knowledge the structure of the world and the possible values that it can take.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: Our work is not theoretical in nature.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We note all major hyper-parameters as well as release the code used to obtain our results. Even without the code, we believe to have provided enough detail to assist future replication.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The Alfworld dataset is open-source, and our code is released under the MIT license. Our code release includes a README and scripts to run all of the mentioned configurations in our paper.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify all details where possible. Since our method is zero-shot, we note all prompts used in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We run all configurations over five different seeds and report the standard deviation on the overall results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Because we argue that LLM-DP is a more efficient planner than a ReAct baseline, we report the number of tokens used as a main result in Table 1. The number of tokens can easily be extrapolated in terms of cost.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research conforms with the above Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: We do not specifically detail the positive nor negative societal impacts of our work. Our work provides a neuro-symbolic method to improve LLM-based planning. We do not envisage our work to lead to negative societal impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: We do not release any models or data, as we only make use of OpenAI models called through its APIs, and the Alfworld open-source dataset.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We appropriately credit the dataset creators Shridhar et al. (2020), previous work Yao et al. (2023) and the model owners OpenAI (2023).

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We release our code along with README instructions on how to run it.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We do not involve human participants.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: We do not involve human participants.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.